Oracle Address, Email, and Phone Verification

Using Oracle Address, Email, and Phone Verification

22A

Oracle Address, Email, and Phone Verification
Using Oracle Address, Email, and Phone Verification

22A

E59772-31

Authors: Michele Cyran, Tom Enderes, Bikram Gill, Keerthy Jayaraj, Ashwini Malthankar, Harisha Shankaramurthy, Alicia Wu

Contents

Get Help

There are a number of ways to learn more about your product and interact with Oracle and other users.

Get Help in the Applications

Use help icons Help icon to access help in the application. If you don't see any help icons on your page, click your user image or name in the global header and select Show Help Icons.

Get Support

You can get support at *My Oracle Support*. For accessible support, visit *Oracle Accessiblity Learning and Support*.

Get Training

Increase your knowledge of Oracle Cloud by taking courses at *Oracle University*.

Join Our Community

Use *Cloud Customer Connect* to get information from industry experts at Oracle and in the partner community. You can join forums to connect with other customers, post questions, and watch events.

Learn About Accessibility

For information about Oracle's commitment to accessibility, visit the *Oracle Accessibility Program*. (if videos) Videos included in this guide are provided as a media alternative for text-based topics also available in this guide.

Share Your Feedback

We welcome your feedback about Oracle Applications user assistance. If you need clarification, find an error, or just want to tell us what you found helpful, we'd like to hear from you.

You can email your feedback to *oracle_fusion_applications_help_ww_grp@oracle.com*.

Thanks for helping us improve our user assistance!

1 Get Started with Address, Email, and Phone Verification

Get Started with Address, Email, and Phone Verification

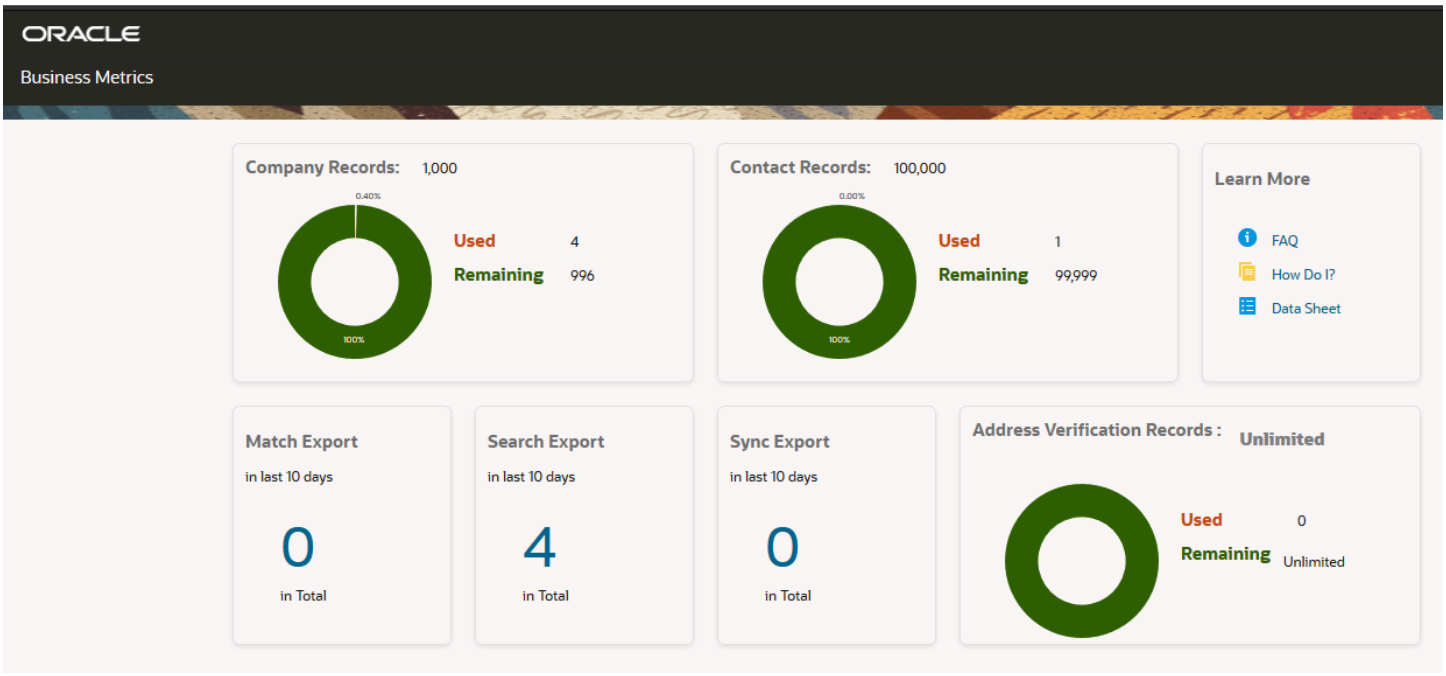
Oracle Address, Email, and Phone Verification lets you verify and standardize data in your applications. For example, you can verify the postal addresses, email addresses, and phone numbers of your prospects and customers to efficiently manage your communication at each stage of the sales or marketing process.

