Oracle Financial Services Fraud Enterprise Edition (Real Time Fraud)

Customization Guide

Release 8.1.2.6.0

October 2023

F82825-01

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.

For information on third party licenses, click here.

Document Control

Table 1: Document Control

Version Number	Revision date	Change Log
8.1.2.6.0	October 2023	There are no updates to this guide in this release.
8.1.2.5.0	June 2023	Created the first version of Fraud Enterprise Edition (Real Time Fraud) Customization Guide for 8.1.2.5.0 Release.

Table of Contents

1 About this Guide 5

1.1 Summary 5

1.2 Audience 5

1.3 Related Documents 5

1.4 Conventions Used in this Guide 5

2 Adding and Displaying New Data on Details Page 7

2.1 Data Model Upload 7

2.2 Database Changes 8

2.3 HTML and JavaScript Changes 11

2.3.1 HTML 11

2.3.2 JavaScript 13

2.4 Internationalization 14

3 Modifying Existing Sections 16

3.1 Data Model Changes 16

3.2 Database Changes 16

3.3 HTML or JavaScript Changes 16

3.3.1 HTML 16

3.3.2 JavaScript 16

4 Example Cases 17

4.1 Wire Example Case 17

4.1.1 Scenario 17

4.1.2 Steps For Customization 17

4.2 Card Example Case 18

4.2.1 Scenario 18

4.2.2 Steps for Customization 18

5 OFSAA Support Contact Details 20

6 Send Us Your Comments 21

1 About this Guide

This guide explains the concept of customizing the details page of the Oracle Financial Services Real Time Fraud Enterprise Edition Application by adding new sections or modifying the existing sections to produce alerts as required and provides comprehensive instructions for configuration.

Topics:

- Summary
- Audience
- Related Documents
- Conventions Used in this Guide

1.1 Summary

Before you begin the customization, ensure that you have access to the Oracle Support Portal with valid login credentials to notify us of any issues at any stage quickly. You can obtain the login credentials by contacting Oracle Support. You can find the latest copy of this document in the Oracle Help Center Documentation Library.

1.2 Audience

This guide is intended for anyone who is customizing the RTF solution. Their roles and responsibilities, as they operate within OFS Real Time Fraud, include the following:

- **Professional Services Team:** A user in this role is responsible for the deployment and adjustments of Oracle Financial Services solution at the client's installation site. The Professional Services Team provides guidance and assistance in the identification and delivery of data.
- **Installer:** Installs and configures OFSBD at a specific deployment site. The Installer also installs and upgrades any additional Oracle Financial Services solution sets and requires access to deployment-specific configuration information, such as machine names and port numbers.

1.3 Related Documents

This section identifies additional documents related to the OFS Real Time Fraud component. You can access the following documents from Oracle Help Center Documentation Library:

- Oracle Financial Services Fraud Enterprise Edition (Real Time Fraud) Administration and Configuration Guide.
- Oracle Financial Services Fraud Enterprise Edition (Real Time Fraud) User Guide.

1.4 Conventions Used in this Guide

Table 1 lists the conventions used in this guide and their associated meanings.

Table 1: Conventions Used in this Guide

Convention	Meaning
Boldface	Boldface type indicates graphical user interface elements associated with an action (menu names, field names, options, button names) or terms defined in text or glossary.

Table 1: Conventions Used in this Guide

Convention	Meaning
Italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates the following: ● Directories and subdirectories ● File names and extensions ● Process names ● Code sample, that includes keywords, variables, and user-defined program elements within the text.
<variable>	Substitute input value

2 Adding and Displaying New Data on Details Page

This section provides information on adding a new section on details page of the Real Time Fraud Application.

Topics:

- Data Model Upload
- Database Changes
- HTML and JavaScript Changes
- Internationalization

2.1 Data Model Upload

1. Open the **BD_XXX.erwin** file for the installed patch version using the Erwin tool. Ensure the Erwin file belongs to the latest patch installed in the environment.
2. Add the additional fields as required to the following Wire and Card Transaction tables, respectively.

