

Oracle® Retail Supplier Evaluation Cloud Service Implementation Guide



G36982-01
July 2025



Oracle Retail Supplier Evaluation Cloud Service Implementation Guide,

G36982-01

Copyright © 2025, Oracle and/or its affiliates.

Primary Author: Bernadette Goodman

Contributing Authors: Aidan Ratcliffe, Simon Tucker

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Send Us Your Comments

Preface

Audience	ix
Documentation Accessibility	ix
Related Documents	ix
Improved Process for Oracle Retail Documentation Corrections	x
Oracle Retail Documentation on the Oracle Help Center (docs.oracle.com)	x
Conventions	x

1 Introduction

Contents of this Guide	1-1
Key Features of Supplier Evaluation Cloud Service	1-2

2 API Overview and Architecture

SOAP and RESTful APIs	2-1
Request and Response XML	2-1
Versioning	2-2
Logging	2-3
Accessing the APIs	2-3
Availability	2-3
Deployment Model	2-3
Authentication	2-3
Authorization	2-7
Listing Available APIs	2-7
REST API Path	2-9
Change History	2-9
Security Considerations	2-10
Password Management	2-10
Rich Text Data	2-10
Excluded Fields	2-10

Secured Connections	2-11
URL Encoding	2-11
Reference Fields	2-11
Glossary and Locale/Language Fields	2-11
Enumeration Fields	2-12
Editable Text Selector Fields	2-12
Key Features of the API Process	2-12

3 APIs Available in Supplier Evaluation

Individual Services	3-1
Services Grouped by Area	3-5
User and Supplier APIs	3-5
Audits API	3-6
Artwork Project Activities	3-6
Data Privacy API	3-6
Product APIs	3-7
Project/Process and Activity APIs	3-7
Miscellaneous	3-7
Dependencies	3-7
Linking to Related Records	3-11

4 RESTful APIs

Parameters and Filtering	4-2
Error Messages	4-4
Wildcard Searches	4-5
Deleted Records	4-5
UserRestService	4-6
List of Values	4-6
Retrieve Record by ID	4-7
Retrieve Record by Business Key	4-8
Check Record Modification Timestamp	4-9
Create Record	4-10
Update Record	4-14
SupplierRestService	4-15
List of Values	4-16
Retrieve Record by ID	4-18
Retrieve Record by Business Key	4-19
Check Record Modification Timestamp	4-20
Create Record	4-20
Update Record	4-24

SiteRestService	4-25
List of Values	4-26
Retrieve Record by ID	4-29
Retrieve Record by Business Key	4-30
Check Record Modification Timestamp	4-31
Create Record	4-32
Update Record	4-34
ContactRestService	4-35
List of Values	4-36
Retrieve Record by ID	4-38
Retrieve Supplier Contact Record by Business Key	4-39
Retrieve Site Contact Record by Business Key	4-40
Check Record Modification Timestamp	4-40
Create Record	4-41
Update Record	4-44
TaskRestService	4-45
List of Tasks	4-45
UrgentItemsRestService	4-46
Number of Urgent Items	4-46
ArtworkRestService	4-47
Started Activities	4-47
Update Record	4-49
BusinessCategoryService	4-50
List of Values	4-51
Retrieve Record by ID	4-52
Check Record Modification Timestamp	4-53
Create Record	4-54
Update Record	4-57
Delete Record	4-59
AuditRestService	4-59
List of Values (Audit)	4-60
Retrieve Record by ID (Audit)	4-61
Retrieve Record by Business Key (Audit)	4-62
Check Record Modification Timestamp (Audit)	4-63
Create Record (Audit)	4-64
Update Record (Audit)	4-68
List of Values (Checklist)	4-70
Retrieve Record by ID (Checklist)	4-71
Retrieve Record by Business Key (Checklist)	4-72
Create Record (Checklist)	4-73
Update Record (Checklist)	4-80
Delete Record (Checklist)	4-81

DataPrivacyService	4-81
Right to Access	4-82
Right to be Forgotten	4-83
AttachmentRestService	4-87
Retrieve Report Attachment	4-89
Retrieve Attachment	4-89
ScorecardRestService	4-90
List of Values	4-90
Retrieve Record by ID	4-92
Retrieve Record by Business Key	4-93
Check Record Modification Timestamp	4-94
Update Record	4-94
Update Sub Status	4-100
ProjectRestService	4-102
List of Values	4-102
Retrieve Record by ID	4-104
Retrieve Record by Business Key	4-105
Check Record Modification Timestamp	4-106
Update Record	4-107
ActivityRestService	4-110
List of Values	4-110
Retrieve Record by ID	4-112
Retrieve Record by Business Key	4-113
Check Record Modification Timestamp	4-114
Update Record	4-114
Update Status	4-116
ProductRecordRestService	4-117
List of Values	4-118
Retrieve Record by ID	4-119
Retrieve Record by Business Key	4-120
Check Record Modification Timestamp	4-121
Create Record	4-122
Update Record	4-126

5 SOAP APIs

Parameters and Filtering	5-1
Project Activities	5-2
Activity Management	5-3

A Appendix: Secure Development Guide

Best Practices for Web Service Use	A-1
Store External System Credentials Securely	A-1
Configure External Systems Permissions Appropriately	A-1
Maintain Strong Passwords for External System Records	A-1
Make Use of the Web Service Log	A-2
Ensure Your Client Supports TLS v1.2 or Higher	A-2
Limit the Size of Web Service Payloads	A-2

B Appendix: Using the Data Privacy API

Definition of Personal Data	B-1
Personal Data Flow	B-2
Right to Access Requests	B-3
Personal Data Searched and Returned	B-3
Security and Logging	B-4
Example of Using the Right to Access Service	B-5
Right to be Forgotten Requests	B-7
Personal Data Updated	B-7
Exceptions	B-8
Purging Inactive User Accounts	B-9
Security and Logging	B-9
Example of Using the Right to be Forgotten Service	B-9
Other Aspects of Data Privacy	B-13

C Appendix: Using the Report Attachment API

D Appendix: Accessibility

User Interface	D-1
----------------	-----

E Appendix: LCA Third-Party Integration

Accredited Bodies	E-1
Assessments Integration	E-2
Using the Scorecard API	E-3

Glossary

Send Us Your Comments

Oracle Retail Supplier Evaluation Cloud Service Implementation Guide

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).



Note:

Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Help Center (docs.oracle.com) web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our web site at <https://www.oracle.com>.

Preface

This Implementation Guide describes the Application Programming Interfaces (APIs) available for this Oracle Retail Supplier Evaluation Cloud Service release.

Audience

This Implementation Guide is intended for the users of the Oracle Retail Supplier Evaluation Cloud Service application integration and implementation staff, and other parties intending to integrate with Supplier Evaluation Cloud Service.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Retail Supplier Evaluation Cloud Service documentation set:

- *Oracle Retail Supplier Evaluation Cloud Service Administration Guide*
- *Oracle Retail Supplier Evaluation Cloud Service Release Readiness Guide*
- *Oracle Retail Supplier Evaluation Cloud Service Security Guide*
- *Oracle Retail Supplier Evaluation Cloud Service User Guide*
- *Oracle Retail Supplier Evaluation Cloud Service Workspace User Guide*

For information on the Oracle Retail Supplier Evaluation Cloud Service modules, see the following documents:

- *Oracle Retail Supplier Evaluation Cloud Service Process User Guide*
- *Oracle Retail Supplier Evaluation Cloud Service Product User Guide*
- *Oracle Retail Supplier Evaluation Cloud Service Reports User Guide*
- *Oracle Retail Supplier Evaluation Cloud Service Supplier User Guide*

Data Dictionary

For information on the content of the Oracle Retail Supplier Evaluation Cloud Service data records, see the associated Data Dictionary documents:

- *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 1 - Framework*
- *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier*
- *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process*
- *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 4 - Reports*
- *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 5 - Product*

For the Oracle Retail Data Store, see the *Oracle Retail Brand Compliance Management and Supplier Evaluation Cloud Services Retail Data Store (RDS) Data Model*.

The Data Dictionary and RDS Data Model documents are available through My Oracle Support.

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times **not** be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Help Center (docs.oracle.com) Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

Oracle Retail documentation is available on the Oracle Help Center (docs.oracle.com) at the following URL:

<https://docs.oracle.com/en/industries/retail/index.html>

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of an document with part number E123456-01.

If a more recent version of the document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Help Center (docs.oracle.com)

Oracle Retail product documentation is available on the following web site:

<https://docs.oracle.com/en/industries/retail/index.html>

(Data Model documents can be obtained through My Oracle Support.)

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
<code>monospace</code>	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Introduction

Oracle Retail Supplier Evaluation Cloud Service is a collaborative cloud service for the onboarding and evaluation of merchandising suppliers, enabling the assessment and governance of ethical, environmental, safety, and quality performance. It manages the selection of suppliers against Environmental, Social, and Governance (ESG), brand standards and governance policies, incorporating supplier self-certification survey and assessment, audit and action management, vendor performance, and incident alert notifications.

Supplier Evaluation includes a number of exposed Application Programming Interfaces (APIs) that can be called by external systems. This document details all of the exposed APIs and the information required in order to access them.

An API is a way of exchanging data between computer systems. The owners of an external system can interact with Supplier Evaluation by developing the necessary functionality to trigger the web service submissions, and handle the returned messages from Supplier Evaluation.

In order to access Supplier Evaluation using its APIs, the portal owner must grant the external system access. This is achieved by creating an External Systems account in the Admin area and assigning the unique access credentials in the form of a login ID and password.



Note:

The terminology used for Scorecard and Project has changed. Scorecard has been renamed to Assessment. Project has been renamed to Process. The associated API names and endpoints/functions are not changed. For Assessments, see the ScorecardRestAPI. For Processes, see the ProjectRestAPI.

Contents of this Guide

This implementation guide addresses the following topics:

- [API Overview and Architecture](#): Overview of the Supplier Evaluation APIs and architecture used.
- [APIs Available in Supplier Evaluation](#): Summary of the APIs, including the release in which the API is available and the type of API.
- [RESTful APIs](#): Details of the available APIs that are based on the RESTful architecture.
- [SOAP APIs](#): Details of the available APIs that are based on the SOAP architecture.
- [Appendix: Secure Development Guide](#): Description of some best practices for web service use.
- [Appendix: Using the Data Privacy API](#): Additional detail on using the Data Privacy API, including examples.
- [Appendix: Using the Report Attachment API](#): Details on using the Report Attachment API.

Key Features of Supplier Evaluation Cloud Service

The application is composed of the following modules:

- Library enables the issue, receipt, and acceptance of policies, guidelines, and key working documents.
- Process supports the development of process briefs, plans, and workflow management.
- Product supports the development of products.
- Reports provides a reporting tool for reporting across the system, using standard templates and custom reports.
- Supplier enables the identification, selection, and approval of suppliers.

2

API Overview and Architecture

The APIs exposed by Supplier Evaluation are implemented in one of two formats, SOAP web services or RESTful web services. In both cases, requests and responses are normally in an XML format.

The APIs are already built and available for clients Supplier Evaluation to build services for their external applications to utilize. The external system makes a call to Supplier Evaluation using HTTPS. If the external system is validated (that is, it is enabled in the Admin area of Supplier Evaluation and the login user ID and password match), the call is accepted.

Calls will typically allow for the retrieval, creation, update, and in some cases deletion, of data through the API. Each call is logged in the Web Service Log in Supplier Evaluation, with the request and response messages being recorded as appropriate.

A call may contain a number of parameters or filters to determine the records to be returned or updated, or to identify where to create a new record, such as to filter Audit of a certain type, status and so on.

SOAP and RESTful APIs

The Supplier Evaluation APIs are web services in both SOAP and RESTful architectural styles. In general, the SOAP services are developed for specific requirements or are historical services. Unless there is a definite requirement for a SOAP implementation of a service, any new services or new versions of services are implemented using the RESTful architectural style.

Request and Response XML

In general, APIs exposed by Supplier Evaluation as web services use XML messages for both requests and responses. Exceptions to this are RESTful services where the use of an XML request is replaced by the use of URI parameters in a GET call. The APIs do not support other message formats, such as JSON.

The XML data schemas are written to allow for both forwards and backwards compatibility. All elements within the XML schemas are marked as optional. The element names are ordered so that new fields are added after existing elements and, within each sequence of elements, the last element is always an *any* element so that fields added in future versions are ignored by clients that are using an old version of the schema. For example:

```
<xs:complexType name="fpsMicroTestLocaleDataFullDTO">
  <xs:sequence>
    <xs:element form="qualified" minOccurs="0" name="description" type="xs:string"/>
    <xs:element form="qualified" minOccurs="0" name="locale" type="xs:string"/>
    <xs:element form="qualified" minOccurs="0" name="id" type="xs:long"/>
    <xs:element form="qualified" minOccurs="0" name="createdOn" type="xs:string"/>
    <xs:element form="qualified" minOccurs="0" name="updatedOn" type="xs:string"/>
    <xs:any maxOccurs="unbounded" minOccurs="0" namespace="##other"
processContents="lax"/>
  </xs:sequence>
</xs:complexType>
```

The element names in the XML Schemas are defined by the core system and do not change in specific implementations. If an implementation has additional fields that have been added to the system (the implementation is non-core), those fields may be exposed after the core fields (before the `xs:any` element) and will incur a maintenance overhead in later versions to cope with any additional core fields added after the custom fields.

Data types in the XML Schemas are the standard datatypes for XSDs, WSDLs, and WADLs. Values should be entered as described on the following web site:

<http://www.w3.org/TR/xmlschema-2/>

The following table lists the common data types used in the Supplier Evaluation schemas:

Data Types

Type	Example
<code>xs:string</code>	A normal string
<code>xs:string</code>	<code><![CDATA[A String with
 html data]]></code>
<code>xs:long</code>	9223372036854775807
<code>xs:int</code>	2147483647
<code>xs:short</code>	-32768
<code>xs:boolean</code>	true or false
<code>xs:boolean</code>	1 or 0
<code>xs:date</code>	2015-01-30
<code>xs:dateTime</code>	2015-01-30T23:59:59

Date Time Parsing

For date time strings in inbound data, the preferred format is `YYYY-MM-DDThh:mm:ss.fffZ`, however all the following formats are permitted:

```

YYYY-MM-DD hh:mm:ss
YYYY-MM-DDThh:mm:ss
YYYY-MM-DDThh:mm:ss.fff
YYYY-MM-DDThh:mm:ss.fffZ

```

The format of date time strings in outbound data is `YYYY-MM-DDThh:mm:ss.fffZ`.

Versioning

Due to the format of the XML schemas as described above, most revisions of the web services should be forwards and backwards compatible to allow clients of those web services to be upgraded and developed without binding the lifecycle of the client services to the upgrade cycle of Supplier Evaluation.

Where the necessary changes to a service cannot be made compatible with previous versions, a new endpoint will be created to expose the changed service. If possible, the previous version of the service will remain available at the original endpoint to enable existing and legacy clients to continue to work, but will be noted as deprecated in documentation to encourage developers of clients to the services to avoid using it.

Logging

All calls to APIs exposed as web services are logged within the application and can be viewed by logging into Supplier Evaluation as a Power User, navigating to Company > Admin> Notifications, and choosing the Web Service Log option. For each call, the name of the web service, date and time of the call, status (in progress/completed/failed), duration of the call (not available for In Progress calls), and a list of error messages if the call failed are shown.

The request and response XMLs are attached to each web service log entry. If a request does not contain XML, no attachment is created.

Accessing the APIs

This section describes how to access the APIs.

Availability

If enabled within the portal, an API is accessible while the portal is online. The support for a portal's availability is subject to the client's service level agreement.

Updates to the APIs are applied as per changes and fixes to the application. Each release involves a scheduled period of downtime. Details of the change and impact are covered in the supporting Release Notes.

**Note:**

After a Supplier Evaluation upgrade release, it may be necessary to change the URLs that the external system uses to call the APIs.

Deployment Model

The APIs, made available by Supplier Evaluation, are all web services implemented using either a SOAP or RESTful architecture.

The web services are exposed over HTTPS and require OAuth 2.0 authentication in order to access the service requested.

Authentication

OAuth 2.0 is the industry standard protocol for authorization. Supplier Evaluation REST Services now support OAuth 2.0. To invoke these services, you need to obtain an access token and use it as a bearer token.

**Note:**

Basic Authentication access is still available for these services, but this has now been deprecated. See the deprecation notice on My Oracle Support at [Doc ID 2921721.1](#).

Steps to Enable

To obtain a token and call the services, use the following steps:

1. Set up an External System record within Supplier Evaluation and allocate only the required Service Access and/or End Point Access that this External System requires.

To see the list of existing External System records, log in to Supplier Evaluation as a Power User, navigate to Company > Admin > Roles and Permissions, and select the External Systems option:

- a. To create a new External System: Select the Actions > New action to open up a tab for creating an External System.
- b. Enter a Login ID (User Name), Email Address, Comment (optional), and select the appropriate Service Access and/or End Point Access that is required, then select the Actions > Save & Exit option.
- c. An Oracle Identity Cloud Service (IDCS) or Oracle Cloud Infrastructure Identity and Access Management (OCI IAM) profile will be automatically created and a notification email sent to the specified email address, containing instructions for setting the password.

The passwords used for External Systems are validated against the same criteria as the passwords for users of Supplier Evaluation. It is likely that a minimum length, mixed case, and digits will be required for the password, depending on the setup of the IDCS or OCI IAM tenant.

2. These steps should be performed one time for each External System record created and requiring access to Supplier Evaluation through OAuth 2.0. This can be done directly from your Retail Home instance.

Perform these steps using the Retail Home feature:

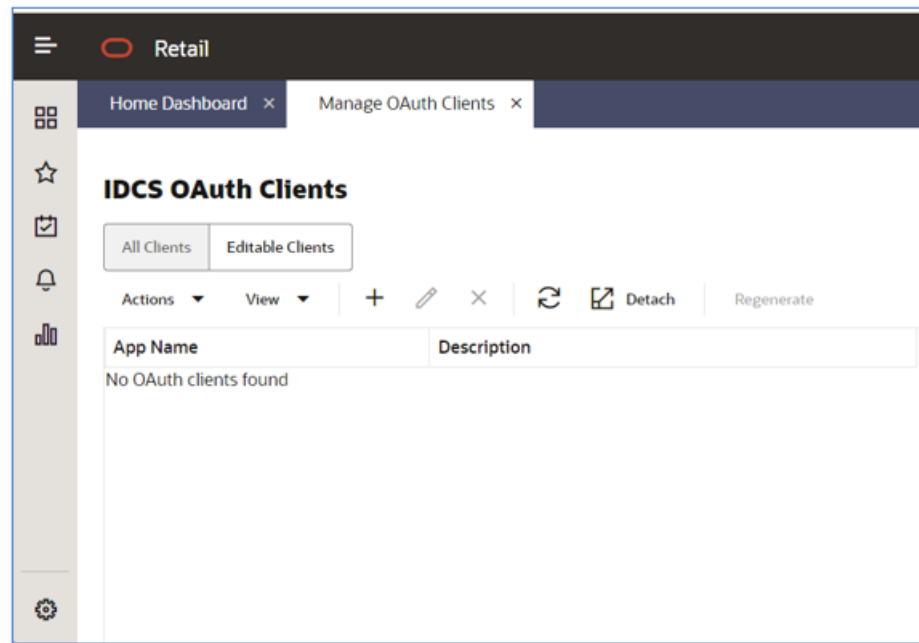
- a. Access the Retail Home associated with the cloud service as a user with Administration rights to Retail Home.

See the OAuth Application Configuration section in the *Oracle Retail Home Administration Guide*. Retail Home documentation can be accessed from the OHC Retail Documentation page:

<https://docs.oracle.com/en/industries/retail/index.html>

- b. From the Retail Home Dashboard, choose the Settings option in the bottom left of the screen. This opens a menu. Select the Application Administration option and then the Manage OAuth Clients option.

The IDCS OAuth Clients page is displayed.



- c. Choose the ADD action to create a new OAuth Client. A dialog window is displayed ready for completion.

App Name: This should be populated with the External System account name that was specified within Supplier Evaluation.

Description: A description for the use of the OAuth client.

Scope 1: For the purposes of Supplier Evaluation, this should be populated with the following:

- Stage environments: rgbu:secs:STG1
- Production environments: rgbu:secs:PRD1

- ANPE environments: rgbu:secs:DEV1
- d. Complete the fields and click **OK**. The fields will be validated, and if all the entries are valid, a dialog window is shown displaying the Client ID and Client Secret for the created OAuth Client.

New IDCS OAuth 2.0 Client

Display Name: ORACLE_OAUTH_EXAMPLE

Client ID: [Redacted] [Copy]

Client Secret: [Redacted] [Copy]

Done

- e. Make note of the Client ID and Secret for use in the following steps for using the OAuth client that has been created.

Note: The Client ID and Secret can be retrieved from the IDCS admin console at any time.

3. Once the setup steps are completed, accessing an API requires that a Bearer token is requested from IDCS, and then this token is used on subsequent calls to Supplier Evaluation rather than sending a Username/Password for each request per Basic Authentication. To obtain the Bearer token, call IDCS with the appropriate details.

This example uses CURL to obtain the Bearer token:

- a. To obtain a Bearer token, an initial call to IDCS is required. Several values will need to be supplied all of which have been mentioned above in the setup of the Application record.

{SCOPE} = rgbu:secs:STG1 / rgbu:secs:PRD1 **Note:** The Scope should be URL encoded to be this format... rgbu%3Asecs%3ASTG1 / rgbu%3Asecs%3APRD1

{IDCS_CLIENTID} = The IDCS client id that was provided during the Application setup in IDCS.

{IDCS_SECRET} = The IDCS secret that was provided during the Application setup in IDCS.

{IDCS_URL} = The IDCS URL for the tenant being accessed, including the "https://".

```
curl -H "Content-Type: application/x-www-form-urlencoded;charset=UTF-8" -d
"grant_type=client_credentials&scope={SCOPE}" --user {IDCS_CLIENTID}:
{IDCS_SECRET} {IDCS_URL}/oauth2/v1/token
```

Note: When substituting the values, ensure that the {} are also replaced.

- b. Once the CURL command has been constructed, this can be executed through a Command Prompt interface. This will then return a Bearer token for use when calling the actual API.

```
{"access_token":"{token}","token_type":"Bearer","expires_in":3600}
```

4. Once the Bearer token is available, this can then be used to call the appropriate API. This example uses CURL to call the User API.

To call the appropriate API, several values will need to be supplied.

{APPLICATION_URL} = The Supplier Evaluation url for the application instance.

{API_NAME} = The Service being executed, and any parameters if required.

```
curl --url "{APPLICATION_URL}/secs/services/rest/{API_NAME}" --header  
"Authorization: Bearer {token}"
```

 Note:

IDCS or OCI IAM is used to authenticate access to the services. Calling a service using a web browser does not generate the necessary header for IDCS or OCI IAM authentication, thus resulting in a 401 or 407 authorization error. Instead, a web service tool should be used.

A web service tool such as PostMan, SoapUI, or a lower-level tool such as curl or wget must be used instead.

Authorization

Authorization is controlled by providing the Service and Endpoint access that is specified on the External System record. If no Service or Endpoint access is specified, then no access will be granted to any API.

These two Service and Endpoint access fields allow the External System to be restricted to specific APIs and to specific access methods on the APIs (such as READ, WRITE, and UPDATE).

When creating an External System, attention should be given to what the purpose is for the usage of the External System and only provide the necessary access to information that is required.

Listing Available APIs

In all installations of Supplier Evaluation, a full list of the deployed web services can be found at /services.

For example, for a system where the application is deployed to URL <https://www.example.com/supplierEvaluation>, you can retrieve a list of all services from <https://www.example.com/supplierEvaluation/services>.

 Note:

In order to access the /services page, the user must first be logged in/authenticated with the Supplier Evaluation portal.

To download the REST API WADL files:

1. Run the Windows *Command Prompt* app.
2. Retrieve a Bearer token. See the [Authorization](#) section.
3. Enter the command

```
curl [url]/services/rest/[api]?_wadl=true --header "Authorization: Bearer [token]" > [api].wadl
```

Where [url] is the portal address; [api] is the name of the REST API; [token] is the Bearer token retrieved. For example:

```
curl https://<hostname>/services/rest/user?_wadl=true --header "Authorization: Bearer [token]" > user.wadl
```

The file will be downloaded to your default directory. To download to a specific directory, provide the full path for the .wadl file. For example:

```
... > c:\Users\JSmith\Downloads\APIs\23.0.101.0\user.wadl
```

The file is saved to the specified location.

4. Import the WADL file into SoapUI.

 Note:

If using SoapUI and the HTTP 401 error is received, click the Auth tab on the request in SoapUI and add OAuth 2.0 authorization. Enter the appropriate details to request a Bearer token.

To download the WSDL files for SOAP APIs:

1. Run the Windows *Command Prompt* app.
2. Enter the command

```
curl [url]/services/[api]?_wsdl --header "Authorization: Bearer [token]" > [api].wsdl
```

Where [url] is the portal address; [api] is the name of the SOAP API; [token] is the Bearer token retrieved. For example:

```
curl https://<hostname>/services/activityManagement?wsdl --header "Authorization: Bearer [token]" > activityManagement.wsdl
```

The file will be downloaded to your default directory. To download to a specific directory, provide the full path for the .wsdl file. For example:

```
... > c:\Users\JSmith\Downloads\APIs\23.0.101.0\activityManagement.wsdl
```

The file is saved to the specified location.

 Note:

If using SoapUI and the HTTP 401 error is received, click the Auth tab on the request in SoapUI and add OAuth 2.0 authorization. Enter the appropriate details to request a Bearer token.

REST API Path

A REST API Path system parameter allows for an alternative path to the REST APIs to be specified. This is of use where for example an alternative data center is being used, or where HTTP is being used instead of HTTPS.

By default, the path of the Base URL system parameter is used for the application and the APIs. The Rest API Path system parameter is only used if there is a need to override the base URL.

Change History

Change History Log

Each record contains a Change History log, which is maintained by the system as a result of the user creating or updating the record in the UI. The log is also maintained by the system if the record is created or updated using the API.

The log comprises:

- Changed On - the date and time the record was last updated.
- Changed By - the name and email address of the user who last updated the record (or 'System' if updated by the API).
- No. Changed Fields - the number of fields that were changed by the update.

Opening an entry shows the details of the individual field changes:

- Field label
- From (before) value
- To (after) value

The log is not included in the API's XML payload so it cannot be retrieved or updated by the API.

Status Change History Log

Records that have a workflow status also have a Status Change History log - a table that has a row added by the system each time an update to the record results in a status change.

The table comprises the columns:

- Status Before - the record's status before the update.
- Status After - the record's status after the update.
- Changed On - the date and time of the update.
- Changed By - the name of the user who made the update (or 'System' if updated by the API).
- Comments - any comments entered as part of the status change action.

The log is included in the API's XML payload for retrieval, within the `statusHistory` element:

```
<statusHistory>
  <comments>
  <localeData>
    <locale>
```

```
<statusFrom>
<statusTo>
<id>
<id>
<createdOn>
<updatedOn>
<statusChangedBy>
  <code>
  <disabled>
  <externalAuthenticationUser>
  <timeZone>
  <userType>
  <id>
```

For records with a Status Change History log:

- The APIs currently allow for rows to be added, removed, and updated in the table.
- The update process is locked down to only allow the Comments column to be updated.
- The Changed By column can be populated with a valid user's name with the `statusChangedBy` element.
- If no data is supplied for `statusChangedBy`, the Changed By field in the UI will default to 'System'. In this case, the underlying value is null, so when retrieving that data using the API, the `statusChangedBy` element will be omitted (this is per the standard rule for omitting elements that have no value).

Security Considerations

This section provides information on aspects of security you need to consider.

Password Management

Unlike interactive logins for users, the use of web services does not permit warnings to be included in the returned messages. Also, there is no guarantee that there is someone handling the call in order to interpret any warnings about password expiry.

Password expiry is managed within IDCS and is the responsibility of the owner to be aware of when passwords need to be rotated. With OAuth, the client id and secret should be kept secure.

Rich Text Data

In Supplier Evaluation, certain fields allow the display of HTML-formatted data. Such data is cleansed before display in Supplier Evaluation to ensure that any attempts at Cross-Site Scripting (XSS) attacks are mitigated, but this cleansing is not applied within the APIs. Therefore, it is advisable to cleanse any HTML-formatted data before it is displayed to users.

Excluded Fields

Many of the APIs published by Supplier Evaluation provide access to manage all of the records of a given type. With certain records, the fields accessible through the API are a subset of the fields within Supplier Evaluation. Any updates leave those fields unchanged.

Secured Connections

As all of the APIs are exposed as web services over HTTPS, it is necessary for callers to communicate over SSL/TLS using the ciphers that the Supplier Evaluation servers accept. To ensure optimal security, it is advised that standard certificate checks are in place and not disabled. In the case of Apache CXF, the Client TLS Parameter `disableCNCheck` should be left as false.

URL Encoding

Certain characters within URLs are reserved for a special meaning. In order to avoid calls that contain these characters being blocked by the API's security, the characters must be encoded and passed as plain text.

The reserved characters include ";", "/", "?", ":", "@", and "=", all of which could be present in the parameters passed to Supplier Evaluation APIs - such as within email addresses or within the names of users or suppliers.

UTF-8 URL encoding should therefore be applied to all calls to the Supplier Evaluation APIs. The APIs will automatically decode the values on receipt.

For example, an un-encoded call of: `.../services/rest/user/byKey/john.smith@example.com` should be encoded as: `.../services/rest/user/byKey/john.smith%40example.com`

The use of forward slash in parameters should be encoded as "%2f", for example an un-encoded call of: `.../services/rest/user/byKey/24/7 Support`

should be encoded as: `.../services/rest/user/byKey/24%2f7 Support`

If a call comprises multiple values, each must be encoded individually.

Reference Fields

This section describes how data is returned for fields that have their values derived by referencing a glossary or system text, such as the selections made from a configurable list of values.

Glossary and Locale/Language Fields

Fields that have their values selected from a glossary of configurable options generally have the code, description, and language translations of the selected value returned, along with the id reference to the glossary record.

The language translations are contained within a `localeData` element. Prior to release 18.0, a separate `localeData` element was returned for each language the portal had configured, now just the languages that the portal actually has active are returned. If a translation is not present for a language, the element is omitted, so if the portal has no alternative languages active, just a `localeData` element for the base language is returned.

**Note:**

Some APIs allow a locale parameter to be passed to optionally return a specific language, or languages. However, use of the parameter can have an impact on the call's response time, due to the additional selection processing.

Enumeration Fields

Certain reference fields have their options defined as system text rather than a configurable glossary. Referred to as enumerations, these are used where the selections are generally static, such as Yes/No options. Prior to release 18.0, just the code of an enumeration was returned, now the description of enumeration value is also returned as a description element.

Editable Text Selector Fields

Certain reference fields allow for a statement to be selected from a glossary and inline values entered after selection. Referred to as editable text selectors, these are used to build statements with variable content, but using a consistent format; for example, to describe cooking instructions in a common way without having to have an entry in the glossary for every permutation of time and temperature. Prior to release 18.0, just the separate statement and value elements were returned, now a packCopyText element containing the formed statement is also returned.

Key Features of the API Process

This section summarizes the key aspects of the API process:

- The external system makes a call to Supplier Evaluation using HTTPS. The external system must be registered in Supplier Evaluation, and an Application record created within IDCS corresponding to the External System.
- Calls to RESTful services use the GET, POST, PUT, and DELETE methods to retrieve, create, update, and delete records respectively. Calls to SOAP services use a predefined set of methods to perform the operations.
- Each call is logged in the Web Service Log in Supplier Evaluation. The request and response XML messages are attached to the logs.
- A predefined set of parameters is available to apply filters to the call, to retrieve a specific record, or set of records.
- Data is returned as XML, as defined by the associated WADL or WSDL schema. Only elements that contain data are returned; empty elements are omitted.
- For RESTful APIs, an initial GET call to the service returns the unique internal key ID of the returned records in either the recordId or the id element. The record ID is then used in a subsequent call using a URI to retrieve the record's full XML data set, or to perform an update operation.

Update functions require the full set of fields and their values to be passed, as the full contents of the record are replaced. If all fields are not present in the external system, the update process would need to first retrieve the current contents of the record from Supplier Evaluation before applying the changes to that data set, and submitting the full data set as the update.

- There could be several initiators for the data exchange, including the following:

- Human initiation, where an individual makes the request, such as by clicking a button to run the request.
- End user action, where the completion of an activity in the external system triggers the exchange.
- Scheduled request (or polling), where the external system sends or checks for data on a regular basis. For example, it may be looking for newly created Suppliers or Sites with a change in status.

For security purposes, the APIs rely on the external system polling, retrieving, or submitting calls to the APIs. Supplier Evaluation does not push data externally.

- The nature of the APIs may require multiple requests to be made, for example:
 - Request a list of Sites with their IDs.
 - Request the timestamp reference for each Site.
 - Identify any Sites changed or created within a given time scale.
 - Request the full details for that Site.

These types of requests can be automated. The external system is altered to enable the operations to take place in sequence according to a pre-defined algorithm.

Certain modifications can also be made to the external system to enable the storage of data returned from Supplier Evaluation. For example, if the external system requests a record to be created in Supplier Evaluation, a confirmation response is sent, containing the record ID. This ID may then be stored in the external system for future reference.



Note:

An External System must be specifically granted access to an API, either by the service or its endpoints. If none are specified, the External System will not have access to any of the APIs.

If a service is selected, it grants access to all of its endpoints, so there is no need to select the individual endpoints (if access is required to all of them); but selecting the service and all its endpoints would have the same effect.

Endpoints can be selected individually to grant granular access to specific endpoints of a service. In this case, it appears that if the service is also selected, that overrides the endpoint selections, thus providing access to them all anyway. So if wanting to grant access to just specific endpoints, the way to do it is not to select the service, but just the individual endpoints.

3

APIs Available in Supplier Evaluation

The APIs deployed in a particular implementation of Supplier Evaluation can be found at / services: <https://<hostname>/<context>/services>.

Individual Services

The following table lists the individual services available in Supplier Evaluation, indicating at which version each was introduced and, where appropriate, at which version it was deprecated.

Supplier Evaluation API Services

Service Name and Functions	Description	Type	From Version	To Version
UserRestService				
List of Values	Retrieves a list of retailer or supplier users in a paged list.	REST		
Retrieve Record by ID	Retrieves a single user's details. There are no request parameters, but the URL contains the id parameter that determines the record to retrieve.	REST		
Retrieve Record by Business Key	Retrieves a single user's details. There are no request parameters, but the URL contains the login id parameter that determines the record to retrieve.	REST		
Check Record Modification Timestamp	Retrieves the last modification time for a user. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Create Record	Creates a new retailer or supplier user.	REST		
Update Record	Updates an existing retailer or supplier user.	REST		
SupplierRestService				
List of Values	Retrieves a list of suppliers in a paged list.	REST		
Retrieve Record by ID	Retrieves a single supplier's details. There are no request parameters, but the URL contains the id parameter that determines the record to retrieve.	REST		
Retrieve Record by Business Key	Retrieves a single supplier's details. There are no request parameters, but the URL contains the key code parameter that determines the record to retrieve.	REST		
Check Record Modification Timestamp	Retrieves the last modification time for a supplier. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		

Service Name and Functions	Description	Type	From Version	To Version
Create Record	Creates a new supplier.	REST		
Update Record	Updates an existing supplier .	REST		
SiteRestService				
List of Values	Retrieves a list of sites in a paged list.	REST		
Retrieve Record by ID	Retrieves a single site's details. There are no request parameters, but the URL contains the id parameter that determines the record to retrieve.	REST		
Retrieve Record by Business Key	Retrieves a single site's details. There are no request parameters, but the URL contains the supplier and site code parameters that determine the record to retrieve.	REST		
Check Record Modification Timestamp	Retrieves the last modification time for a site. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Create Record	Creates a new site.	REST		
Update Record	Updates an existing site.	REST		
ContactRestService				
List of Values	Retrieves a list of contacts in a paged list.	REST		
Retrieve Record by ID	Retrieves a single contact's details. There are no request parameters, but the URL contains the id parameter that determines the record to retrieve.	REST		
Retrieve Supplier Contact Record by Business Key	Retrieves a single supplier contact's details. There are no request parameters, but the URL contains the supplier code, name, and email address parameters that determine the record to retrieve.	REST		
Retrieve Site Contact Record by Business Key	Retrieves a single site contact's details. There are no request parameters, but the URL contains the supplier and site codes, name, and email address parameters that determine the record to retrieve.	REST		
Check Record Modification Timestamp	Retrieves the last modification time for a contact. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
BusinessCategoryService				
List of Values	Retrieves a list of Business Categories in a paged list.	REST		
Retrieve Record	Retrieves a single Business Category (a path of all levels). There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		

Service Name and Functions	Description	Type	From Version	To Version
Check Record Modification Timestamp	Retrieves the last modification time for a Business Category. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Create Record	Creates a new Business Category.	REST		
Update Record	Updates an existing Business Category.	REST		
Delete Record	Deletes an existing Business Category.	REST		
AuditRestService				
List of Values (Audit)	Retrieves a list of Audits or Visits in a paged list.	REST		
Retrieve Record by ID (Audit)	Retrieves a single Audit or Visit (and its Non Conformances/Issues). There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Retrieve Record by Business Key (Audit)	Retrieves a single Audit or Visit (and its Non Conformances/Issues). There are no request parameters, but the URL contains the key code parameter that determines the record to retrieve.	REST		
Check Record Modification Timestamp (Audit)	Retrieves the last modification time for an Audit or Visit. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Create Record (Audit)	Creates a new Audit or Visit.	REST		
Update Record (Audit)	Updates an existing Audit or Visit.	REST		
List of Values (Checklist)	Retrieves a list of Checklists in a paged list.	REST		
Retrieve Record by ID (Checklist)	Retrieves a single Checklist. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Retrieve Record by Business Key (Checklist)	Retrieves a single Checklist. There are no request parameters, but the URL contains the key code parameter that determines the record to retrieve.	REST		
Create Record (Checklist)	Creates a new Checklist.	REST		
Update Record (Checklist)	Updates an existing Checklist.	REST		
Delete Record (Checklist)	Deletes an existing Checklist.	REST		
TaskRestService				
List of Values	Retrieves a list of current tasks for the user in the given language.	REST		
UrgentAssignmentsRestService				
Number of Urgent Items	Retrieves the number of urgent assignments pending for a given user.	REST		
ArtworkRestService				

Service Name and Functions	Description	Type	From Version	To Version
Started Activities	Retrieves a list of Project Activities that are flagged as Artwork activities, and their status has changed to Started within the specified date range.	REST		
Update Record	Updates the sub-status of existing Artwork Project Activities.	REST		
DataPrivacyService				
Right to Access	Retrieves personal data from the system relating to the name of an individual.	REST		
Right to be Forgotten	Erases personal data from the system for an individual (subject to conditions).	REST		
AttachmentRestService				
Retrieve Report Attachment	Retrieves attachment files that have been generated by the system as report outputs.	REST		
Retrieve Attachment	Retrieves files that have been attached to records.	REST		
ScorecardRestService				
List of Values	Retrieves a list of Assessments in a paged list.	REST		
Retrieve Record by ID	Retrieves a single Assessment. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Retrieve Record by Business Key	Retrieves a single Assessment. There are no request parameters, but the URL contains the key code parameter that determines the record to retrieve.	REST		
Check Record Modification Timestamp	Retrieves the last modification time for a Assessment. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Update Record	Updates an existing Assessment.	REST	24.1.301.0	NA
ProjectRestService				
List of Values	Retrieves a list of Processes in a paged list.	REST		
Retrieve Record by ID	Retrieves a single Process. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Retrieve Record by Business Key	Retrieves a single Process. There are no request parameters, but the URL contains the key code parameter that determines the record to retrieve.	REST		
Check Record Modification Timestamp	Retrieves the last modification time for a Process. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Update Record	Updates an existing Process.	REST	25.1.301.0	NA

Service Name and Functions	Description	Type	From Version	To Version
ActivityRestService				
List of Values	Retrieves a list of Process Activities in a paged list.	REST		
Retrieve Record by ID	Retrieves a single Process Activity. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Retrieve Record by Business Key	Retrieves a single Process Activity. There are no request parameters, but the URL contains the key code parameter that determines the record to retrieve.	REST		
Check Record Modification Timestamp	Retrieves the last modification time for a Process Activity. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST		
Update Record	Updates an existing Process Activity.	REST	25.1.301.0	NA
Update Status	Updates the status of an existing Process Activity.	REST	25.1.301.0	NA
ProductRecordService				
List of Values	Retrieves a list of Product Records in a paged list.	REST	24.1.301.0	NA
Retrieve Record by ID	Retrieves a single Product Record's details. There are no request parameters, but the URL contains the id parameter that determines the record to retrieve.	REST	24.1.301.0	NA
Retrieve Record by Business Key	Retrieves a single Product Record's details. There are no request parameters, but the URL contains the key code parameter that determines the record to retrieve.	REST	24.1.301.0	NA
Check Record Modification Timestamp	Retrieves the last modification time for a Product Record. There are no request parameters, but the URL contains the record id parameter that determines the record to retrieve.	REST	24.1.301.0	NA
Create Record	Creates a new Product Record.	REST	24.1.301.0	NA
Update Record	Updates an existing Product Record.	REST	24.1.301.0	NA

Services Grouped by Area

The following sections summarize the service features available by area of Supplier Evaluation functionality.

