

National College of Ireland

BSc in Computing

2017/2018

Name Surname

Rafael Alves

Student ID

x16109716

email

x16109716@student.ncirl.ie

Full Title

IMsearch Technical Report



## ***Table of Contents***

Table of Contents .....	3
Executive Summary .....	6
1. Introduction .....	7
1.1. Background .....	8
1.2. Aims .....	9
1.2.1 Practical usage .....	9
1.3. Technologies .....	10
1.3.1. Android studio .....	10
1.3.2. Firebase .....	10
1.3.3. Java (Android studio) .....	10
1.3.4. Google vision.....	10
1.3.5. eBay API .....	11
1.3.6. JSON .....	11
1.3.7. GitHub / Bitbucket.....	11
1.4. Structure .....	11
1.4.1. System .....	11
1.4.2. Conclusions .....	12
This section will report the benefits and disadvantage of the application, analysing the functionalities results and user satisfaction. ....	12
1.4.3. Further Developments or Research .....	12
1.4.4. References .....	12
2. System .....	13
2.1. Requirements .....	13
2.1.1. Functional requirements .....	13
2.1.2. Nonfunctional requirements .....	22
2.1.3. Data requirements .....	24
2.1.4. User requirements .....	24
2.1.5. Environmental requirements .....	25
2.1.6. Usability requirements .....	25
2.2. Design and Architecture .....	26
2.3. Implementation .....	29

2.4. Graphical User Interface (GUI) Layout .....	46
2.5. Testing .....	47
2.6. Customer testing .....	48
IMsearch questionnaire.....	49
2.7. Evaluation .....	50
3. Conclusions.....	51
4. Further development or research .....	52
5. References.....	53
5.1. Monthly Journals.....	54
• Journal September 2017 .....	54
• Journal October 2017 .....	54
• Journal November 2017 .....	54
• Journal December 2017 .....	54
• Journal January 2018 .....	55
• Journal February 2018 .....	55
• Journal March 2018.....	55
• Journal April 2018.....	56
• Journal May 2018 .....	56
6. Appendix .....	58
6.1. Project Proposal.....	58
Executive summary .....	58
Background.....	59
1.2 Idea .....	60
1.3 Technologies .....	61
2 System .....	61
2.1.1 Functional requirements .....	62
6.2. Project Plan.....	70
6.3. Other Material Used .....	70
6.4. Key terms .....	72

## Executive Summary

The amount of information exchanged through mobile application by users is been increasing for last decade, one of the reasons behind it is the implementation of better features and services that mobile devices can handle, cell phones are easy acquirable products and reach every economic level of society, opening space for development of new product and services to different type of clients narrowing the production of the application for specific service, the result of democratic technology is the generation of new fields to be explored Big data is clear example of technology evolution and data usage. One of the functionalities that became part of the cell phone standard is the camera into the smartphone which varies from a low resolution to really high quality image depending on the hardware provider and cost of it. According with Flickr report the number of public picture posted on the web reaches over 50 million a day, the number of public post has decreased along the years due the fact that network started occupying other roles besides connect people. This measure does not include the private pictures exchanges well known application that are used to perform this service are not included as they belong to a private network just the number of message handle by Facebook and WhatsApp sums up 60 billion message a day including photos, texts, links and documents. With introduction of APIs and Infrastructure as a service (IAAS) have increased the possibilities of combinations from different sources to create products that satisfy users just in simple steps which differentiates one application to another. The availability of API that can identify object is a new feature to be explored and can be used to start to change the conventional way to do tasks, the creation of new services can be explored in this area.

# 1. Introduction

IMsearch is a multiple API's Android Mobile application used to provide users the facility of finding products using image recognition technology. This application provides users an optional way to find available eBay products and purchase using crossing APIs systems.

The application uses different sources of evaluation to compound and filters a purchase search securely, the implementation of social network profile information will be used to add another layer of authenticity verification. This application also provides to the users an entire online shopping experience, from the image search to purchase of products using multiple crossing API's. The development of the application also aims to be intuitive and user-friendly targeting niches of market such people that have writing difficulties, the need of typing will be taken away by using the image extracted data, as result, there will not have type mistakes when the query is sent from the user's device.

The application requires an user registration and has available two methods to sign up, this process will use the application registration link or a valid social network credential which will be evaluated by the API response and will be encrypted and stored in case of positive response. Using the camera of the mobile device the user will produce an image of an object, the image will be evaluated by the image recognition API, filtered and classified as eligible to a search or not considering the application purpose, reducing attempts of inappropriate content for the application context, once the product's image is eligible to be a search the data extracted from the image will be passed in JSON communication language to eBay and the product will be displayed on the user screen.

The user will be able to fill personal and purchase information to facilitate the process of the application, the search will be saved in history and this information will be encrypted stored.

## **1.1. Background**

The need to shape a primary material to generate a service is one of the advantages that technology has implemented over the years, the multiple releases of the same software with different versions are the results of a more communicative and exigent market, this market is also used as tool playing a role that provides suggestions and feedback improving the current version of products. The image recognition technology is one the new features that has space to develop new ways to perform the same task, reaching the same results using a alternative option can be implemented into different areas in the market, create a simple and practical tools that gives satisfactory results is important when comes to the current e-market environment, which has multiples services doing the very same task efficiently, to introduce the new in easy and understandable way to a old customer is a hard task considering automation of process that is implemented in every layer of customer usage when come to mobile development, as result the competition increases in the market and push up the quality of new products and services that need somehow to perform a functionality better than the current scenario offers to strike users loyalty. Currently Technology has reached a point where the personal preference is the key of selling an app or even interfere on the choice of mobile or a car.



## **1.2. Aims**

To develop a product search android application using image recognition technology, integrating multiple existent systems using different API sources to automate search process based on results generate from the different components of the application, mainly eBay and Google Vision interchanging data to provide the user the entire online shopping experience.

The user will be allowed to use the application functionalities once the registration process is passed, to better satisfy convenience, the user will be provided with two options to sign up, being by existing social network credentials or directly on the application, any information given from the user will be encrypted and stored in application database. The application provides the user a mistype facility due the fact that the picture data extraction is converted into text format.

### **1.2.1 Practical usage**

Sign up / Provide valid username (email) and password registered in the application.

Profile page / Fill personal information.

Profile page / Fill purchase information.

Image / Provide the application with a product image.

Image / The photo can be selected from the phone library.

History / Search activities will be stored in the database.

History / Purchase activity such as date, product name and type will be stored in the database.

## **1.3. Technologies**

### **1.3.1. Android studio**

Android Studio provides the fastest tools for building apps on every type of Android device.

World-class code editing, debugging, performance tooling, a flexible build system, and an instant build/deploy system all allow you to focus on building unique and high-quality apps.

### **1.3.2. Firebase**

The Firebase Realtime Database is a cloud-hosted NoSQL database that lets you store and sync data between your users in Realtime, firebase security measure is implemented in the application using SHA key which allow only the IMseach domain to be able to write and read data from the project.

### **1.3.3. Java (Android studio)**

The Java™ Programming Language is a general-purpose, concurrent, strongly typed, class-based object-oriented language. It is normally compiled to the byte-code instruction set and binary format defined in the Java Virtual Machine Specification.

### **1.3.4. Google vision**

Google Cloud Vision API enables developers to understand the content of an image by encapsulating powerful machine learning models in an easy to use REST API. It quickly classifies images into thousands of categories (e.g., "sailboat", "lion", "Eiffel Tower"), detects individual objects and faces within images, and finds and reads printed words contained within images. You can build metadata on your image catalog, moderate offensive content, or enable new marketing scenarios through image sentiment analysis. Analyse images uploaded in the request or integrate with your image storage on Google Cloud Storage.

### **1.3.5. eBay API**

Normally, users buy and sell items using the eBay online interface, interacting with eBay directly. But with the eBay API, you communicate directly with the eBay database in XML format. By using the API, your application can provide a custom interface, functionality and specialised operations not otherwise afforded by the eBay interface.

### **1.3.6. JSON**

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

### **1.3.7. GitHub / Bitbucket**

GitHub / Bitbucket are code repositories services platform created to manage code easily. From open source to business, you can host and review code, manage projects, and build software alongside millions of other developers.

## **1.4. Structure**

### **1.4.1. System**

The system requirements section will be structured firstly giving an overview of each application functionality, description and user case diagram with functionalities of each page available to the user. This section will also hold the different requirements for the application such as data requirements, environmental requirements, user requirements and usability requirements.

This section will also hold the current application design and architecture layers.

#### **1.4.2. Conclusions**

This section will report the benefits and disadvantage of the application, analysing the functionalities results and user satisfaction.

#### **1.4.3. Further Developments or Research**

This section will discuss what can be developed to attend and improve the application functionalities and user experience.

#### **1.4.4. References**

This section will hold the sources of information used during the development of the application.

## **2. System**

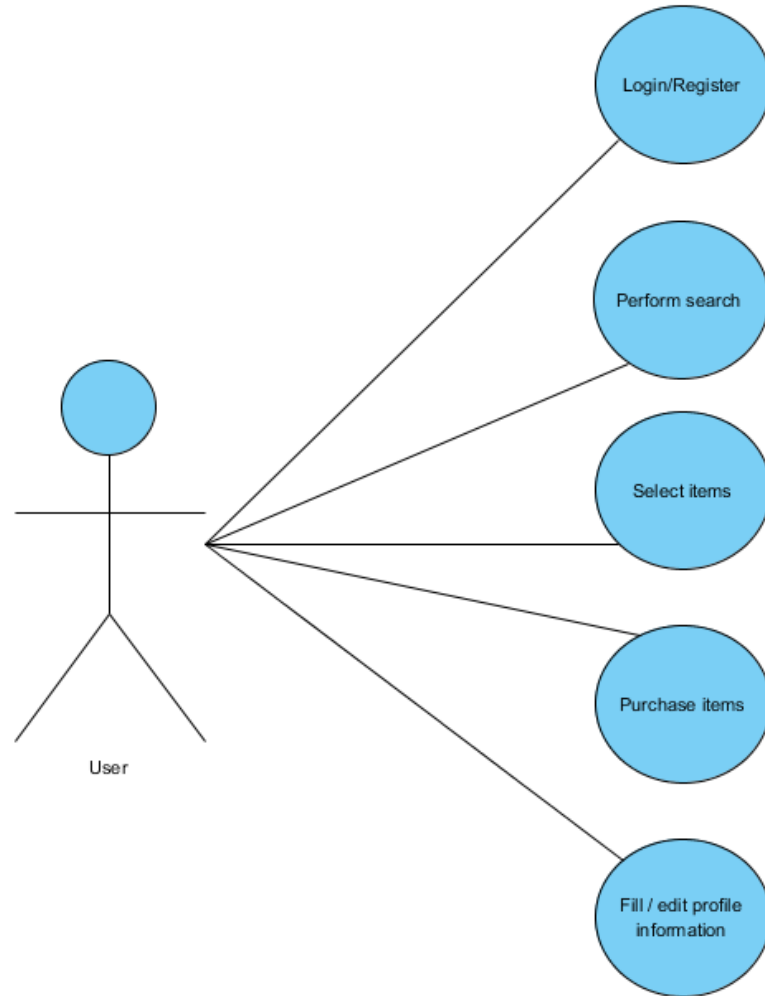
### ***2.1. Requirements***

This section will be divided into Data requirements, User requirements, Environmental requirements and Usability requirements being, the section will have title, description and a use case diagram representing the possible action that the user can take on each of application screens.

#### **2.1.1. Functional requirements**

The functional requirements of the application were defined based on the different sources of information services such as Google vision, eBay API and user experience.