Table 1: Wire and Card Tables

Wire Transaction Tables	Card Transaction Tables
FCC_FR_WIRE_TRANSACTION	FCC_FR_CARD_TRANSACTIONS
FCC_FR_WIRE_ALERTED_TRXN	FCC_FR_CARD_ALERTED_TRXN

3. Add the **Logical** and **Physical** names for the additional fields.
   - **Logical Name:** The name you need to add to the JSON while sending the request.
   - **Physical Name:** It is the column name of the table.
4. Click **Save As** and update the **Repository** type to **XML** to generate the XML file from the Erwin tool.
5. Copy the **XML** file and paste it into the following path.

   `$FTP_SHARE_PATH/CONTEXT_NAME/erwin/erwinXML`

> **NOTE**
> - `$FTP_SHARE_PATH` is a placeholder for the FTP share path for the environment.
> - `CONTEXT_NAME` will vary based on the environment.

6. Login with the Wire or Card Admin and navigate to **Financial Services Fraud Enterprise Edition**. Under **Data Model Maintenance**, click **Data Model Maintenance**.
7. Click **Add**.
8. On the new screen, provide the name, change upload mode to **Sliced**, and upload file details to XML.
9. Select **XML with additional columns** from the **Erwin XML File Type** drop-down.
10. If logs are required, update **Generate DDL Execution Logs** to **Yes**.
11. Click **Upload Model** and check after a few minutes until the model upload succeeds.

12. Login with Wire or Card Admin user.
13. Navigate to **Financial Services Fraud Enterprise Edition**, under **Financial Services Inline Processing Engine**, select **Inline Processing**.
14. Navigate to **Associations and Configuration** tab and select **Business Entities.**
15. In the **Entity Name** drop box, select `FCC_FR_WIRE_TRANSACTION` or `FCC_FR_CARD_TRANSACTIONS` for wire or card respectively.
16. After verifying the information for new attributes in the popup window, click **Save**.
17. Restart WebLogic or Websphere server.

2.2 Database Changes

The following changes must be made in the Database to add a new section.

1. Create a new entity entry in the `FCC_FR_ENTITY_UI_SECTIONS` table. Add the values according to the descriptions provided in Table 2.

NOTE The value provided in `HEADER_NAME` must be added to the `MESSAGES_EN_US` table for internationalization.

Table 2: Column Names and Descriptions of FCC_FR_ENTITY_UI_SECTIONS table

Column Name	Description
`ENTITY_ID`	Unique ID for the entity.
`HEADER_NAME`	Section Heading keyword. It should be the same as `SECTION_NAME` in `FCC_FR_UI_SECTION_DETAILS` table for non-tabbed type and different otherwise.
`ENTITY_TYPE`	This is used to provide the entity type. Example: `RTF_CARD` for real-time fraud card, `RTF_WIRE` for real-time fraud wire.
`WIDTH`	This is used to describe how wide a section should be. 1 = 100%, 2 = 50%, 3 = 33%, and so on. Use values which are factors of 12.
`ENTITY_ORDER`	Order of the section in the page. 1 based indexing per `ENTITY_TYPE`.

Table 3: Column Names and Descriptions of FCC_FR_UI_SECTION_DETAILS table

Column Name	Description
`SECTION_ID`	Unique ID for the section. Since section can be tabbed, this table is used to map multiple tabs to single section or just one section to one detail.
`SECTION_TYPE`	Type name to be used in HTML template.
`SECTION_NAME`	Should be same as Entity Name for non-tabbed or should be different always for tabbed types.

Table 3: Column Names and Descriptions of FCC_FR_UI_SECTION_DETAILS table

Column Name	Description
`SECTION_ORDER`	Give 1 for non-tabbed or 1 based indexing for tabbed types.
`QUERY`	SQL query for displaying data. Query must be written within a specific template as provided.

2. While adding new sections, ensure that the entity order for the `ENTITY_TYPE` provided will determine where the section will get added on the page. Only the last row can have partial content (i.e., summation of the inverse of widths should equal 1 for all rows, except the last if required).