User and Supplier APIs

The following service features are applicable to the functionality available for user and supplier APIs:

Allows Supplier, Site, Contact, and User records to be created, updated, and extracted.
Allows Assessment records to be updated and extracted.
Allows Audit & Visit records to be created, updated, and extracted.
Does not allow deletion of records.
Does not import or export file attachments.
Does not include Alerts.
A system log of service transactions is kept (not reportable).

Audits API

The following service features are applicable to the functionality available for the audits API:

Allows Audit and Visit records to be created, updated, and extracted.
Includes the associated Issues/Non Conformances and Checklists.
Does not allow deletion of records.
Does not import or export file attachments.
A system log of service transactions is kept (not reportable).

Artwork Project Activities

The following service features are applicable to the functionality available for the Artwork Project Activities API:

Enables external Artwork systems to update the sub-status of specific Process Activities that relate to Artwork. The external system is not enabled to complete the activity in Supplier Evaluation; this must be done manually.
Activity templates in the Process module of Supplier Evaluation have an option to integrate that activity with an Artwork system.
Process Activities in Supplier Evaluation, which have been set to integrate with Artwork and which have a status of Started, are available to the external system.
The start and end dates of the Activity are not sent to the Artwork system. The Artwork process is usually well defined and it is usual for a project to be set up in the Artwork system at this point, using the Artwork system's own templates and based on the date that the message is received.

Data Privacy API

The following service features are applicable to the functionality available for the Data Privacy API:

Right to Access requests, to retrieve personal data relating to the name of an individual.
Right to be Forgotten requests, to erase personal data for an individual (subject to conditions).
Certain due diligence approval data is not permitted to be erased.
Data is erased by being overwritten with randomly-generated text.
The erasure of data is not recorded in Change History logs.

Product APIs

The following service features are applicable to the functionality available for product APIs:

Allows Product Records to be created, updated, and extracted.

Does not import file attachments.

Not intended for bulk upload of Product Records.

If the full content of the Product Record is not included in the data feed, subsequent manual data entry may be required.

A system log of service transactions is kept (not reportable).

Project/Process and Activity APIs

The following service features are applicable to the functionality available for the project (process) and activity APIs:

Allows Project (Process) records to be extracted.

Allows Activity records associated to a Project (Process) to be extracted.

Projects (Processes) and Activities can be updated.

Does not allow creation or deletion of records.

Does not import or export file attachments.

A system log of service transactions is kept (not reportable).

Miscellaneous

The following service features are also available:

Tasks & Urgent Items - For retrieval of a specific user's list of Tasks or count of Urgent Items.

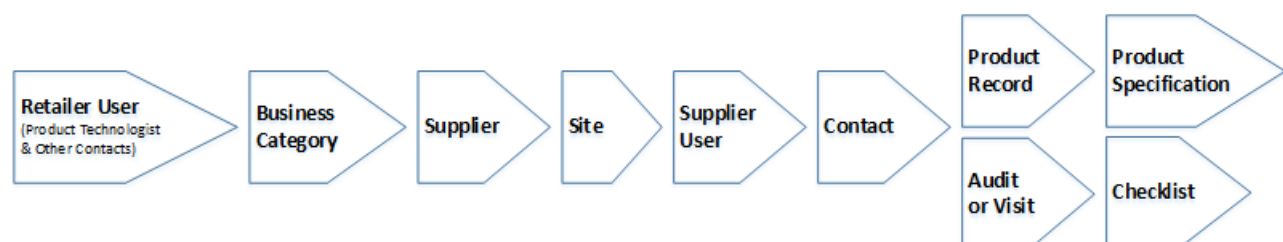
Business Categories - Allows Business Categories to be created, updated, deleted, and extracted.

Attachments & Reports - For retrieval of file attachments that have been generated by the system as report outputs, and files that have been attached to records.

Dependencies

Following is the typical sequence of records to be created in order to create all of the above services:

Figure 3-1 Dependencies



The following table indicates the dependencies of each type of record that can be created using the APIs, that is, what configuration or other records must be in place for the creation to succeed:

Record Creation Dependencies

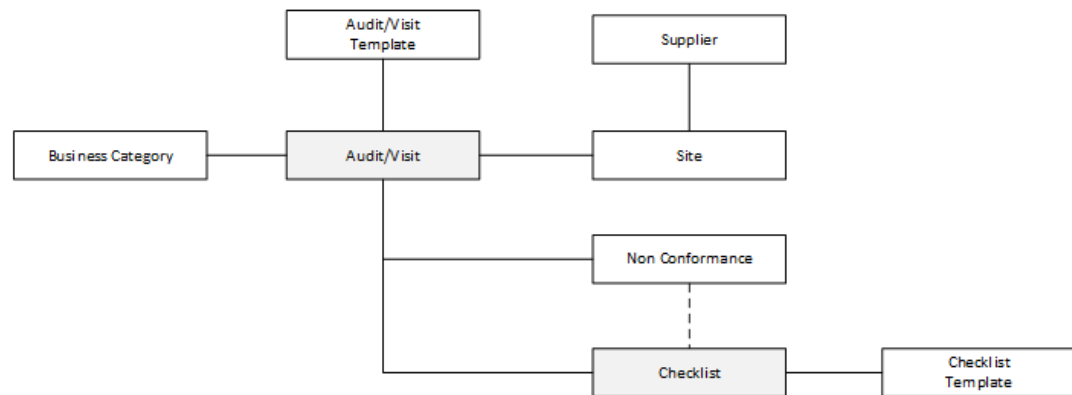
Record Type	Configuration	Other Record Types	Comments
User	Company (mandatory) Languages (mandatory) User Roles (mandatory) Authority Profiles Time Zones Users (Managers and Delegates) Areas	Supplier (if a supplier user)	When creating a user, it is associated to either the retailer or a supplier. If it is a supplier user, the ID of the Supplier record must be provided. The notification email sent to a new user is not generated when creating a user through the API.
Supplier	Billing Codes (mandatory) Business Units (mandatory) Supplier Types (mandatory) Countries Supplier Top Grades Currencies Invoicing Systems Supplier Statuses Areas	NA	When creating a Supplier, omit the Site and Supplier Contact elements, and omit the Billing Details ID. Create Sites, Contacts, AND Users after creating the Supplier. The initial user account is not automatically created when creating a supplier through the API.
Site	Technologists (mandatory) Business Categories (mandatory) Site Types (mandatory) Business Units Currencies Invoicing Systems Countries Audit Reasons Reference Types Reference Statuses Site Top Grades Site Statuses Risk Levels Other Production Details	Supplier User (Technologist)	When creating a Site, the ID of the Supplier record must be provided. Omit the Assessments and Audit elements, and omit the Billing Details ID and Reference ID. If assigning a Business Category, the ID of the Business Category record must be provided; if assigning a Technologist, the IDs of the User record must be provided. Create Contacts after creating the Site.
Contact	Contact Roles (mandatory)	User Supplier Site (if a Site contact)	When creating a Contact, the IDs of the associated User and Person records must be provided. The Person record holds an individual's personal contact details; the Contact record holds the individual's contact details in relation to that role (which may be different per role); the User record represents the individual's user account. A contact is associated to a supplier and/or a site; the IDs of the Supplier and Site records must be provided as appropriate. Omit the Site element if not a site-level contact.

Record Type	Configuration	Other Record Types	Comments
Business Category	Business Categories (parent levels) Specification Types	NA	When creating a Business Category, if the category is a lower-level category, the ID of its parent Business Category record must be provided.
Audit/Visit	Audit/Visit Template (mandatory) Audit/Visit Status (mandatory) Technologist (mandatory) People Present (mandatory) Supplier Contacts Business Categories Non Conformance: Assigned to Completed by Approved by	Supplier Site User	An Audit or Visit is linked to a supplier/site; the IDs of the Supplier and Site records must be provided. If assigning a Business Category, the ID of the Business Category record must be provided; if assigning a Technologist or People Present, the IDs of the User record must be provided. Non Conformances may include links to User records for the users who the issue is assigned to, was completed by, or is approved by. If the person is not a user of the application, a corresponding text field can be used to record the name. Also, see notes below.
Checklists	Audit/Visit (mandatory) Checklist Question (mandatory)	Audit/Visit Checklist Template	A Checklist is linked to an Audit/Visit, the IDs of the Audit or Visit record must be provided. The Checklist's questionnaire questions and available answers are derived from the Checklist Templates assigned to the Audit/Visit, the IDs of the Template records must be provided.
Product Record	Product Record Statuses (mandatory) Technologist (mandatory) Other Contacts (mandatory) Business Categories	Supplier (if Active) Site (if linked to a site) User	A Product Record may be linked to a supplier/site; the IDs of the Supplier and Site records must be provided as appropriate. If assigning a Business Category, the ID of the Business Category record must be provided; if assigning a Technologist or Other Contact, the IDs of the User record must be provided. Technologist and Other Contacts are mandatory, however will be defaulted to TBC on creation if not provided (the dummy TBC User record must already exist). For the status to be Active, the Product Record must be linked to a Supplier.

Notes on the Creation and Maintenance of Audits/Visits and Checklists

- The following diagram shows the main links between Audits/Visits, their Checklists, and other entities:

Figure 3-2 Links Between Audits/Visits and Other Entities



- For the creation of an Audit/Visit and its Checklists, in addition to providing the links to the associated Audit/Visit Template and Checklist Template records, certain values from the templates must also be included in the payload of the Audit/Visit Create and Update functions.
- It is recommended that the creation of Audits/Visits and Checklist be managed within the Supplier Evaluation application. Once created, records can be taken into the external system where the audit is performed, prior to importing back into Supplier Evaluation.

This approach ensures the necessary template fields will have been obtained with the initial Retrieve Record function.

The following diagram shows the stages of a typical scenario:

Figure 3-3 Stages of a Typical Scenario



- If it is necessary to create audits or visits from scratch in the external system, the recommended approach would be to create a set of *gold standard* audits/visits in Supplier Evaluation, which are assigned to a dummy supplier and site.

To create a new audit or visit, the process would be to retrieve the appropriate *gold standard* audit/visit through the GET function, amend the returned XML data to reflect the

audit/visit to be created, and then submit the creation of the new audit/visit (plus any non conformances and checklists) using the POST function.

- If a checklist audit is to be created using the Create Audit service, the Audit/Visit record must first be created using the Create Audit function, followed by the creation of each of its Checklist records.

Creating an audit through the API which is linked to a checklist template does not automatically create the checklists. They must be individually created using the Create Checklist function.

- An audit/visit's non conformances are included as separate elements within the audit/visit data. An audit/visit's non conformances are allocated when creating or updating the audit/visit.

 Note:

The API does not automatically create non conformances based on the answers of checklist questions.

Linking to Related Records

Links to associated records are included in the XML as an id element, which holds the record ID of the related record. In some cases, a code element also holds the related record's business key.

For example, the XML returned from the SiteRestService API will contain a supplier element, which itself contains an id element and a code element. The value in the id element can be used by the SupplierRestService Retrieve Record by ID function to retrieve the full details of the Supplier record. Alternatively, the value in the code element can be used by the SupplierRestService Retrieve Record by Business Key function to retrieve the Supplier record.

The id element is also used in the XML to hold the record ID of a value that is held in a glossary such as a recipe ingredient or a country.

The following table shows where an API provides links to other records, and the id or code elements used to locate those records. The list of notes below the table indicates the service to use to retrieve the related record.

Related Record Links

API	Links to Records
Supplier	Contact: contactDetail/id (Contact record ID link). See note 7. Contact (User): person/id (User record ID link). See note 1. Site: site/id (Site record ID link). See note 3. Site: site/code site (Site business key link). See note 4.

API	Links to Records
Site	Supplier: supplier/id (Supplier record ID link). See note 5. Supplier: supplier/code (Supplier business key link). See note 6. Business Categories: businessCategories/id (Business Category record ID link). See note 8. Lead Business Category: leadBusinessCategory/id (Business Category record ID link). See note 8. Lead Technologist (User): leadTechnicalManager/id (User record ID link). See note 1. Lead Technologist (User): leadTechnicalManager/code (User business key link). See note 2. Other Technologists (User): otherTMs/id (User record ID link). See note 1. Other Technologists (User): otherTMs/code (User business key link). See note 2. Contact: siteContact/id (Contact record ID link). See note 7.
Contact	Contact (User): person/id (User record ID link). See note 1. Supplier: company/id (Supplier record ID link). See note 5. Supplier: Company/code (Supplier business key link). See note 6. Site: site/id (Site record ID link). See note 3. Site: Site/code (Site business key link). See note 4.
User	Supplier: supplier/id (Supplier record ID link). See note 5. Supplier: Supplier/code (Supplier business key link). See note 6.
Audit/Visit	Supplier: supplier/id (Supplier record ID link). See note 5. Supplier: supplier/code (Supplier business key link). See note 6. Site: site/id (Site record ID link). See note 3. Site: Site/code (Site business key link). See note 4. People Present (User): user/id (User record ID link). See note 1. People Present (User): user/code (User business key link). See note 2. Lead Technologist (User): leadTechnicalManager/id (User record ID link). See note 1. Lead Technologist (User): leadTechnicalManager/code (User business key link). See note 2. Business Categories: businessCategories/id (Business Category record ID link). See note 8. Lead Business Category: leadBusinessCategory/id (Business Category record ID link). See note 8. Non Conformance Assigned To (User): assignedTo/id (User record ID link). See note 1. Non Conformance Completed By (User): completedBy/id (User record ID link). See note 1. Non Conformance Approved By (User): approvedBy/id (User record ID link). See note 1.
Checklist	Audit/Visit: audit/id (Audit/Visit record ID link). See note 9.
Scorecard/ Assessment	Supplier: supplier/id (Supplier record ID link). See note 5. Supplier: supplier/code (Supplier business key link). See note 6. Site: site/id (Site record ID link). See note 3. Site: site/code (Site business key link). See note 4. Product Record (Item Assessment): productRecord/id (Product Record record ID link). See note 8. Product Record (Item Assessment): productRecord/code (Product Record business key link). See note 9. Person Responsible (User): personResponsible/id (User record ID link). See note 1. Business Category: businessCategories/id (Business Category record ID link). See note 9. Submitted By (User): submittedBy/id (User record ID link). See note 1.

API	Links to Records
Project/Process	Parent Project (Process): parentProject Supplier: supplier/id (Supplier record ID link). See note 5. Supplier: supplier/code (Supplier business key link). See note 6. Site: site/id (Site record ID link). See note 3. Site: site/code (Site business key link). See note 4. Process Manager (User): projectManager/id (User record ID link). See note 1.
Activity	Project (Process): projectId

Notes:

1. User record ID link: UserRestService - Retrieve Record by ID
2. User business key link: UserRestService - Retrieve Record by Business Key
3. Site record ID link: SiteRestService - Retrieve Record by ID
4. Site business key link: SiteRestService - Retrieve Record by Business Key
5. Supplier record ID link: SupplierRestService - Retrieve Record by ID
6. Supplier business key link: SupplierRestService - Retrieve Record by Business Key
7. Contact record ID link: ContactRestService - Retrieve Record by ID
8. Business Category record ID link: BusinessCategoryService - Retrieve Record
9. Audit/Visit Record ID link: AuditRestService - Retrieve Record by ID (Audit)

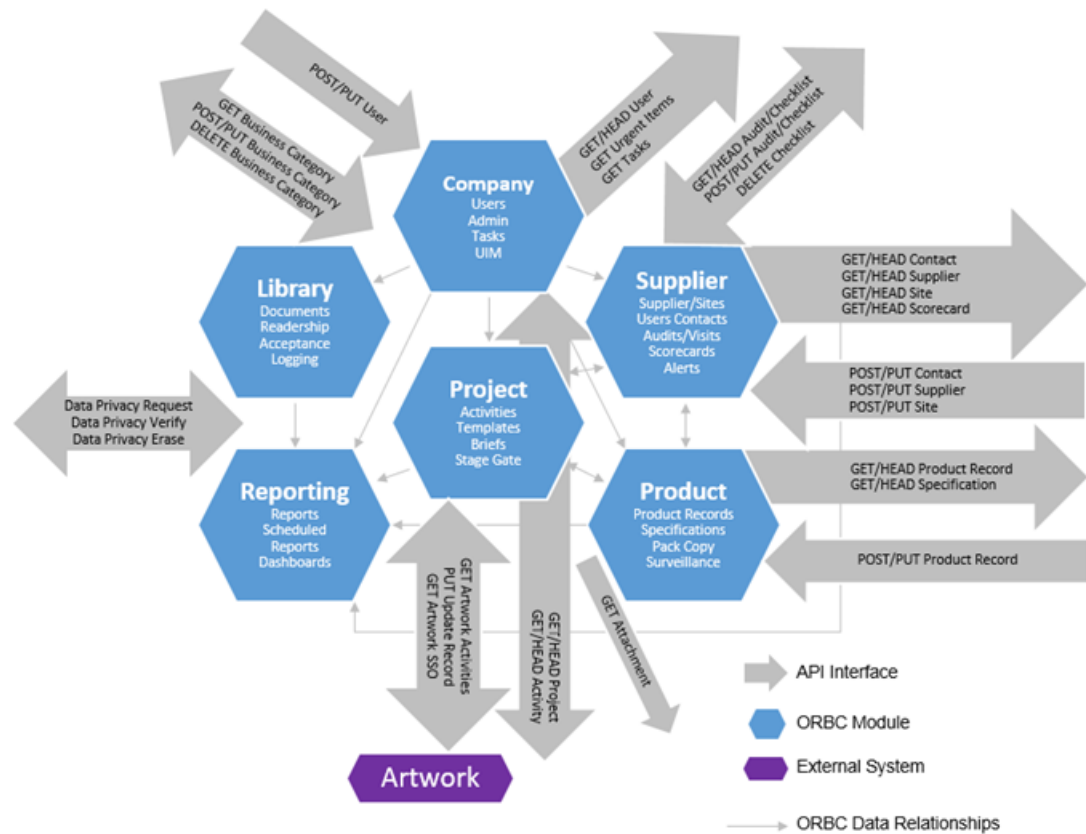
4

RESTful APIs

The following RESTful APIs are available:

- [UserRestService](#): Users
- [SupplierRestService](#): Suppliers
- [SiteRestService](#): Sites
- [ContactRestService](#): Supplier/Site Contacts
- [TaskRestService](#): Users' Tasks
- [UrgentItemsRestService](#): Users' Urgent Items
- [ArtworkRestService](#): Artwork Activities
- [BusinessCategoryService](#): Business Categories
- [AuditRestService](#): Audits and Visits
- [DataPrivacyService](#): Data Privacy Requests
- [AttachmentRestService](#): Attachment Files
- [ScorecardRestService](#): Assessments
- [ProjectRestService](#): Processes
- [ActivityRestService](#): Process Activities
- [ProductRecordRestService](#): Product Records

Figure 4-1 Overview of RESTful APIs

**Note:**

Only minimal business logic and validation is applied by the Supplier Evaluation REST APIs.

When creating or updating records, all values must be included. If a value or element is omitted from the XML, the field contents will be cleared when updating the record.

Generally, only the validation necessary to ensure the system's referential integrity is applied when updating records through the APIs. It is therefore possible to set the contents of some fields with values that are rejected when the user manually edits or validates the record through the UI. In some cases, it may be necessary to regress these records back to a prior status in order to edit and correct values that have been set incorrectly through the UI.

Parameters and Filtering

Various parameters can be included in calls to the APIs, generally to define what data is to be returned, but can also control how records are returned. The following table lists some commonly used parameters.

Common Parameters

Parameter	Type	Description
offset	int	Used with pageSize to control the paging of a returned list of records. Specifies the starting point for the retrieval of records. If not specified, zero is assumed. For example, to retrieve 150 records: 1. A call with offset = 0 and pageSize = 50 returns the first 50 records. 2. Then, a call with offset = 50 and pageSize = 50 returns the next 50. 3. Then, a call with offset = 100 and pageSize = 50 returns the final 50.
pageSize batchSize	int	Used with offset to control the paging of a returned list of records. Specifies the number of entries in each page of returned list of records. If not specified, 30 is assumed. The maximum is 100.
isActive	Boolean	Used to only return records that are flagged as being active (TRUE).
modifiedSince	string	Used to locate records that have been created or updated since a specific date/time.
modifiedUntil	string	Used to locate records that have been updated or created until a specific date/time.

When a list of records is returned, it may include a `previousPage` and `nextPage` element. These elements provide URI links to the previous or next pages of records respectively. A `totalRecords` element is included that shows the total number of records available for retrieval.

For example, using the List of Values function to retrieve a list of User records, where there are 150 records to be retrieved, with the `pageSize` parameter set to 50 returns the first 50 records within entity elements, along with a `nextPage` element containing a URI link to the next 50 records which can be used to retrieve the next 50 records. When the second page of 50 records are returned, the XML includes a `nextPage` URI link to the final 50 records, along with a `previousPage` URI link to the first 50. The following graphic illustrates this example.

Figure 4-2 Example of Retrieving a List of Users



```

<entries xsi:type="userLink" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <recordId>100</recordId>
  <recordLink>http://example.com/services/rest/user/rest/user/100</recordLink>
  <loginId>John Smith</loginId>
  <name>John Smith</name>
</entries>

<previousPage>example.com/services/rest/user/?offset=0&pageSize=50</previousPage>
<nextPage>example.com/services/rest/user/?offset=100&pageSize=50</nextPage>

<totalRecords>150</totalRecords>

```

A callout box with the text "Last entry in the list of 50 returned users" points to the `<name>John Smith</name>` element in the XML snippet.

The `totalRecords` element shows the total number of retrievable records. The `previousPage` and `nextPage` elements are only present if there are previous and/or next pages to be retrieved.

**Note:**

Date/Time parameters must be provided in the YYYY-MM-DD hh:mm:ss format.

When retrieving records, the returned XML only includes elements that actually contain data; empty elements are omitted.

The details of the returned records are contained in an entities element, and repeated in an entryArray element. The entryArray element is a representation from the Java objects and can be ignored.

Where a date/time is passed in as a parameter to an endpoint, such as fromDate and toDate, validation checks for a valid date/time. If not valid, an HTTP 417 is returned, along with an XML error message detailing the exact issue with the date/time.

Error Messages

The Supplier Evaluation APIs use standard web services messaging protocols to notify the success or failure of a call to the service. Where an API handles a specific error condition, details of the returned message can be found in the following sections. The following table lists various generic error messages that may be returned by calls to any of the APIs.

Generic Error Messages

Element	Message	Meaning
userId	HTTP Status 401 - Bad credentials	User ID or password is invalid
Password	HTTP Status 401 - Bad credentials	User ID or password is invalid
Disabled	HTTP Status 401 - User is disabled	External System account is not enabled in Supplier Evaluation
offset	HTTP 417 IllegalStateException: Offset must be a positive integer	Invalid offset value - must be between zero and 2,147,483,647
offset	HTTP 404 Not Found	Not numeric or an integer
pageSize	HTTP 417 IllegalStateException: The Page Size must be between 1 and 100	Invalid page size value
	HTTP 404 Not Found	Record cannot be located due to an invalid key or ID
	HTTP 417 Expectation Failed	Record cannot be located due to an invalid ID
	HTTP Status 500 - Internal Server Error	Unspecified internal error occurred
	HTTP Status 500 - Internal Server Error	Non-numeric value in numeric parameters
modifiedSince	HTTP 417 <errorMessage>	Not a valid date, whereby <errorMessage> will identify the issue For example: IllegalStateException: Cannot parse "2018-03-25 01:00:00": Illegal instant due to time zone offset transition (Europe/London) In this example, due to the clocks moving forward (in the server's time-zone), the passed in date is invalid

Element	Message	Meaning
modifiedUntil	HTTP 417 <errorMessage>	Not a valid date, whereby <errorMessage> will identify the issue (see example above)
	HTTP 400 Bad Request	Malformed XML, or general error in content of the XML

Wildcard Searches

The % character can be used as a wildcard filter when locating the records.

Using the % at the start of the string will search for matches where the given string ends with the text being searched for, for example, %smith could be used to search for all users with a surname of smith.

Using the % at the end of the string will look for matches where the given string begins with the text being searched for, for example, mark% could be used to search for all users with the name mark.

The following table shows which APIs and functions support the use of wildcards.

Wildcard Searches

API	Function
User	List of Values
Supplier	List of Values
Site	List of Values
Contact	List of Values Retrieve Supplier Contact Record by Business Key Retrieve Site Contact Record by Business Key
Business Category	List of Values
Task	List of Values
Urgent Items	Number of Urgent Items

Notes:

- Wildcard searches are not case sensitive so, for example, searching a name field for John Smith or john smith will return the same matches.
- When searching glossary parameters, be aware that they relate to the code (not the description) of the glossary entry.
- Wildcard searches can typically be applied to parameters that are string type. See the Wildcard Search Supported column in the APIs' List of Values functions for which URL parameters support it.

Deleted Records

By default, soft deleted records are returned by the APIs (the XML contains a deleted element, which is set to true). To exclude soft deleted records, it is necessary to use a *softDelete=false* parameter when calling the API. For example:

/creations-core/services/rest/site?softDelete=false

UserRestService

This section describes the API for managing retailer and supplier users. The following functions are available:

- **List of Values:** retrieves a list of users
- **Retrieve Record by ID:** retrieves a User record using its unique identifier
- **Retrieve Record by Business Key:** retrieves a User record using its business key
- **Check Record Modification Timestamp:** retrieves the timestamp when a User record was last updated
- **Create Record:** creates a new User record
- **Update Record:** updates an existing User record

List of Values

Description

Retrieves a list of users in a paged list. Use this function to retrieve a simple list of user names and IDs, or to locate User record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/user **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30
name	Optional	string	No	Yes	John Smith
jobTitle	Optional	string	No	Yes	Process Manager
language	Optional	string	No	Yes	en_GB
country	Optional	string	No	Yes	UK
userRoles	Optional	string	No	Yes	PROCESS ADMINISTRATOR
userType	Optional	string	Yes	No	RETAILER
authorityProfiles	Optional	string	No	Yes	PROCESS MANAGER
siteStatus	Optional	string	Yes	No	ACTIVE-INACTIV E

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
supplierActive	Optional	string	No	No	true returns active users no value returns both active and inactive users
modifiedSince	Optional	string	No	No	2015-05-19 13:30:39
modifiedUntil	Optional	string	No	No	2015-05-19 13:30:39

Example URLs

.../services/rest/user/?offset=2&pageSize=20 .../services/rest/user/?siteStatus=Active .../
services/rest/user/?name=Frank%

Response Details

For a successful response, XML is returned with a UserLinkList root element containing an entries element for each matched user. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	User record's internal ID
recordLink	string	URI to the UserRestService Retrieve record service for this user
loginId	string	Login ID / business key to the User record
name	string	User's name

The returned XML also contains a totalRecords element, which states the total number of retrievable records that match the filter parameters.

Error Messages

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record by ID

Description

Retrieves a single User record's details using the record's internal unique ID (which is not visible in the UI). Use this function to retrieve the full details of an individual user.

Endpoint address: /services/rest/user/{id} **HTTP method:** GET

Request Details

The URL contains the {id} parameter that determines the record to retrieve.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a localeData element returned for the specified locales if data is present.

Example URL

.../services/rest/user/105

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a userFullDTO root element containing the individual attributes of the requested User record. If an ID is not specified, a list of all users is returned (per the List of Values function).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 1 - Framework* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the User to return with id:<###>	Invalid {id} - not found

Retrieve Record by Business Key

Description

Retrieves a single User record's details using its business key (login ID). Use this function to retrieve the full details of an individual user using its Supplier Evaluation login ID.

Endpoint address: /services/rest/byKey/{loginId} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {loginId} parameter that determines the record to retrieve.

Example URL

.../services/rest/user/byKey/Frank Jones

Response Details

If successful, an HTTP 301 redirect response is sent containing a header Location with the URI of the document requested.

The returned URI represents the call to be made to the Retrieve Record by ID function (with the id substituted with the relevant value), which will return a userFullDTO root element containing the individual attributes of the requested User record.



Note:

If your method of calling the API is configured to automatically follow HTTP 301 redirects, the call may automatically return a userFullDTO root element containing the individual attributes of the requested User record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 1 - Framework* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
loginId	HTTP 404 Not found	Invalid {loginId} - blank or not found
loginId	HTTP 404 Not found	Invalid {loginId} - not found

Check Record Modification Timestamp

Description

Retrieves the last modification date and time of a User record. Use this function to determine when a user's details were last updated.

Endpoint address: /services/rest/user/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/user/105

Response Details

If successful, an HTTP 200 response is sent containing the Last-Modified header showing the date and time of the last update of the requested User record. For example:

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Messages

Element	Message	Meaning
id	HTTP 417 Expectation Failed	Invalid {id} - not found
id	HTTP 404 Not found	Invalid {id} - not numeric

Create Record

Description

Creates a new User record. Use this function to create a new user in Supplier Evaluation based on data sourced from the external system.

Dependencies: If creating a supplier user, the Supplier must be present in the application and its record ID obtained. For more information, see [Dependencies](#).

Endpoint address: /services/rest/user **HTTP method:** POST

Request Details

The body of the request contains a UserFullDTO to specify the details of the user to create. Compared to retrieving a user (which uses the same UserFullDTO type), this request is much shorter. Only the attributes that are to be populated on the created User record need to be included. As a minimum, this must include the fields shown in the following table:

Supplier User Mandatory Fields

Field Name	Element Name
Name	person / name
Email	person / email
Login Id	code
User Role	role / code
Supplier Code	person / supplier / code supplier / code
User Type	userType (Fixed value SUPPLIER)
Deleted Record	deleted (Set to false)
Login Id Disabled	disabled (Set to false)
Login Permitted	loginPermitted (Set to true)
Language	language / code

Example Request XML

This example shows the minimum requirement to be able to create a supplier user.

```
<ns0:userFullDTO xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
  <ns0:code>jphillips</ns0:code>
  <ns0:person>
    <ns0:email>jphillips@supplier.co.uk</ns0:email>
    <ns0:name>Jane Phillips</ns0:name>
    <ns0:supplier>
```

```

        <ns1:code>A0174</ns1:code>
      </ns0:supplier>
    </ns0:person>
    <ns0:role>
      <ns1:code>SUPPLIER ADMINISTRATOR</ns1:code>
    </ns0:role>
    <ns0:supplier>
      <ns1:code>A0174</ns1:code>
    </ns0:supplier>
    <ns0:userType>SUPPLIER</ns0:userType>
    <ns0:deleted>false</ns0:deleted>
    <ns0:disabled>false</ns0:disabled>
    <ns0:loginPermitted>true</ns0:loginPermitted>
  </ns0:userFullDTO>

```

**Note:**

When creating a supplier user, the user type must be set to SUPPLIER. The supplier code must be set to match the required supplier from Supplier Evaluation.

Site User Mandatory Fields

Field Name	Element Name
Name	person / name
Email	person / email
Login Id	code
User Role	role / code
Supplier Code	person / supplier / code supplier / code
Sites	sites / code sites/ supplierCode
User Type	userType (Fixed value SITE)
Deleted Record	deleted (Set to false)
Login Id Disabled	disabled (Set to false)
Login Permitted	loginPermitted (Set to true)
Language	language / code

Example Request XML

This example shows the minimum requirement to be able to create a site user.

```

<ns0:userFullDTO xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
  <ns0:code>jphillips</ns0:code>
  <ns0:person>
    <ns0:email>sphillips@supplier.co.uk</ns0:email>
    <ns0:name>Sean Phillips</ns0:name>
    <ns0:supplier>
      <ns1:code>WS0001</ns1:code>
      <ns1:id>22</ns1:id>
    </ns0:supplier>
  </ns0:person>

```

```

<ns0:role>
  <ns1:code>SITE USER</ns1:code>
</ns0:role>
<ns0:sites>
  <ns1:code>WS0001-0001</ns1:code>
  <ns1:id>26</ns1:id>
  <ns1:supplierCode>WS0001</ns1:supplierCode>
  <ns1:id>22</ns1:id>
</ns0:sites>
<ns0:supplier>
  <ns1:code>WS0001</ns1:code>
  <ns1:id>22</ns1:id>
</ns0:supplier>
<ns0:userType>SITE</ns0:userType>
<ns0:deleted>false</ns0:deleted>
<ns0:disabled>false</ns0:disabled>
<ns0:loginPermitted>true</ns0:loginPermitted>
</ns0:userFullDTO>

```

**Note:**

When creating a site user the user type must be set to SITE. The supplier code must be set to match the required supplier from Supplier Evaluation. The site codes must be set to match the required site from Supplier Evaluation.

Retailer User Mandatory Fields

Field Name	Element Name
Name	person / name
Email	person / email
Login Id	code
User Role	role / code
Supplier Code	person / supplier / code supplier / code (Fixed value RETAILER)
User Type	userType (Fixed value RETAILER)
Deleted Record	deleted (Set to false)
Login Id Disabled	disabled (Set to false)
Login Permitted	loginPermitted (Set to true)
Language	language / code

Example Request XML

This example shows the minimum requirement to be able to create a retailer/portal owner user.

```

<ns0:userFullDTO xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
  <ns0:code>james howard</ns0:code>
  <ns0:person>
    <ns0:email>james.howard@example.com</ns0:email>
    <ns0:name>James Howard</ns0:name>
    <ns0:supplier>

```

```

        <ns1:code>RETAILER</ns1:code>
      </ns0:supplier>
    </ns0:person>
    <ns0:role>
      <ns1:code>POWER USER</ns1:code>
    </ns0:role>
    <ns0:supplier>
      <ns1:code>RETAILER</ns1:code>
    </ns0:supplier>
    <ns0:userType>RETAILER</ns0:userType>
    <ns0:deleted>false</ns0:deleted>
    <ns0:disabled>false</ns0:disabled>
    <ns0:loginPermitted>true</ns0:loginPermitted>
  </ns0:userFullDTO>

```



Note:

When creating a retailer/portal owner user, the user type must be set to RETAILER. The supplier code must be set to RETAILER.

Enabling User Accounts

Where the record is linked to another record, such as the Role in these cases, the business key must be provided (not the description) in order to form the link between the records. In general, the business key will be the code attribute of the linked record.

Set loginPermitted to true to ensure a corresponding IDCS or OCI IAM profile is automatically created for the user. If an IDCS or OCI IAM profile is not required, setting the loginPermitted attribute to false will skip the IDCS or OCI IAM profile creation.

Set the deleted attribute to false to ensure the record can be seen in the UI. If omitted, or set to true, it will exist, but will not be visible.

Where the record is linked to another record, such as the Role in these cases, the business key must be provided (not the description) in order to form the link between the records. In general, the business key will be the code attribute of the linked record.

When a new record is created in Supplier Evaluation, a corresponding IDCS or OCI IAM profile is automatically created to manage login authentication. If an IDCS or OCI IAM profile is not required, setting the loginPermitted attribute to false will skip the IDCS or OCI IAM profile creation.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 1 - Framework* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

For a successful response, an HTTP 200 response is sent with a body containing a UserLink root element. The root element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Newly created User record's internal ID
recordLink	string	URI to the newly created User record, for use in a GET request

Element	Type	Description
loginId	string	Login ID / business key to the newly created User record
name	string	User's name

Error Responses

If the supplied data does not result in a valid User (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Element	Message	Meaning
code	ERROR: class com.micros.creations.core.domain.User.code - The condition is invalid	Code not provided
email	ERROR: class com.micros.creations.core.domain.User.person.email - The condition is invalid	Email not provided
name	ERROR: class com.micros.creations.core.domain.User.person.name - The condition is invalid	Name not provided
role	ERROR: class com.micros.creations.core.domain.User.role - A user must have an active authority profile from either roles or additional authority profiles	Role or authority profiles not provided

Update Record

Description

Updates an existing User record. Use this function to update a user's details in Supplier Evaluation based on data sourced from the external system.

Endpoint address: /services/rest/{id} **HTTP method:** PUT

Request Details

The body of the request contains a UserFullDTO to specify the updates to the User record. Compared to retrieving a user (which uses the same UserFullDTO type), this request is much shorter. As a minimum, the values specified as mandatory for the Create Record function (see above) must be included.

The request content is similar to that for creating a user. After the call, the User record is updated to match the request.

**Note:**

When updating records, all values must be included. If a value or element is omitted from the request, the field contents will be cleared on the User record.

Any values passed for Login Id or Email Address are ignored, as they cannot be updated once the record has been created.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 1 - Framework* for details of their mapping to the fields within the Supplier Evaluation UI.

Process for Disabling User Accounts

A user's account can be disabled using the Update Record function as follows:

1. Call the Retrieve Record by Business Key GET function or the Retrieve Record by Id function, passing the login id or the record id respectively, to retrieve the full XML for the user.
2. Amend the returned XML as follows:
Set the account to disabled with `<ns0:disabled>true<ns0:disabled>` and `<ns0:loginPermitted>false<ns0:loginPermitted>`
Remove the localeData and activeAuthorityProfiles elements.
3. Call the Update Record PUT function, passing the full amended XML to disable the account.

Response Details

If successful, an HTTP 200 response is sent with a body containing a UserLink element. The UserLink element consists of the returned elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	User record's internal ID
recordLink	string	URI to the User record, for use in a GET request
loginId	string	Login ID/business key to the User record
name	string	User's name

Error Responses

If the supplied data does not result in a valid user (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

SupplierRestService

This section describes the API for managing suppliers. The following functions are available:

- [List of Values](#): retrieves a list of suppliers

- **Retrieve Record by ID:** retrieves a Supplier record using its unique identifier
- **Retrieve Record by Business Key:** retrieves a Supplier record using its business key
- **Check Record Modification Timestamp:** retrieves the timestamp when a Supplier record was last updated
- **Create Record:** creates a new Supplier record
- **Update Record:** updates an existing Supplier record

**Note:**

When integrated with Merchandising, using the SupplierRestAPI to make changes that affect the Billing Details table can have an impact on the integration.

List of Values

Description

Retrieves a list of suppliers in a paged list. Use this function to retrieve a simple list of supplier names and IDs, or to locate Supplier record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/supplier **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30
supplierStatus	Optional	string	Yes	No	AWAITING REGISTRATION
supplierCode	Optional	string	Yes	Yes	A0001
supplierName	Optional	string	Yes	Yes	West Road Site
supplierType	Optional	string	Yes	Yes	AGENT
country	Optional	string	Yes	Yes	UK
leadBusinessUnit	Optional	string	Yes	Yes	UK
isActive	Optional	string	No	No	true returns active suppliers no value returns both active and inactive suppliers
modifiedSince	Optional	string	No	No	2015-05-19 13:30:39
modifiedUntil	Optional	string	No	No	2015-05-19 13:30:39

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
invoicingRef	Optional	string	No	Yes	ABC123

Example URLs

.../services/rest/supplier/?offset=2&pageSize=20 .../services/rest/supplier/?
supplierStatus=AWAITING REGISTRATION .../services/rest/supplier/?supplierName=API%

Response Details

For a successful response, XML is returned with a SupplierLinkList root element containing an entries element for each matched supplier. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Supplier record's internal ID
recordLink	string	URI to the SupplierRestService Retrieve record service for this supplier
code	string	Supplier code business key to the Supplier record
name	string	Supplier's name
localName	string	Supplier's name in business language (if used)

The returned XML also contains a totalRecords element, which states the total number of retrievable records that match the filter parameters.