### 2.1.1.1. Use case diagram Imsearch functionalities diagram UCD1



**Precondition**

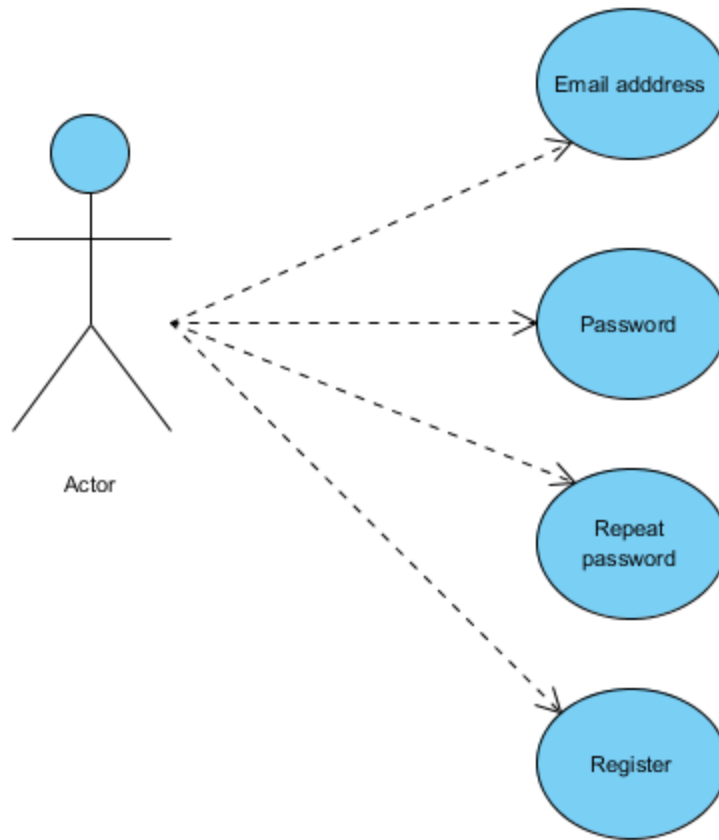
**Trigger**

### 2.1.1.2. Requirements <Registration page>

#### 2.1.1.2.1. Description

The use case diagram represents one of the application available methods for user registration, the first representation is sign up directly with the application page.

#### 2.1.1.2.2. Registration Use case diagram UCD2



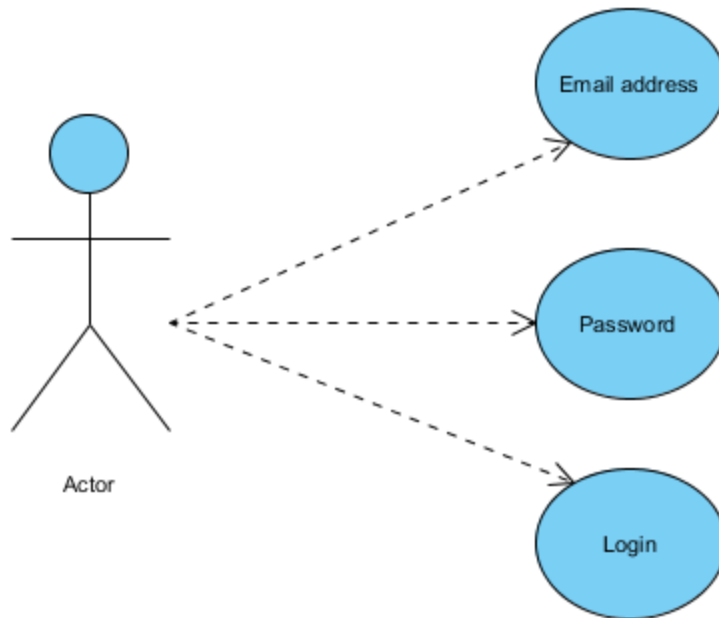
#### Trigger

##### *2.1.1.3. Login requirement <Login page>*

#### 2.1.1.3.1. Description

After registering using one of the methods available by the application the user will be able to login in the application.

#### 2.1.1.3.2. Login use case diagram UCD4



**Precondition**

**Trigger**

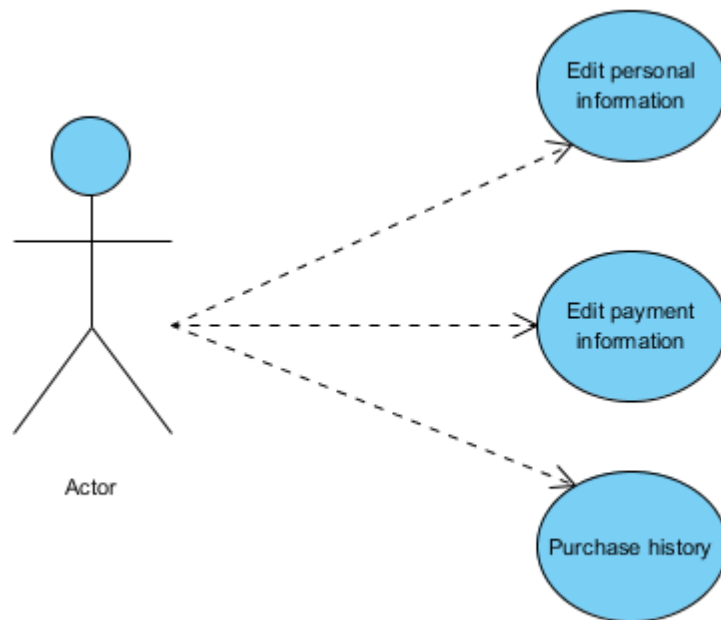
#### 2.1.1.4. *Profile requirement<Profile page>*

##### 2.1.1.4.1. Description

To use all the available application functionality the user will need to fill the profile information form.



#### 2.1.1.4.2. Profile use case diagram UCD5



#### **Precondition**

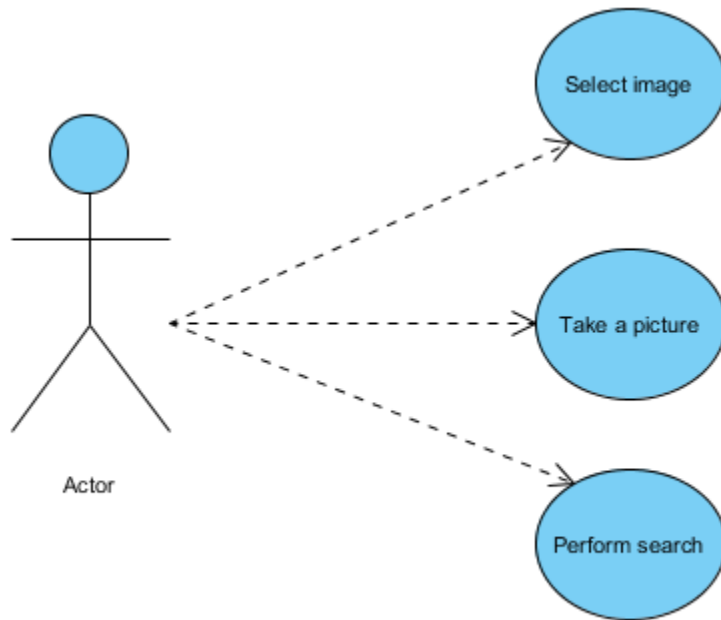
#### **Trigger**

#### *2.1.1.5. Home page requirement<Home page>*

##### 2.1.1.5.1. Home page description

The home page will hold two features where the user can have the option to choose between selecting an image from the phone gallery or take a picture of the desire object to be searched.

#### 2.1.1.5.2. Home page use case diagram



#### **Precondition**

To perform this action the user needs a image from the phone gallery or a working camera on the device.

#### **Trigger**

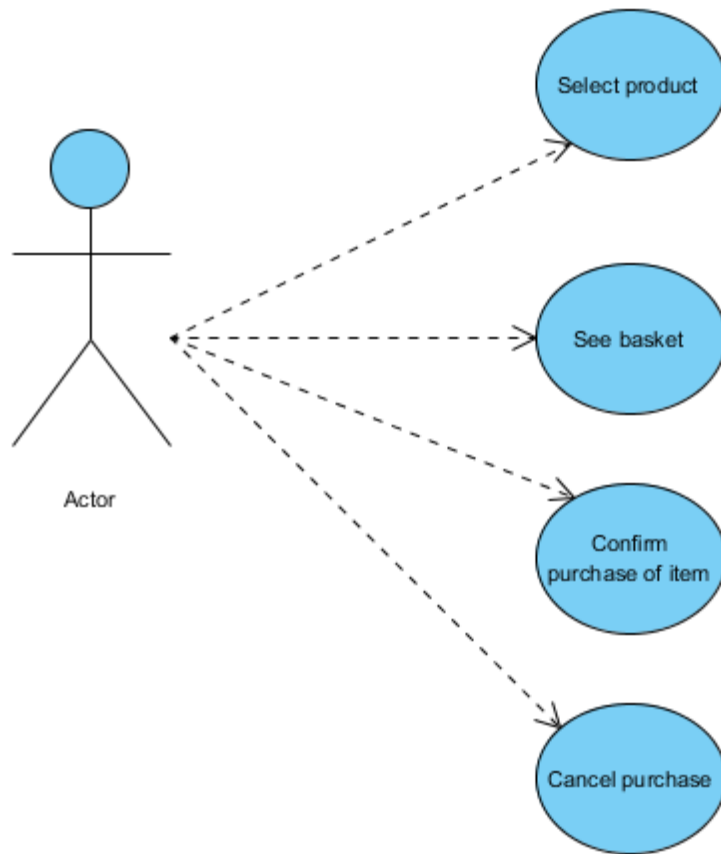
This section pf the application starts when the user selects an available product from the eBay search.

#### *2.1.1.6. Cart requirement <Camera capture screen>*

##### 2.1.1.6.1. Description

The user will be able to search for products using its image, the item selected will be placed in a basket and the user will be able to navigate back to the purchase and select another product.

#### 2.1.1.6.2. Purchase use case diagram UCD7



##### **Precondition**

To perform this action the user must have valid login credentials and full filament of the personal information form, the registration and login stage must be passed as valid to reach this part of the software.

##### **Trigger**

This section of the applications starts when the user selects one or more products from the search results enabling the checkout button.

##### **Precondition**

To perform this action the user needs to have a IMsearh account with valid personal information filled, the desired product must be select into a basket, the purchase confirmation must be validated.

## Trigger

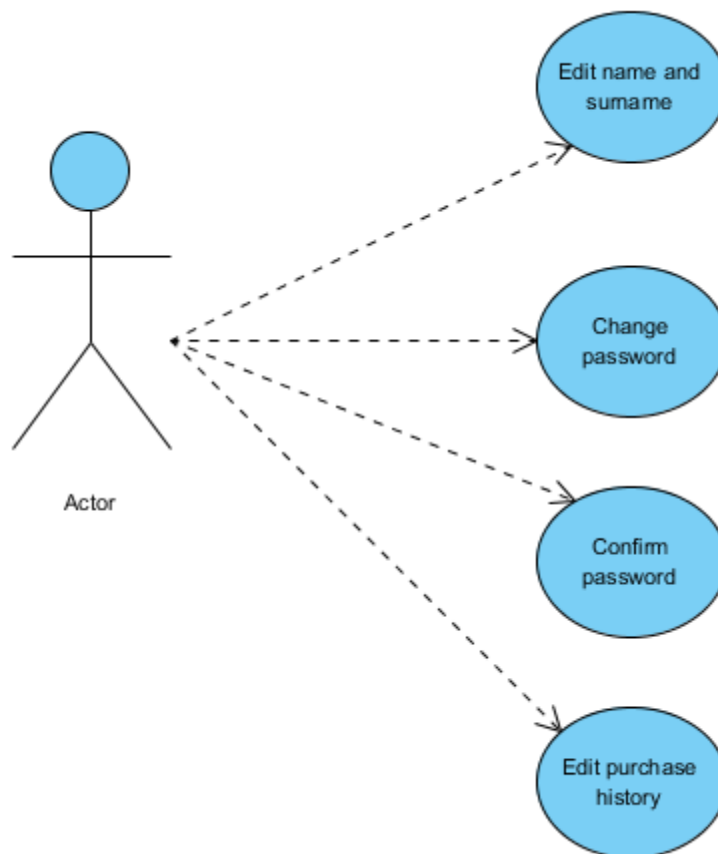
This action of the application starts when the user selects and confirm the products chosen for purchase.

### *2.1.1.7. Edit profile requirement <Profile page edit>*

#### 2.1.1.7.1. Description

The user has the option of change any credentials related to its account information.

#### 2.1.1.7.2. Edit profile use case diagram UCD9



## Precondition

To perform this section of the application the user must have a registered account.

