



## **Inventory Tracking Application**

### **User and Developer Guide**

Developer's Particulars

Name	Position	Department
<b>Lau Wei Tang</b>	<b>Intern</b>	<b>Enterprise Engineering</b>

***Supervisor's Name: Mr. Chandrasekaran, Ashwin***



## Contents

1. Introduction.....	3
1.1. Inventory Tracking Application .....	3
1.2. Design Philosophy .....	3
2. User Guide .....	4
2.1. Filter Feature.....	4
2.1.1. Filtering by Brand .....	5
2.1.2. Filtering by Category .....	6
2.1.3. Filtering by Keywords.....	8
2.2. Picture Feature.....	9
2.2.1. Filtering by Picture.....	9
2.3. Translate Feature .....	10
2.3.1. Filtering by Translate .....	11
2.4. Speech Feature.....	12
2.4.1. Filtering by Speech.....	13
3. Developer Guide .....	14
3.1. Front-End Development .....	14
3.1.1. Setting-Up (Front-End) .....	14
3.1.2. Code Demonstration.....	15
3.1.3. Deployment (Front-End) .....	15
3.2. Back-End Development.....	16
3.2.1. Setting-Up (Back-End).....	16
3.2.2. Setting-Up (Google Cloud Platform) .....	16
3.2.2. Code Demonstration (Back-End) .....	17
3.2.2.1. Creating New Endpoint.....	17
3.2.3. Deployment (Back-End) .....	17

## **1. Introduction**

### **1.1. Inventory Tracking Application**

Inventory Tracking Application (ITA) is a web application aimed at equipping our Sales Associates with real-time data on the availability of our products. The purpose of ITA is to provide a seamless shopping experience to our customers. In order to achieve this goal, ITA has several built-in features designed to minimise the response time with regards to the availability of the products. As majority of DFS' customers are Chinese, most of the features provided by ITA are strategically targeted at them.

### **1.2. Design Philosophy**

ITA was designed with DFS' current system architecture in mind by adopting the 'Developer-First' design philosophy. The objective is to provide a realistic proof of concept while keeping the integrity of our system. This significantly enhances ITA's code maintainability and readability. For example, DFS stores are currently divided into various divisions with its respective division codes. This method greatly reduces traffic by retrieving relevant data only. This technique is similarly incorporated into ITA.

In the following sections, this report will provide an elaborated explanation of the various functionalities of ITA in the [User Guide](#) as well as a detailed guide for developers in developing new features into ITA in the [Developer Guide](#).

## 2. User Guide

This section provides a step-by-step guide in articulating the full functionalities of the various features implemented in ITA. There are four main features incorporated into ITA, namely [Filter](#), [Picture](#), [Translate](#) and [Speech](#).

### 2.1. Filter Feature

As DFS is the world's leading luxury travel retailer, DFS provides ample product selection from a wide range of luxury brands and categories. Furthermore, most customers have their preferred brands that they are loyal to or are looking for a product from a specific category. **Filter** was designed to provide flexibility and ease of product search through filtering by *brand*, *category* and/or *keywords*. This feature also supports for filtering based on multiple criteria. Figure 1 below shows the 'Filter' tab.

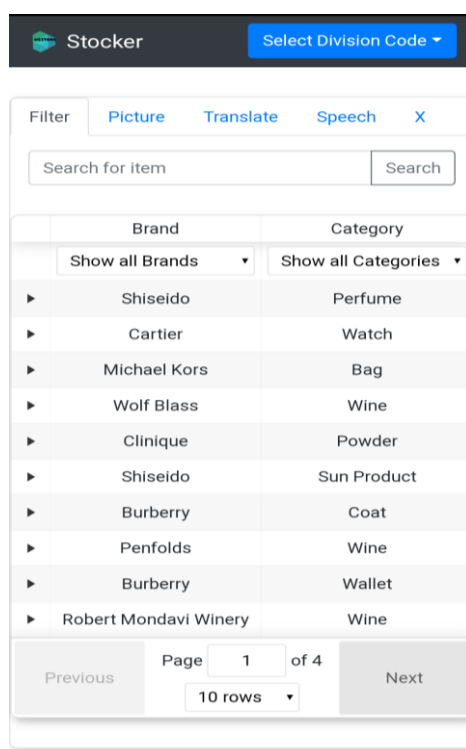


Figure 1

The following sub-sections will show sample usage scenarios using filtering by Brand, Category, and/or Keywords respectively.