Error Messages

Element	Message	Meaning
supplierStatus	HTTP 417 IllegalStateException: The Supplier Status <###> does not exist	Invalid supplier status
supplierStatus	HTTP 417 IllegalStateException: The Supplier Statuses <###~###~###> do not exist	Invalid supplier status
supplierType	HTTP 417 IllegalStateException: The Supplier Type <###> does not exist	Invalid supplier type
supplierType	HTTP 417 IllegalStateException: The Supplier Types <###~###> do not exist	Invalid supplier type
country	HTTP 417 IllegalStateException: The Country <###> does not exist	Invalid country
country	HTTP 417 IllegalStateException: The Countries <###~###> do not exist	Invalid country
leadBusinessUnit	HTTP 417 IllegalStateException: The Business Unit <###> does not exist	Invalid business unit
leadBusinessUnit	HTTP 417 IllegalStateException: The Business Units <###~###> do not exist	Invalid business unit

Element	Message	Meaning
pageSize	Error Code: INVALIDRESTSERVICEPAGESIZE	Page Size must be between 1 and 100
offset	Error Code: INVALIDOFFSET	Offset must be a positive integer
supplierStatus	Error Code: INVALIDSUPPLIERSTATUS	Invalid Supplier Status
supplierType	Error Code: INVALIDSUPPLIERTYPE	Invalid Supplier Type
country	Error Code: INVALIDCOUNTRY	Invalid Country
leadBusinessUnit	Error Code: INVALIDBUSINESSUNIT	Invalid Business Unit
isActive	Error Code: INVALIDBOOLEAN	Invalid isActive value Permitted values (not case-sensitive) are: true; yes; 1; false; no; 0

Retrieve Record by ID

Description

Retrieves a single Supplier record's details using the record's internal ID (which is not visible in the UI). Use this function to retrieve the full details of an individual supplier.

Endpoint address: /services/rest/supplier/{id} **HTTP method:** GET

Request Details

The URL contains the {id} parameter that determines the record to retrieve.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a localeData element returned for the specified locales if data is present.

Example URL

.../services/rest/supplier/9

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a supplierFullDTO root element containing the individual attributes of the requested Supplier record. If an ID is not specified, a list of all suppliers is returned (per the List of Values function).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Supplier to return with id:<###>	Invalid {id} - not found

Retrieve Record by Business Key

Description

Retrieves a single Supplier record's details using its business key (supplier code). Use this function to retrieve the full details of an individual supplier using the Supplier Evaluation supplier code.

Endpoint address: /services/rest/supplier/byKey/{code} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {code} parameter that determines the record to retrieve.

Example URL

.../services/rest/supplier/byKey/A0901

Response Details

If successful, an HTTP 301 redirect response is sent containing a header Location with the URI of the document requested.

The returned URI represents the call to be made to the Retrieve Record by ID function (with the id substituted with the relevant value), which will return a supplierFullDTO root element containing the individual attributes of the requested Supplier record.



Note:

If your method of calling the API is configured to automatically follow HTTP 301 redirects, the call may automatically return a supplierFullDTO root element containing the individual attributes of the requested Supplier record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
code	HTTP 404 Not Found	Invalid {code} - blank
code	HTTP 404 Not Found	Invalid {code}

Check Record Modification Timestamp

Description

Retrieves the last modification date and time of a Supplier record. Use this function to determine when a supplier's details were last updated.

Endpoint address: /services/rest/supplier/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/supplier/9

Response Details

If successful, an HTTP 200 response is sent containing the Last-Modified header showing the date and time of the last update of the requested Supplier record.

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Responses

Element	Message	Meaning
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 Expectation Failed	Invalid {id} - not found

Create Record

Description

Creates a new Supplier record. Use this function to create new suppliers in Supplier Evaluation based on data sourced from the external system.

Dependencies: Create sites and contacts after creating the supplier. For more information, see [Dependencies](#).

Endpoint address: /services/rest/supplier **HTTP method:** POST

Request Details

The body of the request contains a SupplierFullDTO to specify the detail of the supplier to create. Compared to retrieving a supplier (which uses the same SupplierFullDTO type), this request is much shorter. Only the attributes that are to be populated on the created Supplier record need to be included. As a minimum, this must include the fields shown in the following table.

Supplier Mandatory Fields

Field Name	Element Name
Supplier Name	name
Contact Name	supplierContactName
Email	email
Supplier Type	supplierType/code
Lead Business Unit	businessUnit/code
Billing Code	billingCode/code
Status	Status
Supplier Code Confirmed Flag	supplierCodeConfirmed
Deleted Record	deleted (Set to false)
Created On	createdOn (set to current date using format YYYY-MM-DD or to current date and time using format YYYY-MM-DDThh:mm:ss)
Is Active	isActive - see below
Potential Supplier	potentialSupplier (Set to false)

Example Request XML

This example shows the minimum requirement to be able to create a supplier.

```

<ns1:supplierFullDTO
  xmlns:ns2="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
  xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/full">
  <ns1:billingCode>
    <ns2:code>SMALL</ns2:code>
  </ns1:billingCode>
  <ns1:businessUnit>
    <ns2:code>UK</ns2:code>
  </ns1:businessUnit>
  <ns1:email>supplier.contactemail@supplier.co.uk</ns1:email>
  <ns1:name>Name of Supplier</ns1:name>
  <ns1:status>AWAITING REGISTRATION</ns1:status>
  <ns1:supplierContactName>Supplier Contact Name</ns1:supplierContactName>
  <ns1:supplierCodeConfirmed>false</ns1:supplierCodeConfirmed>
  <ns1:supplierType>
    <ns2:code>SUPPLIER_TYPE</ns2:code>
  </ns1:supplierType>
  <ns1:deleted>false</ns1:deleted>
  <ns1:createdOn>2020-09-29T09:00:00</ns1:createdOn>
  <ns1:isActive>false</ns1:isActive>
  <ns1:potentialSupplier>false</ns1:potentialSupplier>
</ns1:supplierFullDTO>

```

Set the deleted attribute to false to ensure the record can be seen in the UI. If omitted or set to true, it will exist, but will not be visible.

The Supplier API only creates the Supplier record, just as the Site API only creates the Site record. The same goes for the User API; it only creates the User record, and similarly the Contact API only creates the Contacts records. Each Supplier record must also have at least one Site record created for it.

Supplier Contacts and their User records must be created using the respective APIs, and linked to the supplier/site with their record IDs. When a new Supplier is created through the API, the process does not automatically create the initial Supplier user.

Use of the `isActive` and `supplierCodeConfirmed` Flags

The `isActive` and `supplierCodeConfirmed` flags on the Supplier record are used to control the Registration process within the application UI.

`isActive` is a Boolean TRUE/FALSE value. It is always FALSE when a Supplier is created in the application UI. If set to FALSE in the API, when the supplier user first logs in to the system, the Supplier Registration wizard runs (as it does if the Supplier has been created through the application UI). Successful completion of the Registration wizard sets the `isActive` flag to TRUE, thus ensuring it only runs once. Therefore, if set to TRUE in the API, when the Supplier user first logs in to the system, the Registration wizard will not run.

`supplierCodeConfirmed` is a Boolean TRUE/FALSE value. It is always FALSE when a Supplier is created through the application UI. It is a security measure for the supplier user logging in for the first time, where they are prompted to enter their Supplier Code. The application changes the value to TRUE if there is a successful match.

Setting both `isActive` and `supplierCodeConfirmed` as FALSE will trigger the Registration wizard process when the supplier user first logs in, prompting for the entry of the Supplier Code.

Setting both `isActive` and `supplierCodeConfirmed` as TRUE indicates that the information captured during the Registration wizard process has effectively been provided, and therefore the wizard will not run.

Setting `isActive` as FALSE and `supplierCodeConfirmed` as TRUE indicates that part of the information captured during the Registration wizard process is present, but the wizard will run to allow the user to complete the data entry.

If a Supplier is created with a status of AWAITING AUTHORISATION, but with `isActive` set as TRUE, the Supplier status is changed to REGISTERED, thus bypassing the Registration process.



Note:

When a new supplier is created manually within the application, the system sends an email to the supplier requesting they complete the registration process. When creating a new supplier through the API, the email is not generated.

Where the record is linked to another record, such as the Supplier Type in this case, the business key must be provided (not the description) in order to form the link between the records. In general, the business key will be the code attribute of the linked record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a `SupplierLink` root element. The root element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Newly created Supplier record's internal ID
recordLink	string	URI to the newly created Supplier record, for use in a GET request
code	string	Supplier code business key to the newly created Supplier record
name	string	Supplier's name
localName	string	Supplier's name in business language (if used)

Error Messages

If the supplied data does not result in a valid Supplier (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Element	Message	Meaning
billingCode	ERROR: class <<hostname>>.Supplier.billingCode - The condition is invalid	No tag provided
billingCode	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>.BillingCode using code:null	No <ns1:code> tag
billingCode	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>..BillingCode using code:<###>	Invalid <ns1:code>
businessUnit	ERROR: class <<hostname>>..Supplier.businessUnit - The condition is invalid	No tag provided
businessUnit	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>..domain.BusinessUnit using code:null	No <ns1:code> tag
businessUnit	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>..BusinessUnit using code:<###>	Invalid <ns1:code>
code	ERROR: class com.micros.creations.core.domain.Supplier.code - ??Another object exists with this value??	Supplier code already exists
email	ERROR: class com.micros.creations.core.domain.Supplier.email - The condition is invalid	No tag provided
email	ERROR: class <<hostname>>.Supplier.email - The attribute is invalid	Malformed email address
supplierContactName	ERROR: class <<hostname>>.Supplier.supplierContactName - The condition is invalid	No tag provided

Element	Message	Meaning
supplierContactName	ERROR: class <<hostname>>.Supplier.supplierContactName - The condition is invalid	No value provided
name (supplier name)	ERROR: class <<hostname>>.Supplier.name - The condition is invalid	No tag provided
name (supplier name)	ERROR: class <<hostname>>.Supplier.name - The condition is invalid	No value provided
supplierType	ERROR: class <<hostname>>.Supplier.supplierType - The condition is invalid	No tag provided
supplierType	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>.SupplierType using code:null	No <ns1:code> tag
supplierType	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>.SupplierType using code:<###>	Invalid <ns1:code>

Update Record

Description

Updates an existing Supplier record. Use this function to update a supplier's details in Supplier Evaluation based on data sourced from the external system.

Endpoint address: /services/rest/supplier/{id} **HTTP method:** PUT

Request Details

The body of the request contains a SupplierUpdateDTO to specify the updates to the Supplier record. Compared to retrieving a supplier (which uses the SupplierFullDTO type), this request is much shorter. As a minimum, the values specified as mandatory for the Create Record function (see above) must be included.

The request content is similar to that for creating a supplier, but crucially, the links to other top-level records (Sites, SiteContact, SupplierContact) are omitted. The omission of those ensures that when updating a supplier, only the supplier details need to be specified, and not the details for the related records that may not require updating (and which should be updated to calls to their respective services). After the call, the Supplier record is updated to match the request.



Note:

When updating records, all values must be included. If a value or element is omitted from the request, the field contents will be cleared on the Supplier record (except for the Sites, SiteContacts, and SupplierContacts).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a SupplierLink element. The SupplierLink element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Supplier record's internal ID
recordLink	string	URI to the Supplier record, for use in a GET request
code	string	Supplier code business key to the Supplier record
name	string	Supplier's name
localName	string	Supplier's name in business language (if used)

Error Messages

If the supplied data does not result in a valid Supplier (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Element	Message	Meaning
billingCode	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>.BillingCode using code:<####>	Invalid <ns1:code>
businessUnit	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>.domain.BusinessUnit using code:<####>	Invalid <ns1:code>
code	MySQLIntegrityConstraintViolationException: Duplicate entry <####> for key 'c_code'	Supplier code provided that already exists for another supplier
name	PropertyValueException: not-null property references a null or transient value: <<hostname>>.domain.Supplier.name	No tag provided
status	NullPointerException:	No status tag provided
status	IllegalArgumentException: Empty string is not allowed, should be null on method: <<hostname>>.domain.Supplier@1df35f3[id=<####>,code=<####>]	No status code provided

SiteRestService

This section describes the API for managing sites. The following functions are available:

- [List of Values](#): retrieves a list of sites
- [Retrieve Record by ID](#): retrieves a Site record using its unique identifier
- [Retrieve Record by Business Key](#): retrieves a Site record using its business key

- [Check Record Modification Timestamp](#): retrieves the timestamp when a Site record was last updated
- [Create Record](#): creates a new Site record
- [Update Record](#): updates an existing Site record

**Note:**

When integrated with Merchandising, using the SiteRestAPI to make changes that affect the Billing Details table can have an impact on the integration.

List of Values

Description

Retrieves a list of sites in a paged list. Use this function to retrieve a simple list of supplier names and IDs, or to locate Site record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/site **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example	Validation
offset	Optional	int	No	No	60	
pageSize	Optional	int	No	No	30	
siteStatus	Optional	string	Yes	No	AWAITING REGISTRATIO N	
siteCode	Optional	string	Yes	Yes	A0001-0001	
siteName	Optional	string	Yes	Yes	West Road Site	
siteType	Optional	string	Yes	Yes	CANNERY	
country	Optional	string	Yes	Yes	UK	

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example	Validation
businessCategory	Optional	string	Yes	Yes	CHEESE_DAIRY	Code in the Business Categories glossary at any level of the hierarchy. Applies the filter to Business Category and Lead Business Category (if used). Returns a result if any of the values match any of the specification's business categories.
businessUnit	Optional	string	Yes	Yes	UK	
supplierName	Optional	string	Yes	Yes	ABC Ltd	
supplierActive	Optional	String	No	No	true (otherwise will return active and inactive suppliers)	
modifiedSince	Optional	string	No	No	2015-05-19 13:30:39	
modifiedUntil	Optional	string	No	No	2015-05-19 13:30:39	
statusChangedFrom	Optional	string	No	No	2015-06-10 09:00:00	If From and Until are both used, then the Until date must be greater than or equal to the From date.
statusChangedUntil	Optional	string	No	No	2015-06-10 09:00:00	
statusChangedTo	Optional	string	Yes	No	ACTIVE	
leadTechnologist	Optional	string	Yes	Yes	John Smith	The user's Login Id.

Example URLs

.../services/rest/site/?offset=2&pageSize=20 .../services/rest/site/?siteStatus=ACTIVE .../services/rest/site/?supplierName=API%

Response Details

For a successful response, XML is returned with a SiteLinkList root element containing an entries element for each matched site. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Site record's internal ID
recordLink	string	URI to the SiteRestService Retrieve record for this site
code	string	Site code business key to the Site record
name	string	Site's name
localName	string	Site's name in business language (if used)

The returned XML also contains a totalRecords element, which states the total number of retrievable records that match the filter parameters.

Error Messages

Element	Message	Meaning
siteStatus	HTTP 417 IllegalStateException: The Supplier Status <####> does not exist	Invalid site status
siteStatus	HTTP 417 IllegalStateException: The Supplier Statuses <####~####~####> do not exist	Invalid site status
siteType	HTTP 417 IllegalStateException: The Supplier Type <####> does not exist	Invalid supplier type
siteType	HTTP 417 IllegalStateException: The Supplier Types <####~####> do not exist	Invalid supplier type
country	HTTP 417 IllegalStateException: The Country <####> does not exist	Invalid country
country	HTTP 417 IllegalStateException: The Countries <####~####> do not exist	Invalid country
businessCategory	HTTP 417 IllegalStateException: The Business Category <####> does not exist	Invalid business category
businessCategory	HTTP 417 IllegalStateException: The Business Categories <####~####> do not exist	Invalid business category
businessUnit	HTTP 417 IllegalStateException: The Business Unit <####> does not exist	Invalid business unit
businessUnit	HTTP 417 IllegalStateException: The Business Units <####~####> do not exist	Invalid business unit
modifiedSince	Invalid dates cause stack trace to come back to "response"	
modifiedUntil	Invalid dates cause stack trace to come back to "response"	
pageSize	Error Code: INVALIDRESTSERVICEPAGESIZE INVALIDPAGESIZE	Page Size must be between 1 and 100
offset	Error Code: INVALIDOFFSET	Offset must be a positive integer
siteStatus	Error Code: INVALIDSITESTATUS INVALIDSITESTATUSSES	Invalid Site Status
siteType	Error Code: INVALIDSITETYPE	Invalid Site Type

Element	Message	Meaning
country	Error Code: INVALIDCOUNTRY	Invalid Country
businessCategory	Error Code: INVALIDBUSINESSCATEGORY	Invalid Business Category
businessUnit	Error Code: INVALIDBUSINESSUNIT	Invalid Business Unit
supplierActive	Error Code: INVALIDBOOLEAN	Invalid supplierActive value Permitted values (not case-sensitive) are: true; yes; 1; false; no; 0
leadTechnologist	Error Code: INVALIDUSER	Invalid Lead Technologist
statusChangedFrom	Error Code: INVALIDDATEFORMAT	Invalid Status Changed From Must be a valid date/time format: Year-Month-DayOfMonth Hour:Minute:Seconds such as 2016-12-30 09:59:59
statusChangedUntil	Error Code: INVALIDDATEFORMAT	Invalid Status Changed Until Must be a valid date/time format: Year-Month-DayOfMonth Hour:Minute:Seconds such as 2016-12-30 09:59:59
statusChangedTo	Error Code: INVALIDSTATUSCHANGEDTO	Either Status Changed From or Status Changed Until must be specified if filtering on Status Changed To

Retrieve Record by ID

Description

Retrieves a Site record's details using the record's internal unique ID (which is not visible in the UI). Use this function to retrieve the full details of an individual site.

Endpoint address: /services/rest/site/{id} **HTTP method:** GET

Request Details

The URL contains the {id} parameter that determines the record to retrieve.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a localeData element returned for the specified locales if data is present.

Example URL

.../services/rest/site/87

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a siteFullDTO root element containing the individual attributes of the requested Site record. If an ID is not specified, a list of all sites is returned (per the List of Values function).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Site to return with id:<###>	Invalid {id} - not found

Retrieve Record by Business Key

Description

Retrieves a single Site record's details using its business key (supplier and codes). Use this function to retrieve the full details of an individual site using the combination of the Supplier Evaluation supplier and site codes.

Endpoint address: /services/rest/site/byKey/{supplierCode}/{siteCode} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {supplierCode} and {siteCode} parameters that determine the record to retrieve.

Example URL

.../services/rest/site/byKey/A0001/A0001-0001

Response Details

If successful, an HTTP 301 redirect response is sent containing a header Location with the URI of the document requested.

The returned URI represents the call to be made to the Retrieve Record by ID function (with the id substituted with the relevant value), which will return a siteFullDTO root element containing the individual attributes of the requested Site record.

**Note:**

If your method of calling the API is configured to automatically follow HTTP 301 redirects, the call may automatically return a siteFullDTO root element containing the individual attributes of the requested Site record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
supplierCode	HTTP 404 Not Found	Invalid {supplierCode} - blank
siteCode	HTTP 404 Not Found	Invalid {siteCode}

Check Record Modification Timestamp

Description

Retrieves the last modification date and time of a Site record. Use this function to determine when a site's details were last updated.

Endpoint address: /services/rest/site/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/site/87

Response Details

If successful, an HTTP 200 response is sent containing the Last-Modified header showing the date and time of the last update of the requested Site record.

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Messages

Element	Message	Meaning
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 Expectation Failed	Invalid {id} - not found

Create Record

Description

Creates a new Site record. Use this function to create new sites in Supplier Evaluation based on data sourced from the external system.

Dependencies: The Supplier must be present in the application and its record ID obtained. If assigning a Business Category or Technologist, the record must be present in the application and its record ID obtained. Create Contacts after creating the Site. For more information, see [Dependencies](#).

Endpoint address: /services/rest/site **HTTP method:** POST

Request Details

The body of the request contains a SiteFullDTO element to specifying the details of the site to create. Compared to retrieving a site (which uses the same SiteFullDTO type), this request is much shorter. Only the attributes that are to be populated on the created Site record need to be included. As a minimum, this must include the fields shown in the following table.

Site Mandatory Fields

Field Name	Element Name
Site Name	name
Site Type	siteType/code
Categories	businessCategories
Lead Technologist	leadTechnicalManager
Supplier	supplier/code
Site Status	siteStatus
Deleted Record	deleted (Set to false)

Example Request XML

This example shows the minimum requirement to be able to create a site against a supplier.

```
<ns1:siteFullDTO
  xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
  xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">

  <ns0:businessCategories>
    <ns1:code>CATEGORY2A</ns1:code>
  </ns0:businessCategories>
  <ns0:leadTechnicalManager>
    <ns1:code>techadmin</ns1:code>
  </ns0:leadTechnicalManager>

  <ns0:name>Site Name</ns0:name>
  <ns0:siteType>
    <ns1:code>SITE_TYPE_EXAMPLE</ns1:code>
  </ns0:siteType>
  <ns0:siteStatus>
    <ns1:status>ACTIVE</ns1:status>
  </ns0:siteStatus>
  <ns0:supplier>
```

```

        <ns1:code>WS0001</ns1:code>
      </ns0:supplier>
      <ns0:deleted>>false</ns0:deleted>

</ns1:siteFullDTO>

```

Set the deleted attribute to false to ensure the record can be seen in the UI. If omitted or set to true, it will exist, but will not be visible.

Where the record is linked to another record, such as the Site Type in this case, the business key must be provided (not the description) in order to form the link between the records. In general, the business key will be the code attribute of the linked record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a SiteLink root element containing the site data, and a SupplierLink element, consisting of the parent supplier data. The returned elements are shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Newly created Site record's internal ID
recordLink	string	URI to the newly created Site record, for use in a GET request
code	string	Supplier and site code business key to the newly created Site record
name	string	Site's name
localName	string	Site's name in business language (if used)
supplierLink/recordId	long	Newly created site's Supplier record internal ID
supplierLink/recordLink	string	URI to the newly created site's Supplier record, for use in a GET request
recordLink/code	string	Supplier code business key to the newly created site's Supplier record
recordLink/name	string	Supplier's name
recordLink/localName	string	Supplier's name in business language (if used)

Error Messages

If the supplied data does not result in a valid Site (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Element	Message	Meaning
supplier/id	Null Pointer Exception	Supplier ID not provided
supplier/id	No row with the given identifier exists: [com.micros.creations.core.domain.Suppli er#<###>]	Supplier not found

Element	Message	Meaning
siteStatus	Null Pointer Exception	Site Status not provided
siteStatus	Cannot locate keyword for class com.micros.creations.core.domain.SiteStatus using status:<###>	Site Status not found
siteType	ERROR: class com.micros.creations.core.domain.Site.siteType - The condition is invalid	Site Type not provided
siteType	Cannot locate keyword for class com.micros.creations.core.domain.SiteType using code:<###>	Site Type not found
name	class com.micros.creations.core.domain.Site.name - The condition is invalid	Site name not provided
leadTechnicalManager	ERROR: class com.micros.creations.core.domain.Site.leadTechnicalManager - The condition is invalid	Lead Technical Manager not provided
leadTechnicalManager	HTTP 417 IllegalStateException: Cannot locate keyword for class com.micros.creations.core.domain.User using code:<###>	Lead Technical Manager not found
businessCategories	ERROR: class com.micros.creations.core.domain.Site.businessCategories - The condition is invalid	Business Category not provided
businessCategories	Cannot locate keyword for class com.micros.creations.core.domain.BusinessCategory using code:<###>	Business Category not found
code	Duplicate site code	The site already exists for this supplier

Update Record

Description

Updates an existing Site record. Use this function to update a site's details in Supplier Evaluation based on data sourced from the external system.

Endpoint address: /services/rest/site/{id} **HTTP method:** PUT

Request Details

The body of the request contains a SiteUpdatedDTO to specify the updates to the Site record. Compared to retrieving a site (which uses the SiteFullDTO type), this request is much shorter. As a minimum, the values specified as mandatory for the Create Record function (see above) must be included.

The request content is similar to that for creating a site, but crucially, the links to other top-level records (SiteContact) are omitted. The omission of those ensures that when updating a site, only the site details need to be specified, and not the details for the related records that may not require updating (and which should be updated with calls to their respective services). After the call, the Site record is updated to match the request

**Note:**

When updating records, all values must be included. If a value or element is omitted from the request, the field contents will be cleared on the Site record (except for the SiteContacts).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a SiteLink root element containing the site data, and a SupplierLink element, consisting of the parent supplier data. The returned elements are shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Updated Site record's internal ID
recordLink	string	URI to the updated Site record, for use in a GET request
code	string	Site code business key to the updated Site record
name	string	Site's name
localName	string	Site's name in business language (if used)
supplierLink/recordId	long	Updated site's Supplier record internal ID
supplierLink/recordLink	string	URI to the updated site's Supplier record, for use in a GET request
recordLink/code	string	Supplier code business key to the updated site's Supplier record
recordLink/name	string	Supplier's name
recordLink/localName	string	Supplier's name in business language (if used)

Error Responses

If the supplied data does not result in a valid Site (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

ContactRestService

This section describes the API for managing supplier and site contacts. The following functions are available:

- [List of Values](#): retrieves a list of contacts
- [Retrieve Record by ID](#): retrieves a Contact record using its unique identifier
- [Retrieve Supplier Contact Record by Business Key](#): retrieves a Supplier Contact record using its business key

- [Retrieve Site Contact Record by Business Key](#): retrieves a Site Contact record using its business key
- [Check Record Modification Timestamp](#): retrieves the timestamp when a Contact record was last updated
- [Create Record](#): creates a new Contact record
- [Update Record](#): updates an existing Contact record

List of Values

Description

Retrieves a list of contacts in a paged list. Use this function to retrieve a simple list of contact names and IDs, or to locate Contact record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/contact **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30
name	Optional	string	Yes	Yes	John Smith
supplierContactRoles	Optional	string	Yes	Yes	HEAD OF MANUFACTURING
siteContactRoles	Optional	string	Yes	Yes	HEAD OF MANUFACTURING
country	Optional	string	Yes	Yes	UK
selectedSites	Optional	string	No	No	A0001-0001
supplierCode	Optional	string	Yes	Yes	A0001
supplierName	Optional	string	Yes	Yes	ABC Ltd.
siteStatus	Optional	string	Yes	No	ACTIVE~INACTIVE
supplierActive	Optional	string	No	No	true returns active contacts no value returns both active and inactive contacts
modifiedSince	Optional	string	No	No	2015-05-19 13:30:39
modifiedUntil	Optional	string	No	No	2015-05-19 13:30:39

Example URLs

.../services/rest/contact/?offset=2&pageSize=20 .../services/rest/contact/?siteStatus=ACTIVE
.../services/rest/contact/?supplierName=API%

Response Details

For a successful response, XML is returned with a ContactLinkList root element containing an entries element for each matched contact. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Contact record's internal ID
recordLink	string	URI to the ContactRestService Retrieve record service for this contact
email	string	Contact's email address
name	string	Contact's name
siteContact	Boolean	Indicates if a contact for the site (true), else false
supplierContact	Boolean	Indicates if a contact for the supplier (true), else false

The returned XML also contains a totalRecords element, which states the total number of retrievable records that match the filter parameters.

Error Messages

Element	Message	Meaning
supplierContactRoles	HTTP 417 IllegalStateException: The Supplier Contact Role <###> does not exist	Invalid supplier contact role
supplierContactRoles	HTTP 417 IllegalStateException: The Supplier Contact Roles <###~###> do not exist	Invalid supplier contact role
siteContactRoles	HTTP 417 IllegalStateException: The Site Contact Role <###> does not exist	Invalid site contact role
siteContactRoles	HTTP 417 IllegalStateException: The Site Contact Roles <###~###> do not exist	Invalid site contact role
country	HTTP 417 IllegalStateException: The Country <###> does not exist	Invalid country code
country	HTTP 417 IllegalStateException: The Countries <###~###> do not exist	Invalid country code
siteStatus	HTTP 417 IllegalStateException: The Supplier Status <###> does not exist	Invalid supplier status code
pageSize	Error Code: INVALIDRESTSERVICEPAGESIZE INVALIDPAGESIZE	Page Size must be between 1 and 100
offset	Error Code: INVALIDOFFSET	Offset must be a positive integer
supplierContactRole	Error Code: INVALIDCONTACTROLE	Invalid Supplier Contact Role
siteContactRole	Error Code: INVALIDCONTACTROLE	Invalid Site Contact Role
country	Error Code: INVALIDCOUNTRY	Invalid Country
selectedSite	Error Code: INVALIDSITE	Invalid Site
supplierCode	Error Code: INVALIDSUPPLIER	Invalid Supplier

Element	Message	Meaning
siteStatus	Error Code: INVALIDSITESTATUS	Invalid Site Status
supplierActive	Error Code: INVALIDBOOLEAN	Invalid supplierActive value Permitted values (not case-sensitive) are: true; yes; 1; false; no; 0

Retrieve Record by ID

Description

Retrieves a single Contact record's details using the record's unique ID. Use this function to retrieve the full details of an individual supplier or site contact.

Endpoint address: /services/rest/contact/{id} **HTTP method:** GET

Request Details

The URL contains the {id} parameter that determines the record to retrieve.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a localeData element returned for the specified locales if data is present.

Example URL

.../services/rest/contact/405

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a contactAndPersonDTO root element containing the individual attributes of the requested Contact record and the person it associates to.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Contact to return with id:<###>	Invalid {id} - not found

Retrieve Supplier Contact Record by Business Key

Description

Retrieves a single Supplier Contact record's details using its business key (supplier code, contact name, and email address). Use this function to retrieve the full details of an individual supplier contact using the combination of the Supplier Evaluation supplier codes, name of the contact, and the contact's email address.

Endpoint address: /services/rest/contact/byKey/supplier/{supplierCode}/{name}/{email} **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Example
supplierCode	Mandatory	string	No	A0001
name	Mandatory	string	No	John Doe
email	Mandatory	string	No	john.doe@email.com

Example URL

.../services/rest/contact/byKey/supplier/A0001/John Doe/john.doe@email.com

Response Details

If successful, an HTTP 301 redirect response is sent containing a header Location with the URI of the document requested.

The returned URI represents the call to be made to the Retrieve Record by ID function (with the id substituted with the relevant value), which will return a contactAndPersonDTO root element containing the individual attributes of the requested Contact record.



Note:

If your method of calling the API is configured to automatically follow HTTP 301 redirects, the call may automatically return a contactAndPersonDTO root element containing the individual attributes of the requested Contact record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Responses

In the document cannot be located, an HTTP 404 response is sent.

Retrieve Site Contact Record by Business Key

Description

Retrieves a single Site Contact record's details using its business key (supplier code, site code, contact name, and contact email). Use this function to retrieve the full details of an individual site contact using the combination of the Supplier Evaluation supplier and site codes, name of the contact, and the contact's email address.

Endpoint address: /services/rest/contact/byKey/site/{supplierCode}/{siteCode}/{name}/{email}
HTTP method: GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Example
supplierCode	Mandatory	string	No	A0001
siteCode	Mandatory	string	No	A0001-0001
name	Mandatory	string	No	John Doe
email	Mandatory	string	No	john.doe@email.com

Example URL

.../services/rest/contact/byKey/site/A0001/A0001-0001/ John Doe/john.doe@email.com

Response Details

For a successful response, XML is returned with a contactAndPersonDTO root element containing the individual attributes of the requested Contact record and the person it associates to.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Responses

In the document cannot be located, an HTTP 404 response is sent.

Check Record Modification Timestamp

Description

Retrieves the last modification date and time of a Contact record. Use this function to determine when a supplier or site contact's details were last updated.

Endpoint address: /services/rest/contact/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/contact/405

Response Details

If successful, an HTTP 200 response is sent containing the Last-Modified header to show the last modification date and time of the last update of the requested Contact record.

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Messages

Element	Message	Meaning
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 Expectation Failed	Invalid {id} - not found

Create Record

Description

Creates a new Contact record (and the associated Person record). Use this function to create new supplier or site contacts in Supplier Evaluation based on data sourced from the external system.

Dependencies: The Supplier (and Site, if a site contact) must be present in the application and its record ID obtained. Omit the Site element if not a site contact. The User/Person must be present in the application and its record ID obtained. For more information, see [Dependencies](#).

Endpoint address: /services/rest/contact **HTTP method:** POST

Request Details

The body of the request contains a ContactAndPersonDTO to specify the details of the contact to create. Compared to the retrieving a contact (which uses the same ContactAndPersonDTO type), this request is much shorter. Only the attributes that are to be populated on the created Contact record need to be included. As a minimum, this must include the fields shown in the following table.

Contact Mandatory Fields

Field Name	Element Name
Name	contactFullDTO/person/name personFullDTO/name

Field Name	Element Name
User Id	contactFullDTO/person/id
Supplier	contactFullDTO/person/supplier/id contactFullDTO/company/code personFullDTO/supplier/code personFullDTO/supplier/id
Email	contactFullDTO/person/email personFullDTO/email
Phone	contactFullDTO/contactDetails/phoneNumber
Supplier Contact Roles	contactFullDTO/supplierContactRole/code
Contact Type	contactFullDTO/dtype
Site Contact Flag	contactFullDTO/siteContact
Supplier Contact Flag	contactFullDTO/supplierContact
Deleted Record	deleted (Set to false)

Example Request XML

This example shows the minimum requirement to be able to create a contact against a supplier.

```
<ns0:ContactAndPersonDTO xmlns:ns4="http://www.oracle.com/orsecs/service/rest/model"
xmlns:ns0="http://www.micros.com/creations/core/domain/dto/vlp0/full" xmlns:ns1="http://
www.micros.com/creations/core/domain/dto/vlp0/simple">
  <ns0:contactFullDTO>
    <ns0:contactDetails>
      <ns0:phoneNumber>01234 56789</ns0:phoneNumber>
    </ns0:contactDetails>
    <ns0:dtype>SupplierContact</ns0:dtype>
    <ns0:person>
      <ns1:email>jphillips@example.com</ns1:email>
      <ns1:name>Jane Phillips</ns1:name>
      <ns1:supplier>
        <ns1:code>A0001</ns1:code>
      </ns1:supplier>
    </ns0:person>
    <ns0:siteContact>false</ns0:siteContact>
    <ns0:supplierContact>true</ns0:supplierContact>
    <ns0:supplierContactRole>
      <ns1:code>ACCOUNT MANAGER</ns1:code>
    </ns0:supplierContactRole>
    <ns0:company>
      <ns1:code>A0001</ns1:code>
    </ns0:company>
  </ns0:contactFullDTO>
  <ns0:personFullDTO>
    <ns0:email>jphillips@example.com</ns0:email>
    <ns0:name>Jane Phillips</ns0:name>
    <ns0:supplier>
      <ns1:code>A0001</ns1:code>
    </ns0:supplier>
  </ns0:personFullDTO>
  <ns0:deleted>false</ns0:deleted>
</ns0:ContactAndPersonDTO>
```

Set the deleted attribute to false to ensure the record can be seen in the UI. If omitted or set to true, it will exist, but will not be visible.

If a contact has multiple supplier or site contact roles, they should be separate entries as follows:

```
<ns0:supplierContactRole>
  <ns1:code>ACCOUNT MANAGER</ns1:code>
</ns0:supplierContactRole>
<ns0:supplierContactRole>
  <ns1:code>AUDITS AND VISITS CONTACT</ns1:code>
</ns0:supplierContactRole>
<ns0:supplierContactRole>
  <ns1:code>EMERGENCY CONTACT</ns1:code>
</ns0:supplierContactRole>
```

Where the record is linked to another record, such as the Contact Role in this case, the business key must be provided (not the description) in order to form the link between the records. In general, the business key will be the code attribute of the linked record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a ContactLink root element. The root element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Newly created Contact record's internal ID
recordLink	string	URI to the newly created Contact record, for use in a GET request
email	string	Email address of the newly created contact
name	string	Contact's name
siteContact	Boolean	Indicates if a contact for the site (true), else false
supplierContact	Boolean	Indicates if a contact for the supplier (true), else false

Error Messages

Element	Message	Meaning
phoneNumber	phoneNumber - The condition is invalid	Missing tag
dtype	IllegalArgumentException: Invalid value for dtype:<###>	Invalid value SupplierContact; SiteContact
name & email	HTTP 417 IllegalStateException: Cannot locate Person record with name: <###> and email: <###>	A valid user must be found with matching name and email address
company	HTTP 417 IllegalStateException: Cannot locate keyword for class com.micros.creations.core.domain.Company using code:<###>	Invalid company/supplier code

Update Record

Description

Updates an existing Contact record. Use this function to update a supplier or site contact's details in Supplier Evaluation based on data sourced from the external system.

Endpoint address: /services/rest/contact/{id} **HTTP method:** PUT

Request Details

The body of the request contains a ContactAndPersonDTO element to specify the updates to the Contact record. As a minimum, the values specified as mandatory for the Create Record function (see above) must be included.

The request content is the same as that for creating a contact. After the call, the Contact record is updated to match the request.



Note:

When updating records, all values must be included. If a value or element is omitted from the request, the field contents will be cleared on the Contact record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a ContactLink element. The ContactLink element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Contact record's internal ID
recordLink	string	URI to the Contact record, for use in a GET request
email	string	Contact's email address
name	string	Contact's name
siteContact	Boolean	Indicates if a contact for the site (true), else false
supplierContact	Boolean	Indicates if a contact for the supplier (true), else false

Error Messages

If the supplied data does not result in a valid Contact (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

TaskRestService

This section describes the API for managing user tasks. The following function is available:

- [List of Tasks](#): Retrieves a list of user tasks

List of Tasks

Description

Retrieves a list of tasks for a user in the given language. Use this function to retrieve a list of a specific user's entries in their Supplier Evaluation Task Manager app. Parameters are used to specify the name of the user, and the language of the returned task details.

Endpoint address: /services/rest/task **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
userId	Mandatory	string	No	No	John Smith
language	Mandatory	string	No	No	en_GB

Example URL

.../services/task/?language=en_GB

The userId parameter is the login ID of the user for which the tasks list is to be retrieved. The language parameter is the code of the language record/locale in which to retrieve the task details.

Response Details

For a successful response, XML is returned with a TaskDTOList root element containing a tasks element for each matched task. The root element consists of the elements in the following table.

Returned Elements

Element	Type	Description
message	string	Name of the task, in the language specified
messageId	string	A language-agnostic identifier for the task
myCreationsLink	string	A URI to the Supplier Evaluation system, which will open the list of items for that task
taskItemCount	int	The number of items in the task

Error Messages

Element	Message	Meaning
userId	userId required	User not provided
userId	Cannot find user with login id <###>	User not found
language	language required	Language not provided
language	Invalid locale format: <###>	Language not found

UrgentItemsRestService

This section describes the API for retrieving a count of a user's urgent items. The following function is available:

- [Number of Urgent Items](#): retrieves the number of urgent items pending for a user

Number of Urgent Items

Description

Retrieves a count of the number of Urgent Item tasks for a user. Use this function to determine the number of pending urgent tasks a specific user has in their Supplier Evaluation Urgent Item Manager app. A parameter is passed to specify the name of the user.

Endpoint address: /services/rest/urgentItems **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
userId	Mandatory	string	No	No	John Smith

The userId parameter is the login ID of the user for which the number of pending urgent items are to be retrieved.

Response Details

For a successful response, XML is returned with an UrgentItemsModel root element containing an itemCount element that specifies the number of Urgent Items.

Error Messages

Element	Message	Meaning
userId	User not provided: userId required	User not provided
userId	Cannot find user with login id <###>	User not found

ArtworkRestService

This section describes the API for Artwork integration. The following functions are available:

- **Started Activities:** retrieves a list of Artwork Activities that have started
- **Update Record:** updates existing Artwork Activities

Started Activities

Description

Retrieves a list of Project Activities that have their useMyArtwork flag set, and their status has changed to Started within the specified date range. This function is used for the integration of Supplier Evaluation with an external Artwork Management system.

Endpoint address: /services/rest/artwork/started **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30
fromDate	Mandatory	string	No	No	2014-12-30 23:59:59
toDate	Mandatory	string	No	No	2015-12-30 23:59:59

Response Details

For a successful response, XML is returned with a CreateArtworkRequestList root element containing an entries element for each matching Artwork project activity. A totalRecords element identifies the number of records returned. Separate activityDetails and project elements group the returned activity and project data. The elements consist of the elements in the following tables.

