National College of Ireland

Bachelor of Science (Honours) in Computing

Software Development

2018-2022

Adam Condon

X18392911

X18392911@student.ncirl.ie

Advanced Movie Store ‘Advocate’

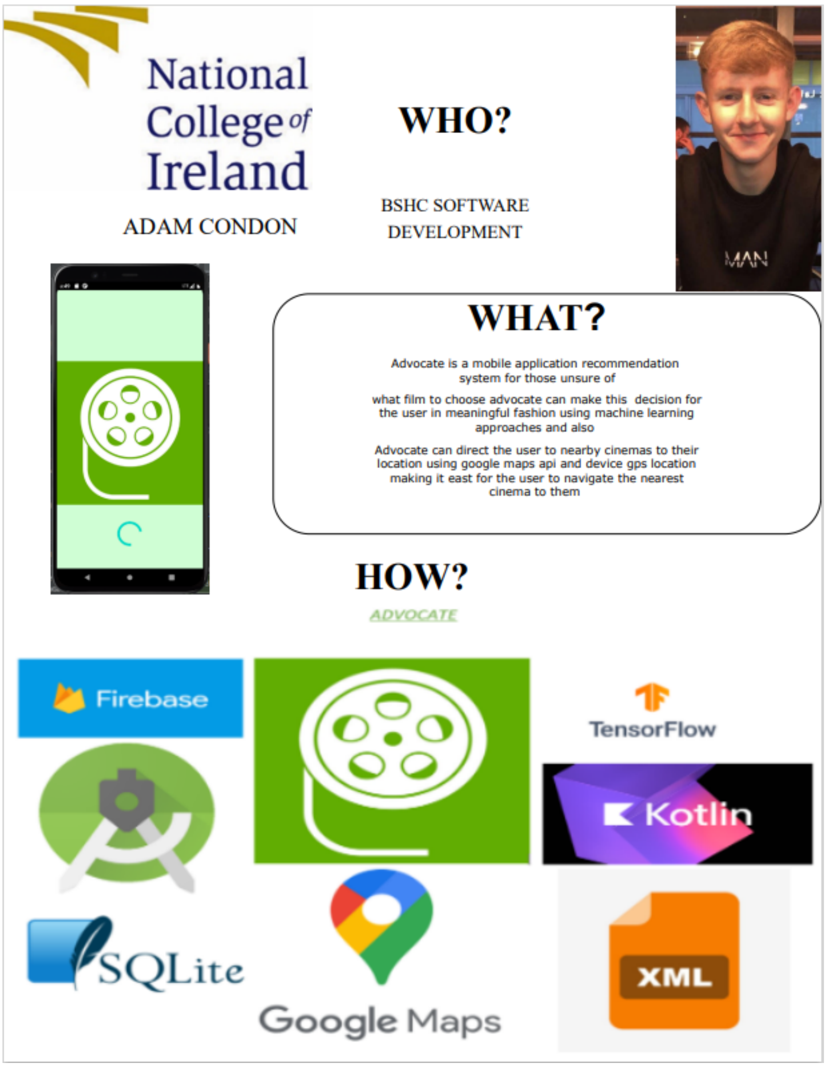
Technical Report

Contents

Executive Summary ... 3
1.0 Introduction ... 4
1.1. Background ... 4
1.2. Aims ... 4
1.3. Technology ... 4
1.3.1 Android Studio ... 4
1.3.2 Kotilin, XML, SQL, JSON ... 4
1.3.3 Sqlite ... 4
1.3.4 TMDB API ... 5
1.3.5 Google Maps API ... 5
1.3.6 TensorFlow lite and Google Big Query ... 5
1.3.7 Junit JVM Testing ... 5
1.3.8 Espresso Testing ... 5
1.3.9 Github ... 5
1.3.10 Structure ... 5
2.0 System ... 6
2.1 Requirements ... 6
2.1.1 Functional Requirements ... 7
2.1.2 Use Case Diagram of the System ... 8
2.1.3 User Registration ... 8
2.1.4 Description and Priority ... 8
2.1.5 Use Case ... 8
2.1.6 Recommednations Model ... 9
2.1.7 Non-Functional Requirements ... 17
2.1.8 Data Requirements ... 18
2.1.9 User Requirments ... 19
2.1.10 Enviornmental Requirements ... 20
2.1.11 Design Archtiecture ... 20
2.1.12 Usability Requirments ... 21
2.1.13 Design and Architecture ... 21
2.1.172 User Guide ... 22
3.0 Implementation ... 373

4.0 Graphical User Interface 41
4.1 Low Fidelity 41
4.2 Wireframes 474
4.3 High Fidelity 47
5.0 Testing 61
5.1 Advocate Consent Forms 61
5.2 Trunk Testing 61
5.3 5 Second Test 63
5.4 J-Unit Testing 64
5.5 Instrumented Testing 64
5.6 Users and Database 65
5.7 Manual Testing 67
5.8 Unit Testing 67
5.9 Performance Testing 67
5.10 Espresso Testing 68
5.11 Testing Results 68
6.0 Evaluation 70
6.1 Heuristic Evaluation 71
6.1.0 Error Prevention 71
6.1.1 Flexibility 72
6.1.2 Aesthetic Design 72
6.1.3 User Guide 72
7.0 Conclusions 73
8.0 Further Development or Research 73
9.0 Further Development or Research 73
10.0 Appendencies 75
10.1 Project Proposal 75
11.0 Objectives 96
12.0 Background 76
13.0 Research 77
Appendix 87

Poster



National College of Ireland

ADAM CONDON

WHO?

BSHC SOFTWARE DEVELOPMENT

WHAT?

Advocate is a mobile application recommendation system for those unsure of

what film to choose advocate can make this decision for the user in meaningful fashion using machine learning approaches and also

Advocate can direct the user to nearby cinemas to their location using google maps api and device gps location making it east for the user to navigate the nearest cinema to them

HOW?

ADVOCATE

Executive Summary

Recommendations are the automated solutions to a lot of indecisive individuals' problems nowadays. Couples' families and individuals that sit down after a week's hard work, family vacation or date night tend to struggle figuring out what films to watch. Advocate does this for them without any effort form the user and provides them with information of each individual movies that they are recommended as well as what's popular, best rated, and upcoming in theatres. Users can filter movies at their own leisure to any genre of preference.

Advocate is an Online Movie platform which allows users to identify movies they would like to watch. Users can set up a secure account, find movies they like, receive recommendations for movies based on their preference and find out how they can access these movies. Advocate allows users to buy movies online or in store. The application contains a dataset of movies. Users view the list and select or like a movie which is then stored on a personal like list. Recommendations are generated based on the liked movies. Advocate also provides a google maps API System that allows users to locate local Cinemas and shops where they can purchase their preferred movies physically or view them. Advocate includes a payment system allowing users to leave a tip if they want to.

1.0 Introduction

1.1. Background

This projected idea occurred to me for my love and passion for movies in my leisure time. This advanced system will allow me and users to learn what people like based off the TMDB API and user movie choices. This idea gave me motivation to create a movie recommendation system based off current and past movies extracting live data using the TMDB API and comparing films based on their actors, genres, themes, ratings, and other attributes to base recommendations to users of advocate to watch. Also, I have never witnessed a movie app which will navigate the user to nearby stores or cinemas where users can also vie and purchase movies they like, and this inspired me to create a google maps API capturing the user's location and presenting nearby cinemas to that location.

1.2. Aims

The projects aim to allow users to set up an account, find movies they like and how they can access the movies and even buy them. The apps aim is to recommend movies to each individual user based on the movies they click into. This is perfect for when a user is unsure of what movie to watch and needs a quick recommendation. The app will then navigate the user to a destination where they can watch the movie in real life at the nearest cinema or purchase one at the closest DVD store to their location. Technology

1.3.1 Android Studio

This will be implemented as an Android mobile application using Android Studio IDE.

Android studio is a mobile app development environment for googles Android operating systems. I can run, debug, test and track the performance of my application on this IDE.

1.3.2 Kotilin, XML, SQL, JSON

The Code will be Implemented in Kotilin, XML, SQL, and JSON. Java is also supported with Android Studio but due to personal preference I preferred to challenge myself more and use Kotilin to develop my skills more in this language and because how appealing the language occurred to me in recent studies throughout college and Work experience.

1.3.3 Sqlite

Is an open-source database used to store data inside the user's device in the form of a text file CRUD operations, API's and other interactions can all be carried out with SQLite Database. This was also used for personal preferences over NOSQL databases due to my experience of SQL inside and outside of college.

1.3.4 TMDB API

TMDB movies dataset will be used to extract large amounts of live data which will be displayed in fragments and the recycler views, this will allow the user to click the relevant movies they like. As for The Recommendations System this will be based off the users' choices of movie and the Attributes to go with it such as genre, names, and movie type will factor the recommendations and based off these similarities recommendations are extracted and pulled from the TMDB database and Displayed beneath the clicked movie.

1.3.5 Google Maps API

The google maps API Feature will be added using Googles API in Kotilin, retrieving a relevant API key to activate the custom map. A full map and markers of nearby cinemas with their relevant details will be displayed once the user permits Advocate access to their location withing the app.

1.3.6 TensorFlow lite and Google Big Query

Is a high-quality mobile library that allows for the deployment of models? TensorFlow allows to a model to be trained using machine learning based off data you enter. A TMDB dataset was used to insert into the TensorFlow lite Recommendations model which also used Google Analytics and Big Query to log events from the user each time they made a change in the app in this case "Liking movies". The more interaction the users make the more the model can be trained, and the more meaningful the users' recommendations will become.

1.3.7 Junit JVM Testing

Is a testing environment designed for Kotilin programming language This allows me to test my application programmatically and automatically once the tests have been written correctly This approach removes the chances of human error manually testing the application J Unit will allow me to test the navigation of buttons and text field inputs such as usernames and passwords.

1.3.8 Espresso testing

Is an integrated testing environment with Android Studio this testing technique allows me to create tests that mock the user interaction with the UI of my application?

1.3.9 Github

Allows me to push pull and delete any changes I make to my app throughout the entire project development stages. Also having the project stored on the cloud allows me to access my code on any device and have it in a safe public environment where I can redownload my code locally or other people such as lectures can review the code and download it if necessary

1.3.10 Structure

- An Introduction is provided to 'Advocate' and a little about the app as a taster.
- We then move into the system and each feature the system will include and what requirements each one has and specializes in.
- Clear use cases are used to display what each feature will consist of and their possible outcomes and conditions.
- The strategies, planning and approaches are all defined and explained through both descriptions and charts.
- Aspects such as the design, architecture, tests, and Implementations are all explained in great clear detail to represent the projects goals and expected outcomes.
- What I would do differently I were to start over or had more time
- References used
- External documents noted and displayed

2.0 System

2.1. REQUIREMENTS

2.1.1. Functional Requirements

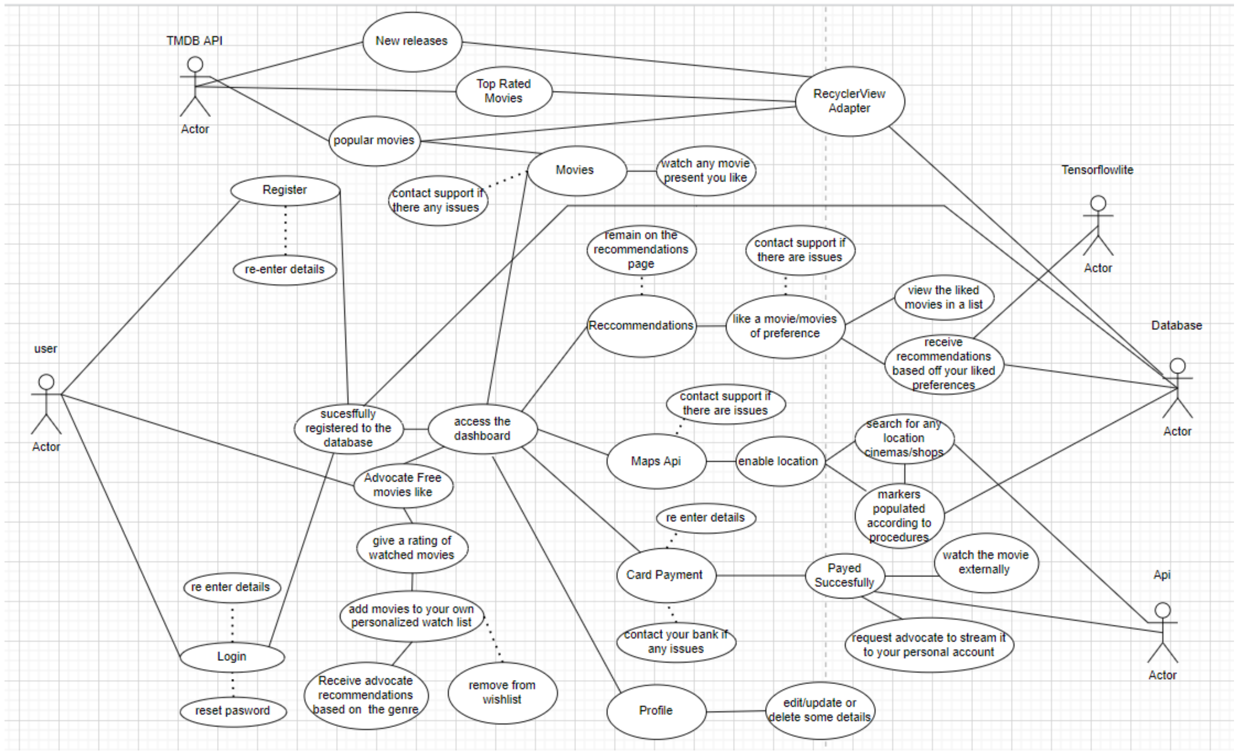
Recommendations System: Recommends movies across individual account based off what movies the user likes providing meaningful recommendations to movies of similar interest. This will require the user to like movies from the movie list provided in the main activity allowing the recommendation system to commence. The movie list was extracted from TMDB movies in JSON format the data is read to the movies List file and is eligible for the user to like any movie that is presented to them. The user can also unlike a movie stopping the system from saving them in the liked list and recommending movies based off them. Each of the 3 lists can be accessed with just a swipe of a finger.

Google maps API Integration: Allows user to access a custom map with Shops and cinemas nearby their location once granted by their device this allows the user to see where they can watch/buy movies and allow them to further buy movies from the system they find of interest. It requires the user to either to accept the permission of their location or manually enter their location with longitude and latitude values. The map will populate shops and cinemas nearby with maps and their distance from the user's location.

Login and Registration System: Allows the user to create an individual account to allow for individual meaningful recommendations once the user has accessed the app. The user details can be stored on the google firebase authenticator and be memorised for each time they login. The user will not have to Login every time they access advocate unless they uninstall or logout of the app themselves. If user forget their password this can be reset through support and the user can pick a new password of choice.

View and buy movies: Customers will be offered movies to buy from external sources such as YouTube and they can decide which ones they can choose to watch at their own leisure and should receive a notification upon any payments they make to their email for confirmation. The user will have the option to stream the movie to Advocate or stream it externally. A user can also view update or delete some of their personal profile information at their own leisure.

2.1.2 Use Case Diagram of the System



TMDB API
Actor
New releases
Top Rated Movies
popular movies
RecyclerView Adapter
Movies
watch any movie present you like
contact support if there any issues
Register
re-enter details
user
Actor
remain on the recommendations page
contact support if there are issues
Reccommendations
like a movie/movies of preference
view the liked movies in a list
receive recommendations based off your liked preferences
Tensorflowlite
Actor
Database
Actor
sucessfully registered to the database
access the dashboard
Advocate Free movies like
contact support if there are issues
Maps Api
enable location
search for any location cinemas/shops
markers populated according to procedures
give a rating of watched movies
re enter details
Card Payment
Payed Succesfully
watch the movie externally
Api
Actor
add movies to your own personalized watch list
contact your bank if any issues
request advocate to stream it to your personal account
re enter details
Login
reset pasword
Receive advocate recommendations based on the genre
remove from wishlist
Profile
edit/update or delete some details

2.1.3 User Registration

The user will be required to Register an account with a Valid Email, Password and Name. Once the User has set up an account, they will then proceed to the apps main Activity. The user will not be required to Login again until they Logout of the app themselves or the app is Uninstalled on their device.

2.1.4 Description & Priority

This requirement is an essential part of this project because it allows the system to separate different individuals across different accounts. If there was no Registration and Login system put in place everyone would receive the same movie recommendations and be able to access the same aspects as every other user can in the app. This would not have meaningful recommendations to individual accounts/users.

2.1.5 Use Case

Firebase Login and Registration.

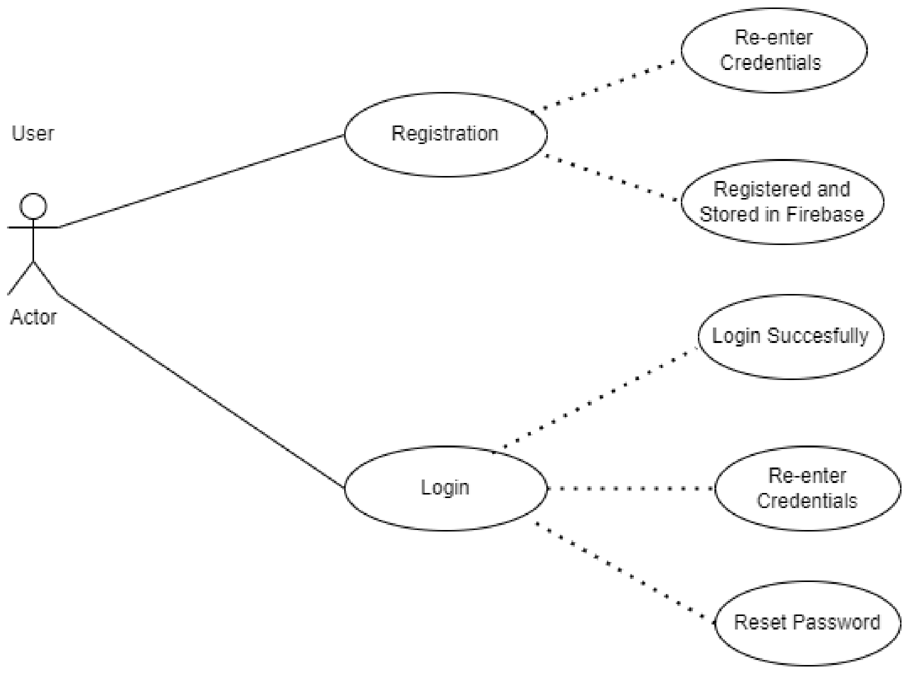
Scope

The scope of this use case is to Successfully register and log the user into the application once the correct steps are followed. It is also designed to disallow users' entry to the app if invalid details are entered especially malicious acts such as SQL injection to the provided text boxes. It will store the Users details in the Google Firebase Engine once registered successfully.

Description

Users Account details and their individual accounts. It shows the outcomes of a User attempting to Register/Login into the system. Once a user has done this successfully their details will be stored in the google firebase authenticator via google cloud and will be remembered each time a user logs in until their account is deleted.

Use Case Diagram



Flow Description

- If the User Registers Successfully, they can the App if they are unsuccessful, they will have to re-enter the credentials until they successfully create an account.
- Once the account is created successfully the user can then use the same credentials they used to register to login and access the app at their own leisure.

Precondition

- The Precondition to this app Is too successfully create an account with a valid email, name, and password.
- Once the Login is accessed the user then must login successfully otherwise request for a password reset.

Activation

- The user may start when the Main Activity is fully loaded after the splash activity of the app's logo is presented.

Main flow

- The System identifies the user's mouse/on click to the registration fields
- Allows them to place in their credentials successfully and register to 'Advocate'
- Once the user is registered, they will successfully login to their account and be greeted to the main activity.

Alternate flow

- User will be provided a user guide if they cannot login successfully, if this still does not help the user can contact the support team (Me) and then the account can be created.
- Once this is created a reset password link will be sent to the users' email where they can set a new password or use the one created from support.

Exceptional flow

- Account is created for the user 1 to 1 and is helped logging and register each step of the way via google meets.

Termination

- The Registration and Logging system will not terminate until the user successfully logs in or closes/deletes the app if you do not make it passed this part you cannot access the application.

Post condition

- The system goes into a waiting state and the user can access the apps features and receive their individual app experience.

Recommendations ML Section

Using AI and Machine Learning the users will receive recommended films to their preference in Advocate in just a click of a button.

2.1.6 Recommendations Model

The recommendations AI will be based off a movie list which was added to the app using an TMDB dataset. The user will have options of hundreds of movies to choose from and will "Like" movies from the list to their preference. Based off the liked list this will be sent to google big query for analysis of the data and tensor flow lite to train the recommendations system to give meaning full recommendations back to the user. This is the main priority of developing Advocate.

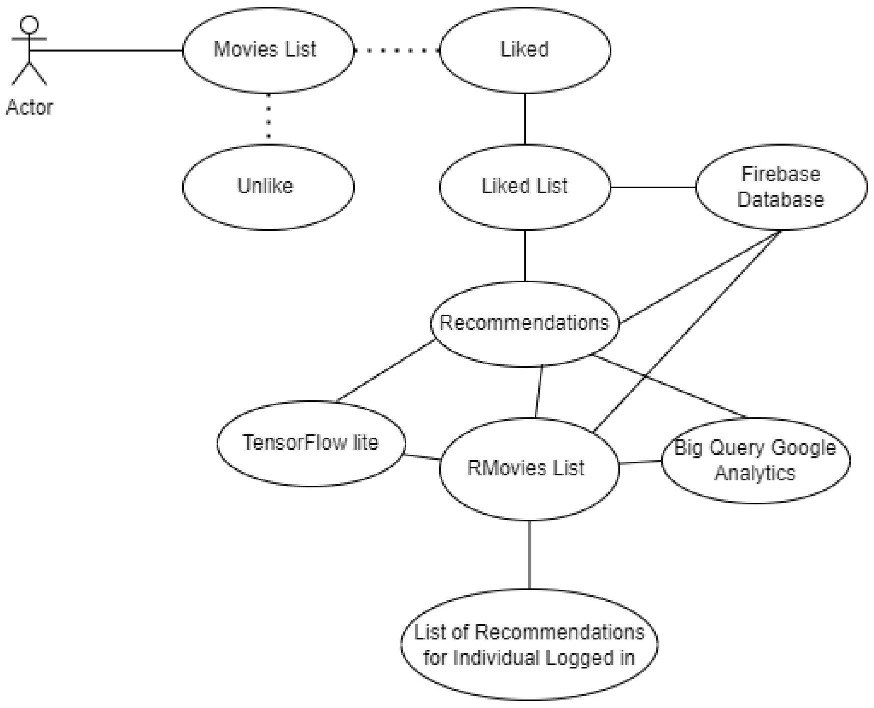
Scope

The scope of this feature is to successfully recommend movies based on a logged in users' decision. The decision that will be taken into consideration will be the movies the user likes. when the user likes a movie from the movies list, they will then be stored to the firebase database and a liked list, also the system will use TensorFlow lite and Google Big Query analytics to analyse the data successfully and provide meaningful recommendations back to the user's account. If the user unlike a movie this will be then removed from the liked list and may change the recommendations prior to the unlike.

Description

The Features aim is to allow users to set up an account find movies they like and how they can access the movies and even buy them. The apps aim is to learn what movies each individual user likes and recommend movies to them based on their decisions. This is perfect for when a person is unsure of what to watch and needs a quick recommendation.

Use Case Diagram



Flow Description

Once the user has logged in, they can access this feature, The user must like movie to receive recommendations or else none will be provided, and an error will be printed to the users. Users can like and unlike movies at their own leisure search for movies or logout.

Pre-Condition

Once the user has logged in, they can access this feature. User can like and unlike movies or search for preferable movies.

Main Flow

User likes a movie/movie the liked movies are stored in a list and database and recommendations are fetched by using this data.

Alternate Flow

User Dislikes movies or likes none the application just provides lists of movies.

Exceptional Flow

User likes a movie/movie the liked movies are stored in a list and database and recommendations are fetched by using this data.

Termination

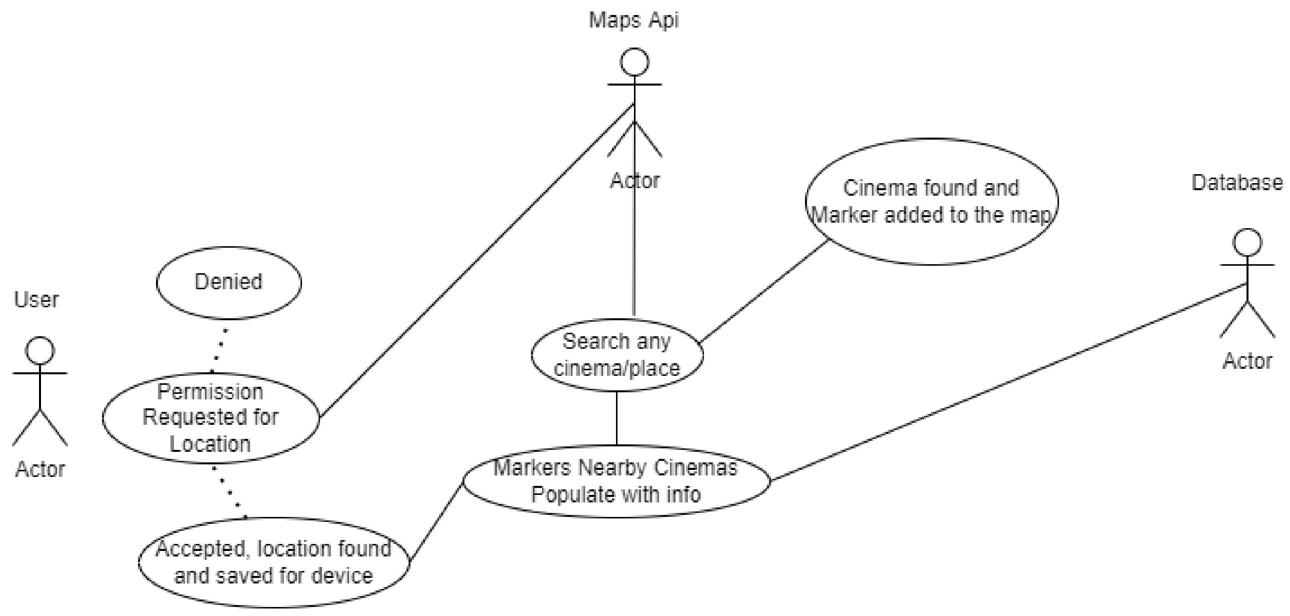
The app is closed or uninstalled

Requirement 3

Google Maps API Local Movie Shops or Cinemas

Integration with google maps Api making a custom map that locates movie stores and cinemas nearby the user's location**.**

Use Case Diagram



Maps Api
Actor
Cinema found and Marker added to the map
Database
Actor
User
Actor
Denied
Permission Requested for Location
Search any cinema/place
Markers Nearby Cinemas Populate with info
Accepted, location found and saved for device

Description

Google maps API integration that allows the user to access a custom google Maps API where they can allow access to their location add their location/house to the map as a marker and view Cinemas, Movie shops that are nearby and at their convenience to buy movies they want to watch.

Scope

The main scope is to provide Movie shops that are nearby and at a user's convenience to buy movies they want to watch. It also allows the user to add their own locations and save them to their relevant accounts. This allows the user to see the closest cinema/movie shop closest to their location with just the click of a button.

Main Flow

Allows the app access to their location and populates the maps based off their location

Alternate Flow

Does not allow the app access to their location, inputs their location Manually and can then see what stores and places it is around

Exceptional Flow

The User allows the permission of their location, but it cannot be found (Runs out of signal or is in an uncharted area). The maps should load until a location is found.

Flow Description

The user clicks on the map’s activity button, they are navigated to the Maps activity, the user is asked for their location permission, the user can accept or enter their location manually through longitude latitude format for accuracy. The relevant stores and cinemas nearby should appear as expected.

Pre-Conditions

User is prompted with permissions question; user has the option to input markers and values into the map.

Post conditions

The user receives a populated map of markers of the nearest shops and cinemas near them how far they are links to their website.

Terminations

The app is closed or uninstalled

Paying for Movies

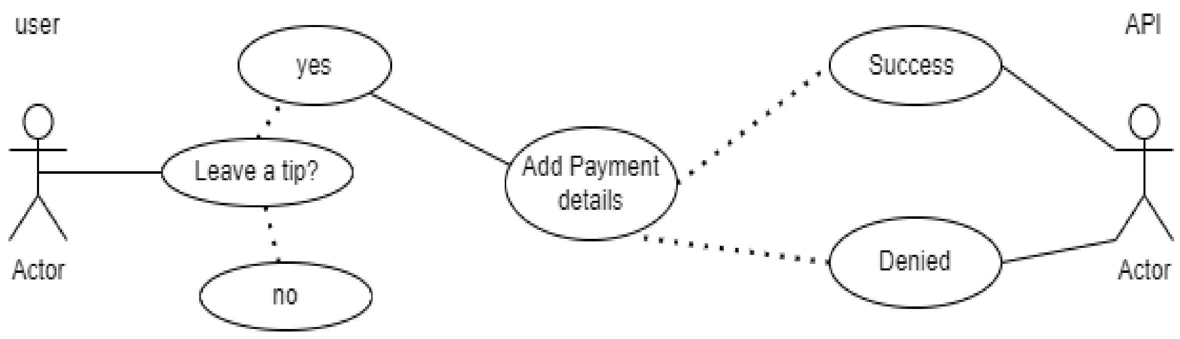
Scope

The scope of this requirement is to allow users to buy a specific movie they want that may not be on the Advocate database.

