Vision Product Search - Android App

Authors: **Shreya Anand**   
Visibility: Across Alphabet   
Originally Proposed: 2021-06-29 / Last Updated: 2021-08-12

Objective

An android application to demonstrate the visual search feature for shopping applications - where the user has the option to upload an image which has the items that he/she is looking for and the application returns a list of visually similar products. The aim is to provide the retail customers with a demoable asset so that they can understand the power of this machine learning solution of Google.

# Requirements

* About Vision Product Search

<https://cloud.google.com/vision/product-search/docs>

* Using Android to Build Applications

<https://cloud.google.com/tools/android-studio/docs>

* Building a product search backend

<https://cloud.google.com/vision/product-search/docs/quickstart>

<https://cloud.google.com/vision/product-search/docs/tutorial>

* Using ML Kit

<https://developers.google.com/ml-kit/guides>

# Inspiration for the Project

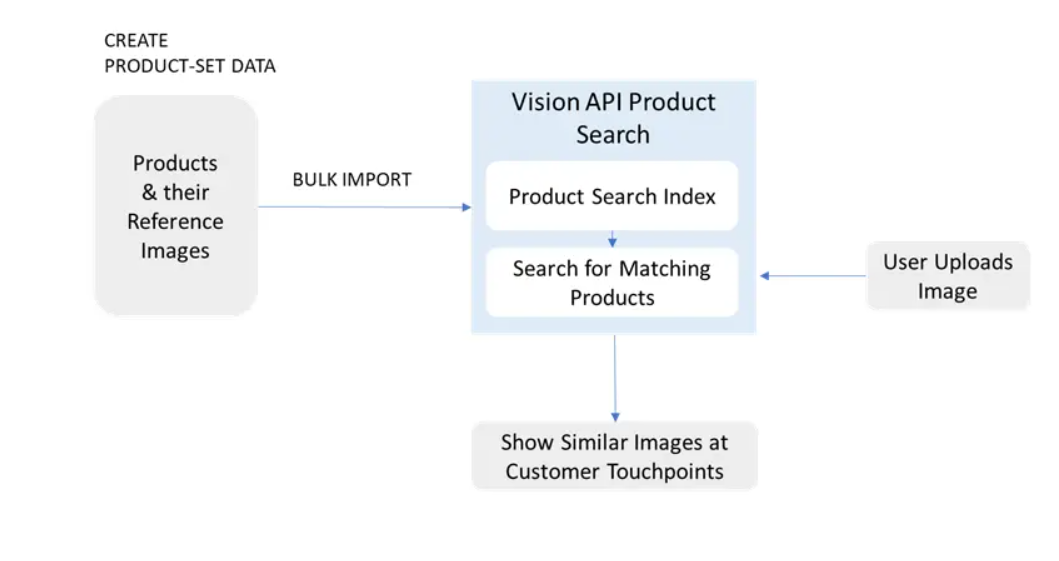
The first thing to notice is the rise in the online retailing activity, with the curve being steeper from the time covid-19 has hit us.

With the pandemic entering our lives, this means it was somewhat restricted. We were only left with our mobile phones and laptops and we obviously started making use of that to shop more than before. This has led to a challenge for the market players to retain the customers and keep up with their ever growing expectations. And amidst this, there is an escalated adoption of artificial intelligence, integrated with their android, web and ios applications.

So, one of the features which aim at increasing the overall satisfaction while shopping, is the visual product search which engages customers in new and exciting ways.

Google helps to achieve this with the Vision Product Search API which uses ML-powered object recognition and lookup. The aim is to Deliver seamless, memorable experiences with AI and experiential shopping.

# Architectural Diagram



# Design and Implementation Details

The application has three broad sections in terms of implementation -

### **1. Object Detection and Tracking**

I have added this feature using the ML Kit Object Detection and Tracking API.