Returned Elements - activityDetails

Element	Type	Description
activityName	string	Project activity's title
activityRecordId	long	Project activity record's internal ID
actualStartDate	string	Actual start date of the project activity
checkpointOnly	Boolean	Is the project activity a checkpoint only type
criticalPath	Boolean	Is the project activity a critical path type
duration	int	Duration of the project activity

Element	Type	Description
isAutoGate	string	Project's type of auto gate
isGate	Boolean	Is the project activity a gate type
isKey	Boolean	Is the project activity a key type
projectId	long	Project record's internal ID
projectTitle	string	Project's title
proposedEndDate	string	Proposed end date of the project activity
proposedStartDate	string	Proposed start date of the project activity
responsibleUserRoles	string	Role names of the responsible users
sequenceNumber	string	Project activity's sequence number
statusCode	string	Code of the project activity's status
subStatusCode	string	Code of the project activity's sub-status

Returned Elements - projects

Element	Type	Description
brand	string	Name of the project's brand
categories	string	Names of the project's categories (the lowest level of the selected category - may be multiples)
masterProject	Boolean	If the project is a master type
newCategoryComments	string	Category comments
packaging	string	Packaging details
productRangeInformation	string	Project's product range details
projectConcept	string	Project's business key ID
projectId	string	Project's business key ID
projectManager	string	Name of the project's project manager
projectRecordId	long	Project record's internal ID
projectTitle	string	Project's title
projectType	string	Type of project
proposedProducts	string	Project's proposed products
specType	string	Type of project's product specification
status	string	Status of the project
subBrand	string	Name of the project's sub-brand
supplier	string	Name of the supplier associated to the project
site	string	Name of the site associated with the project
targetConsumer	string	Project's target consumer details
teamDetails/role	string	Name of the team role
teamDetails/users	string	Name of the team users
templateFolder	string	Name of the project template folder

Element	Type	Description
templateType	string	Type of the project's template
templateUsed	string	Name of the project's template

Error Messages

Element	Message	Meaning
fromDate	fromDate parameter required	From date not provided
toDate	toDate parameter required	To date not provided
fromDate	fromDate and toDate parameters are required	From and To dates not provided
toDate	From date must be before or equal to To date	From date is after To date
fromDate	HTTP 417 <errorMessage>	Not a valid date, whereby <errorMessage> will identify the issue For example: IllegalInstantException: Cannot parse "2018-03-25 01:00:00": Illegal instant due to time zone offset transition (Europe/London) In this example, due to the clocks In this example, due to the clocks moving forward (in the server's time-zone), the passed in date is invalid
toDate	HTTP 417 <errorMessage>	Not a valid date, whereby <errorMessage> will identify the issue (see example above)

Update Record

Description

Updates the sub-status of existing Artwork Project Activities. This function is used for the integration of Supplier Evaluation with an external Artwork Management system, to update the status of an Artwork activity from the external system.

Endpoint address: /services/rest/artwork/update **HTTP method:** POST

Request Details

The body of the request contains an activityUpdateRequest element consisting of the elements shown in the following table.

Request Elements

Element	Type	Description
activityName	string	Project activity's name
activityRecordId	long	Project activity record's internal ID
projectId	string	ID business key to the Project record
projectRecordId	long	Project record's internal ID

Element	Type	Description
subStatusCode	string	Code business key to the activity's sub-status

After the call, the Project Activity record's sub-status is updated to match the request.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing an `ArtworkActivityLink` element. The `ArtworkActivityLink` element consists of the elements shown in the following table.

Returned Elements

If successful, an HTTP 200 response is sent.

Element	Type	Description
recordId	long	Project activity's internal ID
recordLink	string	URI to the Product Activity record, for use in a GET request
code	string	Code business key to the Project Activity record
title	string	Project activity's title

Error Messages

In the event that an error occurs, an HTTP 500 response is sent.

The following conditions may prevent a successful update:

- An invalid project record id or activity record id.
- The activity sub-statuses not being configured in Supplier Evaluation (including the general sub-statuses COMPLETED, MESSAGE SENT, MESSAGE ACCEPTED, and MESSAGE FAILED).
- The activity's template not being flagged as an artwork type, or having the permitted activity sub-statuses assigned, prior to the activity being created (a snapshot of the template settings is stored on the Activity record on creation).

BusinessCategoryService

This section describes the API for managing Business Categories. The following functions are available:

- [List of Values](#): retrieves a list of categories
- [Retrieve Record by ID](#): retrieves a Business Category record using its unique identifier
- [Check Record Modification Timestamp](#): retrieves the timestamp when a Business Category record was last updated
- [Create Record](#): creates a new Business Category record
- [Update Record](#): updates an existing Business Category record

- **Delete Record:** deletes an existing Business Category record

List of Values

Description

Retrieves a list of categories in a paged list. Use this function to retrieve a simple list of categories and IDs, or to locate Business Category record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/businessCategory **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30
code	Optional	string	Yes	Yes	POTATOES_LOOSE
parentCode	Optional	string	Yes	Yes	POTATOES
entityDescription	Optional	string	Yes	Yes	Loose Potatoes
specificationType	Optional	string	Yes	No	FOOD~PRODUCE
topLevelCategory	Optional	Boolean	No	No	TRUE
modifiedSince	Optional	string	No	No	2015-05-19 13:30:39
modifiedUntil	Optional	string	No	No	2015-05-19 13:30:39

Example URLs

.../services/rest/businessCategory/?offset=2&pageSize=20 .../services/rest/businessCategory/?code=POTATOES_LOOSE .../services/rest/businessCategory/?entityDescription=Potato%

The code parameter will attempt to find a single category with a matching code. The parentCode parameter will find any categories whose parent has the given code.

The entityDescription matches any category whose description (irrespective of language) matches the value given; this is just the description of the category in question and does not include those of parents.

The specificationType parameter is a tilde (~) separated list of types.

**Note:**

Including a value in the specificationType filter will exclude any categories for which no specification type has been selected.

The topLevelCategory parameter determines whether only top-level categories (those with no parent) will be included in the results.

Response Details

For a successful response, XML is returned with a BusinessCategoryLinkList root element containing an entries element for each matched category. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Business Category record's internal ID
recordLink	string	URI to the BusinessCategoryService Retrieve record service for this category
code	string	Code business key to the Business Category record
entityDescription	string	Category's name
path	string	Category's full path (including parent levels - separated by '/')

Error Messages

Element	Message	Meaning
specificationType	<###> is not a valid Specification Type	Specification type not found
pageSize	Error Code: INVALIDPAGESIZE	Page Size must be between 1 and 100
offset	Error Code: INVALIDOFFSET	Offset must be a positive integer
code	Error Code: INVALIDBUSINESSCATEGORY	Invalid Business Category
parentCode	Error Code: INVALIDBUSINESSCATEGORY	Invalid Business Category
specificationType	Error Code: INVALIDSPECIFICATIONTYPE	Invalid SpecificationType
topLevelCategory	Error Code: INVALIDBOOLEAN	Invalid topLevelCategory value Permitted values (not case-sensitive) are: true; yes; 1; false; no; 0

Retrieve Record by ID

Description

Retrieves a single Business Category record's details using the record's unique ID. Use this function to retrieve the full details of an individual category.

Endpoint address: /services/rest/businessCategory/{id} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/businessCategory/105

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a BusinessCategoryFullDTO root element containing the individual attributes of the requested Business Category record. If an ID is not specified, a list of all categories is returned (per the List of Values function).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 1 - Framework* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not found	Invalid {id} - not numeric
id	Invalid record id	Invalid {id} - not found

Check Record Modification Timestamp

Description

Retrieves the last modification date and time of a Business Category record. Use this function to determine when a category's details were last updated.

Endpoint address: /services/rest/businessCategory/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/businessCategory/105

Response Details

If successful, an HTTP 200 response is sent containing the Last-Modified header showing the date and time of the last update of the requested Business Category record. For example:

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Messages

Element	Message	Meaning
id	HTTP 417 Expectation Failed	Invalid {id} - not found
id	HTTP 404 Not found	Invalid {id} - not numeric

Create Record

Description

Creates a new Business Category record. Use this function to create a new category in Supplier Evaluation based on data sourced from the external system.

Dependencies: If a lower-level category, the parent Business Category must be present in the application and its record ID obtained. For more information, see [Dependencies](#).

Endpoint address: /services/rest/businessCategory **HTTP method:** POST

Request Details

The body of the request contains a BusinessCategoryFullDTO to specify the details of the category to create. Compared to retrieving a user (which uses the same BusinessCategoryFullDTO type), this request is much shorter. Only the attributes that are to be populated on the created Business Category record need to be included. As a minimum, this must include the fields shown in the following table:

Business Category Mandatory Fields

Field Name	Element Name
Code	code
Description	description
Parent Code	parentCode
Parent or Child level	topLevelCategory
Deleted Record	deleted (Set to false)
Path	path (The category's full path descriptions, concatenated and separated by /)



Note:

If creating a top level category, topLevelCategory must be set to true; if creating a lower level category, topLevelCategory must be set to false, and parentCode becomes mandatory (the parent category's code).

Example Request XML

```
<ns0:businessCategoryFullDTO xmlns:ns0="http://www.micros.com/creations/core/domain/dto/vlp0/full" xmlns:ns1="http://www.micros.com/creations/core/domain/dto/vlp0/simple">
  <ns0:code>X1</ns0:code>
  <ns0:localeData>
    <ns0:description>Category X - Level 1</ns0:description>
  </ns0:localeData>
</ns0:businessCategoryFullDTO>
```

```

    <ns0:path>Category X - Level 1</ns0:path>
  </ns0:localeData>
  <ns0:topLevelCategory>true</ns0:topLevelCategory>
  <ns0:deleted>false</ns0:deleted>
</ns0:businessCategoryFullDTO>

```

Set the deleted attribute to false to ensure the record can be seen in the UI. If omitted or set to true, it will exist, but will not be visible.

The following example adds a second level category to the above top-level parent category:

```

<ns0:businessCategoryFullDTO xmlns:ns0="http://www.micros.com/creations/core/domain/dto/vlp0/full" xmlns:ns1="http://www.micros.com/creations/core/domain/dto/vlp0/simple">
  <ns0:code>X2</ns0:code>
  <ns0:localeData>
    <ns0:description>Category X - Level 2</ns0:description>
    <ns0:path>Category X - Level 1 / Category X - Level 2</ns0:path>
  </ns0:localeData>
  <ns0:parentCode>X1</ns0:parentCode>
  <ns0:topLevelCategory>false</ns0:topLevelCategory>
  <ns0:deleted>false</ns0:deleted>
</ns0:businessCategoryFullDTO>

```

The value of the code element must be unique.

The description and path elements are locale-dependent, so are contained within a localeData element, with a locale element supported language code, for example:

```

<ns0:localeData>
  <ns0:description>Cheese</ns0:description>
  <ns0:locale>en_GB</ns0:locale>
</ns0:localeData>

<ns0:localeData>
  <ns0:description>Fromage</ns0:description>
  <ns0:locale>fr</ns0:locale>
</ns0:localeData>

```

The hierarchy of category levels is maintained through the parent code, linking a category to its immediate parent. All levels except the top level must therefore have a valid parent code specified.

The number of levels in the hierarchy is defined in the Supplier Evaluation Admin area.

Categories may be assigned to specific product specifications, or available for use by all; if specified, the specification type must be match those assigned to its parent categories.

Where the record is linked to another record, such as the Locale in this case, the business key must be provided (not the description) in order to form the link between the records. In general, the business key will be the code attribute of the linked record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 1 - Framework* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

For a successful response, an HTTP 200 response is sent with a body containing a BusinessCategoryLink root element. The root element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Newly created Business Category record's internal ID
recordLink	string	URI to the newly created Business Category record, for use in a GET request
code	string	Code ID business key to the newly created Business Category record
entityDescription	string	Category's description

Error Responses

If the supplied data does not result in a valid Business Category (such as a missing mandatory field), an HTTP 417 response is sent with an XML body message stating the validation errors. The request should not be reattempted with the same content.

Error Messages

Element	Message	Meaning
code	Code {0} has already been used	Codes must be unique
parent	The Parent Code cannot be blank when the Top Level Category flag is false	A parent code must be specified unless the category is a top level category
topLevelCategory	The Top Level Category flag cannot be true when the Parent Code has a value	A category cannot be a top level category and have a parent
description	Description must be provided	Descriptions are mandatory
code	The parent Business Category with code {0} cannot be found	The parent category must exist
parent	A new Business Category cannot be added to the Business Category with the code{0} because it will fall outside of the Business Category Hierarchy	Categories can only be added within the bounds of the Business Category Hierarchy as defined in Supplier Evaluation's Admin/System Control/ Business Category Configuration
specificationType	{0} is not a valid Specification Type(s)	The specification type or types must be valid
specificationType	The Specification Types must be a subset of the parent Business Category's Specification Types: {0}	This message is issued when no specification types are specified on a child, but there are specification types assigned to the parent It lists the specification types selected on the parent
specificationType	The Business Category Specification Types {0} are not a subset of the parent Business Category Specification Types {1}	This message is issued when the category's specification types have not all be selected on the parent The child's specification types are listed in {0} and the parent's specification types are listed in {1}

Update Record

Description

Updates an existing Business Category record. Use this function to update a category's details in Supplier Evaluation based on data sourced from the external system, or to move a category to appear beneath a different parent category.

Endpoint address: /services/rest/businessCategory/{id} **HTTP method:** PUT

Request Details

The body of the request contains a BusinessCategoryUpdateDTO to specify the updates to the Business Category record.

The request content is similar to that for creating a category, but does not include the child categories link. If children are to be managed, then a separate call is required for each child. As a minimum, the values specified as mandatory for the Create Record function (see above) must be included.

After the call, the Business Category record is updated to match the request.



Note:

When updating records, all values must be included. If a value or element is omitted from the request, the field contents will be cleared on the Business Category record.

The id element is used to locate the category to update.

The hierarchy of category levels is maintained through the parent code, linking a category to its immediate parent. All levels except the top level must therefore have a valid parent code specified.

The number of levels in the hierarchy is defined in the Supplier Evaluation Admin area.

Categories may be assigned to specific product specifications, or available for use by all; if specified, the specification type must be match those assigned to its parent categories.

Categories may be moved to the same level within another part of the hierarchy. When a category is moved, its children are automatically moved.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 1 - Framework* for details of their mapping to the fields within the Supplier Evaluation UI.

The body of the request contains a BusinessCategoryUpdateDTO element to specify how the category should appear after the update.

Response Details

If successful, an HTTP 200 response is sent with a body containing a BusinessCategoryLink element. The BusinessCategoryLink element consists of the returned elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Business Category record's internal ID
recordLink	string	URI to the Business Category record, for use in a GET request
code	string	Code ID business key to the Business Category record
entityDescription	string	Category's description
path	string	Category's full path (including parent levels - separated by '/')

Error Responses

If the supplied data does not result in a valid Business Category (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Error Messages

Element	Message	Meaning
recordId	Invalid record id	The ID used to call the PUT process does not relate to an existing record
parent	The Parent Code cannot be blank when the Top Level Category flag is false	A parent code must be specified unless the category is a top level category
topLevelCategory	The Top Level Category flag cannot be true when the Parent Code has a value	A category cannot be a top level category and have a parent
code	The parent Business Category with code {0} cannot be found	The parent category must exist
code	A new Business Category cannot be added to the Business Category with the code{0} because it will fall outside of the Business Category Hierarchy	Categories can only be added within the bounds of the Business Category Hierarchy as defined in Supplier Evaluation's Admin/System Control/ Business Category Configuration
parent	Business Category cannot be moved from a parent at level {0} to a parent at level {1}	Categories can only be moved between parent categories at the same level in the hierarchy
specificationType	{0} is not a valid Specification Type(s)	The specification type(s) must be valid
specificationType	The Specification Types must be a subset of the parent Business Category's Specification Types: {0}	This message is issued when no specification types are specified on a child but there are specification types assigned to the parent It lists the specification types selected on the parent
specificationType	The Business Category Specification Types {0} are not a subset of the parent Business Category Specification Types {1}	This message is issued when the category's specification types have not all be selected on the parent The child's specification types are listed in {0} and the parent's specification types are listed in {1}

Delete Record

Description

Deletes an existing Business Category record, along with any associated child categories. Use this function to remove a single category or a category and all its lower level categories.

Endpoint address: /services/rest/businessCategory/{id} **HTTP method:** DELETE

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to delete.

Response Details

If the Business Category is deleted successfully, an HTTP 200 response is sent.

Error Responses

If a Business Category cannot be deleted due to it being referenced by another record (or a child of the category being referenced by another record), an HTTP 417 response is sent with an ErrorMessage/Message XML body containing the message: "Could not delete: with a URI link to the category."

Error Messages

Element	Message	Meaning
id	HTTP 417 Could not delete: {URI}	Category is referenced by another record
id	HTTP 404 Not found	Invalid {id} - not numeric
id	Invalid record id for deletion	Invalid {id} - not found

AuditRestService

This section describes the API for managing audit and visits. The following functions are available:

- [List of Values \(Audit\)](#): retrieves a list of audits and visits
- [Retrieve Record by ID \(Audit\)](#): retrieves an Audit or Visit record using its unique identifier
- [Retrieve Record by Business Key \(Audit\)](#): retrieves an Audit or Visit record using its business key
- [Check Record Modification Timestamp \(Audit\)](#): retrieves the timestamp when an Audit or Visit record was last updated
- [Create Record \(Audit\)](#): creates a new Audit or Visit record
- [Update Record \(Audit\)](#): updates an existing Audit or Visit record
- [List of Values \(Checklist\)](#): retrieves a list of checklists
- [Retrieve Record by ID \(Checklist\)](#): retrieves a Checklist record using its unique identifier
- [Retrieve Record by Business Key \(Checklist\)](#): retrieves a Checklist record using its business key

- [Create Record \(Checklist\)](#): creates a new Checklist record
- [Update Record \(Checklist\)](#): updates an existing Checklist record
- [Delete Record \(Checklist\)](#): deletes an existing Checklist record

List of Values (Audit)

Description

Retrieves a list of Audits or Visits in a paged list. Use this function to locate Audit or Visit record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/audit **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30
modifiedSince	Optional	string	No	No	2015-05-19 13:30:39
modifiedUntil	Optional	string	No	No	2015-05-19 13:30:39
status	Optional	string	Yes	No	Scheduled~In Progress
softDelete	Optional	string	No	No	true

Example URL

.../services/rest/audit/?offset=2&pageSize=20

Response Details

For a successful response, XML is returned with an AuditVisitLinkList root element containing an entries element for each matched audit or visit. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Audit/Visit record's internal ID
recordLink	string	URI to the AuditRestService Retrieve record service for this audit/visit
code	string	Audit/Visit's code business key
dueDate	string	Audit/Visit's due date
siteCode	string	Audit/Visit's site code
supplierCode	string	Audit/Visit's supplier code

Element	Type	Description
templateCode	string	Audit/Visit's template code

Error Messages

Element	Message	Meaning
pageSize	Error Code: INVALIDPAGESIZE	Page Size must be between 1 and 100

Retrieve Record by ID (Audit)

Description

Retrieves a single Audit or Visit record's details using the record's internal unique ID (which is not visible in the UI). Use this function to retrieve the full details of an individual audit or visit.

Endpoint address: /services/rest/audit/{id} **HTTP method:** GET

Request Details

The URL contains the {id} parameter that determines the record to retrieve.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a localeData element returned for the specified locales if data is present.

Example URL

.../services/rest/audit/105

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of an AuditFullDTO root element containing the individual attributes of the requested Audit or Visit record. If an ID is not specified, a list of all audits/visits is returned (per the List of Values function).

File Structure:

The main elements of the auditFullDTO root element:

- auditTemplateSnapshot and Template - the settings from the associated template.
- nonConformances - If the audit/visit has any non conformances/issues raised against it, they appear within a nonConformances element - a separate element for each.
- supplier and site - details of the supplier and site that the audit/visit is related to.

- udfData - the contents of any user-defined custom fields associated to the Audit/Visit record. nonConformances may also contain udfData elements, if the Non Conformance record has user-defined custom fields configured.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Audit to return with id:<###>	Invalid {id} - not found

Retrieve Record by Business Key (Audit)

Description

Retrieves a single Audit or Visit record's details using its business key (code). Use this function to retrieve the full details of an individual audit or visit using its key code.

Endpoint address: /services/rest/audit/byKey/{code} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {code} parameter that determines the record to retrieve.

Example URL

.../services/rest/audit/byKey/110

Response Details

If successful, an HTTP 301 redirect response is sent containing a header Location with the URI of the document requested.

The returned URI represents the call to be made to the Retrieve Record by ID function (with the id substituted with the relevant value), which will return an AuditFullDTO Element Message containing the individual attributes of the requested Audit or Visit record.



Note:

If your method of calling the API is configured to automatically follow HTTP 301 redirects, the call may automatically return a AuditFullDTO root element containing the individual attributes of the requested Audit or Visit record.

File Structure:

The main elements of the auditFullDTO root element:

- auditTemplateSnapshot and Template - the settings from the associated template.

- **nonConformances** - If the audit/visit has any non conformances/issues raised against it, they appear within a nonConformances element - a separate element for each.
- **supplier and site** - details of the supplier and site that the audit/visit is related to.
- **udfData** - the contents of any user-defined custom fields associated to the Audit/Visit record. nonConformances may also contain udfData elements, if the Non Conformance record has user-defined custom fields configured.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
code	HTTP 404 Not Found	Invalid {code} - blank or not found
code	HTTP 404 Not Found	Invalid {code} - not found

Check Record Modification Timestamp (Audit)

Description

Retrieves the last modification date and time of an Audit or Visit record. Use this function to determine when the audit or visit was last updated.

Endpoint address: /services/rest/audit/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/audit/105

Response

If successful, an HTTP 200 response is sent containing the Last-Modified header to show the last modification date and time of the last update of the requested Audit or Visit record.

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Messages

Element	Message	Meaning
id	HTTP 417 Expectation Failed	Invalid {id} - not found
id	HTTP 404 Not Found	Invalid {id} - not numeric

Create Record (Audit)

Description

Creates a new Audit or Visit record. Use this function to create a new audit or visit in Supplier Evaluation based on data sourced from the external system.

Dependencies: The Supplier and Site must be present in the application and its record ID obtained. If assigning a Business Category, the ID of the Business Category record must be present in the application and its record ID obtained. If assigning a Technologist or People Present, the users must be present in the application and the record IDs of the User record obtained. If non conformances/issues are to be included, and they are associated to a specific user of the application, the record IDs of the User record must be obtained. An audit/visit must be associated to an audit/visit template. For more information, see [Dependencies](#).

Endpoint address: /services/rest/audit **HTTP method:** POST

Request Details

The body of the request contains an AuditFullDTO element to specify the detail of the audit or visit to create, which is based on the AuditFullDTO element returned when retrieving an audit/visit. As a minimum, the fields shown in the following table must be populated.

Audit/Visit Mandatory Fields

Field Name	Element Name
Audit/Visit Template	template / id (or code) auditTemplateSnapshot
Audit/Visit Code	code
Lead Technologist	leadTechnicalManager / code
Supplier	supplier / id
Site	site / id
Due Date	dueDate
Record Type	recordType
Status	status / id
From Date	fromDate
To Date	toDate
People Present	peoplePresent / name
Deleted Record	deleted (Set to false)



Note:

If the status is Scheduled, fromDate, toDate, and peoplePresent are not mandatory.

To ensure a valid unique business key code is automatically assigned to the record being created, a value of zero must be passed in the <code> element of the XML when submitting the POST request.

People Present Table:

This table is used to identify the people present during the audit or visit. The options for completing the table depend on the settings in the audit/visit's template:

- The table must contain at least one entry (whether an audit or a visit).
- If a visit or an internal audit, the entry should be a user of the system who has a Technologist role and is set as an approved auditor. The template may have been set to just allow specific auditors.
- If a third-party audit, the entry should be a Certification Body. The template may have been set to just allow specific auditors.
- Rows of free text may be added to the table.

File Structure:

The main elements of the AuditFullDTO root element are shown in the following table:

Summary of AuditFullDTO Elements

Element Name	Description
announced costChargedToSupplier auditTemplateSnapshot	Values from the audit/visit template, required to control the record's behavior when it is progressed within Supplier Evaluation.
code	The audit/visit's business key.
dueDate fromDate toDate leadTechnical Manager recordType	Values that may be mandatory for the audit/visit to be created.
nonConformances	Container for the non conformance data - a separate element for each non conformance/issue.
supplier	ID of the supplier that the audit/visit is associated to.
site	ID of the site that the audit/visit is associated to.
status	The audit/visit's status.
template	ID of the audit/visit template that the audit/visit is associated to.

Example Request XML

This example shows the minimum requirement to be able to create an In Progress audit or visit.

```
<ns0:auditFullDTO xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full"
xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple">
  <ns0:announced>true</ns0:announced>
  <ns0:costChargedToSupplier>false</ns0:costChargedToSupplier>
  <ns0:auditTemplateSnapshot>
    <ns0:allowWorkingOffline>false</ns0:allowWorkingOffline>
    <ns0:allowableScoreOptions>
      <ns1:code>AMBER</ns1:code>
    </ns0:allowableScoreOptions>
    <ns0:allowableScoreOptions>
      <ns1:code>GREEN</ns1:code>
    </ns0:allowableScoreOptions>
    <ns0:auditType>INTERNAL</ns0:auditType>
  </ns0:auditTemplateSnapshot>
</ns0:auditFullDTO>
```

```

<ns0:auditVisibility>OPTIONAL</ns0:auditVisibility>
<ns0:autoSchedule>>false</ns0:autoSchedule>
<ns0:buyerInformationRequired>>true</ns0:buyerInformationRequired>
<ns0:categoriesRequired>2</ns0:categoriesRequired>
<ns0:categoryRequired>>false</ns0:categoryRequired>
<ns0:certificationBodiesSelectable>ALL</ns0:certificationBodiesSelectable>
<ns0:closureDeadline>56</ns0:closureDeadline>
<ns0:closureDeadlineNoOfDays>0</ns0:closureDeadlineNoOfDays>
  <ns0:commentsNotVisibleToOrganisationRequired>true
</ns0:commentsNotVisibleToOrganisationRequired>
<ns0:costChargedToSupplier>>false</ns0:costChargedToSupplier>
<ns0:defaultFrequency>12</ns0:defaultFrequency>
<ns0:defaultIssueCompletion>28</ns0:defaultIssueCompletion>
<ns0:extendedDeadline>56</ns0:extendedDeadline>
<ns0:extendedDeadlineNoOfDays>0</ns0:extendedDeadlineNoOfDays>
<ns0:internalAuditorsSelectable>ALL</ns0:internalAuditorsSelectable>
<ns0:isActive>true</ns0:isActive>
<ns0:localeData>
  <ns0:name>Checklist Audit</ns0:name>
  <ns0:id>49</ns0:id>
</ns0:localeData>
<ns0:mayExternalManagersEditConfig>MAY_NOT_EDIT
</ns0:mayExternalManagersEditConfig>
<ns0:nonConformanceCreationMethod>VIA_CHECKLISTS
</ns0:nonConformanceCreationMethod>
<ns0:organisationType>SITE</ns0:organisationType>
<ns0:recordType>AUDIT</ns0:recordType>
<ns0:scoreAudit>true</ns0:scoreAudit>
<ns0:scoreRequired>true</ns0:scoreRequired>
<ns0:stampOrgWithLastAuditDate>true</ns0:stampOrgWithLastAuditDate>
<ns0:supplierGenerated>>false</ns0:supplierGenerated>
<ns0:useAuditVisitStandards>>false</ns0:useAuditVisitStandards>
<ns0:id>6</ns0:id>
</ns0:auditTemplateSnapshot>
<ns0:dueDate>2017-04-27</ns0:dueDate>
<ns0:fromDate>2017-04-27</ns0:fromDate>
<ns0:leadTechnicalManager>
  <ns1:code>PROD-TECH</ns1:code>
  <ns1:disabled>>false</ns1:disabled>
  <ns1:externalAuthenticationUser>>false</ns1:externalAuthenticationUser>
  <ns1:timeZone>Europe/London</ns1:timeZone>
  <ns1:userType>RETAILER</ns1:userType>
  <ns1:id>12</ns1:id>
</ns0:leadTechnicalManager>
<ns0:nonConformances>
  <ns0:description><p>Test</p></ns0:description>
  <ns0:reference>AR01</ns0:reference>
  <ns0:status>
    <ns1:status>OPEN</ns1:status>
  </ns0:status>
  <ns0:type>
    <ns1:code>MAJOR</ns1:code>
  </ns0:type>
  <ns0:code>AR01-11</ns0:code>
</ns0:nonConformances>
<ns0:recordType>AUDIT</ns0:recordType>
<ns0:site>
  <ns1:id>27</ns1:id>
</ns0:site>
<ns0:status>
  <ns1:status>IN PROGRESS</ns1:status>
</ns0:status>

```

```

<ns0:supplier>
  <ns1:id>21</ns1:id>
</ns0:supplier>
<ns0:template>
  <ns1:id>8</ns1:id>
</ns0:template>
<ns0:toDate>2017-04-27</ns0:toDate>
<ns0:code>AR1</ns0:code>
<ns0:deleted>false</ns0:deleted>
</ns0:auditFullDTO>

```

Set the deleted attribute to false to ensure the record can be seen in the UI. If omitted or set to true, it will exist, but will not be visible.

Where the record is linked to another record, such as the Template, the business key or ID must be provided (not the description) in order to form the link between the records. In general, the business key will be the code attribute of the linked record. The supplier and site elements can contain just the ID of the respective record links.

A unique code for the audit/visit must be assigned.

Templates:

The audit/visit is linked to its template using the template and auditTemplateSnapshot elements. The template element can contain just the code or ID of the associated Template record.

The auditTemplateSnapshot element must be completed to provide the parameters that control the audit/visits' behavior as it progresses through its Supplier Evaluation workflow. If scoring is used, the allowable options can be specified using the code or ID of the glossary record; if none are specified, all options will be available when the user selects the score.

Non Conformances:

If the audit/visit is to be created with non conformances/issues already raised against it, they can be added within a nonConformances element - a separate element for each. A unique code must be assigned to each. The status can be set using its code or ID.

For further information about the process for creating audit, non conformances and checklists, see [Dependencies](#).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a AuditVisitLink root element. The root element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Newly created Audit/Visit record's internal ID
recordLink	string	URI to the newly created Audit/Visit record, for use in a GET request
code	string	Audit/Visit's code business key
dueDate	string	Audit/Visit's due date
siteCode	string	Audit/Visit's site code

Element	Type	Description
supplierCode	string	Audit/Visit's supplier code
templateCode	string	Audit/Visit's template code

Error Responses

If the supplied data does not result in a valid Audit or Visit (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Error Messages

Element	Message	Meaning
peoplePresent	WARNING: class com.micros.creations.core.domain.Audit.peoplePresent - There must be at least one entry in the People Present table	People Present not provided
recordType	IllegalArgumentException: No enum constant com.micros.creations.core.type.AuditRecordType. <###>	Record Type not found
supplier/id code site/id code supplierId supplierCode	HTTP 417 IllegalStateException: Cannot locate keyword for class com.micros.creations.core.domain.Site using id: <###>, code:<###>, supplierCode:<###> and supplierId:<###>	Site Id not found Site Code not found Supplier Id not found Supplier Code not found
template	EntityNotFoundException: Unable to find com.micros.creations.core.domain.AuditTemplate with id <###>	Template not found
fromDate	ERROR: class com.micros.creations.core.domain.Audit.fromDate - The condition is invalid	From Date not provided
code	ERROR: class com.micros.creations.core.domain.Audit.code - The condition is invalid	Audit/Visit code not provided
code	ERROR: class com.micros.creations.core.domain.Audit.code - Another object exists with this value	Audit/Visit code already exists

Update Record (Audit)

Description

Updates an existing Audit or Visit record. Use this function to update an audit or visit's details in Supplier Evaluation based on data sourced from the external system.

Endpoint address: /services/rest/audit/{id} **HTTP method:** PUT

Request Details

The body of the request contains an AuditFullDTO element to specify the updates to the Audit or Visit record, which is based on the AuditFullDTO type returned when retrieving an audit/visit.



Note:

The request content is similar to that for creating an audit or visit.

As a minimum, the values specified as mandatory for the Create Record function (see above) must be included. However, when updating records, all values must be included.

If a value or element is omitted from the request, the field contents will be cleared on the Audit/Visit record.

If the audit/visit has any non conformances/issues raised against it, they appear within a nonConformances element - a separate element for each. The non conformances/issues can be updated or created when updating the audit/visit. If creating a new Non Conformance record, assign a unique code and omit the id element.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing an AuditVisitLink element. The AuditVisitLink element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Audit/Visit record's internal ID
recordLink	string	URI to the Audit/Visit record, for use in a GET request
code	string	Audit/Visit's code business key
dueDate	string	Audit/Visit's due date
siteCode	string	Audit/Visit's site code
supplierCode	string	Audit/Visit's supplier code
templateCode	string	Audit/Visit's template code

Error Responses

If the supplied data does not result in a valid Audit or Visit (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

List of Values (Checklist)

Description

Retrieves a list of Checklists in a paged list. Use this function to locate Checklist record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/audit/{auditId}/checklist **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
auditId	Optional	long	No	No	105
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30

Example URL

.../services/rest/audit/105/checklist/?offset=2&pageSize=20

Response Details

For a successful response, XML is returned with an AuditChecklistLinkList root element containing an entries element for each matched audit or visit. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Checklist record's internal ID
recordLink	string	URI to the AuditRestService Retrieve record service for this checklist
parentId	string	The parent Audit/Visit's internal ID
auditCode	string	The parent Audit/Visit's code business key
globalId	string	Checklist's code business key

Error Messages

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record by ID (Checklist)

Description

Retrieves a single Checklist record's details using the record's internal unique ID (which is not visible in the UI) and its parent audit/visit's internal unique Id. Use this function to retrieve the full details of an individual audit or visit.

Endpoint address: /services/rest/audit/{auditId}/checklist/{id} **HTTP method:** GET

Request Details

The URL contains the {auditId} parameter that identifies the parent Audit/Visit record, and the {id} parameter that determines the record to retrieve.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a localeData element returned for the specified locales if data is present.

Example URL

.../services/rest/audit/105/checklist/7

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of an auditChecklistFullDTO root element containing the individual attributes of the requested Checklist record. If a Checklist ID is not specified, a list of all the audit/visit's checklists is returned (as per the List of Values function).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
auditId	HTTP 404 Not found	Invalid {auditId} - not numeric
auditId	HTTP 417 IllegalStateException: Cannot find the Audit to return with id:<###>	Invalid {auditId} - not found
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Audit to return with id:<###>	Invalid {id} - not found

Retrieve Record by Business Key (Checklist)

Description

Retrieves a single Checklist record's details using its business key (code) and its parent audit/visit's code. Use this function to retrieve the full details of an individual checklist using its key code.

Endpoint address: /services/rest/audit/byKey/{auditCode}/checklist/{checklistCode} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {auditCode} parameter that identifies the parent Audit/Visit record, and the {checklistCode} parameter that determines the record to retrieve.

Example URL

.../services/rest/audit/byKey/110/checklist/7374be1e-2a6c-11e7-8fda-0021f60dd918

Response Details

If successful, an HTTP 301 redirect response is sent containing a header Location with the URI of the document requested.

The returned URI represents the call to be made to the Retrieve Record by ID function (with the id substituted with the relevant value), which will return an auditChecklistFullIDTO root element containing the individual attributes of the requested Checklist record.



Note:

If your method of calling the API is configured to automatically follow HTTP 301 redirects, the call may automatically return an auditChecklistFullIDTO root element containing the individual attributes of the requested Checklist record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
auditCode	HTTP 404 Not Found	Invalid {auditCode} - blank or not found
checklistCode	HTTP 404 Not Found	Invalid {checklistCode} - blank or not found

Create Record (Checklist)

Description

Creates a new Checklist record. Use this function to create a new checklist in Supplier Evaluation based on data sourced from the external system.

Dependencies: To create an audit with a checklist, the parent Audit record must be created using the Create Record (Audit/Visit) function, before creating the associated Checklist records.

Therefore, when creating the checklist, the parent Audit/Visit record must be present in the application and its record ID obtained. A checklist's questions and the permitted answers are determined by the checklist template that has been assigned to the audit/visit's template. The IDs of the Checklist Template record must also be provided when creating a template. For more information, see [Dependencies](#).

Endpoint address: /services/rest/audit{auditId}/checklist **HTTP method:** POST

Request Details

The body of the request contains an auditChecklistFullDTO element to specify the detail of the checklist to create, which is based on the auditChecklistFullDTO element returned when retrieving a checklist. As a minimum, the fields shown in the following table must be populated.

Checklist Mandatory Fields

Field Name	Element Name
Audit/Visit Id	audit / id See note 1 .
Audit/Visit Code	audit /code See note 1 .
Checklist Template	localeData / id
Checklist Comments Allowed	commentsAllowed
Questionnaire Questions	questionnaire / questionnaireQuestion / questionDetails
Question Answers	questionnaire / questionnaireQuestion / questionnaireAnswer questionnaire / questionnaireQuestion / quesAvailAnsFull
Deleted Record	deleted (Set to false)

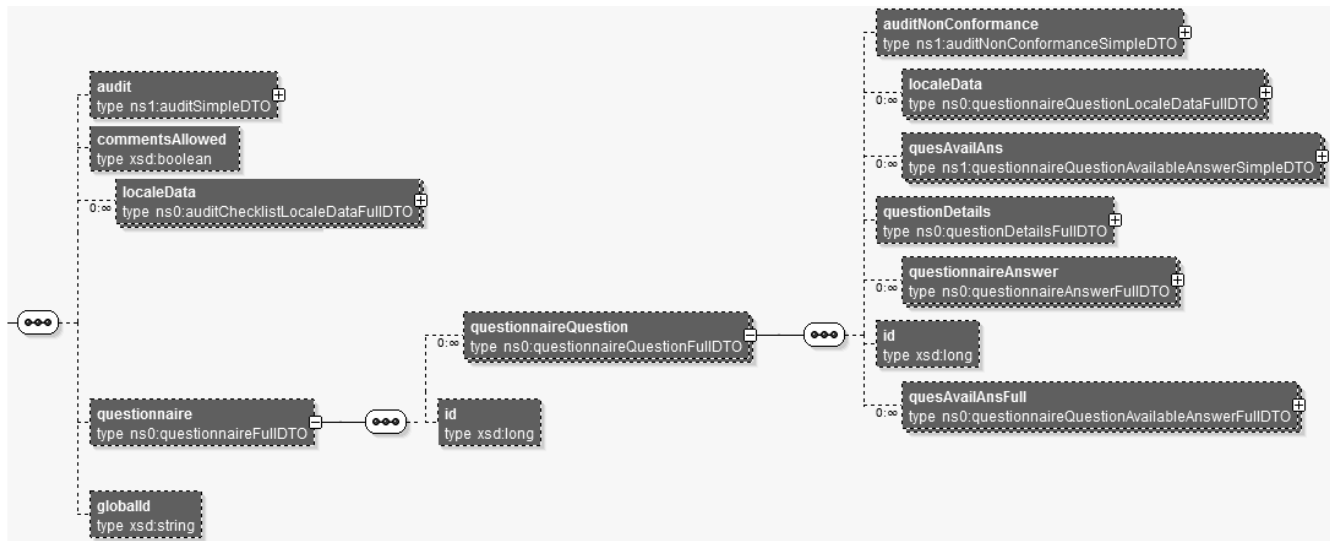
Notes:

1. Either code or ID may be specified to identify the audit/visit.

File Structure:

The following diagram shows the structure of the auditChecklistFullDTO request file structure:

Figure 4-3 RESTful Checklist Request



Summary of AuditChecklistFullIDTO Elements

Element Name	Description
audit	Identifies the parent audit record by code or id.
commentsAllowed	A value from the checklist template, required to control the record's behavior when it is progressed within Supplier Evaluation.
localeName	Identifies the checklist template by name and id.
questionnaire	Groups the questions and their answers.
questionnaireQuestion	Groups the checklist questions and their answers - repeated for each question in the checklist. reference is the question reference.
questionDetails	Individual questions. question is the question text. answerComment is the answer's comments (if configured). answerFieldType is code of answer type (MULTI_SELECTION, TEXT, or SINGLE_SELECTION). commentAllowed is whether comments are allowed.
questionnaireAnswer	Groups the answer details.
questionnaireAvailableAnswer	Answers for the individual questions. code is the question's unique key. commentMandatory is whether comments are mandatory or not. generateNonConformance is whether an answer will generate an issue.
quesAvailAnsFull	The available answers - repeated for each answer. code is the question's unique key. commentMandatory is whether comments are mandatory or not. id and keywordTypeEnum are the keys to the Issues glossary.

