

National College of Ireland

< BSc. (Honours) in Computing >

<Software Development>

<Academic Year i.e. 2021/2022>

<Jack Flahive>

<x19117019>

<x19117019@student.ncirl.ie>

<Project Title>

Technical Report

Contents

Executive Summary	2
1.0 Introduction	2
1.1. Background	2
1.2. Aims.....	2
1.3. Technology	2
1.4. Structure	2
2.0 System.....	2
2.1. Requirements.....	2
2.1.1. Functional Requirements	3
2.1.1.1. Use Case Diagram	3
2.1.1.2. Requirement 1 <Name of requirement in a few words>	3
2.1.1.3. Description & Priority.....	3
2.1.1.4. Use Case	3
2.1.2. Data Requirements	11
2.1.3. User Requirements	11
2.1.4. Environmental Requirements	12
2.1.5. Usability Requirements	12
2.2. Design & Architecture	12
2.3. Implementation	13
2.4. Graphical User Interface (GUI)	18
2.5. Testing.....	27
2.6. Evaluation	39
3.0 Conclusions	28
4.0 Further Development or Research	28
5.0 References	28
6.0 Appendices.....	28
6.1. Project Proposal	28
6.1. Ethics Approval Application (only if required)	28
6.2. Reflective Journals	28
6.3. Invention Disclosure Form (Remove if not completed)	Error! Bookmark not defined.
6.4. Other materials used	28

Executive Summary

This report is a technical report of an android studio project called DermaCare. The application uses a TensorFlow Lite model trained on seven classes of skin diseases. The user can upload a photo and run the model on it and save their results. The application allows for business accounts which can post on a news feed-like feature to promote their practices.

1.0 Introduction

1.1. Background

I chose to undertake this project to deepen my knowledge of Deep learning and Android development.

1.2. Aims

The project aims to achieve the following:

- Create a TensorFlow model image classification model trained on seven classes of skin diseases
- Integrate it in an android application where it can be run on a photo the user submits
- The app will have two types of accounts with different access levels
- Users can save their results.

1.3. Technology

The technology used in this project will be:

- Android Studio
- Java
- TensorFlow
- Python
- Google Colab Sheet
- Firebase

1.4. Structure

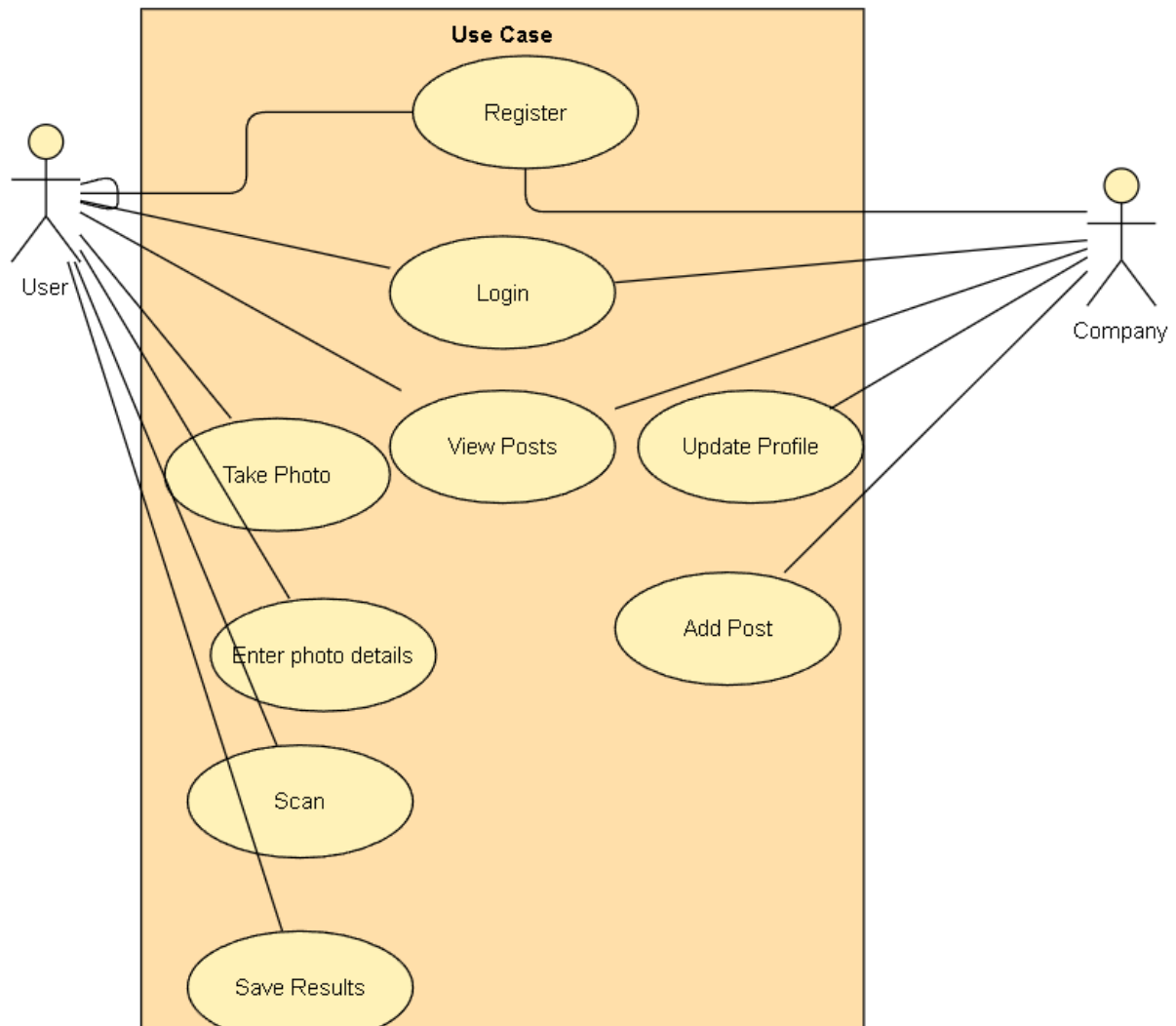
The structure of this report will include the system requirements including user, environmental, and usability requirements. It will include the different use cases for the application and descriptions of the flows of the use cases. It will include data structure and architecture of the application. Also included will be screenshots of the application and a section on testing.

2.0 System

2.1. Requirements

2.1.1. Functional Requirements

2.1.1.1. Use Case Diagram



2.1.1.2. Requirement 1: Register as a user

2.1.1.3. Description & Priority

This requirement describes how the user can register an account on the app as a “user”

2.1.1.4. Use Case

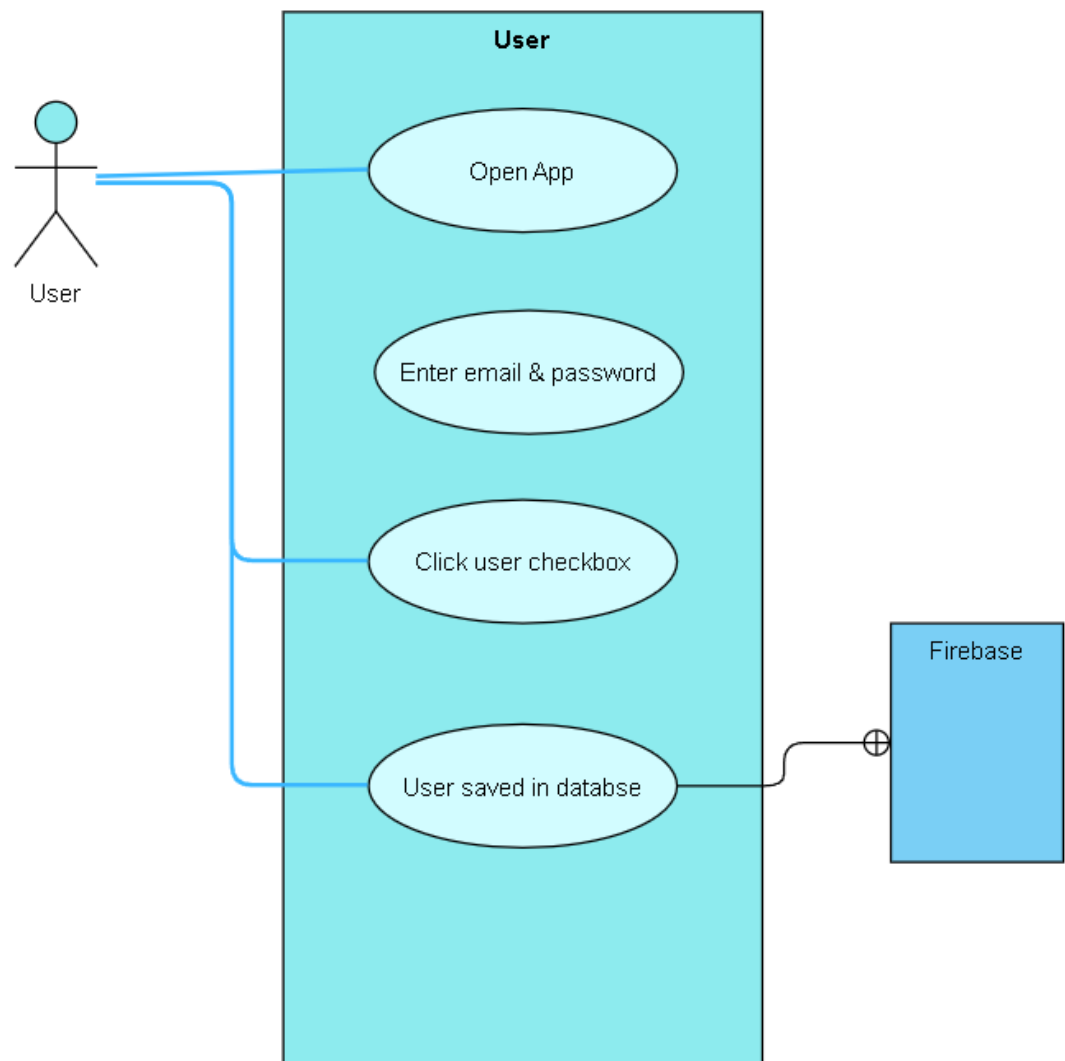
Scope

The scope of this use case is to show the user registering for one of the two specific types of accounts with their access levels.

Description

The use case describes how the user will register for a “user” account and be sent to the user home page.

Use Case Diagram



Flow Description

Precondition

The user has opened the application and hits the register button to navigate to the registration page.

Activation

The use case starts when the user has navigated to the registration page and enters their email and password and selects the user checkbox.

Main flow

1. The system identifies the user wants to register.
2. The user then enters their email and password

3. The user then clicks the checkbox which says user.
4. The user is the registered in the firebase database.

Alternate flow

A1 : <Invalid>

1. The system identifies the user wants to register
2. The user enters their email and password.
3. The email entered is invalid either because of the format or the email has already been registered in the firebase cloud database.