   For example:
   If you want a page with 1st row with one section, 2nd row with two sections of 50% width each, and 3rd row with three sections of 33% each, and 4th row (last) with any value of width.

   - **Row 1**: Order 1, Width 1 ($^1/_1$= 1)
   - **Row 2**: Order 2, Width 2 and Order 3, Width 2 (½ + ½ = 1)
   - **Row 3**: Order 4, Width 3 and Order 5, Width 3 and Order 6, Width 3 (⅓ + ⅓ + ⅓ = 1)
   - **Row 4**: Order 7, Width 2 (Only the last row can be partial or full)

3. Add entries in `FCC_FR_UI_SECTION_DETAILS` of this newly created entity entry details. `ENTITY_ID` is the foreign key that needs to match the newly created entity. `SECTION_TYPE` is the name that will match against values present in the HTML file.

> **NOTE** The value provided in `HEADER_NAME` must be added to the `MESSAGES_EN_US` table for internationalization.

4. Enter the `SECTION_NAME` based on the following scenarios.
   - If you want the section heading not to be tabbed and section names are the same then `SECTION_NAME` in Table 3 should be the same as the `HEADER_NAME` in Table 2.
   - If you want the section heading not to be tabbed, they must have different names.
   - If you want the same value in the tabbed heading and section heading, you can do that by adding two separate entries in `MESSAGES_EN_US`.

5. The `QUERY` is the column in which we add the `SELECT` query to display the fields in a section. The query can be of 2 types. You can select a single row of data (can have joined, but the result is a single row), or multiple rows. Choose the query template based on your requirement.
   - **Single Row Select Query Template**

```
WITH CHANGE_ME_QUERY AS (
    SELECT
        COL1.TABLE_NAME name1,
        COL2.TABLE_NAME name2,
        COL3.TABLE_NAME name3_HL,
        FROM TABLE_NAME
        WHERE TABLE_NAME.N_ALERT_ID = ?
```

```
)
SELECT
    COALESCE (
        JSON_OBJECT (
            'CHANGE_ME' : (
                SELECT JSON_ARRAYAGG (jo returning clob)
                FROM (
                    SELECT JSON_OBJECT (* returning clob) jo
                    FROM (
                        SELECT *
                        FROM CHANGE_ME_QUERY
                    )
                )
            ) returning clob
        ), TO_CLOB('[]')
    ) JSON_OP
FROM DUAL
```

- **Multiple Row Select Query Template**

```
WITH CHANGE_ME_QUERY AS (
    SELECT
        COL_NAME name1,
        COL_NAME name2_HL,
    FROM TABLE_NAME
    INNER JOIN OTHER_TABLE_NAME ON TABLE_NAME.col3 =
OTHER_TABLE_NAME.col2
    WHERE SOME_COLUMN = ?
) SELECT
    COALESCE (
        JSON_OBJECT (
            'CHANGE_ME': (
                SELECT JSON_OBJECT(* returning clob)
                FROM (
                    SELECT *
                    FROM CHANGE_ME_QUERY
                )
            )
```

```
                returning clob
            ), TO_CLOB('[]')
        ) JSON_OP
    FROM DUAL
```

6. Once you choose the required template, you must make the folloiwng updates as required.
   - Replace `CHANGE_ME` with any name of your choice.
   - Change the `SELECT` part of the template as per requirement.
   - Order of data received on front-end is same as the order in which columns are present in `SELECT`.
   - All columns in `SELECT` query should have an alias name, and they must be added to `MESSAGES_EN_US` for internationalization.
   - To highlight the field, add `_HL` at the end of the column name.
   - To have an empty space “placeholder” alias_name can be added in `SELECT`.
   - If `WHERE` condition requires value from the front-end client code, add a question mark (?) as placeholder (and needs to be specified in JavaScript file. For more information, see).
   - Query of any complexity can be written with the templates, any number of columns, table joins, conditions, aggregations can be used.