Process for Creating Checklist Audits

The general process for creating checklist audits:

1. Create the Audit/Visit. While it is possible to create a checklist audit using the Create Audit function, it is recommended that they be created within the Supplier Evaluation application.

A method of creating a new audit:

- a. Retrieve an existing audit using a GET function.
- b. Omit the id and statusChangeHistory elements from the retrieved file.
- c. Omit localData elements from status and allowableAuditStandards.
- d. Assign a new unique code.
- e. Link to the Supplier, Site, and Audit Template records using the relevant IDs.
- f. Change other content to reflect the new audit.
- g. Submit using the POST function.
- h. If non conformances are to be created at the time of creating the audit/visit, include them as nonConformances elements.

Note:

The API does not automatically create a non conformance based on the answer of a checklist question. Non conformances must be assigned when either creating or updating the Audit/Visit record.

2. Retrieve the code or ID of the newly created Audit, and its template detail using a GET function.
3. Create the checklist using the POST function (repeat for each checklist):
 - Identify the parent audit/visit by setting the code or id within audit element.
 - Can retrieve an existing checklist using a GET function, in which case the id and questionnaireQuestionAvailableAnswer elements must be omitted. Also omit any id elements within the questionnaire element, except those that relate to the auditNonConformance element.

Example Request XML

This example shows the minimum requirement to be able to create an audit checklist. This example contains three questions, each with the available answers of Yes, No, or Not Applicable.

Figure 4-4 Checklist Questions

Details Attachments Change History		
Questions		
Ref1	Question 1	
	Yes	Yes comment
Ref2	Question 2	
	No	No comment
	Assigned to: Assignee	Deadline: 20/05/17
Ref3	Question 3	
	Not Applicable	N/A comment

```

<ns0:auditChecklistFullDTO xmlns:ns1="http://www.micros.com/creations/core/domain/dto/
v1p0/simple" xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
  <ns0:audit>
    <ns1:code>AUD09</ns1:code>
    <ns1:id>132</ns1:id>
  </ns0:audit>
  <ns0:commentsAllowed>false</ns0:commentsAllowed>
  <ns0:localeData>
    <ns0:name>
      <![CDATA[ API Test Template ]]>
    </ns0:name>
    <ns0:id>563</ns0:id>
  </ns0:localeData>
  <ns0:questionnaire>
    <ns0:questionnaireQuestion>
      <ns0:localeData>
        <ns0:reference>
          <![CDATA[Ref1]]>
        </ns0:reference>
      </ns0:localeData>
      <ns0:questionDetails>
        <ns0:answerComment>
          <![CDATA[Yes comment]]>
        </ns0:answerComment>
        <ns0:answerFieldType>SINGLE_SELECTION</ns0:answerFieldType>
        <ns0:commentAllowed>true</ns0:commentAllowed>
        <ns0:localeData>
          <ns0:question>
            <![CDATA[Question 1]]>
          </ns0:question>
        </ns0:localeData>
      </ns0:questionDetails>
      <ns0:questionnaireAnswer>
        <ns0:questionnaireAvailableAnswer>
          <ns1:code>QQAA7236</ns1:code>
          <ns1:commentMandatory>false</ns1:commentMandatory>
          <ns1:generateNonConformance>false</ns1:generateNonConformance>
          <ns1:localeData>
            <ns0:answer>Yes</ns0:answer>
          </ns1:localeData>
        </ns0:questionnaireAvailableAnswer>
        <ns0:questionnaireQuestion>
          <ns1:localeData>
            <ns0:reference>
              <![CDATA[Ref1]]>
            </ns0:reference>
          </ns1:localeData>
        </ns0:questionnaireQuestion>
      </ns0:questionnaireAnswer>
      <ns0:quesAvailAnsFull>
        <ns0:code>QQAA7236</ns0:code>
        <ns0:commentMandatory>false</ns0:commentMandatory>
        <ns0:generateNonConformance>false</ns0:generateNonConformance>
        <ns0:localeData>
          <ns0:answer>Yes</ns0:answer>
        </ns0:localeData>
      </ns0:quesAvailAnsFull>
      <ns0:quesAvailAnsFull>
        <ns0:code>QQAA7237</ns0:code>
        <ns0:commentMandatory>true</ns0:commentMandatory>
        <ns0:generateNonConformance>true</ns0:generateNonConformance>

```

```

        <ns0:localeData>
          <ns0:answer>No</ns0:answer>
        </ns0:localeData>
      </ns0:quesAvailAnsFull>
    <ns0:quesAvailAnsFull>
      <ns0:code>QQAA7238</ns0:code>
      <ns0:commentMandatory>false</ns0:commentMandatory>
      <ns0:generateNonConformance>false</ns0:generateNonConformance>
      <ns0:localeData>
        <ns0:answer>Not Applicable</ns0:answer>
      </ns0:localeData>
    </ns0:quesAvailAnsFull>
  </ns0:questionnaireQuestion>
  <ns0:questionnaireQuestion>
    <ns0:localeData>
      <ns0:reference>
        <![CDATA[Ref2]]>
      </ns0:reference>
    </ns0:localeData>
    <ns0:questionDetails>
      <ns0:answerComment>
        <![CDATA[No comment]]>
      </ns0:answerComment>
      <ns0:answerFieldType>SINGLE_SELECTION</ns0:answerFieldType>
      <ns0:commentAllowed>true</ns0:commentAllowed>
      <ns0:localeData>
        <ns0:question>
          <![CDATA[Question 2]]>
        </ns0:question>
      </ns0:localeData>
    </ns0:questionDetails>
    <ns0:questionnaireAnswer>
      <ns0:questionnaireAvailableAnswer>
        <ns1:code>QQAA7240</ns1:code>
        <ns1:commentMandatory>true</ns1:commentMandatory>
        <ns1:generateNonConformance>true</ns1:generateNonConformance>
        <ns1:localeData>
          <ns0:answer>No</ns0:answer>
        </ns1:localeData>
      </ns0:questionnaireAvailableAnswer>
    </ns0:questionnaireQuestion>
    <ns1:localeData>
      <ns0:reference>
        <![CDATA[Ref2]]>
      </ns0:reference>
    </ns1:localeData>
  </ns0:questionnaireQuestion>
</ns0:questionnaireAnswer>
<ns0:quesAvailAnsFull>
  <ns0:code>QQAA7239</ns0:code>
  <ns0:commentMandatory>false</ns0:commentMandatory>
  <ns0:generateNonConformance>false</ns0:generateNonConformance>
  <ns0:localeData>
    <ns0:answer>Yes</ns0:answer>
  </ns0:localeData>
</ns0:quesAvailAnsFull>
<ns0:quesAvailAnsFull>
  <ns0:code>QQAA7240</ns0:code>
  <ns0:commentMandatory>true</ns0:commentMandatory>
  <ns0:generateNonConformance>true</ns0:generateNonConformance>
  <ns0:localeData>
    <ns0:answer>No</ns0:answer>
  </ns0:localeData>
</ns0:quesAvailAnsFull>

```

```

        </ns0:localeData>
    </ns0:quesAvailAnsFull>
    <ns0:quesAvailAnsFull>
        <ns0:code>QQAA7241</ns0:code>
        <ns0:commentMandatory>false</ns0:commentMandatory>
        <ns0:generateNonConformance>false</ns0:generateNonConformance>
        <ns0:localeData>
            <ns0:answer>Not Applicable</ns0:answer>
        </ns0:localeData>
    </ns0:quesAvailAnsFull>
</ns0:questionnaireQuestion>
<ns0:questionnaireQuestion>
    <ns0:localeData>
        <ns0:reference>
            <![CDATA[Ref3]]>
        </ns0:reference>
    </ns0:localeData>
    <ns0:questionDetails>
        <ns0:answerComment>
            <![CDATA[N/A comment]]>
        </ns0:answerComment>
        <ns0:answerFieldType>SINGLE_SELECTION</ns0:answerFieldType>
        <ns0:commentAllowed>true</ns0:commentAllowed>
        <ns0:localeData>
            <ns0:question>
                <![CDATA[Question 3]]>
            </ns0:question>
        </ns0:localeData>
    </ns0:questionDetails>
    <ns0:questionnaireAnswer>
        <ns0:questionnaireAvailableAnswer>
            <ns1:code>QQAA7244</ns1:code>
            <ns1:commentMandatory>false</ns1:commentMandatory>
            <ns1:generateNonConformance>false</ns1:generateNonConformance>
            <ns1:localeData>
                <ns0:answer>Not Applicable</ns0:answer>
            </ns1:localeData>
        </ns0:questionnaireAvailableAnswer>
        <ns0:questionnaireQuestion>
            <ns1:localeData>
                <ns0:reference>
                    <![CDATA[Ref3]]>
                </ns0:reference>
            </ns1:localeData>
        </ns0:questionnaireQuestion>
    </ns0:questionnaireAnswer>
</ns0:quesAvailAnsFull>
<ns0:quesAvailAnsFull>
    <ns0:code>QQAA7242</ns0:code>
    <ns0:commentMandatory>false</ns0:commentMandatory>
    <ns0:generateNonConformance>false</ns0:generateNonConformance>
    <ns0:localeData>
        <ns0:answer>Yes</ns0:answer>
    </ns0:localeData>
</ns0:quesAvailAnsFull>
<ns0:quesAvailAnsFull>
    <ns0:code>QQAA7243</ns0:code>
    <ns0:commentMandatory>true</ns0:commentMandatory>
    <ns0:generateNonConformance>true</ns0:generateNonConformance>
    <ns0:localeData>
        <ns0:answer>No</ns0:answer>
    </ns0:localeData>
</ns0:quesAvailAnsFull>

```

```

    <ns0:quesAvailAnsFull>
      <ns0:code>QQA7244</ns0:code>
      <ns0:commentMandatory>false</ns0:commentMandatory>
      <ns0:generateNonConformance>false</ns0:generateNonConformance>
      <ns0:localeData>
        <ns0:answer>Not Applicable</ns0:answer>
      </ns0:localeData>
    </ns0:quesAvailAnsFull>
  </ns0:questionnaireQuestion>
</ns0:questionnaire>
<ns0:globalId>7470e250-36cc-4dea-a0f1-eb6040185865</ns0:globalId>
<ns0:deleted>false</ns0:deleted>
</ns0:auditChecklistFullDTO>

```

Set the deleted attribute to false to ensure the record can be seen in the UI. If omitted or set to true, it will exist, but will not be visible.

For further information about the process for creating audit, non conformances, and checklists, see [Dependencies](#).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing an AuditChecklistLink root element. The root element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Checklist record's internal ID
recordLink	string	URI to the newly created Checklist record, for use in a GET request
parentId	long	The parent Audit/Visit's internal ID
auditCode	string	The parent Audit/Visit's code business key
globalId	string	Checklist's code business key

Error Responses

If the supplied data does not result in a valid Checklist (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Error Messages

Element	Message	Meaning
audit / id audit / code	HTTP 417 IllegalStateException: The checklists audit cannot be located	Audit/Visit id/code not found
id	HTTP 417 IllegalStateException: Cannot find matching relation to update on <###> looking for <###>	Invalid link to related record

Update Record (Checklist)

Description

Updates an existing Checklist record. Use this function to update a checklist's details in Supplier Evaluation based on data sourced from the external system.

Endpoint address: /services/rest/audit/{auditId}/checklist/{id} **HTTP method:** PUT

Request Details

The body of the request contains an auditChecklistFullDTO element to specify the updates to the Audit or Visit record, which is based on the auditChecklistFullDTO type which is returned when retrieving an audit/visit.



Note:

The request content is similar to that for creating a checklist.

As a minimum, the values specified as mandatory for the Create Record function (see above) must be included. However, when updating records, all values must be included.

If a value or element is omitted from the request, the field contents will be cleared on the Checklist record.

The link to the parent Audit/Visit record is defined by the code or ID within the audit element.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing an auditChecklistLink element. The auditChecklistLink element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Checklist record's internal ID
recordLink	string	URI to the Checklist record, for use in a GET request
parentId	long	The parent Audit/Visit's internal ID
auditCode	string	The parent Audit/Visit's code business key
globalId	string	Checklist's code business key

Error Messages

If the supplied data does not result in a valid Audit or Visit (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Delete Record (Checklist)

Deletes an existing Checklist record. Use this function to delete a checklist that has been created in error from the external system

Endpoint address: /services/rest/audit/{auditId}/checklist/{id} **HTTP method:** DELETE

Request Details

There are no request parameters, but the URL contains the {auditId} parameter that identifies the parent Audit/Visit record, and the {id} parameter that determines the record to delete.



Note:

It is not possible to manually delete checklists from an audit/visit within the application; this delete function has been provided solely for the deletion of checklists that have been created in error by the API.

To delete a checklist, the external system must be granted access to the AUDIT_CHECKLIST_DELETE endpoint.

Example URL

.../services/rest/audit/105/checklist/7

Response Details

If successful, an HTTP 200 response is sent.

Error Messages

Element	Message	Meaning
auditId	HTTP 404 Not Found	Invalid {auditId} - not numeric
auditId	HTTP 417 IllegalStateException: Cannot find the Audit to return with id:<###>	Invalid {auditId} - not found
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Audit Checklist to return with id:<###>	Invalid {id} - not found

DataPrivacyService

This section describes the API for executing Data Privacy requests. The following functions are available:

- **Right to Access:** retrieves personal data from the system relating to the name of an individual
- **Right to be Forgotten:** erases personal data from the system for an individual

Although minimal personal data is held in Supplier Evaluation, it does include the names and contact details of retailer/portal owner, supplier, and third-party users. The Data Privacy API provides a means for the retailer/portal owner to:

1. Request details of personal data relating to an individual be held within the system.
2. Request that the personal data relating to the individual be removed from the system.

For additional detail on using the Data Privacy API, including examples, see [Appendix: Using the Data Privacy API](#).

Right to Access

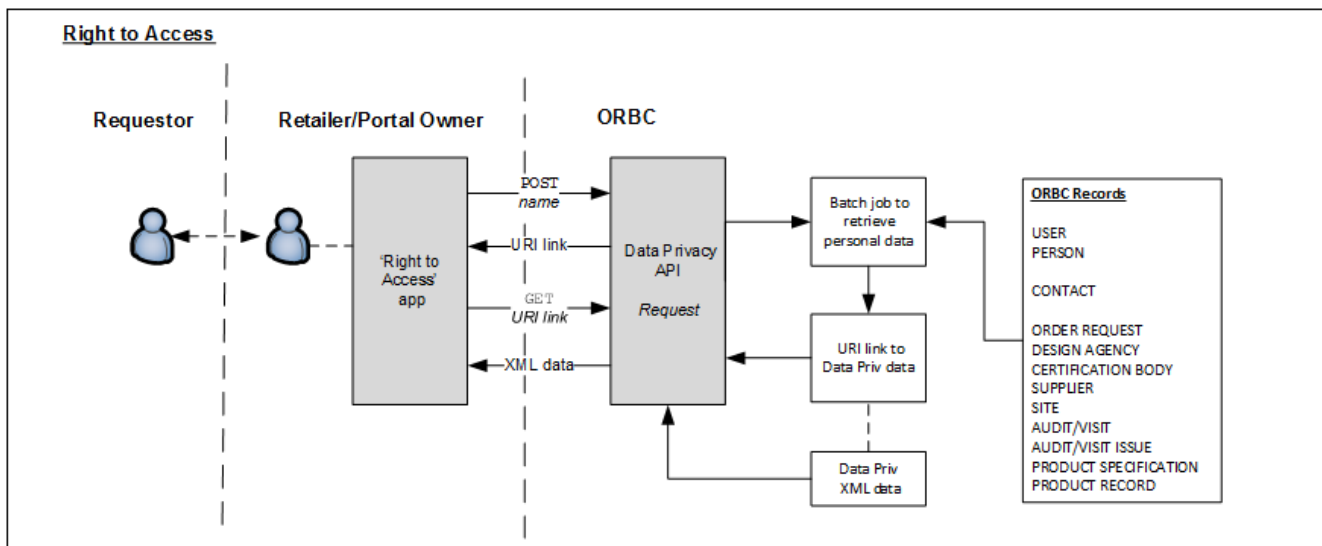
Description

Retrieves a set of personal data that relates to the name of an individual. Use this function to perform Right to Access requests for the following scenarios:

- An employee of the retailer/portal owner requests an electronic copy of their personal data that is held within the system.
- An employee of one of the retailer/portal owner's suppliers requests an electronic copy of their personal data that is held within the system.
- An employee of the retailer/portal owner's partner organizations requests an electronic copy of their personal data that is held within the system.

The API's access function allows the name of the individual to be passed, with an XML message being returned containing any personal data found for that name. The XML message is a structured text format, which is both machine and human readable.

Figure 4-5 Right to Access Process



The steps for completing a Right to Access request are as follows (for further details, see [Appendix: Using the Data Privacy API](#)):

1. Submit a POST call, passing the name of the individual.

A batch job is submitted within Supplier Evaluation to search the system for personal data for the presence of the name (for which records and fields are searched, see [Appendix:](#)

[Using the Data Privacy API](#)). On completion, a Data Privacy record containing the retrieved personal data is created within Supplier Evaluation.

A link to the Data Privacy record is returned as the response to the POST call.

2. On receipt of the response to the POST call, submit a GET call, passing the returned URI link.

 Note:

If the GET call is submitted before the batch job has finished compiling the personal data, just a header will be returned, and the GET call should be retried.

A message containing the personal data is returned. The XML data is structured within a `dataprivRequestDTO` root element (for details of the message structure, see [Appendix: Using the Data Privacy API](#)).

Just the personal data fields and values to identify the record are returned, not the full record contents.

Addresses may contain multiple `localeData` elements for the language translation of countries.

XML elements will only be present if they contain data.

3. Assess whether the returned data relates to the individual who made the request, editing and/or reformatting it if necessary, before passing it to the requestor.

Endpoint address: `/services/rest/datapriv/access` **HTTP methods:** POST, GET

Request Details

For the POST method, a `name` parameter is passed in the payload.

For the GET method, an `id` parameter is passed as a URI parameter.

URI Parameters

Example URL (POST method): `.../services/rest/access/`

Example URL (GET method): `.../services/rest/access/?id=16`

Response Details

For a successful response to the POST method, a link to the Data Privacy record is returned.

For a successful response to the GET method, XML is returned with a `dataprivRequestDTO` root element containing an element for each record where the personal data has been found.

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Right to be Forgotten

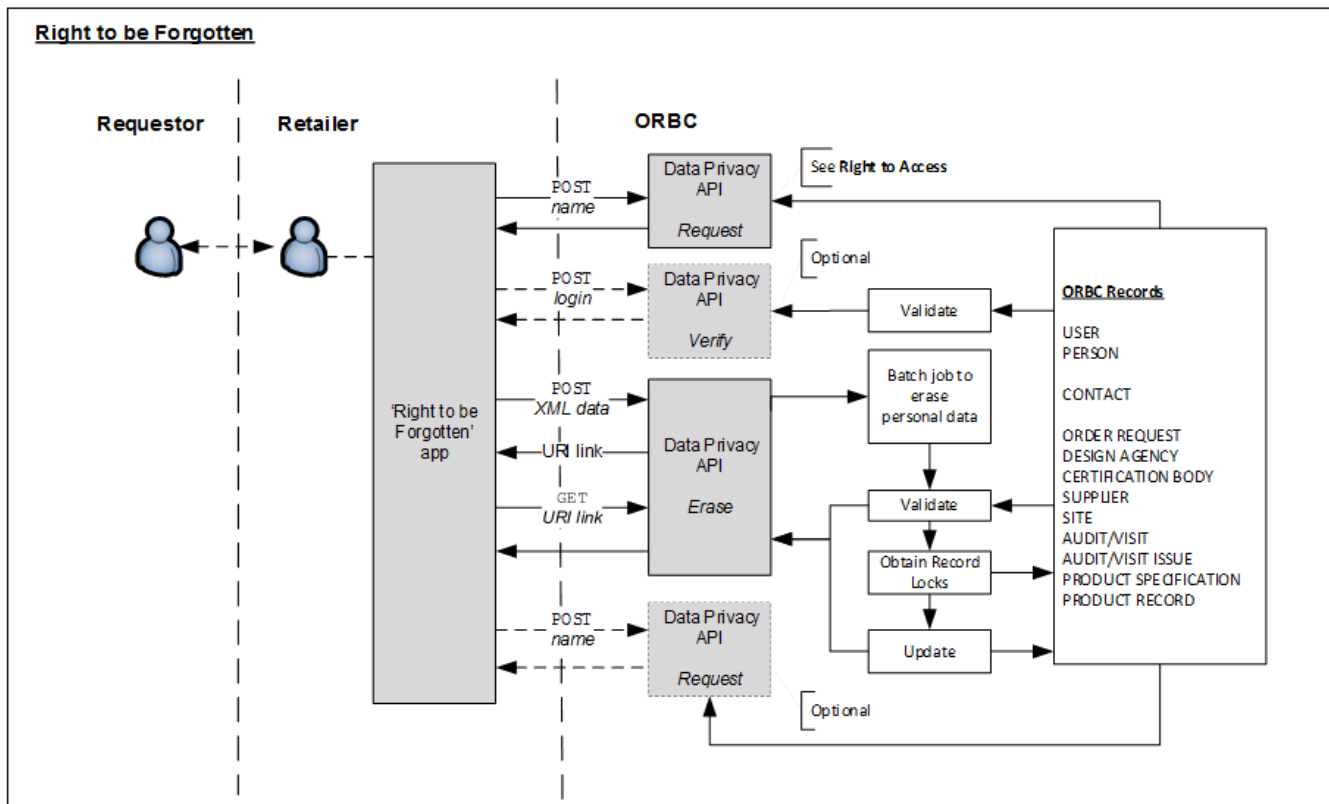
Description

Erases a set of personal data that relates to an individual. Use this function to perform Right to be Forgotten requests for the following scenarios:

- An employee of the retailer/portal owner requests their personal data be erased from the system.
- An employee of one of the retailer/portal owner's suppliers requests their personal data be erased from the system.
- An employee of a partner organizations requests their personal data be erased from the system.
- The retailer/portal owner initiates the purging of an inactive user's personal data from the system.

An erase function allows the XML message returned from a Right to Access request to be used to erase the data.

Figure 4-6 Right to be Forgotten Process



The steps for completing a Right to be Forgotten request are as follows:

1. The Right to Access service must be used to retrieve the personal data for the individual. For details, see [Right to Access](#).
2. View, and if necessary, edit the XML returned from the Right to Access request.
As there may be any number of users with the same name, it is necessary to decide which data represents the individual who made the request, removing any that is not relevant (for further information on editing the XML message, see [Appendix: Using the Data Privacy API](#)).
3. Optionally submit a POST call to the verify function, passing the login id of the individual (obtained from the XML message returned in Step 2).

This step is used to check whether the erasure of data for the individual will be permitted, before actually submitting the request. The erase will not be permitted if the individual is responsible for the completion of a supplier site's Assessment.

XML is returned with a `dataprivVerifyDTO` root element containing a status and a message element. If the call is successful the message will contain *Success*, otherwise it will contain *Failure* and the reason for the failure will be present in the message element.

4. Submit a POST call to the erase function, passing the XML retrieved from the Right to Access request (with any edits having been applied).

A batch job is submitted to validate the request, verify that the erasure is permitted (that is, the individual is not responsible for the completion of an Assessment), and obtain locks on the records to be updated.

5. If the validation fails, or a lock cannot be obtained, the process is terminated and an XML is returned with a `dataprivErasureDTO` root element containing a message element that contains the reason for the failure, and the status element set to *Failure*.

If no exceptions are encountered, a Data Privacy record containing the personal data to erase is created and a link to the record is returned as the response to the POST call.

6. On receipt of the response to the POST call, submit a GET call to the erase function, using the returned URI link.

The update processing replaces the data passed in the XML message with 30 characters of randomly-generated text. Email addresses are changed to `x@example.com`, where `x` is 15 characters of randomly-generated text*. GPS coordinate numbers and country selections are blanked.

Some fields have a specific length, in which case the 30 characters of obfuscated text is overridden:

- Name 50 characters
- Login 20 characters
- Address lines 20 characters
- Post Code 5 characters

* Email addresses that are stored as free text are fully obfuscated, rather than set to `x@example.com`.

Only certain fields are permitted to be erased (for details, see [Appendix: Using the Data Privacy API](#)).

When updating a user account, the account is set to deactivated, and the password is set to expired.

 Note:

If the GET call is submitted before the batch job has finished compiling the personal data, just a header will be returned, and the GET call should be retried.

The erasure of data is not reversible.

7. Optionally submit another Right to Access request to check that the data has been erased as expected.
8. Inform the requestor of the outcome of their request.

The Right to be Forgotten facility may also be used for purging user accounts that are no longer active.

Endpoint address: /services/rest/datapriv/erase **HTTP methods:** POST (verify and erase), GET (erase)

Request Details

For the verify POST method, a loginId parameter is passed in the payload.

For the erase POST method, an XML message is passed in the payload.

For the erase GET method, an id parameter is passed as a URI parameter.

URI Parameters

Example URL (verify POST method): .../services/rest/verify/

Example URL (erase POST method): .../services/rest/erase/

Example URL (erase GET method): .../services/rest/erase/?id=32

Response Details

For a successful response to the verify POST method, XML is returned with a dataprivVerifyDTO root element containing a status of *Success*.

For a successful response to the erase POST method, a link to the Data Privacy record is returned.

For a successful response to the erase GET method, an HTTP 200 response is sent with a dataprivErasureDTO root element containing a status of *Success*.

Error Responses

For a failure response to the verify method, XML is returned with a dataprivVerifyDTO root element containing a status of *Failure*, with the reason in the message element.

For a failure response to the erase method, XML is returned with a dataprivErasureDTO root element containing a status of *Failure*, with the reason in the message element.

Error Messages

Element	Message	Meaning
user	Only one User can be erased at a time	More than one User was passed in to the erase function. The XML must only relate to a single user.
person	Only one Person can be erased at a time	More than one Person was passed in to the erase function. The XML must only relate to a single person.
id	The User is linked to Person with ID: %id but the Person included has ID: %id	The wrong Person has been included. %id is the id taken from the XML.
id	The Contacts are linked to Person with ID: %id but the Person included has ID: %id	The wrong Person has been included. %id is the id taken from the XML.
id	The Contact with ID %id is linked to a different User than the one being erased	The Contacts passed do not match the user being erased. %id is replaced with the id from the XML.

Element	Message	Meaning
person	All Contacts being erased must be linked to the same Person	Where no User is being erased, all Contacts being erased must be for the same person.
person	Please include Person with ID: %id	If User or Contacts are passed, the matching Person must also be included.
id	Cannot find entity of type: %rec and ID: %id	An id is passed that is not found. %rec is the internal name of the record, such as ProductRecord. %id is the id taken from the XML.
id	Entity of type: %rec and ID: %id is currently locked, please try again later	One of the records to be erased is currently locked. %rec is the internal name of the record, such as ProductRecord. %id is the id taken from the XML. Retry the request until the record becomes available.
id	Entity of type: %rec with id: %id has been edited during the erase process, please verify data	One of the records updated has been edited between up front validation and the processing of that particular record. %rec is the internal name of the record, such as ProductRecord. %id is the id taken from the XML.
user	The user cannot be erased because they are responsible for the approval of an Assessment	The user is named as being responsible for completing an assessment, so they cannot be erased.
login id	Cannot find User with login ID: %id	No User account found for the passed login id. %id is the login id.
	The erase has failed due to unexpected internal error	Other unexpected condition has occurred.

AttachmentRestService

This section describes the API for retrieving the file attachments from certain records. The following function is available:

- [Retrieve Report Attachment](#): retrieves attachment files that have been generated by the system as report outputs
- [Retrieve Attachment](#): retrieves files that have been attached to records

Record types that have an attachments element contain a globalId element for each of the record's file attachments. The Retrieve Attachment service can be used to retrieve the actual attachment files by passing globalId as a parameter.

The record types that are capable of returning attachments data are:

- Users
- Suppliers
- Sites
- Audits

- Audit Issues (Non Conformances)
- Scorecards
- Projects
- Activities

The following attributes are included in the attachment element of the XML:

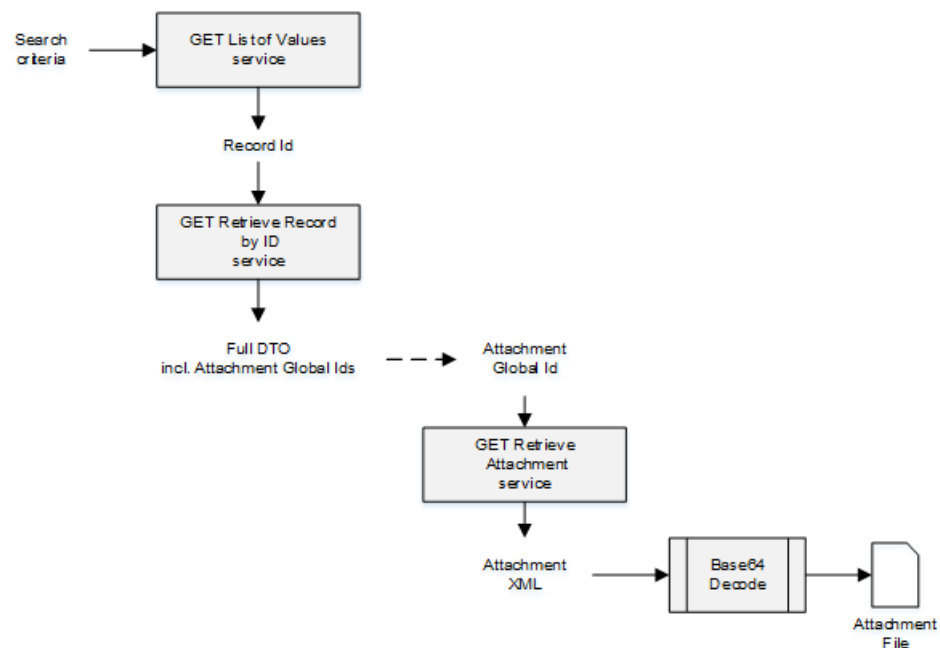
- ID
- Record Link (global Id for the attachment file)
- File Name/URL
- Description
- Bytes
- Attached by ID
- Attached On

The Attachments API is GET (retrieve) only, there is no PUT/POST (update/create) capability.

Retrieving a file returns a fileName element containing the file name, plus the encoded attachment. Retrieving a URL returns a uri element containing the URL.

The following describes the steps for retrieving file attachments:

Figure 4-7 Steps for Retrieving Attachments



- Call the List of Values service for the record with appropriate filters to locate the records for which attachments are to be retrieved. Record ids are returned for any records that match search criteria.
- Call the Retrieve Record by ID service for each record id returned from the List of Values call. The record's full DTO element is returned, containing the record's data, including an

attachments element which has a globalId element for each of the record's file attachments.

- Call the Retrieve Attachment service passing the globalId as a parameter. The attachment file's data is returned as a Base 64 text string. Use a suitable decoder to generate the actual attachment file.

Retrieve Report Attachment

Description

Retrieves an attachment file that has been generated as the output of a report. The file types will be PDF, Excel, RTF, or HTML. When a report is generated, the outputs are stored in the Report Outputs area, grouped within a hierarchical folder structure.

The API provides a means of automatically locating reports for further analysis of the data within an external system. The ID of the attachment to be retrieved is sent to the external system's designated email address.

For additional detail on using the Report Attachment API, see [Appendix: Using the Report Attachment API](#).

Endpoint address: /services/rest/attachment/report/{globalId} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {globalId} parameter that determines the record to retrieve.

Example URL

.../services/rest/attachment/report/f25eebdd-e7c5-4bb2-b61c-f35171438386

Response Details

For a successful response, XML is returned with a data element which contains the actual attachment as a Base64 encoded text string. The string must then be processed by a Base64 decoder to convert it to the actual report file.

Error Messages

Element	Message	Meaning
globalId	HTTP 404 Not found	Invalid {globalId} - not numeric
globalId	HTTP 417 IllegalStateException: Cannot find the Attachment to return with id:<###>	Invalid {globalId} - not found

Retrieve Attachment

Description

Retrieves a file that has been attached to a record. The files types will typically be documents or images.

Endpoint address: /services/rest/attachment/attachment/{globalId} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {globalId} parameter that determines the record to retrieve.

Example URL

.../services/rest/attachment/attachment/RTxy92Ha4lsQE6Bjc-18t1wJ7smbYoCH

Response Details

For a successful response, XML is returned with a data element which contains the actual attachment as a Base64 encoded text string. The string must then be processed by a Base64 decoder to convert it to the actual attachment file.

Error Messages

Element	Message	Meaning
globalId	HTTP 404 Not found	Invalid {globalId} - not numeric
globalId	HTTP 417 IllegalStateException: Cannot find the Attachment to return with id:<###>	Invalid {globalId} - not found

ScorecardRestService

This section describes the API for managing assessments. The following functions are available:

- [List of Values](#): retrieves a list of assessments
- [Retrieve Record by ID](#): retrieves an Assessment record using its unique identifier
- [Retrieve Record by Business Key](#): retrieves an Assessment record using its business key
- [Check Record Modification Timestamp](#): retrieves the timestamp when an Assessment record was last updated
- [Update Record](#): updates an existing Assessment record
- [Update Sub Status](#): updates the sub status of an existing Assessment record

List of Values

Description

Retrieves a list of assessments in a paged list. Use this function to locate Assessment record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/scorecard **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30
status	Optional	string	Yes	Yes	IN PROGRESS~COM PLETE
siteStatus	Optional	string	Yes	Yes	AWAITING REGISTRATION
siteType	Optional	string	Yes	Yes	PRIMARY
siteCode	Optional	string	Yes	Yes	A0001-0001
siteName	Optional	string	Yes	Yes	West Road Site
siteNameBusiness Language	Optional	string	Yes	Yes	West Road Site
supplierCode	Optional	string	Yes	Yes	A0001
supplierName	Optional	string	Yes	Yes	ABC Ltd
supplierNameBusin essLanguage	Optional	string	Yes	Yes	ABC Ltd
businessCategory	Optional	string	Yes	Yes	POTATOES_LOOS E
templateCode	Optional	string	Yes	Yes	ETHICAL_ASSES SMENT
dueDate	Optional	string	Yes	No	2015-05-19
modifiedSince	Optional	string	No	No	2015-05-19 13:30:39
modifiedUntil	Optional	string	No	No	2015-05-19 13:30:39

Example URLs

.../services/rest/scorecard/?offset=2&pageSize=20 .../services/rest/scorecard/?status=IN
PROGRESS~COMPLETE .../services/rest/scorecard/?
businessCategory=POTATOES_LOOSE

Response Details

For a successful response, XML is returned with a ScorecardLinkList root element containing an entries element for each matched assessment. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Assessment record's internal ID
recordLink	string	URI to the ScorecardRestService Retrieve record service for this assessment
scorecardCode	string	Assessment's code business key
siteCode	string	Assessment's site code

Element	Type	Description
supplierCode	string	Assessment's supplier code
templateCode	string	Assessment's template code

The returned XML also contains a totalRecords element, which states the total number of retrievable records that match the filter parameters.

Error Messages

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record by ID

Description

Retrieves a single Assessment record's details using the record's internal unique ID (which is not visible in the UI). Use this function to retrieve the full details of an individual assessment.

Endpoint address: /services/rest/scorecard/{id} **HTTP method:** GET

Request Details

The URL contains the {id} parameter that determines the record to retrieve.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a localeData element returned for the specified locales if data is present.

Example URL

.../services/rest/scorecard/54

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a scorecardFullDTO root element containing the individual attributes of the requested Assessment record. If an ID is not specified, a list of all assessments is returned (per the List of Values function).

File Structure

The main elements of the scorecardFullDTO root element:

- scorecardTemplateSnapshot and Template - the settings from the associated template.

- scorecardQuestions - the assessment's questions and their answers - a separate element for each.
- supplier and site - details of the supplier and site that the assessment is related to.
- udfData - the contents of any user-defined custom fields associated to the Assessment record.
- attachments - a child element containing the ids of any file attachments the assessment has.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Assessment to return with id:<####>	Invalid {id} - not found

Retrieve Record by Business Key

Description

Retrieves a single Assessment record's details using its business key (code). Use this function to retrieve the full details of an individual assessment using its key code.

Endpoint address: /services/rest/scorecard/byKey/{code} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {code} parameter that determines the record to retrieve.

Example URL

.../services/rest/scorecard/byKey/184

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a scorecardFullDTO root element containing the individual attributes of the requested Assessment record.

File Structure

The main elements of the scorecardFullDTO root element:

- scorecardTemplateSnapshot and Template - the settings from the associated template.
- scorecardQuestions - the assessment's questions and their answers - a separate element for each.
- supplier and site - details of the supplier and site that the assessment is related to.
- udfData - the contents of any user-defined custom fields associated to the Assessment record.

- attachments - a child element containing the ids of any file attachments the assessment has.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
code	HTTP 404 Not found	Invalid {code} - blank or not found

Check Record Modification Timestamp

Description

Retrieves the last modification date and time of an Assessment record. Use this function to determine when a assessment was last updated.

Endpoint address: /services/rest/scorecard/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/scorecard/54

Response Details

If successful, an HTTP 200 response is sent containing the Last-Modified header showing the date and time of the last update of the requested Assessment record. For example:

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Messages

Element	Message	Meaning
id	HTTP 417 Expectation Failed	Invalid {id} - not found
id	HTTP 404 Not found	Invalid {id} - not numeric

Update Record

Description

Updates an existing Assessment record. Use this function to update an Assessment's details in Supplier Evaluation based on data sourced from the external system..

Endpoint address: /services/rest/scorecard/{id} **HTTP method:** PUT

Request Details

The body of the request contains a ScorecardFullDTO to specify the updates to the Assessment record, which is based on the ScorecardFullDTO type returned when retrieving an Assessment.



Note:

When updating records, all values must be included. If a value or element is omitted from the request, the field contents will be cleared on the Assessment record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 2 - Supplier* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a ScorecardRecordLink element. The ScorecardRecordLink element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Assessment record's internal ID
recordLink	string	URI to the Assessment record, for use in a GET request
scorecardCode	string	Assessment's code business key
siteCode	string	Assessment's site code
supplierCode	string	Assessment's supplier code
templateCode	string	Assessment's template code

Error Responses

If the supplied data does not result in a valid Assessment (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Assessment Questions & Answers

This section explains the basic structure of the Scorecard API's payload XML, focussing on the elements which identify the Assessment and contain its Questions and Answers.

The key elements in the parent scorecardFullDTO node which are of relevance when responding to Assessments are as follows.

```
<scorecardFullDTO>
  <autoGenerationEnabled>
  <businessCategories>
  <calculatedScore>
  <code>           The Assessment's business key.
  <dueDate>        The date by which the Assessment is due to be completed.
  <extraScorecard>
```

```

<overdue>
<period>
<personResponsible>
<quarterEndDate>
<quarterStartDate>
<scheduled>
<scoreTemplate>    The Assessment's questions & answers - see Questions and Answers
below.
<scorecardScore>
<scorecardValue>
<status>            The status of the Assessment - see Status and Sub Status below.
<statusHistory>
<supplierGenerated>
<udfData>
<week>
<year>
<id>                The Assessment's record Id.
<createdOn>         When the Assessment was created.
<updatedOn>         When the Assessment was last updated.
<due>
<scorecardTarget>   The Assessment's target supplier & site - see Supplier and Site below.
<subStatusText>     The sub status of the Assessment.
<automaticDisqualificationCount>
<scorecardTargetSites>
<scorecardTargetSupplier>
<createdFromAssessmentRequestWizard>

```

Supplier and Site

The scorecardTarget element identifies the target supplier and site.

```

<ns1:scorecardTarget>
  <ns1:site>
    <ns2:code>A0072-00020</ns2:code>
    <ns2:localName>BS Site 123</ns2:localName>
    <ns2:name>BS Site 123</ns2:name>
    <ns2:id>813</ns2:id>
    <ns2:supplierId>490</ns2:supplierId>
    <ns2:supplierCode>A0072</ns2:supplierCode>
  </ns1:site>
</ns1:scorecardTarget>

```

Additional supplier name and contact information is included in the scorecardTargetSupplier element.