Access the REST API in the following format: `https://serviceName-identityDomain.data.us2.oraclecloud.com/av/api`

For example: `https://data1234-usoracle1234.data.us2.oraclecloud.com/av/api/v3/address/find`

Verification services can be used with Oracle Account and Contact Enrichment (also known as Data as a Service, or DaaS) or as a standalone service with other applications. For the list of supported countries, see *Loqate Data Coverage*.

There is no user interface, but you can see your address record usage on the DaaS landing page. (This URL is listed on Oracle Cloud Console.) The **Address Verification Records** section shows the number of address records remaining in your subscription (or it shows "Unlimited" if you purchased unlimited records). If you haven't purchased a subscription for DaaS, then this landing page shows only the **Address Verification Records** and the **Learn More** sections.



How to Begin with Address, Email, and Phone Verification

Here's a summary of the key steps to get new Address, Email, and Phone Verification subscriptions provisioned in an Oracle Cloud Infrastructure (OCI) data center.

1. Order a subscription for Address, Email, and Phone Verification by contacting Oracle Sales. Your subscription will be made in the same data center region as your Oracle Cloud account.

   For details about ordering subscriptions, see *Order Oracle Cloud Applications* in *Getting Started with Oracle Cloud Applications*.

2. Add subscription to your Oracle Cloud account.

   You'll receive an email titled “Action Required: Welcome to Oracle Cloud. Please Add Your Subscription(s) to a Cloud Account to Get Started.” Follow the instructions in the email:

   - If you have an Oracle Cloud account, then click the **Activate into Existing Cloud Account**. We recommend using your existing Oracle Cloud account to manage all your Oracle services.
   - If you need a new Oracle Cloud account, then click the **Create New Cloud Account** button.

3. Activate your service.

   You'll receive a second email titled “Action Required: Please Activate Your Services.” In the email, under Access Details, click the **Create Instance** button.

   Sign in with the temporary password listed in the email. You're prompted to change this password. (Temporary passwords expire after 60 days. Contact Oracle Support if you need to request a new password.)

4. Create the Data environment.

   a. On the Oracle Cloud Dashboard, click **Create Instance**.
   b. You'll see a list of services that can be provisioned. Click **Create** for the **Data** service. Data is the cloud service instance that hosts Address, Email, and Phone Verification, as well as Account and Contact Enrichment (also known as Data as a Service).
   c. Enter a name for this instance, confirm your administrator, then click **Create**.
   d. Click **Create** again, to confirm the instance details.

5. You (or the person you designated as the administrator) will receive an email titled “Action Required: Your new Data as a Service instance in Cloud Account is ready.”

   In the email, click **Access Your New Service**.

   On the Oracle Cloud Console, click the **Data** application to see the Address, Email, and Phone Verification API URL.

6. Create accounts for your users and assign them appropriate roles.

   The following roles are created during provisioning.

   - DATASERVICE_ADMINISTRATOR (administrator access)
   - DATASERVICE_USER (user access)
   - DATASERVICE_CLIENT_API_APPID (necessary for integrating with other applications)

   To add users, see *Add Users to a Cloud Account with Identity Cloud Service* in *Getting Started with Oracle Cloud Applications*.

Note: To associate another Oracle Cloud service with both Address, Email, and Phone Verification *and* Account and Contact Enrichment, they must be part of the same service instance; that is, both services must be part of the same subscription order. For example, if you order Account Enrichment first and later add Address Verification, then you must upgrade your Account Enrichment subscription to include Address Verification, or vice-verse. Account and Contact Enrichment *and* Address, Email, and Phone Verification cannot be on separate service instances when associating them with the same instance of another Oracle Cloud service.

How to Update Your Subscription

When you need to expand or renew your subscription, you modify it in the Oracle Cloud Console.

1. Expand or renew your subscription by contacting Oracle Sales.

   You'll get an activation email titled: *Your service has been updated.*
2. On the Cloud Console Dashboard, select your active Data service.
3. On the Overview page, under Service Environments, click the Action Menu to see the actions dropdown list.
4. Select the **Modify** action.
5. On the Modify page, leave all default values, and click the **Modify** button.
6. In the dialog, click **Modify** again to confirm that you want to update this instance.

   You'll get an activation email titled: *Your service instance has been updated.*

Integrate with Oracle Sales or Procurement

You can enable Oracle Sales to integrate with Address, Email, and Phone Verification for real-time verification and autocomplete address functionality on data entry. You can also enable Oracle Procurement to integrate with Address Verification for autocomplete address functionality.

This assumes you have an active Sales or Procurement subscription, and then add a subscription for Oracle Address, Email, and Phone Verification.

1. In CX Sales or Procurement, configure the **Manage Integration with Oracle Verification Services** task.

   In the Setup and Maintenance work area, go to the following:

   - Offering: Sales
   - Functional Area: Integrations
   - Task: Manage Integration with Oracle Verification Services

   > **Note:** The user doing the integration must have the DATASERVICE_CLIENT_API_APPID role. See *How to Begin with Address, Email, and Phone Verification*.

