Before you begin

Beta

This product is in a pre-release state and might change or have limited support. For more information, see the product launch stages (https://cloud.google.com/products/#product-launch-stages).

This page outlines the steps you must take before using Recommendations AI.

Note: Recommendations AI is in limited Beta release. To become a Beta customer, contact your Google account manager.

Set up your project

You must create a Google Cloud project in order to access Recommendations AI. To set up a project, follow these steps:

1. Go to the Cloud Console Manage resources (https://console.cloud.google.com/cloud-resource-manager) page.

   GO TO THE MANAGE RESOURCES PAGE (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/CLOUD-RESOUR
2. On the drop-down at the top of the page, select the organization in which you want to create a project.
3. Click **Create Project**.
4. In the **New Project** window that appears, enter a project name and select a billing account as applicable.
5. If you want to add the project to a folder, enter the folder name in the **Location** box.
6. When you're finished entering new project details, click **Create**.
7. After you have been added to the Beta program, enable the **Recommendation Engine** API for your GCP project.

ENABLE THE RECOMMENDATION ENGINE API (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/APIS/LIBR

Click the **ENABLE** button in the GCP console to enable the API.

Set up authentication for your application

You must set up both of the following authentication methods to access the Recommendations AI API:

- **Service Account** - Applications use a service account to authenticate calls to the product catalog item APIs (https://cloud.google.com/recommendations-ai/docs/reference/rest/v1beta1/projects.locations.catalogs.catalogItems) .
- **API Key** - Applications use an API key for calls to user event logging (https://cloud.google.com/recommendations-ai/docs/reference/rest/v1beta1/projects.locations.catalogs.eventStores.userEvents) and recommendation prediction (https://cloud.google.com/recommendations-ai/docs/reference/rest/v1beta1/projects.locations.catalogs.eventStores.placements/predict) APIs.

Create a service account

1. Go to the Cloud Console Credentials (https://console.cloud.google.com/apis/credentials) page.
2. Select the project that you are creating credentials for (the project may already be selected).
3. Click **Create credentials** and then select **Service account key**.
4. Fill in the following fields:

Field	Value
Service account	New service account
Service account name	***enter a name for your service account***

Field	Value
Role	Service Accounts > Service Account Token Creator Recommendations AI > Recommendations AI Editor
Key type	JSON

5. Click **Create** to create the service account.

   The JSON file that contains the public/private key for the new service account is automatically downloaded to your computer. **This JSON file is the only copy of the key for your service account. Be sure to store it in a secure location**. The JSON key file must be stored in a location that is accessible from your application (see ***your-service-account-json-key-file*** in Authenticating using a service account (OAuth 2.0) (#authenticate-java).

Add the service account to your local environment

If you want to make calls to the Recommendations AI by using the `cURL` command-line tool, you'll need to make your service account available in your local environment. `cURL` uses the `gcloud auth application-default print-access-token` command to obtain an access token for your service account using the Google Cloud Platform (GCP) Cloud SDK. To download and install the SDK, see Cloud SDK (https://cloud.google.com/sdk/).

```
export GOOGLE_APPLICATION_CREDENTIALS=path-to-service-account-json-key-file
```

Note: The `GOOGLE_APPLICATION_CREDENTIALS` environment variable will not persist if you close down your command window. To make sure that the variable is set to the path to your service account file when you open a new command window, set the value in an initialization shell script.

Create an API key

1. Go to the Cloud Console Credentials (https://console.cloud.google.com/apis/credentials) page.
2. Select the project that you are creating credentials for (the project may already be selected).
3. Click **Create credentials** and then select **API key**. *Do not add any referrer restrictions*. Some user privacy settings are known to *not* pass the referrer url.

- Take note of the generated API key, which you will use when calling user event logging and recommendation prediction APIs

4. For increased security, add an HTTP restriction (https://cloud.google.com/docs/authentication/api-keys#api_key_restrictions) to your API Key to restrict access to the Recommendations AI service at `https://recommendationengine.googleapis.com/*`.

Register an API key for predict calls

API keys are the fastest and easiest way to provide authentication when you make a call to the predict method. However, if you also use API keys for other API calls, such as logging user events, the API key is embedded in your website, which means it is visible to your users. Because predictions include PII (personally identifiable information) and incur charges, Recommendations AI provides an extra layer of security for prediction calls by requiring you to register any API keys you use for calls to the predict method. You can register up to 20 keys per project.

Important: You must ensure that your registered key is secure. Do not register any API key that you have previously exposed. After you register your key, do not embed it in any URL, JavaScript pixel or tag in your public website or otherwise make it publicly visible.