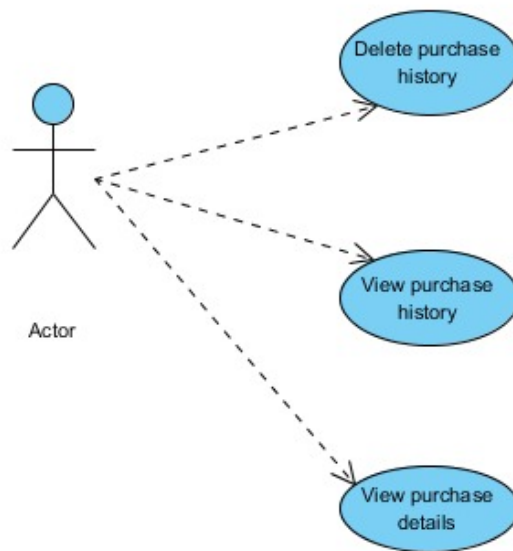
## Trigger

### *2.1.1.8. User purchase history requirement<Purchase history page>*

#### 2.1.1.8.1. History page description

In this section the user will be able to verify the activity in its account.

#### 2.1.1.8.2. History page use case diagram UCD11

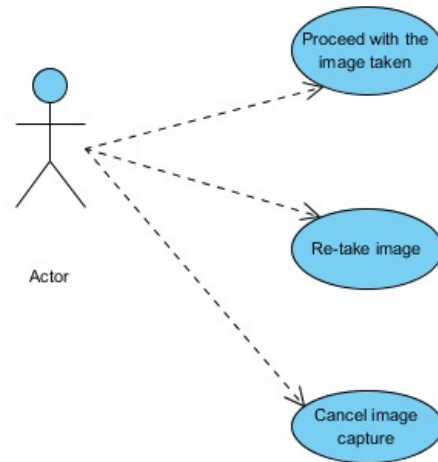


### *2.1.1.9. Re-take image requirement <Re-take image>*

#### 2.1.1.9.1. Re-take image screen description

In this section the user will be able to re-take the image picture, the action will redirect the user to the camera screen.

#### 2.1.1.9.2. Re-take image use case diagram UCD12



## 2.1.2. Nonfunctional requirements

### 2.1.2.1. Availability requirement

The application should have a high level of availability as it relies on different API that are well designed and used in high demand from multiple different softwares and platforms as source of information, the installation on the user's device combined with internet connection should give access to the services in Imsearch with no issues. A third party database choice was due the storage availability and encryption matter as the application will hold some user's personal information, google cloud instance as server handling the API request, google cloud vision and eBay API also were chosen to provide the end user a fine level of services and availability. The user has the option to select a image from their library which can work as an option when internet connection is not present, giving the user the facility of taking the picture of the desired product and search for it in another occasion where there is internet connection. In fact if the camera of the mobile with the installation of Imsearch is broken yet the user can select picture from the library received from other applications on the phone.

### 2.1.2.2. Security requirement

The security measure taken in the application are user's data oriented, where encryption and availability are present in all data stored, in case of data loss of accidental detection of data, automated backups are set save information time to

time and it can be recovered to the previous state in short time fashion, as result it mitigates the usability within other possible issues. The registration process also is used to reduce the number of possible pseudo users, using industry standards for email registration as well as password characters.

#### *2.1.2.3. Third party services requirements*

The application relies most of its functionality based on third party services in order to work and provide to the end user product image recognition search, the search results will be successful only if all parts are working combined, as the initial request come from the users products interest, in a second stage the image of the product is stored in the phone's library, the third stage is image recognition from google vision API, the data extracted and filtered from the picture is sent back to the application and redirected to another API, eBay, and finally fetching the desired product to the user's phone screen, the user has the option to purchase the selected product if they want to.

#### *2.1.2.4. Reliability requirements*

As the number of users in application can increase exponentially due the amount of users that can have access to it just browsing couple of pages and downloading the application, is hard to measure how popular an application can become, the users feedback is also crucial to the application to gain space and popularity, reliability would be measured how well the application is able to respond to requests and yet provide the user a nice and reliable experience. The action taken in Imsearch to keep these items working is to use third party technologies services that already have high demand market relied on it, the application will not work only if one of these third party elements are not functioning as expected.

#### *2.1.2.5. Maintainability requirement*

This section of the development should be pointed out as the application must be designed and written in a clear matter where the different classes and responses are able to be changed and updated reducing the interference one to another,

the application source code has each action separated and can be understood, changed if there is need for updates or modifications in the application functionality or front-end experience.

### **2.1.3. Data requirements**

#### ***2.1.3.1 Firebase***

Firebase will store and retrieve user's information and purchase activity related to the users profile. As the database is used as a service the storage maintainability is done by the third party entity.

#### ***2.1.3.2 JSON***

JSON is the exchanging information language in Imsearch in fact the application needs to rely in well designed and understandable machine language acceptable in different APIs, with three distinct main services compounding one search results to users.

### **2.1.4. User requirements**

Imsearch aims to provide the end user a option to automatically search products in eBay using image recognition instead of text based queries. For better use engagement the application automates the typing step currently used in most of application that has the nature of searching products online.

The end user must own an android device where the application will be installed.

The user will fill out personal information in order to register in the application to build he/she profile.

The user must have internet access as the entire application relies on third party available services.

Only users registered in the application database will be able to perform searches.



The user will be able to select a product and place it into a cart, where the purchase information will be kept until further decision.

The user is required to take an image where it can be recognised by the application.

The application will retrieve information to the user screen according with the first four items recognised by google vision API, in case the user is not satisfied with the result given the option re-take image will appear on the screen.

The user will have an response from the application in case of failure or the image is not recognisable by the application.

## **2.1.5. Environmental requirements**

### *2.1.5.1 Android devices*

The application is build to run on android devices with basic application installation.

### *2.1.5.2 Android studio IDE*

Android studio is the development tool used for the application development where all the different API parts support the language, the IDE offers debug and suggestion for the functionality of the application including virtual devices which gives room to perform test in different android devices.

## **2.1.6. Usability requirements**

### *2.1.6.1 Performance*

The application should give the users the desired results in reasonable time fashion and the option of re-taking the image if they are not satisfied with the

current one. In case of an object is not identified, the application will produce an error message as results of the query.

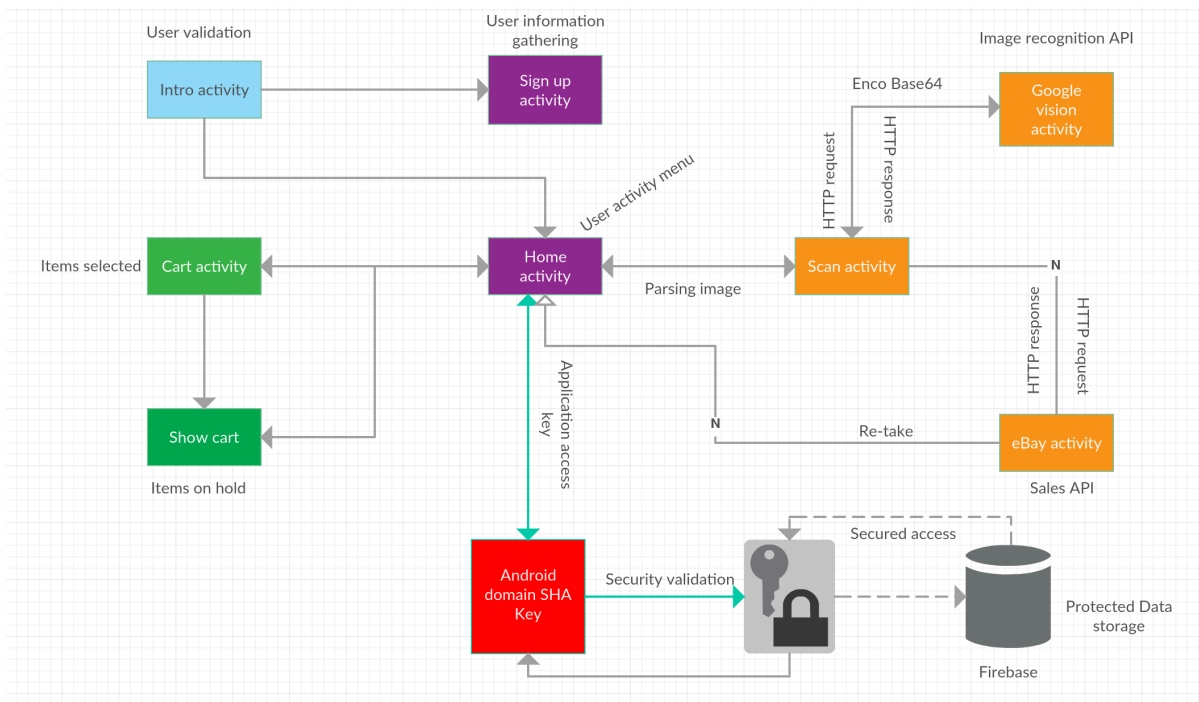
### 2.1.6.2 User interaction

The user interface home page is compound of few different icons, which have intuitive text that will navigate the user as it is described, no more than one functionality is given to a screen.

## 2.2. Design and Architecture

### 2.2.1 Activity flow diagram

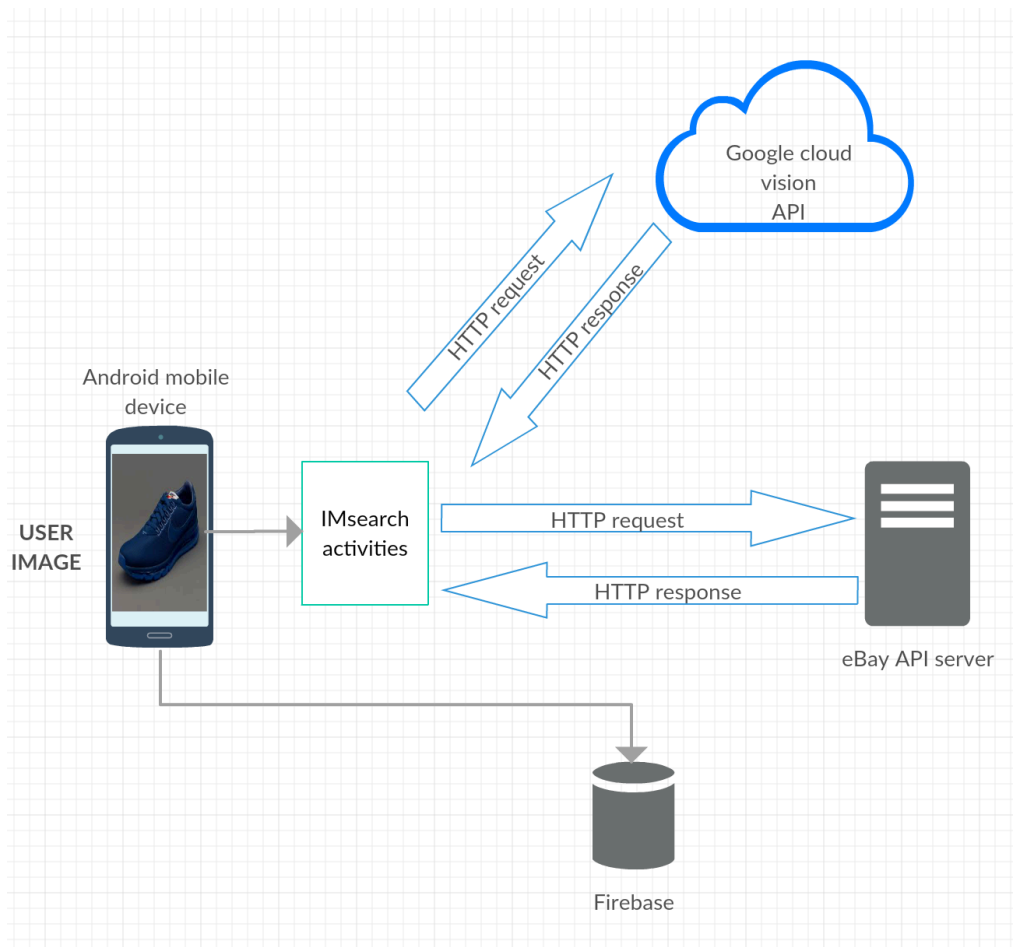
The above activity flow diagram illustrate the class that are responsible for the exchanging information to accomplish image search functionalities.