Status and Sub Status

The status element contains the Assessment's current status, within the status child element, for example:

```

<ns1:status>
  <ns2:applyTemplateChanges>>false</ns2:applyTemplateChanges>
  <ns2:localeData>
    <ns1:description>En cours</ns1:description>
    <ns1:locale>fr</ns1:locale>
    <ns1:id>15</ns1:id>
  </ns2:localeData>
  <ns2:localeData>
    <ns1:description>In behandeling</ns1:description>
    <ns1:locale>nl</ns1:locale>
    <ns1:id>21</ns1:id>
  </ns2:localeData>

```

```

        </ns2:localeData>
    <ns2:localeData>
        <ns1:description>In Progress</ns1:description>
        <ns1:id>14</ns1:id>
    </ns2:localeData>
    <ns2:status>IN PROGRESS</ns2:status>
    <ns2:id>2</ns2:id>
</ns1:status>

```

Multiple localeData nodes represent the language translations of the status description. The base language (English) description has no locale node. The status child element contains the status code.

The subStatusText node contains the sub status of the Assessment.

Questions and Answers

The scoreTemplate element contains the main body of the Assessment, including its questions and answers.

The multiple localeData nodes represent the language translations of the Assessment description, based on its template. The base language (English) description has no locale node. The id node contains the record id of the Assessment's template.

An Assessment's questions are represented by universalQuestions, generalQuestions or adHocQuestions elements. The structure of each is the same, the difference being how they were configured within the Assessment's template: Universal questions appear by default in all templates; General questions are added to the template from a glossary of standard questions; Ad Hoc questions are specific to the template, for that type of Assessment.

A question and its answer are structured as follows:

- Multiple localeData nodes represent the language translations of the Question text. The base language (English) description has no locale node. The questionId node contains the Question's unique id.
- scorecardAnswerType indicates the type of question. The possible types are TEXT, NUMERIC, DATE, URL, RADIO_BUTTON, CHECKBOX, SINGLE_PICKER, MULTI_PICKER and TABULAR.
 - For Text, Numeric, Date and URL answer types, a single scorecardQuestionAnswers element contains the response, within an answerText node.

```

<ns1:scorecardAnswerType>TEXT</ns1:scorecardAnswerType>
<ns1:scorecardQuestionAnswers>
    <ns1:answerText>text response</ns1:answerText>
    <ns1:id>25702</ns1:id>
    <ns1:sequence>0</ns1:sequence>
</ns1:scorecardQuestionAnswers>

```

- For Radio Button answer types, a single scorecardQuestionAnswers element contains the response, within an answerText node. Multiple scorecardAvailableAnswers elements represent the available answer options. Also copy the relevant scorecardAvailableAnswers element to scorecardQuestionAnswers

```

<ns1:scorecardAnswerType>RADIO_BUTTON</ns1:scorecardAnswerType>
<ns1:scorecardQuestionAnswers>
    <ns1:answerText>Yes</ns1:answerText>
    <ns1:scorecardAvailableAnswer>
        <ns1:id>25706</ns1:id>
        <ns1:sequence>0</ns1:sequence>
    </ns1:scorecardAvailableAnswer>
</ns1:scorecardQuestionAnswers>

```

```
...
<ns1:scorecardAvailableAnswers>
<ns1:scorecardAvailableAnswers>
```

- For Checkbox answer types, one or more scorecardQuestionAnswers elements contain the responses, each within an answerText node. Multiple scorecardAvailableAnswers elements represent the available answer options. Also copy the relevant scorecardAvailableAnswers elements to scorecardQuestionAnswers.

```
<ns1:scorecardAnswerType>CHECKBOX</ns1:scorecardAnswerType>
  <ns1:scorecardQuestionAnswers>
    <ns1:answerText>Fresh</ns1:answerText>
    <ns1:comment>Locally sourced only</ns1:comment>
    <ns1:scorecardAvailableAnswer>
      <ns1:id>25703</ns1:id>
      <ns1:sequence>0</ns1:sequence>
    </ns1:scorecardQuestionAnswers>
  </ns1:scorecardQuestionAnswers>
  <ns1:scorecardQuestionAnswers>
    <ns1:answerText>Chilled</ns1:answerText>
    <ns1:scorecardAvailableAnswer>
      <ns1:id>25704</ns1:id>
      <ns1:sequence>0</ns1:sequence>
    </ns1:scorecardQuestionAnswers>
  </ns1:scorecardQuestionAnswers>
  ...
  <ns1:scorecardAvailableAnswers>
  <ns1:scorecardAvailableAnswers>
  <ns1:scorecardAvailableAnswers>
```

- For Single-select Picker answer types, a single scorecardQuestionAnswers element contains the response, within a single scorecardPicklistSingle child element. Multiple scorecardAnswerOptionDetail elements within scorecardPicklistSnapshot represent the available answer options. Copy the contents of the relevant scorecardAnswerOptionDetail element to scorecardPicklistSingle to populate the response.

```
<ns1:scorecardAnswerType>SINGLE_PICKER</ns1:scorecardAnswerType>
  <ns1:scorecardQuestionAnswers>
    <ns1:id>25707</ns1:id>
    <ns1:scorecardPicklistSingle>
      <ns2:mandatoryComments>false</ns2:mandatoryComments>
      <ns2:localeData>
        <ns1:description>Frozen</ns1:description>
        <ns1:id>105522</ns1:id>
      </ns2:localeData>
      <ns2:code>FROZEN</ns2:code>
      ...
    </ns1:scorecardPicklistSingle>
    <ns1:sequence>0</ns1:sequence>
  </ns1:scorecardQuestionAnswers>
  ...
  <ns1:scorecardPicklistSnapshot>
    ...
    <ns1:scorecardAnswerOptionDetail>
    <ns1:scorecardAnswerOptionDetail>
    <ns1:scorecardAnswerOptionDetail>
  </ns1:scorecardPicklistSnapshot>
```

- For Multi-select Picker answer types, one or more scorecardPicklistMulti elements contains the responses, within a single scorecardQuestionAnswers element. Multiple scorecardAnswerOptionDetail elements within scorecardPicklistSnapshot represent the available answer options. Copy the contents of the relevant scorecardAnswerOptionDetail elements to scorecardPicklistMulti to populate the responses.

```

<ns1:scorecardAnswerType>MULTI_PICKER</ns1:scorecardAnswerType>
<ns1:scorecardQuestionAnswers>
  <ns1:id>25708</ns1:id>
  <ns1:scorecardPicklistMulti>
    <ns2:mandatoryComments>>false</ns2:mandatoryComments>
    <ns2:localeData>
      <ns1:description>Frozen</ns1:description>
      <ns1:id>105524</ns1:id>
    </ns2:localeData>
    <ns2:code>FROZEN</ns2:code>
    ...
  </ns1:scorecardPicklistMulti>
  <ns1:scorecardPicklistMulti>
    <ns2:mandatoryComments>>false</ns2:mandatoryComments>
    <ns2:localeData>
      <ns1:description>Chilled</ns1:description>
      <ns1:id>105525</ns1:id>
    </ns2:localeData>
    <ns2:code>CHILLED</ns2:code>
    ...
  </ns1:scorecardPicklistMulti>
  <ns1:sequence>0</ns1:sequence>
</ns1:scorecardQuestionAnswers>
...
<ns1:scorecardPicklistSnapshot>
  ...
  <ns1:scorecardAnswerOptionDetail>
  <ns1:scorecardAnswerOptionDetail>
  <ns1:scorecardAnswerOptionDetail>
</ns1:scorecardPicklistSnapshot>

```

- For Tabular answer types, a single scorecardQuestionAnswers defines the answer type of each column of the table, and contains the responses.

The column types may be text, numeric, URL or single-select picker, defined in in columnxAnswerType nodes, where x is the column number.

The responses are contained within nodes which identify the column and answer type:

- * Text answers in a cellAnswerx node, where x is the column number.
- * Numeric answers in a numberAnswerx node, where x is the column number.
- * URL answers in a urlAnswerx node, where x is the column number.
- * Single-select picker answers within a scorecardPicklistx element, where x is the column number. Multiple scorecardAnswerOptionDetail elements within a corresponding picklistSnapshotx element represent the available answer options. Copy the contents of the relevant scorecardAnswerOptionDetail element to scorecardPicklistx to populate the response.

The following example is a table of four columns (text, numeric, URL, single-select picker), populated with one row:

Site Certifications	Certification Body	Certification Year	Product Category
Red Tractor	rt@example.com	2020	Poultry

```

<ns1:scorecardAnswerType>TABULAR</ns1:scorecardAnswerType>
<ns1:scorecardQuestionAnswers>
  <ns1:id>25709</ns1:id>
  <ns1:column1AnswerType>TEXT</ns1:column1AnswerType>
  <ns1:column2AnswerType>URL</ns1:column2AnswerType>
  <ns1:column3AnswerType>NUMERIC</ns1:column3AnswerType>

```

```

<ns1:column4AnswerType>SINGLE_PICKER</ns1:column4AnswerType>
<ns1:cellAnswer1>Red Tractor</ns1:cellAnswer1>
<ns1:numberAnswer3>2020</ns1:numberAnswer3>
<ns1:scorecardPicklist4>
  <ns2:mandatoryComments>false</ns2:mandatoryComments>
  <ns2:localeData>
    <ns1:description>Poultry</ns1:description>
    <ns1:id>105527</ns1:id>
  </ns2:localeData>
  <ns2:code>POULTRY</ns2:code>
  ...
</ns1:scorecardPicklist4>
<ns1:urlAnswer2>https://rt@example.com</ns1:urlAnswer2>
<ns1:sequence>0</ns1:sequence>
</ns1:scorecardQuestionAnswers>
...
<ns1:picklistSnapshot4>
  <ns2:code>CATEGORIES</ns2:code>
  <ns2:localeData>
    <ns1:description>Product Categories</ns1:description>
    <ns1:id>58276</ns1:id>
  </ns2:localeData>
  <ns2:id>2095</ns2:id>
  <ns1:scorecardAnswerOptionDetail>
    <ns2:mandatoryComments>false</ns2:mandatoryComments>
    <ns2:localeData>
      <ns1:description>Dairy</ns1:description>
      <ns1:id>105526</ns1:id>
    </ns2:localeData>
    <ns2:code>DAIRY</ns2:code>
    ...
  </ns1:scorecardAnswerOptionDetail>
  <ns1:scorecardAnswerOptionDetail>
    <ns2:mandatoryComments>false</ns2:mandatoryComments>
    <ns2:localeData>
      <ns1:description>Poultry</ns1:description>
      <ns1:id>105527</ns1:id>
    </ns2:localeData>
    <ns2:code>POULTRY</ns2:code>
    ...
  </ns1:scorecardAnswerOptionDetail>
</ns1:picklistSnapshot4>

```

- A comment node can be included with the answer for any supporting information. If commentsMandatory is true, a comment is mandatory.
- When an answer has an alternative Summary Attribute label, a summaryLabel node is present with the answer's description node.

Update Sub Status

Description

Updates the sub status of an existing Assessment record. Use this function to update an Assessment's sub status, such as when being progressed by an Accredited Body as part of third-party integration for the sourcing of data from an external system.

Two alternative endpoints are available, to locate the Assessment either by its record id or by its Code business key. They both return a record link to the Assessment.

Endpoint address: /services/rest/scorecard/{id}/substatus and /services/rest/scorecard/{byKey}/substatus

HTTP method: PUT

Request Details

The body of the request of the request is an assessmentSubStatusDTO message containing the sub status of the Assessment and any comments.

Example Request XML

```
<ns1:assessmentSubStatusDTO
  xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
  xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
  <ns1:substatus>RETRIEVED</ns1:substatus>
  <ns1:comments></ns1:comments>
</ns1:assessmentSubStatusDTO>
```

An Assessment's sub status is initially ISSUED; once assigned to a third party Accredited Body it becomes PENDING, making it available for the designated third party to retrieve. The workflow requires that before the third party submits their response data they first acknowledge they have retrieved the Assessment by setting the sub status to RETRIEVED.

As the regular Update (PUT) endpoint requires the full Assessment payload to be included, the Update Sub Status endpoint provides a simpler method of just returning the sub status value, locating the Assessment by its record id or business key.

Subsequently, the external system will submit the full Assessment payload with their responses using the existing regular Update (PUT) endpoint, setting the sub status RESPONDED, indicating their response is complete.

When called by the external system of a registered Accredited Body, only Assessments that are assigned to that Accredited Body can be updated.

For additional detail on using the Scorecard API for third party integration, see [Appendix: LCA Third-Party Integration](#)

Response Details

If successful, an HTTP 200 response is sent with a body containing a ScorecardRecordLink element. The ScorecardRecordLink element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Assessment record's internal ID
recordLink	string	URI to the Assessment record, for use in a GET request
scorecardCode	string	Assessment's code business key
siteCode	string	Assessment's site code
supplierCode	string	Assessment's supplier code
templateCode	string	Assessment's template code

Error Responses

If the supplied data does not result in a valid Assessment (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

ProjectRestService

This section describes the API for managing processes. The following functions are available:

- [List of Values](#): retrieves a list of processes
- [Retrieve Record by ID](#): retrieves a Process record using its unique identifier
- [Retrieve Record by Business Key](#): retrieves a Process record using its business key
- [Check Record Modification Timestamp](#): retrieves the timestamp when a Process record was last updated
- [Update Record](#): updates an existing Process record

List of Values

Description

Retrieves a list of processes in a paged list. Use this function to locate Process record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/project **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30
projectStatus	Optional	string	Yes	Yes	IN PROGRESS~COM PLETED
templateCode	Optional	string	Yes	Yes	NEW_BAKERY_R ANGE
masterProject	Optional	Boolean	No	No	TRUE
parentProject	Optional	string	Yes	Yes	NEW_LINE_DEVE LOPMENT
projectType	Optional	string	Yes	Yes	CHANGE OF SUPPLIER
projectName	Optional	string	Yes	Yes	New Potato Product Launch
projectManager	Optional	string	Yes	Yes	John Smith
businessCategory	Optional	string	Yes	Yes	POTATOES_LOOS E
specificationType	Optional	string	Yes	Yes	FOOD
siteStatus	Optional	string	Yes	Yes	AWAITING REGISTRATION
siteType	Optional	string	Yes	Yes	PRIMARY

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
siteCode	Optional	string	Yes	Yes	A0001-0001
siteName	Optional	string	Yes	Yes	West Road Site
siteNameBusiness Language	Optional	string	Yes	Yes	West Road Site
supplierCode	Optional	string	Yes	Yes	A0001
supplierName	Optional	string	Yes	Yes	ABC Ltd
supplierNameBusin essLanguage	Optional	string	Yes	Yes	ABC Ltd
modifiedSince	Optional	string	No	No	2015-05-19 13:30:39
modifiedUntil	Optional	string	No	No	2015-05-19 13:30:39

Example URLs

.../services/rest/user/?offset=2&pageSize=20 .../services/rest/user/?projectStatus=IN
PROGRESS~COMPLETED .../services/rest/user/?
templateCode=NEW_LINE_DEVELOPMENT

Response Details

For a successful response, XML is returned with a ProjectLinkList root element containing an entries element for each matched process. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Process record's internal ID
recordLink	string	URI to the ProjectRestService Retrieve record service for this process
projectId	string	Process's business key
projectName	string	Process's name
parentProjectId	string	Parent process's business key
parentProjectName	string	Parent process's name

The returned XML also contains a totalRecords element, which states the total number of retrievable records that match the filter parameters.

Error Messages

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record by ID

Description

Retrieves a single Process record's details using the record's internal unique ID (which is not visible in the UI). Use this function to retrieve the full details of an individual process.

Endpoint address: /services/rest/project/{id} **HTTP method:** GET

Request Details

The URL contains the {id} parameter that determines the record to retrieve.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a localeData element returned for the specified locales if data is present.

Example URL

.../services/rest/project/20

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a projectFullDTO root element containing the individual attributes of the requested Process record. If an ID is not specified, a list of all processes is returned (per the List of Values function).

File Structure

The main elements of the projectFullDTO root element:

- projectTemplateSnapshot and Template - the settings from the associated template.
- briefs - the associated process briefs; a separate element for each.
- teams - the users and their roles associated to the process.
- linkedRecords - the associated linked records; a separate element for each.
- udfData - the contents of any user-defined custom fields associated to the Process record.
- attachments - a child element containing the ids of any file attachments the process has.

Processes and their activities can be nested within a hierarchy. In the absence of a function to extract the full hierarchy with a single GET call, to retrieve the hierarchy of a process a number of successive calls is required. The Parent Project field is a Boolean value which will be blank for processes that have no hierarchy, or are at the top level of the hierarchy. Use the id or code of the Parent Project to traverse the levels. Use the Activities endpoints to retrieve the process's activity details. Master processes will not have a parent project id or a process name.

If a process is linked to a record such as an Audit, the linkedRecords element contains a list of the linked records, containing the id and business key of the record (audit code, supplier code, or site code), its type, name and status, and the date it was created.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Process to return with id:<###>	Invalid {id} - not found

Retrieve Record by Business Key

Description

Retrieves a single Process record's details using its business key (code). Use this function to retrieve the full details of an individual process using its key code.

Endpoint address: /services/rest/project/byKey/{code} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {code} parameter that determines the record to retrieve.

Example URL

.../services/rest/project/byKey/943

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a projectFullIDTO root element containing the individual attributes of the requested Process record.

File Structure

The main elements of the projectFullIDTO root element:

- projectTemplateSnapshot and Template - the settings from the associated template.
- teams - the users and their roles associated to the process.
- briefs - the associated process briefs; a separate element for each.
- linkedRecords - the associated linked records; a separate element for each.
- udfData - the contents of any user-defined custom fields associated to the Process record.
- attachments - a child element containing the ids of any file attachments the process has.

Processes and their activities can be nested within a hierarchy. In the absence of a function to extract the full hierarchy with a single GET call, to retrieve the hierarchy of a process a number of successive calls is required. The Parent Project field is a Boolean value which will be blank for processes that have no hierarchy, or are at the top level of the hierarchy. Use the id or code

of the Parent Project to traverse the levels. Use the Activities endpoints to retrieve the process's activity details. Master processes will not have a parent project id or a process name.

If a process is linked to a record such as an Audit, the linkedRecords element contains a list of the linked records, containing the id and business key of the record (audit code, supplier code, or site code), its type, name and status, and the date it was created.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
code	HTTP 404 Not found	Invalid {code} - blank or not found

Check Record Modification Timestamp

Description

Retrieves the last modification date and time of a Process record. Use this function to determine when the process was last updated.

Endpoint address: /services/rest/project/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/project/20

Response Details

If successful, an HTTP 200 response is sent containing the Last-Modified header showing the date and time of the last update of the requested Processord. For example:

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Messages

Element	Message	Meaning
id	HTTP 417 Expectation Failed	Invalid {id} - not found
id	HTTP 404 Not found	Invalid {id} - not numeric

Update Record

Description

Updates an existing Process record. Use this function to update a Process in Supplier Evaluation based on data sourced from the external system.

Process records can be updated, such as to set start/end dates, process brief details, or other attributes, from an external system. Updates do not trigger any automatic re-scheduling of a Process. Dates and other values are simply updated.

The PATCH method allows for partial updates to a record, to update values without the need to submit the full record. This endpoint supports either an XML or JSON payload. The creation of new records is not supported.

Endpoint address: /services/rest/project/{id} **HTTP method:** PATCH

Request Details

The body of the request contains a projectFullIDTO to specify the updates to the Process record, which is based on the projectFullIDTO type returned when retrieving a Process.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process* for details of their mapping to the fields within the UI. The data dictionary indicates which fields are capable of being updated.

The Process unique id is included in the request URI to identify the Process to be updated. Each transaction will represent a single Process. The payload need only include the mandatory values and the values of elements to be updated. A Process at any status can be updated.

Updating the Process record does not trigger the Process to be re-scheduled, even if the start or end dates are updated. Updates to the Supplier, Site, and Process Manager are not supported.

Example JSON Requests

JSON requests consist of the following elements:

`op` specifies the operation to be performed on a field:

- `replace` a value
- `add` a value to a list
- `remove` a value from a list
- `move` a value within a list

`path` specifies the location of the field to be updated.

`value` specifies the value to be applied by the update.

Note that JSON indexes are zero based.

1. Replace an existing value:

```
{
  "op" : "replace",
  "path" : "/udfData/rtfField001",
```

```
      "value" : "new value"
    },
```

2. Remove an item, in this case at the first index:

```
{
  "op" : "remove",
  "path" : "/sites/0"
},
```

3. Add a new item, in this case at the first index:

```
{
  "op" : "add",
  "path" : "/sites/0",
  "value": {
    "code": "A0006-00001",
    "supplierCode": "A0006"
  }
},
```

4. Replacing a value in multiple nested collections, referencing the index of each localeDataMap is keyed by locale, in this case referencing the default locale, but 'en_GB' or 'fr' for example would also be valid:

```
{
  "op" : "replace",
  "path" : "/briefs/0/sections/0/briefQuestions/1/availableAnswer/1/
localeDataMap/null",
  "value" : {
    "description": "new value"
  }
},
```

Example XML Requests

XML requests consist of the following elements:

- The XML element specifies the operation to be performed on a field:
- `replace` a value
- `add` a value to a list
- `remove` a value from a list
- There is no `move` operation for XML, `remove`, and `add` are used instead.
- `sel` specifies the location of the field to be updated.

Note that XML indexes are 1 based.

1. Replace an existing value:

```
<replace sel="ns4:projectFullDTO/ns4:udfData/ns1:rtfField001/text()
[2]">new value</replace>
```

2. Remove an item from a collection, in this case the first item:

```
<remove sel="ns4:projectFullDTO/ns4:sites[1]" />
```

3. Add two items, in this case two sites:

```
<add sel="ns4:projectFullDTO">
  <ns4:sites>
    <ns2:code>A0006-00001</ns2:code>
    <ns2:supplierCode>A0006</ns2:supplierCode>
  </ns4:sites>
  <ns4:sites>
    <ns2:code>A0006-00002</ns2:code>
    <ns2:supplierCode>A0006</ns2:supplierCode>
  </ns4:sites>
</add>
```

4. Replacing an existing value with a code to a glossary entry:

```
<replace sel="/ns4:activityFullDTO/ns4:activitySubStatus">
  <ns4:activitySubStatus>
    <ns6:code>ARTWORK</ns6:code>
  </ns4:activitySubStatus>
</replace>
```

This example is updating a code value which is a key to glossary record, in this case the Activity Sub Status. For this *aggregation type* relationship, the code must be specified as shown above. Specifying the code in the following way will result in an incorrect update:

```
<replace sel="/ns4:activityFullDTO/ns4:activitySubStatus/ns6:code">
  <ns6:code>ARTWORK</ns6:code>
</replace>
```

Response Details

The Process record is updated. All updatable values will be replaced with the values in the request.

If successful, an HTTP 200 response is sent with a body containing a ProjectRecordLink element. The ProjectRecordLink element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Process record's internal ID
recordLink	string	URI to the Process record
projectId	string	Code ID business key to the Process record
projectName	string	Process record's description

Error Responses

If the supplied data does not result in a valid Process, an HTTP 417 response is sent with an ErrorMessage/Message message stating the validation errors. The request should not be reattempted with the same content.

If the validation fails, no updates are made for the request.

Error Messages

Element	Message	Meaning
id	Process Record not found in the portal, based on Process ID	Cannot locate the Process record
projectType	Process Type with code {0} is not found in the portal	Invalid process type

ActivityRestService

This section describes the API for managing process activities. The following functions are available:

- [List of Values](#): retrieves a list of activities
- [Retrieve Record by ID](#): retrieves an Activity record using its unique identifier
- [Retrieve Record by Business Key](#): retrieves an Activity record using its business key
- [Check Record Modification Timestamp](#): retrieves the timestamp when an Activity record was last updated
- [Update Record](#): updates an existing Activity record
- [Update Status](#): updates the status of an existing Activity record

List of Values

Description

Retrieves a list of activities in a paged list. Use this function to locate Activity record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/activity **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
offset	Optional	int	No	No	60
pageSize	Optional	int	No	No	30
recordId	Optional	string	Yes	Yes	1

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Wildcard Search Supported?	Example
projectId	Optional	string	Yes	Yes	PR0000007
projectName	Optional	string	Yes	Yes	New Potato Product Launch
activityName	Optional	string	Yes	Yes	Artwork
activityCode	Optional	string	Yes	Yes	762
activityStatus	Optional	string	Yes	Yes	STARTED~COMPL ETED
activitySubStatus	Optional	string	Yes	Yes	MESSAGE SENT
activityType	Optional	string	Yes	Yes	PRODUCT_ARTW ORK
templateCode	Optional	string	Yes	Yes	ARTWORK
linkedRecord	Optional	string	Yes	No	Note: Options are: ASSESSMENT SUPPLIER SITE AUDIT PRODUCT_RECO RD PRODUCT_SPECI FICATION
personResponsible Name	Optional	string	Yes	Yes	John Smith
proposedStartDate	Optional	string	No	No	2015-05-19
proposedEndDate	Optional	string	No	No	2015-05-19
actualStartDate	Optional	string	No	No	2015-05-19
actualEndDate	Optional	string	No	No	2015-05-19
sequenceNumber	Optional	string	Yes	Yes	1.0
modifiedSince	Optional	string	No	No	2015-05-19 13:30:39
modifiedUntil	Optional	string	No	No	2015-05-19 13:30:39

Example URLs

.../services/rest/user/?offset=2&pageSize=20 .../services/rest/user/?
activityStatus=STARTED~COMPLETED

Response Details

For a successful response, XML is returned with a ProjectActivityLinkList root element containing an entries element for each matched process. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Process Activity record's internal ID

Element	Type	Description
recordLink	string	URI to the ActivityRestService Retrieve record service for this activity
activityCode	string	Process Activity's business key
projectRecordId	string	The parent Process's internal ID
projectId	string	The parent Process's business key

The returned XML also contains a `totalRecords` element, which states the total number of retrievable records that match the filter parameters.

Error Messages

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record by ID

Description

Retrieves a single Activity record's details using the record's internal unique ID (which is not visible in the UI). Use this function to retrieve the full details of an individual process activity.

Endpoint address: `/services/rest/activity/{id}` **HTTP method:** GET

Request Details

The URL contains the `{id}` parameter that determines the record to retrieve.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a <code>localeData</code> element returned for the specified locales if data is present.

Example URL

`.../services/rest/activity/5`

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a `projectActivityFullDTO` root element containing the individual attributes of the requested Activity record.

File Structure

The main elements of the `projectActivityFullDTO` root element:

- activityTemplateSnapshot and Template - the settings from the associated template.
- briefs - the associated process briefs; a separate element for each.
- linkedRecords - the associated linked records; a separate element for each.
- udfData - the contents of any user-defined custom fields associated to the Activity record.
- attachments - a child element containing the ids of any file attachments the activity has.

If an activity is linked to a record such as an Audit, the linkedRecords element contains a list of the linked records, containing the id and business key of the record (audit code, supplier code, or site code), its type, name and status, and the date it was created.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Activity to return with id:<###>	Invalid {id} - not found

Retrieve Record by Business Key

Description

Retrieves a single Activity record's details using its business key (code) and its parent process's code. Use this function to retrieve the full details of an individual activity using its key code.

Endpoint address: /services/rest/activity/byKey/{activityCode} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {activityCode} parameter that determines the record to retrieve.

Example URL

.../services/rest/activity/byKey/762

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a projectActivityFullDTO root element containing the individual attributes of the requested Activity record.

File Structure

The main elements of the projectActivityFullDTO root element:

- activityTemplateSnapshot and Template - the settings from the associated template.
- briefs - the associated process briefs; a separate element for each.
- linkedRecords - the associated linked records; a separate element for each.
- udfData - the contents of any user-defined custom fields associated to the Activity record.

- attachments - a child element containing the ids of any file attachments the activity has.

If an activity is linked to a record such as an Audit, the linkedRecords element contains a list of the linked records, containing the id and business key of the record (audit code, supplier code, or site code), its type, name and status, and the date it was created.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
activityCode	HTTP 404 Not found	Invalid {activityCode} - blank or not found

Check Record Modification Timestamp

Description

Retrieves the last modification date and time of an Activity record. Use this function to determine when the process activity was last updated.

Endpoint address: /services/rest/activity/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/activity/5

Response Details

If successful, an HTTP 200 response is sent containing the Last-Modified header showing the date and time of the last update of the requested Activity record. For example:

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Messages

Element	Message	Meaning
id	HTTP 417 Expectation Failed	Invalid {id} - not found
id	HTTP 404 Not found	Invalid {id} - not numeric

Update Record

Description

Updates an existing Process Activity record. Use this function to update an Activity in Supplier Evaluation based on data sourced from the external system.

Activity records can be updated, such as to set start/end dates, process brief details, or other attributes, from an external system. Updates do not trigger any automatic re-scheduling of a Process. Dates and other values are simply updated.

The PATCH method allows for partial updates to a record, to update values without the need to submit the full record. This endpoint supports either an XML or JSON payload. The creation of new records is not supported.

Endpoint address: /services/rest/activity/{id} **HTTP method:** PATCH

Request Details

The body of the request contains an activityFullIDTO to specify the updates to the Activity record, which is based on the activityFullIDTO type returned when retrieving an Activity.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process* for details of their mapping to the fields within the UI. The data dictionary indicates which fields are capable of being updated.

The Activity unique id is included in the request URI to identify the Activity to be updated. Each transaction will represent a single Activity. The payload need only include the mandatory values and the values of elements to be updated. Can update any updatable value in the Activity record, with the exception of the Activity Status field which can be updated using the Update Status endpoint.

Activities at Not Started, Started, or Approval Required status can be updated; Completed Activities cannot be updated.

Duration (Days) is an integer. Completed By can be set to a valid user or to *Auto Complete*. Brief question answers can be populated by identifying the corresponding question.

Updating the Activity record does not trigger the Process to be re-scheduled, even if the activity durations are updated. Updates to the Supplier, Site, and Process Manager are not supported. This means that the overall pattern of activity dates within a Process could appear differently to those generated by the system scheduling tool.



Note:

See the [Update Record](#) function of the [ProjectRestService](#) for examples of XML and JSON requests.

Response Details

The Activity record is updated. All updatable values will be replaced with the values in the request (excluding Activity Status).

If successful, an HTTP 200 response is sent with a body containing an ActivityRecordLink element. The ActivityRecordLink element consists of the returned elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Activity record's internal ID
recordLink	string	URI to the Activity record

Element	Type	Description
activityCode	string	Code ID business key to the Activity record
projectId	string	Activity record's description

Error Responses

If the supplied data does not result in a valid Process, an HTTP 417 response is sent with an ErrorMessage/Message message stating the validation errors. The request should not be reattempted with the same content.

If the validation fails, no updates are made for the request.

Error Messages

Element	Message	Meaning
code	Activity not found in the portal, based on Activity Code	Cannot locate the Activity record
status	Activity is at Completed status	Cannot update if already completed
subStatus	Sub Status with code {0} is not found in the portal	Invalid sub status
completedBy	Completed By User with code {0} is not found in the portal	Invalid user

Update Status

Description

Updates the status of an existing Process Activity record. Use this function to update an Activity's status from the external system, triggering any further updates to the associated Process or to other Activities within the Process.

Two alternative endpoints are available, to locate the Activity either by its record id or by its Code business key. They both return a record link to the Activity. These endpoints support a JSON payload.

Endpoint address: /services/rest/activity/{id}/status and /services/rest/activity/byKey{activityCode}/status } **HTTP method:** PUT

Request Details

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process* for details of their mapping to the fields within the UI. The data dictionary indicates which fields are capable of being updated.

The Activity unique id is included in the request URI to identify the Activity to be updated. Each transaction will represent a single Activity. The payload will only include the Activity Status to be updated.

If completing an Activity, the Completed Date and Completed By fields can be updated using the Update Record endpoint.

Activities at Not Started, Started, or Approval Required status can be updated; Completed Activities cannot be updated.

Update of the status will trigger the processing to apply updates to the associated Process or to other Activities within the associate Process, including:

- Update of keys and gates when the Activity is Completed.
- Auto-completion of Gates.
- If the last Activity in the Process is set to Completed, the Process Status is set to Completed.

Response Details

The Activity Status is updated. All updatable values will be replaced with the values in the request (excluding Activity Status).

If successful, an HTTP 200 response is sent with a body containing an ActivityRecordLink element. The ActivityRecordLink element consists of the returned elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Activity record's internal ID
recordLink	string	URI to the Activity record
activityCode	string	Code ID business key to the Activity record
projectId	string	Activity record's description

Error Responses

If the supplied data does not result in a valid Activity, an HTTP 417 response is sent with an ErrorMessage/Message message stating the validation errors. The request should not be reattempted with the same content.

Error Messages

Element	Message	Meaning
code	Activity not found in the portal, based on Activity Code	Cannot locate the Activity record
status	Activity is at Completed status	Cannot update if already completed
status	Status with code {0} is not found in the portal	Invalid status

ProductRecordRestService

This section describes the API for managing Product records. The following functions are available:

- [List of Values](#): retrieves a list of Product records
- [Retrieve Record by ID](#): retrieves a Product record using its unique identifier
- [Retrieve Record by Business Key](#): retrieves a Product record using its business key
- [Check Record Modification Timestamp](#): retrieves the timestamp when a Product record was last updated

- **Create Record:** creates a new Product record
- **Update Record:** updates an existing Product record

**Note:**

When integrated with Merchandising, using the ProductRecordRest API to make changes that affect the Product Codes table can have an impact on the integration.

List of Values

Description

Retrieves a list of Product records in a paged list. Use this function to retrieve a simple list of product titles and IDs, or to locate Product record IDs prior to a retrieve or update operation. Parameters are available to apply specific selection criteria for filtering the returned records.

Endpoint address: /services/rest/productRecord **HTTP method:** GET

Request Details

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/ Optional	Value Type	Multiple Value Separator (~) Supported?	Example
offset	Optional	int	No	60
pageSize	Optional	int	No	30
modifiedSince	Optional	string	No	2015-05-19 13:30:39
modifiedUntil	Optional	string	No	2015-05-19 13:30:39
productRecordCode	Optional	integer	No	123456
altProductNumber	Optional	string	No	ABC001
division	Optional	string	Yes	DIV_A~DIV_B Must be the code of an entry in the Divisions glossary

Example URLs

.../services/rest/productRecord/?offset=2&pageSize=20.../services/rest/productRecord/?
productRecordCode=123456

Response Details

For a successful response, XML is returned with a ProductRecordLinkList root element containing an entries element for each matched Product Record. The entries element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Product record's internal ID
recordLink	string	URI to the ProductRecordRestService Retrieve record service for this Product record
code	string	Code business key to the Product record
title	string	Product's title

The returned XML also contains a `totalRecords` element, which states the total number of retrievable records that match the filter parameters.

Error Messages

Element	Message	Meaning
modifiedSince	Invalid dates cause stack trace to come back to "response"	
modifiedUntil	Invalid dates cause stack trace to come back to "response"	
pageSize	Error Code: INVALIDPAGESIZE	Page Size must be between 1 and 100
offset	Error Code: INVALIDOFFSET	Offset must be a positive integer

Retrieve Record by ID

Description

Retrieves a single Product record's details using the record's internal unique ID (which is not visible in the UI). Use this function to retrieve the full details of an individual Product Record.

Endpoint address: `/services/rest/productRecord/{id}` **HTTP method:** GET

Request Details

The URL contains the `{id}` parameter that determines the record to retrieve.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 5 - Product* for details of their mapping to the fields within the Supplier Evaluation UI.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Multiple Value Separator (~) Supported?	Example
locale	Optional	string	Yes	en_GB~fr Locale country code. If used, fields that have language translations have a <code>localeData</code> element returned for the specified locales if data is present.

Example URL

.../services/rest/productRecord/99

Response Details

For a successful response, an HTTP 301 message is returned, which is automatically acted upon to retrieve XML consisting of a productRecordFullDTO root element containing the individual attributes of the requested Product record. If an ID is not specified, a list of all Product records is returned (per the List of Values function).

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 5 - Product* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 IllegalStateException: Cannot find the Product Record to return with id:<###>	Invalid {id} - not found

Retrieve Record by Business Key

Description

Retrieves a single Product record's details using its business key (code). Use this function to retrieve the full details of an individual Product record using its Supplier Evaluation product code.

The Product record's business key is the product code that is visible in the UI. This an individual number assigned by Supplier Evaluation. It is not the product's traded unit identifier such as a SKU, that is, the product number, which is also visible in the UI.

Endpoint address: /services/rest/productRecord/byKey/{code} **HTTP method:** GET

Request Details

There are no request parameters, but the URL contains the {code} parameter that determines the record to retrieve.

Example URL

.../services/rest/productRecord/byKey/31

Response Details

If successful, an HTTP 301 redirect response is sent containing a header Location with the URI of the document requested.

The returned URI represents the call to be made to the Retrieve Record by ID function (with the id substituted with the relevant value), which will return a productRecordFullDTO root element containing the individual attributes of the requested Product record.

**Note:**

If your method of calling the API is configured to automatically follow HTTP 301 redirects, the call may automatically return a productRecordFullDTO root element containing the individual attributes of the requested Product record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 5 - Product* for details of their mapping to the fields within the Supplier Evaluation UI.

Error Messages

Element	Message	Meaning
code	HTTP 404 Not Found	Invalid {code} - blank
code	HTTP 404 Not Found	Invalid {code} - not found
code	NumberFormatException: For input string: <###>	Invalid {code} - not numeric

Check Record Modification Timestamp

Description

Retrieves the last modification date and time of a Product record. Use this function to determine when a Product record's details were last updated.

Endpoint address: /services/rest/productRecord/{id} **HTTP method:** HEAD

Request Details

There are no request parameters, but the URL contains the {id} parameter that determines the record to retrieve.

Example URL

.../services/rest/productRecord/99

Response Details

If successful, an HTTP 200 response is sent containing the Last-Modified header to showing the date and time of the last update of the requested Product record.

```
HTTP/1.1 200 OK
Date: Wed, 13 Jul 2016 07:52:14 GMT
Last-Modified: Fri, 08 Jul 2016 06:44:46 GMT
Content-Type: application/xml
Content-Length: 0
```

Error Messages

Element	Message	Meaning
id	HTTP 404 Not Found	Invalid {id} - not numeric
id	HTTP 417 Expectation Failed	Invalid {id} - not found

Create Record

Description

Creates a new Product record. Use this function to create new Product records in Supplier Evaluation based on data sourced from the external system.

Dependencies: If linked to a supplier or site, the Supplier/Site must be present in the application and its record ID obtained. If assigning a Business Category, Technologist, or Other Contact, the record must be present in the application and its record ID obtained.

Technologist and Other Contacts may be omitted to default to TBC if the TBC user is present in the application. For the status to be Active, the Product record must be linked to a supplier. For more information, see [Dependencies](#).



Note:

The Use Alternative Code for Product API system parameter enables a facility for holding alternative supplier codes against Supplier records. This feature is of use if the external system does not hold the Supplier Evaluation supplier codes. If enabled, and the supplier code submitted by the API is not found in Supplier Evaluation, a secondary lookup will be performed to attempt to locate the Supplier using the alternative codes facility.

Endpoint address: /services/rest/productRecord **HTTP method:** POST

Request Details

The body of the request contains a ProductRecordFullIDTO to specify the Product record to create. Compared to retrieving a Product record (which uses the same ProductRecordFullIDTO type), this request is much shorter. Only the attributes that are to be populated on the created Product record need to be included. As a minimum, this must include the fields shown in the following table.

Product Record Mandatory Fields

Field Name	Element Name
Product Title	title
Status	productRecordStatus
Variant Name	variantName
Quantity	quantity
Technologist	technologist
Other Contacts	productRecordOtherUser
Supplier Details	supplier
Site Details	site
Deleted Record	deleted (Set to false)
Language	label18NKey

**Note:**

To ensure a valid unique business key code is automatically assigned to the record being created, a value of zero must be passed in the <code> element of the XML when submitting the POST request.

Example Request XML

Where the record is linked to another record, such as the Technologist in this case, the business key must be provided (not the description) in order to form the link between the records. In general, the business key will be the code attribute of the linked record.

**Note:**

The API does not apply status-based validation as is applied using the Supplier Evaluation UI when manually progressing a Product record.

Single Variant Mode

```
<productCovered>
  <productTitle/>
  <variantName/>
  <quantity/>
  <retailerProductNumber/>
</productCovered>
```

Multi Variant Mode

```
<productCovered>
  <productTitle/>
  <variantName/>
  <quantity/>
  <retailerProductNumber/>
</productCovered>
<productCovered>
  <productTitle/>
  <variantName/>
  <quantity/>
  <retailerProductNumber/>
</productCovered>
```

Where the record is linked to another record, such as the Technologist in this case, the business key must be provided (not the description) in order to form the link between the records. In general, the business key will be the code attribute of the linked record.