2.3 HTML and JavaScript Changes

2.3.1 HTML

Make the following changes to the HTML file as required.

1. Create a new section under the existing script as provided in the example template to match the `SECTION_TYPE` value provided in `FCC_FR_UI_SECTION_DETAILS` table.

   HTML file Path: <Deployed_Area>/applications/BD.ear/BD.war/RealTimeFraud/js/view/Fraud/alertDetails.html

   Existing Script:

```
<oj-bind-for-each data="[[alertDetailsData]]">
    <template data-oj-as="row">
        <div class="oj-flex">
            <oj-bind-for-each data="[[row.data]]">
                <template data-oj-as="section">
                    <div :class="[[sectionClass(section.data.width)]]">
                        <div class="oj-panel oj-panel-shadow-md oj-sm-
margin-1x" style="max-height: 450px;">
```

The template provided supports both placeholder (empty space can be used for ordering columns in specific way) and highlighted fields.

Make the following updates to the example template as required.

- `#SECTION_TYPE` should be changed to the value added.
- `#CHANGE_ME` is supposed to be changed with number of columns in a row.
- Any kind of HTML or OJET element can be added in this template, the default template is only for reference to create new templates and the how to use the data inside the templates.

HTML Example Template

```
<oj-bind-if test="[[section.data.type == '#SECTION_TYPE']]">
    <!-- heading -->
    <div class="oj-sm-12"
        style="font-size: 20px; padding-bottom: 10px; border-bottom: 2px
solid #c5bfbf; margin-bottom: 10px">
        <span><oj-bind-text
value="[[RTFraudMessageConstants[section.data.name]]]"></oj-bind-text></
span>
    </div>


    <!-- body -->
    <oj-bind-for-each
        data="[[nPairwise(section.data.details, #CHANGE_ME)]]">
        <template data-oj-as="fields">
            <div class="oj-sm-12 oj-flex-bar">
                <oj-bind-for-each data="[[fields.data]]">
                    <template data-oj-as="curField">
                        <div :class="[[sectionClass(#CHANGE_ME)]]"
style="border-right: 2px solid #c5bfbf; padding: 10px;">
                            <oj-bind-if test="[[curField.data.value !=
'placeholder']]">
                                <oj-label-value>
                                    <oj-label slot="label">
                                        <span>
                                            <oj-bind-text

value="[[RTFraudMessageConstants[curField.data.key]]]">
                                            </oj-bind-text>
                                        </span>
                                    </oj-label>
                                    <span slot="value"
:style="[[isHighlighted(curField.data.key)]]">
                                        <oj-bind-text
                                        value="[[curField.data.value]]">
```

```
                                        </oj-bind-text>
                                    </span>
                                </oj-label-value>
                            </oj-bind-if>
                        </div>
                    </template>
                </oj-bind-for-each>
            </div>
        </template>
    </oj-bind-for-each>
</oj-bind-if>
```

2.3.2 JavaScript

Make the following changes to the JavaScript files as required.

1. Add the addition filter and the parameters required in the SQL query to the following JavaScript files under the existing script.

   JavaScript File Paths:

   - `<Deployed_Area>/applications/BD.ear/BD.war/RealTimeFraud/`js/viewModels/Fraud/alertMainTable.js
   - `<Deployed_Area>/applications/BD.ear/BD.war/RealTimeFraud/`js/viewModels/Fraud/alertDetails.js

NOTE	
	• The files support single filter as well as nested multiple filters. • Provide `SECTION_NAME` as the key and value as required for that key.