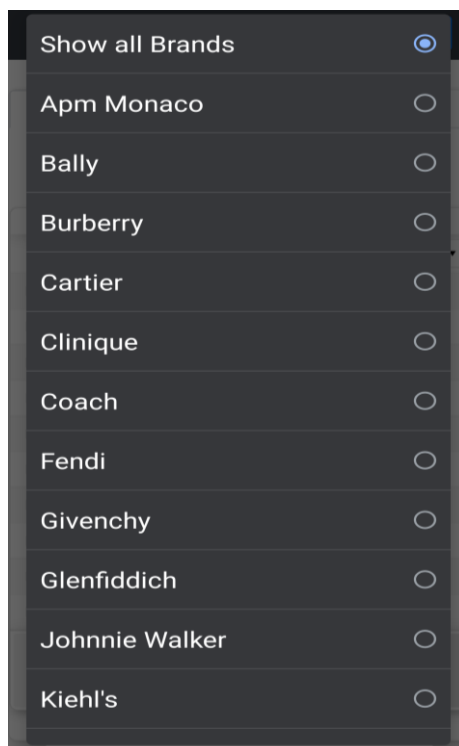
# Inventory Tracking Application

## 2.1.1. Filtering by Brand

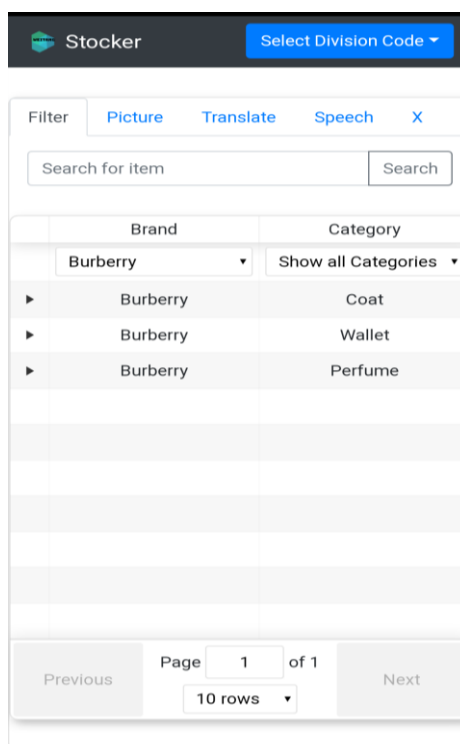
Step 1: Launch the web application

Step 2: Click on the 'Filter' tab to show Figure 1

Step 3: Click on 'Show all Brands' to show the following screen



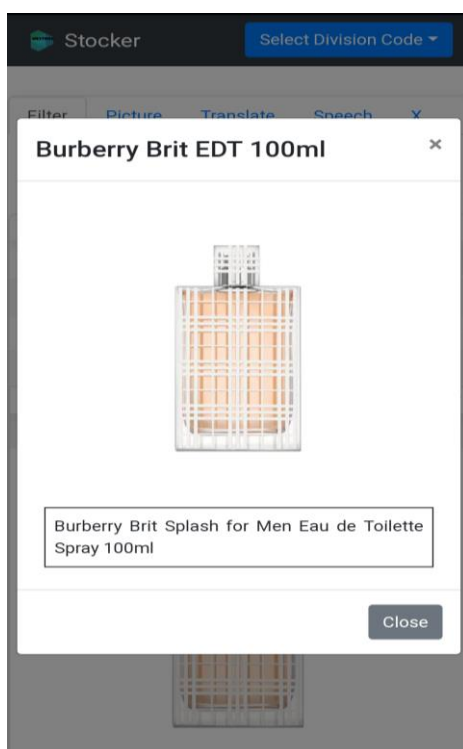
Step 4: Click on the brand that the customer is interested in to obtain the filtered list