2. Select **OAuth Authentication**.

   For information about adding a confidential application and getting the Client ID and Secret, see *Add a Confidential Application* in *Administering Oracle Identity Cloud Service*.

   > **Note:** On the Client page of the Add Confidential Application wizard, in the Authorization section, select **Resource Owner**, **Client Credentials**, **Authorization Code**, and **Refresh Token** as Allowed Grant Types. Leave **Redirect URL** blank.

   > **Note:** If you use Basic Authentication, do not include the question mark special character (?) in the password.

3. For **URL**, remove `/data/ui` from the end of the instance address listed in your Welcome email and in the Cloud Console.

   For example, `https://mydataservice-myidentitydomain.data.us2.oraclecloud.com`.

About Address Verification

Oracle Address Verification can correct misspelled city or street names, complete postal codes, and standardize abbreviations like Pkwy. The service takes your address data and returns matches—one match or multiple matches—and you can select which version you want to use as the standard.

Address verification can be used in real time to verify mailing addresses or to provide type-ahead smart data when entering addresses. It can also be used in batch mode to verify large numbers of records. The following videos provide a summary of address verification functionality.

- Verify Addresses in Real Time: Click the **Verify Address** button when creating or editing an account to make sure you have complete and standardized addresses.

  Watch video

- Verify Addresses using Type-Ahead Smart Data: Enable address suggestions to automatically appear as you start entering an address: the more you enter, the more the results narrow down to the closest match.

  Watch video

- Run Batch Address Cleansing: You can verify all addresses in your application that have been updated or imported since your last batch cleansing job. With Simulated mode, you can review each verified address before accepting and importing it into Customer Data Management.

  Watch video

Use Address Verification REST API

Address verification lets you search and verify global addresses in your applications.

Watch this video for an overview of the REST API features.

Watch video

Topics

- *Search Addresses*
- *Verify Addresses*

Search Addresses

Address Verification contains functionality for type-ahead address search and address completion. We provide this functionality in three endpoints, available in version 2 (v2) and version 3 (v3). New users should use v3.

You can use the Search API in either Indirect Mode or Direct Mode.

Indirect Mode

REST endpoints:

```
/api/v3/address/find
```

```
/api/v3/address/retrieve
```

With indirect mode, the client doesn't call the third-party provider directly – the client only interacts with Oracle Address Verification Service. Oracle then forwards the `find` call to the third-party provider.

First, `find` is called to narrow down the search, and then `retrieve` is called to obtain the complete address record. Because the performance for `find` is slower in this mode, we recommend using Direct Mode whenever possible.

Direct Mode

REST endpoints:

```
/api/v3/address/preparefind
```

```
/api/v3/address/retrieve
```

Here the client first calls `preparefind` at application page initialization time. As a response to `preparefind`, the client receives information and credentials to perform `find` calls directly on the third-party application. `Retrieve` is then again performed against the verification endpoint, not the third-party application (see use cases in subsequent sections).

Simple Search Use Case

Suppose the first `find` request is issued as soon as you type the first three letters of an intended US address as “123”. This is the result request and response:

> **Note:** Examples show Indirect Mode. Functionality is identical for Direct Mode, but with an initial `preparefind`.

Request

```
http://adc00qrw.oracle.com:7767/av/api/v3/address/find?Text=123
```

Response


```
[
 {
 "Id": "US|US|ENG|94107-CA-SAN_FRANCISCO--ST-TOWNSEND--123",
 "Type": "BuildingNumber",
 "Text": "123 Townsend St",
 "Highlight": "0-3",
 "Description": "San Francisco CA 94107 - 320 Addresses"
 },
 {
 "Id": "US|US|ENG|94105-CA-SAN_FRANCISCO--ST-MISSION--123",
 "Type": "BuildingNumber",
 "Text": "123 Mission St",
 "Highlight": "0-3",
 "Description": "San Francisco CA 94105 - 48 Addresses"
 },
 {
 "Id": "US|US|A|Z222445177|123",
 "Type": "Address",
```

```
"Text": "123 2nd St",
"Highlight": "0-3",
"Description": "San Francisco CA 94105"
},
{
"Id": "US|US|A|Z222445178|1",
"Type": "Address",
"Text": "123 2nd St Ste 1",
"Highlight": "0-3",
"Description": "San Francisco CA 94105"
},
{
"Id": "US|US|A|Z223933823|123",
"Type": "Address",
"Text": "123 Mission St",
"Highlight": "0-3",
"Description": "San Francisco CA 94105"
},
{
"Id": "US|US|ENG|94103-CA-SAN_FRANCISCO--ST-GUERRERO--123",
"Type": "BuildingNumber",
"Text": "123 Guerrero St Apt",
"Highlight": "0-3",
"Description": "San Francisco CA 94103 - 3 Addresses"
},
{
"Id": "US|US|A|Z222437371|123",
"Type": "Address",
"Text": "123 Guerrero St",
"Highlight": "0-3",
"Description": "San Francisco CA 94103"
},
{
"Id": "US|US|A|Z222457206|123",
"Type": "Address",
"Text": "123 Waverly Pl",
"Highlight": "0-3",
"Description": "San Francisco CA 94108"
},
{
"Id": "US|US|A|Z222454357|123",
"Type": "Address",
"Text": "123 Joice St",
"Highlight": "0-3",
"Description": "San Francisco CA 94108"
},
{
"Id": "US|US|ENG|94108-CA-SAN_FRANCISCO--ST-JOICE--123",
"Type": "BuildingNumber",
"Text": "123 Joice St Apt",
"Highlight": "0-3",
"Description": "San Francisco CA 94108 - 4 Addresses"
}
]
```