Exceptional flow

E1 : <No internet connection>

4. The system identifies the user wants to register.
5. The user enters their email and password.
6. The user attempts to register but fails as there is no internet connected needed to register the account on firebase.

Termination

The use case terminates hen the user successfully registers an account or the registration fails.

Post condition

If the user successfully registers an account, they will be navigated to the user home page. If the user registration attempt is unsuccessful there will be an error message and the user can attempt to register again.

[2.1.1.5. Use Case Diagram](#)

[2.1.1.6. Requirement 2: Register as a company](#)

[2.1.1.7. Description & Priority](#)

This requirement describes how the user can register as a company account and have different access levels to the user account described in the use case above. This use case is a high priority as a user must register as one type of account to use the application.

[2.1.1.8. Use Case](#)

Scope

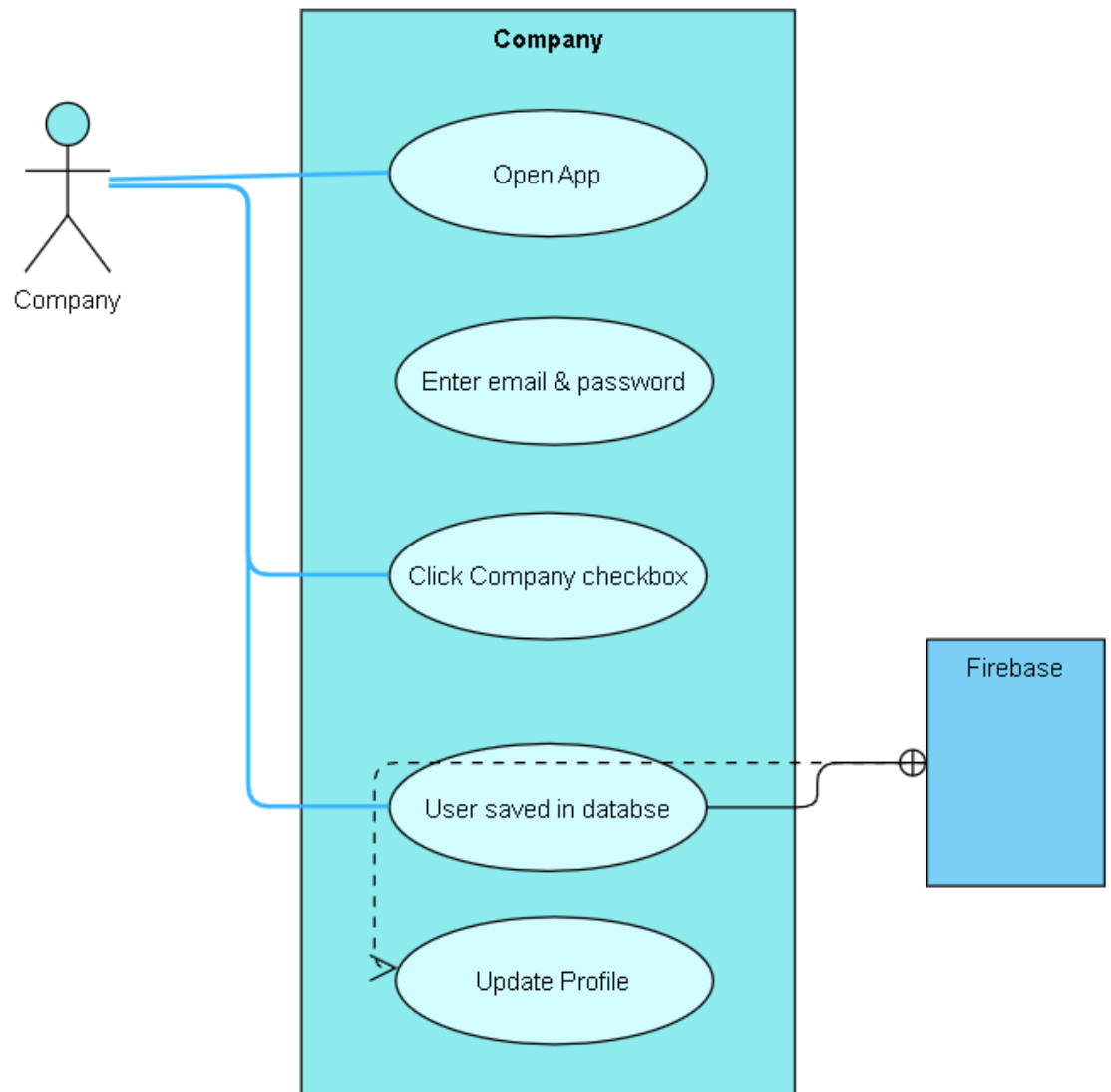
The scope of this use case is for the user to register as company account.

Description

The user will enter their email and password and select the company checkbox. Once they register, they will be navigated to a new activity where they will be

required to enter more details. The company account is a public account as they have access to post on the news feed of the app. Therefore, they are required to have a username and a photo and details about their company such as address, phone, and website link.

Use Case Diagram



Flow Description

Precondition

The user has opened the app and has navigated to the registration activity.

Activation

The use case starts when the user enters an email, password and selects the company checkbox.

Main flow

5. The system identifies the user intends to register.
6. The user enters email, password, and checks company checkbox.
7. The user email and password is registered in the firebase authentication system.
8. The user is navigated to another activity where they upload a photo, enter the username (company title), company address, company phone number, and the web address of the company website. The details are saved in the firebase cloud database.

Alternate flow

A1 : <Invalid details>

7. The system identifies the user intends to register
8. The user enters email and password
9. The user enters an email with an invalid format, or which is already registered

Exceptional flow

E1 : <No internet>

10. The user attempts to register an account with no internet connection, and therefore cannot connect to firebase cloud database.

Termination

The use case terminates when the user has registered as a company profile.

Post condition

The user is navigated to the company home page.

[2.1.1.9. Use Case Diagram](#)

[2.1.1.10. Requirement 3 Upload Post](#)

[2.1.1.11. Description & Priority](#)

This use case describes how a profile registered as a company can post promotions on the news feed. This use case is a medium priority as it is not essential to use the application.

[2.1.1.12. Use Case](#)

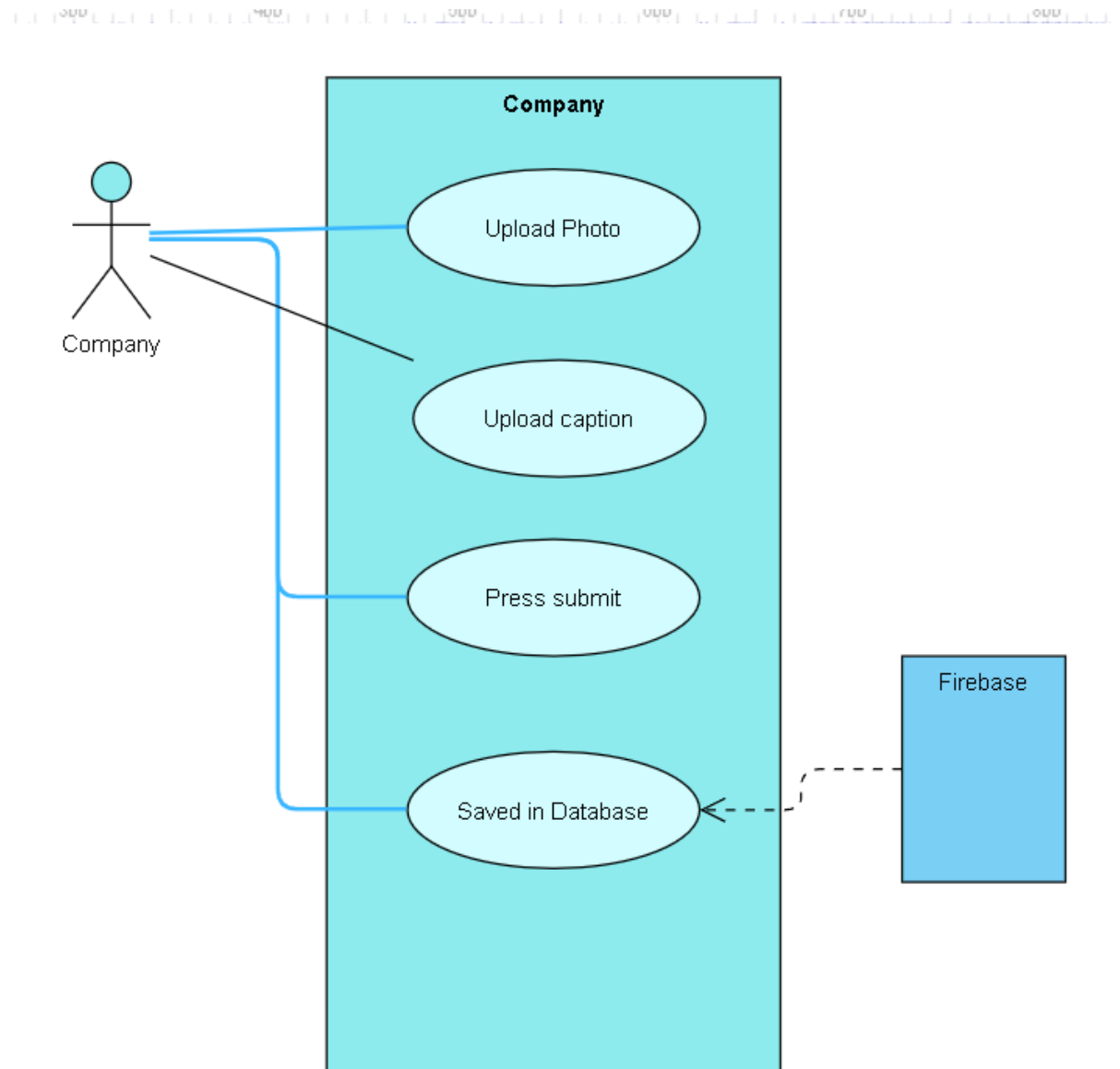
Scope

The scope of this use case is how a company profile can post.

Description

This use describes how a company profile can post.

Use Case Diagram



Flow Description

Precondition

The profile must be registered as a company profile and logged in.

Activation

The use case starts when the user navigates to the add post activity.

Main flow

9. The user enters the content of the post, including and image and text.
10. The user selects the upload button, and the post is uploaded.

11. Along with the image and description, the profile details are attached to the post. These include the company address, website address, and phone number.
12. Once uploaded the user is navigated to the posts section where they can see their posts along with others.

Alternate flow

A1 : <No internet>

11. The user enters the content of the post and attempts to upload the post to the cloud database.
12. The user does not have the internet connection to upload the post and an error message is shown.