# Inventory Tracking Application

---

Step 5: Click on the item for more information

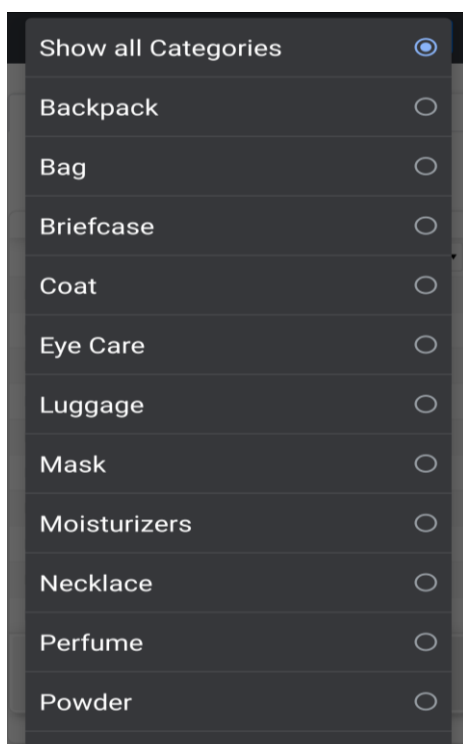


## 2.1.2. Filtering by Category

Step 1: Launch the web application

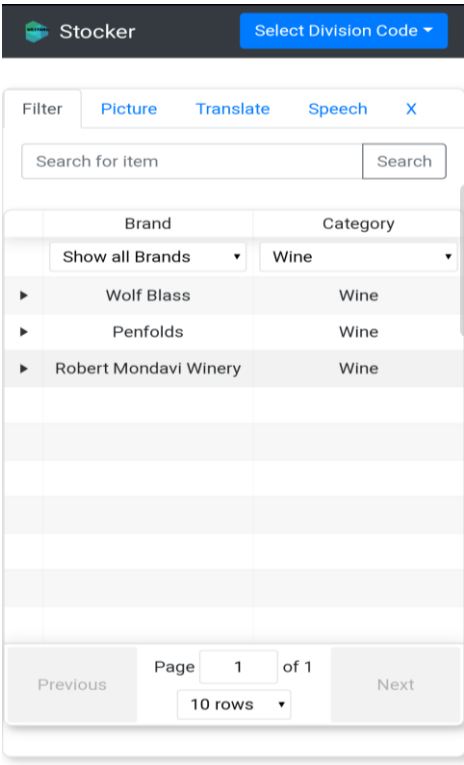
Step 2: Click on the 'Filter' tab to show Figure 1

Step 3: Click on 'Show all Categories'

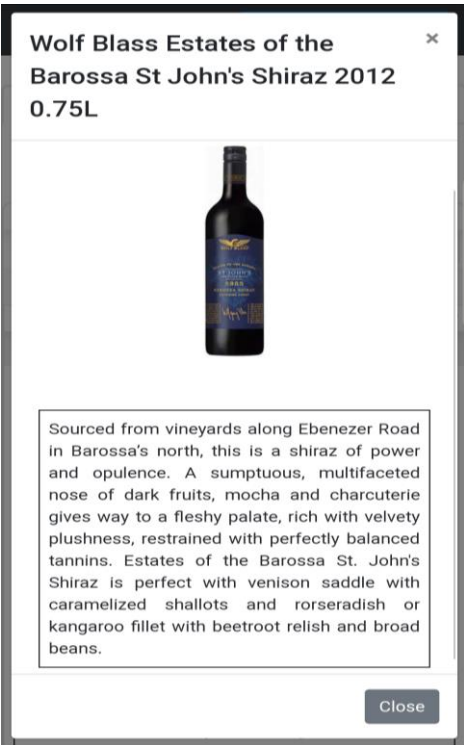


# Inventory Tracking Application

Step 4: Click on the category that the customer is interested in to obtain the filtered list