Description

Credit card Payment system which allows users to successfully pay an amount of their choice. This is treated as a tip for Advocate for future development and endeavours. The user can enter their card details either manually or automatically by using their device camera and allowing google pay and wallet API to recognise the card details through text recognition.

Use Case



Pre-Condition

Users have selected a movie and want to pay for the movie selected the user must have a valid credit card/account to pay and pay for it externally.

Post condition

The user should receive a notification upon their purchase and receive the movie they wanted from Advocate itself or external resources. The user will have an opportunity to either rent or buy a movie.

Main Flow

The user selects a movie, the user is brought to a source of the, live where the movie can be purchased, the movie is purchased via direct debit.

Alternate Flow

The User Deselects a movie or makes an unsuccessful payment due to bank issues or insufficient funds an error will occur, and an invalid transaction will be recorded.

Exceptional Flow

The user would like to change the movie selected and choose another and continues to search for other movies while exiting the transaction stage and then choosing another movie to purchase.

Flow Description

The user should choose a film at their own leisure that they want to buy (If it is already not provided), pay the fee for the movie either to advocate or an external movie resource. The user can then watch the movie in advocate, or an external resource based on their preference. The user will have an opportunity to either rent or buy a movie.

Watch Movies

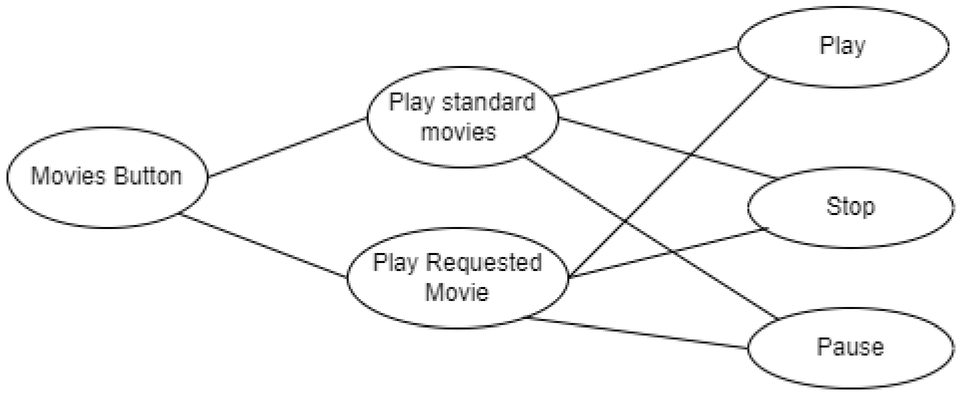
Scope

The scope of this requirement is to allow users to view a movie they bought specifically or just watch the standard movies provided by advocate.

Description

This activity will allow the user to watch movies at their own leisure and enjoy advocates movie experience.

Use case



Pre-condition

User is required to buy a movie if it is not already on the advocate app. The user can access the movies once they have successfully made an account.

Post Condition

Users can access films at their own leisure with the help od Advocate making their decisions easier and support team ensuring the experience is superb**.**

Main Flow

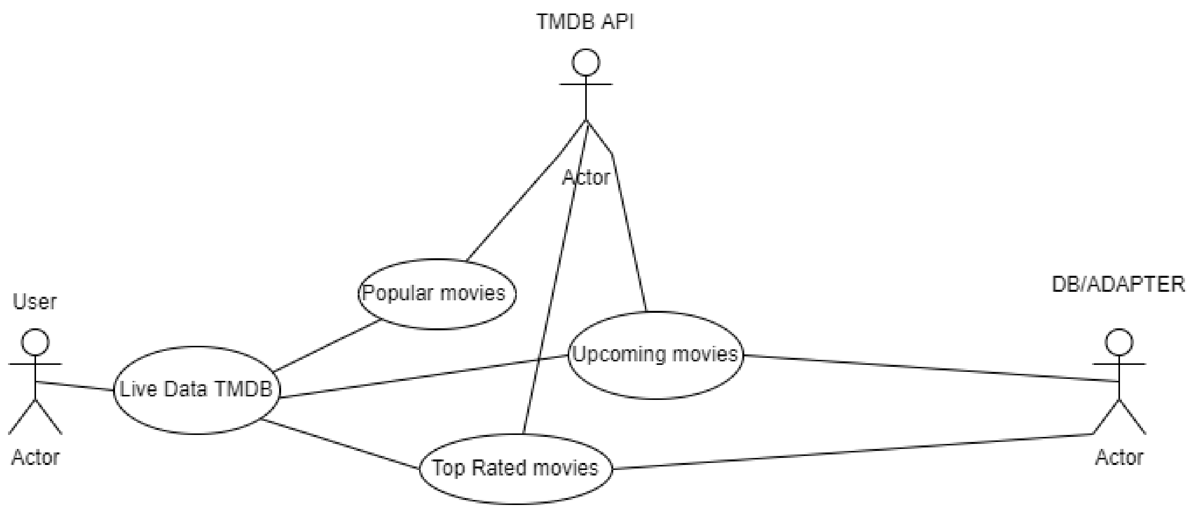
Once the movie is added/stored in the users account the user can watch the movie.

Alternate Flow

The user can watch the movie on an external source where they could have bought it from

Use Case:

Live Data from TMDB For the user to see Top rated, Popular and upcoming movies which changes daily depending on user interaction.



Scope

The scope of this requirement is to allow users to view top rated, upcoming and popular movies at the moment extracted from the TMDB API and Displayed on advocate and saved on a local database.

Description

This activity will allow the user to see live movies that are currently in Cinemas, have the highest rating based off user ratings and also popular movies at the moment such as latest movies and popular trends.

Pre-condition

User is required to have a stable Wi-Fi connection and navigate to whatever type of data they want to see.

Post Condition

Users can access films at their own leisure with the help of Advocate to display and title each movie with some of their details also**.**

Main Flow

The user can access the Live data and will remain present.

Alternate Flow

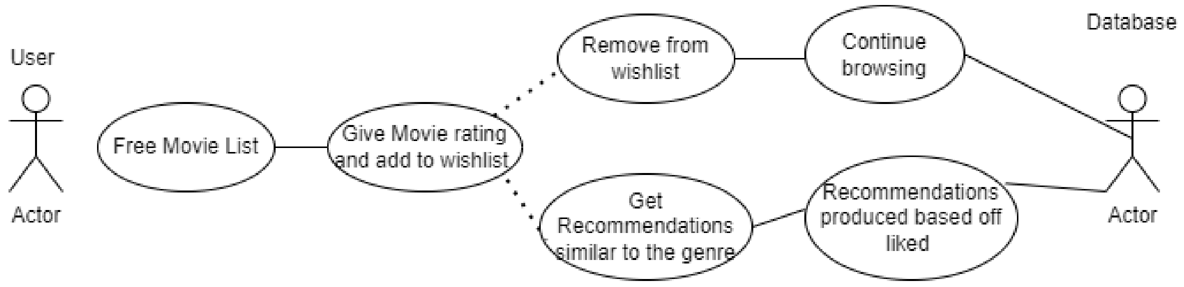
The user loses Wi-Fi connection, and the API has trouble gathering the Movies and their data.

Exceptional Flow

The user looks through the movies and either watches it on advocate, checks local cinemas available through our API or buys it externally.

Use Case

Advocate Free Movie and Genre Recommendations:


User
Actor
Free Movie List
Give Movie rating and add to wishlist
Remove from wishlist
Continue browsing
Get Recommendations similar to the genre
Recommendations produced based off liked
Database
Actor

Scope

The scope of this requirement is to allow users to view all the free movies advocate stores for users to watch and allow to rate them and them to a Wishlist based off what they liked watching. From here a user can remove the movies from their Wishlist or get more recommendations from what genres they like/dislike and can watch them through advocate in the movie’s activity.

Description

This activity will allow the user to see movies that are free to access and get recommendations based off liked and disliked genres as well as storing movies they like in a Wishlist in their personal accounts.

Pre-condition

User is required to give the movie a rating out of 5 before adding any movie to their Wishlist then either click remove or recommendations button to either remove their list or receive more recommendations.

Post Condition

Users can access their Wishlist and recommendations at any time as well as rate and view any free movie with their title and Description.

Main Flow

The user can access their own lists every time they login and see all free movie as well as recommendations.

Alternate Flow

The user doesn’t rate or add any movies to a list and just views the movie details.

Exceptional Flow

The user doesn't avail of free movie and uses other parts of the app.

2.1.7 Non-Functional Requirements

1) Recommendations Performance

Is important in this application. Advocate must have wifi and interact correctly with the TMDB API to receive meaningful recommendations off each movie the user selects

2) Google Maps Api Permission Performance

must correctly accept the user's permission for their location or else it is illegal to track the user's location once they accept the user's decision and then find their local device GPS.

3) Smooth Interface

The user should be eligible to navigate through the system successfully and be able to clearly determine between certain activities with buttons, text and titles clearly dividing each topic and activity in the application. Also, helpful images and bright appealing colours attract the user's attention and really help the fluidity of the app. It should take no longer then ten minutes to understand advocate's interface.

4) Availability

Availability is important for my application as a lot of the app relies on live data which constantly changes daily from User reviews, ratings, popularity, top movies, and new upcoming movies scheduled to be in the cinema in coming weeks. A bad Wi-Fi connection or no Wi-Fi can result in this data not loading or populating to date.

5) Scalability

Is also quite important for advocate regarding requests and responses required of the user firstly one being the Login and Registration System which requires the user to create an account and login saving their details to the advocate database. Once a user is logged in they have their own personal liked list of movies that's saved to their account and also on the google maps API feature their own personal device location is tracked with permission firstly granted so wherever the user goes the device will pick up nearby cinemas in a 10km radius.

2.1.8 Data Requirements

The Application will use Googles Big Query Data and Recommendation AI to determine user recommendations personally according to their accounts choices and liked movies list. TMDB dataset was extracted and Used to Populate the movies List page formatted in JSON data. The user's data upon register and login will be saved to the app's firebase authenticator.

2.1.9 User Requirements

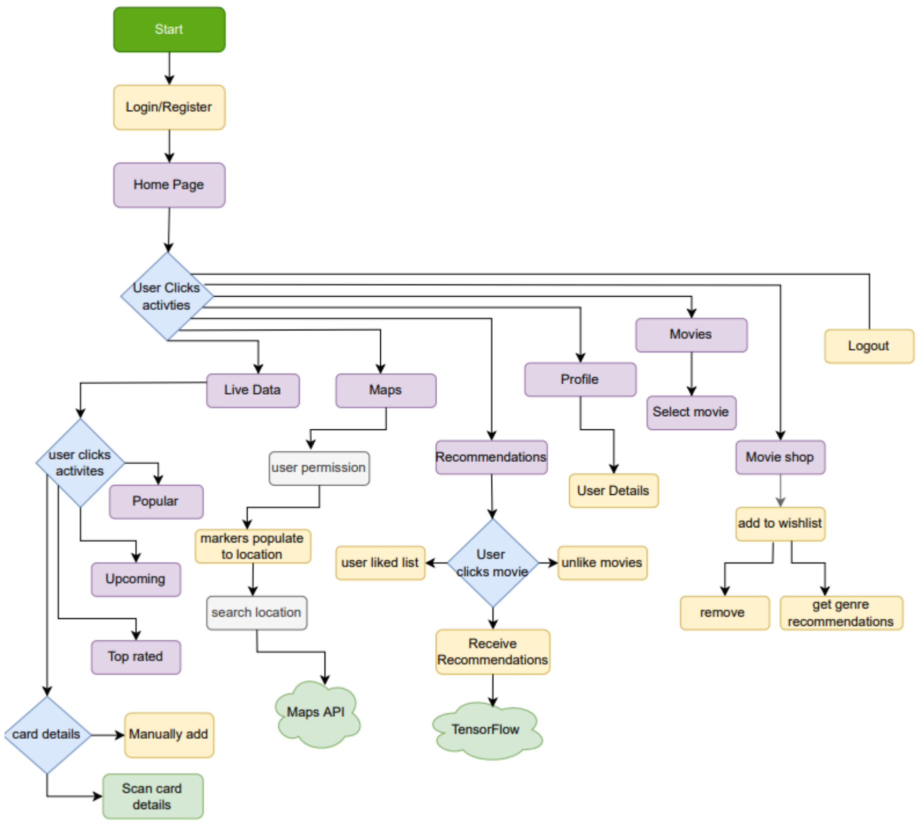
The User is Required to have a functioning email address, name and is required to choose whatever movies they want to watch and like. The user also may or not accept their location but are required to accept to avail of the Maps API activity. The user is required to have Google pay if they would like to leave a tip. Or function online paying service if they want to purchase certain movies externally. The User should also be familiar with general application navigation. The user should also always have a stable Wi-Fi or mobile data connection to retrieve live data and recommendations.

2.1.10 Environmental Requirements

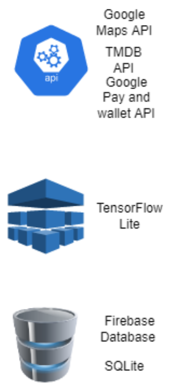
All Users emails, name, and Hashed passwords (Not Visible to admins or anyone else) will be stored on the Google Firebase database along with their movie choices and liked movies. Their location also may be captured upon the user's acceptance.

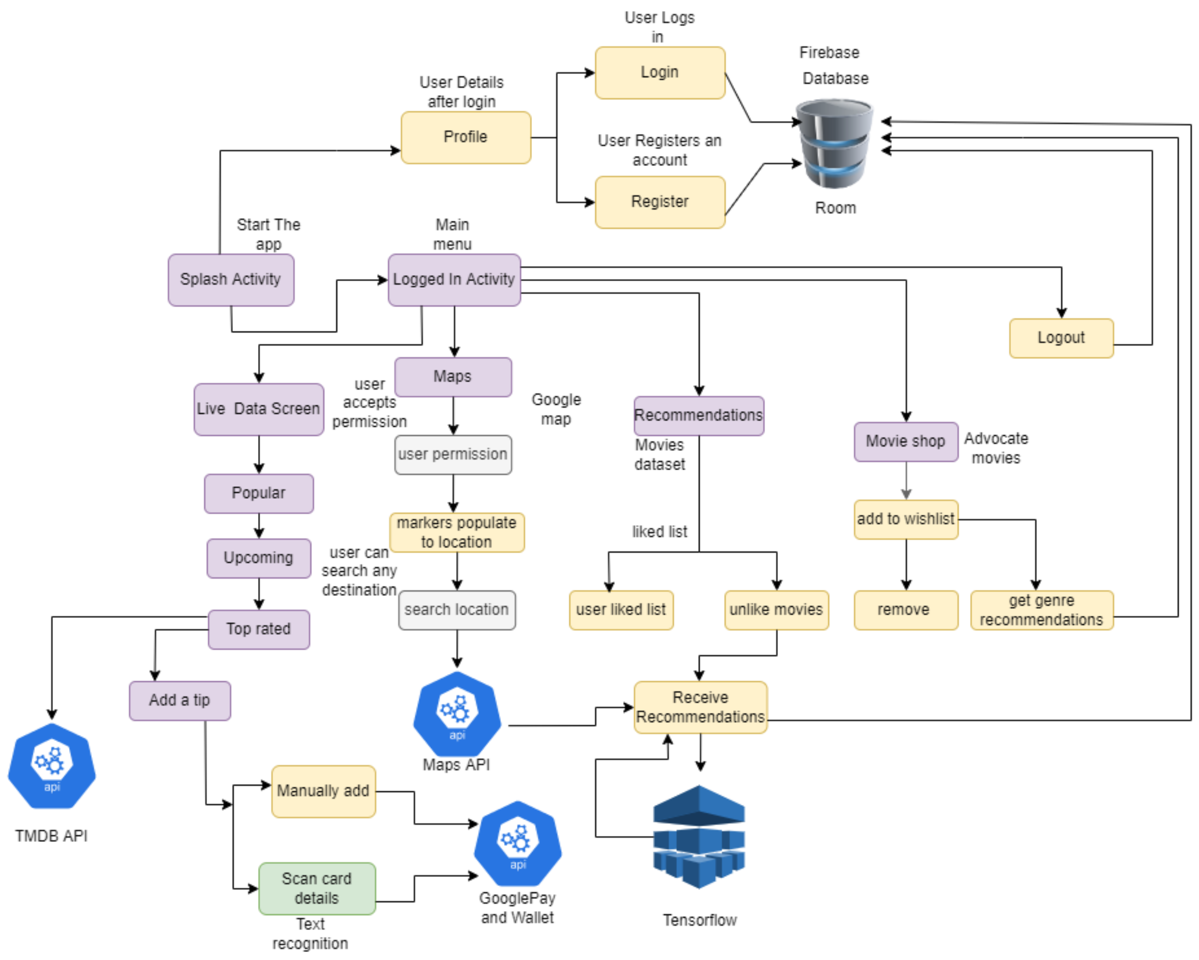
2.1.11 Design Architecture

Activity Flow Diagram



Activity Architectural Structured Diagram





2.1.12 Usability Requirements

The user can have zero to little UI experience to access the app. Once the register/login system is complete successfully the user will be directed with ease upon movie selection and what they wish to do in the app. This should be a straightforward app to access, and the user can view which every aspect of the app as they please.

2.1.13 Design & Architecture

The main algorithms implemented will be googles AI Recommendation system, Google Maps API integration in Kotilin Programming language. Advocate's recommendation system/process is split up into 3 parts. 1 being the list of movies extracted using a recycler view. This will give the user options to choose from by listing hundreds of movies and giving them the option to like and unlike them later if needed or if they were pressed by accident. From there all the liked movies will be stored in the next section "liked" where all the liked movies can be viewed by the user. This will then allow the final section "Recommendations" to decide what recommendations to make based off what the user likes. These will also be displayed in a list like the Liked movies and movies.

Google Maps API will also be developed through Kotilin programming language allowing the user to view a map with its populated markers and information also using android studios maps activity with the relevant google XML implementations. This will look like google maps with advocates own custom twist but the same foundation architecture once the user provides their location to the Maps functions will then populate the map with nearby movie theatres and their details.

Payment system will involve text boxes for the user to provide their relevant details also designed using Kotilin and XMl. A notification will be produced through googles notification API once a successful transaction is fully complete. Allow users to provide a tip to advocate developers which can benefit the apps development.

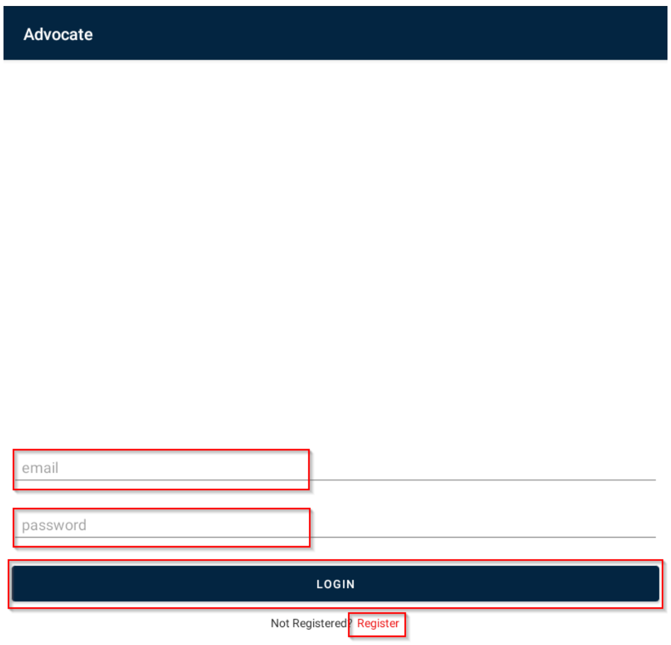
Live data extracted from TMDB involves Retrofit kit And Kotilin Adapters and Recycler views which Display the data through the Api acting as a client and stored in a local Database. The activities are also designed using a custom xml format after the data is extracted and queried using an api key query and parses correctly.

Free Movies Ratings and Recommendations allows the user to access all of advocates free movie details allowing them to rate the movies after watching them and adding them to their own watchlist and getting more recommendations based off the genres they like like movies in their Watchlist and the beauty of this is that they movies recommended are on the Application to be viewed rather than finding the movie externally.

2.1.14 User Guide

User guide to Register and Login

When Advocate is Opened You will be greeted to this Login Page. If you are an existing user, you can simply enter your registered email and password into the text fields below and click the Login button. If you do not have an account and need to create one you can simply navigate to the Register page by clicking the register link under the login button.

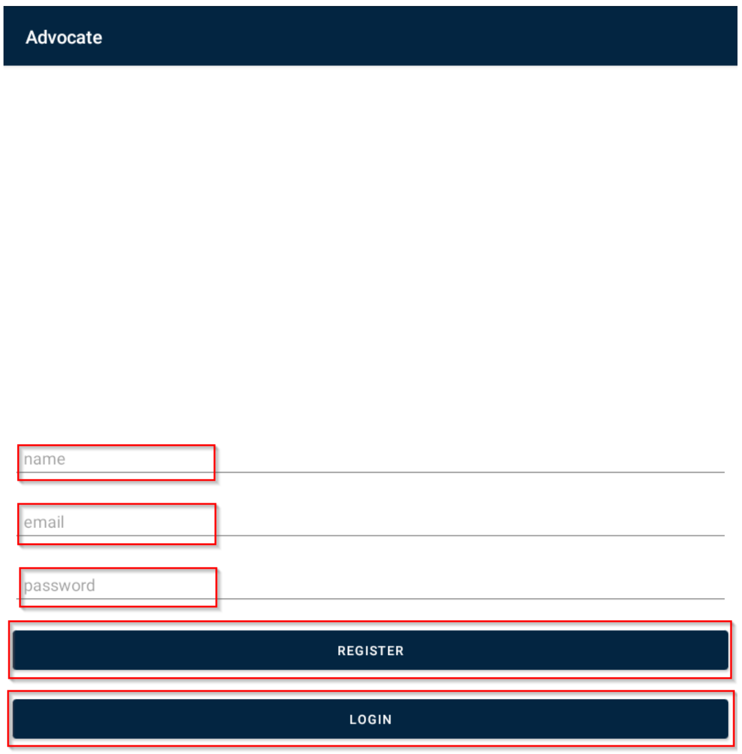


When registering an account, you must enter a valid Name Email and Password:

Valid Name = “Adam” this must be a name not a number decimal or special character

Valid Email = adam.condon@phcol.ie this must contain an @ symbol and a . to be a valid email.

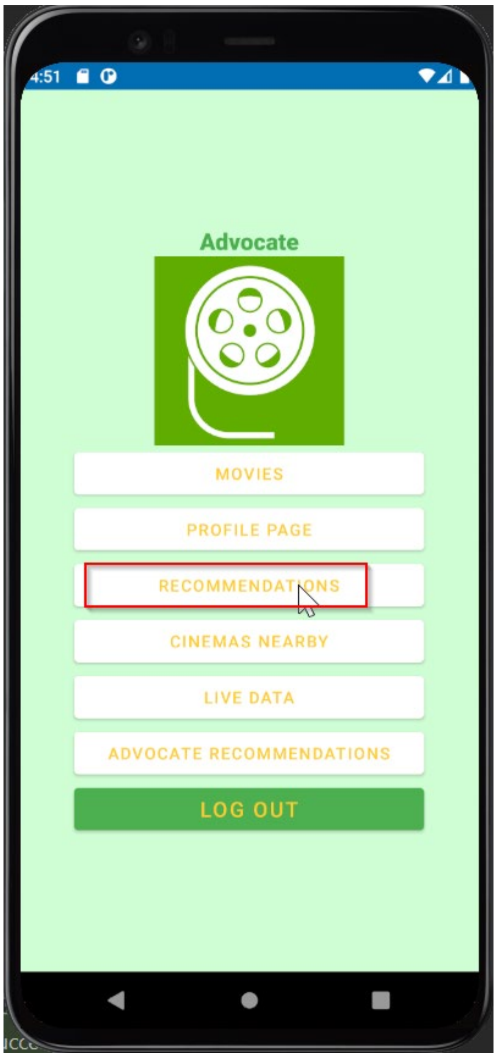
Valid Password = “test1234” must exceed 6 characters, numbers and special characters are advised to prevent account hacks.



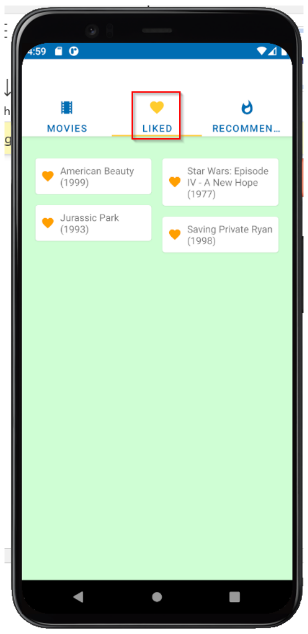
To Register Successfully you must enter the details explained previously into the correct fields where it prompts your name, email, and password for a new Advocate account. Once you have done this the user can click “Register” which will successfully register the user and then the next step is to then click the “Login” button to navigate to the login page where the user can enter the newly created credentials. Id there are any issues in the creation of this account please read over this manual or contact x18392911@student.ncirl.ie for assistance.

User Guide to Like Movies get their liked Movies and receive Advocates Recommendations through the TensorFlow lite model.

Once the user Is Registered, they will have access to the Main Home Page which stores all the buttons for the app navigation. The user should click the Recommendations button.

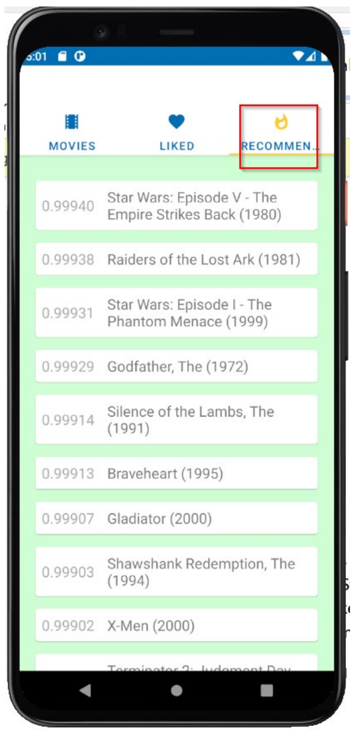


From here the user should click the love heart icon on whatever movies they like form the movies list. The more they like the better as they will receive more meaning full recommendations based off what they like, and it will train the TensorFlow lite model.



American Beauty (1999)
Star Wars: Episode IV - A New Hope (1977)
Jurassic Park (1993)
Saving Private Ryan (1998)

From here the user can then access their own personalised liked list for this example I chose the movie American beauty, Star Wars, Jurassic Park, and Saving Private Ryan. Now that we have a personalised liked list, we can now get personalised recommendations based off these movies. To access this the user must click the recommendations list icon.

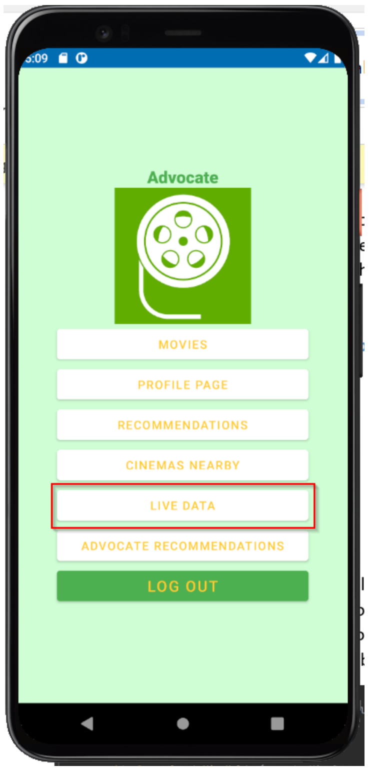


```
0.99940  Star Wars: Episode V - The Empire Strikes Back (1980)
0.99938  Raiders of the Lost Ark (1981)
0.99931  Star Wars: Episode I - The Phantom Menace (1999)
0.99929  Godfather, The (1972)
0.99914  Silence of the Lambs, The (1991)
0.99913  Braveheart (1995)
0.99907  Gladiator (2000)
0.99903  Shawshank Redemption, The (1994)
0.99902  X-Men (2000)
         Terminator 2: Judgment Day
```

If the user wants to change their Recommendations, they can do this by picking more movies or removing movies from their list and the data will be changed based off the changes by the TensorFlow lite machine learning model.