Termination

The use case terminates when the user has uploaded the post.

Post condition

The user is navigated to the posts section.

2.1.1.13. Use Case Diagram

2.1.1.14. Requirement 4: Scan Activity

2.1.1.15. Description & Priority

This use case is how a user registered as a user profile can upload a photo and use the TensorFlow lite mode to scan for the seven categories of skin diseases it is trained on.

2.1.1.16. Use Case

Scope

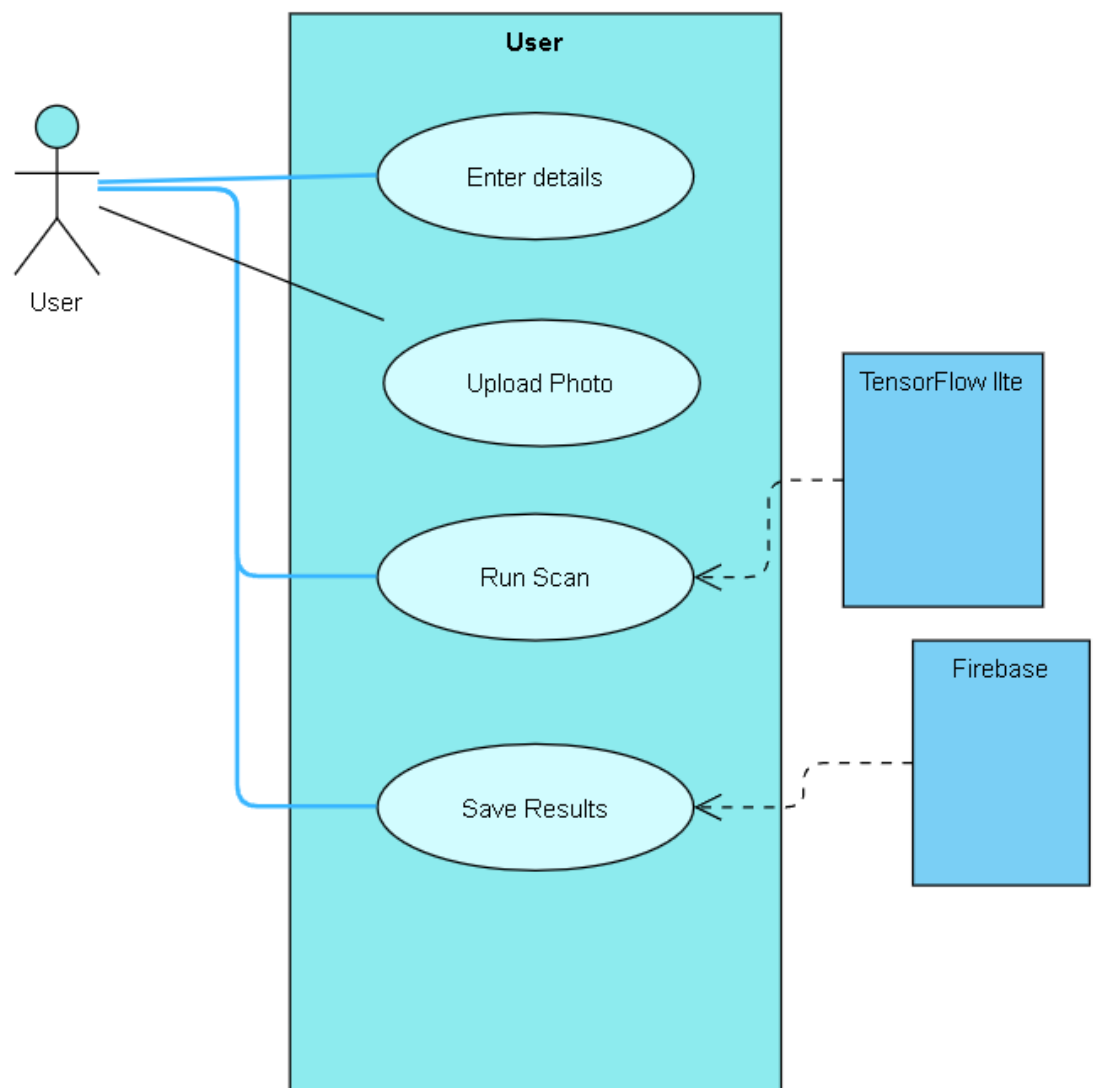
The scope of this use case is how the user can run the TensorFlow lite model on an image they submit. They also are given the option to take a photo with a in-app

camera. They are required to enter details about the image, and they are given the option to save the image, result, and details that they entered. This will be saved to the Firebase cloud database.

Description

This use case describes how the user can submit a photo and run the TensorFlow lite model. It also describes how they can take a photo, submit details, and save the photo and results.

Use Case Diagram



Flow Description

Precondition

The user is logged in as a user profile.

Activation

This user has navigated to the scan activity.

Main flow

13. The user selects the camera button and takes a photo.
14. The user enters their answers to the questions relating to the skin photo they are scanning.
15. The user uploads the photo they took from their gallery.
16. The user selects a button to run the TensorFlow model on the photo.
17. A Bottom sheet pops up with a percentage chart of the highest confidence result and a bar chart of the confidence values for all the different classes of the model is trained on.
18. The user selects the save button and the image, results, and details they entered are saved to the database.

Alternate flow

A1 : <No Camera>

13. The already has a photo and does not want to use the camera. The select the skip button on the camera activity instead.
14. The user enters the details required.
15. The user submits a photo and runs the TensorFlow Lite model.
16. The user is then given the option to save and declines.

Termination

The use case is terminated when the user runs the model and either saves the image, result, and details and declines to.

Post condition

If the user saves the results they are navigated to the results activity. If they do not save, they left on the same activity.

2.1.2. Data Requirements

2.1.3. User Requirements

The user requirements for the application are as follows;

- The user has an android device
- An internet connection is needed to use the application
- An email address to register

- Companies registering on the app will provide name, address, phone, email, password and website link
- Users registering on the app will provide just email and password

2.1.4. Environmental Requirements

The environmental requirements for the application are as follows;

- Android device
- An internet connection
- Android device with camera to use optional feature

2.1.5. Usability Requirements

Navigation:

The application should be easy to navigate to all features of the application. The user should have a well-designed home page where all the features are viewable and labelled. When the user writes to the database, the application should bring the user to view their input displayed in the database.

Graphical display of data:

When the user uses the model on an image they submitted, the results and returned data should be displayed in an well designed and readable form, with the use of charts.

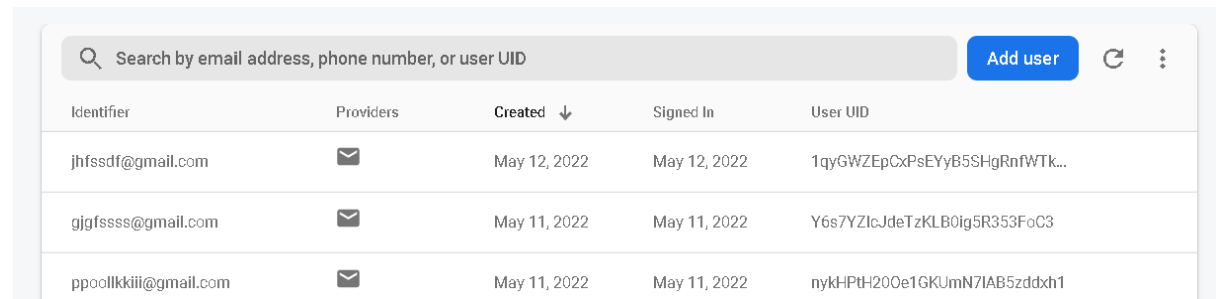
2.2. Design & Architecture

Describe the design, system architecture and components used. Describe the main algorithms used in the project. (Note use standard mathematical notations if applicable).

An architecture diagram may be useful. In case of a distributed system, it may be useful to describe functions and/or data structures in each component separately.

For data storage and user authentication, this application uses Firebase cloud Fire store and Firebase authentication.

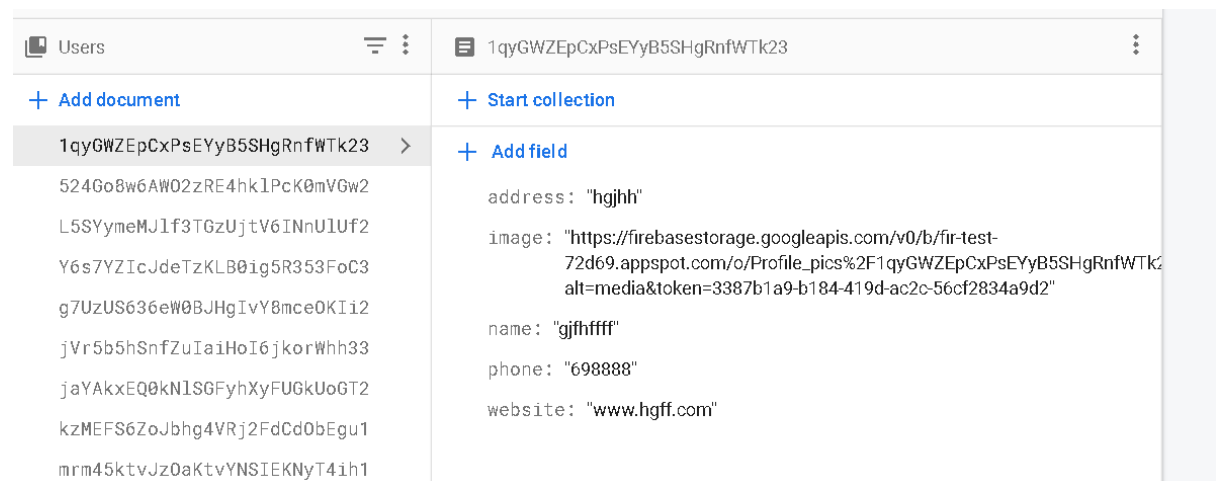
Firebase authentication:



A screenshot of the Firebase Authentication console. At the top, there is a search bar with the placeholder text "Search by email address, phone number, or user UID" and an "Add user" button. Below the search bar is a table with the following columns: Identifier, Providers, Created, Signed In, and User UID. The table contains three rows of user data.