Mailing addresses can be found in the response of all JSON objects that have the `Type` field set to `Address`. In this simple use case, the user interface could just suppress all other types (such as BuildingNumber, Street, and so on) and display only address objects.

To find the address “123 Guerrero St”, you click it. This causes the client application to issue a request to the `/address/retrieve` endpoint, passing an ID parameter with the value `US|US|A|Z222437371|123`, which then retrieves the full address record.

Otherwise, you must continue typing the address, while the UI continually issues equivalent `find` requests until the address you want is found.

Retrieve Addresses

After an ID is obtained, the full and componentized information for this address can be obtained by calling the `/address/retrieve` endpoint with that `ID` as a parameter.

Doing so for the simple `find` use case produces the following results:

Request

```
http://adc00qrw.oracle.com:7767/av/api/v3/address/retrieve?Id=US|US|A|Z222437371|123
```

Response


```
[
 {
 "Id": "US|US|A|Z222437371|123",
 "Premise": "123",
 "Thoroughfare": "Guerrero St",
 "Locality": "San Francisco",
 "DeliveryAddress1": "123 Guerrero St",
 "SubAdministrativeArea": "San Francisco",
 "AdministrativeArea": "CA",
 "AdministrativeAreaName": "California",
 "PostalCode": "94103-1072",
 "CountryName": "United States",
 "ISO3166-2": "US",
 "ISO3166-3": "USA",
 "ISO3166-N": "840",
 "Address": "123 Guerrero St|SAN FRANCISCO CA 94103-1072",
 "DeliveryAddress": "123 Guerrero St",
 "Address1": "123 Guerrero St",
 "Address2": "SAN FRANCISCO CA 94103-1072",
 "PostalCodePrimary": "94103",
 "PostalCodeSecondary": "1072"
 }
]
```


Advanced Search Use Case

Suppose you want to search for the address “941 Dolores St, unit #3” in the USA, and you start typing the characters “941”. The corresponding REST transaction initiated by the UI is shown in Request 1 and Response 1.

> **Note:** Examples show Indirect Mode. Functionality is identical for Direct Mode, but with an initial `preparefind`.

In this case, the UI takes all types of records into account, not only `Type=Address`. Looking at the results from Response 1, you might not continue typing additional characters, but instead select the result “Dolores St Apt”, where the `Type` is `BuildingNumber`. The UI should understand that this is not a final selection of an address, but rather a drilldown into one of the results, which contains four more addresses. This is accomplished by using the result’s ID and passing it as a `Container` parameter into a new request, as shown in Request 2. In Response 2, you see that the previous result is not visible anymore, but all four apartments are shown.

After seeing the #3 unit, you get the record by calling the `/address/retrieve` endpoint using the ID value of `US|US|A|Z222467771|3`.

Request 1

```
http://adc00qrw.oracle.com:7767/av/api/v3/address/find?Text=941
```

Response 1


```
[
 {
 "Id": "US|US|A|Z222467773|941",
```

```
"Type": "Address",
"Text": "941 Dolores St",
"Highlight": "0-3",
"Description": "San Francisco CA 94110"
},
{
"Id": "US|US|ENG|94110-CA-SAN_FRANCISCO--ST-DOLORES--941",
"Type": "BuildingNumber",
"Text": "941 Dolores St Apt",
"Highlight": "0-3",
"Description": "San Francisco CA 94110 - 4 Addresses"
},
{
"Id": "US|US|ENG|94117-CA-SAN_FRANCISCO--ST-PAGE--941",
"Type": "BuildingNumber",
"Text": "941 Page St Apt",
"Highlight": "0-3",
"Description": "San Francisco CA 94117 - 5 Addresses"
},
{
"Id": "US|US|A|Z222490758|941",
"Type": "Address",
"Text": "941 Page St",
"Highlight": "0-3",
"Description": "San Francisco CA 94117"
},
{
"Id": "US|US|A|Z222412877|941",
"Type": "Address",
"Text": "941 Ridgeview Ct Unit",
"Highlight": "0-3",
"Description": "South San Francisco CA 94080"
},
{
"Id": "US|US|A|Z222412876|941",
"Type": "Address",
"Text": "941 Ridgeview Ct",
"Highlight": "0-3",
"Description": "South San Francisco CA 94080"
},
{
"Id": "US|US|A|Z213529633|941",
"Type": "Address",
"Text": "941 Shorepoint Ct",
"Highlight": "0-3",
"Description": "Alameda CA 94501"
},
{
"Id": "US|US|ENG|94501-CA-ALAMEDA--CT-SHOREPOINT--941",
"Type": "BuildingNumber",
"Text": "941 Shorepoint Ct Apt",
"Highlight": "0-3",
"Description": "Alameda CA 94501 - 146 Addresses"
},
{
"Id": "US|US|ENG|94608-CA-EMERYVILLE--ST-37TH--941",
"Type": "BuildingNumber",
"Text": "941 37th St Apt",
"Highlight": "0-3",
"Description": "Emeryville CA 94608 - 5 Addresses"
},
{
"Id": "US|US|A|Z224475658|941",
"Type": "Address",
"Text": "941 Aileen St",
"Highlight": "0-3",
```

```
 "Description": "Emeryville CA 94608"
 }
]
```