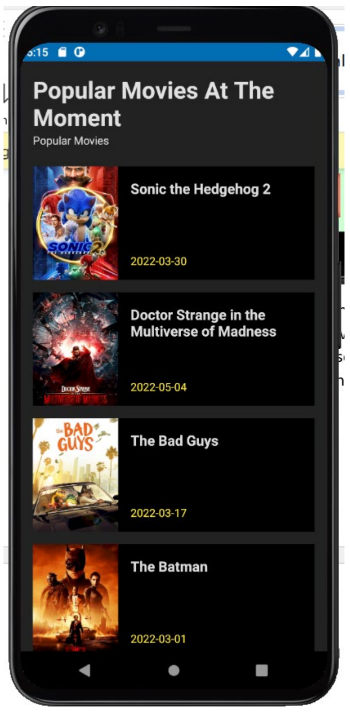
How the user can View Live data from the TMDB API and Access Popular, Top Rated and Upcoming Movies:

Once the user is logged in to their Advocate account, they should click the "Live data" Button To access the Live data API aspects of Advocate.



From here when the user clicks on the "Live Data" Button they will be navigated to a sub menu where they can then choose which activity, they want to view either Popular, Top Rated or Upcoming Movies. If the user is enjoying the app, they can also leave advocate development team a tip to help with future productions.

Popular Movies:


Popular Movies At The Moment

Popular Movies

Sonic the Hedgehog 2

2022-03-30

Doctor Strange in the Multiverse of Madness

2022-05-04

The Bad Guys

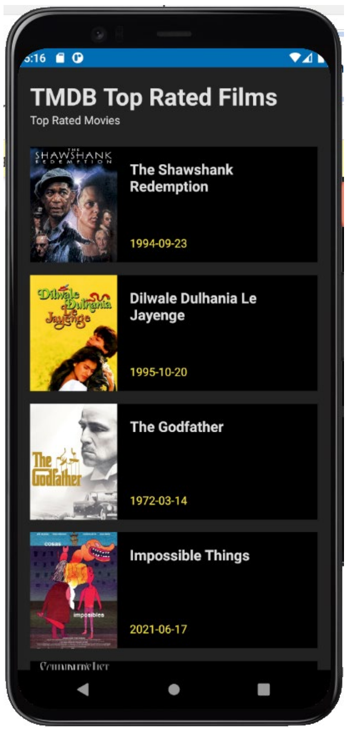
2022-03-17

The Batman

2022-03-01

Popular Movies at the Current time of 11/05/2022 17:16

Top rated:


TMDB Top Rated Films

Top Rated Movies

The Shawshank Redemption

1994-09-23

Dilwale Dulhania Le Jayenge

1995-10-20

The Godfather

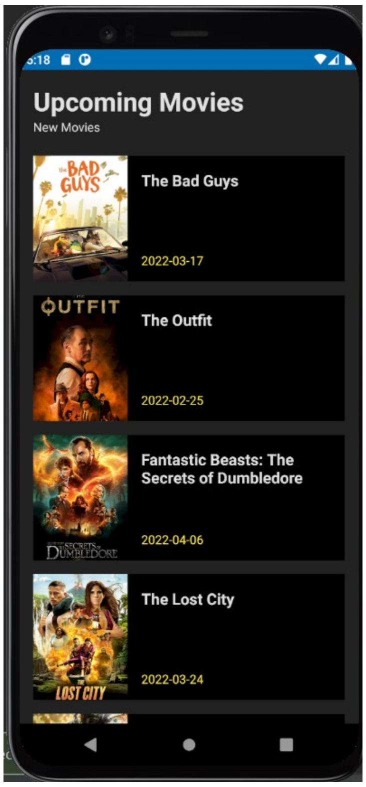
1972-03-14

Impossible Things

2021-06-17

Popular Movies at the Current time of 11/05/2022 17:16

Upcoming Movies:


5:18
Upcoming Movies
New Movies

The Bad Guys
2022-03-17

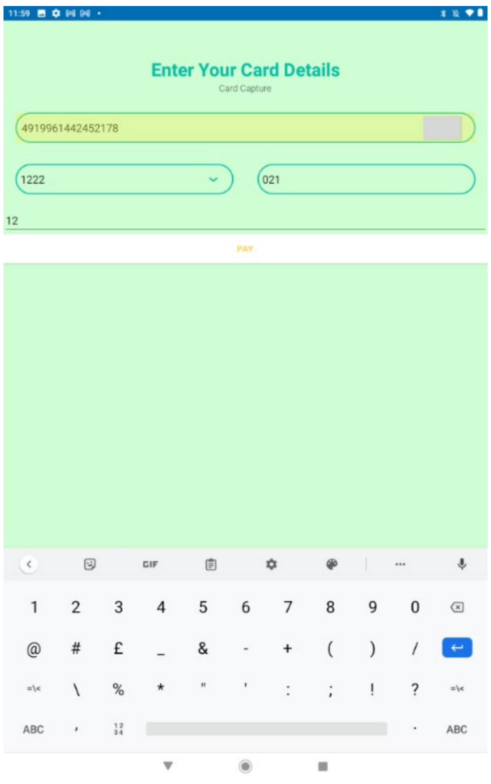
The Outfit
2022-02-25

Fantastic Beasts: The Secrets of Dumbledore
2022-04-06

The Lost City
2022-03-24

New Releases and upcoming Movies at the Current time of 11/05/2022 17:19

Add A tip with google Pay or debit of your choice:


11:59
Enter Your Card Details
Card Capture

4919961442452178

1222

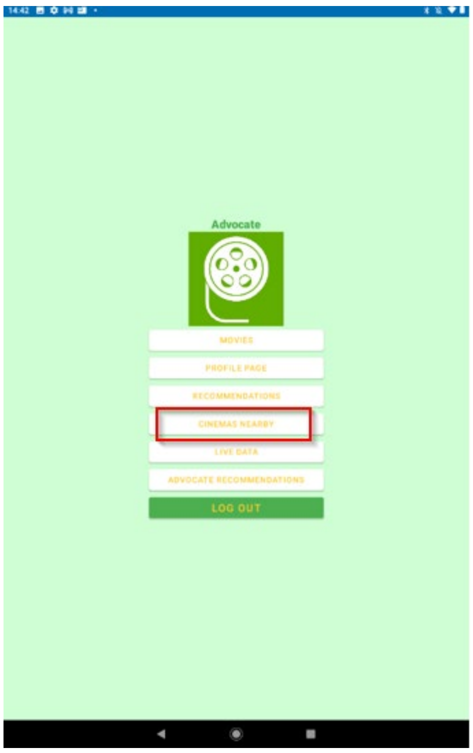
021

12

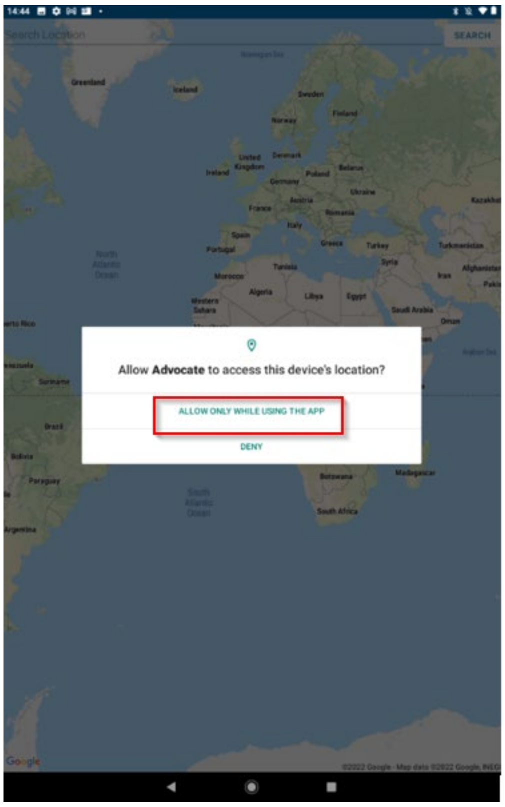
PAY

User guide to receive Local Movie Theatres and search for other Movie Theatres

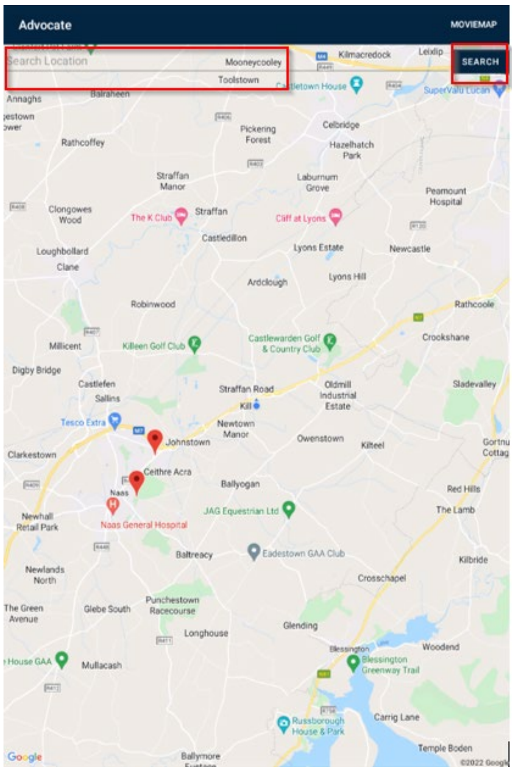
From the previous steps of the user registration and logging in successfully they will be brought to the main Logged in page of Advocate allowing the users to access the maps Activity by clciking the button shown “Cinemas Nearby” from hear the user will be navigated to the Maps Activity.



The user can then access the local movie theatres the google maps API will fetch for them once they accept the permission of allowing advocate access their current device location. This will then get the users devices location and show the location on the map.

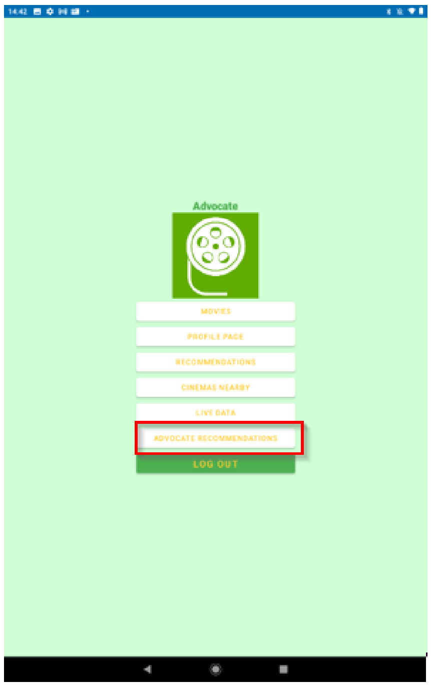

Search Location
SEARCH
Allow **Advocate** to access this device's location?
ALLOW ONLY WHILE USING THE APP
DENY

The user can also search for any Movie Theatre in the world or any location by typing the correct value in the text box and clicking search which will place a marker where the location is found.

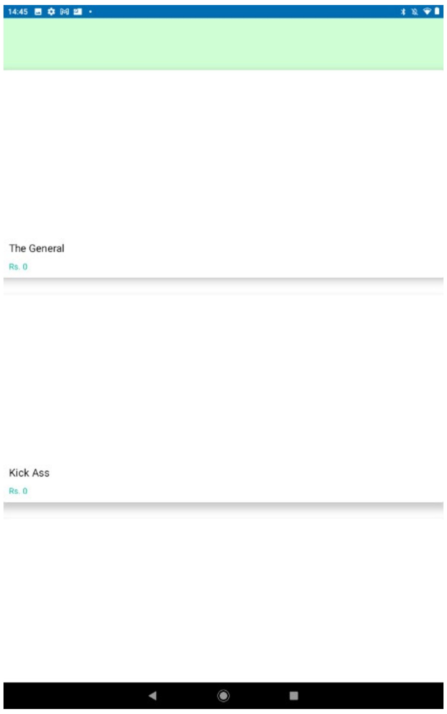


User Guide to get Movie Recommendations by Genre

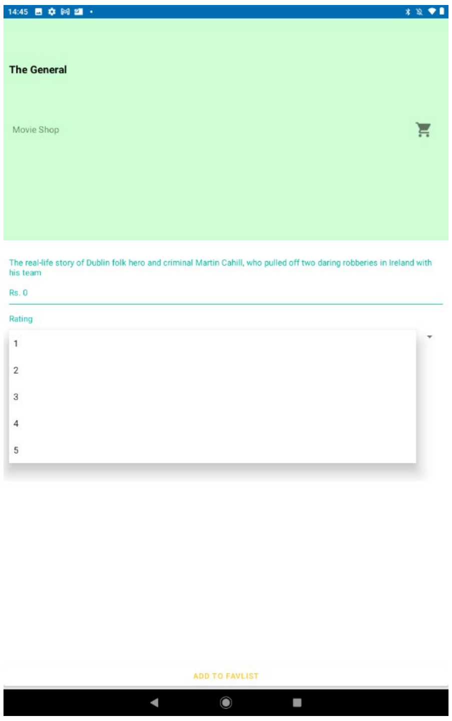
Once Logged in the user can navigate to the Home Recommendations activity by clicking the "Advocate Recommendations" button to navigate to the free movie list.



The user will be brought to the Free Movie list Advocate provides where they can click into each.



There will be a description of the movie a rating of the movie and an option to add the movie to the perosnalised user watchlist which will change from blank to red + the number of movies added to the list and saved to the individuals account.



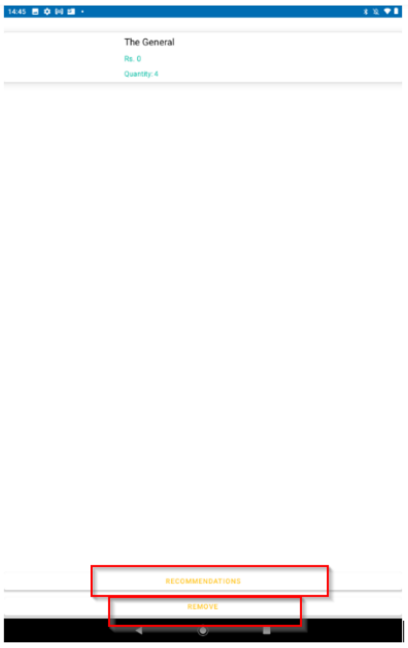
The real-life story of Dublin folk hero and criminal Martin Cahill, who pulled off two daring robberies in Ireland with his team

Rs. 0

Rating

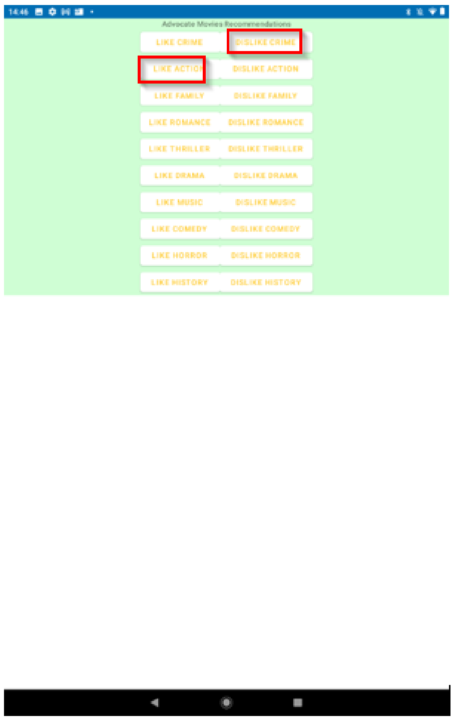
1
2
3
4
5

Once movie/movies are added to the watchlist the user can remove the movies from the list or receive recommendations based off similar genres.

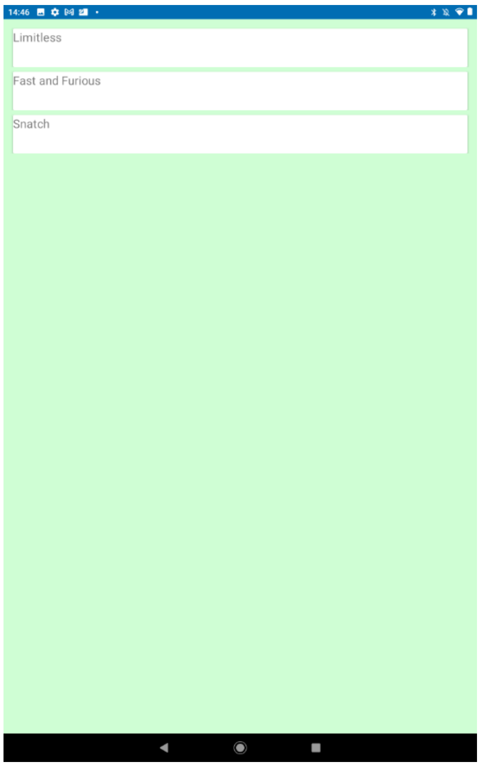


The General
Rs. 0
Quantity: 4

If the user clicks the Recommendations button as shown, they will be brought to the following activity where they can either like or dislike a genre.



If the user likes a particular genre, they will then be brought to the following activity of a recycler view and be given recommendations based on the genre and be eligible to access these movies for free in the movies section.



Limitless

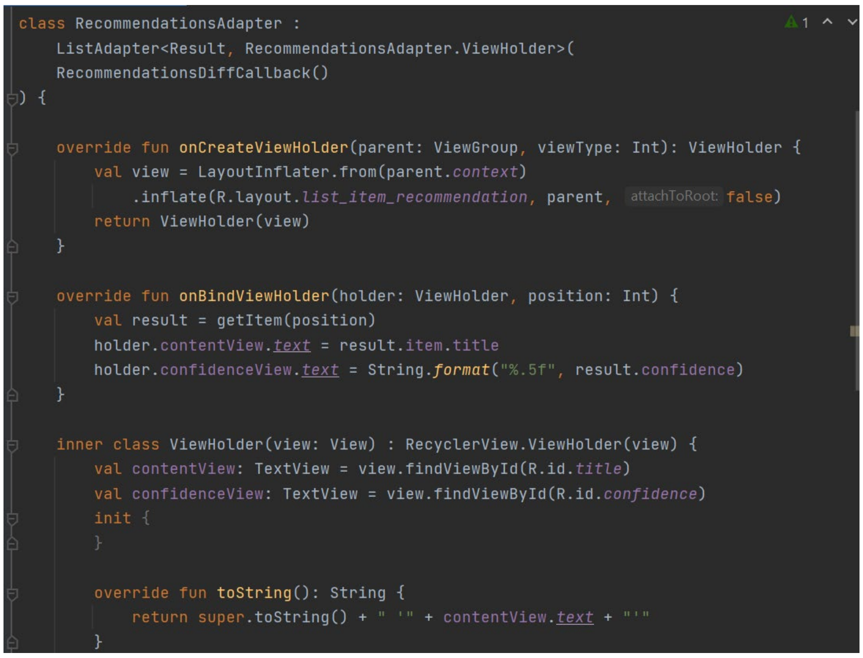
Fast and Furious

Snatch

3.0 Implementation

Recommendations Adapter

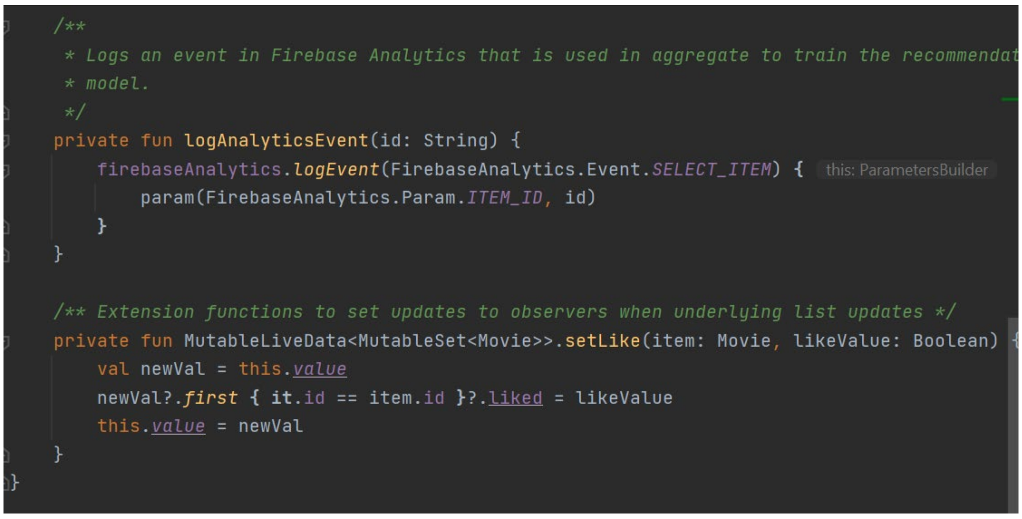
This adapter is what retrieves and populates the recommendations from the TensorFlow lite model allowing the user to see what movies are recommended to them based on their decisions. The title of the movie and “confidence” will be retrieved which is the integer of similarity or average the movie is given by the recommendation model.



```
class RecommendationsAdapter :
    ListAdapter<Result, RecommendationsAdapter.ViewHolder>(
    RecommendationsDiffCallback()
) {

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
        val view = LayoutInflater.from(parent.context)
            .inflate(R.layout.list_item_recommendation, parent,  attachToRoot: false)
        return ViewHolder(view)
    }

    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        val result = getItem(position)
        holder.contentView.text = result.item.title
        holder.confidenceView.text = String.format("%.5f", result.confidence)
    }

    inner class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val contentView: TextView = view.findViewById(R.id.title)
        val confidenceView: TextView = view.findViewById(R.id.confidence)
        init {
        }

        override fun toString(): String {
            return super.toString() + " '" + contentView.text + "'"
        }
```

Recommendations Training

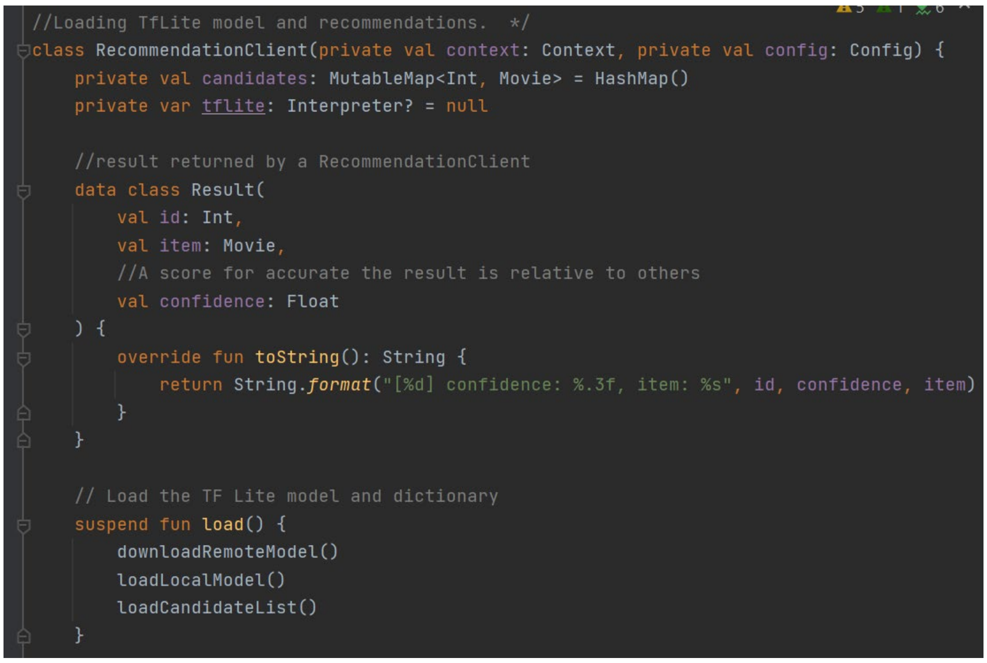
As shown below the code is used to train the TensorFlow lite model by logging events with user interactions eg. When they like a movie from the dataset an event is logged and initiates the TensorFlow lite model.



```
    /**
     * Logs an event in Firebase Analytics that is used in aggregate to train the recommendat
     * model.
     */
    private fun logAnalyticsEvent(id: String) {
        firebaseAnalytics.logEvent(FirebaseAnalytics.Event.SELECT_ITEM) {  this: ParametersBuilder
            param(FirebaseAnalytics.Param.ITEM_ID, id)
        }
    }

    /** Extension functions to set updates to observers when underlying list updates */
    private fun MutableLiveData<MutableSet<Movie>>.setLike(item: Movie, likeValue: Boolean) {
        val newVal = this.value
        newVal?.first { it.id == item.id }?.liked = likeValue
        this.value = newVal
    }
}
```

Recommendation Client

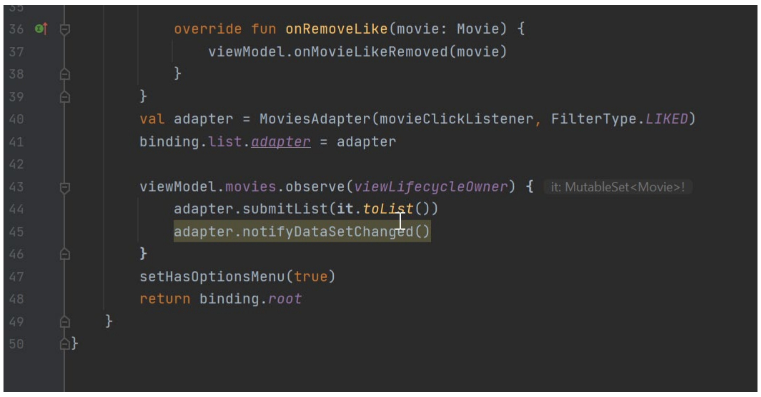
Acts as a client to the Server (TensorFlow) where the model and movie dictionary are loaded into Advocate by their id, movie and “confidence” the higher the integer the better as this judges the accuracy of the recommendation by the model judging by this dataset.



```kotlin
//Loading TfLite model and recommendations.  */
class RecommendationClient(private val context: Context, private val config: Config) {
    private val candidates: MutableMap<Int, Movie> = HashMap()
    private var tflite: Interpreter? = null

    //result returned by a RecommendationClient
    data class Result(
        val id: Int,
        val item: Movie,
        //A score for accurate the result is relative to others
        val confidence: Float
    ) {
        override fun toString(): String {
            return String.format("[%d] confidence: %.3f, item: %s", id, confidence, item)
        }
    }

    // Load the TF Lite model and dictionary
    suspend fun load() {
        downloadRemoteModel()
        loadLocalModel()
        loadCandidateList()
    }
```

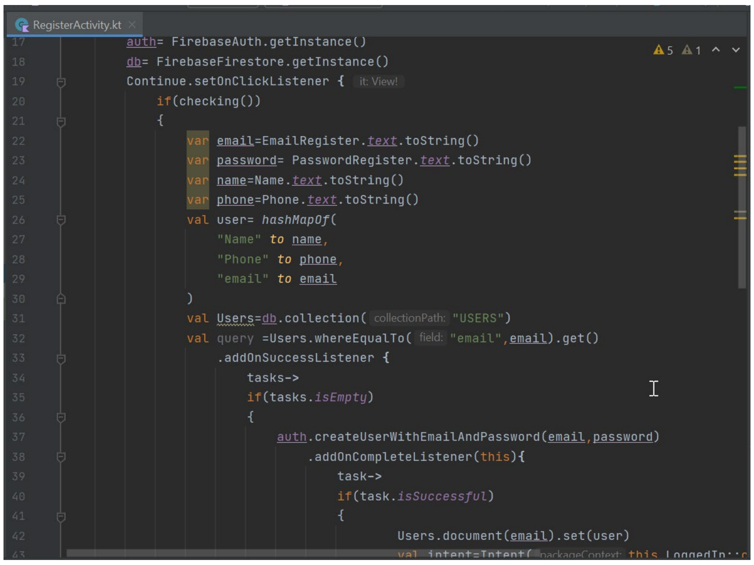
Liked Movies Fragment

which gathers all the movies from the movies list that have been liked by the user. Once the user accesses the app and logs in successfully, they will be introduced to a movies list. The HomeViewPagerFragment class will be present in the apps main activity where it will allow the users to slide between Movies, Liked Movies List and Recommendations Movie list. As you can see in the below snippet that once a movie is clicked (liked) an Onclicklistener is used that detects the user’s selection of a movie and adds it to the liked list if the user clicks the movie twice, they will unlike the movie which will then remove the previously liked movie from the liked list.