Identifier	Providers	Created	Signed In	User UID
jhfssdf@gmail.com	📧	May 12, 2022	May 12, 2022	1qyGWZEpcXpSEyB5SHgRnfWTK...
gjfssss@gmail.com	📧	May 11, 2022	May 11, 2022	Y6s7YZIcJdeTzKLB0ig5R353FoC3
ppoolkkiii@gmail.com	📧	May 11, 2022	May 11, 2022	nykHPtH200e1GKUmn7IAB5zddxh1

All of the users registered are stored here. When the company users add additional details to their profile, those details are stored in the cloud Fire store:



A screenshot of the Firebase Cloud Firestore console. The left sidebar shows the "Users" collection. The main area displays a document with the ID "1qyGWZEpcXpSEyB5SHgRnfWTK23". The document contains the following fields:

Field	Value
address	"hgjhh"
image	"https://firebasestorage.googleapis.com/v0/b/fir-test-72d69.appspot.com/o/Profile_pics%2F1qyGWZEpcXpSEyB5SHgRnfWTK23?alt=media&token=3387b1a9-b184-419d-ac2c-56cf2834a9d2"
name	"gjfhffff"
phone	"698888"
website	"www.hgff.com"

There are two other database collections which hold the results of the scan and the posts:

Home > Results > 6VVmxL59WZqj...		
fir-test-72d69	Results	6VVmxL59WZqj...
+ Start collection	+ Add document	+ Start collection
Posts	6VVmxL59WZqjxRi8ssAr >	+ Add field
Results >	CVJagEQpsI37HcHEjs0Y IJb10U3GIfXCQJXKcesB PAb1TFhXKBZuSK96WYB4 SmaHxh3HW9kwTZ2nw177 gJVIOgTu5zRPkZVNo0Jp mKm0HwrJatvYoAKnGcf0	Answer1 Answer2 Question1 Question2 Result: image: " ; & time: M user: "n
Users		

The TensorFlow lite model was trained on the HAM10000 ("Human Against Machine with 10000 training images") dataset which contains 10000 images of seven types of skin diseases.

The seven different classes of skin diseases are listed below:

- Melanocytic nevi
- Melanoma
- Benign keratosis-like lesions
- Basal cell carcinoma
- Actinic keratoses
- Vascular lesions
- Dermatofibroma

Screenshot from the Google Colab sheet:

```
[ ] data = ImageClassifierDataLoader.from_folder(image_path)
train_data, test_data = data.split(0.9)

INFO:tensorflow:Load image with size: 10015, num_label: 7, labels: akiec, bcc, bkl, df, mel, nv, vasc.

[ ] model = image_classifier.create(train_data)

INFO:tensorflow:Retraining the models...
Model: "sequential"

Layer (type)                Output Shape                Param #
=====
hub_keras_layer_v1v2 (HubKer (None, 1280)                3413024
rasLayerV1V2)

dropout (Dropout)           (None, 1280)                0

dense (Dense)                (None, 7)                   8967
=====
Total params: 3,421,991
Trainable params: 8,967
Non-trainable params: 3,413,024

None
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/gradient_descent.py:102: UserWarning: The `lr` argument is deprec
super(SGD, self).__init__(name, **kwargs)
Epoch 1/5
281/281 [=====] - 407s 1s/step - loss: 1.1290 - accuracy: 0.6965
Epoch 2/5
281/281 [=====] - 433s 2s/step - loss: 1.0294 - accuracy: 0.7348
Epoch 3/5
281/281 [=====] - 433s 2s/step - loss: 1.0067 - accuracy: 0.7440
```

Implementation

Describe the main algorithms/classes/functions used in the code. Consider to show and explain interesting code snippets where appropriate.

Deployment of TensorFlow Lite model in android studio application:

```
try{
    tflite=new Interpreter(loadmodelfile( activity: MainActivity.this));
}catch (Exception e) {
    e.printStackTrace();
}
```

Initialise interpreter in a try/catch block and loads the model as a byte buffer by calling the method below. Function Mapped Byte Buffer takes in the TensorFlow Lite model and returns it as a byte buffer.


```
private MappedByteBuffer loadmodelfile(Activity activity) throws IOException {
    AssetFileDescriptor fileDescriptor=activity.getAssets().openFd( fileName: "model.tflite");
    FileInputStream inputStream=new FileInputStream(fileDescriptor.getFileDescriptor());
    FileChannel fileChannel=inputStream.getChannel();
    long startoffset = fileDescriptor.getStartOffset();
    long declaredLength=fileDescriptor.getDeclaredLength();
    return fileChannel.map(FileChannel.MapMode.READ_ONLY,startoffset,declaredLength);
}
```

Using the interpreter, this block of code executed when the user submits an image and hits the run button. Gets the input tensor to get the shape, height and width

```
int imageTensorIndex = 0;
int[] imageShape = tflite.getInputTensor(imageTensorIndex).shape();
imageSizeY = imageShape[1];
imageSizeX = imageShape[2];
DataType imageDataType = tflite.getInputTensor(imageTensorIndex).dataType();
```

Get the shape, height, and width.

```
inputImageBuffer = loadImage(bitmap);

tflite.run(inputImageBuffer.getBuffer(),outputProbabilityBuffer.getBuffer().rewind());
```

Load image function converts bitmap from image submitted into a tensor image and resizes it to optimal size for model. Then the tensor image is put it in the Image buffer. Input Image Buffer is now ready to be used for the TensorFlow Lite model.

Finally, the method is called to show the outputs, with the confidence levels and labels. The labels are read from the label file of the names of the different classes. The method then derives the probability or confidence output from the output probability buffer and maps it with the labels as a string(label) and float(probability). The map is looped through to retrieve the label with the max value.

```

try{
    labels = FileUtil.loadLabels( context: MainActivity.this, filePath: "labels.txt");
}catch (Exception e){
    e.printStackTrace();
}
Map<String, Float> labeledProbability =
    new TensorLabel(labels, probabilityProcessor.process(outputProbabilityBuffer))
        .getMapWithFloatValue();
float maxValueInMap =(Collections.max(labeledProbability.values()));

for (Map.Entry<String, Float> entry : labeledProbability.entrySet()) {
    if (entry.getValue()==maxValueInMap) {
        classitext.setText(entry.getKey() + " " + maxValueInMap * 100 + "%");
        mChart.setProgress( progress: maxValueInMap*100, animate: true);
    }

    String[] label = labeledProbability.keySet().toArray(new String[0]);
    Float[] labelProbability = labeledProbability.values().toArray(new Float[0]);
}

```

Different types of users with different access levels:

After researching, I found no direct way in which firebase allows for an application to have multiple classes of users with their own separate access levels. To get around this, when a user registers on the app, they can select an option to be a user or company and for each one they select, they will be sent to separate activities upon registering an email and password. From then on, they can submit more profile data suitable for the type of user they are.

```

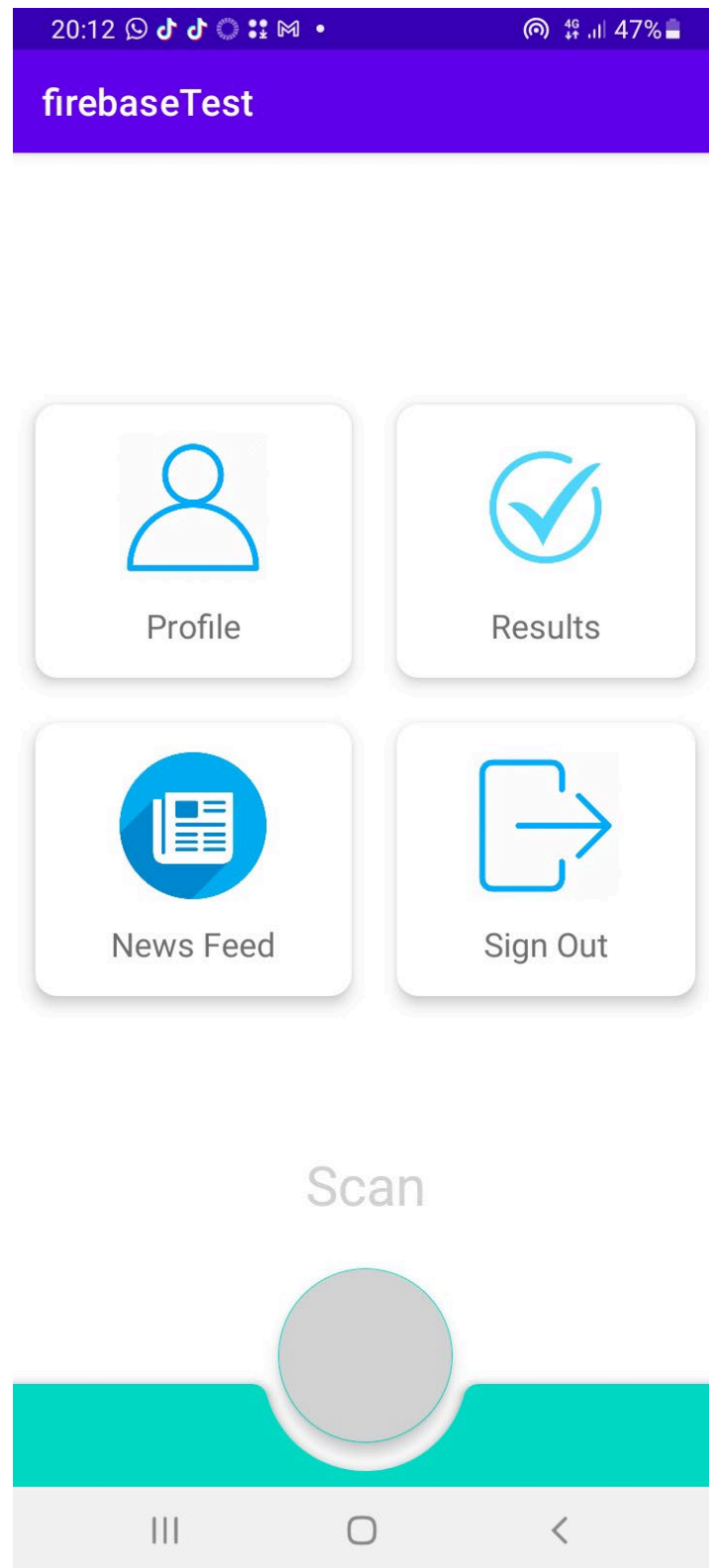
if(isCompanyBox.isChecked()){
    userInfo.put( k: "isCompany", v: "1");
}else{
    userInfo.put( k: "isUser", v: "1");
}

if(isCompanyBox.isChecked()){
    Intent intent = new Intent( packageContext: Register.this, AdminActivity.class);
    startActivity(intent);
    finish();
}else{
    |
    Intent intent = new Intent( packageContext: Register.this, userHome.class);
    startActivity(intent);
    finish();
}

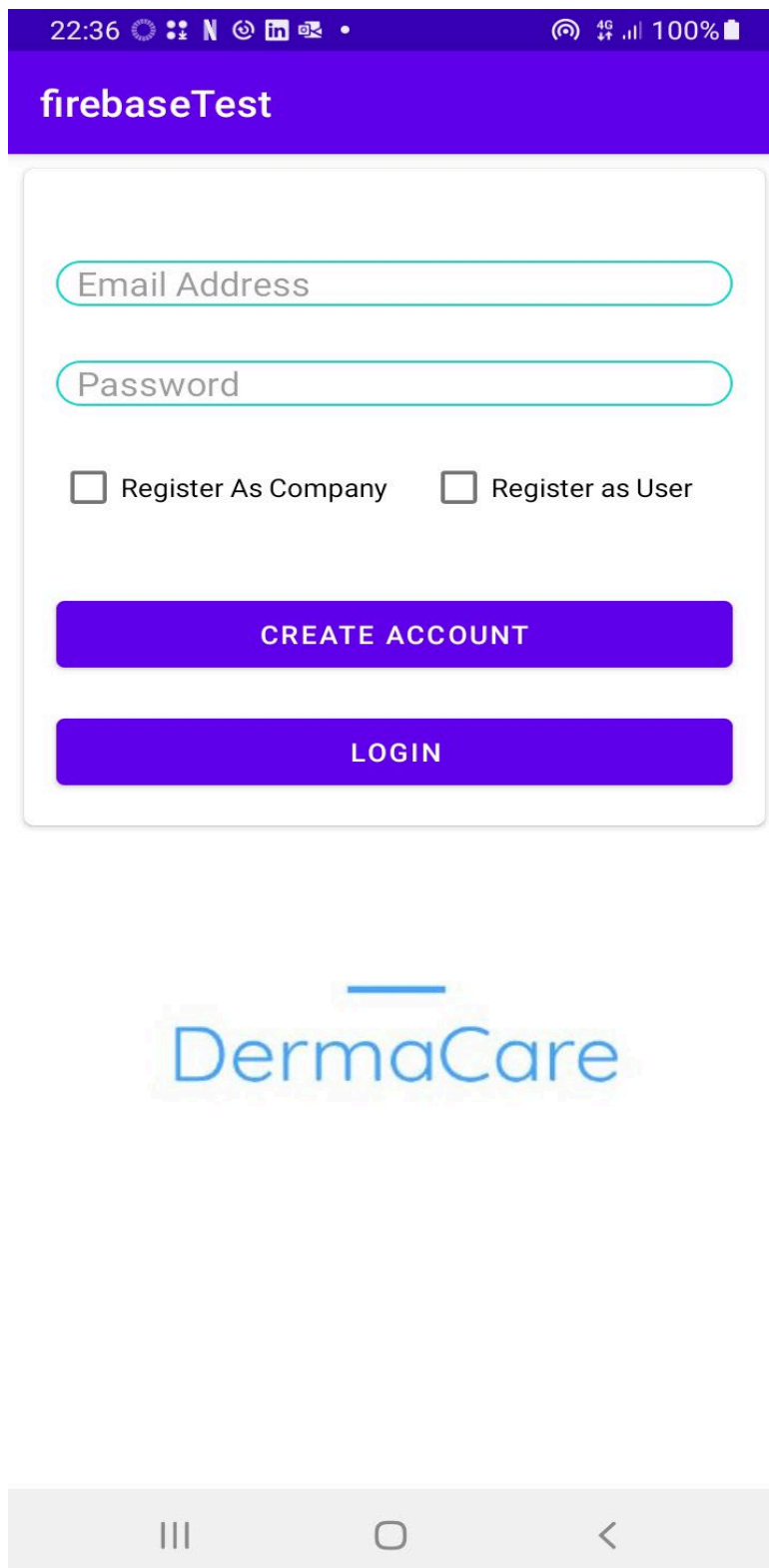
```