- The intro activity is the class responsible for checking validation of the user credentials in the database. In case the user is not able to log in the application class handles a warning message to the user.
- The sign up flow activity class is responsible for gathering the user's information mandatory needed for the registration process, respecting email format and password policy applied into the application sign up requirements.
- The home activity class is responsible for handling the users' information and options to navigate through the different application screen, all the possible activities are concentrate in home class of the application where the user can choose which item of the menu to go to.
- The Scan activity class is responsible for capturing the captured image and upload it to the google cloud vision API, the class provides the user with an error handler in case the image could not be upload or recognised by the third party component, as security feature in this section the application freezes the screen while the response is not complete in order to mitigate the possible faulty requests.
- The home activity class is responsible for handling the converted information extracted from the image uploaded as response from google cloud vision API, the results with the highest matching probabilities are sent to eBay servers for search query and then the results are forwarded to eBay Activity class which gives users an option to buy those items.
- The eBay activity class is responsible for handling the text extracted from google cloud vision activity class, the data is parsed into eBay API request to query the desired, returning the results to the user's application phone screen (Home activity).
- The cart class activity is responsible for holding the user potential purchases, in case the user wants to buy more than one product at once.

- The show cart activity class is responsible for display the selected products by the user while the purchase is not complete.
- Firebase as security feature offers SHA access key the key is installed in the application and the database will just accept request from the domain registered.

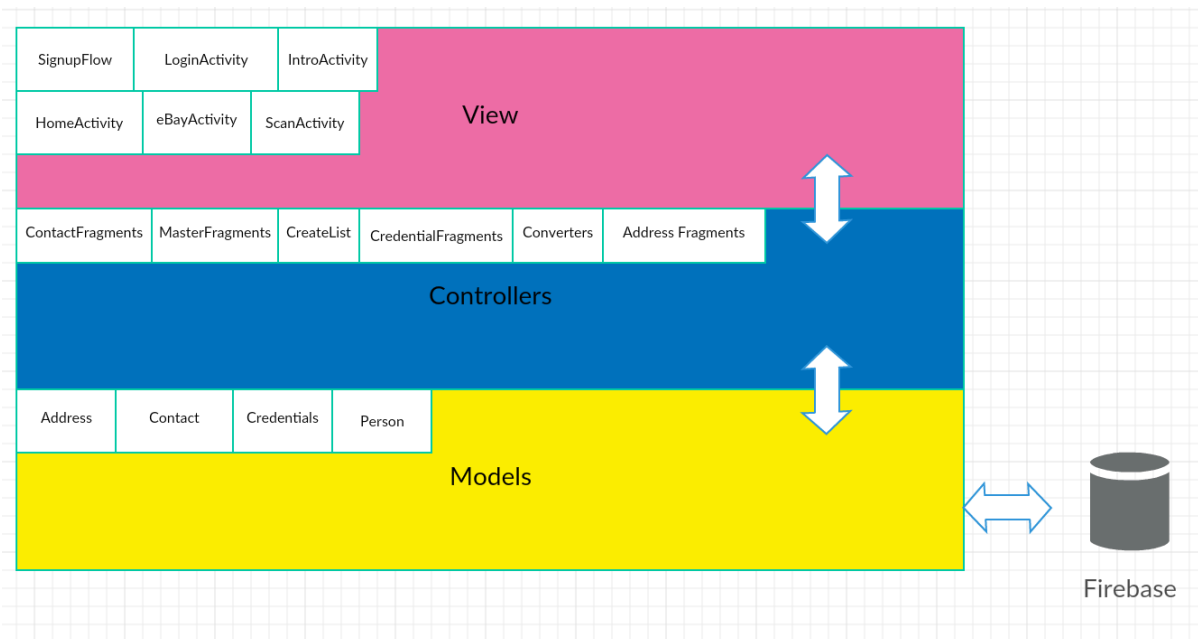
### 2.2.2 Hardware diagram



The hardware diagram above illustrates the requirements used in IMsearch development environment, mostly composed of API requests and responses on the backend side, both Google and eBay API documentation require an application ID and an access key to the services provided. The information is

secure between Android studio and the APIs as the use of an access key is mandatory.

### 2.2.3 System architecture diagram



The above diagram illustrate the system architecture for IMsearch. As the application requires two ways API to accomplish a search, firstly from google vision and after the data is extracted and sent to eBay API.

The view section holds all the activities that the user once logged in is able to perform.

The controller section ensures the data passed is actually the one meant to be to the different parts.

The model section is responsible for handling the validate user data to storage destination (Firebase).

### 2.3. Implementation

IMsearch was developed based on services providers and their products availability. The application is compounded of different classes that extract information from each of the services providers and results in a product search.

### 2.3.1 Scan upload activity

One of the crucial activity used in the application is to handle the user's image to the recognition API the image is stored temporarily in the database until the query is confirmed or cancelled, the google cloud vision API must be enable for this feature functionality.

```
public class ScanActivity extends AppCompatActivity {

    Button newScan;
    FirebaseStorage storage;
    StorageReference storageReference;
    DatabaseReference database;
    private String personId;

    private final String image_titles[] = {
        "Img1",
        "Img2",
        "Img3",
        "Img4",
        "Img5",
        "Img6",
        "Img7",
        "Img8"
    };

    private final Integer image_ids[] = {
        R.drawable.img1,
        R.drawable.img2,
        R.drawable.img3,
        R.drawable.img4,
        R.drawable.img5,
        R.drawable.img6,
        R.drawable.img7,
        R.drawable.img8
    };

    ArrayList<CreateList> scanImages;
    MyAdapter adapter;
    ProgressDialog progress;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_scan);
    }
}
```

```

        newScan = (Button)findViewById(R.id.new_scan);
        newScan.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                dispatchTakePictureIntent();
            }
        });

        RecyclerView recyclerView =
(RecyclerView)findViewById(R.id.imagegallery);
        recyclerView.setHasFixedSize(true);

        RecyclerView.LayoutManager layoutManager = new
GridLayoutManager(getApplicationContext(),3);
        recyclerView.setLayoutManager(layoutManager);

        scanImages = new ArrayList<>();

        adapter = new MyAdapter(getApplicationContext(),
scanImages);
        recyclerView.setAdapter(adapter);

        database =
FirebaseDatabase.getInstance().getReference();
        storage = FirebaseStorage.getInstance("gs://
medigocare-ab6aa.appspot.com/");

        Intent intent = getIntent();

        if(intent != null && intent.hasExtra("id")){
            personId = intent.getStringExtra("id");
            storageReference =
storage.getReference().child(personId);
            prepareData(personId);
        }
        private void prepareData(String personId){

            progress = new ProgressDialog(this);
            progress.setTitle("Loading");
            progress.setMessage("Wait while loading...");

```

```

        progress.setCancelable(false); // disable dismiss
        by tapping outside of the dialog
        progress.show();

        database.child("users").child(personId).child("scans").addL
        istenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot
            dataSnapshot) {

                Iterator<DataSnapshot> iterator =
                dataSnapshot.getChildren().iterator();

                while(iterator.hasNext()){
                    String imageName =
                    iterator.next().getKey();

                    downloadImage(imageName);

                    Log.d(TAG, imageName);
                }

                progress.dismiss();
            }

            @Override
            public void onCancelled(DatabaseError
            databaseError) {

            }

        });
    }

    static final int REQUEST_IMAGE_CAPTURE = 1;
    private String TAG = "Debugging";

    private void dispatchTakePictureIntent() {
        Intent takePictureIntent = new
        Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        if
        (takePictureIntent.resolveActivity(getPackageManager()) !=
        null) {
            startActivityForResult(takePictureIntent,
            REQUEST_IMAGE_CAPTURE);
        }
    }

```



```

    }

    @Override
    protected void onActivityResult(int requestCode, int
resultCode, Intent data) {
        if (requestCode == REQUEST_IMAGE_CAPTURE &&
resultCode == RESULT_OK) {
            Bundle extras = data.getExtras();
            Bitmap bitmap = (Bitmap) extras.get("data");

            ByteArrayOutputStream baos = new
ByteArrayOutputStream();
            bitmap.compress(Bitmap.CompressFormat.JPEG,
100, baos);
            byte[] bitmapData = baos.toByteArray();

            final String imageName =
String.valueOf(System.currentTimeMillis());

            UploadTask uploadTask =
storage.getReference().child(personId).child(imageName).put
Bytes(bitmapData);
            uploadTask.addOnFailureListener(new
OnFailureListener() {
                @Override
                public void onFailure(@NonNull Exception
exception) {
                    // Handle unsuccessful uploads
                    Log.d(TAG, "Image upload
failed"+exception.getLocalizedMessage());
                }
            }).addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
                @Override
                public void
onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
                    // taskSnapshot.getMetadata() contains
file metadata such as size, content-type, and download URL.

                    downloadImage(imageName);

                    database.child("users").child(personId).child("scans").chil
d(imageName).setValue(true).addOnCompleteListener(new
OnCompleteListener<Void>() {
                        @Override

```

```

        public void onComplete(@NonNull
Task<Void> task) {

            if(task.isSuccessful()){
                Log.d(TAG, "Image reference
stored successfully");
            }
        }
    });
}
});
}
}

private void downloadImage(String imageName){

storageReference.child(imageName).getBytes(Long.MAX_VALUE).
addOnSuccessListener(new OnSuccessListener<byte[]>() {
    @Override
    public void onSuccess(byte[] image) {
        // Use the bytes to display the image

        BitmapFactory.Options options = new
BitmapFactory.Options();
        Bitmap bitmap =
BitmapFactory.decodeByteArray(image, 0, image.length,
options);

        CreateList createList = new CreateList();
        createList.setImage_title("test");
        createList.setImage_ID(122);
        createList.setBitmap(bitmap);

        scanImages.add(createList);

        //refresh the adapter
        adapter.notifyDataSetChanged();

        if(progress.isShowing()){
            progress.dismiss();
        }

        Log.d(TAG, "image found");
    }
}).addOnFailureListener(new OnFailureListener() {
    @Override

```

```

        public void onFailure(@NonNull Exception
exception) {
            // Handle any errors
            Log.d(TAG, "image
"+exception.getLocalizedMessage());
        }
    });
}
}

```

### 2.3.2 eBay API activity

The eBay API activity class has the responsibility of handling the arrays of data extracted from the image recognition API in raw format, the query is handled to the sell API where the products are available.

```

public class EbayActivity extends AppCompatActivity {

    public static final String[] titles =new String[]    };
    public static final Integer[] images =
{ R.drawable.img1,
    R.drawable.img2, R.drawable.img3 };
    ListView list;
    List<RowData> rowDataAll;
    private String TAG = "Debugging";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_ebay);

        Intent intent = getIntent();
        String message =
intent.getStringExtra(HomeActivity.EXTRA_MESSAGE);

        rowDataAll =new ArrayList<RowData>();

        ArrayList<RowData> smallSet =
Converters.convertStringToRowData(message);

        rowDataAll.addAll(smallSet);
    }
}

```

```

        list = (ListView) findViewById(R.id.list);
        EbayAdapter adapter = new EbayAdapter(this,
rowDataAll);
        list.setAdapter(adapter);
    }
}

```

### 2.3.3 Sign up flow activity class

This class is responsible for the user's registration requirements, the text typed in the application is evaluated and checked according with the application restriction for certain fields such as password policy and unique registered email address, all the fields if are not filled correctly the application will warn the user of faulty information typed.

```

public class SignUpFlow extends AppCompatActivity
implements
    CredentialFragment.CredentialFragmentInteractionListener,
    ContactFragment.ContactFragmentInteractionListener,
    AddressFragment.AddressFragmentInteractionListener,
    View.OnClickListener {

    private static final int NUM_PAGES = 3;
    private ViewPager mPager;
    private PagerAdapter mPagerAdapter;
    private String TAG = "Debugging";
    private static HashMap<Integer, MasterFragment>
fragmentsGroup = new HashMap<>();
    int currentPageNumber = 0;
    private Button nextButton, previousButton,
registerButton;
    private Credential credential;
    private Contact contact;
    private Address address;
    private FirebaseAuth mAuth;
    private DatabaseReference database;

    static {
        fragmentsGroup.put(0,
CredentialFragment.newInstance());
    }
}