Request 2


```
http://adc00qrw.oracle.com:7767/av/api/v3/address/find?Text=941&Container=US|US|ENG|94110-CA-SAN_FRANCISCO--ST-DOLORES--941
```


Response 2


```
[[
 {
 "Id": "US|US|A|Z222467771|1",
 "Type": "Address",
 "Text": "941 Dolores St Apt 1",
 "Highlight": "",
 "Description": "San Francisco CA 94110"
 },
 {
 "Id": "US|US|A|Z222467771|2",
 "Type": "Address",
 "Text": "941 Dolores St Apt 2",
 "Highlight": "",
 "Description": "San Francisco CA 94110"
 },
 {
 "Id": "US|US|A|Z222467771|3",
 "Type": "Address",
 "Text": "941 Dolores St Apt 3",
 "Highlight": "",
 "Description": "San Francisco CA 94110"
 },
 {
 "Id": "US|US|A|Z222467771|4",
 "Type": "Address",
 "Text": "941 Dolores St Apt 4",
 "Highlight": "",
 "Description": "San Francisco CA 94110"
 },
 {
 "Id": "US|US|A|Z222467772|A",
 "Type": "Address",
 "Text": "941 Dolores St Apt A",
 "Highlight": "",
 "Description": "San Francisco CA 94110"
 }
]
```


For detailed information, see *REST API for Oracle Address, Email, and Phone Verification.*

Verify Addresses

Use Address Verify to do the following:

- Correct and enhance an address. For example, you can correct misspelled city or street names; add missing elements, like full postal code or state; and get geographic location, such as latitude, longitude.
- Get accurate representation of address from postal files. For example, change “street” to “St” or “parkw” to “Pkwy”.

- Transliterate an address using `outputscript` (v2) or `OutputScript` (v3). The following values are supported:
  - `Native`: This returns output in the script most commonly used in that country (see below).
  - `Latn`: This returns output in the Latin alphabet (English).
  - no value (default): This tries to match the address output script to whatever is provided as the address input script.

  For example, input Japan addresses that are in Latin script and get the verified output in Hani script. This is supported for addresses for certain countries.

  Here are the native scripts (in ISO and full name) with the country where this is known to be supported:

  - Latn – Latin (English transliteration wherever possible)
  - Cyrl – Cyrillic (Russia)
  - Grek – Greek (Greece)
  - Hebr – Hebrew (Israel)
  - Hani – Kanji (Japan)
  - Hans – Simplified Chinese (China)
  - Arab – Arabic (United Arab Emirates)
  - Thai – Thai (Thailand)
  - Hang – Hangul (South Korea)
  - Native – Output in the native script wherever possible

  Transliteration is bi-directional and generally happens from Native to Latn and Latn to Native using data for countries stated above next to the scripts. However, Latn to Hans is not available for any country other than China.

  Loqate does not support processing nor transliteration between Latin and native scripts outside of these stated countries. For example, if Greek (Grek) is used on a Portugal address, Loqate cannot parse or validate the record, nor transliterate it into Latin script. Similarly, a Portugal address entered in Latin cannot transliterate to any other script.