2.3. Graphical User Interface (GUI)

User Home Activity:



Register Activity:



The image shows a mobile application interface for a registration activity. At the top, a purple status bar displays the time 22:36, various system icons, and a 4G signal with 100% battery. Below this is a purple header with the text "firebaseTest". The main content area is a white card with rounded corners. It contains two input fields: "Email Address" and "Password", both with light blue borders. Below these fields are two checkboxes: "Register As Company" and "Register as User". At the bottom of the card are two purple buttons: "CREATE ACCOUNT" and "LOGIN". Below the card, the "DermaCare" logo is displayed in blue, featuring a horizontal line above the text. At the very bottom, a grey navigation bar contains three icons: a hamburger menu, a home icon, and a back arrow.

firebaseTest

Email Address

Password

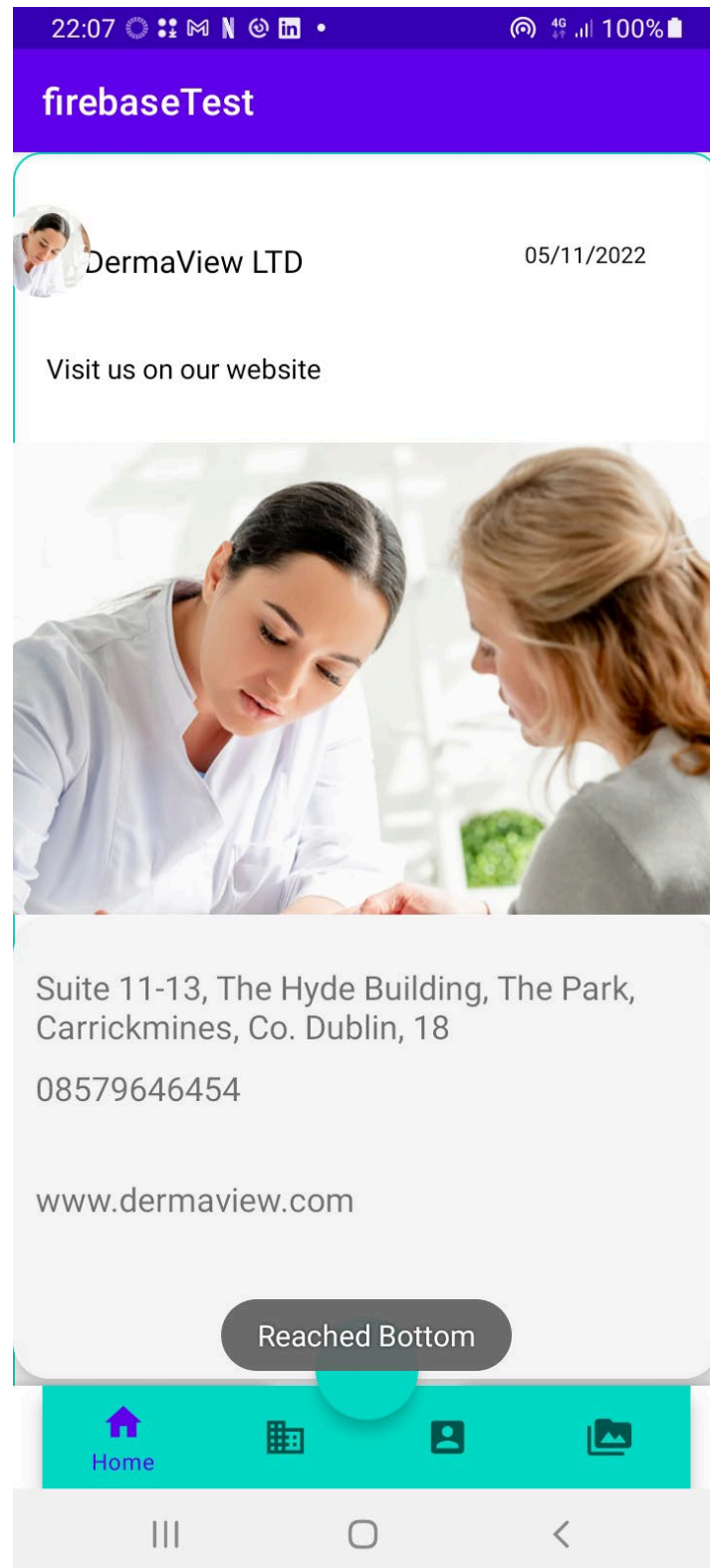
☐ Register As Company ☐ Register as User

CREATE ACCOUNT

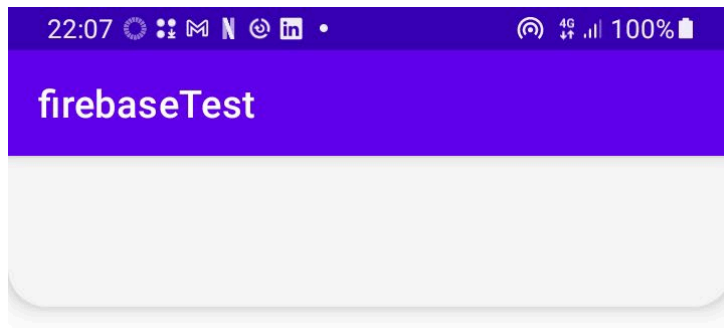
LOGIN

DermaCare

View Posts Activity:

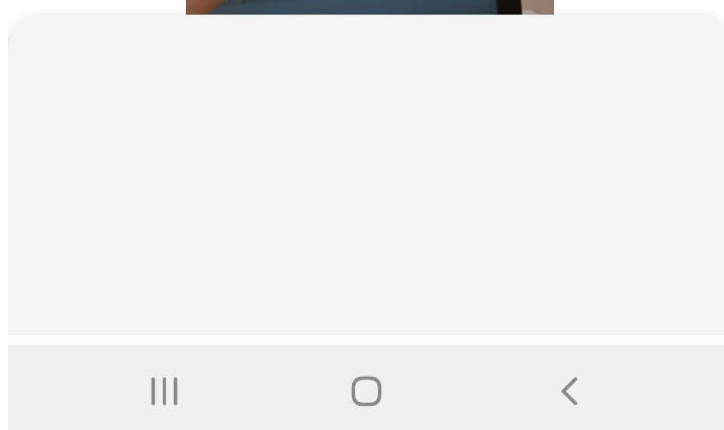
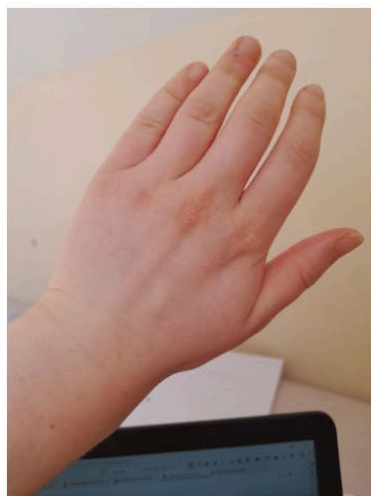


View Saved Results Activity:

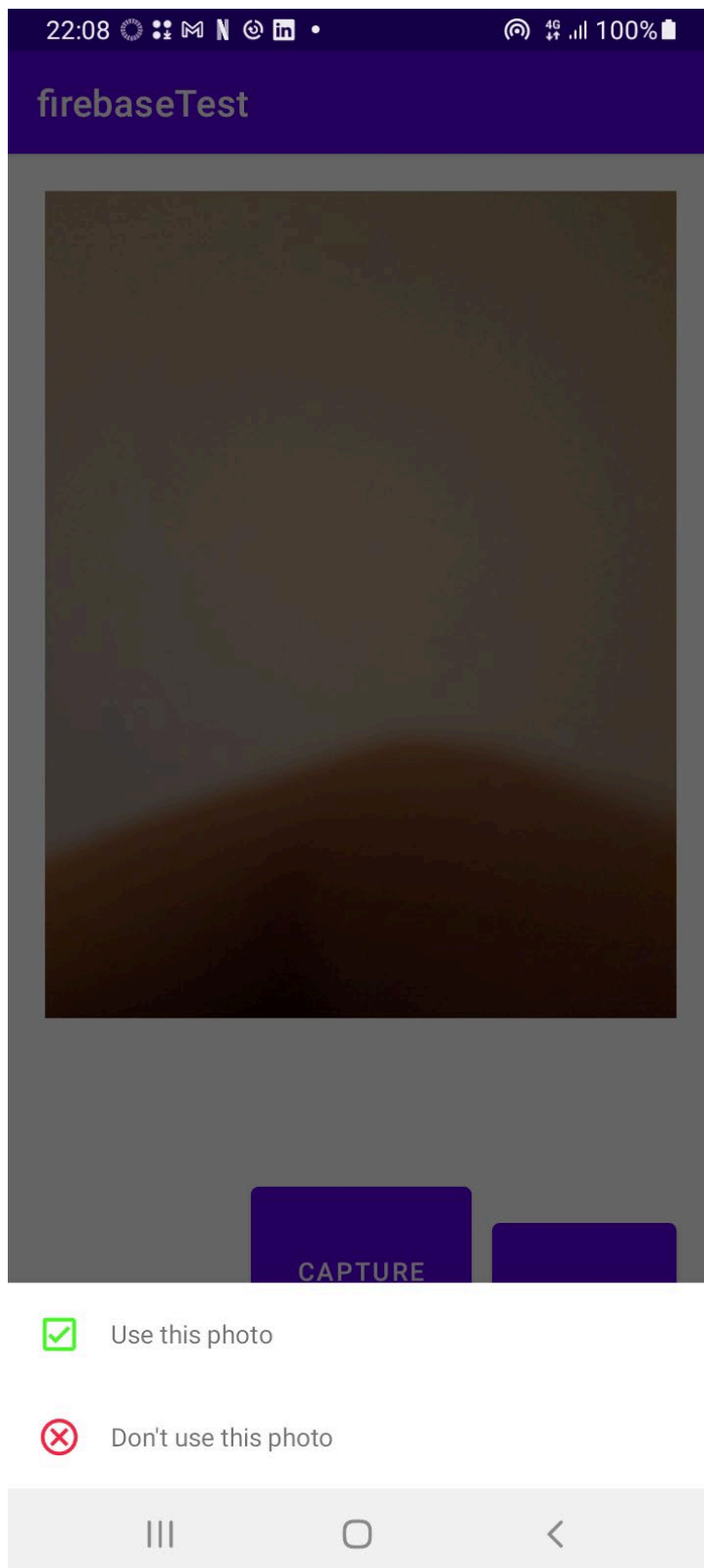


Jack Flahive

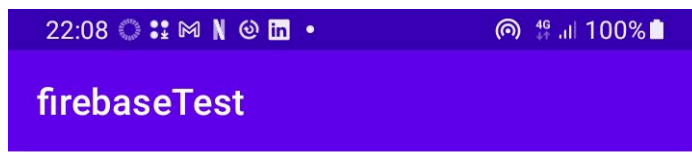
05/14/2022



Camera Activity:



Enter Details Activity:



How long have you had this spot?

answer

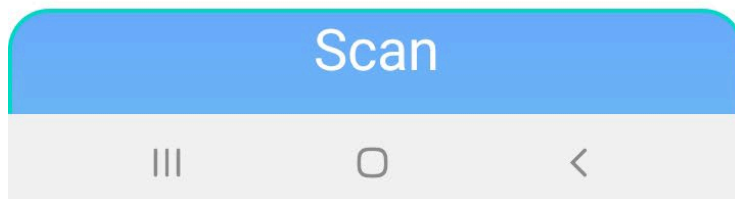
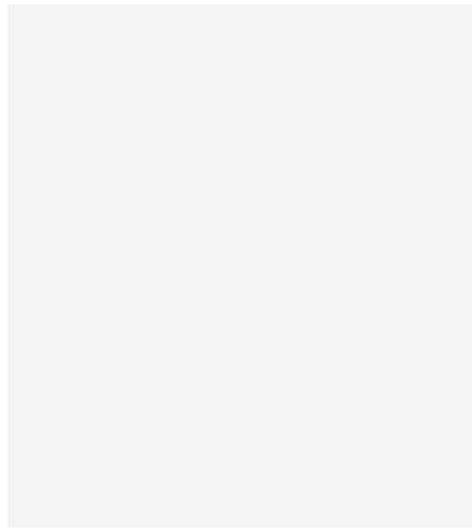
Has it changed in size, shape, color over time?

answer

NEXT








Submit Photo for Scan:



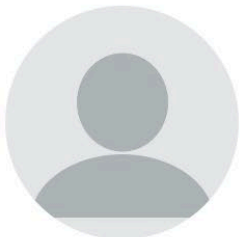
Response from running model:



Company update profile activity:

22:16   N   100% 

firebaseTest



username


address


phone


website

SAVE

LOGOUT







2.4. Testing

```
private Login mLogin = null;

@Before
public void setUp() throws Exception {

    mLogin = mActivityTestRule.getActivity();
}

@After
public void tearDown() throws Exception {
}

@Test
public void onCreate() {
    View view = mLogin.findViewById(R.id.loginEmail);
    assertNotNull(view);
}
```

single and multi-device test results.

>> ✓ Tests passed: 1 of 1 test – 6 s 515 ms

15 ms Testing started at 19:01 ...

05/15 19:01:02: Launching 'onCreate()' on samsung SM-
Install successfully finished in 17 s 744 ms.
Running tests

\$ adb shell **Tests passed: 1** sent -w -m -e debug false -e

Database Inspector Run Profiler Build Logout

```

@Rule
public ActivityTestRule mActivityTestRule= new ActivityTestRule<Register>(Register.class);

private Activity mActivity = null;

Instrumentation.ActivityMonitor monitor = getInstrumentation().addMonitor(Login.class.getName(), result: null, block: false);

@Before
public void setUp() throws Exception {
    mActivity = mActivityTestRule.getActivity();
}

@Test
public void testLaunchLoginActivityfrombutton()
{
    assertNotNull(mActivity.findViewById(R.id.gotoLogin));
    onView(withId(R.id.gotoLogin)).perform(click());

    Activity Login = getInstrumentation().waitForMonitorWithTimeout(monitor, timeout: 5000);

    assertNotNull(Login);
}

```

3.0 Conclusions

Advantages:

One of the advantage of this project is that neural networks and deep learning are an important feature of current and future computing.

Limitations:

The TensorFlow model is limited and could be improved by having more datasets other than the seven classes it was trained on.

4.0 Further Development or Research

5.0 References

Please include references throughout your document where appropriate. See [here](#) for a guide on referencing from the NCI library.

6.0 Appendices

This section should contain information that is supplementary to the main body of the report.

6.1. Project Proposal

6.1. Ethics Approval Application (only if required)

6.2. Reflective Journals

6.3. Other materials used

Any other reference material used in the project for example evaluation surveys etc.

Proposal

Executive Summary

Max 300 words. Summarise the key points of the report. Restate the purpose of the report, highlight the major points of the report, and describe any results, conclusions, or recommendations from the report.

7.0 Introduction

7.1. Background

I aimed to undertake this project as a challenge to attempt in an area of software development that I haven't done before. I also have a strong interest in android development and have developed android applications before and am looking to further develop my skills in it.

7.2. Aims

The aims of this project are to develop an android application that uses a convoluted neural network which will return a similarity results match to a photo that a user will upload.

7.3. Technology

The project will be broken up into two main parts. The first part is developing and training the neural network. This will be done using python and TensorFlow. The second main part is implementing the neural network into an android application using TensorFlow lite. The android application will use Java. The cloud storage and authentication will be handled by firebase.

7.4. Structure

Provide a brief overview of the structure of the document and what is addressed in each section.

8.0 System

8.1.1.1. Registration

This is a high priority requirement that allows both types of users to register on the application. There will be two registration forms, one for regular users and one for companies/dermatologists. The two classes of users will be implemented through firebase role-based action control.

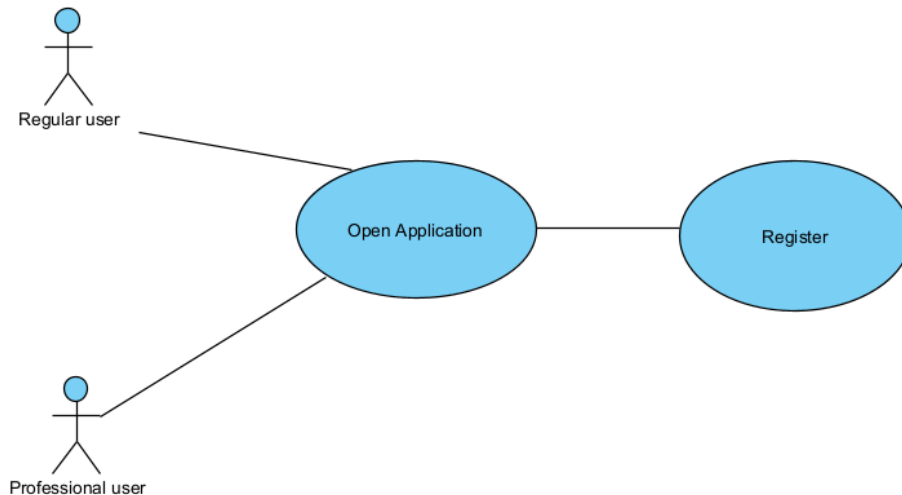
Scope

The scope of this use case is for the two types of users to register.

Description

Once the app is opened there will be a registration button, then if clicked there will be two registration options to choose from: User & Professional. The two registration forms will have different fields.

Use Case Diagram



Flow Description

Precondition

The application is installed and opened successfully.

Activation

The use case starts when the actor opens the application and clicks register. The actor will then select which registration form is correct for them, user or professional.

Main flow

19. The user successfully opens and installs the application.
20. The user is brought to a landing page with an option to register or login
21. The user selects register and is prompted with two options to select from:
user or professional
22. The user selects the correct option for them
23. They enter the correct form details
24. The user is successfully added to the firebase authentication database
25. The user is then navigated to the home page of the application.