Step 5: Click on the item for more information



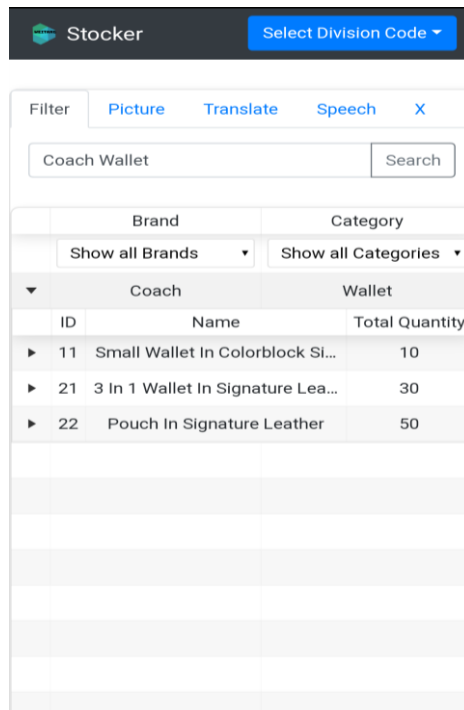
# Inventory Tracking Application

## 2.1.3. Filtering by Keywords

Step 1: Launch the web application

Step 2: Click on the 'Filter' tab to show Figure 1

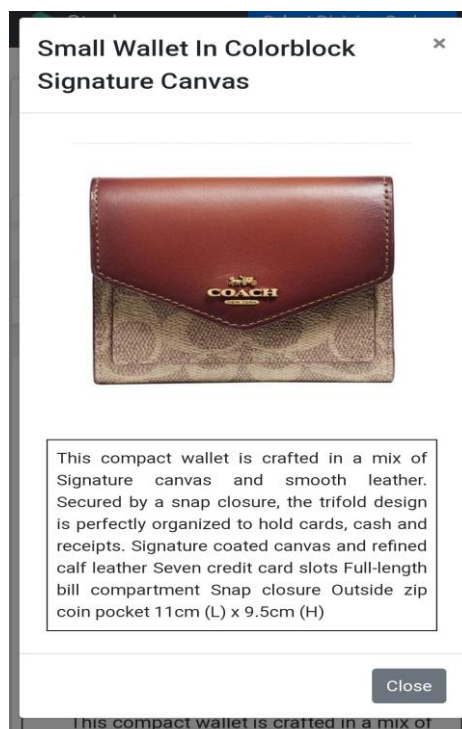
Step 3: Enter the keyword(s) in the 'Search for item' textbox to obtain results that matches all the keyword(s)



The screenshot shows the 'Stocker' application interface. At the top, there is a 'Select Division Code' button. Below it, a 'Filter' tab is active, with sub-tabs for 'Picture', 'Translate', 'Speech', and 'X'. A search bar contains the text 'Coach Wallet' and a 'Search' button. Below the search bar, there are two dropdown menus: 'Brand' (set to 'Show all Brands') and 'Category' (set to 'Show all Categories'). The search results are displayed in a table with columns for 'ID', 'Name', and 'Total Quantity'. The results are filtered to show only items under the 'Coach' brand and 'Wallet' category.