```

```

        fragmentsGroup.put(1,
ContactFragment.newInstance());
        fragmentsGroup.put(2,
AddressFragment.newInstance());
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sign_up_flow);

        previousButton = (Button)
findViewById(R.id.previous);
        registerButton = (Button)
findViewById(R.id.register);
        nextButton = (Button) findViewById(R.id.next);

        previousButton.setOnClickListener(this);
        registerButton.setOnClickListener(this);
        nextButton.setOnClickListener(this);

        // Instantiate a ViewPager and a PagerAdapter.
        mPager = (ViewPager) findViewById(R.id.pager);
        mPager.setOnTouchListener(new
View.OnTouchListener()
        {
            @Override
            public boolean onTouch(View v, MotionEvent
event)
            {
                return true;
            }
        });
        mPagerAdapter = new
ScreenSlidePagerAdapter(getSupportFragmentManager());
        mPager.setPageTransformer(true, new
ZoomOutPageTransformer());
        mPager.setAdapter(mPagerAdapter);
    }

    @Override
    protected void onStart() {
        super.onStart();

        mAuth = FirebaseAuth.getInstance();

```

```

        database =
FirebaseDatabase.getInstance().getReference();
    }

    @Override
    public void onClick(View v) {

        currentPageNumber = mPager.getCurrentItem();

        if(v.getId() == R.id.previous){

            if(currentPageNumber == 1){

previousButton.setVisibility(View.INVISIBLE);
            }
            else if(currentPageNumber == 2){
                nextButton.setVisibility(View.VISIBLE);

registerButton.setVisibility(View.INVISIBLE);
            }
            currentPageNumber = (currentPageNumber - 1)%3;
        }
        else if(v.getId() == R.id.register){

            if(getCurrentFragment().isAllFieldsPopulated())
            {

            }

        }
        else if(v.getId() == R.id.next){

            if(getCurrentFragment().isAllFieldsPopulated())
            {

                if(currentPageNumber == 0){

previousButton.setVisibility(View.VISIBLE);
                }
                else if(currentPageNumber == 1){

nextButton.setVisibility(View.INVISIBLE);

registerButton.setVisibility(View.VISIBLE);
                }
                currentPageNumber = (currentPageNumber +
1)%3;
            }
        }
    }
}

```

```

    }

    mPager.setCurrentItem(currentPageNumber);
}

private MasterFragment getCurrentFragment(){

    return
SignUpFlow.fragmentsGroup.get(mPager.getCurrentItem());
}

@Override
public void onFragmentInteraction(Credential
credential) {
    this.credential = credential;

    if(credential != null){
        Log.d(TAG, "Email "+credential.getEmail());
        Log.d(TAG, "Password
"+credential.getPassword());
    }
}

@Override
public void onFragmentInteraction(Contact contact) {
    this.contact = contact;

    if(contact != null){
        Log.d(TAG, "First name
"+contact.getFirstName());
        Log.d(TAG, "Last name "+contact.getLastName());
        Log.d(TAG, "Phone Number "+contact.getPhone());
    }
}

@Override
public void onFragmentInteraction(Address address) {
    this.address = address;

    final Person person = new Person(credential,
contact, address);

    registerUser(person);

    if(address != null){
        Log.d(TAG, "House "+address.getHouse());
    }
}

```

```

        Log.d(TAG, "Street "+address.getStreet());
        Log.d(TAG, "City "+address.getCity());
        Log.d(TAG, "County"+address.getCounty());
        Log.d(TAG, "Country"+address.getCountry());
    }
}

public void registerUser(final Person person){

    final ProgressDialog progress = new
ProgressDialog(this);
    progress.setTitle("Loading");
    progress.setMessage("Wait while loading...");
    progress.setCancelable(false); // disable dismiss
by tapping outside of the dialog
    progress.show();

    mAuth.createUserWithEmailAndPassword(person.getCredential()
.getEmail(),
person.getCredential().getPassword()).addOnCompleteListener
(this, new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull
Task<AuthResult> task) {

            if (task != null && task.isSuccessful()) {
                Log.d(TAG,
"createUserWithEmail:success");
                FirebaseUser user =
mAuth.getCurrentUser();

                person.setId(user.getUid());

                HashMap<String, Object> dic = new
HashMap<String, Object>();
                dic.put("user", person);

                database.child("users").child(user.getUid()).updateChildren
(dic, new DatabaseReference.CompletionListener() {
                    @Override
                    public void
onComplete(DatabaseError databaseError, DatabaseReference
databaseReference) {

```



```

        if(databaseError != null){
            Log.d(TAG, "Some error
"+databaseError.getDetails());
        }
        else {
            Log.d(TAG, "User created");

            progress.dismiss();

            Intent intent = new
Intent(getApplicationContext(), HomeActivity.class);
            startActivity(intent);
        }

    });

    } else {
        Log.w(TAG,
"createUserWithEmail:failure", task.getException());
        Toast.makeText(SignUpFlow.this,
"Authentication failed.",
            Toast.LENGTH_SHORT).show();
    }

    });
}

private class ScreenSlidePagerAdapter extends
FragmentStatePagerAdapter {
    public ScreenSlidePagerAdapter(FragmentManager fm)
    {
        super(fm);
    }

    @Override
    public Fragment getItem(int position) {
        return SignUpFlow.fragmentsGroup.get(position);
    }

    @Override
    public int getCount() {
        return NUM_PAGES;
    }
}

```

```
    }
}
```

```

    public class ZoomOutPageTransformer implements
ViewPager.PageTransformer {
    private static final float MIN_SCALE = 0.85f;
    private static final float MIN_ALPHA = 0.5f;

    public void transformPage(View view, float
position) {
        int pageWidth = view.getWidth();
        int pageHeight = view.getHeight();

        if (position < -1) { // [-Infinity,-1)
            // This page is way off-screen to the left.
            view.setAlpha(0);

        } else if (position <= 1) { // [-1,1]
            // Modify the default slide transition to
            shrink the page as well
            float scaleFactor = Math.max(MIN_SCALE, 1 -
Math.abs(position));
            float vertMargin = pageHeight * (1 -
scaleFactor) / 2;
            float horzMargin = pageWidth * (1 -
scaleFactor) / 2;
            if (position < 0) {
                view.setTranslationX(horzMargin -
vertMargin / 2);
            } else {
                view.setTranslationX(-horzMargin +
vertMargin / 2);
            }

            // Scale the page down (between MIN_SCALE
and 1)

            view.setScaleX(scaleFactor);
            view.setScaleY(scaleFactor);

            // Fade the page relative to its size.
            view.setAlpha(MIN_ALPHA +
                (scaleFactor - MIN_SCALE) /
                (1 - MIN_SCALE) * (1 -
MIN_ALPHA));

        } else { // (1,+Infinity]

```

```

        // This page is way off-screen to the
right.
        view.setAlpha(0);
    }
}

public Person test(){

    Credential credential = new Credential();
    credential.setEmail("rafaelalves9@gmail.com");
    credential.setPassword("imsearch34");

    Contact contact = new Contact();
    contact.setFirstName("rafael");
    contact.setLastName("alves");
    contact.setPhone("9029202020202");

    Address address = new Address();
    address.setHouse("60");
    address.setStreet("hebert");
    address.setCity("hebert lane");
    address.setCounty("dublin");
    address.setCountry("ireland");

    Person person = new Person(credential, contact,
address);

    //person.setId("ddddwww");

    return person;

}
}

```

#### 2.3.4 Converters class

The converters class is responsible for the machine communication relating the different parts of the application API, using JSON format and HTTP to exchange information.

```

/**
 * Created by Rafael Alves on 08/04/2018.
 */

```

```

public class Converters {

    public static ArrayList<RowData>
    convertStringToRowData(String s){

        ArrayList<RowData> rowDataArrayList = new
        ArrayList<>();

        try {
            JSONObject masterJson = new
            JSONObject(s).getJSONArray("findItemsByKeywordsResponse").g
            etJSONObject(0);

            JSONArray searchResult =
            masterJson.getJSONArray("searchResult");

            JSONArray items =
            searchResult.getJSONObject(0).getJSONArray("item");

            for(int i=0; i < 10; i++){

                JSONObject jsonObject =
                items.getJSONObject(i);

                RowData rowData = new RowData();

                if(jsonObject.has("title"))
                rowData.setTitle(jsonObject.getJSONArray("title").getSring
                (0));

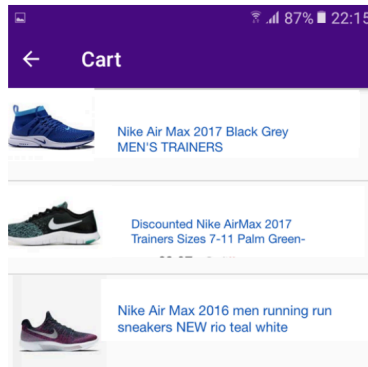
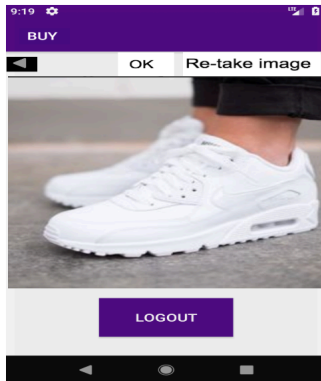
                if(jsonObject.has("subtitle"))
                rowData.setSubtitle(jsonObject.getJSONArray("subtitle").get
                String(0));

                if(jsonObject.has("galleryURL"))
                rowData.setImageUrl(jsonObject.getJSONArray("galleryURL").g
                etString(0));

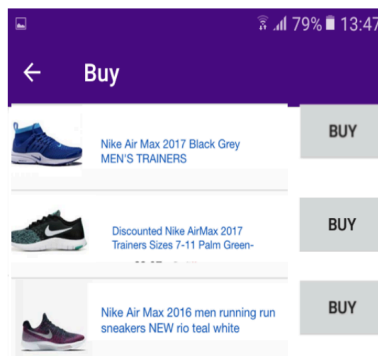
                rowDataArrayList.add(rowData);
            }

        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
}

```



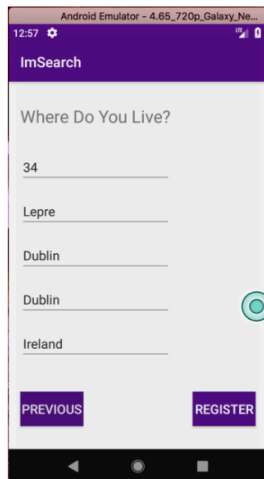
return



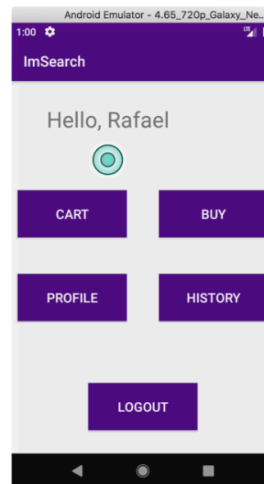
Email and password registration step



Contact information registration step



Address information registration step



User registered logged in

rowDataArrayList;

}

## ***2.4. Graphical User Interface (GUI) Layout***

The above table of screenshot illustrate the registration action that every user needs to go through to be able to use IMsearch services. Once the user is registered and logged in, the option of “BUY” appears on the menu and once selected, it opens the device camera enabling the user to capture the image desired and start the product search, the user is able to capture the image and automatically is redirected to the eBay products available querying the product according with the request from google parsed into the API.

## **2.5. Testing**

The application test was based on the image recognition of objects and the data extracted results and timely response from google cloud vision and eBay platform. The purpose of the test was to guarantee that the user will use the full functionality of all the services available combined, the response time of all the API are working in acceptable time manner max of 20 seconds to respond to an image search request.

Android studio updates and performance enhancement of the application was used to increased the performance and connection with the different parts of the application.

Internet connection bandwidth has an impact on performance directly related to the application as the software relies on upload of image and response from API. To better usage of IMsearch, the user is recommended to be connected to a reliable and rapid internet connection.