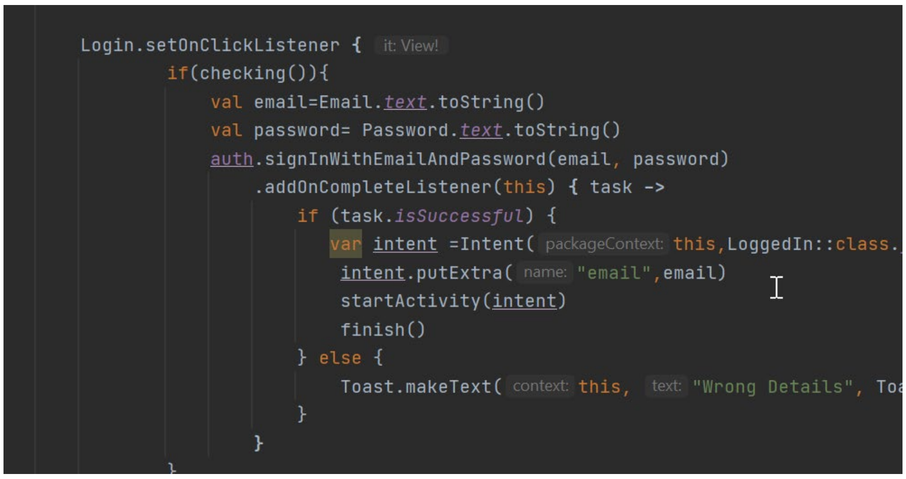
```kotlin
            override fun onRemoveLike(movie: Movie) {
                viewModel.onMovieLikeRemoved(movie)
            }
        }
        val adapter = MoviesAdapter(movieClickListener, FilterType.LIKED)
        binding.list.adapter = adapter

        viewModel.movies.observe(viewLifecycleOwner) {
            adapter.submitList(it.toList())
            adapter.notifyDataSetChanged()
        }
        setHasOptionsMenu(true)
        return binding.root
    }
}
```

The Homepage Fragment allows the users to slide between these features with ease and they can easily access liked movies and recommendations as well as a huge movies list. The recommendations fragment has not been fully implemented as of yet but it will aim to take the liked movies form the user to then give meaningful movie recommendations by using googles Big Query analytics to analyse the data and give good recommendations the more that are liked the more the system is trained and will also use TensorFlow lite for this machine learning aspect and define each person's choices accurately



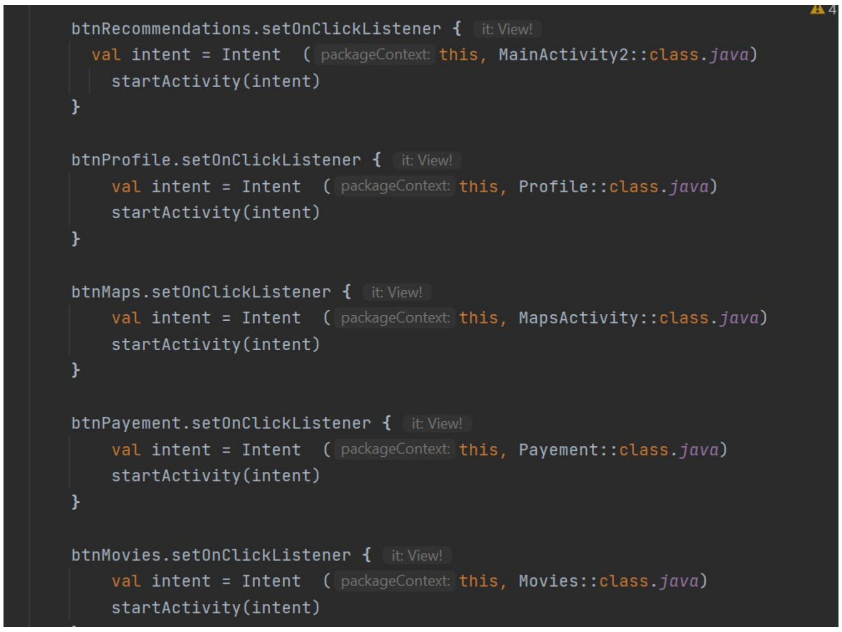
```
RegisterActivity.kt
        auth= FirebaseAuth.getInstance()
        db= FirebaseFirestore.getInstance()
        Continue.setOnClickListener {  it: View!
            if(checking())
            {
                var email=EmailRegister.text.toString()
                var password= PasswordRegister.text.toString()
                var name=Name.text.toString()
                var phone=Phone.text.toString()
                val user= hashMapOf(
                    "Name" to name,
                    "Phone" to phone,
                    "email" to email
                )
                val Users=db.collection( collectionPath: "USERS")
                val query =Users.whereEqualTo( field: "email",email).get()
                    .addOnSuccessListener {
                        tasks->
                        if(tasks.isEmpty)
                        {
                            auth.createUserWithEmailAndPassword(email,password)
                                .addOnCompleteListener(this){
                                    task->
                                    if(task.isSuccessful)
                                    {
                                            Users.document(email).set(user)
                                            val intent=Intent( packageContext: this,LoggedIn::c
```

As you can see from the code snippet if statement is used to check the users' details added to the registration fields id the user enters the correct details they will be navigated to the main page of the app and their details will be stored in google firebase.



```
    Login.setOnClickListener {  it: View!
            if(checking()){
                val email=Email.text.toString()
                val password= Password.text.toString()
                auth.signInWithEmailAndPassword(email, password)
                    .addOnCompleteListener(this) { task ->
                        if (task.isSuccessful) {
                            var intent =Intent( packageContext: this,LoggedIn::class.
                            intent.putExtra( name: "email",email)
                            startActivity(intent)
                            finish()
                        } else {
                            Toast.makeText( context: this,  text: "Wrong Details", Toa
                        }
                    }
```

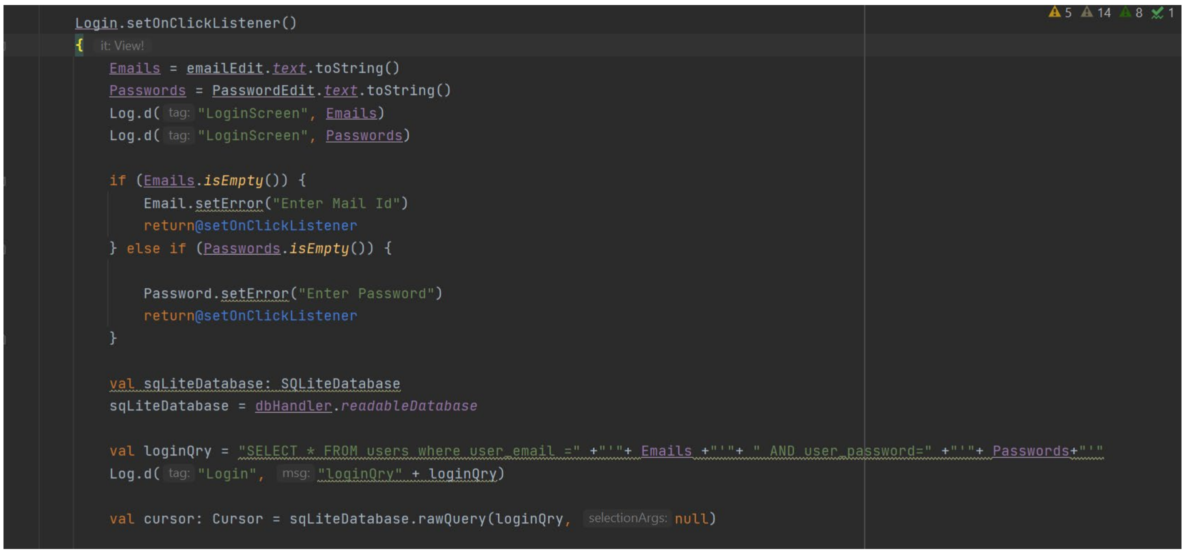
Like the register activity the login activity check if the users' details have already been stored in the database if they are the user gains entry to advocate, if not the user must register or re-enter their correct pre-registered details.



```
btnRecommendations.setOnClickListener { it: View!
    val intent = Intent  ( packageContext: this, MainActivity2::class.java)
    startActivity(intent)
}

btnProfile.setOnClickListener { it: View!
    val intent = Intent  ( packageContext: this, Profile::class.java)
    startActivity(intent)
}

btnMaps.setOnClickListener { it: View!
    val intent = Intent  ( packageContext: this, MapsActivity::class.java)
    startActivity(intent)
}

btnPayement.setOnClickListener { it: View!
    val intent = Intent  ( packageContext: this, Payement::class.java)
    startActivity(intent)
}

btnMovies.setOnClickListener { it: View!
    val intent = Intent  ( packageContext: this, Movies::class.java)
    startActivity(intent)
```

As you can see form the code snippet above, I have added on click listeners to every button in the main screen this allows each button to open new activity according to their button name and purpose. The name of the button is related to the page they are navigating to and will present the pages implementations when fully complete.

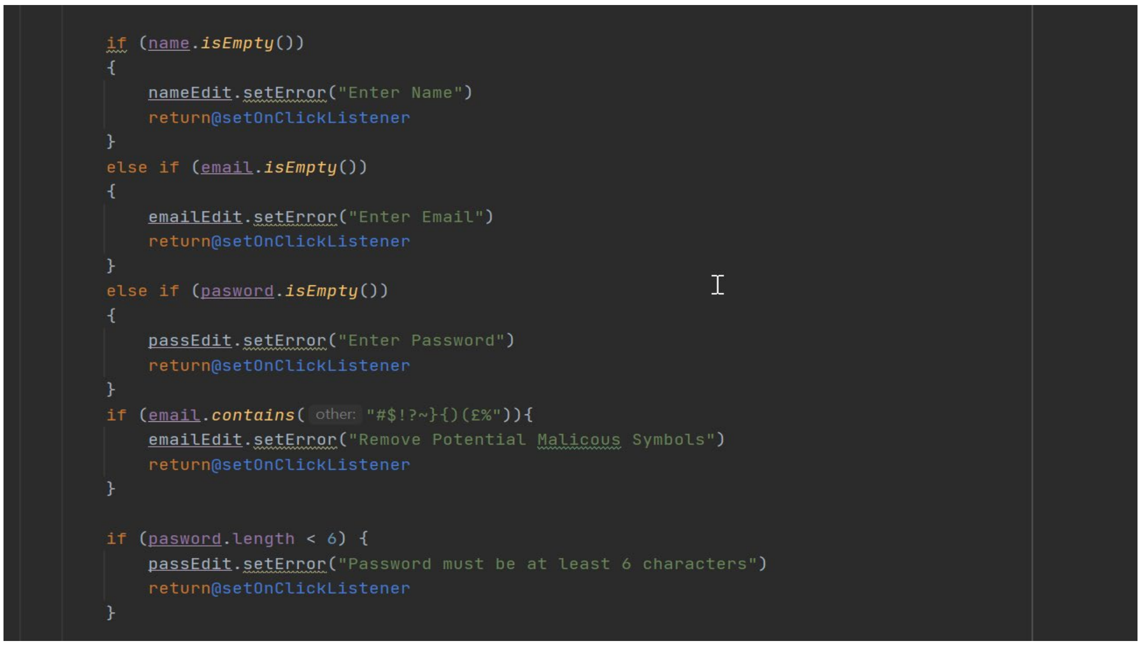
Login Activity



```
Login.setOnClickListener()
{ it: View!
    Emails = emailEdit.text.toString()
    Passwords = PasswordEdit.text.toString()
    Log.d( tag: "LoginScreen", Emails)
    Log.d( tag: "LoginScreen", Passwords)

    if (Emails.isEmpty()) {
        Email.setError("Enter Mail Id")
        return@setOnClickListener
    } else if (Passwords.isEmpty()) {

        Password.setError("Enter Password")
        return@setOnClickListener
    }

    val sqLiteDatabase: SQLiteDatabase
    sqLiteDatabase = dbHandler.readableDatabase

    val loginQry = "SELECT * FROM users where user_email =" +"'"+ Emails +"'"+ " AND user_password=" +"'"+ Passwords+"'"
    Log.d( tag: "Login",  msg: "loginQry" + loginQry)

    val cursor: Cursor = sqLiteDatabase.rawQuery(loginQry,  selectionArgs: null)
```

Snippet of code showing how advocate gets the users details if they are correctly added to the text boxes and registered as a user already on the Advocate database.

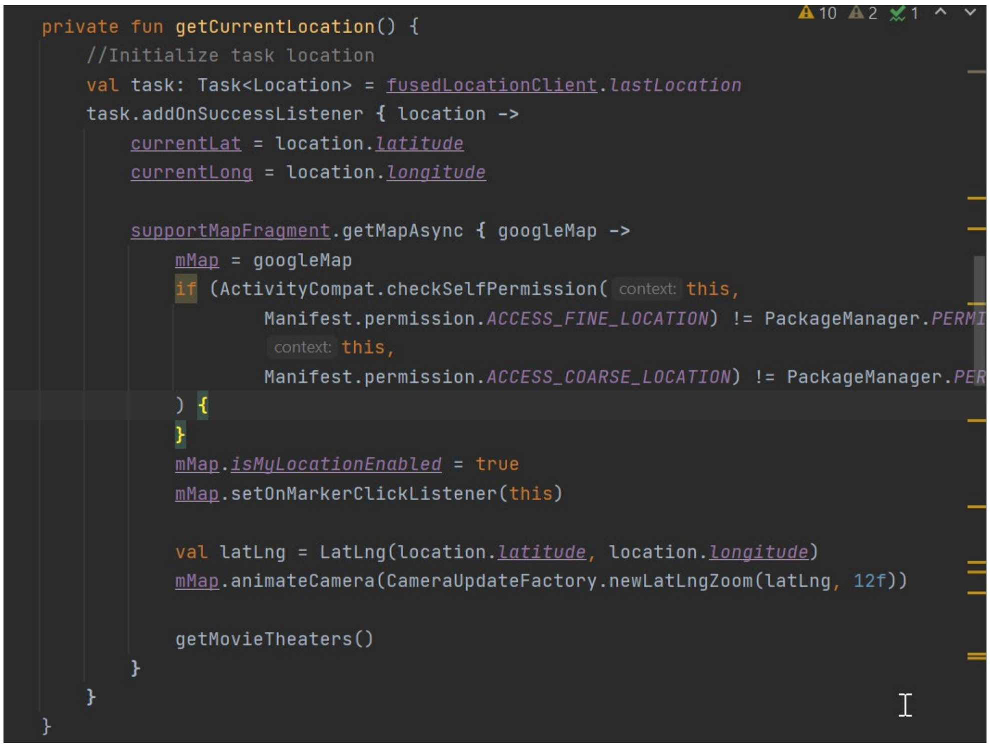
Register Activity



```
if (name.isEmpty())
{
    nameEdit.setError("Enter Name")
    return@setOnClickListener
}
else if (email.isEmpty())
{
    emailEdit.setError("Enter Email")
    return@setOnClickListener
}
else if (pasword.isEmpty())
{
    passEdit.setError("Enter Password")
    return@setOnClickListener
}
if (email.contains( other: "#$!?~}{)(£%")){
    emailEdit.setError("Remove Potential Malicous Symbols")
    return@setOnClickListener
}

if (pasword.length < 6) {
    passEdit.setError("Password must be at least 6 characters")
    return@setOnClickListener
}
```

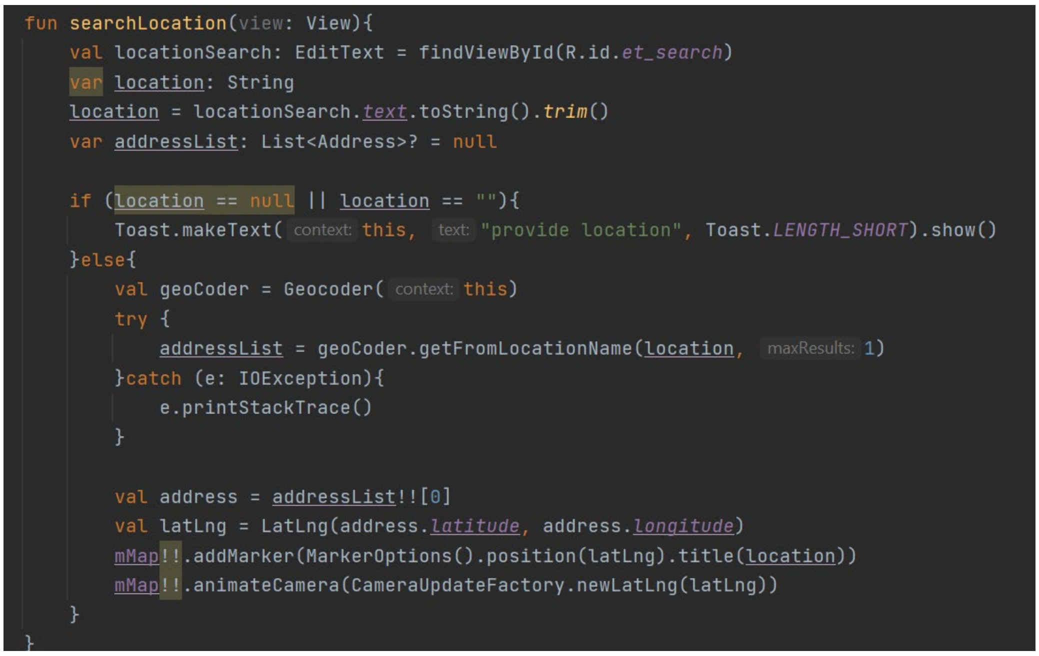
Snippet of code representing the restrictions of the register page if a user attempts to submit a password below 6 characters, adds symbols to their email as shown above or does not enter a value at all into any column they will be flagged with an error message.

Google Maps Activity

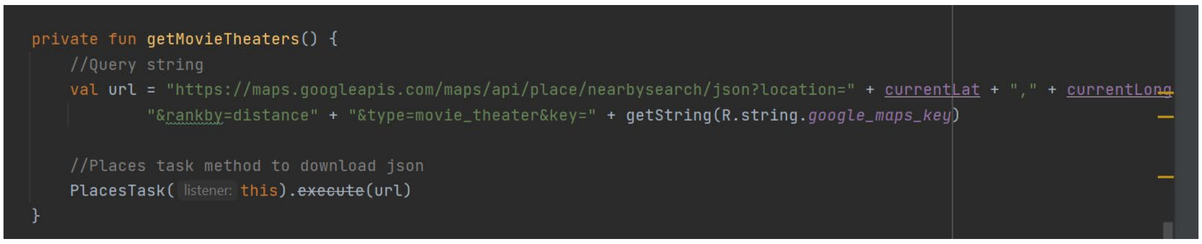


```
private fun getCurrentLocation() {
    //Initialize task location
    val task: Task<Location> = fusedLocationClient.lastLocation
    task.addOnSuccessListener { location ->
        currentLat = location.latitude
        currentLong = location.longitude

        supportMapFragment.getMapAsync { googleMap ->
            mMap = googleMap
            if (ActivityCompat.checkSelfPermission( context: this,
                    Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMI
                     context: this,
                    Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PER
            ) {
            }
            mMap.isMyLocationEnabled = true
            mMap.setOnMarkerClickListener(this)

            val latLng = LatLng(location.latitude, location.longitude)
            mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(latLng, 12f))

            getMovieTheaters()
        }
    }
}
```

This snippet of code demonstrates Advocate prompting the user for consent of tracking their location. If the user accepts the prompt the map proceeds to get nearby places and if they deny it the map won't be displayed to the user as it requires their location to populate Movie theatre markers in proximity.

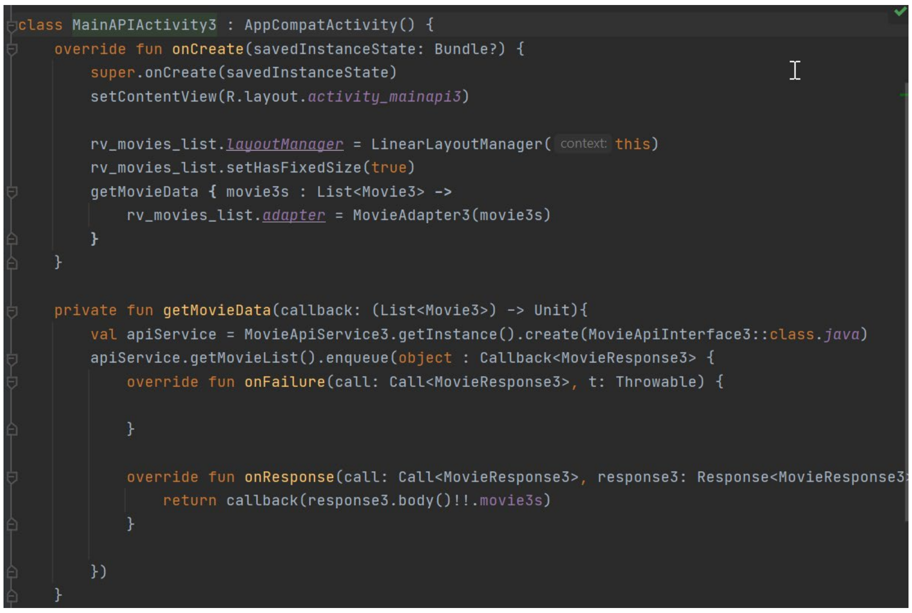

```
fun searchLocation(view: View){
    val locationSearch: EditText = findViewById(R.id.et_search)
    var location: String
    location = locationSearch.text.toString().trim()
    var addressList: List<Address>? = null

    if (location == null || location == ""){
        Toast.makeText( context: this,  text: "provide location", Toast.LENGTH_SHORT).show()
    }else{
        val geoCoder = Geocoder( context: this)
        try {
            addressList = geoCoder.getFromLocationName(location,  maxResults: 1)
        }catch (e: IOException){
            e.printStackTrace()
        }

        val address = addressList!![0]
        val latLng = LatLng(address.latitude, address.longitude)
        mMap!!.addMarker(MarkerOptions().position(latLng).title(location))
        mMap!!.animateCamera(CameraUpdateFactory.newLatLng(latLng))
    }
}
```

Within the maps activity I also implemented a search feature which allows the user to search any location in the world and a marker will be added to whatever location they enter which is good for the user if they wanted to search a particular cinema instead of one that's nearby out of curiosity and the map will navigate to the location.


```
private fun getMovieTheaters() {
    //Query string
    val url = "https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=" + currentLat + "," + currentLong
            "&rankby=distance" + "&type=movie_theater&key=" + getString(R.string.google_maps_key)

    //Places task method to download json
    PlacesTask( listener: this).execute(url)
}
```

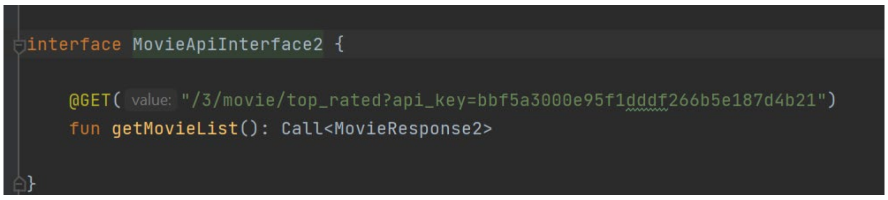
This Query was used to extract all movie theatres close to the devices location once the permission was granted using the Google maps API key and Uri format.

Live Data



```
class MainAPIActivity3 : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_mainapi3)

        rv_movies_list.layoutManager = LinearLayoutManager( context: this)
        rv_movies_list.setHasFixedSize(true)
        getMovieData { movie3s : List<Movie3> ->
            rv_movies_list.adapter = MovieAdapter3(movie3s)
        }
    }

    private fun getMovieData(callback: (List<Movie3>) -> Unit){
        val apiService = MovieApiService3.getInstance().create(MovieApiInterface3::class.java)
        apiService.getMovieList().enqueue(object : Callback<MovieResponse3> {
            override fun onFailure(call: Call<MovieResponse3>, t: Throwable) {

            }

            override fun onResponse(call: Call<MovieResponse3>, response3: Response<MovieResponse3
                return callback(response3.body()!!.movie3s)
            }

        })
    }
```

This snippet calls the TMDB API getting the data for its top-rated movies on the database and is saved to Advocate locally, stored in an array list and is then shown in a recycler view that was created in XML Displaying the Movie names, rating and release dates.



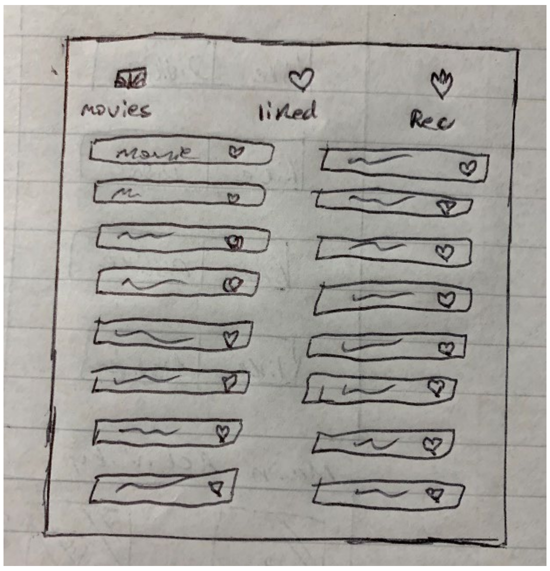
```
interface MovieApiInterface2 {

    @GET( value: "/3/movie/top_rated?api_key=bbf5a3000e95f1dddf266b5e187d4b21")
    fun getMovieList(): Call<MovieResponse2>

}
```

Above is the snippet used to extract the data we wanted to extract which in this case was the top-rated movies on the TMDB database. This also provides the users with movie ideas including other Live data extractions used like Upcoming releases and popular movies now according to TMDB which change daily.