ID	Name	Total Quantity
11	Small Wallet In Colorblock Si...	10
21	3 In 1 Wallet In Signature Lea...	30
22	Pouch In Signature Leather	50

Step 4: Click on the item for more information







# Inventory Tracking Application

Step 4: Either select or take a picture of the product

Step 5: Click the 'Upload' button to view the result

Brand	Category
▶ Show all Brands	▶ Show all Categories
▶ Cartier	Watch
▶ Mido	Watch

## 2.3. Translate Feature

As DFS has retail stores globally, it is inevitable that the sales staff will attend to customers from different nationalities. **Translate** aims to remove the language barrier by allowing you to enter the product descriptions in any language and return all relevant products. Figure 3 below shows the 'Translate' tab.

Brand	Category
▶ Show all Brands	▶ Show all Categories

Figure 3

# Inventory Tracking Application

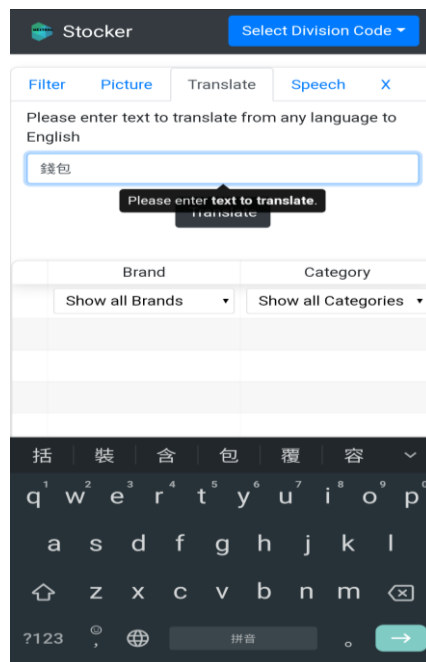
The following sub-section will show a sample usage scenario that provides a clear view of the inner workings of the Picture feature.

## 2.3.1. Filtering by Translate

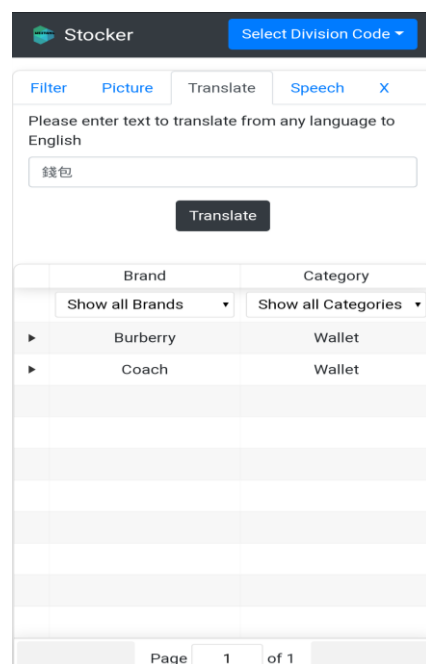
Step 1: Launch the web application

Step 2: Click on the 'Picture' tab

Step 3: Enter the keyword(s) from any language in the 'Text to Translate' textbox



Step 4: Click on the 'Translate' button to view the result



## 2.4. Speech Feature

The biggest limitation of the **Translate** feature is that it assumes you have the relevant keyboard preinstalled onto your device. Without the appropriate keyboard of a specific language, the feature is unusable. Furthermore, if you can enter the keywords using a foreign language, you most probably know the translated English keyword too. **Speech** aims to overcome this limitation by translating speech from a foreign language directly into English keyword(s). However, given the current lack of support for Google Cloud Platform (GCP) Speech-To-Text API, this feature is unable to detect the language of the speech and thus is set to Chinese by default. However, this detection feature is currently in beta. Figure 4 below shows the ‘Speech’ tab.