* Added the following to my dependencies file (build.gradle)

implementation **‘com.google.mlkit:object-detection:16.2.4’**

* Then I made the following imports in my main file -

com.google.mlkit.vision.common.InputImage

com.google.mlkit.vision.objects.ObjectDetection

com.google.mlkit.vision.objects.defaults.ObjectDetectorOptions

* The function in the code sets up Object Detection and Tracking in 3 simple steps -

1. Configuring the object detector (the mode is SINGLE\_IMAGE\_MODE as I will be running it on a static image).
2. Preparing the input image from a Bitmap (for preparing the input image from other methods, visit [documentation](https://developers.google.com/ml-kit/vision/object-detection/android)).
3. Processing the image and if successful, I am only keeping the FASHION\_GOOD products for the purpose of the application.

### **2. Building the Product Search Back-end**

* The next step was to create the back-end for the vision API, which can then be called from the application to perform a visual search.

1. Before starting with any actual work, I enabled billing for the google cloud project. Also, I enabled the vision API for my project.
2. Created a service account and gave it the Basic>Owner role. Then, I created the service account key (the json key was downloaded to my computer).
3. Set the value of the variable GOOGLE\_APPLICATION\_CREDENTIALS to the path of the downloaded credentials file (Note that the value applies only for the current session of the Cloud Shell).
4. Then, I got the product set that would be used as the product catalog for my back-end service. I used the publicly available [product\_catalog.csv](https://storage.cloud.google.com/cloud-samples-data/vision/product_search/product_catalog.csv) located in a public Cloud Storage bucket.

* I used the bulk import to create the catalog for my backend using the following curl command -

curl -X POST \

-H “Authorization: Bearer “$(gcloud auth application-default print-access-token) \

-H “Content-Type: application/json; charset=utf-8” \

-d @import\_request.json \

<https://vision.googleapis.com/v1/projects/$PROJECT_ID/locations/$LOCATION_ID/productSets:import>

* Then, I made sure that the Product Search index of products is complete. I used the following command to verify if the indexing was complete -

curl -X GET \

-H “Authorization: Bearer $(gcloud auth application-default print-access-token)” \

-H “Content-Type: application/json” \

<https://vision.googleapis.com/v1/projects/$PROJECT_ID/locations/$LOCATION_ID/productSets>

* A successful response will return the indexTime of the product set for which indexing is complete.

**Note:** The indexing takes about 30 minutes to complete so it takes around 30 minutes for the changes to be reflected in the app.

### **3.** **Calling the Back-end from the Android Application**

* The final step is to call the back-end from the application — send the image as a query and receive a list of visually similar products and fetch the reference images of the products returned to display.
* Firstly, I created an API key and stored it. This will be used by the application to interact with the Vision API. To avoid unauthorized use, I restricted access of the API key to my application.

VISION\_API\_URL — <https://vision.googleapis.com/v1>

VISION\_API\_KEY — API key

VISION\_API\_PROJECT\_ID — Google Cloud project ID

VISION\_API\_LOCATION\_ID — The Region Name e.g. us-east1

VISION\_API\_PRODUCT\_SET\_ID — ID of the product catalog

* Then, I used the [projects.locations.images.annotate](https://cloud.google.com/vision/product-search/docs/reference/rest/v1/projects.locations.images/annotate) endpoint to send the query image to the server and receive a list of products from the product catalog that are visually similar to the query image.
* Finally, I called the [projects.locations.products.referenceImages.get](https://cloud.google.com/vision/product-search/docs/reference/rest/v1/projects.locations.products.referenceImages/get) API to get the URIs of the product images returned and displayed them in the application’s UI.

# Link to Code

Here is the link to the code of the project present in my public GitHub repository.

<https://github.com/shreyaanand29/Vision-Product-Search-Android-App/tree/master>

I code can be used as it is to build the application on one’s own system. One thing which should be taken into consideration - the developer must insert his/her own API key in the code where it is required, I have left that blank in the repo for the purpose of confidentiality.

# Next Steps

* The steps to the application can be to build a stronger search mechanism by making use of a larger dataset while building the product search backend and then integrating it with the application.
* To learn more about how to create a custom dataset and use it build the backend, visit

<https://cloud.google.com/vision/product-search/docs/product-categories>

<https://cloud.google.com/vision/product-search/docs/create-product-set#vision-product-search-create-prod-set-drest>

* Also, for custom datasets lying in the category not supported by vision product search, AutoML can be used to build the model using their own dataset and then the REST API provided at the end can be integrated with the application.
* To know more, visit <https://cloud.google.com/automl/docs#automl-vision>