1. Create an API key specifically for predict calls by following the instructions in Create an API key (#create-key).
2. Register the key, using the predictionApiKeyRegistration.create (https://cloud.google.com/recommendations-ai/docs/reference/rest/v1beta1/projects.locations.catalogs.eventStores.predictionApiKeyRegistrations/create) method:

```
curl -X POST \
 -H "Authorization: Bearer $(gcloud auth application-default print-access-token
 -H "Content-Type: application/json; charset=utf-8" \
 --data "{
       "predictionApiKeyRegistration": {
            "apiKey": 'apiKey'
        }
    }"
 "https://recommendationengine.googleapis.com/v1beta1/projects/[PROJECT_ID]/loc
```

Examples

Authenticating using a service account (OAuth 2.0)

Here is a Java example of OAuth 2.0 authentication using a service account. More detailed instructions can be found in the authentication documentation (https://developers.google.com/identity/protocols/OAuth2WebServer). Google has client libraries in 7 languages (https://developers.google.com/identity/protocols/OAuth2WebServer#libraries) that you can use to implement OAuth2 authentication in your application. If you prefer to implement the HTTP/REST directly, follow the REST authentication instructions (https://developers.google.com/identity/protocols/OAuth2WebServer).

In the example, replace ***your-service-account-json-key-file*** with the path to the JSON key file for your service account.

```
// Simple Java example of using Google Cloud OAuth client library.
//
// Please see here for the list of libraries in different languages:
// https://developers.google.com/identity/protocols/OAuth2#libraries
//
// The following example depends on the google api client library.
//
// Maven:
//    <dependency>
//      <groupId>com.google.api-client</groupId>
//      <artifactId>google-api-client</artifactId>
//      <version>1.22.0</version>
//    </dependency>
import com.google.api.client.googleapis.auth.oauth2.GoogleCredential;
import com.google.api.client.googleapis.javanet.GoogleNetHttpTransport;
import com.google.api.client.http.GenericUrl;
import com.google.api.client.http.HttpRequest;
import com.google.api.client.http.HttpRequestFactory;
import com.google.api.client.http.HttpResponse;
import com.google.api.client.http.HttpTransport;
import com.google.api.client.http.json.JsonHttpContent;
import com.google.api.client.json.GenericJson;
import com.google.api.client.json.jackson2.JacksonFactory;
import java.io.FileInputStream;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.util.List;
```

```
public class RecommendationEngineApiSample {
  public static final String CREATE_CATALOG_ITEM_URL =
     "https://recommendationengine.googleapis.com/v1beta1/projects/[PROJECT_ID]/loca

  public static GoogleCredential authorize() throws Exception {
    return GoogleCredential.fromStream(new FileInputStream("your-service-account-jso
        .createScoped(Collections.singleton("https://www.googleapis.com/auth/cloud-p
        .setExpirationTimeMilliseconds(new Long(3600 * 1000));;
  }

  // Build an example catalog item.
  public static GenericJson getCatalogItem() {
    List<Object> categories = new ArrayList<Object>();
    categories.add(new GenericJson().set("categories", Arrays.asList("Electronics",
    categories.add(new GenericJson().set("categories", Arrays.asList("Laptops")));
    return new GenericJson()
        .set("catalog_item_id", "123")
        .set("title", "Sample Laptop")
        .set("description", "Indisputably the most fantastic laptop ever created.")
        .set("item_categories", categories)
        .set("language_code", "en")
        .set("original_price", 1000.00)
        .set("display_price", 800.00)
        .set("currency_code", "USD")
        .set("stock_status", "IN_STOCK")
        .set("available_quantity", 1219)
        .set("number_customer_reviews", 812)
        .set("average_rating", 4.5);
  }

  public static void main(String[] args) {
    try {
      GoogleCredential credential = RecommendationEngineApiSample.authorize();

      HttpTransport httpTransport = GoogleNetHttpTransport.newTrustedTransport();
      HttpRequestFactory requestFactory = httpTransport.createRequestFactory();
      HttpRequest request = requestFactory.buildPostRequest(new GenericUrl(CREATE_CA
          new JsonHttpContent(new JacksonFactory(), RecommendationEngineApiSample.ge
      credential.initialize(request);
      HttpResponse response = request.execute();
      System.out.println("Response content: " + response.parseAsString());
    } catch (Exception e) {
      e.printStackTrace();
    }
```

```
  }
}
```

Authenticating using an API key

Here is an example of authenticating using an API key from the command line using the `curl` command. Replace ***api-key*** with the API key that you created in the previous section.

```
URL="https://recommendationengine.googleapis.com/v1beta1/projects/[PROJECT_ID]/locat

DATA='{
  "user_attributes": {
    ...
  },
  "user_event_detail": {
    ...
  }
}'

echo $URL
curl -H 'Content-Type: application/json' -X POST -d "${DATA}"  $URL
```


Last updated January 17, 2020.