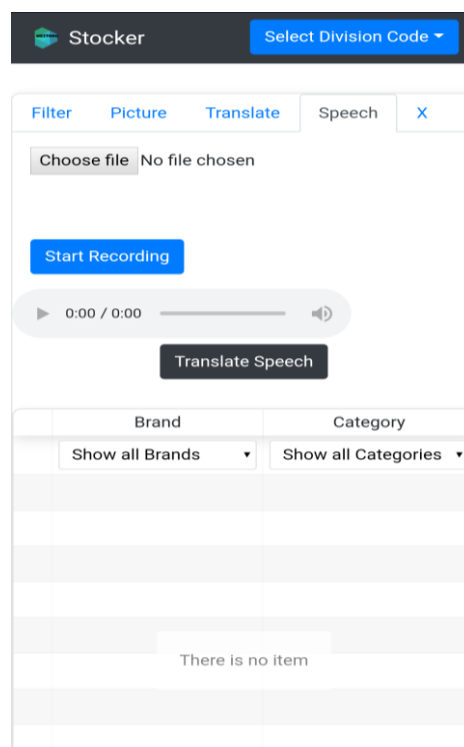


Figure 4

The following sub-section will show a sample usage scenario that provides a clear view of the inner workings of the Picture feature.

# Inventory Tracking Application

## 2.4.1. Filtering by Speech

Step 1: Launch the web application

Step 2: Click on the 'Picture' tab

Step 3: Click on 'Start Recording' button to record your speech. Click on 'Stop Recording' button once you are done recording. You may click play on the audio player to verify the recorded speech

Step 4: Click on 'Translate Speech' button to view the result

The screenshot shows the 'Stocker' application interface. At the top, there is a dark header with the 'Stocker' logo and a 'Select Division Code' dropdown menu. Below the header, there are four tabs: 'Filter', 'Picture', 'Translate', and 'Speech'. The 'Speech' tab is currently selected, and a close button 'X' is visible. Inside the 'Speech' tab, there is a 'Choose file' button and the text 'No file chosen'. Below this is a blue 'Start Recording' button. Underneath is an audio player with a play button, a progress bar showing '0:00', and a volume icon. Below the audio player is a 'Translate Speech' button. At the bottom, there is a table with two columns: 'Brand' and 'Category'. Each column has a 'Show all' dropdown menu. The table contains two rows of data: 'Burberry' and 'Coach' under the 'Brand' column, and 'Wallet' under the 'Category' column.

Brand	Category
Show all Brands	Show all Categories
Burberry	Wallet
Coach	Wallet

If your device does not support audio recording or prefers to select a recorded audio, you can use the 'Choose file' button instead.

### Upload Picture

Choose file No file chosen

## **3. Developer Guide**

This section provides a detailed tutorial in assisting you in understanding the code. This guide aims to reduce the time needed for new developers to get accustomed to the code structure. There are two main portions to this guide, **Front-End** and **Back-End** development. In each of these parts, there will be Setting-Up, Code Demonstration and Deployment guide.

### **3.1. Front-End Development**

The Front-End of the application is written in React.js. Before getting started with this tutorial, it is highly recommended that you have JavaScript and basic React knowledge. Here are some useful tutorials in helping you to understand the respective programming language / framework.

[JavaScript Tutorial](#)

[React Tutorial](#)

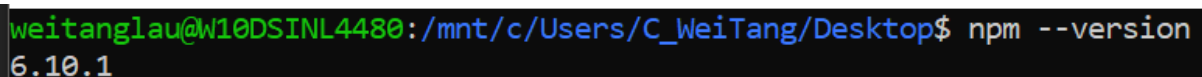
Please feel free to skip these tutorials if you have prior coding experiences in JavaScript / React.

#### **3.1.1. Setting-Up (Front-End)**

Step 1: Ensure that you have Linux coding environment set up. E.g. [Windows Subsystem for Linux](#).

Step 2: Install [Node.js](#)

Step 3: Ensure that you have Node.js installed by entering `npm --version`.



```
weitanglau@W10DSINL4480:/mnt/c/Users/C_WeiTang/Desktop$ npm --version
6.10.1
```

Step 4: Fork and download the [repo](#). Ensure that you have the read/write permission.

Step 5: Navigate to the repo and enter 'npm install'. This command will download all the relevant dependencies for this project.