   Existing Script:

```
ko.postbox.publish('callAjax', {
    'paramData': {
        'actionType': 'GET_ALERT_DETAILS',
            'transactionType': getTransType.toUpperCase(),
            'filters': {
                'RTF_ALERT_SUMMARY': {
                        ALERT_ID: self.alertid()
                    },
                    'RTF_ALERT_DECISION': {
                        lang: 'en_US',
                        status: "'R', 'B'"
```

```
                        },
                    'RTF_SCORE_CARD': {
                        ALERT_ID: self.alertid()
                    },
                    'RTF_CUSTOMER_DETAILS': {
                        ALERT_ID: self.alertid()
                    },
                    'RTF_TRANSACTION_RISK_INFO': {
                        'RTF_SESSION_DETAILS': {
                            ALERT_ID: self.alertid()
                        },
                        'RTF_MERCHANT_DETAILS': {
                            ALERT_ID: self.alertid()
                        }
                    },
                    'RTF_ACCOUNT_DETAILS': {
                        ALERT_ID: self.alertid()
                    },
                    'RTF_CARDS_DETAILS': {
                        ALERT_ID: self.alertid()
                    }
                }
            },
        'publishID': 'set_cardAlertDetails',
        'url': ''
})
```

2.4 Internationalization

To add an alias name of `RTF_MERCHNT_NAME`, follow these steps.

Database:

- Replace `RTF_MERCHNT_NAME` with alias name provided in `SELECT` clause of the `FCC_FR_UI_SECTION_DETAILS` table and execute the following command in CONFIG schema.

  ```
  INSERT INTO MESSAGES_EN_US (
      MSG_PACKAGE,
      MSG_IDENTIFIER,
      MSG_CODE,
  ```

```
        MSG_DESCRIPTION,
        MSG_TYPE,
        MSG_APPL_NAME,
        MSG_MODULE_NAME
    )
    VALUES (
        'RENDERER',
        'RTF_MERCHNT_NAME',
        '555',
        'Merchant Name',
        'L',
        '',
        ''
    )
```

JSP:

- Replace `RTF_MERCHANT_NAME` with alias names and add the following command in `RTFraudMessageConstatns.jsp` file.

  ```
  RTFraudMessageConstants.RTF_MERCHNT_NAME =
  "<%=MessageFramework.getMessageFromLocaleSpeficCache("RENDERER.RTF_MERCH
  NT_NAME", currentMsgLocale.toString())%>";
  ```

  File path: `Deployed Area/Context_Name.ear/Context_Name.war/RealTimeFraud / RTFraudMessageConstants.jsp`

3 Modifying Existing Sections

This section provides information on modifying the existing sections.

Topics:

- Data Model Changes
- Database Changes
- HTML or JavaScript Changes

3.1 Data Model Changes

Refer to Data Model Upload section of this guide for the data model changes.

3.2 Database Changes

To modify the existing section, follow these steps as required.

1. Find the section detail entry in the `FCC_FR_UI_SECTION_DETAILS` table, update the QUERY column, and make the required changes in the `SELECT` part inside the `WITH` clause.
2. All columns in `SELECT` query should have an alias name, and they must be added to `MESSAGES_EN_US` for internationalization.
3. To highlight the field, add `_HL` at the end of the column name.
4. To have an empty space “placeholder” unique_alias_name can be added in `SELECT`.

To understand the flow better, refer to Adding and Displaying New Data on Details Page.

3.3 HTML or JavaScript Changes

3.3.1 HTML

If the modifications are for the Summary Section of the Real Time Fraud Application page, No HTML changes are required.

3.3.2 JavaScript

> **NOTE** Changes to the Summary and Decision sections are not allowed.

If Database changes are done, new fields will be displayed on the page. Add the new entries added in MESSAGES_EN_US table to the JSP file. Refer to Internationalization for more information.

4 Example Cases

This section provides examples for Wire and Card Cases.

Topics:

- Wire Example Case
- Card Example Case

4.1 Wire Example Case

Client would like to produce alerts at instances where the country of the counterparty does not match the country of their financial institution.

4.1.1 Scenario

`V_COUNTERPARTY_FI_COUNTRY_CD` does not match `V_COUNTERPARTY_COUNTRY_CODE`

4.1.2 Steps For Customization

For customizing the UI to achieve the above scenario, follow these steps.

1. Add the following fields to `FCC_FR_WIRE_TRANSACTION`, `FCC_FR_WIRE_ALERTED_TRXN` data model tables in the Erwin software.