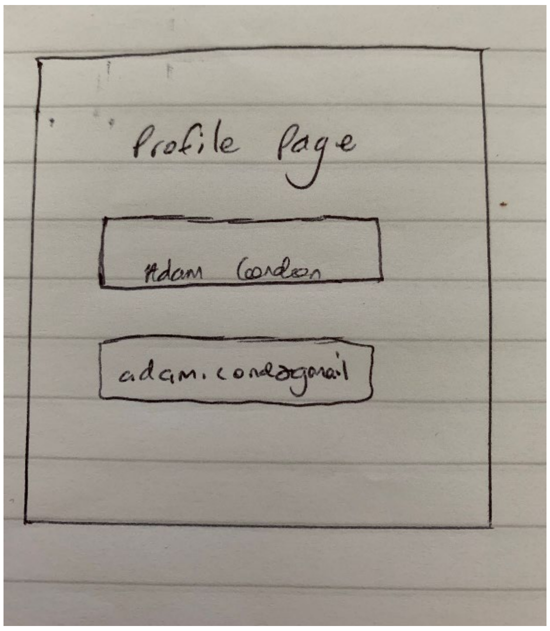
4.0 Graphical User Interface (GUI)

4.1 **Low Fidelity**

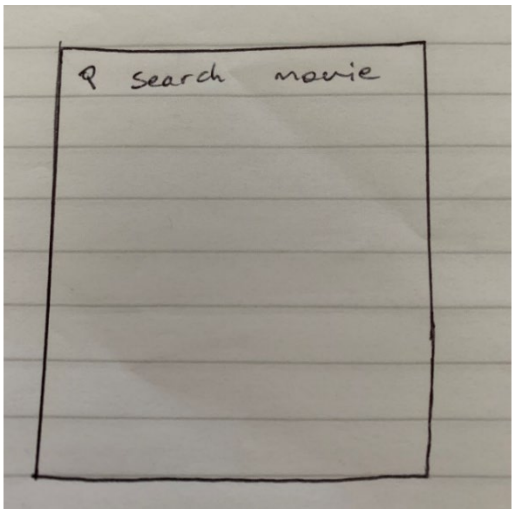
Main Activity



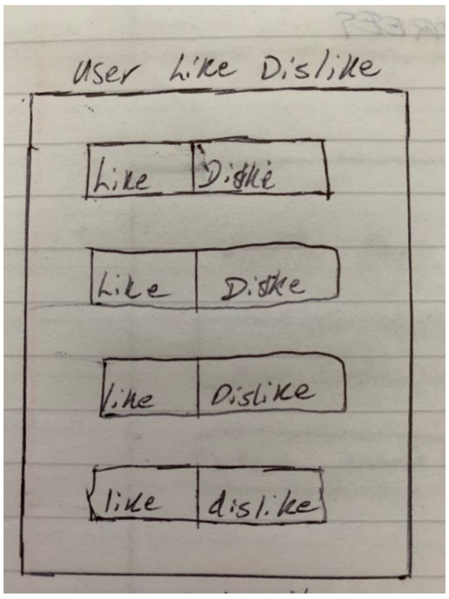
Profile Page



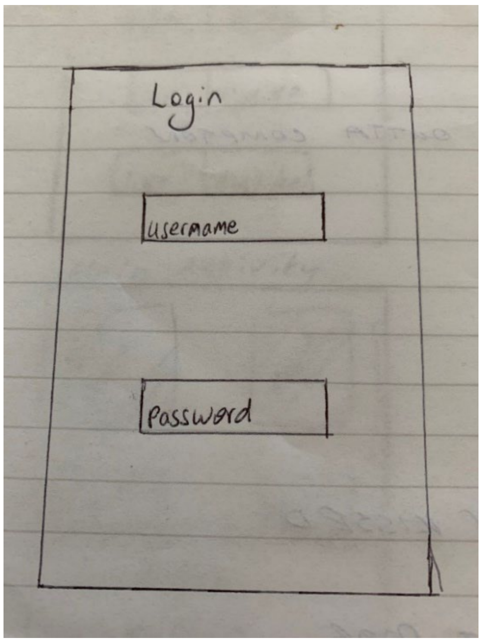
Search Movie Fragment



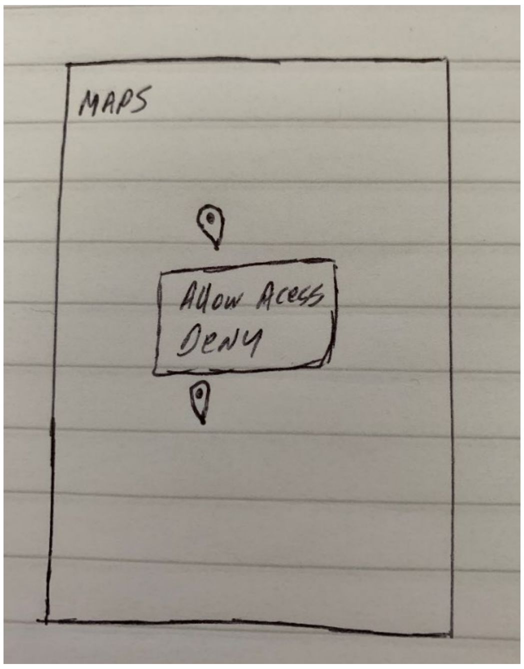
User Like Dislike Page



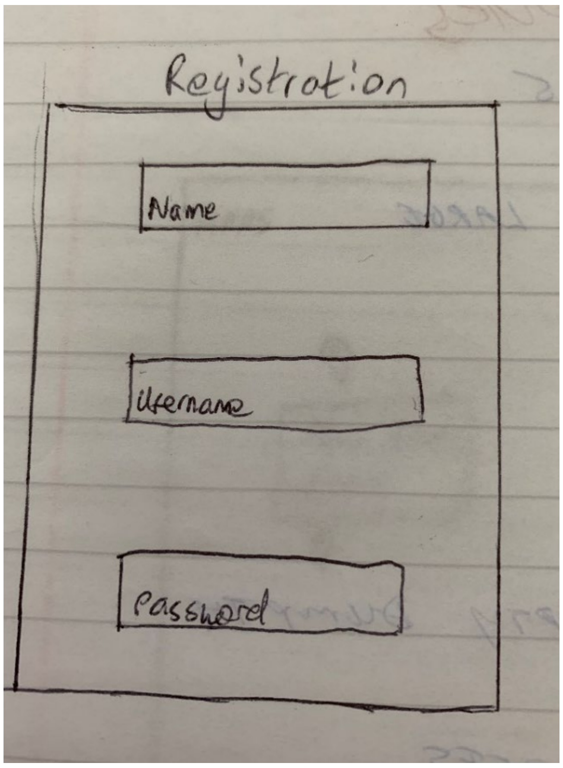
Login Screen



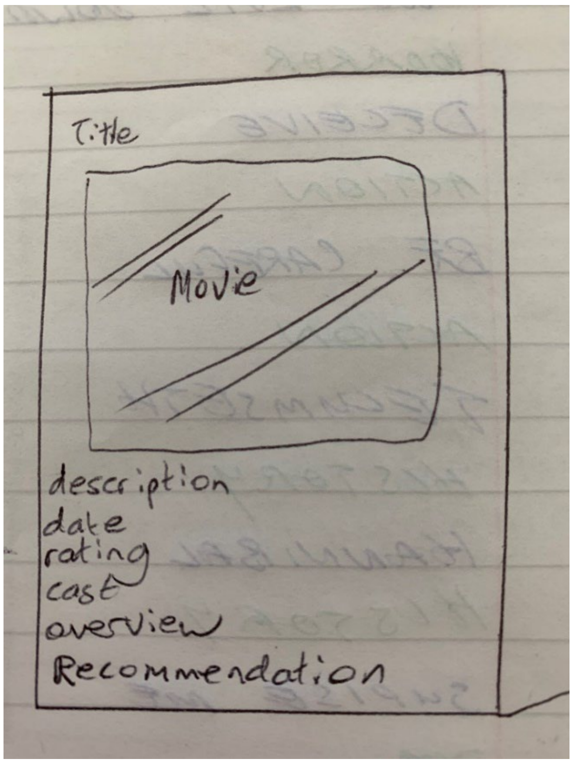
Maps Activity



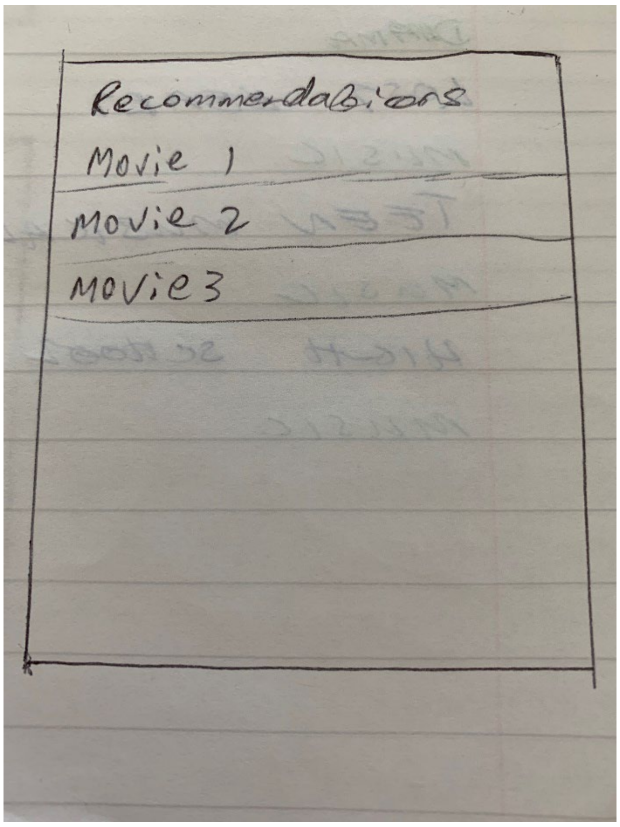
Registration Page



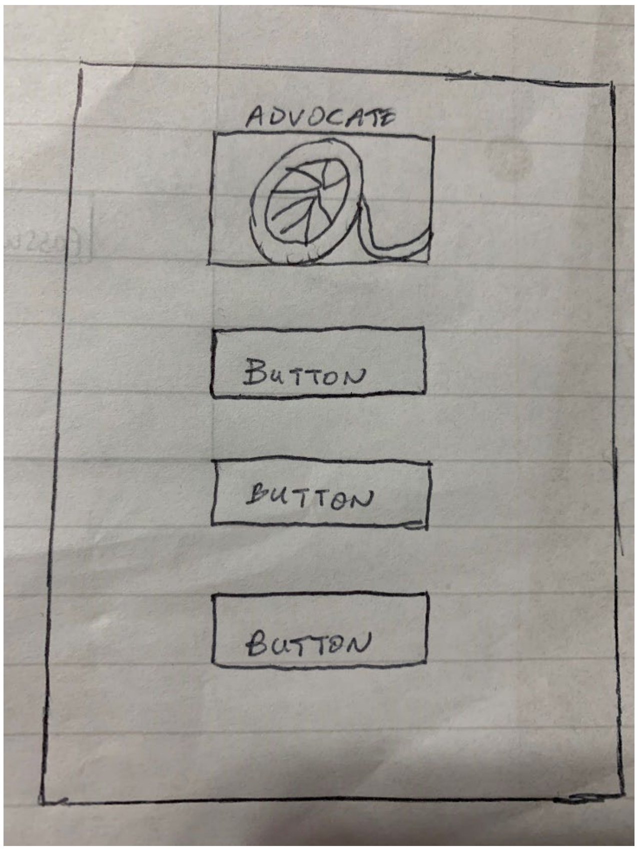
Upcoming/Top Rated/Popular Movie Details



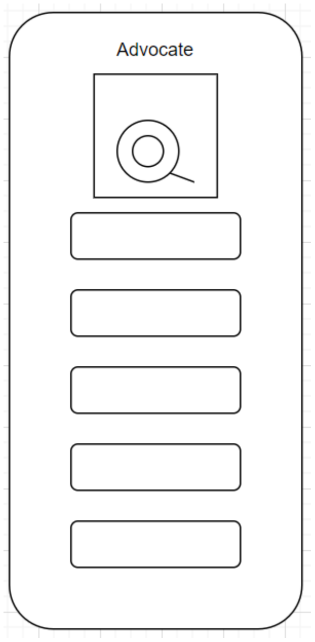
Meaningful Recommendations

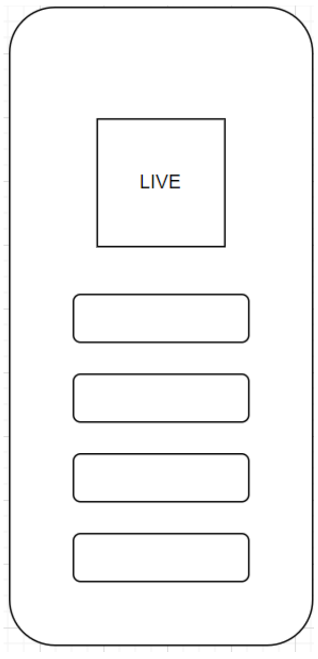


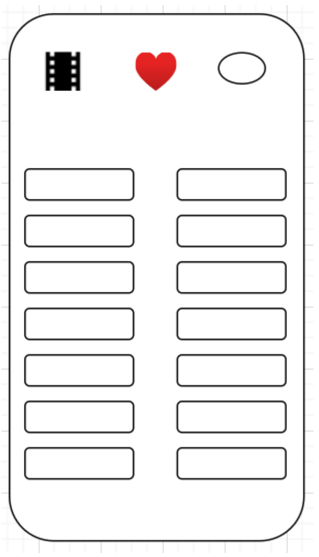
Live Data Menu

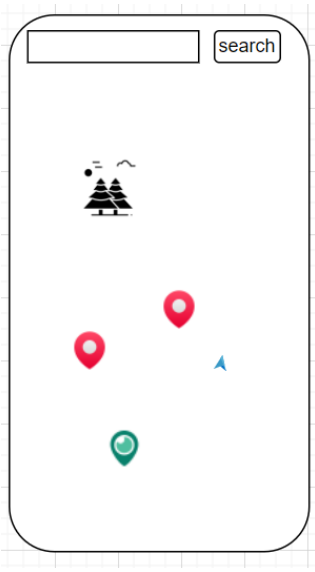


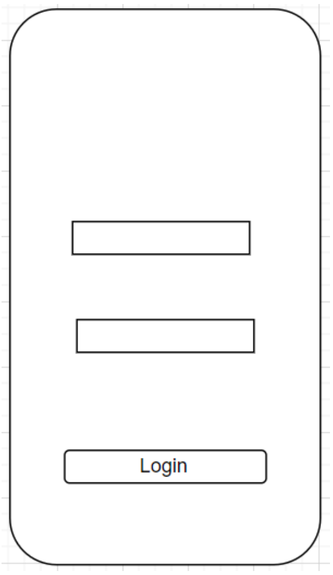
4.2 Wireframes

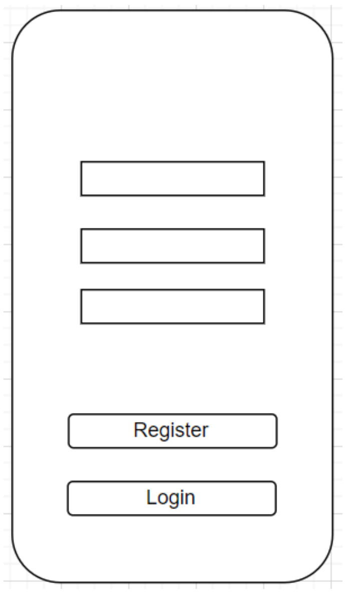






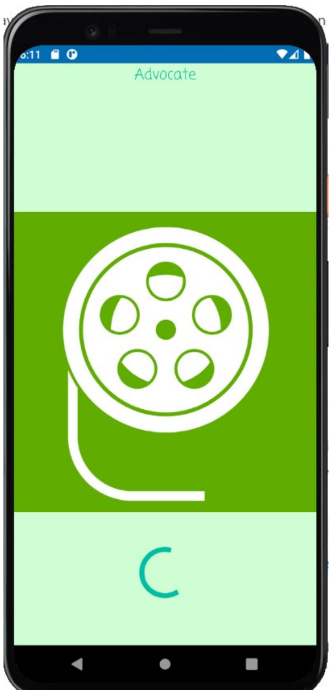

search


Login

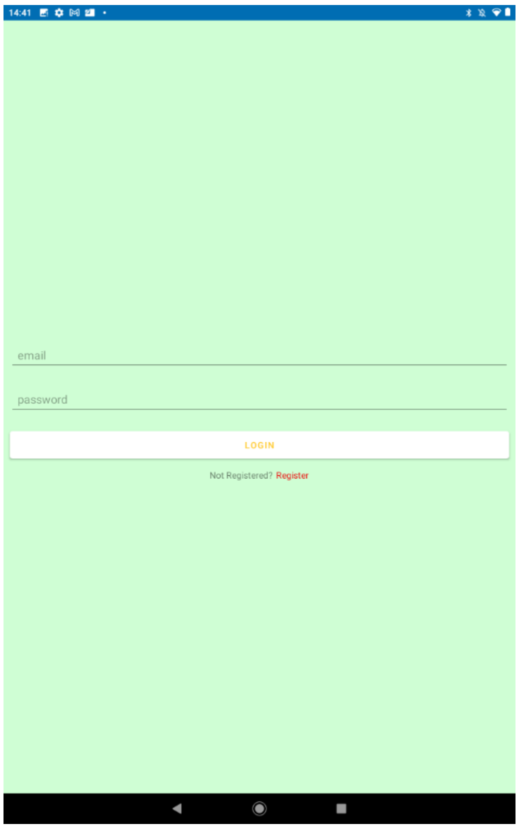


4.3 High Fidelity

Splash Screen



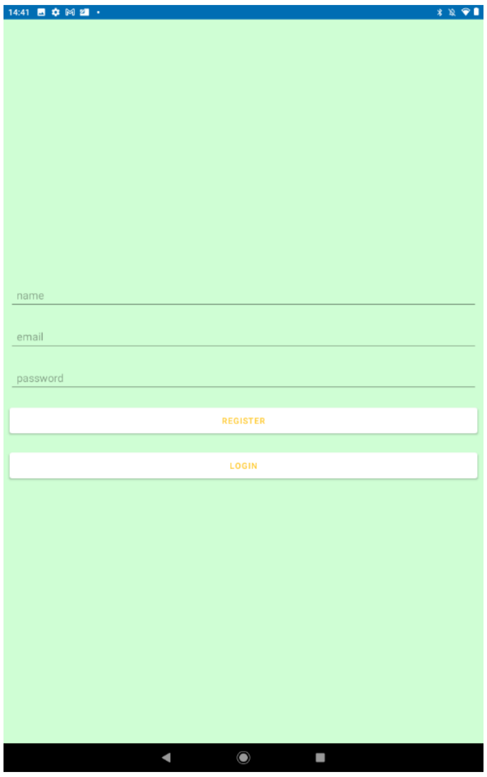
Login Screen



Has two input boxes for a user's email and password if they previously made an account. The user can enter their details for their account and gain access to the app.

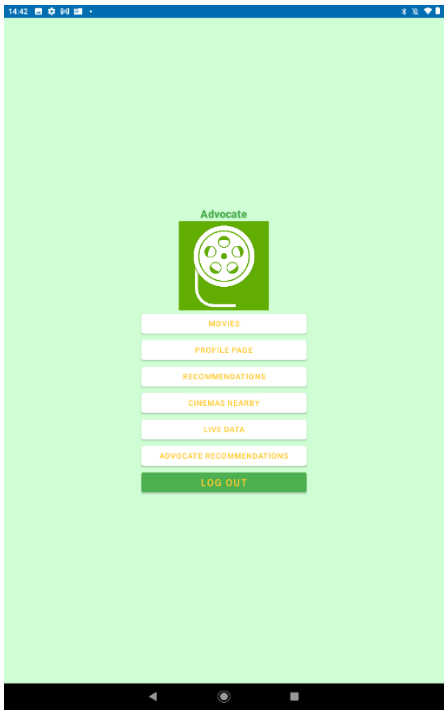
Registration Screen

Provides input boxes where users can enter their details and create an account once the correct steps are followed. The user will then gain access to the app once their details are entered, and they will be stored in the firebase database.

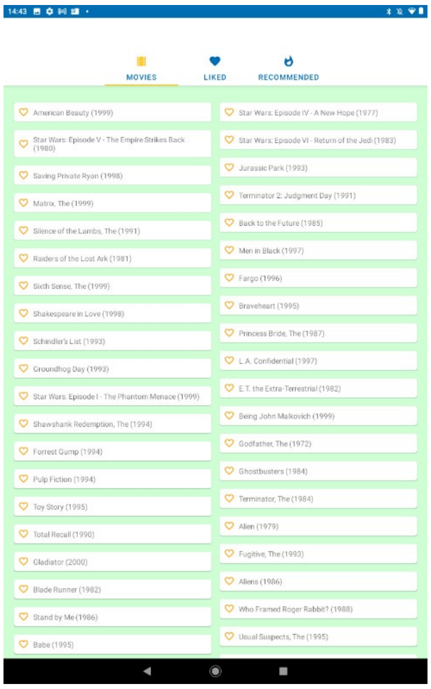


LoggedIn Activity

The Main Activity will allow the user to like any movie or movies of their choice. The more movies the user likes the more meaningful their Personal Recommendations will become and will allow to train the TensorFlow Model. The user should gain access to this activity once the movie of choice is clicked. The user has a large list of movies to choose from, whichever movie they click on this means they have liked/chose the movie. The user can unlike the movie by simply clicking the movie again and removing it from their liked list.



Movie List

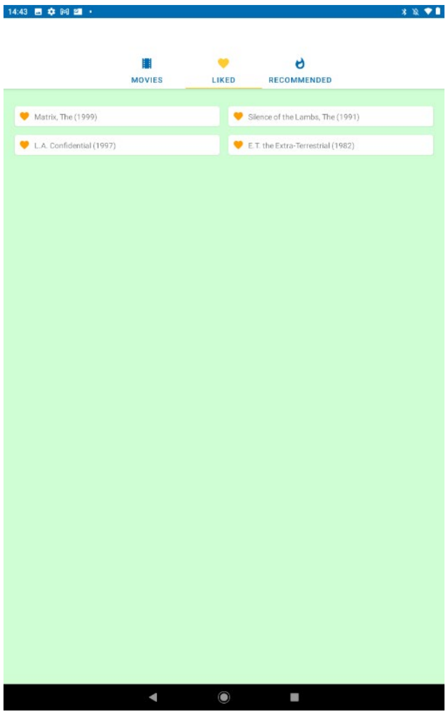


American Beauty (1999)
Star Wars: Episode IV - A New Hope (1977)
Star Wars: Episode V - The Empire Strikes Back (1980)
Star Wars: Episode VI - Return of the Jedi (1983)
Saving Private Ryan (1998)
Jurassic Park (1993)
Matrix, The (1999)
Terminator 2: Judgment Day (1991)
Silence of the Lambs, The (1991)
Back to the Future (1985)
Raiders of the Lost Ark (1981)
Men in Black (1997)
Sixth Sense, The (1999)
Fargo (1996)
Shakespeare in Love (1998)
Braveheart (1995)
Schindler's List (1993)
Princess Bride, The (1987)
Groundhog Day (1993)
L.A. Confidential (1997)
Star Wars: Episode I - The Phantom Menace (1999)
E.T. the Extra-Terrestrial (1982)
Shawshank Redemption, The (1994)
Being John Malkovich (1999)
Forrest Gump (1994)
Godfather, The (1972)
Pulp Fiction (1994)
Ghostbusters (1984)
Toy Story (1995)
Terminator, The (1984)
Total Recall (1990)
Alien (1979)
Gladiator (2000)
Fugitive, The (1993)
Blade Runner (1982)
Aliens (1986)
Stand by Me (1986)
Who Framed Roger Rabbit? (1988)
Babe (1995)
Usual Suspects, The (1995)

Is the recycler view displaying all of the movies on the TMDB data set I am using and gives the option to like any movie at the user's leisure?

Liked List

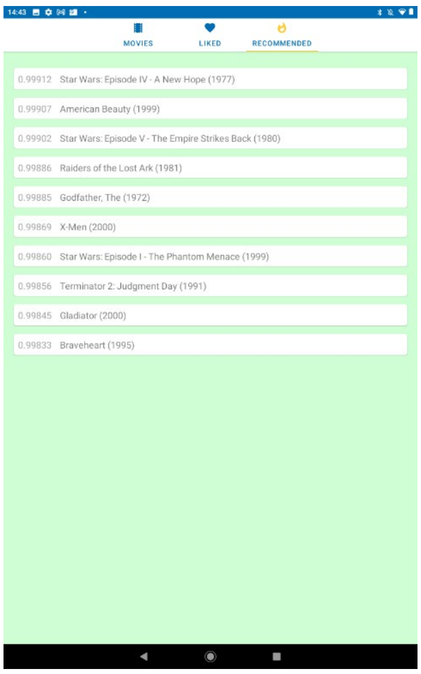
Once the user has liked a certain number of movies they will appear in the stored "Liked List" and saved in a local Array list/Adapter.



Matrix, The (1999)
Silence of the Lambs, The (1991)
L.A. Confidential (1997)
E.T. the Extra-Terrestrial (1982)

Recommended List

Based off the users liked list the Recommendations model will be populated using TensorFlow lite machine learning to determine similar movies based on the genre, rating count, and the movies are designated an average prediction eg 9.09 which could be Toy Story and Nemo as they have quite similar attribute's having family/kids based genres and ratings. Also, if any film has a sequel or a series of movies such as Star Wars for example these will be taken into consideration by the model.

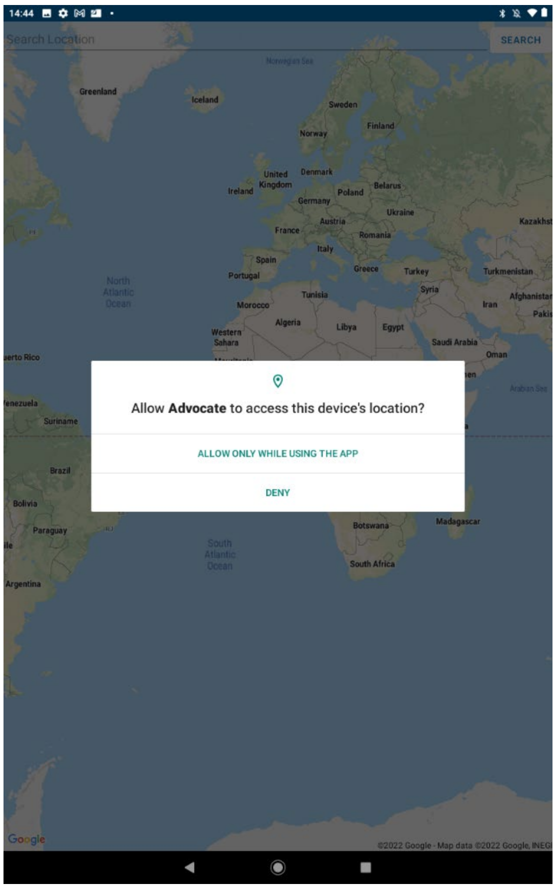


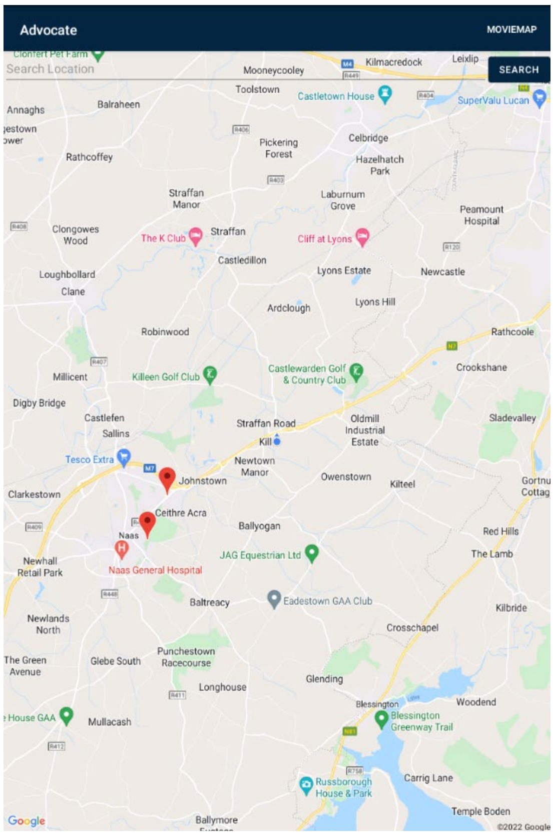
```
0.99912   Star Wars: Episode IV - A New Hope (1977)
0.99907   American Beauty (1999)
0.99902   Star Wars: Episode V - The Empire Strikes Back (1980)
0.99886   Raiders of the Lost Ark (1981)
0.99885   Godfather, The (1972)
0.99869   X-Men (2000)
0.99860   Star Wars: Episode I - The Phantom Menace (1999)
0.99856   Terminator 2: Judgment Day (1991)
0.99845   Gladiator (2000)
0.99833   Braveheart (1995)
```

This will now give the next slide "Recommendations the chance to analyse the liked movies and give meaningful recommendations back to the user and similarly be displayed as shown. The liked video list will be sent to googles big query tool for analysing and TensorFlow lite where then the external AI's will send back recommended movies form the same dataset.

Google Maps API

Which requires the user's location and then the map populates with markers of local cinemas in the user's area. Some details are also extracted about the cinema.

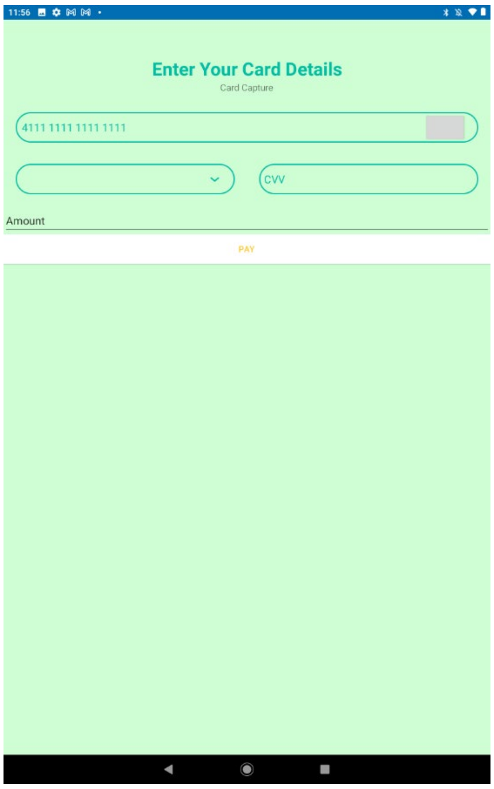

14:44
Search Location
SEARCH
Allow **Advocate** to access this device's location?
ALLOW ONLY WHILE USING THE APP
DENY
Google
©2022 Google · Map data ©2022 Google, INEGI


Advocate
MOVIEMAP
Search Location
SEARCH
Celbridge
Straffan
Clane
Kill
Johnstown
Naas
Naas General Hospital
Blessington
Google
©2022 Google

Users will have the option to enable their location to the Maps Feature from here the user will see Markers that have been found and populated on the maps API of cinemas nearby their area based off their devices GPS location.

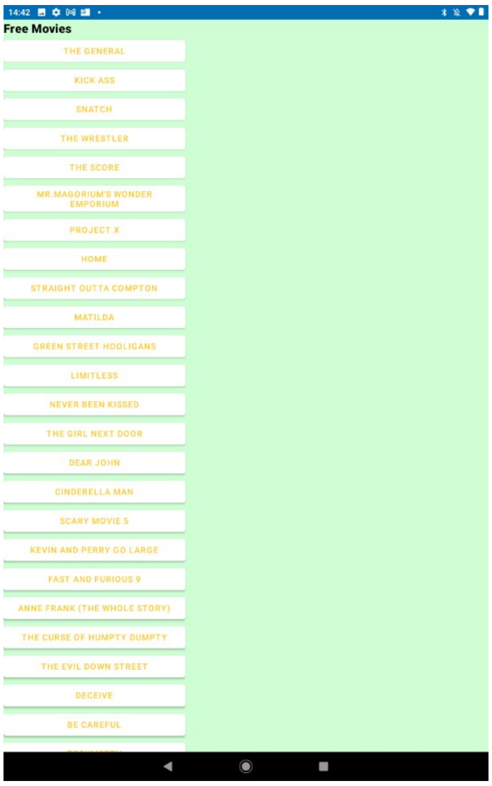
Payment System

The user should be brought to the payment system of a movie they want to buy providing their card details as shown below this can be done manually or the user can add their card details . The user will have an opportunity to either rent or buy a movie.



Movies

The user can watch movies free of charge and receive the links through advocate to YouTubeAPI.