**Note:**

The API does not apply status-based validation as is applied using the Supplier Evaluation UI when manually progressing a Product record.

It is therefore possible, using the API, to create a Product record at Active status without the supplier or site being populated.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 5 - Product* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a ProductRecordLink root element. The root element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Newly created Product record's internal ID
recordLink	string	URI to the newly created Product record, for use in a GET request
code	string	Code business key to the newly created Product record
title	string	Product's title

Error Messages

If the supplied data does not result in a valid Product record (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

Element	Message	Meaning
productCoverage	ERROR: class <<hostname>>.ProductRecord.productCovered - The condition is invalid	No tag provided
code	ERROR: class <<hostname>>.ProductRecord.code - A Product Record's code is mandatory	No tag provided
code	ERROR: class <<hostname>>.ProductRecord.code - A Product Record's code is mandatory	No value provided
code	ERROR: class <<hostname>>.ProductRecord.code - A Product Record's code must be unique:<###>	Code already exists
code	HTTP Status 400 - Bad request	Code is not numeric
title	DtoValidationException: title is mandatory	No tag provided No value provided
productRecordStatus	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>.ProductRecordStatus using status:<###>	No tag provided No value provided Status not found
variantName	ERROR: class <<hostname>>.ProductRecord.productCovered[###].variantName - Variant name is mandatory	No tag provided No value provided
quantity	ERROR: class <<hostname>>.ProductRecord.productCovered[###].quantity - Quantity name is mandatory	No tag provided No value provided

Element	Message	Meaning
technologist	MySQLIntegrityConstraintViolationException: Cannot add or update a child row: a foreign key constraint fails ('<###>'.t_prdrctd,CONSTRAINT '<###>' FOREIGN KEY ('fk_technologist') REFERENCES 't_user' ('c_id'))	No tag provided No value provided
technologist	ObjectNotFoundException: No row with the given identifier exists: [com.micros.creations.core.domain.User#<###>]	Technologist not found
productRecordOtherUser/ user	DtoValidationException: ProductRecordOtherUser has no valid user for role:<###> ProductRecordOtherUser Roles are mandatory for the following missing Roles: [<###>]	No tag provided User not found
productRecordOtherUser/ user	ObjectNotFoundException: No row with the given identifier exists: [<<hostname>>.User#<###>]	No value provided
productRecordOtherUser/ role	DtoValidationException: ProductRecordOtherUser has no valid user for role:<###> ProductRecordOtherUser Roles are mandatory for the following missing Roles: [<###>]	No tag provided User not found
productRecordOtherUser/ role	ObjectNotFoundException: No row with the given identifier exists: [<<hostname>>.Role#<###>]	No value provided
supplier	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>.Company using code:<###>	No tag provided
supplier	MySQLIntegrityConstraintViolationException: Cannot add or update a child row: a foreign key constraint fails ('<###>'.t_prdrctd,CONSTRAINT '<###>' FOREIGN KEY ('fk_supplier') REFERENCES 't_company' ('c_id'))	No value provided Supplier not found
site	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>.Site using id:<###>, code:<###>, supplierCode:<###> and supplierId:<###>	No tag provided No value provided Supplier not found
countryWhereSold	HTTP 417 IllegalStateException: Cannot locate keyword for class <<hostname>>.CountryWhereSold using code:<###> and spec type:<###>	No value provided
countryWhereSold	ObjectNotFoundException: No row with the given identifier exists: [<<hostname>>.CountryWhereSold#<###>]	Country not found

Element	Message	Meaning
	ERROR: class <<hostname>>.ProductRecord.barcode[<###> - Bar Code must be either 8 or 13 characters long	Invalid code length
	ERROR: class<<hostname>>.ProductRecord.shippingCaseCode[<###> - Shipping Case Code must be either 14 digits long	Invalid code length

Update Record

Description

Updates an existing Product record. Use this function to update a Product record's details in Supplier Evaluation based on data sourced from the external system.

Endpoint address: /services/rest/productRecord/{id} **HTTP method:** PUT

Request Details

The body of the request contains a ProductRecordFullDTO to specify the updates to the Product record. Compared to retrieving a Product record (which uses the same ProductRecordFullDTO type), this request is much shorter. As a minimum, the values specified as mandatory for the Create Record function (see above) must be included.

The request content is the same as that for creating a Product Record. After the call, the Product Record is updated to match the request.



Note:

When updating records, all values must be included. If a value or element is omitted from the request, the field contents will be cleared on the Product record.

See the associated WADL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 5 - Product* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing a ProductRecordLink element. The ProductRecordLink element consists of the elements shown in the following table.

Returned Elements

Element	Type	Description
recordId	long	Product record's internal ID
recordLink	string	URI to the Product record, for use in a GET request
code	string	Code business key to the Product record
title	string	Product's title

Error Responses

If the supplied data does not result in a valid Product record (such as a missing mandatory field), an HTTP 417 response is sent with an ErrorMessage/Message XML body message stating the validation errors. The request should not be reattempted with the same content.

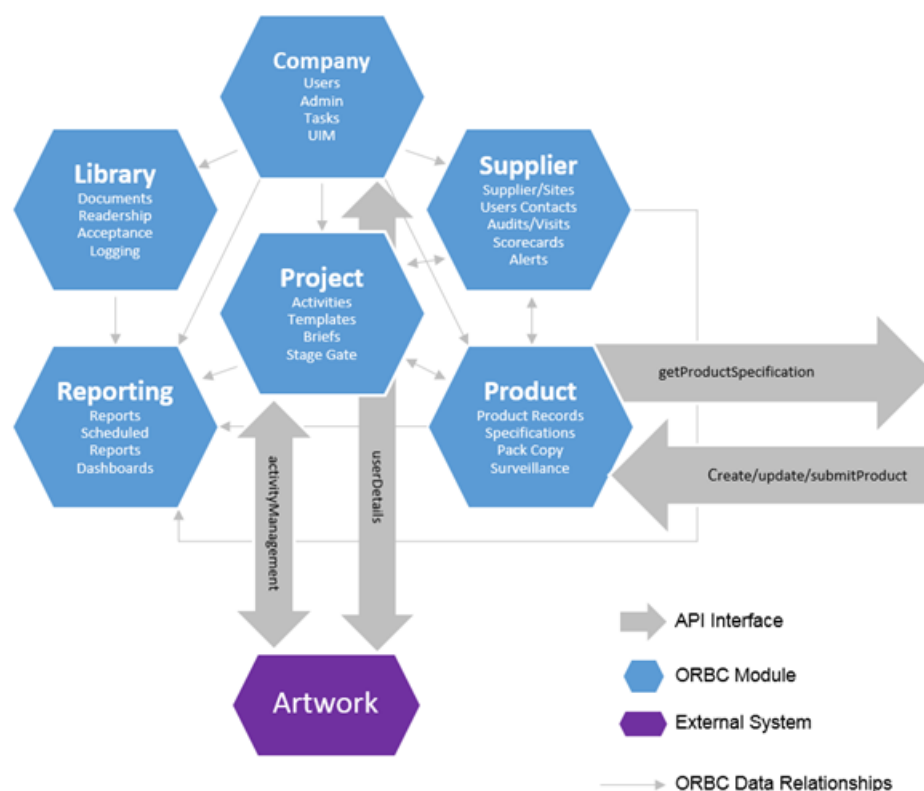
5

SOAP APIs

The following SOAP APIs are available:

- [Project Activities](#): Project Activities

Figure 5-1 Overview of SOAP APIs



Parameters and Filtering

Various parameters can be included in calls to the APIs, generally to define what data is to be returned, but can also control how records are returned. The following table lists some commonly used parameters.

Common Parameters

Parameter	Type	Description
offset	int	Used with pageSize to control the paging of a returned list of records. Specifies the starting point for the retrieval of records. If not specified, zero is assumed. For example, to retrieve 150 records: 1. A call with offset = 0 and pageSize = 50 returns the first 50 records. 2. Then, a call with offset = 50 and pageSize = 50 returns the next 50. 3. Then, a call with offset = 100 and pageSize = 50 returns the final 50.
batchSize	int	Used with offset to control the paging of a returned list of records. Specifies the number of entries in each page of returned list of records. If not specified, 30 is assumed. The maximum is 100.



Note:

Date/Time parameters must be provided in the YYYY-MM-DD hh:mm:ss format.

When retrieving records, the returned XML only includes elements that actually contain data; empty elements are omitted.

The following example provides the first 50 Product Specifications that changed status to either Active or Off Range in May 2016. Searching with either fromDate or toDate specified without specStatus, will filter based on the last amended date of the Product Specification.

```
<soapenv:Envelope xmlns:soapenv="<<hostname>>/soap/envelope/" xmlns:v1="<<servicename>>/">
<soapenv:Header/>
<soapenv:Body>
  <v1:getProductSpecification>
    <request>
      <batchsize>50</batchsize>
      <offset>0</offset>
      <specStatus>ACTIVE</specStatus>
      <specStatus>OFF_RANGE</specStatus>
      <fromDate>2016-05-01T00:00:00</fromDate>
      <toDate>2016-05-31T24:00:00</toDate>
    </request>
  </v1:getProductSpecification>
</soapenv:Body>
```

Project Activities

This section describes the API for Artwork integration. The following function is available:

- **Activity Management:** updates existing Artwork activities

Activity Management

Description

Updates the sub status of existing Artwork Project Activities. This function is used for the integration of Supplier Evaluation with an external Artwork Management system, to update the status of an artwork activity from the external system.



Note:

This service has been superseded by the Artwork Restful Service.

Endpoint address: /services/activityManagement **HTTP method:** updateActivityStatus

Request Details

The body of the request contains an updateActivityStatus/activityUpdateRequest element consisting of the elements shown in the following table.

Request Elements

Element	Type	Description
activityName	string	Project activity's name
activityRecordId	string	Project activity record's internal ID
projectId	string	ID business key to the Project record
projectRecordId	string	Project record's internal ID
subStatusCode	string	Code business key to the activity's sub status

After the call, the Project Activity record's sub status is updated to match the request.

See the associated WSDL for a full list of the attributes, and the *Oracle Retail Supplier Evaluation Cloud Service Data Dictionary, Volume 3 - Process* for details of their mapping to the fields within the Supplier Evaluation UI.

Response Details

If successful, an HTTP 200 response is sent with a body containing an updateActivityStatusResponse root element, consisting of a return element populated with SUCCESS.

Error Messages

If an error occurs, the updateActivityStatusResponse root element's return element will contain FAILURE.

A

Appendix: Secure Development Guide

The web service APIs provided by Supplier Evaluation are designed to be secure by default. When consuming these services, there are recommended best practices for the clients which call them. These development and usage guidelines are provided to maintain secure use, and preserve integrity throughout the system domain.

The Supplier Evaluation web services are secured using OAuth 2.0. In order to use them, calling clients require an External System within Supplier Evaluation and a corresponding Application within IDCS with which to authenticate. Transport encryption is enabled using TLS v1.2 as a minimum standard.

Best Practices for Web Service Use

This section describes some best practices.

Store External System Credentials Securely

In order to grant access to web services, credentials must be set up within Supplier Evaluation in an External System record and a corresponding Application record in IDCS. These credentials, used by the client to call web services, should be managed securely in the client's domain. It is advised not to store passwords in plaintext, either in a file or a database table.

Configure External Systems Permissions Appropriately

The permissions of External Systems are configurable, and determine which services may be accessed. The permissions model governs not only the service which is accessible (for example, The Product Service), but also the endpoints (or actions) permissible on that service (for example, Read a Product or Add a Product).

It is recommended that External Systems are set up with the minimum authorities required to perform their required function. External System records should have granular permissions wherever feasible, rather than be granted access to many disparate functions. This minimizes the level of disclosure possible if credentials are compromised.

Maintain Strong Passwords for External System Records

Configurable parameters within the Supplier Evaluation Oracle Identity Cloud Service (IDCS) or Oracle Cloud Infrastructure Identity and Access Management (OCI IAM) tenant govern the strength of passwords used for External Systems and the rules surrounding their expiry. These parameters are set to values which strike a suitable balance between usability and security by default.

It is advised that should a client wish to change any of these parameter values, it should only be to increase security. Changes which slacken security are *not* recommended.

Those responsible for maintaining clients which consume services should note that passwords for External Systems expire in the same manner as those for regular application users.



Note:

A process for updating the credentials used by the client should be put in place, as IDCS or OCI IAM does not send automated notifications of pending password expiry.

For full details of this functionality, and of External Systems in general, see the *Oracle Retail Supplier Evaluation Cloud Service Administration Guide*.

Make Use of the Web Service Log

Intrusion detection mechanisms are employed within the hosted environment of Supplier Evaluation to capture to prevent malicious behavior at the network level. In addition to this, protections against XML injection attacks exist at the logical tier.

The application also provides a Web Service Log for users with sufficient permissions to see what calls have been made to its services. This provides a user-centric audit of the web service activity within the application, and can help clients monitor usage, or gather metrics to understand usage behavior further.

Ensure Your Client Supports TLS v1.2 or Higher

All Supplier Evaluation deployments require communication over HTTPS, and use TLS v1.2 as a minimum encryption protocol. Clients of the web service must be developed within a technology which supports TLS v 1.2 or higher. For example, Java 7 does not support TLS v1.2 out of the box.

Limit the Size of Web Service Payloads

The web services are designed for continued, real-time use. It is recommended that calls to the services are made in a manner to retrieve or send only what is required.

Where larger result sets are necessary however, provision is made within the web service APIs to allow paging of data. By providing an offset and a page size, it is possible to retrieve a page of results at a time.

B

Appendix: Using the Data Privacy API

The Supplier Evaluation Data Privacy web service API provides a means for the retailer/portal owner to execute data privacy requests. The requests are classified as Right to Access, where an individual requests details of all their personal data that is held in the system; and Right to be Forgotten, where the individual requests that their personal data be erased from the system.



Note:

The provision of the Data Privacy API is a security enhancement which aids the retailer/portal owner to carry out data privacy requests.

It is the responsibility of the retailer/portal owner to manage the fulfilment of data privacy requests. In order to do so, they will need to build an application to call the Data Privacy API, and handle the returned data. The API will be permanently available for access on demand.

The retailer/portal owner's responsibilities include verifying the identity of the individual making the request, presenting the information back to them in a secure manner, and deleting any local copies of the data; also the configuration of the portal's Terms and Conditions to include statements about the consent to store personal data, and any opt-out procedure.

This appendix provides additional detail to support the [DataPrivacyService](#) section of this document. It covers the following areas:

- [Definition of Personal Data](#): an explanation of some of the terminology used, and the type of data within Supplier Evaluation that is considered to be personal data.
- [Personal Data Flow](#): an overview of the flow of data within Supplier Evaluation which is considered to be personal data.
- [Right to Access Requests](#): further detail on this type of request, with an example of its execution.
- [Right to be Forgotten Requests](#): further detail on this type of request, with an example of its execution.
- [Other Aspects of Data Privacy](#): other aspects of Data Privacy and how they relate to Supplier Evaluation.

Definition of Personal Data

The following terms are used to describe the actors involved in the handling and processing of personal data:

- **Data Subject** - The person whom the personal data describes. In Supplier Evaluation, this refers to an employee of the retailer/portal owner, an employee of a supplier, or an employee of a third party such as a design agency or certification body.

- Data Controller - The retailer/portal owner. In a cloud environment, Oracle becomes the data custodian, but the retailer/portal owner retains ownership and control of the data.
- Data Processor - Applications that work with personal data, such as Supplier Evaluation, and any external applications that share its data.

The definition of personal data is any information that can be mapped to a unique user, whether the identifiable information is stored with the key or not.

The personal data held in Supplier Evaluation is business contact information, such as:

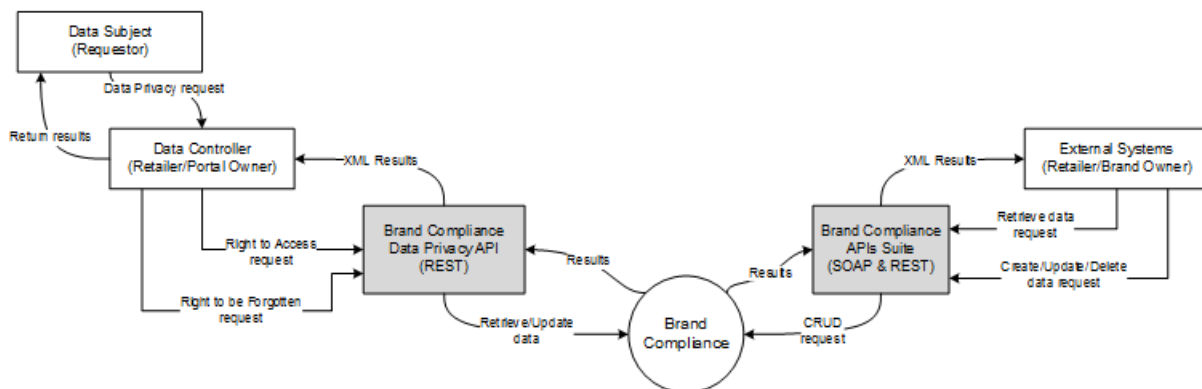
- Name of the data subject
- Contact addresses
- Email addresses
- Telephone and Fax numbers

A general assumption is that user-defined/custom fields and file attachments are not used to store personal data. This is applied as a policy constraint rather than specific controls within the system.

Personal Data Flow

The following diagram outlines the flow of personal data within Supplier Evaluation:

Figure B-1 Personal Data Flow



The Data Subject (requestor) makes the request to the Data Controller (retailer/portal owner) who submits the request to the Supplier Evaluation Data Privacy API. The API performs the search or erase Supplier Evaluation processing accordingly and returns the results for the retailer/portal owner to relay back to the requestor.

The full suite of Supplier Evaluation APIs allows the retailer/portal owner to make requests from external systems to retrieve, create, update, (and in some cases delete) Supplier Evaluation data. The data handled by these APIs may include personal data.

Access to Supplier Evaluation data through the APIs is secured by an extension of the system's configurable Permissions security model. Based on the role-based access control model which restricts users' access to data and functionality based on their role within the system, external systems are granted access to the API services. If necessary, the access can be restricted to individual endpoints of the services. Authentication is achieved using individual ids and passwords.

**Note:**

Supplier Evaluation does not *push* data to external systems; the retailer/portal owner *pulls* the data. It is therefore the responsibility of the retailer/portal owner to trace and flag any personal data that has been taken from Supplier Evaluation, as appropriate.

Right to Access Requests

The Right to Access service facilitates the request from the data subject (individual requestor) to the data controller (retailer/portal owner) to provide details of what personal data concerning them is held, and to provide a copy of the data in electronic format.

The majority of personal data is held within User/Person records (an account for each user of the system) and Contact records (supplier users may be designated specific contact roles). There are also instances of personal data that are not linked to a user account, such as the names and email addresses of general contacts, who are not users of the system. The retailer/portal owner must decide which of that data returned by a search relates to the requestor.

When a retailer/portal owner receives a request, they use the *Right to Access* application they have developed to call the Data Privacy API. The API searches Supplier Evaluation and returns the results for the retailer/brand owner to analyze, and edit or reformat if necessary, before relaying back to the requestor.

Personal Data Searched and Returned

The following table describes the areas of the system that hold personal data, and which fields are searched and returned.

Personal Data

Record/ Element Name	Details
User <user>	Each user of the system has a User record and a Person record which contain their Name, Email, Phone, Mobile, Fax, and Address details. Each account has a Login id, which is unique to the user. Names and email addresses are not necessarily unique. Search Field: Name Returned Fields: Record Type; Name; Email; Phone; Mobile; Fax; Login; Address; User Type (RETAILER, SUPPLIER or SITE); User record id; Person record id
Contact <contact>	A supplier user may be designated a contact for the supplier or its sites. Contact records share data with the User and Person records; they contain the Name, Email, Alternative Email, Phone, Mobile, Fax, and Address fields. A contact may be an individual who is not a user of the system. Search Field: Name (where contact is flagged as not a user of the system) Returned Fields: Record Type; Name; Email; Alternative Email; Phone; Mobile; Fax; Address; Contact record id; Supplier Name; Supplier Code

Record/ Element Name	Details
Company <retailer>	The single Company record holds the main retailer/portal owner contact details of a Contact Name and associated Email, Phone, Fax, and Address fields. The contact may or may not be a user of the system. Search Field: Contact Name Returned Fields: Record Type; Contact Name; Email; Phone; Fax; Address; Company record id
Certification Body <certificationBody>	A glossary of Certification Bodies is used to assign to Third Party Audits. Each record contains a Contact Name and supporting Email, Phone, Fax, and Address, which refers to an individual at a third-party organization. Search Field: Contact Name Returned Fields: Record Type; Contact Name; Email; Phone; Fax; Address; Certification Body record id
Supplier <supplier>	The Supplier record contains a Contact Name and associated Email, Phone, Fax, and Address fields, which refer to an individual who may or may not be a user of the system. Search Field: Contact Name Returned Fields: Record Type; Contact Name; Email; Phone; Fax; Address; Supplier record id; Supplier Name; Supplier Code The Supplier record also contains billing details entered using the registration wizard, including the name of an Individual or Department, a Billing Contact Name, and associated Email, Phone, Fax, and Address. They refer to an individual who may or may not be a user of the system. Search Field: Individual or Department; Billing Contact Name Returned Fields: Record Type; Individual or Department; Billing Contact Name; Email; Phone; Fax; Address; Supplier record id; Supplier Name; Supplier Code
Audit/Visit <audit>	The Audit/Visit record may have free text Name entries in the People Present table for attendees who are not users of the system. Entries for users also store the name in a <i>snapshot</i> text field. The Audit/Visit Issue record may have non-conformance actions with a free text Assigned To Name or a free text Completed By Name. Audit Checklists also share the Assigned To Name when a non-conformance is being created for a Checklist. Search Field: Name in People Present table; Issue Assigned To Name; Issue Completed By Name Returned Fields: Record Type; Name, Assigned To Name; Completed By Name; Audit/Visit record id; Audit/Visit Template Name; Audit/Visit Code; Supplier Name; Supplier Code; Site Name; Site Code; Audit/Visit Issue record id; Audit/Visit Issue Code

Security and Logging

The following Service and Endpoint configurations must be configured within the Supplier Evaluation Admin area, and assigned to the External Systems that are to have access to the Right to Access service:

Service:	DATAPRIV	Data Privacy Service
Endpoint:	DATAPRIV_ACCESS_POST	Creates a new DataPriv request and returns a link to the record's details
Endpoint:	DATAPRIV_ACCESS_GET	Retrieves DataPriv details for a given id

All calls to the service will be logged in the Supplier Evaluation Web Service Log.

The Data Privacy records created by the service are not viewable in Supplier Evaluation, and are not purged.

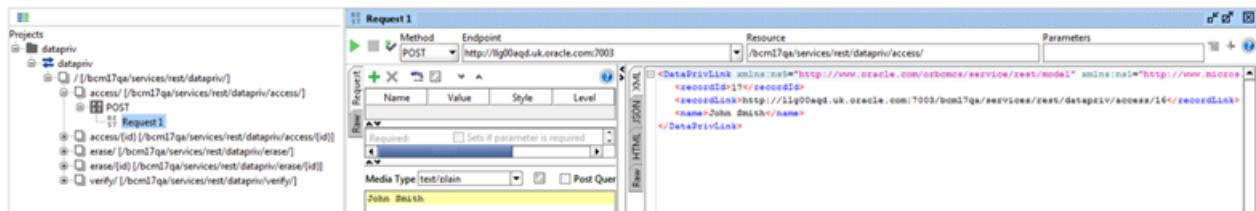
Example of Using the Right to Access Service

The request returns the actual data held, not the context in which it is used within the system. For example, to return the name and contact details of a Technologist, but not the fact that within the system what they are responsible for approving.

The stages involved in executing a Right to Access request are as follows. For this worked example, the call is made using a web service testing application.

1. Download the datatpriv WADL definition from the /services area of the Supplier Evaluation portal.
2. Submit a POST call to the services/rest/datatpriv/access service, passing the name of the individual in the payload.

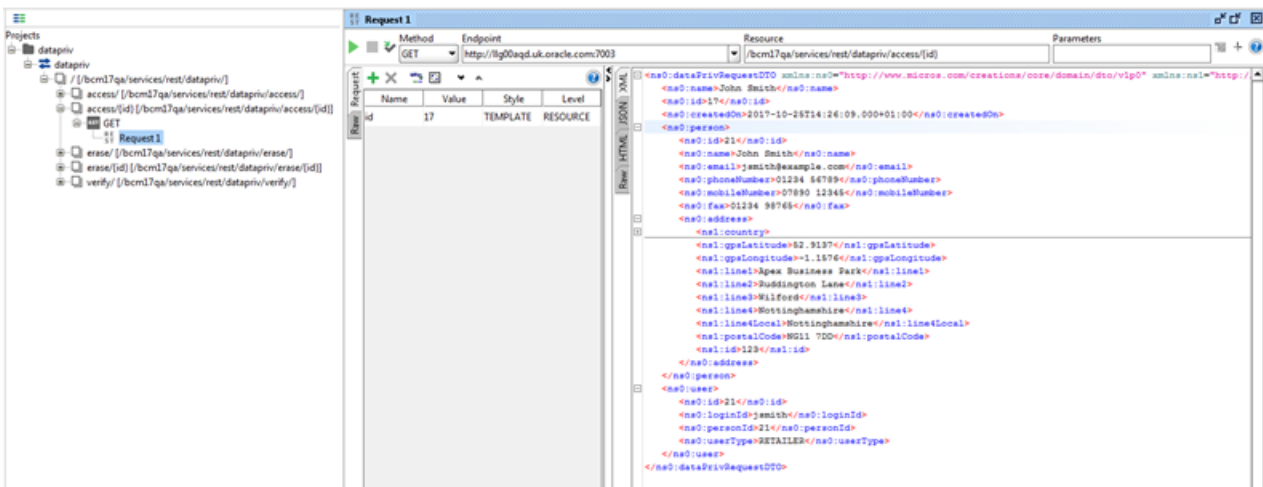
Figure B-2 Access Post Request



A recordLink element is returned with a link to a Data Privacy record.

3. Submit a GET call to the services/rest/datatpriv/access service, passing the returned recordId value as the id parameter.

Figure B-3 Access Get Request



A message is returned containing the personal data that matches the name, within a dataprivRequestDTO root element. The structure of the XML message is as follows:

Figure B-4 Access Returned Message



Returned Elements

Element Name	Details
name	The name of the individual as passed on the POST request.
id	The record id of the Data Privacy record created for this request.
createdOn	Timestamp of when the request was submitted.
person	Personal data fields from the Person records that match the name parameter.
user	Personal data fields from the User records that match the name parameter.
certificationBody	Personal data fields from the Certification Body records that match the name parameter.
retailer	Personal data fields from the Company/Retailer record that matches the name parameter.
supplier	Personal data fields from the Supplier record that match the name parameter.
site	Personal data fields from the Site record that match the name parameter.
contact	Personal data fields from the Contact records that match the name parameter.
audit	Personal data fields from the Audit/Visit and Issue records that match the name parameter.

The personal data fields are contained within tags that are similar to their (English) field labels in the UI:

```
<ns0:user>
  <ns0:id>7</ns0:id>
  <ns0:loginId>john smith</ns0:loginId>
  <ns0:name>John Smith</ns0:name>
  <ns0:email>jsmith-supplier@example.com</ns0:email>
  <ns0:userType>SUPPLIER</ns0:userType>
</ns0:user>
```

Additional metadata fields are included to identify the record by its code/id, name/description, and type.

Address elements may contain multiple localeData elements for country language translations:

```
<ns0:address>
  <ns1:country>
    <ns2:code>GB</ns2:code>
    <ns2:countryType>COUNTRY</ns2:countryType>
    <ns2:internetDomainSuffix>co.uk</ns2:internetDomainSuffix>
  <ns2:localeData>
    <ns1:description>Reino Unido</ns1:description>
    <ns1:locale>es</ns1:locale>
    <ns1:id>1904</ns1:id>
  </ns2:localeData>
```

There may be duplicates of the elements, where the system has more than one user with the same name, or where there are multiple records associated to an individual.

4. Edit the message using a text editor to remove duplicates, or entries that do not relate to the individual who made the request, and reformat if necessary, before delivering back to the requestor.

Right to be Forgotten Requests

The Right to be Forgotten service facilitates the request from the data subject (individual requestor) to the data controller (retailer/portal owner) to erase the personal data concerning them. The erasure of data is subject to certain predefined conditions within the system, and to the discretion of the data controller as to if/what data may be erased. The erasure is not reversible.

Data is erased by obfuscation, that is, replacing the value with a string of random text.

When a retailer/portal owner receives a request, they use the *Right to be Forgotten* application they have developed to call the Data Privacy API. The API validates the request and erases the requested data from Supplier Evaluation. The retailer/portal owner then relays the result back to the requestor.

Personal Data Updated

Generally, personal data is erased from User and Contact records, with the obfuscation of names and email addresses being reflected elsewhere in the system wherever they are referenced. Personal data that is not linked to a User or Contact record may also be erased; the retailer/portal owner must make a judgement on whether the data relates to the individual making the request.

The following table describes which areas of the system may have data erased.

Erasable Personal Data

Record	Fields
Users	Name; Email; Phone; Mobile; Fax; Login; Address
Contacts	Name; Email; Alternative Email; Phone; Mobile; Fax; Address
Company	Contact Name; Email; Phone; Fax
Design Agency	Contact Name; Email; Phone; Fax; Address
Counter Ticket Producer	Contact Name; Email; Phone; Fax; Address
Certification Body	Contact Name; Email; Phone; Fax; Address
Supplier	Contact Name; Email; Phone; Fax Individual or Department; Billing Contact Name; Email; Phone; Fax in the Billing Details
Audit/Visit	Name in the People Present table Assigned To Name; Completed By Name in the Issue record

Text fields are replaced with randomly-generated text; the numeric GPS address fields are blanked; email addresses are set to x@example.com where x is 15 characters of randomly-generated text; countries and region selections are set to blank. If a field is blank, it remains unchanged.

If a User record is updated, the account is also deactivated and the password is expired.



Note:

The obfuscation of data in Supplier Evaluation as a result of a Right to be Forgotten erasure request is not reversible.

Personal data may also be updated using the individual APIs, or manually edited within the system. Physical deletion is not possible, due to referential integrity.

Exceptions

The key purpose of Supplier Evaluation is to assure due diligence in the management of the retailer/portal owner's products, and the suppliers thereof. There are legal or regulatory obligations to retain the approved technical specifications for a set number of years, and during that time they may be required as legal evidence as part of a product safety issue. Where the approval or authorization by an individual forms part of the due diligence evidence, it is not feasible for the individual to be *forgotten*.

The following table indicates which fields are considered to be authorizations, and are preserved for due diligence purposes. These exceptions are therefore not erasable.

Preserved Personal Data

Record	Field
Assessment	Completed By

Purging Inactive User Accounts

The Right to be Forgotten service may also be used for purging user accounts that are no longer active, that is, those of users who have left the organization or are no longer involved in using the system. This is achieved by calling the API in the same way as if processing a Right to be Forgotten request.

Automatic date-based purging, such as a period of time since the user's account was disabled, or since their last login, is not provided, however the retailer/portal owner could feasibly develop a means of recursively calling the API based on a set of logic rules.

Security and Logging

The following Service and Endpoint configurations must be configured within the Supplier Evaluation Admin area, and assigned to the External Systems that are to have access to the Right to be Forgotten service:

Service:	DATAPRIV	Data Privacy Service
Endpoint:	DATAPRIV_VERIFY_POST	Creates a new verification request for a login id
Endpoint:	DATAPRIV_ERASE_POST	Creates a new Erasure request and returns a link to the record's details
Endpoint:	DATAPRIV_ERASE_GET	Retrieves Erase details for a given id

All calls to the service will be logged in the Supplier Evaluation Web Service Log.

The Data Privacy records created by the service are not viewable in Supplier Evaluation, and are not purged.

Entries within the Supplier Evaluation Change History logs, Attachments logs, and Comments logs use a foreign key to reference the name of the user who updated the record, attached the file, or added the comment. This means that if the name of the user is obfuscated in the User record, the name will automatically become overwritten with randomly-generated text in the logs.

Updates to records to erase personal data through the Data Privacy API are not recorded in the record's Change History log. This avoids showing what the value was before and after obfuscation.

Example of Using the Right to be Forgotten Service

The stages involved in executing a Right to be Forgotten request are as follows. For this worked example, the call is made using a web service testing application.

1. Download the datatpriv WADL definition from the /services area of the Supplier Evaluation portal.
2. Use the Right to Access service to retrieve the individual's set of personal data (see [Right to Access Requests](#)).
3. Edit the returned XML message using a suitable text editor, removing duplicate user and person elements.

Figure B-5 XML for Forgotten Returned

```

1 <ns0:dataPrivRequestDTO xmlns:ns0="http://www.micros.com/creation:
  http://www.micros.com/creations/core/domain/dto/vlp0/full" xmlns:r
2   <ns0:name>John Smith</ns0:name>
3   <ns0:id>17</ns0:id>
4   <ns0:createdOn>2017-10-25T14:26:09.000+01:00</ns0:createdOn>
5   <ns0:person>
6     <ns0:id>21</ns0:id>
7     <ns0:name>John Smith</ns0:name>
8     <ns0:email>jsmith@example.com</ns0:email>
9     <ns0:phoneNumber>01234 56789</ns0:phoneNumber>
10    <ns0:mobileNumber>07890 12345</ns0:mobileNumber>
11    <ns0:fax>01234 98765</ns0:fax>
12    <ns0:address>
13      <ns0:street></ns0:street>
14      <ns0:city></ns0:city>
15      <ns0:country></ns0:country>
16      <ns0:postalCode></ns0:postalCode>
17    </ns0:address>
18  </ns0:person>
19  <ns0:person>
20    <ns0:id>7</ns0:id>
21    <ns0:name>John Smith</ns0:name>
22    <ns0:email>jsmith-supplier@example.com</ns0:email>
23    <ns0:phoneNumber>09876 54321</ns0:phoneNumber>
24    <ns0:mobileNumber>07890 12345</ns0:mobileNumber>
25    <ns0:fax>09876 54320</ns0:fax>
26    <ns0:address>
27      <ns0:street></ns0:street>
28      <ns0:city></ns0:city>
29      <ns0:country></ns0:country>
30      <ns0:postalCode></ns0:postalCode>
31    </ns0:address>
32  </ns0:person>
33  <ns0:user>
34    <ns0:id>21</ns0:id>
35    <ns0:loginId>jsmith</ns0:loginId>
36    <ns0:personId>21</ns0:personId>
37    <ns0:userType>RETAILER</ns0:userType>
38  </ns0:user>
39  <ns0:user>
40    <ns0:id>7</ns0:id>
41    <ns0:loginId>john smith</ns0:loginId>
42    <ns0:name>John Smith</ns0:name>
43    <ns0:email>jsmith-supplier@example.com</ns0:email>
44    <ns0:userType>SUPPLIER</ns0:userType>
45    <ns0:address>
46      <ns0:street></ns0:street>
47      <ns0:city></ns0:city>
48      <ns0:country></ns0:country>
49      <ns0:postalCode></ns0:postalCode>
50    </ns0:address>
51  </ns0:user>
52  <ns0:designAgency>
53    <ns0:agencyName></ns0:agencyName>
54    <ns0:address>
55      <ns0:street></ns0:street>
56      <ns0:city></ns0:city>
57      <ns0:country></ns0:country>
58      <ns0:postalCode></ns0:postalCode>
59    </ns0:address>
60  </ns0:designAgency>
61 </ns0:dataPrivRequestDTO>

```

In this example, there are two pairs of user and person elements because there are two user accounts with the name John Smith - one for a retailer user and one for a supplier user (user name and email address are not unique, it is the login id that uniquely identifies a user).

In order to pass validation, there must only be one user and person element in the submitted XML message. Remove any user and person elements that are not the individual being erased. Delete all the data between the opening and closing tags, just leaving a single person and user:

Figure B-6 XML for Forgotten One Person

```

1 <ns0:dataPrivRequestDTO xmlns:ns0="http://www.micros.com/creation:
  http://www.micros.com/creations/core/domain/dto/vlp0/full" xmlns:r
2   <ns0:name>John Smith</ns0:name>
3   <ns0:id>17</ns0:id>
4   <ns0:createdOn>2017-10-25T14:26:09.000+01:00</ns0:createdOn>
5   <ns0:person>
6     <ns0:id>21</ns0:id>
7     <ns0:name>John Smith</ns0:name>
8     <ns0:email>jsmith@example.com</ns0:email>
9     <ns0:phoneNumber>01234 56789</ns0:phoneNumber>
10    <ns0:mobileNumber>07890 12345</ns0:mobileNumber>
11    <ns0:fax>01234 98765</ns0:fax>
12    <ns0:address>
13      <ns0:street></ns0:street>
14      <ns0:city></ns0:city>
15      <ns0:country></ns0:country>
16      <ns0:postalCode></ns0:postalCode>
17    </ns0:address>
18  </ns0:person>
19  <ns0:user>
20    <ns0:id>21</ns0:id>
21    <ns0:loginId>jsmith</ns0:loginId>
22    <ns0:personId>21</ns0:personId>
23    <ns0:userType>RETAILER</ns0:userType>
24  </ns0:user>
25  <ns0:designAgency>
26    <ns0:agencyName></ns0:agencyName>
27    <ns0:address>
28      <ns0:street></ns0:street>
29      <ns0:city></ns0:city>
30      <ns0:country></ns0:country>
31      <ns0:postalCode></ns0:postalCode>
32    </ns0:address>
33  </ns0:designAgency>
34 </ns0:dataPrivRequestDTO>

```

 Note:

The message is XML and can be edited using any UTF-8 compliant text editor.

It is vital that the structure and syntax of the message is maintained (with matching pairs of tags), so editing with a tool that recognizes the XML file format is recommended. For details of the message structure, see [Right to Access Requests](#).

Metadata fields that are included in the message to identify the records by their code/id, name/description or type are not erased. For details of which fields may be erased, see the [Erasable Personal Data](#) table.

Person/User/Contact Relationships

There may be a combination of user, person, and contact elements for an individual, depending on their role within the system:

- Each user of the system has a single User record and a single Person record.
- If a supplier user is a contact, in addition to their User and Person records, they have a Contact record for each of their designated contact roles.
- If an individual is a contact but they are not actually a user of the system, they have a Person record, plus a Contact record for each of their designated contact roles.

When submitting a Right to be Forgotten request, the following validation checks are applied:

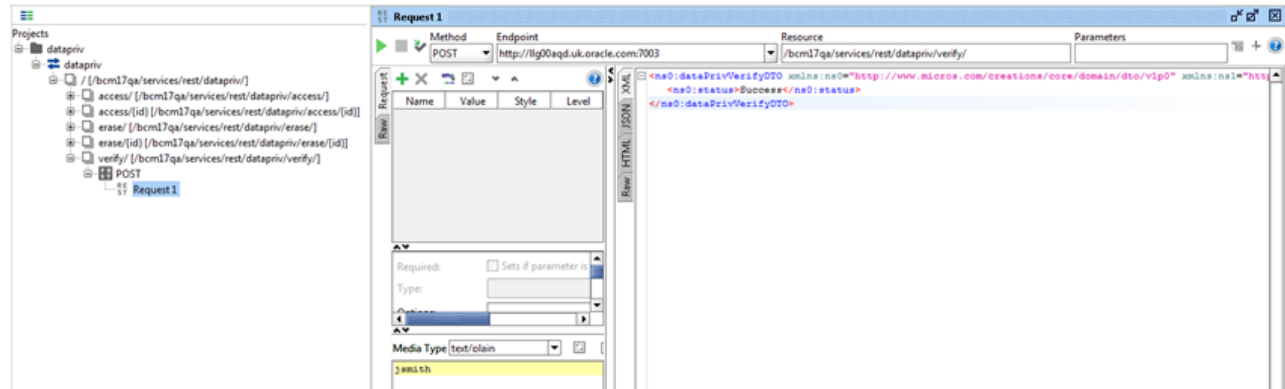
- Only one user element must be present.
 - Only one person must be present.
 - The associated person element must be present for a user or a contact element.
 - The personId in the user element must match the id in the person element.
 - The personId in the contact element must match the id in the person element.
 - The user id in the contact element must match that in the user element.
 - All contacts must be linked to the same person.
 - A person element must have an associated user or contact element.
4. Make further edits to the XML message to check that the data in the elements for designAgency, supplier, and so on, all relate to the individual making the request. If not, delete the elements.