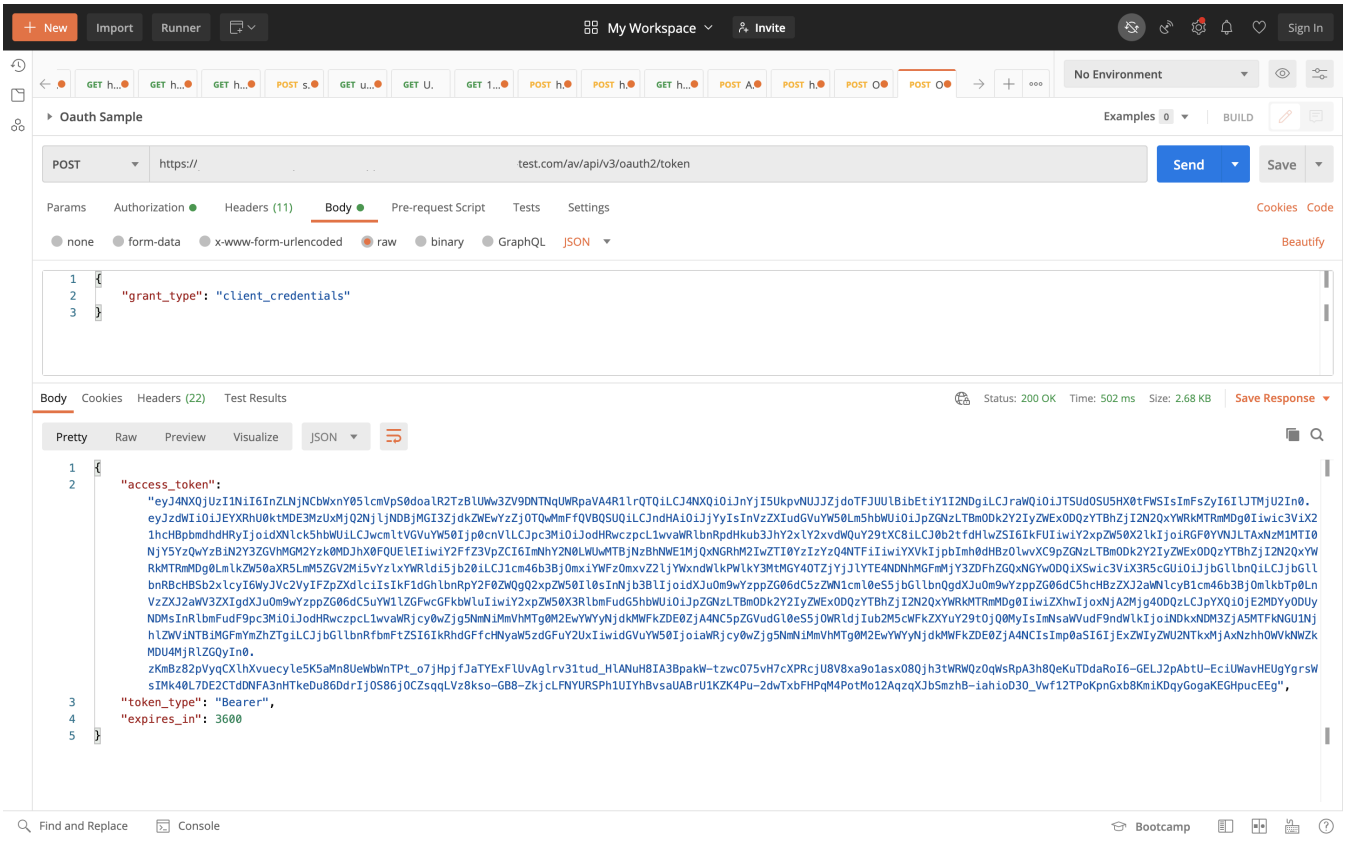
  For information on Loqate's transliteration support, see *https://support.loqate.com/character-scripts/*.

> **Note:** Do not set `minimumverificationmatchscore` to 100 unless you know you have a valid address and you just want to standardize it. In general, use a country's default `minimumverificationlevel` (for example, US=4). The exception is if you need to verify a zip code.

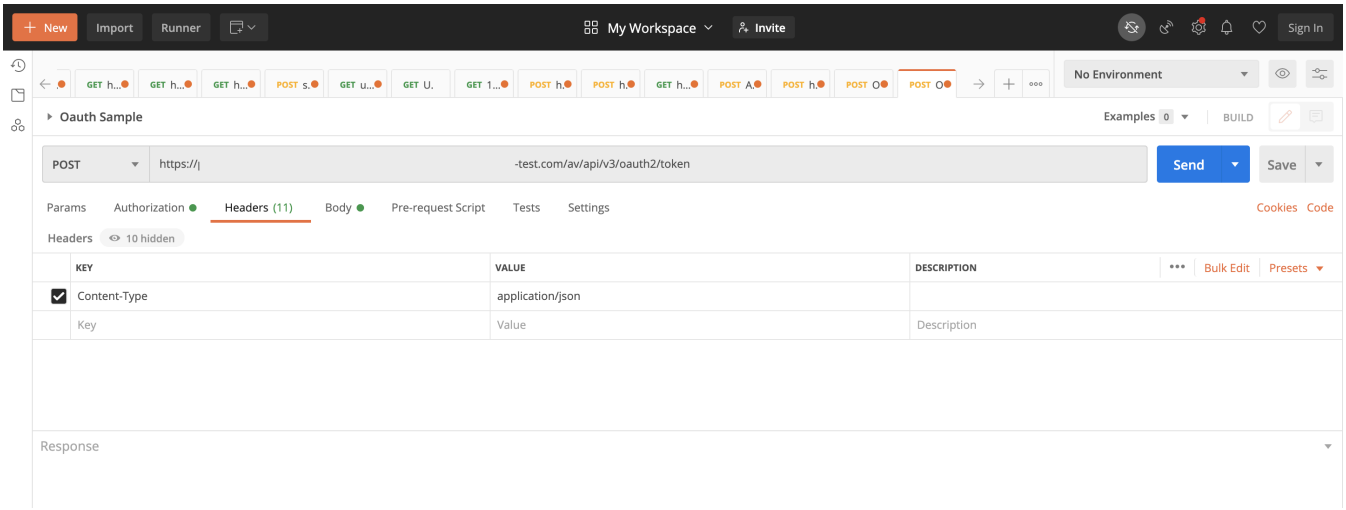
The following example shows the steps to verify an address. The process is the same for a single address verification request and for multiple addresses.