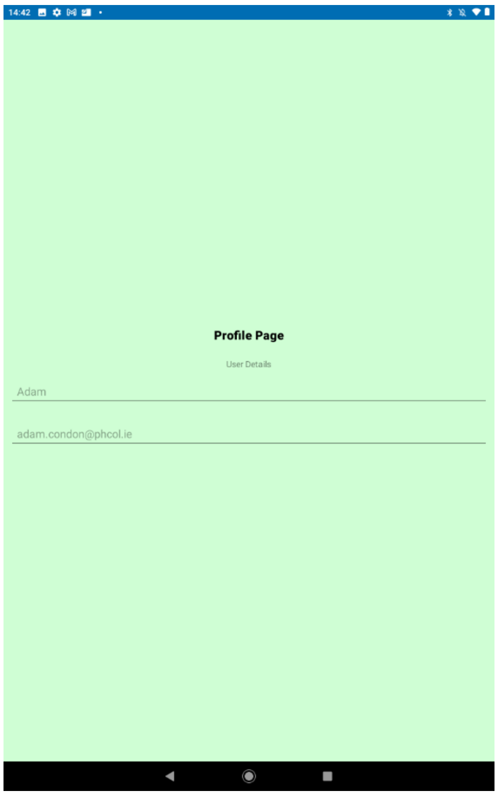
14:42

Free Movies

THE GENERAL
KICK ASS
SNATCH
THE WRESTLER
THE SCORE
MR.MAGORIUM'S WONDER EMPORIUM
PROJECT X
HOME
STRAIGHT OUTTA COMPTON
MATILDA
GREEN STREET HOOLIGANS
LIMITLESS
NEVER BEEN KISSED
THE GIRL NEXT DOOR
DEAR JOHN
CINDERELLA MAN
SCARY MOVIE 5
KEVIN AND PERRY GO LARGE
FAST AND FURIOUS 9
ANNE FRANK (THE WHOLE STORY)
THE CURSE OF HUMPTY DUMPTY
THE EVIL DOWN STREET
DECEIVE
BE CAREFUL

Profile Page

Should Store the users' details that are stored in the database. The users can view these details, update them if needed and add a bio or preference to their account. Example is obfuscated for client details privacy.



14:42

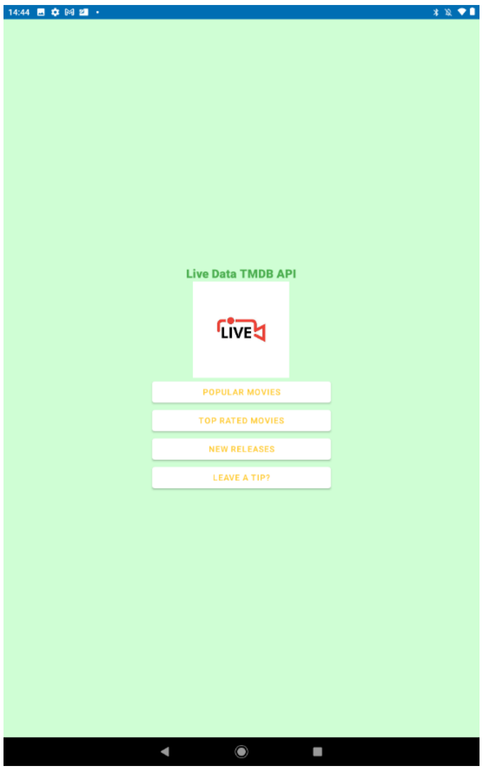
Profile Page

User Details

Adam

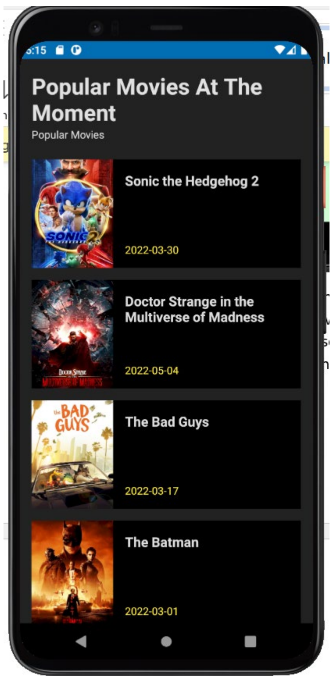
adam.condon@phcol.ie

Live Data Home Page



This page is used similarly to the logged in activity allowing the user to navigate to either Popular, top rated or new released movies based off live data extracted from the TMDB API.

Popular Movies:


Popular Movies At The Moment

Popular Movies

Sonic the Hedgehog 2
2022-03-30

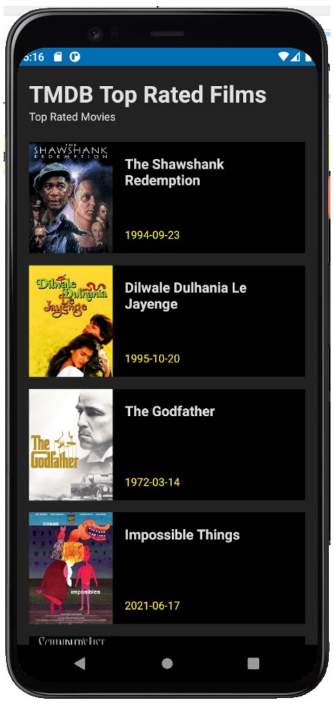
Doctor Strange in the Multiverse of Madness
2022-05-04

The Bad Guys
2022-03-17

The Batman
2022-03-01

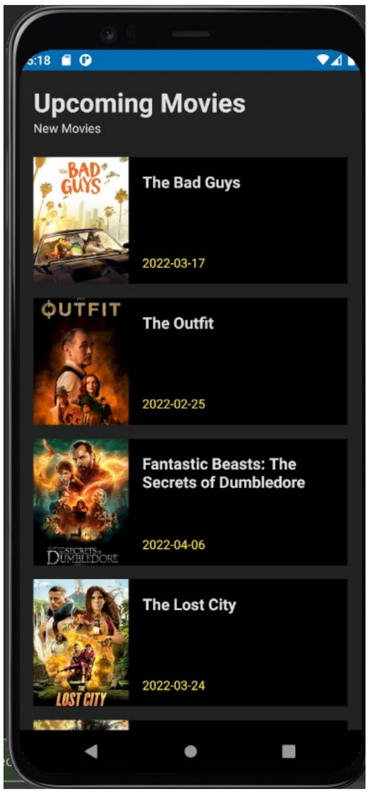
Popular Movies at the Current time of 11/05/2022 17:16

Top rated:



Popular Movies at the Current time of 11/05/2022 17:16

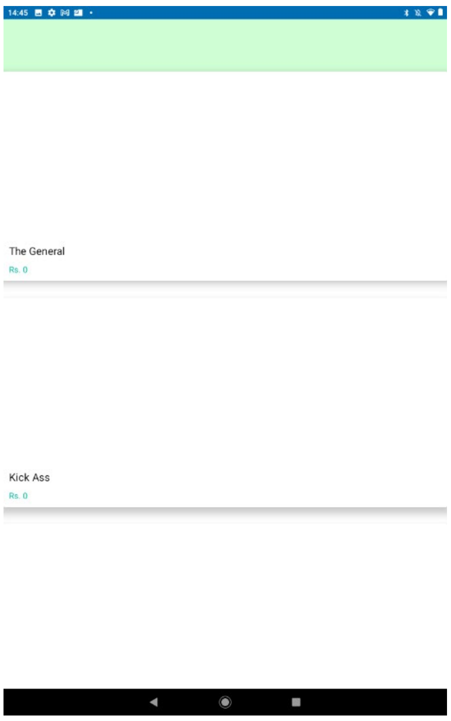
Upcoming Movies:



New Releases and upcoming Movies at the Current time of 11/05/2022 17:19

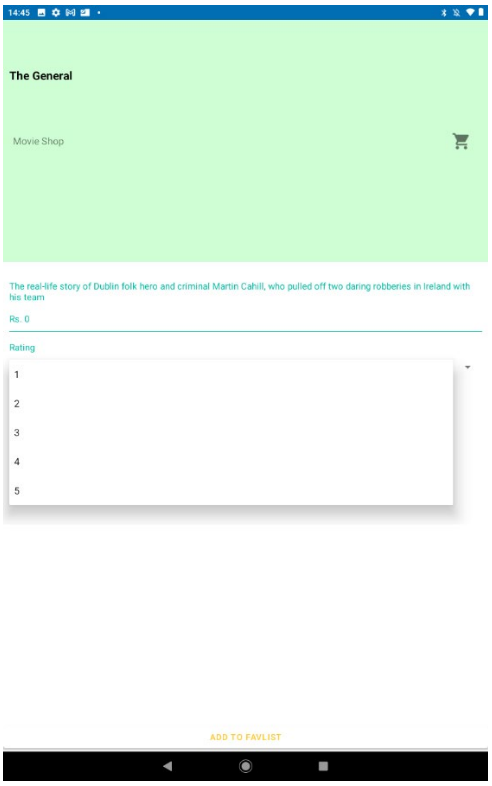
Home Free Movie Recommendations

This activity is used to access all the free movies advocate has on their local database for the users to rate, add to a Wishlist and receive recommendations based off what genres they like. The recommendations the user will receive are movie links stored in the Movies activity so a user can watch recommended movies right away without a charge fee.



Cart Activity

Allows the user to add a rating and add a movie to their Wishlist that they like or watched and save it to their own account.



14:45

The General

Movie Shop

The real-life story of Dublin folk hero and criminal Martin Cahill, who pulled off two daring robberies in Ireland with his team

Rs. 0

Rating

1

2

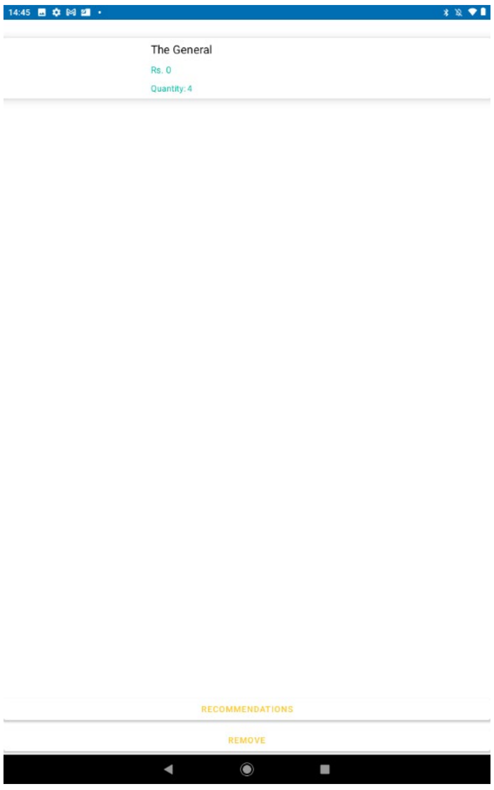
3

4

5

ADD TO FAVLIST

Wishlist



14:45

The General

Rs. 0

Quantity: 4

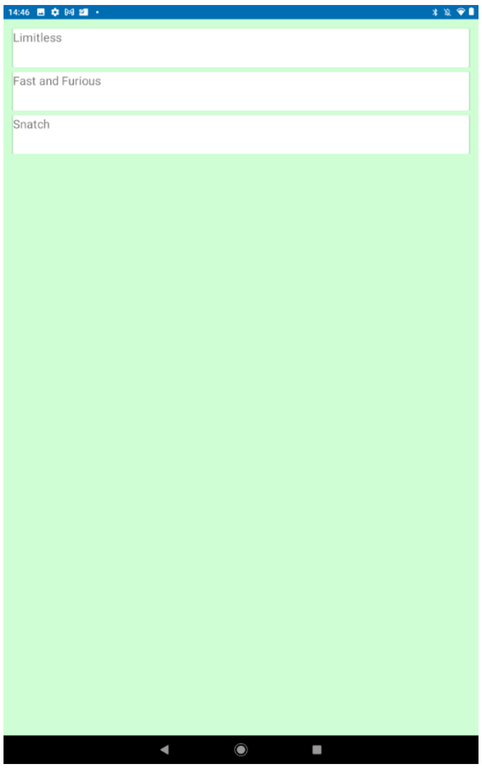
RECOMMENDATIONS

REMOVE

Stores the users watched/liked movies and they can navigate to recommendations from the cart.

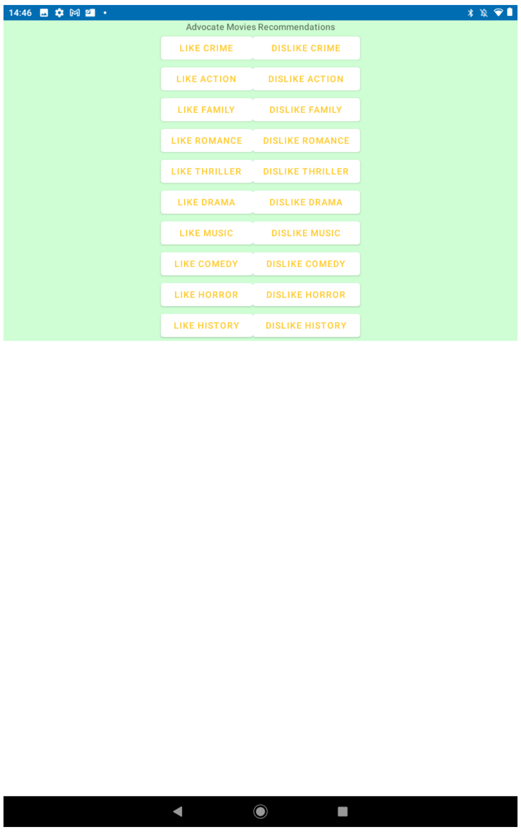
Genre Recommendations

Sample activity of Action movies recommended by Advocate for user to avail of for free.



Liked Unliked for Genre Recommendations

This activity allows the user to like and dislike movies based off their genre is the user likes a certain genre/genre’s they can receive recommendations for movies to watch for free within the app and dislike any other genre they don’t like.



5.0 Testing

I took various approaches to testing advocate programmatically, automatically, and through user testing. All tests that involved users where required to fill out this consent form and allowed me to take some of the following tests. The following approaches and methods have all been recorded consensually and a few examples have been noted for reference. You can find the results for these tests at the bottom of the report or Test Results in the Contents page.

5.1 Advocate Consent Forms

Consent Form for User testing Advocate

- I ………………………… agree to perform tests on Advocate
- I Understand and I can withdraw or pass any questions or tasks asked of me
- I understand my answers and activity will be recorded and be audible
- I am free to be contacted, if necessary, after the tests are preformed

Signature ……………………………..

Date ……………………………………

• I understand that all testing techniques can be recorded both visually and audibly

. • I understand that discussed topics and answers can be anonymously quoted in documentation and reports.

• I understand that participation involves testing an application on an Android device.

• I understand that I am free to contact the testing hosts to seek further information or clarification on anything asked or covered during or after testing has concluded.

Signed……………………………………………

Date ……………………………………………

Host/Research Form

- I ………………………… agree to participate in Advocate group to help benefit the development of Advocate with Adam Condon.
- I am aware of Advocates Aims and Purposes.
- I understand if I do not sign this form Advocate will not be eligible to store and record any answers or data I provide.
- I understand I am free to contact the group if I am curious about my data or have any other queries.
- I can withdraw from the group at any time once requested.

Signed …………………………..

Date ………………………………

5.2 Trunk Testing

Evidence of Trunk test:
https://studentncirl.sharepoint.com/sites/Testing839/Shared%20Documents/General/Recordings/View%20Only/New%20channel%20meeting-20220511_003444-Meeting%20Recording.mp4?web=1

Questions:

How would you go about registering an account with Advocate?

How would you navigate and Login to advocate?

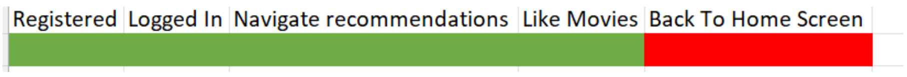
How would you go about getting recommendations for a movie?

How do you think you could like a movie and view it in your own personalised liked list?

How Would you navigate back to the Home Page?

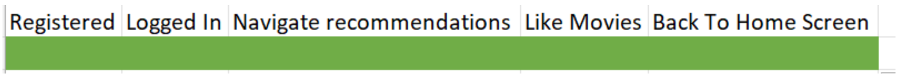
Lastly What do you think is the most appealing thing about the app's features, colours, and theme

User Bradley score 5/6 83.33%:



Registered	Logged In	Navigate recommendations	Like Movies	Back To Home Screen
🟩	🟩	🟩	🟩	🟥

Taylor Score 6/6 100%:



Registered	Logged In	Navigate recommendations	Like Movies	Back To Home Screen
🟩	🟩	🟩	🟩	🟩

Bradley Managed to Complete each task Successfully bar navigating back to the home screen of navigate after liking movies this occurred to me and I added a button to navigate back to the home screen in each activity.

Taylor Managed to Complete each task Successfully which was a good sign of the navigation flow but also showed his knowledge of tech devices.

5.3 5 Second Test

Evidence of 5 second test:

https://youtu.be/R588QrVkuNw

The five second test was used to see what a blind user would recognise most about the app over a 5 second period of viewing some of the apps main features and what stands to them the most including what doesn't. the following questions were asked to each user. A countdown was used to display a screenshot of the app and the user would have to gather and memorise what they see in each slide per 5 seconds.

Questions:

What stands out the most on this activity?

What's the apps name?

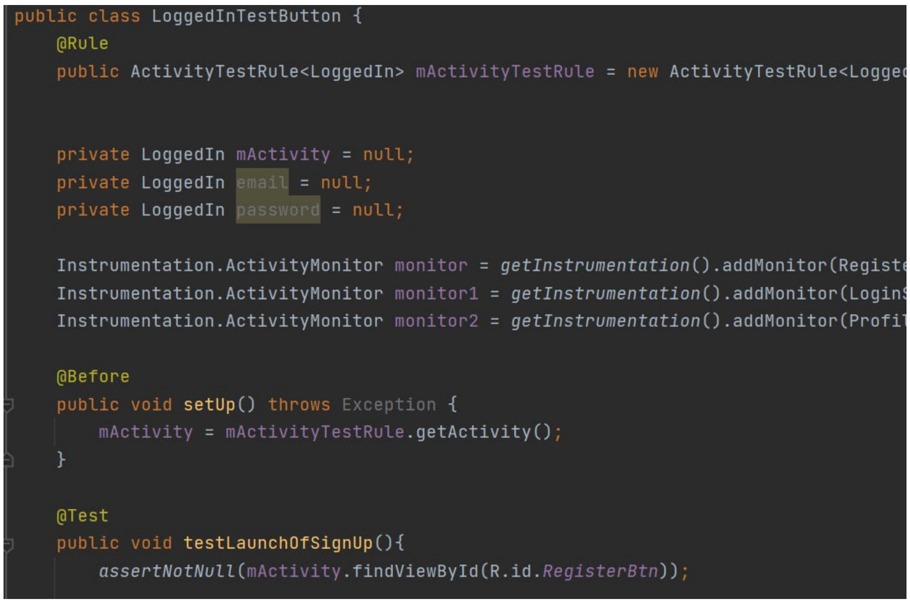
Did any colours, text or features stand out in comparison to others?

What do you think was the standout feature and how do you think you would navigate to movie recommendations?

Taylor noted that he didn't know what advocate meant which was actually the name of the app and didn't mean anything to do with navigation around the app, he was unaware of this as he was blind to the app for this test but noted everything else was appealing to his eye.

5.4 J-Unit Testing

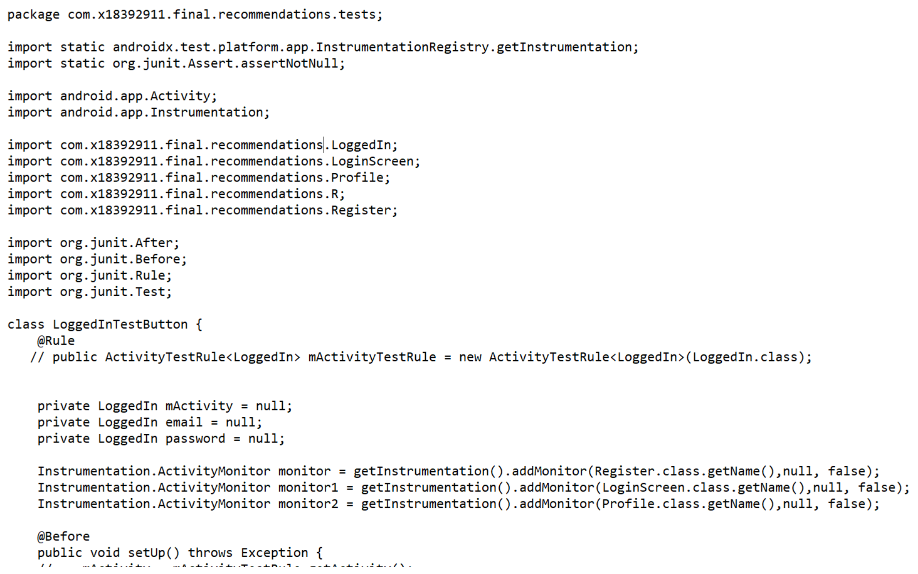
Perform J Unit test using the JAR and Kotilin JDK through NetBeans for complex activities. This tests for the quality of methods and gives the app a good sense of automated testing rather than manual testing. It's a great approach to testing and ensures app quality. I used NetBeans built in J Unit testing terminal.



```
public class LoggedInTestButton {
    @Rule
    public ActivityTestRule<LoggedIn> mActivityTestRule = new ActivityTestRule<Logge


    private LoggedIn mActivity = null;
    private LoggedIn email = null;
    private LoggedIn password = null;

    Instrumentation.ActivityMonitor monitor = getInstrumentation().addMonitor(Registe
    Instrumentation.ActivityMonitor monitor1 = getInstrumentation().addMonitor(LoginS
    Instrumentation.ActivityMonitor monitor2 = getInstrumentation().addMonitor(Profi

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

    @Test
    public void testLaunchOfSignUp(){
        assertNotNull(mActivity.findViewById(R.id.RegisterBtn));
```

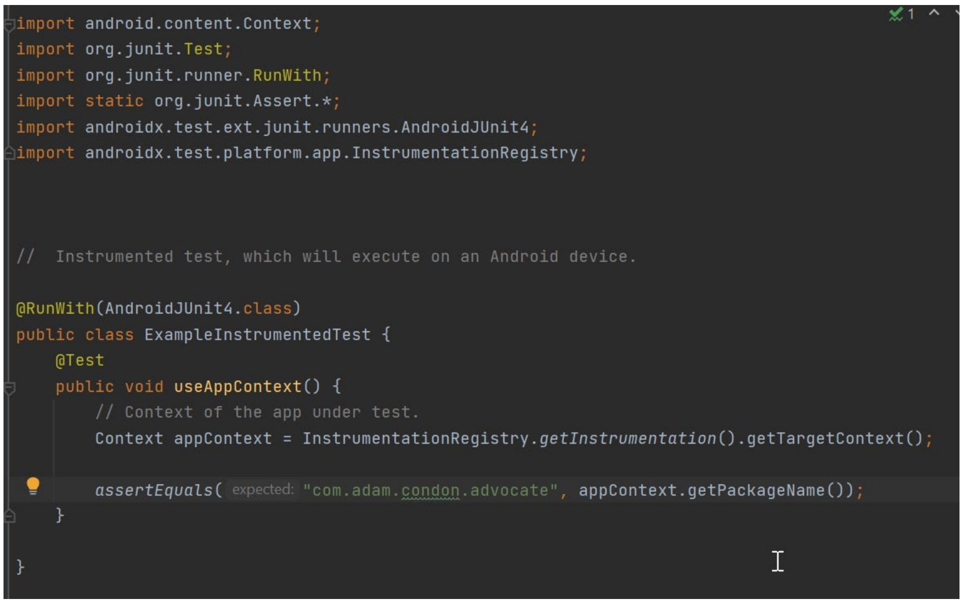
Junit test for LoggedIn Activity to test all the buttons used to navigate to other activities and from the login and register activities.

This is a Junit test carried out to test the restrictions of the Registration class when you access Advocate. It checks if the email has an @ symbol inside and asserts true or false if its with or without it and a sample email address for reference so the test knows what to expect. It also tests the password length if it’s too small it fails and if it’s the right limit or more it passes.



```
package com.x18392911.final.recommendations.tests;

import static androidx.test.platform.app.InstrumentationRegistry.getInstrumentation;
import static org.junit.Assert.assertNotNull;

import android.app.Activity;
import android.app.Instrumentation;

import com.x18392911.final.recommendations.LoggedIn;
import com.x18392911.final.recommendations.LoginScreen;
import com.x18392911.final.recommendations.Profile;
import com.x18392911.final.recommendations.R;
import com.x18392911.final.recommendations.Register;

import org.junit.After;
import org.junit.Before;
import org.junit.Rule;
import org.junit.Test;

class LoggedInTestButton {
    @Rule
   // public ActivityTestRule<LoggedIn> mActivityTestRule = new ActivityTestRule<LoggedIn>(LoggedIn.class);


    private LoggedIn mActivity = null;
    private LoggedIn email = null;
    private LoggedIn password = null;

    Instrumentation.ActivityMonitor monitor = getInstrumentation().addMonitor(Register.class.getName(),null, false);
    Instrumentation.ActivityMonitor monitor1 = getInstrumentation().addMonitor(LoginScreen.class.getName(),null, false);
    Instrumentation.ActivityMonitor monitor2 = getInstrumentation().addMonitor(Profile.class.getName(),null, false);

    @Before
    public void setUp() throws Exception {
```

5.5 Instrumented Testing



```
import android.content.Context;
import org.junit.Test;
import org.junit.runner.RunWith;
import static org.junit.Assert.*;
import androidx.test.ext.junit.runners.AndroidJUnit4;
import androidx.test.platform.app.InstrumentationRegistry;



//  Instrumented test, which will execute on an Android device.

@RunWith(AndroidJUnit4.class)
public class ExampleInstrumentedTest {
    @Test
    public void useAppContext() {
        // Context of the app under test.
        Context appContext = InstrumentationRegistry.getInstrumentation().getTargetContext();

        assertEquals( expected: "com.adam.condon.advocate", appContext.getPackageName());
    }

}
```

This instrumented test can be run on an android device which builds the application and tests for any bugs or issues inside the system.

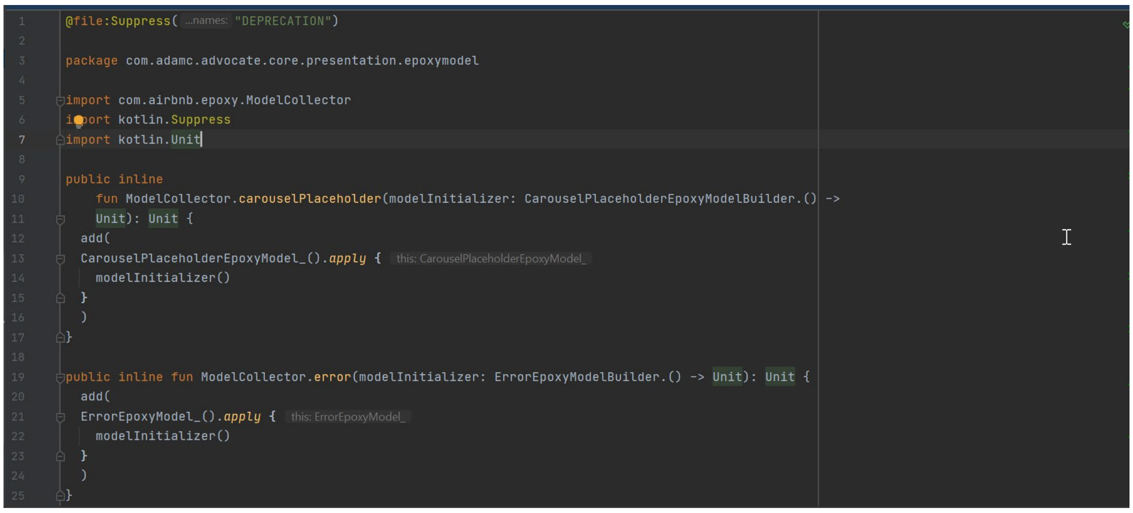
5.6 Users and Database

- Ensure users are saved and stored on the database
- Ensure removed users cannot access the App and non-registered users also.
- Ensure the database only saves relevant and good data.
- Ensure database reacts within 4 seconds upon creating and removing users
- Ensure the database saves all the data and receives and displays data as expected
- Ensure activities take no longer than 5000 milliseconds to load.

5.7 Manual Testing

- Ensure login and Registration works as expected, The User can successfully register and login to the system by typing in the correct credentials.
- Once a user is logged in, they should not have to constantly login unless logged out.
- Typing malicious code into the textboxes of registration and login to ensure SQL Injection cannot be performed.
- All tests for Correct and incorrect passwords will be accrued out manually and should contain at least 8 characters and a special character.
- Test on phone and other devices such as tablet to ensure its responsive for all devices.
- I will also make sure each ethics form is completed fully and accurately after receiving my supervisor’s approval.