The user personal data registered is held in is kept encrypted in firebase database. Firebase configuration is set to accept, only request from IMsearch

application key, increasing the level of security and access to sensitive information.

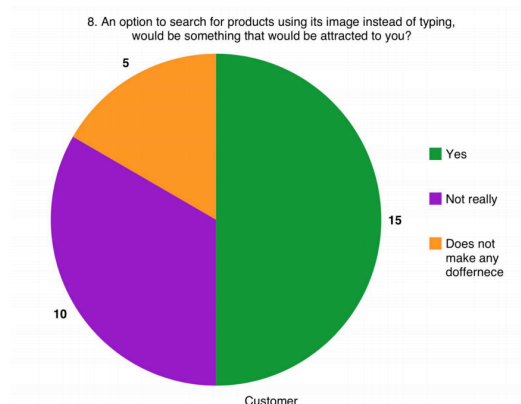
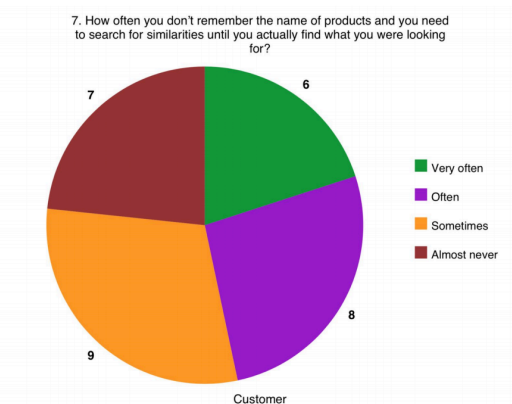
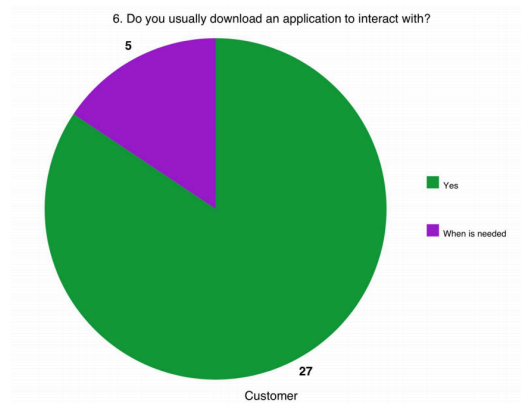
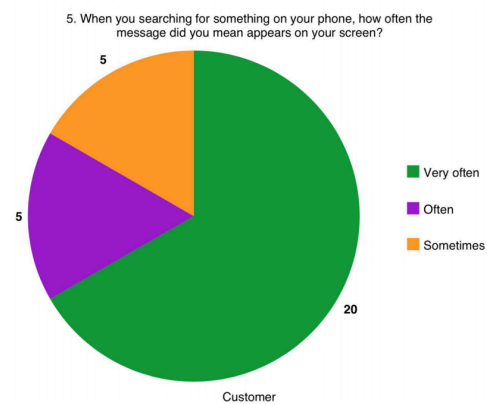
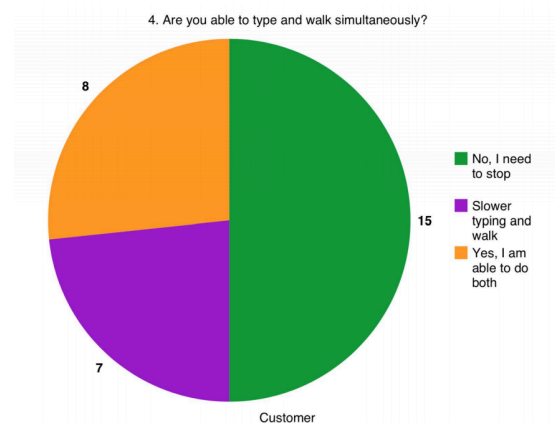
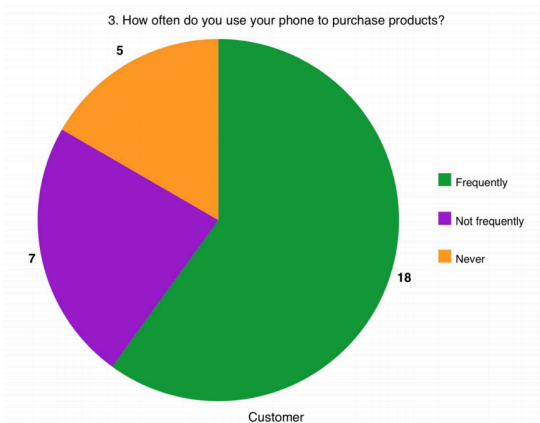
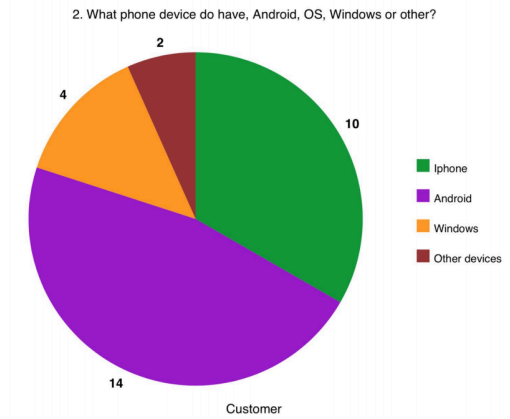
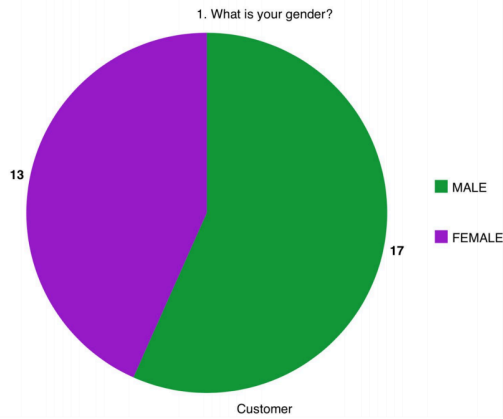
The credentials are accepted only if typed right by the user, an email standardization is set to avoid mistyping.

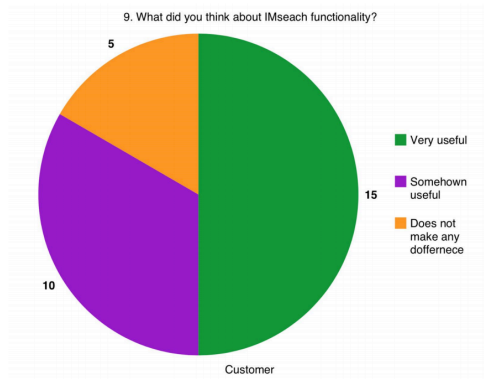
## ***2.6. Customer testing***

The testing with customer did not require any input of personal data rather it focus in the application functionality, a dummy user was created to perform the test to all the users. The number of people able to make part of the application is 30, from different age groups and genders, which had full access to the GUI and were able to navigated to every page. The following questionnaire was asked to the user after the test being finished:



# IMsearch questionnaire





## 2.7. Evaluation

The IMsearch system working functionalities evaluation was performed on mobile Galaxy premium android device and customers, the application was executed successfully in this device and the connection with the different parts of the application worked well as the API and registration keys are registered and validated. The customer feedback also contributed to the evaluation of the system and its usability when interacts with users, the major issue found in the system was the fact that image recognition yet is a new technology and some of the results were not found correctly when the image was not very clear, however, the positive feedback was the scope of possible situation that the application can be placed, when for some reason the user cannot type or actually does not remember the name of the object, in fact, the user can select image from its gallery and search for it.

### **3. Conclusions**

Combining well established services with new technologies such as cloud computing storage and image recognition API, aiming to provide to the users an alternative way to experience online shopping retrieving products available in e-commerce platform.

In fact image recognition is a new technology, the number of application that are going to be developed around it is not limited, how to identify products better based on their image is yet one feature that this application has path and need of further development.

The range of opportunities is large, in fact the coming years promise to be benefit in image recognition oriented applications, another variable that is critical for this application is the hardware capacity available using cloud computing, where in other scenarios this application development would not be possible using ordinary hardware.

The limitation of the application is directly proportional with the image recognition technologies, the more accurate the object recognition the more accurate is the results.

#### **4. Further development or research**

The IMsearch project idea is to provide the user an entirely new online shopping experience where the user can take advantage of search and shopping automation, implementing a payment API and a login API are the next development stage, from the registration process until the purchase every action would be automated, giving the user more time to spend with the application functionality browsing experience itself.

The API payment functionality is a feature that would provide the user another automated service, the application has as one of its aims to avoid typing on the phone screen, the usage of voice recognition to bring the image to the phone gallery would be another advantage to the user.

## 5. References

Google Cloud. (2018). *Building an image search application using Cloud Vision API | Google Cloud Big Data and Machine Learning Blog | Google Cloud*. [online] Available at: <https://cloud.google.com/blog/big-data/2018/05/building-an-image-search-application-using-cloud-vision-api>. [Accessed Dec 2017].

Google Cloud. (2018). *Build an Android App Using Firebase and the App Engine Flexible Environment | Solutions | Google Cloud*. [online] Available at: <https://cloud.google.com/solutions/mobile/mobile-firebase-app-engine-flexible>. [Accessed Jan 2018].

Go.developer.ebay.com. (2018). *eBay Developers Program*. [online] Available at: <https://go.developer.ebay.com>. [Accessed Feb 2018].

freeCodeCamp. (2018). *What is an API? In English, please. – freeCodeCamp*. [online] Available at: <https://medium.freecodecamp.org/what-is-an-api-in-english-please-b880a3214a82> [Accessed Feb 2018].

www.kpcb.com, K. (2018). *2017 Internet Trends Report*. [online] Kpcb.com. Available at: <http://www.kpcb.com/internet-trends> [Accessed Mar 2018].

Smart Insights. (2018). *Mobile marketing statistics 2018*. [online] Available at: <https://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/> [Accessed Mar 2018].

SearchWinDevelopment. (2018). *What is JSON (Javascript Object Notation)? - Definition from WhatIs.com*. [online] Available at: <https://searchwinddevelopment.techtarget.com/definition/JSON-Javascript-Object-Notation> [Accessed Apr 2018].

Webopedia.com. (2018). *What is HTTP - HyperText Transfer Protocol? Webopedia Definition*. [online] Available at: <https://www.webopedia.com/TERM/H/HTTP.html> [Accessed Apr 2018].

Help.github.com. (2018). *GitHub Glossary - User Documentation*. [online] Available at: <https://help.github.com/articles/github-glossary/> [Accessed Apr 2018].

## **5.1. Monthly Journals**

### **• Journal September 2017**

Finally fourth year, Super glad I made up to this point. Now was pretty much everything that we have got up to this point would resume in a project that actually nobody knows if it will have any usability, that is the challenge, with a loads of energy and imagination my project proposal was to apply statistics to football matches and try to predict results based on previous games behaviour. That is it, write the whole idea in a paper, find the lecture that was assigned to me, go into a room stand, talk about your idea, listen to comments and ideas regarding your project, I started using grammar as a writing aid. The security part of my application was not yet clear due the fact my idea was not well defined by me.

### **• Journal October 2017**

After my first idea been “turned down” actually in parts i agreed with the comments, the application would take a lot of different parts to compound the result and yet it would need another sort of evaluation to conclude that the functionality was valid, from that point I decided to take another approach where I would simplify the functionality of my application and yet be useful. Multiple applications do the same job, however, with different details where gives users room to decide what the would use based on their preferences. Based on that The idea of using image recognition to e-commerce started to build in as idea for my final year project.

### **• Journal November 2017**

At this stage I was sure that I wanted to develop something that is not quite clear and precise defined in the market, however, has a big potential, machine image recognition for example, in fact, the specialisation available in my course was not the one that I choose to be my first, however the number of students in my class optioned to go for security or cloud when I had chosen big data, which brought me to another scenario where I had to apply cyber security principals in order to develop my application in all aspect of SDLC.

### **• Journal December 2017**

At this stage I was about to consolidate my idea in fact I had too many things happening at the same including exams and delivery of final year project proposal, which in my case I did not do so well on my first presentation, by the fact that I believe the course is not designed for part-time student, working full

time. My second idea was finally cleared by the time of the second presentation in fact what you possibly imagine that happens in your application until you start developing is totally different. So finally I came down to the point where I wanted to develop an application that uses image recognition to search products, it was made possible by the fact that Google released an API where the image is upload to google and the results are displayed. With that feature in mind, I had the idea of putting these technologies together instead of creating a whole new environment.