1. First, set up authorization. The API uses a standard OAuth 2.0 flow to authorize and authenticate API clients and users. If you're using the API, you should already have an OAuth client ID and client secret. You

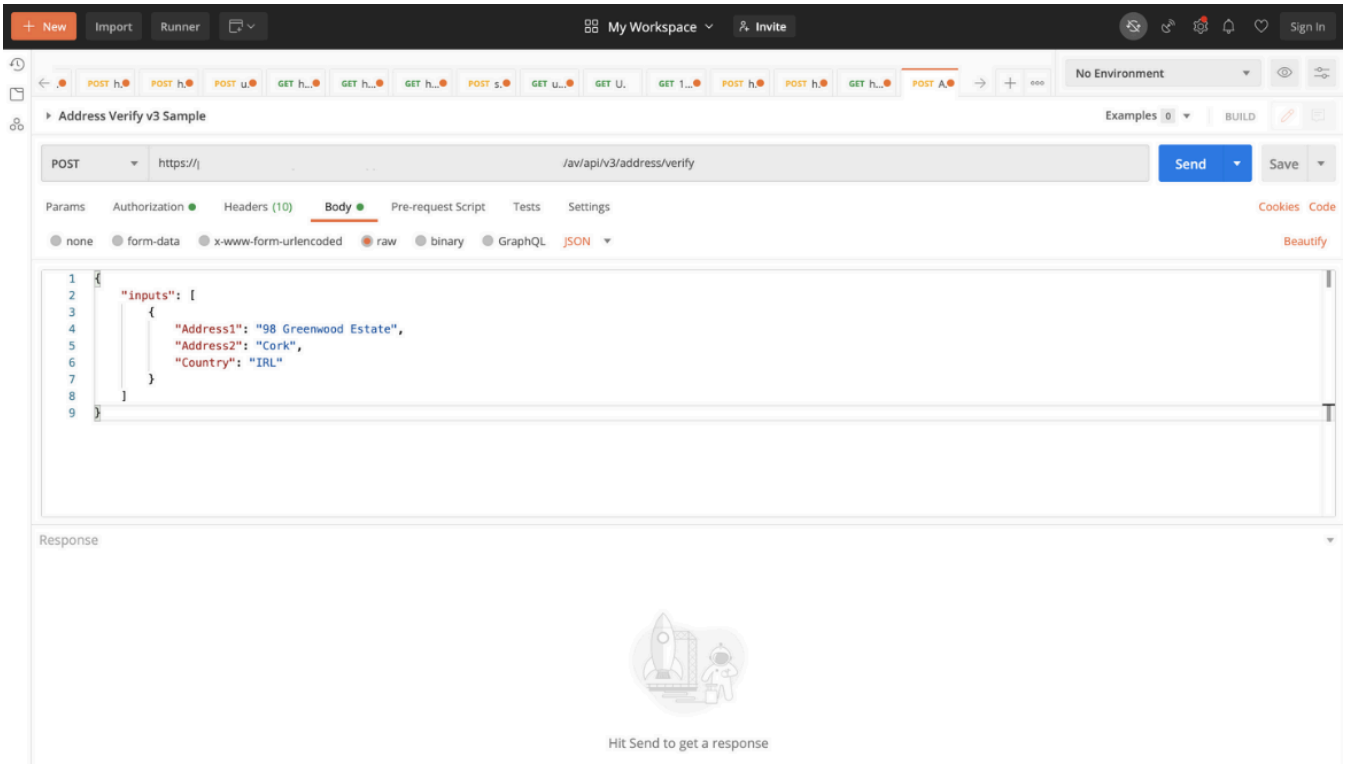
can request an access token with the following POST method: **`http://host:port/av/api/v3/oauth2/token`**.