Alternate flow

A1 : Incorrect details

17. The user installs and opens the application
18. The user selects a registration form
19. The user enters their details in the registration form

20. The email address which is used to authenticate the user is misspelled or does not exist
21. The user is prompted that that email is incorrect

Exceptional flow

E1 : No internet access

22. The user installs and opens the application
23. The user fills in the correct details in the registration form
24. For a successful registration in the firebase real time database their must be an internet connection
25. The user does not have an internet connection and cannot register.

Termination

The user is successfully registered in the real time database.

Post condition

They are brought to the home page of the application.

8.1.1.2. Login

This requirement is for the login. It is a high priority as the application cannot be used for without a user successfully logged in.

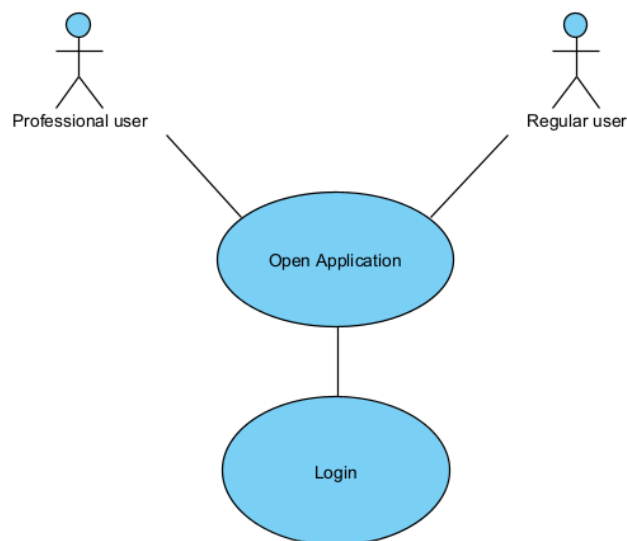
Scope

The scope of this use case is to successfully allow a registered user to login.

Description

This use case describes the login process of a user that is already registered and has a working internet connection.

Use Case Diagram



Flow Description

Precondition

The user has already registered an account.

Activation

This use case starts when the user opens the application and selects the login button on the landing page.

Main flow

26. The user opens the application
27. The user selects the login button on the landing page
28. The user will have two options to login as: User or Professional
29. The user selects the type of account they have registered
30. The user enters the correct login details
31. Login successful and the user is brought to the home page

Alternate flow

A1 : Incorrect login details

26. The user opens the application
27. The user selects the login button
28. The user enters misspelled details or tries to login with an account that is not registered
29. The user is prompted that the login details are incorrect and does not navigate to anywhere

Exceptional flow

E1 : No internet connection

30. The user opens the application and selects login
31. The user has an account registered and enters the correct details
32. For firebase authentication to work successfully the user must be connected to the internet
33. The user is not connected to internet and is cannot login and is informed with an alert dialogue.

Termination

The user successfully logs in.

Post condition

The user is brought to the home page.

8.1.1.3. Quick scan

This requirement is the highest priority in the application. This is where the convoluted neural network is used to get a results match for an image that the user uploads.

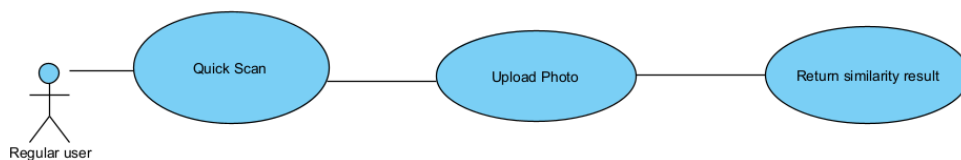
Scope

The scope of this use case is to allow the user to upload a photo of a potential skin condition the have and to get a potential match based on similarity from the image data on the firebase cloud storage

Description

This use case describes how the user can upload a photo and get a match for it based on similarity based on the image data stored on firebase cloud storage.

Use Case Diagram



Flow Description

Precondition

A user who is registered as a regular user is successfully logged in and at the home page.

Activation

This use case starts when the user selects the 'Quick Scan' activity.

Main flow

32. The user selects the quick scan activity.
33. The user is brought to the activity and selects the 'check' option.
34. The user uploads a photo of a potential skin condition.
35. The application returns a similarity result to the user.

Alternate flow

A1 : No internet access

34. The user has attempted to upload a photo.

- 35. There is a break in their internet connection.
- 36. The user cannot continue with the process without an internet connection and is given an alert message.

Exceptional flow

E1 : Slow internet

- 37. The user has uploaded a photo and is waiting for result
- 38. The user has a slow internet connection and the application is taking too long to return
- 39. The user closes the application and tries again.

Termination

The system returns the similarity results of the scan to the user.

Post condition

The user can now act on the results by going to the Recyclerview of dermatologists/companies registered on the application.

8.1.1.4. Browse dermatologist/companies

The regular user can browse a Recyclerview list of registered professional accounts on the application of dermatologists and companies.

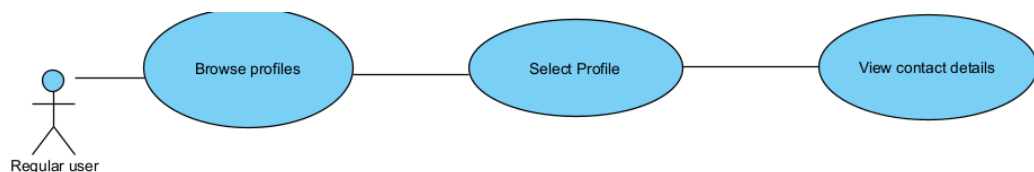
Scope

The scope of this use case is to allow the user to act on their results by getting in contact with a professional.

Description

This use case describes how the regular user can browse professional profiles containing contact details and web links to get in contact about their results.

Use Case Diagram



Flow Description

Precondition

The user is logged in to the application.

Activation

This use case starts when the user selects on the browse profile activity which only shows professional profiles.

Main flow

- 36. The user selects the browse professional profile activity.
- 37. The user browses through Recyclerview of profiles
- 38. The user looks at the contact details/web pages/descriptions linked on the profile.
- 39. The user uses the contact details to get in touch with professional.

Alternate flow

A1 : The user selects the browse professional profile activity

- 40. The user browses through the profiles
- 41. The user selects a profile and read on it
- 42. The user navigates back to home page

Exceptional flow

E1 : The user selects the browse professional profiles activity

- 43. The user browses through the profiles
- 44. The user selects a profile and reads information on it
- 45. The user selects the report button which supports the profile as suspicious of being illegitimate.
- 46. The user confirms the report submission and is navigated back to the main list on the browse profile activity.

Termination

The user exits the activity.

Post condition

The user is navigated back to the home page.

8.1.1.5. [Edit Profile](#)

This requirement is to allow both type of users to edit their profiles.

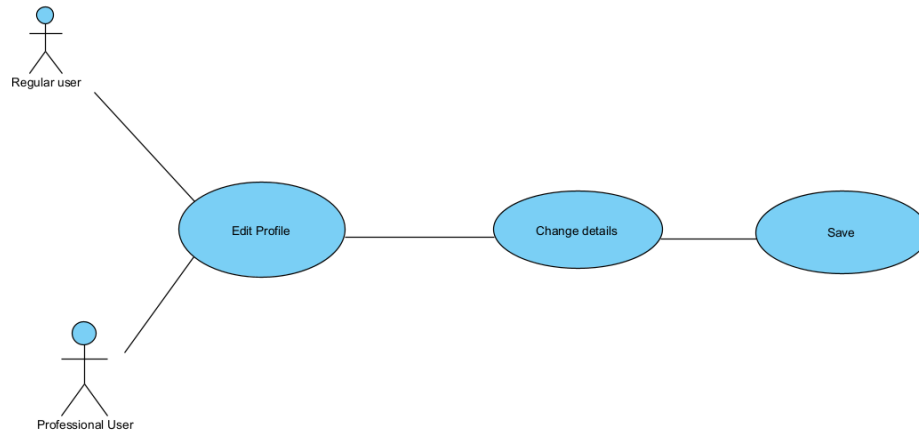
Scope

The scope of this use case is to allow both regular and professional users to edit the details of their profile.

Description

This use case describes how both users can change the details of their profile. The professional account will have more fields in it.

Use Case Diagram



Flow Description

Precondition

The user is registered and logged in.

Activation

This use case starts when the user selects the hamburger icon and selects the edit profile activity.

Main flow

- 40. The user selects the edit profile activity
- 41. The user edits the details of their profile
- 42. The user selects save and the changes are saved to their profile in firebase.

Alternate flow

A1 : Incorrect email

- 47. The user selects edit profile activity
- 48. The user changes the email of the account
- 49. The email cannot be authenticated
- 50. The user is prompted to re enter the email

Exceptional flow

E1 : Duplicate email

- 51. The user selects edit profile activity
- 52. The user changes the email of the account
- 53. The user enters an email of another registered profile
- 54. The user is prompted to use a different email

Termination

The use case terminates when the user successfully saves new details of their account or selects the back button which leaves the original details as they were.

Post condition

The user is navigated back to the home page.

8.1.1.6. Posts section

This section allows for only professional accounts to make posts promoting their business or answering common questions relating to skin care.

Scope

The scope of this use case is to allow professional accounts to submit posts and delete posts.

Description

This use case describes how professional accounts can post and delete posts relating to promotions and information posts.

Use Case Diagram



Flow Description

Precondition

A professional account is logged in. Option to create post will be set to invisible for regular users.

Activation

This use case starts when a professional account selects the create post activity.

Main flow

- 43. The account selects the create post activity
- 44. The they write the post including an optional image
- 45. The post is shown in a public Recyclerview viewable to all registered accounts

Alternate flow

A1 : Does not fit into set framework

- 55. The account selects create post activity
- 56. The account creates a post which does not abide by the character length allowed or tries to submit a file which is not an image

57. The account is prompted with an alert that the submission failed and to re-enter

Exceptional flow

E1 : Loss of internet connection

58. The user selects create post activity
59. The user creates post which conform to the set parameters.
60. The user has a loss of internet connection in the process of submitting
61. The user will be prompted with an alert dialogue informing them the process failed.

Termination

The use case terminates when the professional user either successfully submits a post or decides to abandon the post.

Post condition

The user is navigated back to home page.

List further functional requirements here, using the same structure as for Requirement1.

8.1.2. Data Requirements

8.1.3. User Requirements

8.1.4. Environmental Requirements

8.1.5. Usability Requirements

8.2. Design & Architecture

Describe the design, system architecture and components used. Describe the main algorithms used in the project. (Note use standard mathematical notations if applicable).