Step 6: Enter 'npm start' to open the application in [localhost:3000](#)

# Inventory Tracking Application

---

## 3.1.2. Code Demonstration

As this application obtains information from the [Back-End](#) server, you have to make sure that the application is pointing to the correct API to send the HTTP request. Navigate to 'App.js', edit the 'url' state as shown below. Similarly, enter the default division code if necessary.

```
this.state = {  
  url: /* Enter the API base URL HERE */,  
  divisionCodes: [],  
  selectedDivisionCode: "41", // Enters default division code  
  items: [],  
  filteredItems: [],  
  brands: [],  
  categories: []  
};
```

If there is a change in the HTTP endpoint, edit the url in the fetch method. For example, if the brands endpoint has been changed to divisionBrands, simply change the url to 'this.state.url + "divisionBrands/"'.

```
fetch(this.state.url + "brands/", {  
  method: "get",  
  headers: {  
    "Content-Type": "application/json",  
    Accept: "application/json"  
  }  
})  
  .then(response => response.json())  
  .then(data => {  
    this.setState({ brands: data });  
  });
```

## 3.1.3. Deployment (Front-End)

This application was designed to deploy onto Github Pages, you might need to refer to external deployment tutorial other deployment site is being used.

For deployment, navigate to 'package.json' file. Enter the deployment site in the 'homepage' property as shown below.

```
"homepage": /* Enter the deployment URL here */,
```

After you are done, enter 'npm run deploy' command to deploy.

## **3.2. Back-End Development**

The Back-End of the application is written in Java Spring Boot and Maven is used as the build tool. Before getting started with this tutorial, it is highly recommended that you have Java and basic Spring Boot knowledge. Here are some useful tutorials in helping you to understand the respective programming language / framework.

[Java Tutorial](#)

[Spring Boot Tutorial](#)

Please feel free to skip these tutorials if you have prior coding experiences in Java / Spring Boot.

### **3.2.1. Setting-Up (Back-End)**

Step 1: Ensure that you have Linux coding environment set up. E.g. [Windows Subsystem for Linux](#).

Step 2: Install [Java 8 JDK](#)

Step 3: Ensure that you have Java installed by entering `java --version`.

Step 4: Fork and download the [repo](#). Ensure that you have the read/write permission.

Step 5: Open the source code using an appropriate IDE (e.g. IntelliJ).

Step 6: Build and run the project

Step 7: Ensure that the project has been properly set up by entering

<http://localhost:8081/inventories/divisionCodes> to show all the division codes.

### **3.2.2. Setting-Up (Google Cloud Platform)**

As the application uses advance features provided by Google Cloud Platform (GCP), there are several set up necessary to use Google's software.

Please refer to this sample [tutorial](#) to enable the GCP Vision API. Please ensure that the environment variable GOOGLE\_APPLICATION\_CREDENTIALS is pointing to the JSON file that contains the service account before deployment.



## 3.2.2. Code Demonstration (Back-End)

As this application serves as a REST API, it is crucial that you are aware of how to create relevant endpoints for the users.

### 3.2.2.1. Creating New Endpoint

All endpoints are created in the **ItemController.java**. Sample '/brands' endpoint as shown in the image below. To create a new endpoint, simply change the value to reflect the endpoint, the HTTP method and create a new method in **itemService.java** (please use same method name for simplicity). The logic portion is being coded in **itemService.java**.

```
/**
 * Returns the list of brands in JSON format.
 * @return The JSON format of all the brands
 */
@RequestMapping(value = "/brands", method = RequestMethod.GET)
public ResponseEntity<String> getAllBrands() { return itemService.getAllBrands(); }
```

## 3.2.3. Deployment (Back-End)

Java Spring Boot provides production ready solution to its users. Simply enter 'mvn install' to obtain the JAR file which can then be used for deployment. Please ensure that the environment variable **GOOGLE\_APPLICATION\_CREDENTIALS** is pointing to the JSON file that contains the service account before deployment.