### • ***Journal January 2018***

Yes it was new year already and by this time I had bits and pieces and yet nothing working together, the amount of information and the dots that I needed to put together were quite challenging for a undergraduate student, my technologies and functionality were way clearer in my had and finally I got to a point where my application gained at least a pseudo shape. An android application that users can register and search for products using eBay platform combined with image recognition, there is not much of material out there helping with this kind of technology which brought me to another challenge, followed by hours of tutorial misinterpretation of documentation and any other thing that could have gone wrong. The project gave me the experience that, there is no way in SDLC that if don't know what you want you will get eventually.

### • ***Journal February 2018***

Time is becoming something really short by this stage of development, work, college work, and personal life were not well balanced at all. After days drawing how the application should work and how people would use the application functionality and why they would use it. I decided to start doing some practical. Evaluate what would be the best options available at the moment and how they would help me from a job market perspective. I decided to use cloud technologies to host my application which would make easier the interactivity with other components of the application.

Until this point, I could say that I was going to use google cloud technologies in two-way image recognition and cloud instance storage.

### • ***Journal March 2018***

eBay, this part of my project gave me one of the biggest headaches ever, is not really well supported the API technologies, especially in Java (Android studio) I also figured out that Python would've done the job much easier. I thought of changing the technologies but I decided not to because it would result in other problems that I would face eventually. After weeks going around the same problem, I finally got my first API request right, in fact, I was just happy that a red message did not come up in my screen, also figured out that a lot of information comes into the eBay API request. Now was time to figured out how to clean the

search. On the google, part was a little bit easier, in fact, you type it on youtube or any another IT development support community there are loads of material out there supporting such technology.

- ***Journal April 2018***

Now I was 1 month away from the due date, did not have much time to meet my lecturer to discuss my project if I could I believe small details, and advice would be a fit and probably things would have gone smoother. I started looking at previous years projects and check what they were writing in their journal, sincerely I did not find much of transparency. Everyone seemed to have control of what was happening in their project. I was not actually sure how I was in mine if you are using mine as a reference to yours believe I almost went crazy with all these things happening. By now I had google and eBay API working, but yet not together, As solution to this part I had to find a way to extract the text found by the google search and pass the value to eBay filtered search, The implementation of others API were in the initial idea however with the amount of time that I had available to work on the project and other college duties I would not be able to accomplish my entire intention. Initially, MySQL was the chosen database for my application, however, placing everything on the cloud would make things easier, that is when firebase came into play and it would also include the security measures need to my application such as CIA and encryption. Frameworks are the first thing that the majority things of, how is it going to look like, again against my student mind I had to fight them too much imagination and complexity, looking into designed for very well used application and check their navigation structure to replicate something that the users would be comfortable with when using mine. Yes, they pretty much look the same. Register page, Login, a profile and the functionality of the application.

That's what I did, in fact, what is new in my application is the facility of using the image to do something differently that is been done over the last 20 years using internet by typing the name and find the product.

- ***Journal May 2018***

The month of deliverables for the project, which is mostly spent in customer research, surveys and poster development. IMsearch has the main functionality responsive to the the user request and the registration process working well, Firebase has the SHA implemented to guarantee that only the "IMsearch.com" android studio project domain has access and permission to the stored data, for security purposes, The main functionality of the application relies in another API to parse the data extracted from the image details which can become an inconvenient if the image is not clear enough to be recognised properly. The project has given me as a student an idea of how big are scope of details in order to develop an application, from the idea formation up to the point where actually



something is built and translated into services. An important details to mention is the fact that when you are building an project specially when you are a student, all the goals must be clear otherwise it will become way more complicated to develop something that is not cleared interpreted by the one who is developing.

## 6. Appendix

### 6.1. Project Proposal

---

**Rafael Alves**

x16109716

BSc (Honours) in Computing Evening part-time

Specialisation: Cyber security

## IMsearch October 2017

### ***Executive summary***

The evolution of technologies in image recognition has grown in the last 5 years reaching a point where anyone can have access to tools such as clarifai and google cloud vision using API's to bring new image features into applications.

My goal in this project is to develop an application where the users can search products based on the image recognition data processed such as logo, labels and any kind of useful data generated from the image recognition analysis.

The application will be developed in a intuitive way where the users will seldom find difficulty in use it to perform a search aiming also groups that are not able to adapt to touch screen (coordination movements disorders) an image is easier to be recognised if is analysed, for instance someone like a jacket or bag but is not sure what is the brand if a picture is taken and its been recognised before it will quickly answer these questions.

As image recognition is a very wide and new field this project will be developed in Market search where the words and characteristic of a product will be extracted by an picture and forward to e-commerce site also using its API and fetch the most accurate found result

The implementation of location API will be a input in the project where the person also will be answered where such object is for sale based on their location.

## **Background**

As the number of mobile devices and image technologies has grown in the last decade exponentially, according with kpcb reports the number of pictures taken a day reaches over 1.8 everyday, one thing that is not yet possible is to predict what is been taken picture of, however is possible to identify objects based in theirs characteristics which leads the market to explore new products and facilities.

The motive behind this project is the amount of image data that is generate a day, every person carrying a camera mobile device has the potential to create image data. Sourced by millions of devices daily image recognition feature has gained a wide space in market in 2017 specially, reaching a level of availability of services where developers can use API in order to create new software application related to image. Available in google APIs, the evolution and use of advanced image recognition technologies to explore this field and create new products to users is a niche to explore, with powerful image recognition technologies a new market has opened in several industry that can use it productively.

## **1.2 Idea**

The aim of this project is with a simple application that does not require much technical knowledge provide user the facility of having an optional way to search for products based on their image. Adding a new experience to customers in daily basis task with phone based on images recognition where instead of typing the desired product the user would just take a picture of it and the search would be made based on the objects identified in the picture, it also can be a easier way to make search for people that suffer from movements disorder, or people that have writing difficulties.

Develop an user friendly android application to search products in few steps. The mobile application will have a signup and login page which will also contain the option to use other networks apis in order to log in, once the user is logged in it will be redirected to the take a picture page where the option of retain the image information in case the user wants to use the app for personal purpose is also to be included.

The application itself will not generate any cost as it will entirely be developed by me. The use of open sources programs will be applied whenever is necessary in order to not generate costs.

## **1.3 Technologies**

The android application is going to be developed using several different tools combined to produce the end service specifically Google cloud instances, Google API, Java, Python, CSS and javascript.

Using SQL database to keep crypto-graphed users' information as we are dealing with sensitive information.

In the login side a two way authentication will be implemented to improve the user data protection.

Android studio is a very handy development toolkit being flexible to the operational system it is being used. The application will have multiple APIs working simultaneously XML, JSON, HTML.

## **2 System**

### **2.1 Requirements**

The reason why requirements are the first thing that is put in check when it comes to software development is because it must have a purpose, an aimed market, in my case people that are not comfortable with writing or rather have more information of a picture than actually could identify.

Speaking with a number of students, workmates and random people asked if they could search and buy a product based on the picture instead of typing. Interesting was shown from majority of asked people.

### **2.1.1 Functional requirements**

An android mobile phone, login and registration will be the basic requirements to use the application. To better manager security the app will have a password policy where the composition of the password will be explain to the users (Length and characters).

#### **Functionality description**

The users are going to be registered and their data will be kept in safe storage and be used when desire to log in. As the app will also hold implementation of some of the networks logins API the user easily can log in.

#### **Precondition**

Download the application from google store and use one of the options for registration.

#### **Termination**

The user will have the option to log off as them wish.

#### **Post condition**

As the user is registered in the database their profile will enter in a wait mode until next action from their side.

#### **Camera view**

The camera will be the main feature to be used in this application as it will the data input to the users, the picture will be held in SQL and it will interact with user and application as soon it is taken.

#### **Map implementation**

One of the goals of this project is simplify the user experience when it comes to use an android mobile application, implementation the function of showing the location that potentially have the products in sales.

### **2.1.2 Data Requirements**

- **SQL requirements:** In this project the function of the database is to save encrypted user's information and pictures that will be stored in a cloud server where the API will have access.
- **APIS requirements:** The project will run eBay API, Google cloud vision API, google maps API.

### 2.1.3 User requirements

The application will be user friendly narrowing the requirements on the user side.

- **Internet access:** As the application engineer is built in google cloud internet access is needed in order to use the application
- An android mobile phone is must as it is the platform that the application is going to be developed.

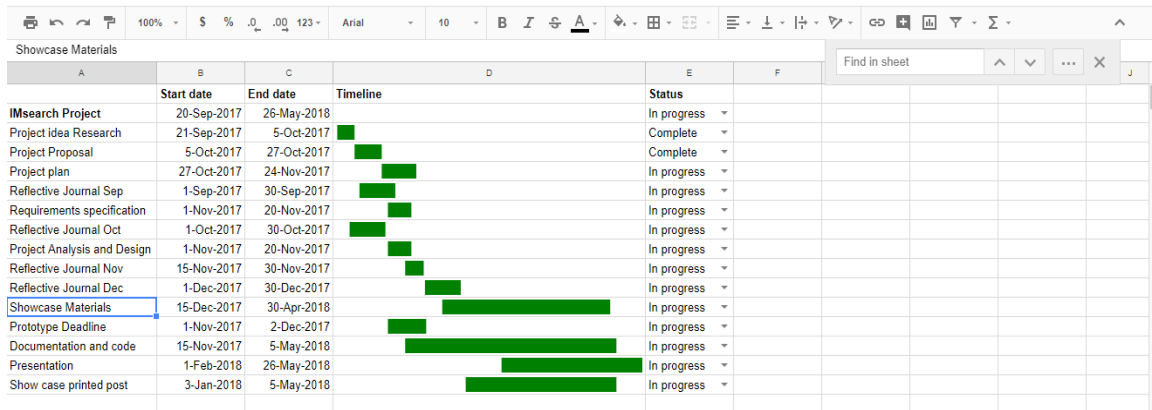
### 2.1.4 Environment requirements

- **OS:** The application is going to be developed in an instance of google cloud holding Linux 16.04 and over 7.5 Gb of ram memory it is scalable if increase is necessary.
- **Internet access:** Internet is a must for this project.

### 2.1.5 Usability requirements

- **Understandable:** The application will be user friendly, clear and intuitive instructions.
- **Operable:** The application must bring back information that the images have, in case of failure in the search the user will be notified
- **Interaction:** Nicely design to make the user feel comfortable when is navigating through the application with few commands able to achieve the goal of the search.

## Gantt chart



## 2.1.6 Competitor's application at google store



### ShotShop : shot a product and find cheaper

ShotShop lets you find the cheapest online store that is selling the product you want via a simple screenshot.

it is useful for finding a less price for a product you came accross on facebook , instagram , pinterest or whatsapp ... and not paying the extra fees for all these marketers around.

## Requirements

### Functional requirements.

Use case diagram.





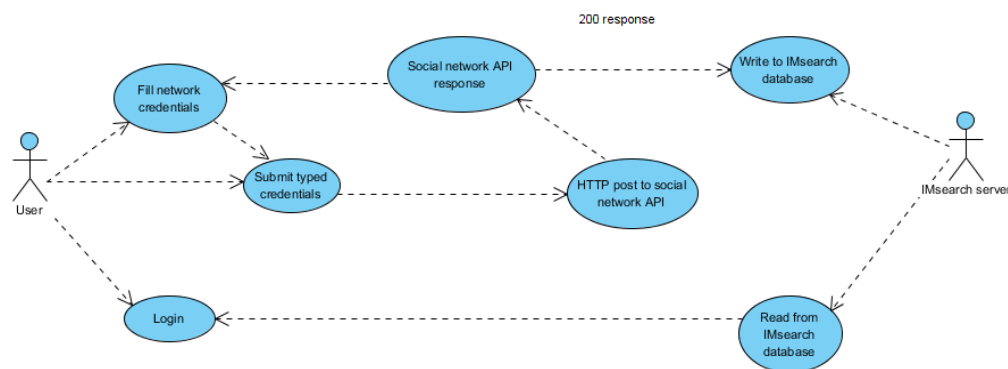
## **Scope**

The scope for this use case is to request a response from social networks APIs to register the user in case the response is successful.