Table 1: Additional Fields and Descriptions

Business Name	M/O	FCC_FR_TRASACTIONS	Data Type	Description
Counterparty Financial Institution Name	O	V_COUNTERPARTY_FI_NM	CHAR(70)	Name of the financial Institute
Counterparty Financial Institution Identifier	O	V_COUNTERPARTY_FI_ID	CHAR(34)	Identifier for the Financial Institute
Counterparty Financial Institution Country Code	O	V_COUNTERPARTY_FI_COUNTRY_CD	CHAR(2)	Country Code for the Financial Institute

2. Upload the Data Model to add the new fields to the application. Refer to Data Model Upload section.
3. After adding the new fields, configure the new scenarios and add additional rules for alert generation. For more information, see Oracle Financial Services Inline processing Engine User Guide.
4. The highlights can be used to add additional information about the new fields. However, if you want the fields to display on the details page separately, follow these steps.
   a. Based on the requirement of adding a new section or modifying an existing, perform the following steps.
      - **Adding a New Section:** Refer to Adding and Displaying New Data on Details Page.
      - **Modifying an Existing Section:** Edit the QUERY column in FCC_FR_UI_SECTION_DETAILS and add the entries in MESSAGES_EN_US. The new fields will be displayed. If a new query parameter is added, update the JavaScript files to reflect the same.

4.2 Card Example Case

Client has recently introduced multi-factor authentication for credit card purchases. As part of this authentication, a unique 4-digit PIN is sent to the customer's phone, who is then prompted to enter the PIN when a transaction is made. ABC Bank allows customers self-select a dollar threshold at which MFA is required.

The client is requesting the following.

- A scenario that detects multiple failed MFA attempts.
- Display the 4-digit PIN that was sent to the customer's phone in the UI.
- Display the 4-digit PIN the customer entered to validate the purchase in the UI.
- Display the dollar threshold the customer self-selected that prompted the MFA.

4.2.1 Scenario

Wrong OTP/PIN entry for more than specified times: If the OTP/PIN is entered wrongly for (N)(times), then alert for a possible fraud is to be triggered.

(Count(*) Where N_CARD_NUMBER= 'XYZ' and N_PIN_VERIFICATION_RESULT=4) > N Correct

4.2.2 Steps for Customization

For customizing the UI to achieve the above scenario, follow these steps.

1. Add the below fields in the data model in `FCC_FR_CARD_TRANSACTIONS`, `FCC_FR_CARD_ALERTED_TRXN` tables in the Erwin software.

Table 2: Additional Fields and Descriptions

Business Name	M/O	FCC_FR_TRASACTIONS	Data Type	Description
OTP Sent PIN	O	N_OTP_SENT	Char(4)	OTP four digit PIN received by the customer.
OTP Entered PIN	O	N_OTP_ENTERED	Char(4)	OTP four digit PIN entered by the customer.

2. Upload the Data Model to add the new fields to the application. Refer to Data Model Upload section.
3. After adding the new fields, configure the new scenarios and add additional rules for alert generation. For more information, see Oracle Financial Services Inline processing Engine User Guide.
4. The highlights can be used to add additional information about the new fields. However, if you want the fields to display on the details page separately, follow these steps.
   a. Based on the requirement of adding a new section or modifying an existing, perform the following steps.
   - **Adding a New Section:** Refer to Adding and Displaying New Data on Details Page.

- **Modifying an Existing Section:** Edit the QUERY column in FCC_FR_UI_SECTION_DETAILS and add the entries in MESSAGES_EN_US. The new fields will be displayed. If a new query parameter is added, update the JavaScript files to reflect the same.

OFSAA Support Contact Details

Raise a Service Request (SR) in My Oracle Support (MOS) for queries related to OFSAA applications.

Send Us Your Comments

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, indicate the title and part number of the documentation along with the chapter/section/page number (if available) and contact the Oracle Support.

Before sending us your comments, you might like to ensure that you have the latest version of the document wherein any of your concerns have already been addressed. You can access My Oracle Support site which has all the revised/recently released documents.