When submitted, the personal data elements that remain in the message will result in the corresponding fields being erased in Supplier Evaluation (the field contents are replaced with randomly-generated text).

If an element is removed, the field contents remain unchanged.

5. Optionally submit a POST call to the services/rest/datapriv/verify service, passing the login id of the individual (obtained from the User element of the XML message returned in Step 2) in the payload:

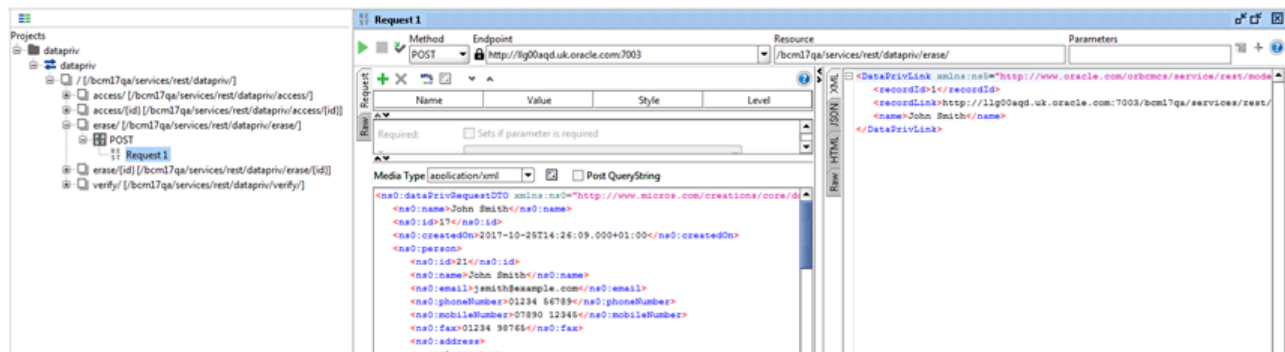
Figure B-7 Forgotten Post Call



This step checks whether the individual is responsible for the completion of an Assessment, in which case they will not be permitted to be erased.

- Submit a POST call to the services/rest/datapriv/erase service, passing the XML message in the payload:

Figure B-8 Forgotten Post Second Call

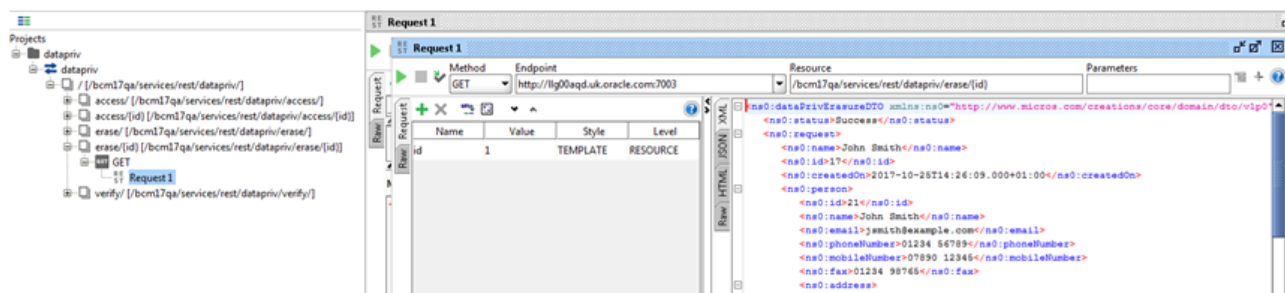


A recordLink element is returned with a link to a Data Privacy record.

For possible failure conditions, see the [Error Messages](#) table.

- Submit a GET call to the services/rest/datapriv/erase service, passing the returned recordId value as the id parameter:

Figure B-9 Forgotten Get Call



A confirmation message is returned, with the original data in a `dataprivErasureDTO` root element.

8. Optionally perform another a Right to Access request to check that the personal data has been erased as expected.

Otherwise, viewing the erased records within Supplier Evaluation will show the obfuscated data:

Figure B-10 Forgotten Removed Data

User Details	Custom Fields	Change History	Attachments
= Details			
Name:	au1W4VXcQvXGcXlFHOYVoTDCVNB0YwY6G08BKmLEdpmYCrymM	Email:	2OF8C9a8xCAvfu@example.com
Login Id:	NWPubeoXHLPyommZ2fJl	Phone:	n40PPAmy9Kue5WakaeSatRz9hm1I
Login id Disabled?:	Yes	Mobile Phone:	bttsvKGDAdBkmswXAMR-EdUOgClu
Password Expired?:	Yes	Fax:	0VCYxKw6zAe1WAlRFFokhRd7T7POC
Job Title:	-		
Department:	-		
Team Manager:	-		
Delegates:	-		
Language:	English (British)		
Time Zone:	Greenwich Mean Time (Europe/London)		
Comments:	-		
Testing Support Enabled?:	-		
External Authentication:	No		
= Address			
Local Address			
Country:	-		
Address 1:	nKmk3UoztExFo3Muappq		
Address 2:	PcuhoUWslUHFt5Zbryng		
City:	BVhQjUUTofYOnf23PLX		
County:	5Cj364AU4CwRxb4AQYOm		
Post Code:	J51uG 		
GPS latitude:	-		
GPS longitude:	- 		

9. Advise the requestor of the outcome of their request.

Other Aspects of Data Privacy

This section describes some other aspects of Data Privacy, and how they relate to Supplier Evaluation.



Note:

The data passed to IDCS or OCI IAM includes personal information (user name and email address). The user may also manually enter personal contact information in IDCS or OCI IAM.

The Supplier Evaluation *Right to Access* and *Right to Erase/Right to be Forgotten* data privacy services are not extended to access IDCS or OCI IAM, so the retailer must also check for a corresponding user profile in IDCS or OCI IAM, and where necessary, remove the personal information from IDCS or OCI IAM.

When a new user is created in Supplier Evaluation, the User record has a flag set to indicate that its data has been passed to IDCS or OCI IAM.

Data Portability

Data Portability is the right for the data subject to receive the data concerning them, in a common, structured machine-readable format. The Right to Access service and other APIs provide access to the majority of Supplier Evaluation data. The Reports module also allows the export of data in various formats.

Privacy by Design and Data Minimization

Privacy by Design and Data Minimization relates to making data privacy and protection integral to the design of the system, such as minimizing the data set to that which is necessary to perform the function of the system, and restricting the access to personal data to just the users that need it. The personal data held in Supplier Evaluation is already minimal; it relates to business contact data.

Access to the Supplier Evaluation data is restricted by the Permissions security model, based on the user's roles and what organization they are associated to; the configurable access controls are granular, down the level of individual fields.

There is no need to specifically mark fields as *sensitive* in Supplier Evaluation, however, the retailer/portal owner could chose to do so by configuring the field label system text themselves.

Data Deletion

The deletion of data is limited due to the due diligence nature of Supplier Evaluation. Certain data must be retained indefinitely, so options to delete or purge data are limited to a select few areas. Where data is deleted, it is not possible for it to be re-accessed.

The Right to be Forgotten solution includes the retailer/portal owner's ability to purge personal data from inactive user accounts. As certain data cannot be physically deleted, the erasure is by obfuscating the appropriate fields.

Notice and Consent

Notice and Consent is necessary where personal data is being held or processed. In Supplier Evaluation, the Terms and Conditions pages can be configured by the retailer/portal owner to include the necessary notices for the user to accept or otherwise.

Availability

Backup and disaster recovery facilities for Supplier Evaluation are provided as standard as part of the Cloud Service solution.

The majority of Supplier Evaluation data is available for export using the suite of secure APIs.

Tracking Technologies

Supplier Evaluation does not include any Tracking Technologies such as IP addresses or device fingerprints. A single cookie is used, for session connectivity.

Separation of Duties

Some tasks, such as the approval or completion of an Audit, require actions by separate users; the completion of the tasks is controlled in Supplier Evaluation by workflow rules.

Logging

Maintenance of Supplier Evaluation records is recorded in the Change History log. Read access is not logged.

The Web Service log records details of all API calls; these are accessible only to administrators.

The Supplier Evaluation data is secured by HTTPS/SSL in transit; data at rest in the MySQL application database is not encrypted.

C

Appendix: Using the Report Attachment API

The Supplier Evaluation Reports module provides a suite of *standard* reports with the facility for building new reports using the Oracle BI Publisher Template Builder plug-in for Word. Reports are scheduled in the Report Scheduler area where the recipients are assigned, along with whether the report is a single or recurring run, and the type of output (PDF, Excel, RTF, or HTML). When the report has been generated, it is stored as a file attachment in the Report Outputs area, within a hierarchical folder structure.

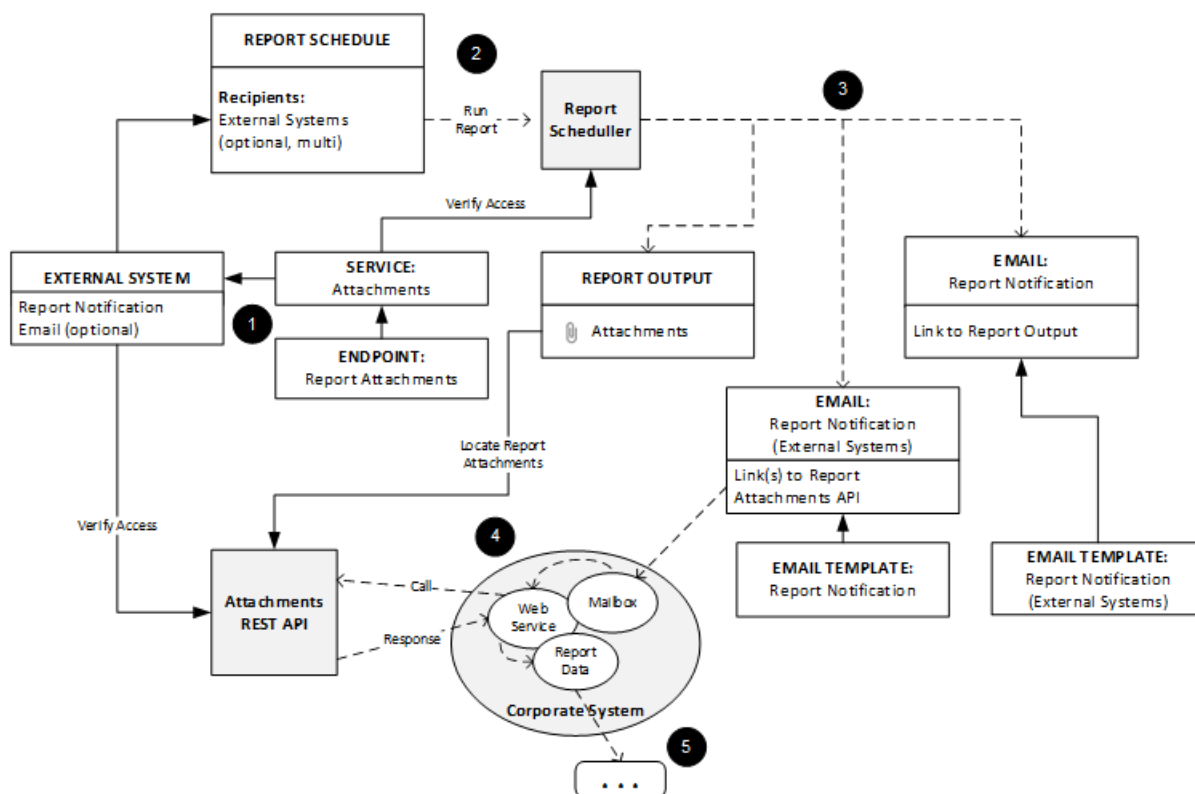
The Report Attachment API provides a means of automatically retrieving the report output attachments from the Report Outputs area, for taking the data into an external system for further analysis or processing.

When a report is generated, the recipients are notified by an email which contains a link to the report. An external system can be defined as a recipient. The External System record in the Supplier Evaluation Admin area can have an email address assigned, making it available for selection as the recipient of a report. The email sent to the external system contains a link to the report along with a link to the actual attachment file; the Report Attachment API can then be used to retrieve the attachment using this link.

The data returned by the API is a Base64 encoded text string which must then be processed by a Base64 decoder to convert it to the actual report file.

The following diagram describes the stages of the process for using the Report Attachment API to retrieve a report output.

Figure C-1 Process Flow for Using the Report Attachment API



1. To access the Report Attachment API, the external system must be registered in Supplier Evaluation as an External System with the ATTACHMENT service, the REPORT_ATTACHMENT_GET endpoint, and an email address assigned.
2. When a report is scheduled, the external system is selected as a recipient.
3. When the report runs, if the recipient is an external system, an email is sent containing a link to the report output attachment (may be more than one if multiple formats have been selected).

In order to prevent the URI to the Attachment record being predictable, a 32-character non-sequential globalId key is used.

The template of the REPORTNOTIFICATION email may be configured in the Supplier Evaluation Admin area; the default content is:

Subject: Report [Report Name] is ready to view

Body: The report [Report Name] has been created.

Use the following link(s) to retrieve the attachment data from the attachment API.

File name: [File Name].[File Type] [Link to Report Attachments API] File name: [File Name] File Type: [File Type] [Link to Report Attachments API]

4. The external system initiates the call to the Report Attachment API. This may be triggered automatically on receipt of the email, or could be triggered by another event, by a scheduled job, or by a manual user action.

5. The external system processes the returned messages and report attachment data according to its intended purpose. The data must first be decoded from Base64 encoding.

Response Example:

```
<ns1:attachmentFullDTO
  xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/full"
  xmlns:ns2="http://www.micros.com/creations/core/domain/dto/v1p0/simple">
  <ns1:attachedBy>
    <ns2:code>oau</ns2:code>
    <ns2:disabled>>false</ns2:disabled>
    <ns2:externalAuthenticationUser>>false</ns2:externalAuthenticationUser>
    <ns2:timeZone>Europe/London</ns2:timeZone>
    <ns2:userType>RETAILER</ns2:userType>
    <ns2:id>22</ns2:id>
  </ns1:attachedBy>
  <ns1:attachedOn>2018-06-28T10:20:27.000+01:00</ns1:attachedOn>
  <ns1:dataLength>3786</ns1:dataLength>
  <ns1:fileName>RT1.pdf</ns1:fileName>
  <ns1:localeData>
    <ns1:description><![CDATA[Test]]></ns1:description>
    <ns1:id>1190</ns1:id>
  </ns1:localeData>
  <ns1:id>1190</ns1:id>
  <ns1:ownerClass>com.micros.creations.core.domain.ReportResult</ns1:ownerClass>
  <ns1:ownerId>156</ns1:ownerId>
  <ns1:data>JVBERi0xLjYNCjUgMCBvYmoNCjw8DQovVHlwZSAvUGFnZQ0KL1Bhc...</ns1:data>
</ns1:attachmentFullDTO>
```



Note:

It is the responsibility of the retailer/portal owner to build an application to call the Report Attachment API and handle the returned messages and data, including any further processing of the data or interfaces with external systems.

For portals that use the Enhanced Access Control (EAC) level of permissions, the retrieval of reports as attachments through the API does not apply EAC area permissions filtering (as is the case for all other Supplier Evaluation APIs).

D

Appendix: Accessibility

Accessibility involves making your application usable for persons with disabilities such as low vision or blindness, deafness, or other physical limitations. This means, for example, creating applications that can be:

- Used without a mouse (keyboard only).
- Used with assistive technologies such as screen readers and screen magnifiers.
- Used without reliance on sound, color, animation, or timing.

User Interface

This user interface is designed using Oracle JavaScript Extension Toolkit (JET). Oracle JET components have built-in accessibility support that conforms to the Web Content Accessibility Guidelines version 2.0 at the AA level (WCAG 2.0 AA), developed by the World Wide Web Consortium (W3C). Please note that since different browsers themselves support accessibility somewhat differently, the user experience tends to differ on different web browsers.

Oracle JET components provide support for:

- Keyboard and touch navigation
Oracle JET components follow the Web Accessibility Initiative - Accessible Rich Internet Application (WAI-ARIA) guidelines.
- Zoom
Oracle JET supports browser zooming up to 200%.
- Screen reader
Oracle JET supports screen readers such as JAWS, Apple VoiceOver, and Google Talkbalk by generating content that complies with WAI-ARIA standards, and no special mode is needed.
- Oracle JET component roles and names
Each Oracle JET component has an appropriate role, such as button, link, and so on, and each component supports an associated name (label), if applicable.
- Sufficient color contrast
Oracle JET provides the Alta theme which is designed to provide a luminosity contrast ratio of at least 4.5:1.

Besides the use of JET components, the following updates were made to the application:

- Added alternative texts to images/logos/icons as needed.
- Added labels as needed to GUI widgets.
- Color Contrast was adjusted.
- Improved error identification and display, so screens readers can read meaningful information.

Appendix: LCA Third-Party Integration

Supplier Evaluation can be configured to allow for the integration of Assessments with third parties. This is of use where the retailer or supplier is using a third party such as a Life Cycle Assessment (LCA) partner, to source specialist data, for example sustainability, environmental and ethical analysis information about a supplier or its products.

For a third party to integrate with the Supplier Evaluation system, the retailer portal owner must register them as an Accredited Body and create an External System account, granting them access to the Scorecard API (they may also be granted access to other APIs for the retrieval of supporting Supplier, Site or Product data).

Assessments can be configured to be the type which will be used to capture data from a third party, in which case one or more Accredited Bodies are assigned. When an Assessment of that type is created, one of the permitted Accredited Bodies is selected. That third party is then able to retrieve the Assessment using the Scorecard REST API to supply their data by populating the answers to its questions. They then submit the Assessment's response answers using the same API.

Once an Assessment is returned it is reviewed, approved and completed. The data provided by the third party then being available for viewing within the Assessment and for use in reports. Summary Attributes may be configured to label responses in a way that they can be shown in the information panels of the associated, Assessment, Site or Product.

The integration may be *supplier owned*, where the supplier liaises with the third party and has responsibility for the final completion of the Assessment, or *retailer owned*, where the retailer typically works with the third party without involving the supplier.

When registering a third party as an Accredited Body it is possible to configure a link to the third party's proprietary system, in which case an option will appear in the top level Accredited Bodies navigation Tasks menu.

Accredited Bodies

The retailer controls which third parties may collaborate by creating an Accredited Body record to register them as a partner, and an External System record to grant them access to the Scorecard API (endpoints for read and update access). The combination of Accredited Body and External System forms the security controls for which third parties may access the API, and to restrict access to specific Assessments.

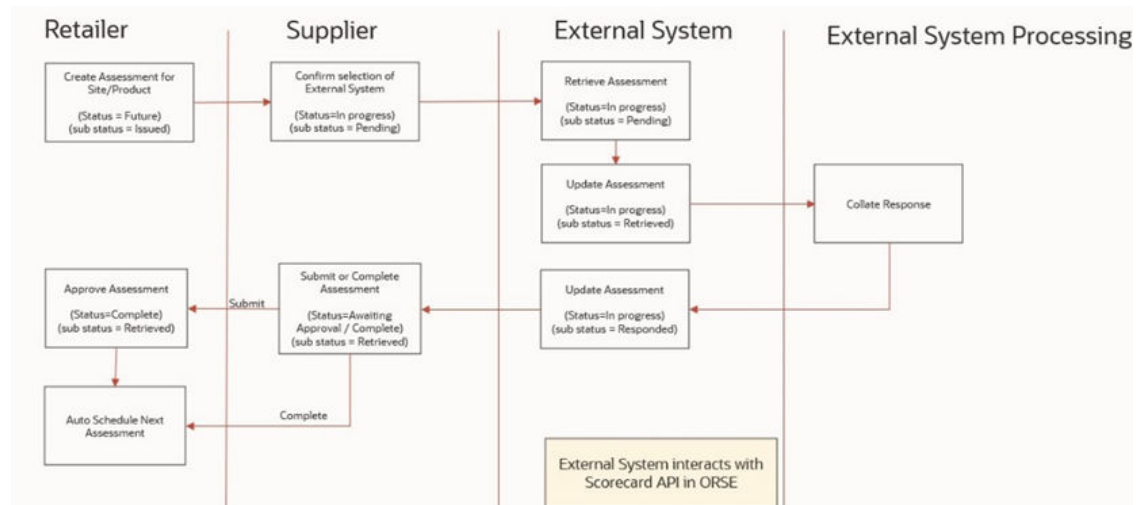
An Integration Type field on the Assessment Template identifies those where data is to be sourced from third parties. The options of Retailer Owned and Supplier Owned define whether the Assessments are being managed by the retailer or the supplier. Once an Accredited Body is registered, it can be selected within the template as being one of the third parties who may be assigned to Assessments of that type (multiple Accredited Bodies may be assigned to a template).

When an Assessment is created from a template with an Accredited Body assigned, the user selects which single third party will be used to source the data for that Assessment. When the Assign Accredited Body action is used, the status of the Assessment becomes In Progress, and its sub status is set to Pending - meaning it is available for retrieval by the designated third party.

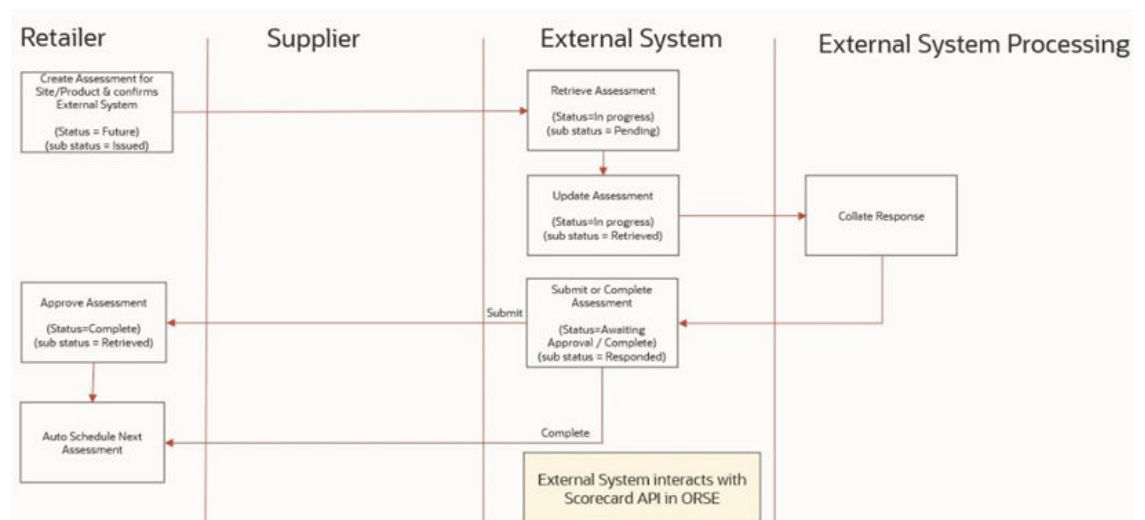
Assessments Integration

The process for third parties to retrieve, respond to, and return Assessments they have been assigned is shown in the following diagrams. The key difference between the two models is: for Supplier Owned, the supplier is notified of the new Assessment and returns the populated Assessment to the retailer, whereas for Retailer Owned, the Assessment is created by the retailer and is assigned directly to the third party, who returns the populated Assessment to the retailer.

Supplier Owned Integration Type

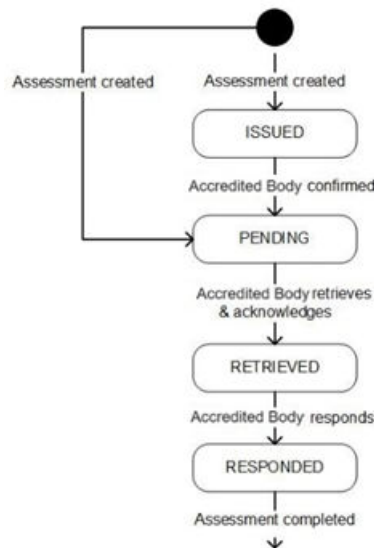


Retailer Owned Integration Type



Sub Status

A Sub Status field on the Assessment is used to control the workflow of those being progressed with a third party:



Third parties retrieve Assessments that have been assigned to them and submit an update to set the sub status to indicate they have been retrieved. Once the questions have been populated with responses that represent the data being provided, a further update is submitted, setting the sub status to indicate they have responded.

Assessment templates have options to set whether the third party and/or the supplier can set an Assessment as being complete, or whether they are submitted for the retailer's approval.

Using the Scorecard API

The Scorecard API is a RESTful API with XML based interaction and payload, supporting GET and PUT methods for the retrieval and update of Assessments.

Access to the API and its endpoints is controlled by the retailer portal owner, as is the access to the individual Assessments. The retailer portal owner may also grant access to other APIs, such as the Supplier, Site or Product APIs for retrieval of additional supplier or item information. Authentication can be configured utilizing both OAuth 2.0 and Basic Authentication, this must also be configured by the retailer portal owner. The recommendation is to utilize OAuth wherever possible.

The third party's access to Assessments is restricted to those to which they are the designated Accredited Body, and which are at the appropriate status for retrieval.

Assessments are retrieved with a GET method, and then the responses to the Assessment's questions can then be returned with a PUT method to update the Assessment. The third party should create a process to periodically call the API to determine if new assessments are available for retrieval. The system does not send notifications when Assessments are assigned to a third party.

When calling the Assessment API, the `modifiedSince` and `modifiedUntil` parameters can be used to focus on any assessments that have been created or amended since the previous call. Filtering on the status of the Assessment can be used to exclude any that have previously been completed since the previous call. To facilitate this, the external system will need to record the retailer specific ORSE portal's URL, authentication method, last called time (for the polling service) and integration frequency/time.

The response to the Get List of Values call will provide a list of links to the Assessments available for retrieval. The individual Assessments are then retrieved by a Retrieve Record by ID call. Once retrieved a call should be made to update the Sub Status of the Assessment,

setting it to RETRIEVED to indicate it has been retrieved by the third party – either with the Update Sub Status endpoint or with a full Update message.

The third party can then analyze the questions in the Assessment in the external system and collate the answers before returning their responses with a full Update response. This is achieved by returning the retrieved payload with the responses populated and the Sub Status set to RESPONDED to indicate it is ready for the retailer or supplier to review.

Assessments may be configured to allow the third party to set the Status to COMPLETE, or to set to AWAITING APPROVAL, in which case the supplier or retailer will review for approval before completing it.

The remainder of this section provides more detail on the integration process and the two use cases.

Assessments Integration Process

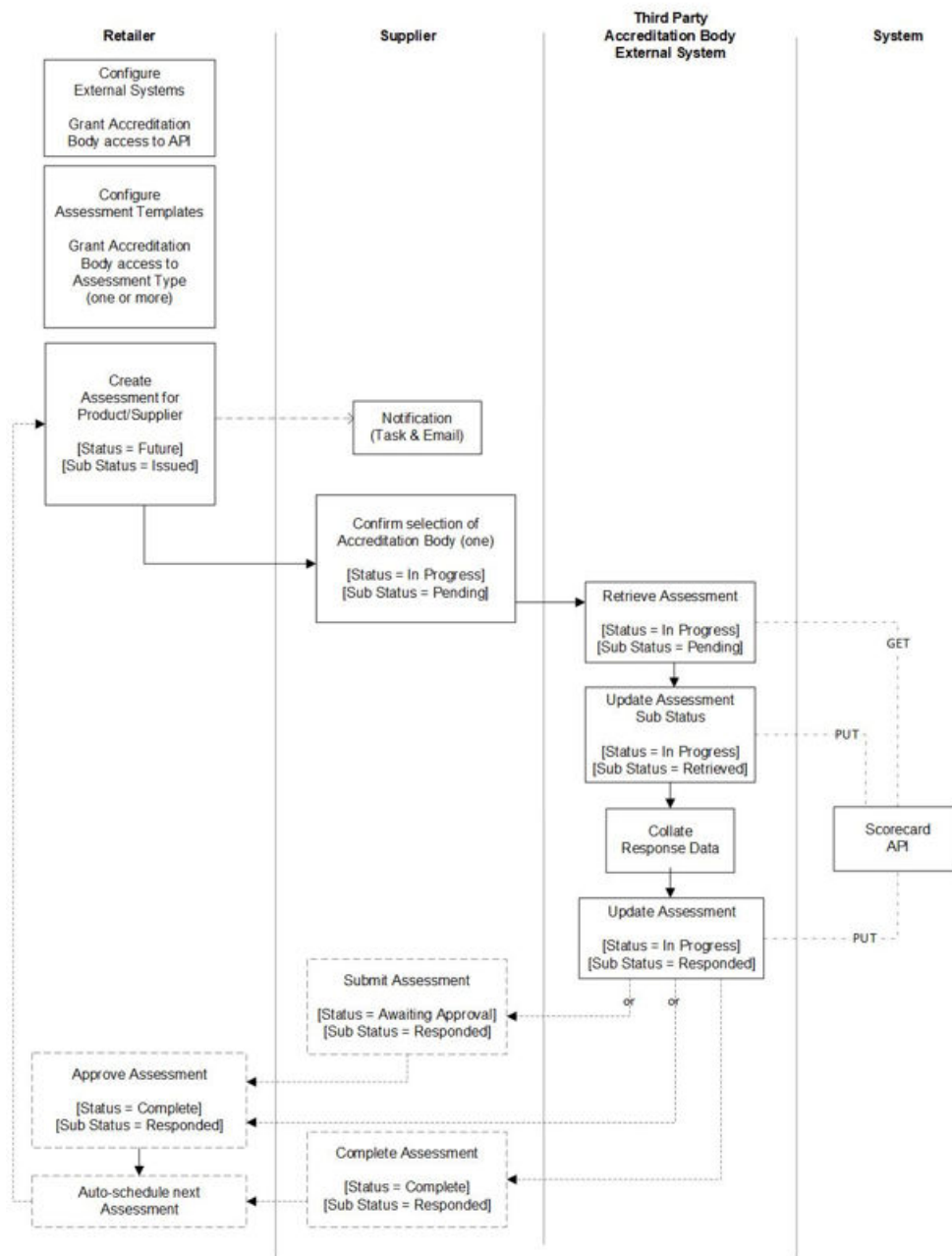
Consider the following when using the Scorecard API to provide third party data as part of the Assessments integration process.

- Assessments may be Item Assessments for an item/product or Site Assessments for a supplier's site location. The type is indicated by the **Assessment Level** field. The same process and endpoints are used for both types.
- As the same Product Number (SKU) may be used by different suppliers (and may change), it is recommended that the **Product Code** be used to identify a product's Assessments, as it is the unique key of the Product Record.
- Assessment questions have a **Question Code**, which can be used to map the responses to the questions. Where questions have multiple answer options (checkboxes, radio buttons and picklists), the individual permissible answers each have an **Answer Code** to be used when populating the answer.
- The **Sub Status** is initially PENDING, with Status being IN PROGRESS. This is the state when it should be retrieved. Once retrieved the Sub Status should be updated to RETRIEVED. When responses are submitted the Sub Status should be set to RESPONDED, and the Status set to COMPLETE or AWAITING APPROVAL.
 - The Assessment's template will control whether the third party may progress the Assessment to the Completed status - populating the Completed On (date), Completed By (name of the External System), or whether it must be set to Awaiting Approval for the retailer or supplier to finalize the workflow.
- The third party will liaise offline with the retailer and/or supplier if further action is required.
- The API applies minimal **validation**; full validation and business logic (calculations, workflow triggers, status changes) when an Assessment record is subsequently edited and progressed within the system. Validation is included in the API to:
 - Prevent the Accredited Body and Integration Type, *Accredited Body Can Complete Assessment* and *Retailer Approval Required* values being changed.
 - Control whether the Status can be set to Complete or Awaiting Approval based on the rules for the assessment type:
 - * If Accredited Body Can Complete Assessment is Yes, Status can only be set to Complete if Retailer Approval Required is No.
 - * If Accredited Body Can Complete Assessment is Yes, Status can only be set to Awaiting Approval if Retailer Approval Required is Yes.

- It is important that when resubmitting an Assessment for update that the **full payload remains populated** as it was when retrieved - with just the response, status, and any comments data being updated - otherwise the data will be cleared.
- For the scenario where a third party is to **periodically update or refresh** data as an on-going task, the assessments can be configured by the retailer to **automatically schedule the next Assessment** on completion.

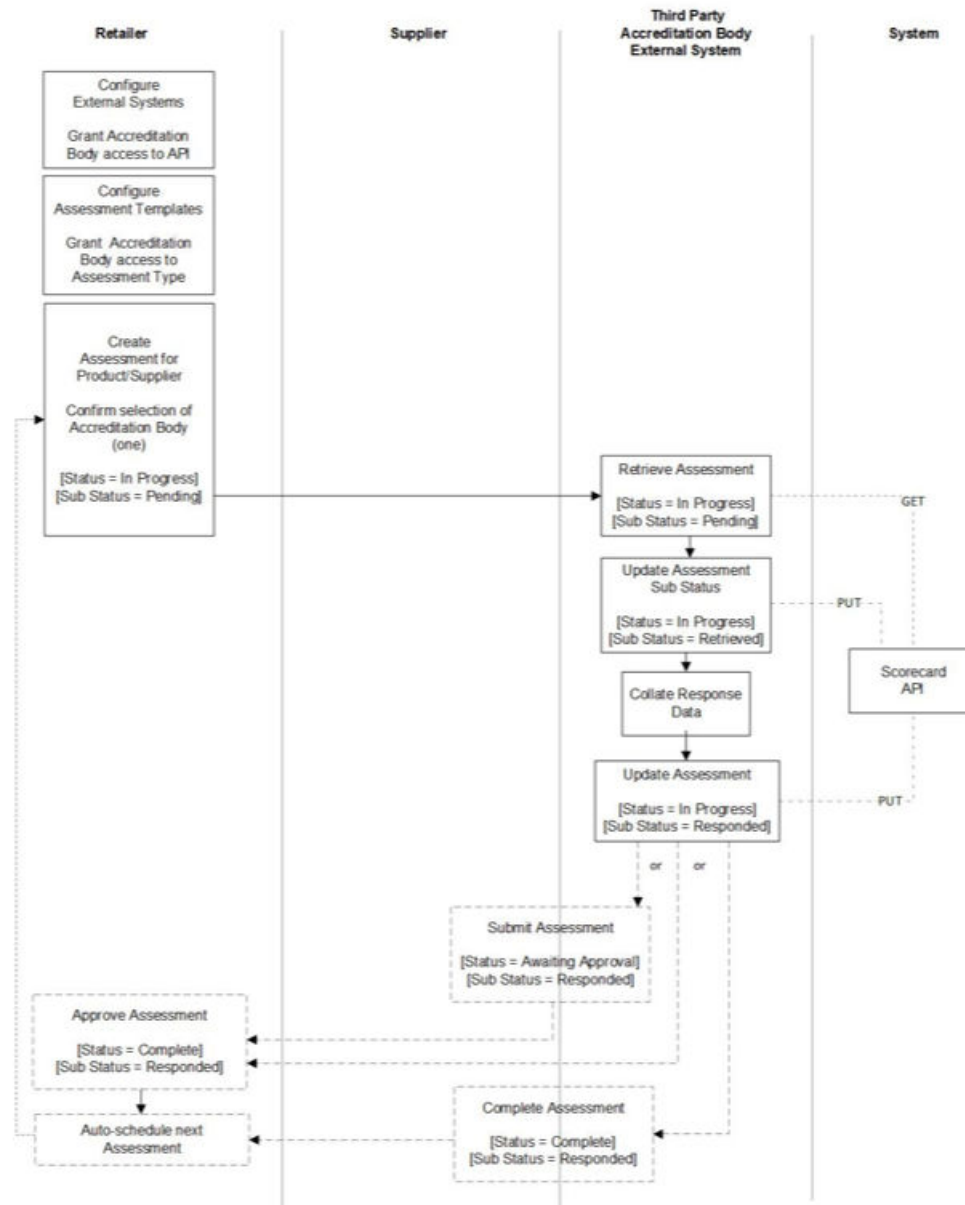
Supplier Owned Integration Type

For this use case, the retailer requests individual suppliers to progress an Assessment which contains questions for which responses requires third party data. The retailer typically creates the Assessment for the supplier to take responsibility for its completion; alternatively, the supplier may create the Assessment themselves. An Assessment may contain a combination of questions requiring responses to be sourced from the third party and also responses from the supplier. The supplier then submits the Assessment to the retailer if approval is required, otherwise completes it themselves.



Retailer Owned Integration Type

For this use case, the retailer requests a third party to provide data for a product or range of products. The retailer creates the Assessment, which is associated to a supplier (so the supplier will also have access to the Assessment). The third party submits the Assessment to the retailer if approval is required, else completes it themselves.



Enabling Sub Status Endpoints

To enable the endpoints for updating an Assessment's sub status, add the following Endpoint Access records to the Scorecard Service Access record in the Roles & Permissions section of the Admin area:

Code: SCORECARD

Description: Scorecard RESTful Service

Endpoint Access: SCORECARD_SUB_STATUS and
SCORECARD_SUB_STATUS_BY_CODE

Select Endpoint Access

Search scorecard_sub

Available

☐ Description
☒ SCORECARD_SUB_STATUS - Updates an In Progress Scorecards sub status and returns a link to the record's details
☒ SCORECARD_SUB_STATUS_BY_CODE - Locates a scorecard by its code to update it's sub status, returns a link to the record's details

Selected

☒ Description
☒ SCORECARD_CHECK - Retrieves the header of information of a single assessment
☒ SCORECARD_EXISTS - Retrieves the detail of a single assessment by its business key
☒ SCORECARD_GET - Retrieves the details of a single assessments
☒ SCORECARD_GETVALUES - Retrieves a single page list of assessments
☒ SCORECARD_SUB_STATUS - Updates an In Progress Scorecards sub status and returns a link to the record's details
☒ SCORECARD_SUB_STATUS_BY_CODE - Locates a scorecard by its code to update it's sub status, returns a link to the record's details

Page 1 of 1

Ok Cancel Clear

Glossary

API

Application Programming Interface, a type of web service. It provides a way of communicating with a particular computer program or internet service, such as the interfaces between Supplier Evaluation and other systems, by exchanging data as XML files.

DELETE

A web service call used to delete an existing resource (not generally used in the APIs exposed in Supplier Evaluation).

GET

A web service call used to retrieve data without modifying it, for example, to request details of specifications.

HEAD

Identical to GET except that the server does not return a message-body in the response. The header information retrieved is the same as for a GET request and obtains meta-information without transferring the entity-body itself.

HTTP

Hyper Text Transfer Protocol, an application protocol used as the foundation for data communication on the World Wide Web.

HTTPS

Communication over Hyper Text Transfer Protocol which is protected with an additional security layer. Used to authenticate the site visited and protect the exchanged data.

LCA

Life Cycle Assessment, a methodology for assessing environmental impacts associated with the stages of the life cycle of a product, process, or service. LCA partners may be used in the process of assessing products or supplier for the retailer (as a third-party accredited body).

oneTimePad

A secure encryption technique which uses a string of randomly generated numbers to produce a key for encoding and decoding a message. Both the sender and receiver of the message have a copy of the key and each key (or section of the key) is used only once.

POST

A web service call used to create new resources, for example, to create a new supplier or new user record.

PUT

A web service call used to update an existing resource, for example, to update/amend an existing supplier or user record.

REST / RESTful

Representational State Transfer, an architectural style consisting of a coordinated set of architectural constraints applied to components, connectors, and data elements, within a distributed system.

In Supplier Evaluation, most new services or new versions of services will use RESTful architecture.

SOAP

Simple Object Access Protocol, an XML-based protocol for exchanging data over HTTP. In Supplier Evaluation, SOAP services are developed for specific, definite requirements or tend to be associated with historical services.

URI

Uniform Resource Identifier, a string of characters used to identify the name of a resource and enable it to be accessed over a network such as the World Wide Web. A common example of a URI is the Uniform Resource Locator (URL), or web address.

WADL

Web Application Description Language, an XML-based interface description language for describing the functionality offered by a RESTful web service.

WSDL

Web Services Description Language, an XML-based interface description language for describing the functionality offered by a SOAP web service.

XML

Extensible Markup Language, a language used to encode a document in a way which is both human readable and machine readable.

Markup symbols are used to define the content of the document and to indicate what that content is associated to in the sending or receiving system.

XSD

XML Schema Definition, defines the structure and data type of the elements in an XML document. XSD is used to validate the content of an XML document.