## **Description**

The user case below illustrate the action of the user interacting with the system in order to create a new account using already validated credentials from existing social networks accounts.

## **Use case diagram:**



## **Requirement 2 <Product image recognition>:**

The image recognition function is the most important in IMsearch application. This part of the program will allow the user to point the mobile phone to a product that they want to buy or as option go to their phone library to upload a photo of a product that they are interested in. The google cloud vision API will receive a image post request from the application, the result will be analysed with pertinent features of products such as brand, text found and object recognition, in case the accuracy is not enough to perform search the user will be notified.

**Priority: High**

**Use-case**

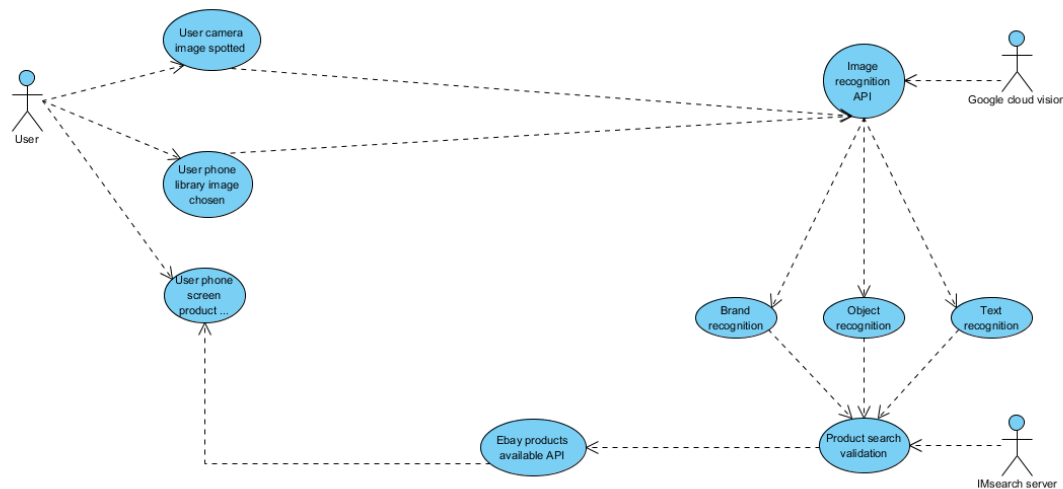
## **Scope**

The scope for this use case is to request a response from google cloud vision API with pertinent information to search for products having or not a successful response from the APIs services.

### **Description**

The user case below illustrates the action of using an image to search for products in by the phone library or taking pictures.

### **Use case diagram:**



---

### **Requirement 3<User profile>:**

#### **Description - High priority**

The profile functionality allows the user to edit personal information such name, payment details, search results and shopping records. A GET request will be sent to the server in order to fetch user profile information. The application will send a POST request to the database with the new information given from the user response will be displayed to aware the user the result of the action.

#### **Scope**

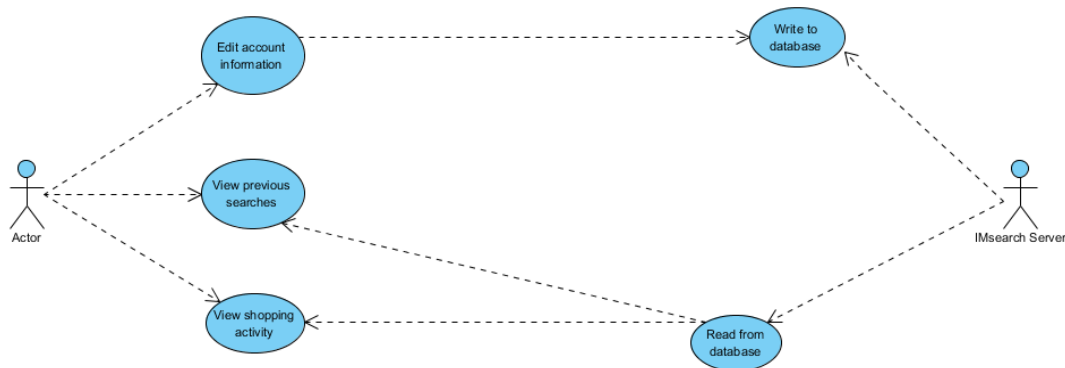
The scope of this use case is to receive the login API response from network and validate it fetching information from the database and fill the user information on the phone screen using a GET request.

### **Description**

The below use case describes the function of the users to edit their personal profile information once their login information is validated.

### **Scope**

The scope of this use case is to send to the server a POST request with updated profile information, searches made by the user and check shopping activity.



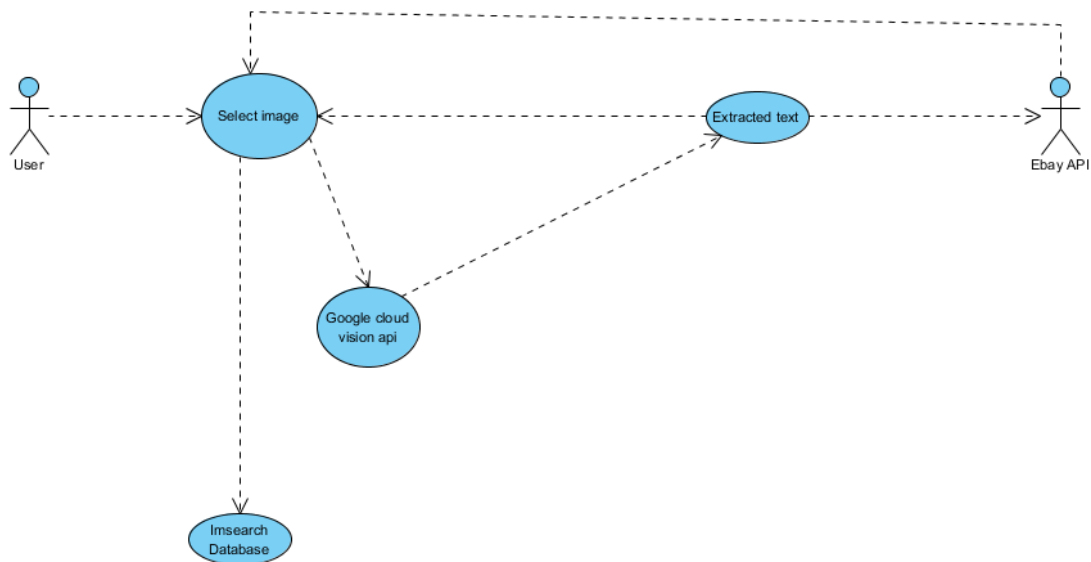
### **Requirement 4<Select product>:**

#### **Description - High priority**

The select product functionality allows the user to choose the found product from the application search. This section of the program will use the eBay API using keywords extracted from google cloud vision to run the search, once the application display the found product the user will be able to select and proceed with the purchase.

#### **Scope**

The scope of this use case is allow the user to select the product based on the API results, a GET request will be sent to the eBay API to perform the search.



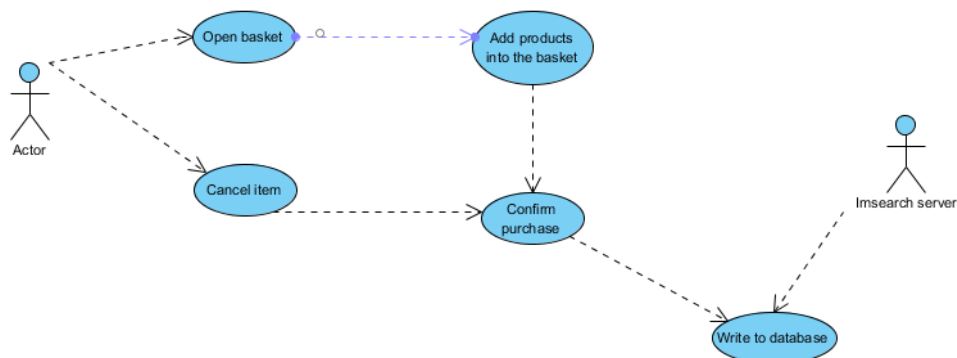
#### Requirement 4<Add to basket>:

##### Description - Medium priority

This functionality of the application will allow the user to add the selected item to a basket and or cancel a product if they wish.

##### Scope


The scope of this functionality is to allow the user to select more than one product or cancel an item if is needed and confirm the purchase.



## 6.2. Project Plan

Showcase Materials						Find in sheet					
A	B	C	D	E	F						
	Start date	End date	Timeline	Status							
IMsearch Project	20-Sep-2017	26-May-2018		In progress							
Project idea Research	21-Sep-2017	5-Oct-2017		Complete							
Project Proposal	5-Oct-2017	27-Oct-2017		Complete							
Project plan	27-Oct-2017	24-Nov-2017		In progress							
Reflective Journal Sep	1-Sep-2017	30-Sep-2017		In progress							
Requirements specification	1-Nov-2017	20-Nov-2017		In progress							
Reflective Journal Oct	1-Oct-2017	30-Oct-2017		In progress							
Project Analysis and Design	1-Nov-2017	20-Nov-2017		In progress							
Reflective Journal Nov	15-Nov-2017	30-Nov-2017		In progress							
Reflective Journal Dec	1-Dec-2017	30-Dec-2017		In progress							
Showcase Materials	15-Dec-2017	30-Apr-2018		In progress							
Prototype Deadline	1-Nov-2017	2-Dec-2017		In progress							
Documentation and code	15-Nov-2017	5-May-2018		In progress							
Presentation	1-Feb-2018	26-May-2018		In progress							
Show case printed post	3-Jan-2018	5-May-2018		In progress							

## 6.3. Other Material Used



IMsearch

1. What is your gender?

☐ Male

☐ Female

2. What phone device do have, Android, OS, Windows or other?

☐ Android

☐ OS (Iphone)

☐ Windows

☐ Other

3. How often do you use your phone to purchase products?

☐ Very often

☐ Often

☐ Sometimes

☐ Never

4. Are you able to type and walk simultaneously?

☐ Yes, I am able to walk and type on my phone screen

☐ Yes, but it slows sown my walking and typing

☐ No way

5. When you searching for something on your phone, how often the message did you mean appears on your screen?

- ☐ Very often
- ☐ Often
- ☐ Sometimes
- ☐ Hardly

6. Do you usually download an application to interact with?

- ☐ Yes, Always
- ☐ Sometimes
- ☐ Almost never
- ☐ Never

7. How often you don't remember the name of products and you need to search for similarities until you actually find what you were looking for?

- ☐ Very often
- ☐ Often
- ☐ Sometimes
- ☐ Almost never
- ☐ Never

8. An option to search for products using its image instead of typing, would be something that would be attracted to you?

- ☐ Yes
- ☐ No
- ☐ Could be relevant
- ☐ Does not matter

9. What did you think about IMseach functionality?



DONE

## 6.4. Key terms

Term	Definition
SHA	In cryptography, <b>SHA-1</b> is a cryptographic hash function which takes an input and produces a 160-bit (20-byte) hash value known as a message digest - typically rendered as a hexadecimal number, 40 digits long.
HTTP	HTTP (Hypertext Transfer Protocol) is the set of rules for transferring files (text, graphic images, sound, video, and other multimedia files) on the World Wide Web. As soon as a Web user opens their Web browser, the user is indirectly making use of HTTP. HTTP is an application protocol that runs on top of the TCP/IP suite of protocols (the foundation protocols for the Internet).
API	A set of subroutine <b>definitions</b> , protocols, and tools for building <b>application</b> software, used to exchange information among different languages.
Image recognition (Google cloud vision)	Google Cloud Vision API enables developers to understand the content of an image by encapsulating powerful machine learning models in an easy to use REST API. It quickly classifies images into thousands of categories
eBay	an online auction and shopping <b>website</b> in which people and businesses buy and sell a wide variety of goods and services worldwide.
Firebase	Cloud Messaging (FCM) is a cross-platform solution for messages and notifications for Android, iOS, and web applications, which currently can be used at no cost.
JSON	