An architecture diagram may be useful. In case of a distributed system, it may be useful to describe functions and/or data structures in each component separately.

8.3. Implementation

Describe the main algorithms/classes/functions used in the code. Consider to show and explain interesting code snippets where appropriate.

8.4. Graphical User Interface (GUI)

Provide screenshots of key screens and explain what can be seen in each one.

8.5. Testing

Describe any testing tools, test plans and test specifications used in the project. Provide evidence for and results of all Unit, Integration and End User testing that is carried out.

8.6. Evaluation

How was the system evaluated and what are the results? This may consist of usage data. It may also include performance evaluations, scalability, correctness, etc. depending on the focus of the project. Quantative results may be reported in tables or figures.

9.0 Conclusions

Describe the advantages/disadvantages, strengths and limitations of the project

10.0 Further Development or Research

With additional time and resources, which direction would this project take?

11.0 References

Please include references throughout your document where appropriate. See [here](#) for a guide on referencing from the NCI library.

12.0 Appendices

This section should contain information that is supplementary to the main body of the report.

12.1. Project Proposal

13.0 Objectives

The objective for this project is to create a deep learning dermatology android application. The application will use a neural network using image similarity to detect skin conditions/diseases. The application will have two categories of users: regular users and dermatology professionals. The regular user will be able to upload a photo of their skin and the neural network will be used to detect a condition. They will then be able to request and set up an appointment with a dermatologist registered on the application. There will be a 'Quora' style news feed where users

registered as dermatologist and regular users will be able to post common questions and answers. The feed will also allow dermatologists to post promotions for their practices.

14.0 Background

Inspiration for this project came from browsing image data sets on the website Kaggle. I decided to undertake implementing a neural network for this project as a challenge having no experience with it. I believe this project captures a niche in the market for medical apps. It also has a positive health effect which encourages people to check their skins' health.

15.0 State of the Art

In my research so far into this project I have found some similar apps in the skin care field. One of the main ones is '*Skin Vision*' and has over 1.2 million downloads. It uses artificial intelligence to detect skin cancer in images that the user upload.

Another similar app is '*FirstDerm*'. This is more of a broader dermatology app than skin vision as it deals with all sorts of skin conditions. The user uploads an image, and it is analysed not by artificial intelligence but by a dermatologist.

My project will undertake to combine aspects from these applications. It will undertake to use the deep learning from the first mentioned app '*Skin Vision*', and the general dermatology platform with a community and in-person interaction from '*FirstDerm*'.

16.0 Technical Approach

The project plan in this document will be a Gantt chart, where the different parts of the project will be separated into different requirements and milestones. I will be using the Kanban methodology in the implementation of the software project. User stories are a valuable tool for this project as there will be two different types of users. I will also be implementing the project in scrum style sprints for aspects of each milestone. The first milestone of work will be the implementation of the neural network. Then next milestone will be implementing it in an android studio application. The next milestone will be the implementation of the database. The final milestone will be creating a modern and impressive user interface. The requirements for this project are as followed:

The regular user can:

Register/login:

The registration as of now will not require too much sensitive information. As of now it will require first name, last name, email address and age.

Neural Network:

They will then reach the home page, where they can select from the two main features of the application. One will be to upload an image of a section of their skin that is concern. The neural network will come back with a result with details about the match.

Set up appointment:

After they do the neural network image test they can select from a list of dermatologists registered on the application to set up an appointment with. They can select this on the appointments' activity. They will be able to request an appointment along with a message. The dermatologist can accept or reject the request. From then on further details will be made through the email accounts of both parties outside of the application.

News feed:

Regular users can post only questions on the news feed, where only dermatologist can respond with answers.

Dermatologist user can:

Register/login:

Users registering as dermatologist will be required to provide more information in the registration activity. They will be required to register first name, last name, age, their official company email, education qualifications, relevant employment history and other qualifications and the company they are representing currently.

News feed:

Dermatologist users will be the main posters on the news feed. They can post their own answers to common questions, answer user questions, and post promotions for their own practices.

Appointments:

They will be able to see appointment requests from regular users and will be able to accept or reject them. As listed above, detailing appointments will be done through the email accounts registered on the application.

17.0 Technical Details

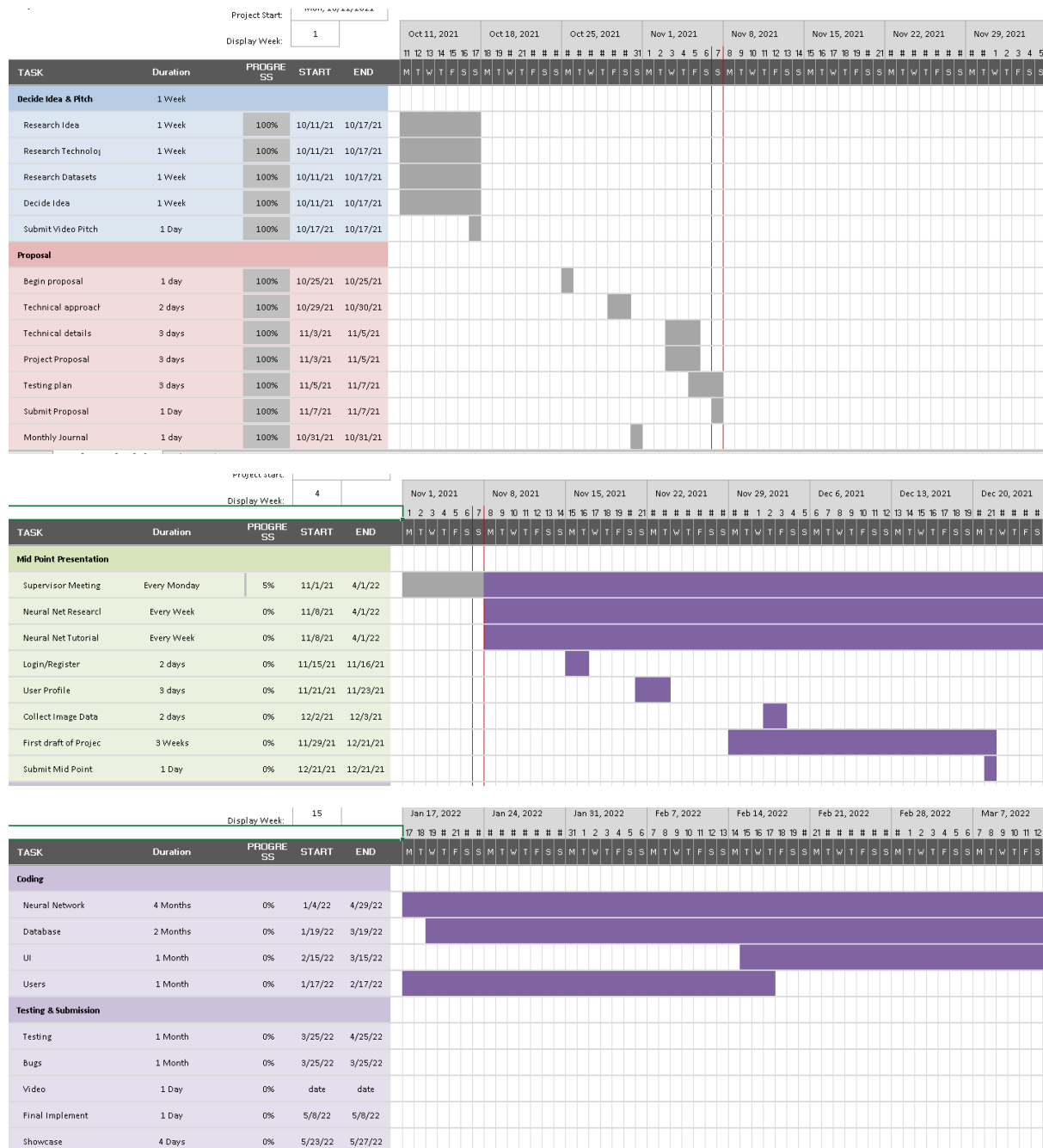
The programming languages used for this project will be python and java. The project will be broken down into two parts. The first part will be building a Siamese Neural Network which uses image similarity and classification to detect skin diseases. This will be done in python with the deep learning libraries TensorFlow, using the Keras API. The IDE for this will be Jupyter notebooks. Some important python libraries will be pip, numpy and lambda.

The android application will be written in Java. The most important dependency used will be TensorFlow lite, which will allow for the Neural network created to be used in the android application. The application will use Firebase as a database. Firebase authentication will be used to hold the users and passwords and the app's login and registration activities. Firebase storage will be used to hold images of photos users upload, both of images to be used for the neural network and for the professional account profile. Firebase real-time database will be used for the 'feed' section where the posts and promotions will be. The real-time database will be used for appointments as well. The activities will make use of RecyclerViews with adapter and model classes.

18.0 Special Resources Required

The Special resources required for this project will be the image data used to train the neural network. I will be collecting thousands of images of different skin conditions from different sources. The main source of image data I will be collecting from is Kaggle.

19.0 Project Plan



20.0 Testing

This project will use both system testing, unit testing and integration tests. An example of unit testing will be testing that the login and registration activities work correctly, and that two categories of users can be created and registered with separate details for the two categories of users.

Integration testing will be done after every commit in the application to test if each of the activities are not causing errors for other activities. For example, integration testing is needed to ensure that one feature or activity is not causing another activity to crash or not work properly. Example would be that the user uploads their image, and the image gets checked by the neural network using TensorFlow lite. Integration testing should ensure that these integrated activities flow seamlessly with no errors.

System testing will be the final set of tests done at the end of the project. This will make sure the whole project works together. User stories and detailed use cases will aid the system testing. Performance tests will be done for the time to execute functionality.

References:

Prasad, K., 2021. *Siamese Networks - Line by line explanation for beginners*. [online] Medium. Available at: <<https://towardsdatascience.com/siamese-networks-line-by-line-explanation-for-beginners-55b8be1d2fc6#:~:text=In%20case%20of%20a%20CNN,probably%20with%20a%20softmax%20function.&text=With%20siamese%20networks%2C%20it%20has,t%20have%20a%20softmax%20layer.>>> [Accessed 7 November 2021].

Saha, S., 2021. *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way*. [online] Medium. Available at: <<https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>> [Accessed 7 November 2021].

The Medical Futurist. 2021. *Digital Skin Care: Top 8 Dermatology Apps - The Medical Futurist*. [online] Available at: <<https://medicalfuturist.com/digital-skin-care-top-8-dermatology-apps/>> [Accessed 7 November 2021].

Rana, K., 2021. *System vs Integration Testing - Know the Difference*. [online] ArtOfTesting. Available at: <<https://artoftesting.com/difference-between-system-and-integration-testing>> [Accessed 7 November 2021].