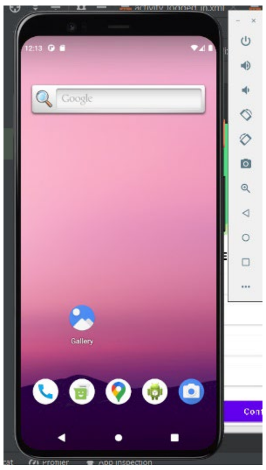
5.8 Unit Testing



```
@file:Suppress( ...names: "DEPRECATION")

package com.adamc.advocate.core.presentation.epoxymodel

import com.airbnb.epoxy.ModelCollector
import kotlin.Suppress
import kotlin.Unit

public inline
    fun ModelCollector.carouselPlaceholder(modelInitializer: CarouselPlaceholderEpoxyModelBuilder.() ->
    Unit): Unit {
  add(
  CarouselPlaceholderEpoxyModel_().apply {  this: CarouselPlaceholderEpoxyModel_
    modelInitializer()
  }
  )
}

public inline fun ModelCollector.error(modelInitializer: ErrorEpoxyModelBuilder.() -> Unit): Unit {
  add(
  ErrorEpoxyModel_().apply {  this: ErrorEpoxyModel_
    modelInitializer()
  }
  )
}
```

5.9 Performance Testing

- Testing the Application logging on when the users Wi-Fi is down or bad.
- Ensuring the users login or loading time is no longer than 4 seconds to successfully load.
- Make sure logging out takes no longer than 4 seconds to load.
- Running the application successfully on Emulator, Tablet and Mobile device.
- Ensuring all buttons and activities take no longer than 4 seconds to load.



Here is an example of me testing Advocate on an emulator device.

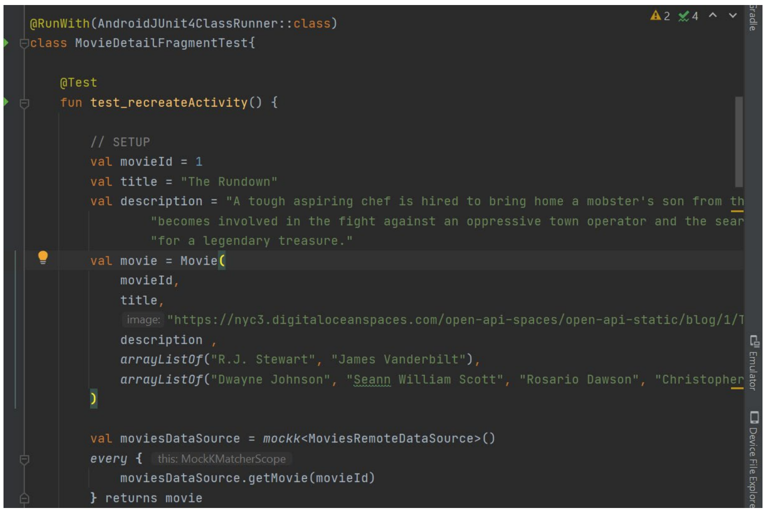
I use the emulator to run my application and test the app on other android devices.

From android 6 to the latest and different phones/tablets.

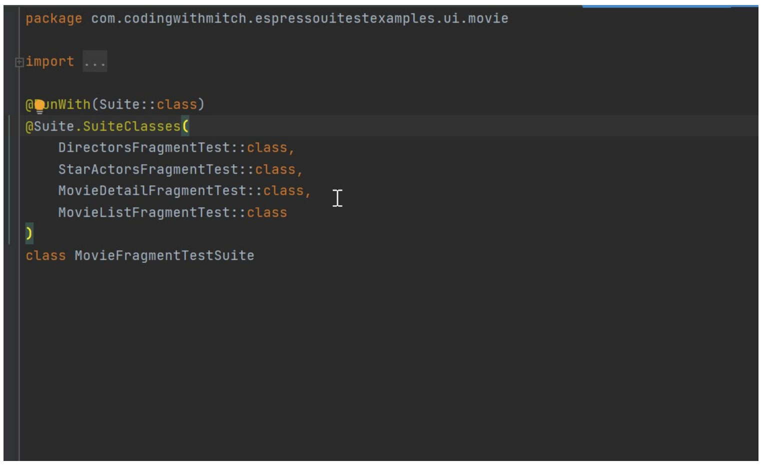
I also use an acer tablet android 10 to test the application on a real working device and for its responsiveness.

5.10 Espresso Testing

Is embedded in android studio for Ui testing, it allows me to create tests that mock the user’s interaction with the GUI and automatic navigation. The test was made to recreate the movie detail fragment.

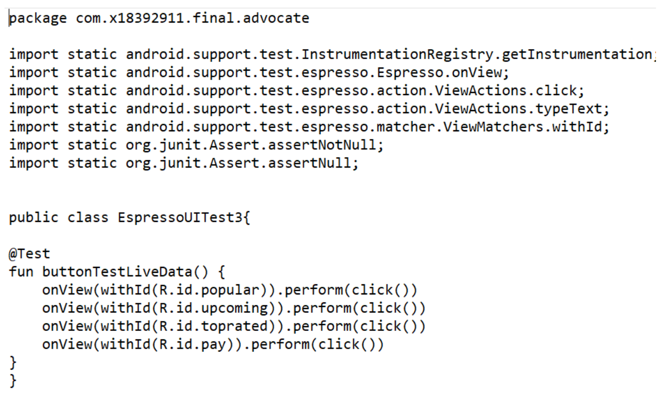


```
@RunWith(AndroidJUnit4ClassRunner::class)
class MovieDetailFragmentTest{

    @Test
    fun test_recreateActivity() {

        // SETUP
        val movieId = 1
        val title = "The Rundown"
        val description = "A tough aspiring chef is hired to bring home a mobster's son from th
                "becomes involved in the fight against an oppressive town operator and the sear
                "for a legendary treasure."
        val movie = Movie(
            movieId,
            title,
            image: "https://nyc3.digitaloceanspaces.com/open-api-spaces/open-api-static/blog/1/T
            description ,
            arrayListOf("R.J. Stewart", "James Vanderbilt"),
            arrayListOf("Dwayne Johnson", "Seann William Scott", "Rosario Dawson", "Christopher
        )

        val moviesDataSource = mockk<MoviesRemoteDataSource>()
        every {  this: MockKMatcherScope
            moviesDataSource.getMovie(movieId)
        } returns movie
```

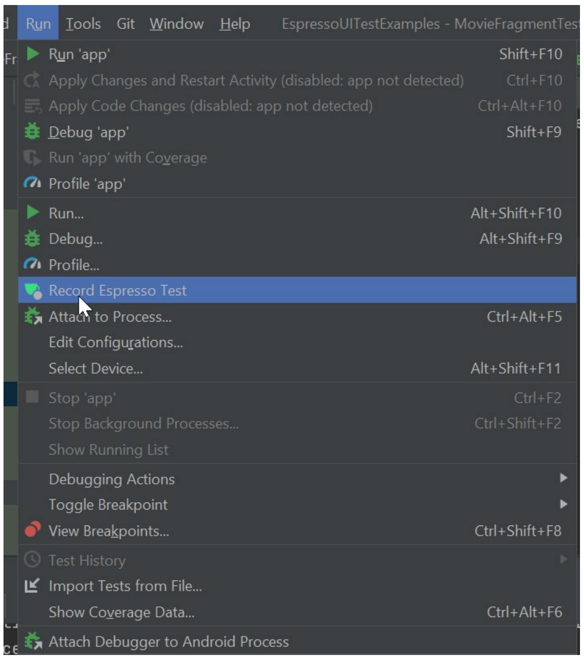


```
package com.codingwithmitch.espressouitestexamples.ui.movie

import ...

@RunWith(Suite::class)
@Suite.SuiteClasses(
    DirectorsFragmentTest::class,
    StarActorsFragmentTest::class,
    MovieDetailFragmentTest::class,
    MovieListFragmentTest::class
)
class MovieFragmentTestSuite
```

```
package com.x18392911.final.recommendations.tests;

import static android.support.test.InstrumentationRegistry.getInstrumentation;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.action.ViewActions.typeText;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static org.junit.Assert.assertNotNull;
import static org.junit.Assert.assertNull;

import android.util.Patterns;

import org.junit.Test;

//espresso onclick test

public class MapsTest {


onView(withContentDescription("Google Map")).perform(click());

onView(withContentDescription("marker title. ")).perform(click());

private UiDevice uiDevice = UiDevice.getInstance(getInstrumentation());
private UiObject mMarker1 = uiDevice.findObject(new UiSelector().descriptionContains("The title of marker"));
try {
    mMarker1.click();
} catch (UiObjectNotFoundException e) {

    e.printStackTrace();
}

}
```



```
package com.x18392911.final.advocate

import static android.support.test.InstrumentationRegistry.getInstrumentation;
import static android.support.test.espresso.Espresso.onView;
import static android.support.test.espresso.action.ViewActions.click;
import static android.support.test.espresso.action.ViewActions.typeText;
import static android.support.test.espresso.matcher.ViewMatchers.withId;
import static org.junit.Assert.assertNotNull;
import static org.junit.Assert.assertNull;


public class EspressoUITest3{

@Test
fun buttonTestLiveData() {
    onView(withId(R.id.popular)).perform(click())
    onView(withId(R.id.upcoming)).perform(click())
    onView(withId(R.id.toprated)).perform(click())
    onView(withId(R.id.pay)).perform(click())
}
}
```



Run
Tools
Git
Window
Help
EspressoUITestExamples - MovieFragmentTes
Run 'app' Shift+F10
Apply Changes and Restart Activity (disabled: app not detected) Ctrl+F10
Apply Code Changes (disabled: app not detected) Ctrl+Alt+F10
Debug 'app' Shift+F9
Run 'app' with Coverage
Profile 'app'
Run... Alt+Shift+F10
Debug... Alt+Shift+F9
Profile...
Record Espresso Test
Attach to Process... Ctrl+Alt+F5
Edit Configurations...
Select Device... Alt+Shift+F11
Stop 'app' Ctrl+F2
Stop Background Processes... Ctrl+Shift+F2
Show Running List
Debugging Actions
Toggle Breakpoint
View Breakpoints... Ctrl+Shift+F8
Test History
Import Tests from File...
Show Coverage Data... Ctrl+Alt+F6
Attach Debugger to Android Process

For testing purposes, I used Espresso to test the UI of the Advocate Free movies activity and Cart Activity allowing the Espresso Idle Resource to be used and tested with dummy data and to see if the data would be populated correctly with the navigation of buttons.

5.11 Testing Results:

Name	Taylor
5 Second Test	100% successful but did not know what advocate meant initially but still seen it.
Trunk Test	100% Successful navigated successfully around the app
Gender	Male
Tech Level	High

Name	Bradley
5 Second Test	100% Successful.
Trunk Test	86% successful did not know how to navigate back to the home page.
Gender	Male
Tech Level	Moderate

Name	Liz
5 Second Test	100% successful
Trunk Test	80% Couldn't register successfully the first time but got it the second time round.
Gender	Female
Tech Level	Low

5.12 Exit Survey Questions

(Liz Answers Example)

Did you like the app?

Yes, I found the project extremely benefice business wise

Did you find the App complex at all?

It was not too complex, and I have experience in this area especially testing.

Anything you'd change?

I'd like to see more of the same and one or two more movies.

What apps does it remind you of and what do they do that Advocate does not?

This app reminds me an awful lot about Netflix the only thing this app lacks in comparison to Netflix is the movie streaming and the number of movies the contains but therefore there is subscriptions in place unlike Advocate.

Did you manage to use all our workflows successfully?

All the workflows ran individually and were very smoothly transitioned.

Could the app be designed better?

I think the design is appropriate for this sort of mobile app maybe more pictures would help.

Was there any issue with buffering/pages loading?

No everything ran smoothly

Would you recommend Advocate to a friend?

Yes, absolutely this app is very useful.

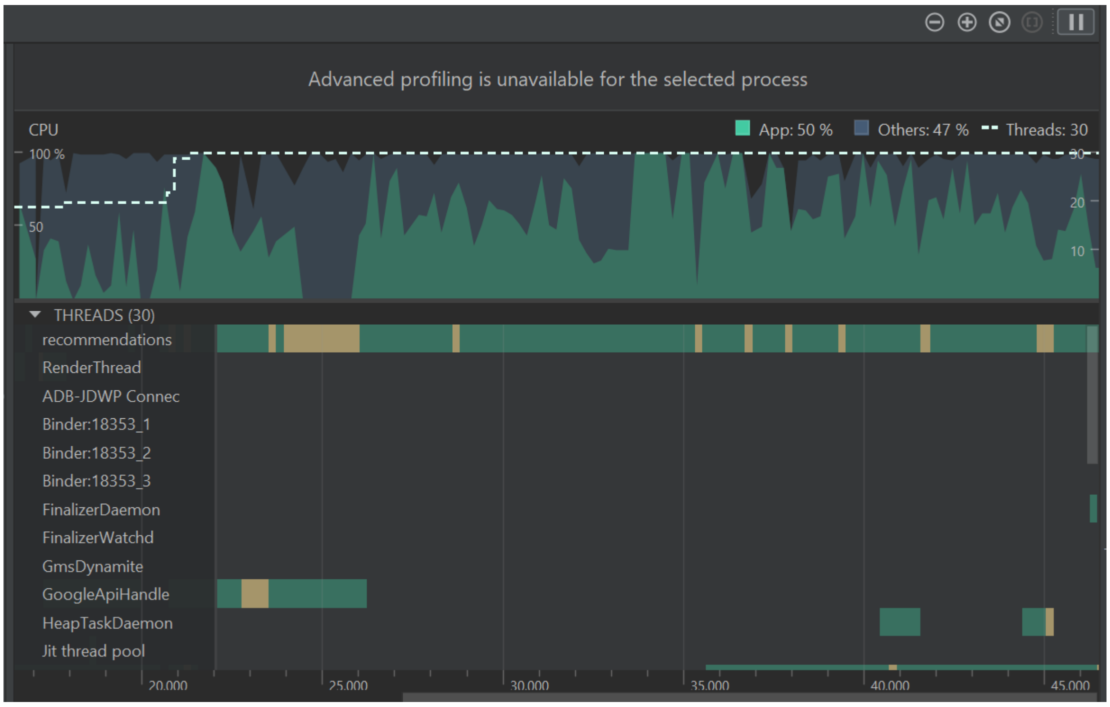
What would you rate Advocate out of 10?

I rate the app 10/10

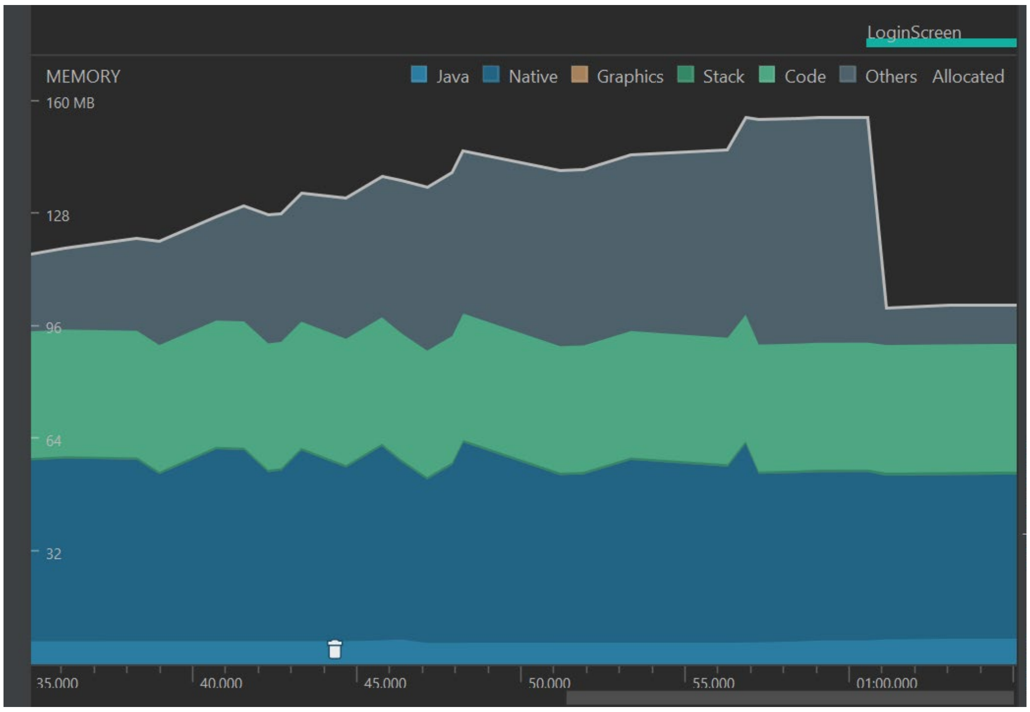
6.0 Evaluation

Throughout my project process Android studio seemed to take a lot out of my machine running tests the emulator and the app build/debug methods but after checking my CPU memory and power usage it resulted relatively low in comparison to other applications and processes, I used. To speed things up between sessions id clear some cache or restart my computer to enhance the performance time. Below is an example of the power used in the process.

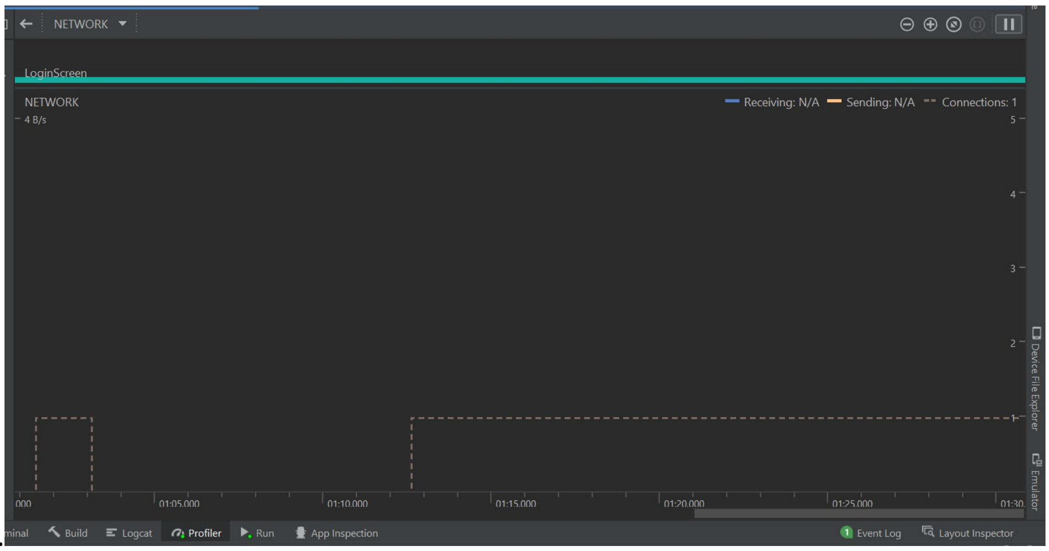
While developing the application Advocate was monitored by the Android Studio “profiler” where the CPU manager never exceeded 50% which means the use of memory was relatively moderate on my machine. The most memory taking as you can see at the top of the chart is pointed towards the recommendations model which communicates to TensorFlow lite and takes a good chunk of the memory.



As you can see from the next graph the activity that stands out the most is the login screen as it is the most visited screen on the app to enter Advocate you must access this page. The graph also shows some of the code and languages that take most of the memory up apart from the Login screen which is great to know and to take into consideration when going into production.

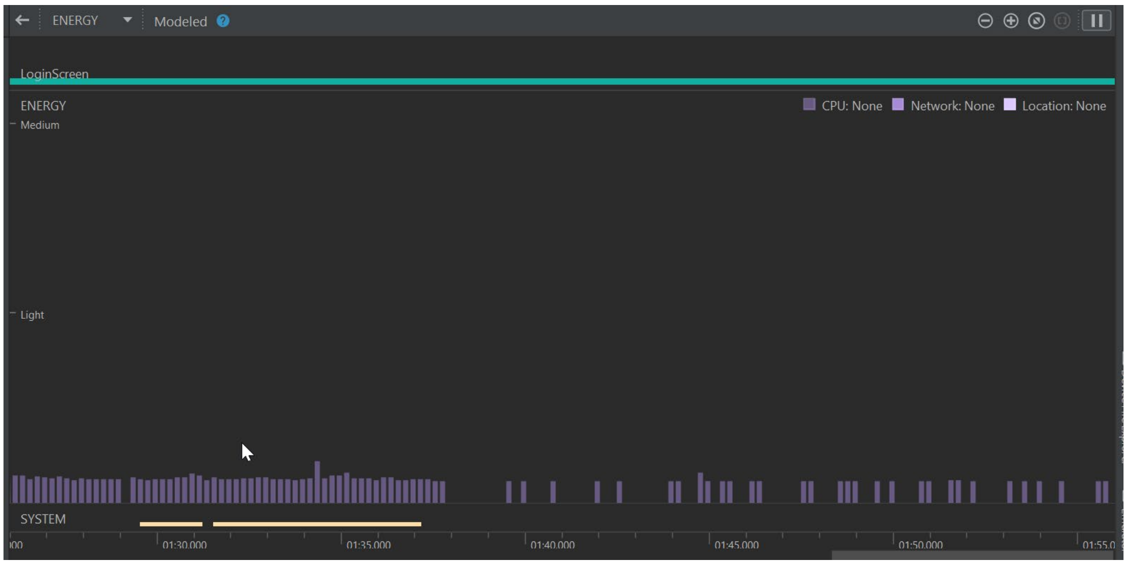


Once again on the Network Graph the login screen is the activity with the most use of network bites although the network usage is very low for this app and is very good for speed and performance for users when using Advocate.

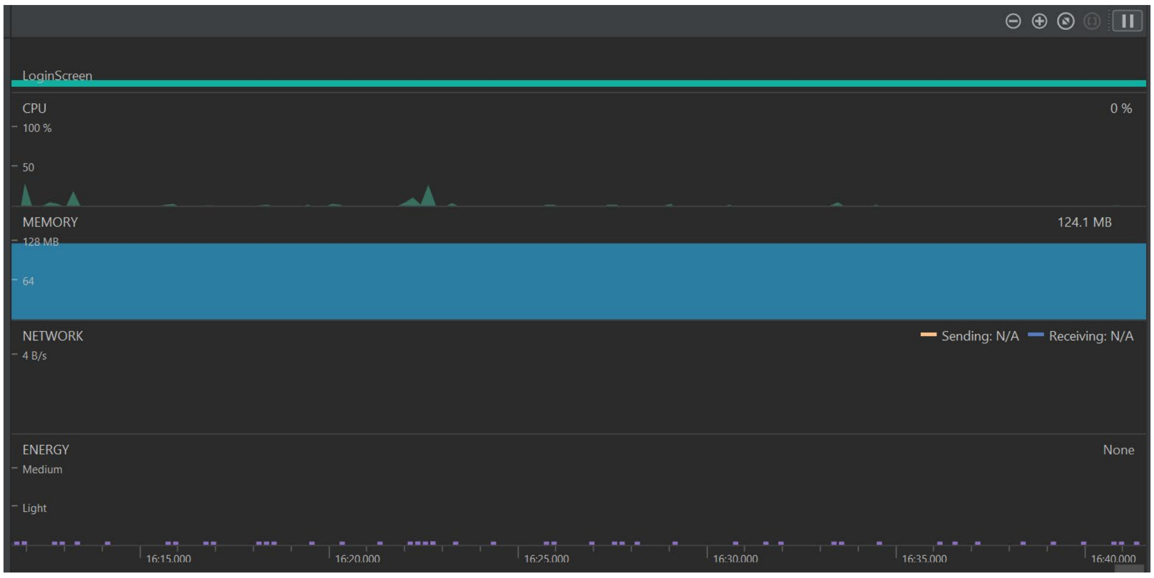


.

CPU Energy data was also monitored using the Android Profile With the network and location of the product monitored which also stayed consistently low throughout the project process. The chart has a limit of Data which is considered “Light” and as you can see the Energy Usage of Advocate is way below this.



Full Screen of all processes Running:



6.1 Heuristic Evaluation

From testing and my results that I received from end users I had a great idea of how user will use advocate who have very little to no knowledge in tech to quite advanced in tech which was a great variety and comparison of opinions based off what the user sees, how they interact with the app and how they behave in certain situations. Based on the users feedback I changed a few things like the apps name font, buttons layout and colours

6.1.0 Error Prevention

Advocates use of email validation and malicious security in the Login and register activities gives the app a great sense of error and malicious prevention through error handling also the app provides toast messages to the user if an incorrect value is entered of if any navigation or internet issues are present.

6.1.1 Flexibility

The user can use a keyboard to type into any text fields Advocate requires information or they can use the text recognition feature in the tipping section of the app to scan their details and allow the system to populate the fields automatically.

6.1.2 Aesthetic design

The use of the colour scheme came traditionally from being a strong republican that I am choosing green, white, and orange for the apps main colour but mainly for the fact that the users can clearly identify and titles buttons or pages clearly with the difference of colours and their suitability to the app.

6.1.3 User guide

Allows users to read through a step-to-step guide on how to access every activity in the app and to contact the support page x18392911@student.ncirl.ie if they cannot troubleshoot the problems they are having through the user guide which shouldn't really be necessary due to the high rate of successful testing.

7.0 Conclusions

Advocate as an idea is an excellent one. A lot of movie applications now a days cost up to 10€ a month that provide recommendations to user but only after they have watched movies. With just the click of a few buttons Advocates Machine Learning trained model can be used to give a user meaningful recommendation to their own personal accounts. Not only does advocate recommend movies in this way it allows users to watch some movies free of charge on a legal site "YouTube" it also allows the user to locate nearby Movie Theatres according to the user's current location once this permission is granted by the user.

In all the movie applications I have seen this feature has never been implemented and is quite useful and innovative to use due to the fact the user just must click one button and can see what cinemas are closest to them and then can watch some of the movies they were either recommended or that they see from the Apps Live data such as Popular/Newly released Movies. This saves more time and money when the user wants to view the movie.

Advantages

- Gives individual meaningful Recommends of high-quality movies at the click of the button
- Offers the user movies, to like certain movies form a huge dataset, search for a movie of choice.
- UI Friendly and is free of charge unlike most movie apps

- Unlike any other movie apps, it shows local Cinemas and movie shops where the user can buy movies all at the click of a button.
- Allows the user to buy movies they want specifically if they can then be found on the advocate database such as YouTube movies. The user should then receive email confirmation upon purchase.
- Provides a positive user experience and is suited to all Movie lovers.

Disadvantages

- Very high Competition in the market
- Users can't watch all movies requested for free and will have to pay externally for movies wanted.
- Users must have an android device to install the app.

8.0 Further Development or Research

With more time and Resources this project could compete with the best movies on the app store now by getting all the newest and greatest movies of all time. This app could very well have potential with the right guidance and support. Regarding the recommendations of movies lately a lot of users tend to struggle to pick what movie they want to watch and tend to stick to "comfort movies". Some studies have proven that those who stick to watching the same things or movies suffer with a mental illness.

Advocate is made to give the user meaningful recommendations which will prevent users from watching the same things constantly by choosing the best possible alternatives to what they already like watching and from here the users can decide from a variety of recommendations as well as accessing the live data movie types. This mental illness has led to social anxiety and loneliness of individuals as they are constantly watching the same movies/tv shows all the time and fear change. This then transfers into real life making them constantly stay in their "comfort zone". Allowing the user to get out of their comfort zone and discover new movies allows them to feel a sense of achievement and joy when they enjoy them.

9.0 References

References

Berry, S., 2011. *asistdl.onlinelibrary.wiley.com.* [Online]
Available at: https://asistdl.onlinelibrary.wiley.com/doi/full/10.1002/meet.14504701402
[Accessed November 2021].

cbs, 2022. *cbsnews.* [Online]
Available at: https://www.cbsnews.com/news/depression-loneliness-linked-to-binge-watching-television/

IMDB, 2021. *www.imdb.com.* [Online]
Available at: https://www.imdb.com/

N, J., 2016. *https://expertboxing.com.* [Online]
Available at: https://expertboxing.com/best-boxing-gloves-review
[Accessed 07/11/21 November 2021].

Schmydt, M., 2020. *link.springer.com.* [Online]
Available at: https://link.springer.com/chapter/10.1007/978-3-030-61146-0_15
[Accessed November 2021].

unknown, 2018. *https://www.runningshoesguru.com.* [Online]
Available at: https://www.runningshoesguru.com/best-running-shoes-wizard/
[Accessed 07/11/21 November 2021].

10.0 Appendices

10.1 New Amended Project Proposal

National College of Ireland

Project Proposal

Advocate

01/11/2021

Bachelor Of Science (Honours) in Computing

Software Development

2021/22

Adam Condon

X18392911

X18392911@student.ncirl.ie

Contents

1.0 Objectives.. 75

1.0 Overview 75

1.1 Goal to be achieved 2

1.2 Machine Learning/AI 75

1.3 Features 75

2.0 Background .... **Error! Bookmark not defined.**

2.1 Reason...3

3.0 State of the Art...3

3.1 Prime examples of Studies 3

3.2 Personal Experience 3

4.0 Technical Approach.. 4

4.1 Use Case Diagram and Approach 4

4.2 Schedule and Management... 4

4.3 Main Aspects of the App 4

5.0 Technical Details 5

6.0 Special Resources Required... 5

7.0 Project Plan 6

8.0 Testing... 7

11.0 Objectives

Overview

Payment	A User should be able to buy a film and receive a confirmation email
API	Maps API providing local Cinemas where movies can be viewed.
Movies	The user will have access to hundreds of movies choices where they can determine which ones they like from the list and be recommended movies of similar interest with Googles Recommendation AI implementation. Individual users will be recommended films when they login to their respective accounts.
Maps	Will Present Cinemas near by the user's location once permission is granted

Firebase	Google Firebase will be used for Login and Registration system and other relevant data to be stored/retrieved.