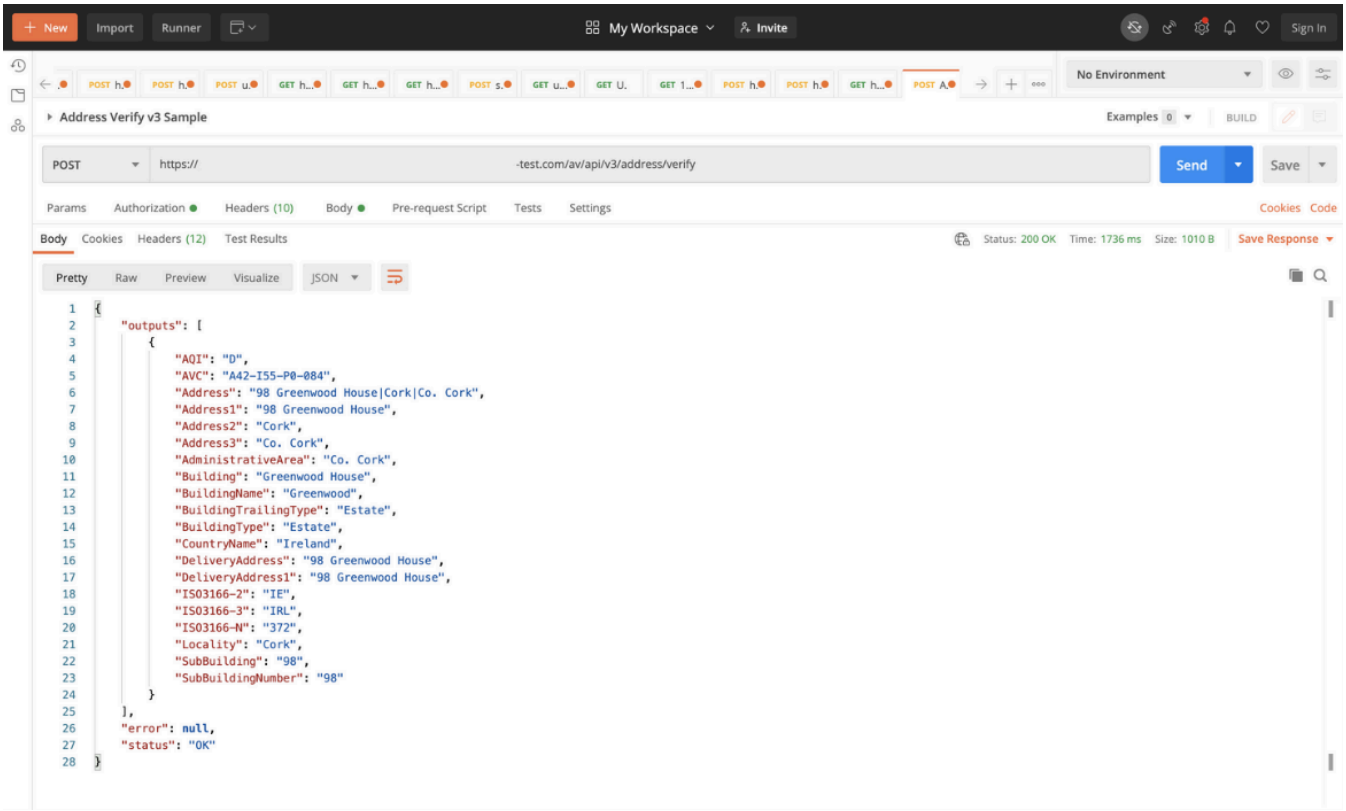
Then enter the credentials (key value) you received.



2. Enter the Address Verify URL as `http://host:port/av/api/v3/address/verify`, and then enter the address you want verified as input in the body. This example provides only Address1, Address2, and Country.



3. Click **Send**, and the complete verified, standardized address is returned as the response.



Use Email Verification REST API

Email verification lets you verify global email addresses in your applications. You can do individual or batch verifications.

This takes as input the email address to verify. The response signifies the following:

- Valid: The address was successfully verified.
- Partially-Valid: The email domain is valid, but the email account could not be verified.
- Invalid: The address was not successfully verified.
- The request timed out before it could be completed.

Invalid and partially-valid verifications show the date of the most recent verification. To show the valid status on valid emails right away, change the verification threshold date. For example, if you're working in Oracle CX Sales and the threshold is set to 180, then 180 days after the last valid verification was run, the UI renders with the email address the **Verification Status**, **Verification Date**, a **Re-verify** icon, and an **Overwrite** icon on the Contact and Account Profile pages. To show the valid status with these icons right away, change the verification threshold to 0.

For detailed information, see Email Verify in *REST API for Oracle Address, Email, and Phone Verification.*

Use Phone Verification REST API

Phone verification lets you verify global phone numbers in your applications. You can do individual or batch verifications.

This takes as input the phone number to verify in international format (for example, +447528471411) or in national format with a country code (for example, 07528471411 and GB as the country parameter). The response signifies the following:

- Valid: The phone number was successfully verified.
- Invalid: The phone number was not successfully verified.
- The request timed out before it could be completed.

Invalid and partially-valid verifications show the date of the most recent verification. To show the valid status on valid phone numbers right away, change the verification threshold date. For example, if you're working in Oracle CX Sales and the threshold is set to 180, then 180 days after the last valid verification was run, the UI renders with the phone number the **Verification Status**, **Verification Date**, a **Re-verify** icon, and an **Overwrite** icon on the Contact and Account Profile pages. To show the valid status with these icons right away, change the verification threshold to 0.

For detailed information, see Phone Verify in *REST API for Oracle Address, Email, and Phone Verification.*