1.1 Goal to be Achieved

This project is set to achieve the best possible movie recommendations to users. The user should be able to Register, and Login have a pleasant experience having hundreds of movies to choose from. The user should be able to like a movie choice and view them all in a list where they can also unlike them. The user can also then easily access any cinemas nearby them once they accept their location to the google maps API. The user can also have the option to buy some movies.

1.2 Machine Learning/AI

The project will be based off everyone's choices of movies that are liked. Google recommendation AI, Google Big Query, Analytics and Firebase will be all used in storing filtering and allow the application to learn by the users' choices. Each time a movie is liked a group of similar movies will be recommended in the recommended section of the app. Big query is a google analytics product that allows you to analyse large amounts of data at once.

1.3 Features

Other than Movie Recommendations the user should be able to Register and Login to the app purchase movies/membership on their relevant accounts. The user should be able to see what cinemas are near their location once the permission has been accepted on the user's device. It will also contain a search engine that will allow the user to search the dataset for whatever movie they are looking for.

12.0 Background

I am a lover of movies personally and this app was a no brainer for me to choose. I love to wind down and watch a good movie. Not always do I know what movie to watch but knowing what you like helps a lot. So having movies recommended to me is a fantastic way to enhance a user's choice. Also having a large variety to choose from is helpful and beneficial to the user and the more movies the more similarities and benefits the user will gain using the app. The app will continue to learn based off the users' choices.

2.1 Reason

When it comes to selecting a movie whether it be a couple, family or just an individual movie can be repetitive, and choices can also be repetitive. By using Advocate, the user can choose a movie suitable to what everyone likes in just a tap of the screen. If a date or family night out was being planned, then the user could also allow access to the google maps feature where they could view this movie or buy it externally.

13.0 Research

There are quite a few apps such as Netflix, Disney plus and prime that recommend movies regarding users' choices. These are all paid subscriptions of movies that are stored and can be played on the app. Advocate gives this option for free and has other feature such as google maps API which Mainstream recommenders do not have. Advocate also promotes apps like Netflix and Prime Video by the fact they can be bought by the user but only if the Movie is present on their systems otherwise, they will have to look elsewhere.

3.1 Prime Examples of Studies

Apps such as next flix contain their own personalised recommendations engines based off a certain users account and choices. The app contains thousands of cluster that will eventually decide on what movies the user likes and recommend more movies ion groups based off them. I recently followed the release of Googles Recommendations AI release in 2020 and was extremely interested to how this system would work. After a lot of research and brainstorming I figured this Artificial intelligent invention could be added in my final Year project of Advocate.

3.2 Personal Experience

From personal experience I have subscribed to Netflix and have enjoyed its recommendations engine, however I wished there would be more of a variety of both old and new movies. Netflix's choice to movies can be varied leaving them vulnerable to other apps websites and streamers. Advocate allows more of a user's choice with helpful links and options to access the movie a user wants to watch that may not be on a subscripted movie app.

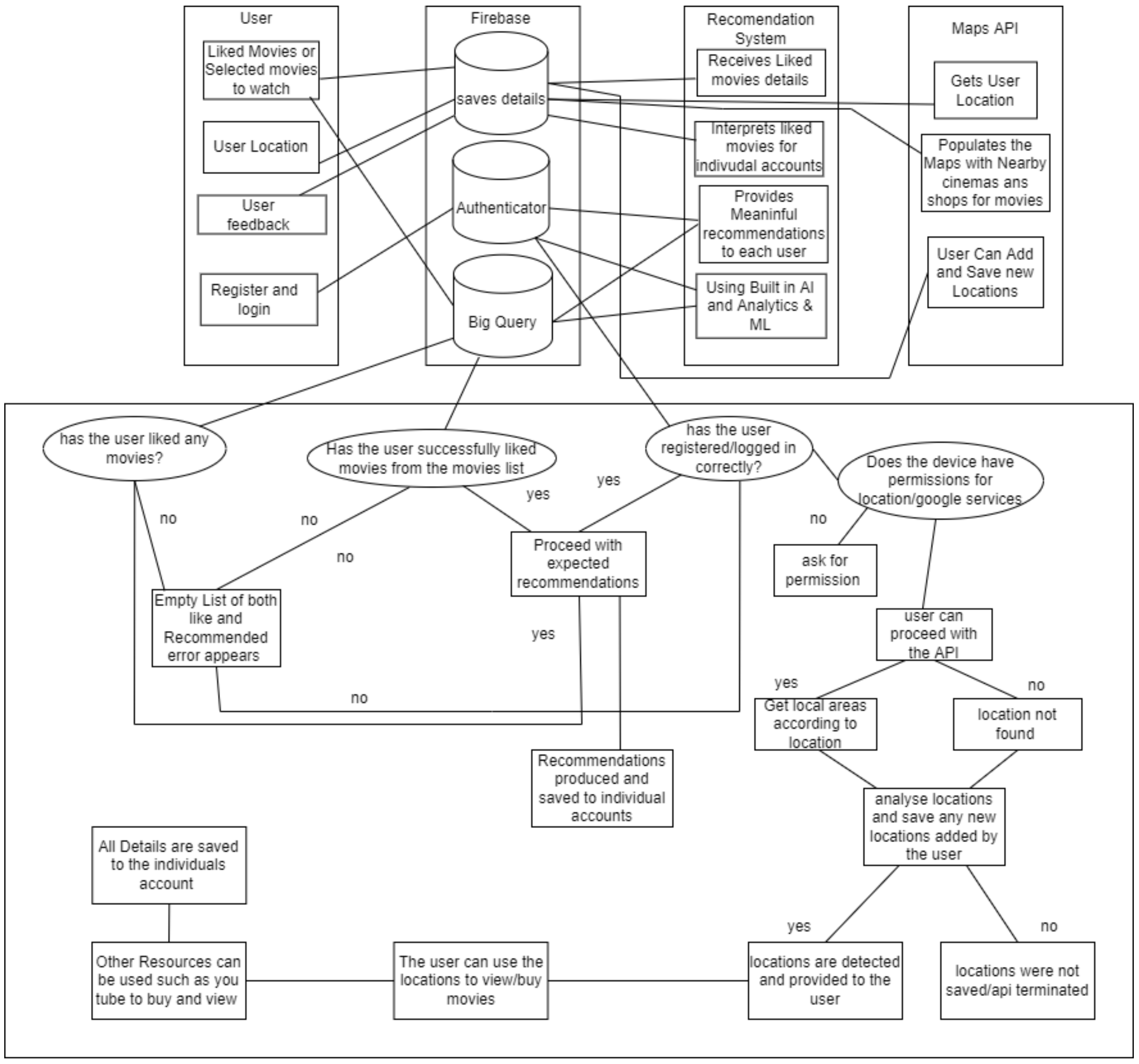
4 Technical Approach

4.1 Use Case Diagram and Approach

Below I have demonstrated a use case for my project of the different aspects and functions the app will contain. I have also provided a Gant Chart further down the document that I will follow strictly while in the development stage as it is crucial to meet the deadlines I set out, otherwise I will fall behind during the development process. I will have several submissions to meet regarding presentations and documents for this project so extra time will be needed to incorporate these assignments.

4.2 Schedule and Management

In between my implementation and development schedule I will meet once every two weeks with my project Supervisor and received feedback both on my work and for future implementation to see if I am going in the right direction or if I need to fix a few features. I am also using Trello to break down small weekly micro and macro tasks I set out to complete weekly to keep me on track.


User
Liked Movies or Selected movies to watch
User Location
User feedback
Register and login
Firebase
saves details
Authenticator
Big Query
Recomendation System
Receives Liked movies details
Interprets liked movies for indivudal accounts
Provides Meaninful recommendations to each user
Using Built in AI and Analytics & ML
Maps API
Gets User Location
Populates the Maps with Nearby cinemas ans shops for movies
User Can Add and Save new Locations
has the user liked any movies?
Has the user successfully liked movies from the movies list
has the user registered/logged in correctly?
Does the device have permissions for location/google services
Proceed with expected recommendations
Empty List of both like and Recommended error appears
ask for permission
user can proceed with the API
Get local areas according to location
location not found
Recommendations produced and saved to individual accounts
analyse locations and save any new locations added by the user
All Details are saved to the individuals account
Other Resources can be used such as you tube to buy and view
The user can use the locations to view/buy movies
locations are detected and provided to the user
locations were not saved/api terminated
yes
no

4.3 Main Aspects of the App

- Registration and Login system made developed with Kotilin and XMl to give an enhancing look to the activities. A valid registration will require an appropriate email and password these must contain certain characters and a certain number of characters to proceed, and the user will be notified if they fail in doing so. Google Firebase database will be used to store the data inputted by users.
- Google Maps Api that gets the Users location on the device upon approval and will show nearby cinemas/movie stores.
- Movies to choose from a large dataset, the option to like a movie and view recommended movies based off the liked movies the user as selected. These choices should be saved and delivered to each user that has successfully made an account.
- The user can search for movies they like amongst the app.
- The user can buy or view where to purchase a movie and will receive a confirmation email when this is complete.

5 Technical Details

The languages Implemented will be Kotilin and XML to design the UI of the App in android studio editor. External sources like Firebase will be used for databases and Tensor Flow light for the Machine learning aspect along with Google Recommendations Ai and Big Query as part of Google Analytics. I will also use Google Maps API and relevant key using Kotilin and Google Map functions. This will require google services and JSON libraries.

The Google Maps Api Feature will be developed through Kotilin Xml and of course Google Maps Api. This will need google services permissions from the user to work. All data and markers will also be stored in an internal database.

6 Special Resources Required

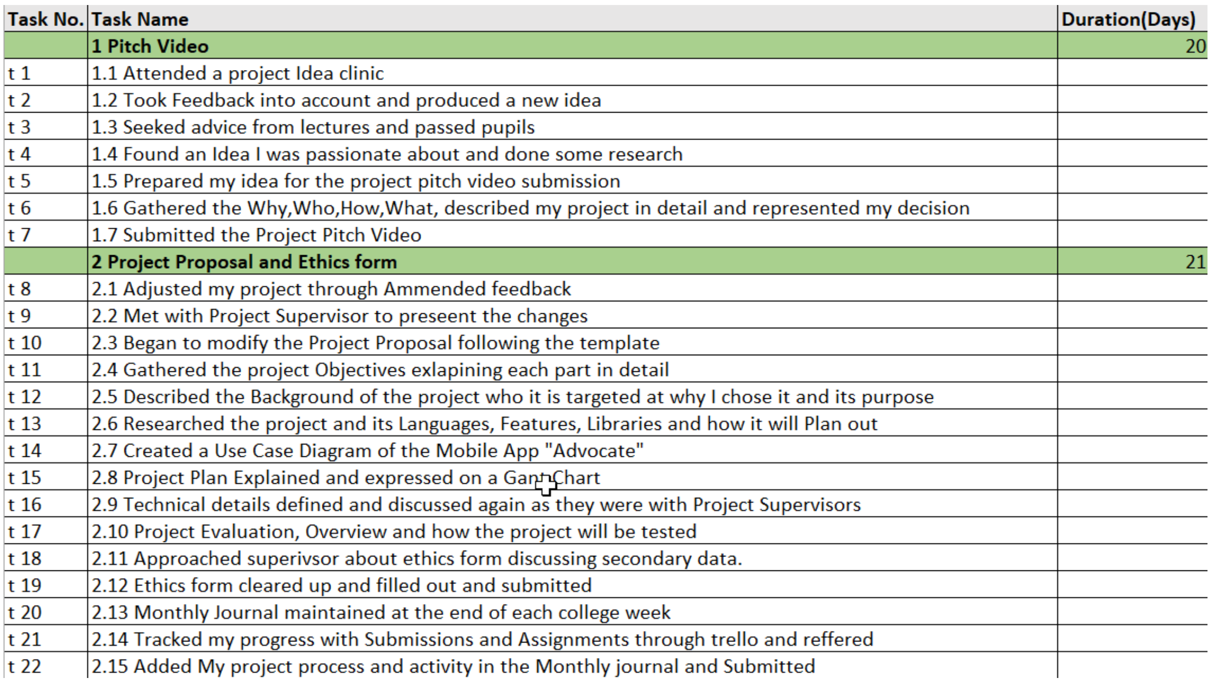
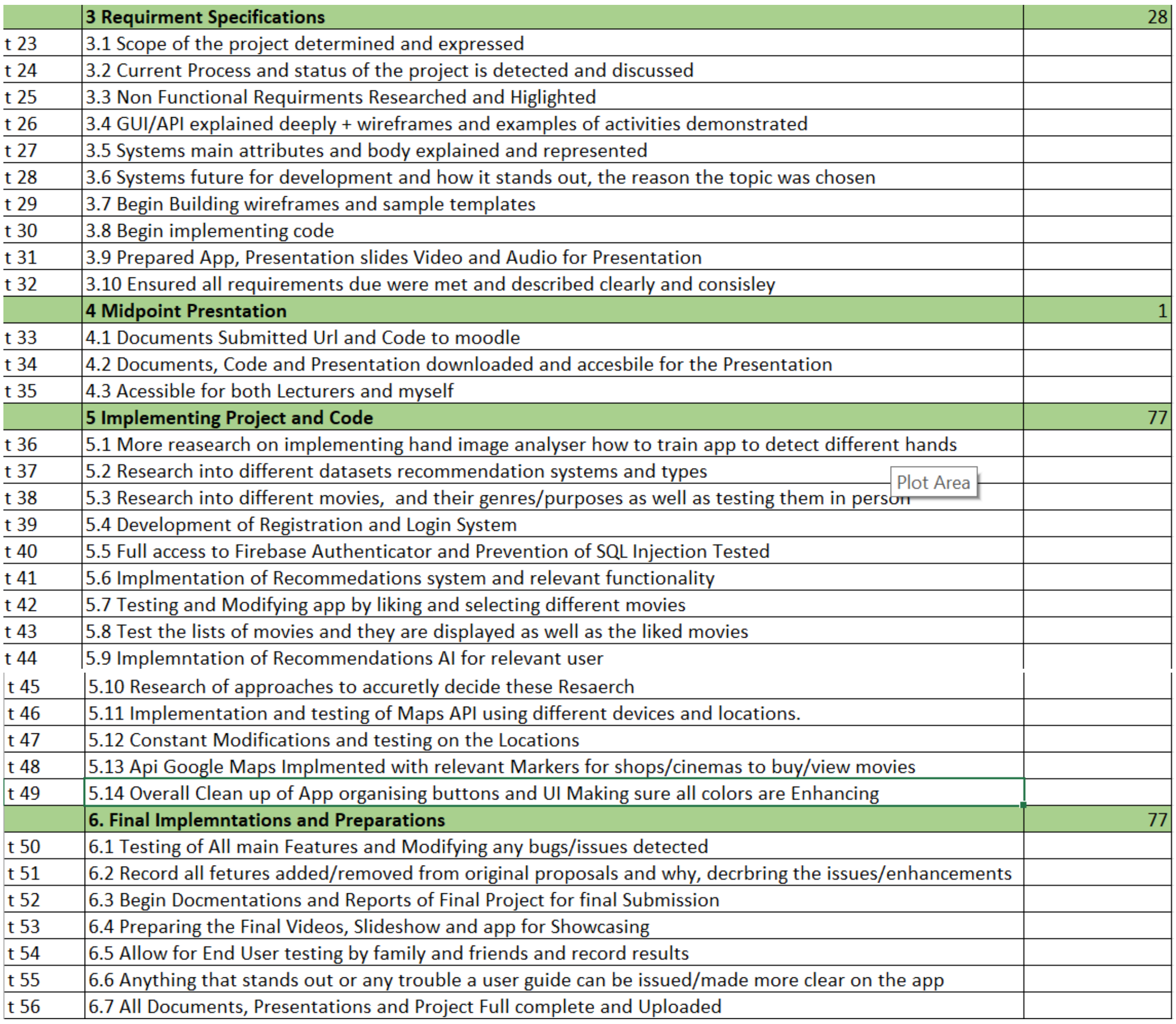
The Special resources required for this project will involve TensorFlow Light and Machine learning. The project will contain large amounts of data which will be used to determine an overall decision of Providing suitable movies to a user account. I will also need an API key from google to access their API services and libraries perhaps a key will also be need for YouTube movies integration.

Other external resources such as firebase will be used to store and retrieve data on this application. This is a google database that is compatible with android studio and many other applications. The recommendations data can be represented on a bar chat to show how familiar each picture that Is taken or selected are to each other on google cloud platform.

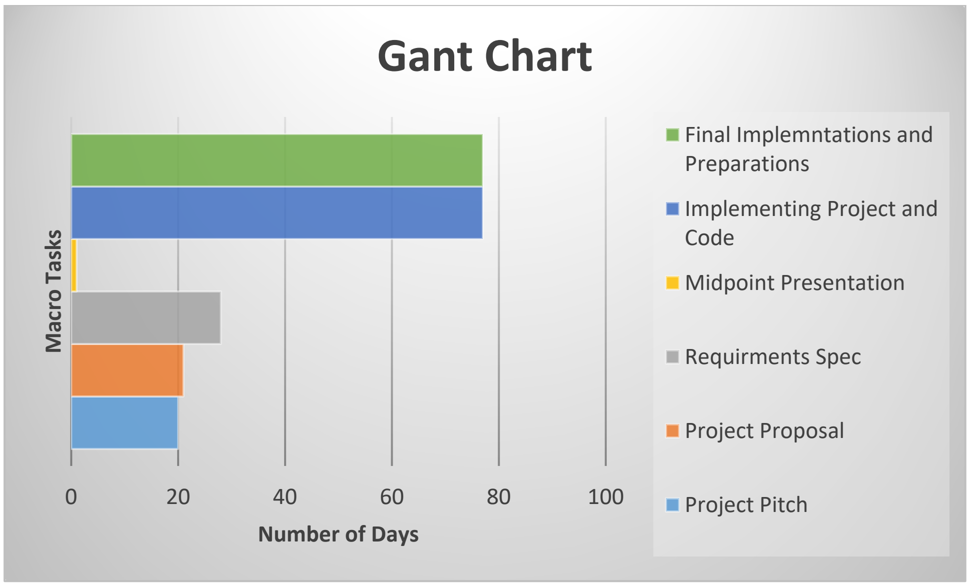
Due to the high number of activities and special features I will constantly need to test this application both on the android emulator and android devices. An Api Key will also need to be retrieved from my google analytics account and the right libraries and privacy setting to be enabled on each device the app is being used on.

7 Project Plan

Plot and Plan of the process my software project will take.

Task No.	Task Name	Duration(Days)
	1 Pitch Video	20
t 1	1.1 Attended a project Idea clinic	
t 2	1.2 Took Feedback into account and produced a new idea	
t 3	1.3 Seeked advice from lectures and passed pupils	
t 4	1.4 Found an Idea I was passionate about and done some research	
t 5	1.5 Prepared my idea for the project pitch video submission	
t 6	1.6 Gathered the Why,Who,How,What, described my project in detail and represented my decision	
t 7	1.7 Submitted the Project Pitch Video	
	2 Project Proposal and Ethics form	21
t 8	2.1 Adjusted my project through Ammended feedback	
t 9	2.2 Met with Project Supervisor to preseent the changes	
t 10	2.3 Began to modify the Project Proposal following the template	
t 11	2.4 Gathered the project Objectives exlapining each part in detail	
t 12	2.5 Described the Background of the project who it is targeted at why I chose it and its purpose	
t 13	2.6 Researched the project and its Languages, Features, Libraries and how it will Plan out	
t 14	2.7 Created a Use Case Diagram of the Mobile App "Advocate"	
t 15	2.8 Project Plan Explained and expressed on a Gantt Chart	
t 16	2.9 Technical details defined and discussed again as they were with Project Supervisors	
t 17	2.10 Project Evaluation, Overview and how the project will be tested	
t 18	2.11 Approached superivsor about ethics form discussing secondary data.	
t 19	2.12 Ethics form cleared up and filled out and submitted	
t 20	2.13 Monthly Journal maintained at the end of each college week	
t 21	2.14 Tracked my progress with Submissions and Assignments through trello and reffered	
t 22	2.15 Added My project process and activity in the Monthly journal and Submitted	
	3 Requirment Specifications	28
t 23	3.1 Scope of the project determined and expressed	
t 24	3.2 Current Process and status of the project is detected and discussed	
t 25	3.3 Non Functional Requirments Researched and Higlighted	
t 26	3.4 GUI/API explained deeply + wireframes and examples of activities demonstrated	
t 27	3.5 Systems main attributes and body explained and represented	
t 28	3.6 Systems future for development and how it stands out, the reason the topic was chosen	
t 29	3.7 Begin Building wireframes and sample templates	
t 30	3.8 Begin implementing code	
t 31	3.9 Prepared App, Presentation slides Video and Audio for Presentation	
t 32	3.10 Ensured all requirements due were met and described clearly and consisley	
	4 Midpoint Presntation	1
t 33	4.1 Documents Submitted Url and Code to moodle	
t 34	4.2 Documents, Code and Presentation downloaded and accesbile for the Presentation	
t 35	4.3 Acessible for both Lecturers and myself	
	5 Implementing Project and Code	77
t 36	5.1 More reasearch on implementing hand image analyser how to train app to detect different hands	
t 37	5.2 Research into different datasets recommendation systems and types	
t 38	5.3 Research into different movies, and their genres/purposes as well as testing them in person	
t 39	5.4 Development of Registration and Login System	
t 40	5.5 Full access to Firebase Authenticator and Prevention of SQL Injection Tested	
t 41	5.6 Implmentation of Recommedations system and relevant functionality	
t 42	5.7 Testing and Modifying app by liking and selecting different movies	
t 43	5.8 Test the lists of movies and they are displayed as well as the liked movies	
t 44	5.9 Implemntation of Recommendations AI for relevant user	
t 45	5.10 Research of approaches to accuretly decide these Resaerch	
t 46	5.11 Implementation and testing of Maps API using different devices and locations.	
t 47	5.12 Constant Modifications and testing on the Locations	
t 48	5.13 Api Google Maps Implmented with relevant Markers for shops/cinemas to buy/view movies	
t 49	5.14 Overall Clean up of App organising buttons and UI Making sure all colors are Enhancing	
	6. Final Implemntations and Preparations	77
t 50	6.1 Testing of All main Features and Modifying any bugs/issues detected	
t 51	6.2 Record all fetures added/removed from original proposals and why, decrbring the issues/enhancements	
t 52	6.3 Begin Docmentations and Reports of Final Project for final Submission	
t 53	6.4 Preparing the Final Videos, Slideshow and app for Showcasing	
t 54	6.5 Allow for End User testing by family and friends and record results	
t 55	6.6 Anything that stands out or any trouble a user guide can be issued/made more clear on the app	
t 56	6.7 All Documents, Presentations and Project Full complete and Uploaded	



These tasks are broken down into micro tasks from each Submission/Deadline I have due. I have explained briefly what each task will consist of and what is involved in each one. There is also a bar chart that represents the number of days each macro task will take in total. Tasks may be added/removed as the deadline becomes closer and if any amendments are made by my Project Supervisor.

8 Testing

J-Unit Testing

Perform J Unit test using the JAR and Kotilin JDK through NetBeans for complex activities. This tests for the quality of methods and gives the app a good sense of automated testing rather than manual testing. It's a great approach to testing and ensures app quality. I used NetBeans built in J Unit testing terminal.

Users and Database

- Ensure users are saved and stored on the database
- Ensure removed users cannot access the App and non-registered users also.
- Ensure the database only saves relevant and good data.
- Ensure database reacts within 4 seconds upon creating and removing users
- Ensure the database saves all the data and receives and displays data as expected
- Ensure activities take no longer than 5000 milliseconds to load.

Manual and Unit Testing

- Ensure login and Registration works as expected, The User can successfully register and login to the system by typing in the correct credentials.
- Once a user is logged in, they should not have to constantly login unless logged out.
- Typing malicious code into the textboxes of registration and login to ensure SQL Injection cannot be performed.
- All tests for Correct and incorrect passwords will be accrued out manually and should contain at least 8 characters and a special character.

- Test on phone and other devices such as tablet to ensure its responsive for all devices.
- I will also make sure each ethics form is completed fully and accurately after receiving my supervisor's approval.

Performance Testing

- Testing the Application logging on when the users Wi-Fi is down or bad.
- Ensuring the users login or loading time is no longer than 4 seconds to successfully load.
- Make sure logging out takes no longer than 4 seconds to load.
- Running the application successfully on Emulator, Tablet and Mobile device.
- Ensuring all buttons and activities take no longer than 4 seconds to load.

References

Berry, S., 2011. *asistdl.onlinelibrary.wiley.com.* [Online]
Available at: https://asistdl.onlinelibrary.wiley.com/doi/full/10.1002/meet.14504701402
[Accessed November 2021].

Schmydt, M., 2020. *link.springer.com.* [Online]
Available at: https://link.springer.com/chapter/10.1007/978-3-030-61146-0_15
[Accessed November 2021].

2.1. Ethics Approval Application (only if required)

National College of Ireland

DECLARATION OF ETHICS CONSIDERATION

School of Computing

Student

Name: …Adam Condon……………………….

Student ID: …x18392911…………………………….

Programme …Software Development…………… **Year:** …2021………….

Module: …Software Project……………………..

Project Title: …FindMyGlove…………………………………………..

Please circle (or highlight) as appropriate

This project involves human participants	[✓ Yes] / No

Introduction

The following decision table will assist you in deciding if you must complete the Declaration of Ethics Consideration Form or/and the Ethics Application Form.

Public Data	[✓ Y]	Y	Y	Y	N	N	N	N
Private Data	[✓ Y]	Y	N	N	Y	Y	N	N
Human Participants	[✓ Y]	N	Y	N	Y	N	Y	N
Declaration of Ethics Consideration Form	x	X	x	X	X	X	x	
Ethics Application Form	X		X		X		X	

Please circle (or highlight) as appropriate

The project makes use of secondary dataset(s) created by the researcher	Yes / [✓ No]
The project makes use of public secondary dataset(s)	Yes [✓ / No]
The project makes use of non-public secondary dataset(s) Approval letter from non-public secondary dataset(s) owner received	Yes / [✓ No] Yes / [✓ No]

Sources of Data:

Evidence for use of secondary dataset(s)

Include dataset(s) owner letter/email or cite the source for usage permission

Use of TMDB Dataset is permitted and referenced within reports. Users data is recorded to google firebase. Users location is saved only if it is permissible to the user (If their Device Accepts the prompt) Users testing data used for testing the app after consent was given

CHECKLIST

Non-public/private secondary dataset(s) -Owner letter/email is attached to this form ***OR*** Citation and link to the web site where permission is granted – provided in this form	Yes / [✓ No] Yes / [✓ No]

ETHICS CLEARANCE GUIDELINES WHEN HUMAN PARTICIPANTS ARE INVOLVED

The Ethics Application Form must be submitted on Moodle for approval prior to conducting the work.

<u>Considerations in data collection</u>

- Participants will not be identified, directly or through identifiers linked to the subjects in any reports produced by the study
- Responses will not place the participants at risk of professional liability or be damaging to the participants' financial standing, employability or reputation
- No confidential data will be used for personal advantage or that of a third party

<u>Informed consent</u>

- Consent to participate in the study has been given freely by the participants
- participants have the capacity to understand the project goals.

- Participants have been given information sheets that are understandable
- Likely benefits of the project itself have been explained to potential participants
- Risks and benefits of the project have been explained to potential participants
- Participants have been assured they will not suffer physical stress or discomfort or psychological or mental stress
- The participant has been assured s/he may withdraw at any time from the study without loss of benefit or penalty
- Special care has been taken where participants are unable to consent for themselves (e.g children under the age of 18, elders with age 85+, people with intellectual or learning disability, individuals or groups receiving help through the voluntary sector, those in a subordinate position to the researcher, groups who do not understand the consent and research process)
- Participants have been informed of potential conflict of interest issues
- The onus is on the researcher to inform participants if deception methods have to be used in a line of research

I have read, understood, and will adhere to the ethical principles described above in the conduct of the project work.

Signature: ……Adam Condon………………………………………………………………

Date: ……02/11/2021………………………………………………………………………………

Appendix I

13.1 Reflective Journals

14.0 November Reflection

Student Name	Adam Condon
Student Number	X18392911
Course	BHS in Computing (Software Development)

15.0

16.0 Month:

What? This month was a huge breaking point for my project deciding a project proposal. The project requirements, difficulty and realistic aspects were all factors I had to consider. Another idea was produced by myself where I found myself once again having to change my idea. I resorted to choosing one of the faculty recommended projects from Moodle with was a mobile app recommendation system. I spoke to my supervisor about this proposal, we amended some changes to the idea so I could then finalise my idea and begin working on it. I attended various workshops to improve my report writing, advice with ethics and good quality data and completed a lot more research into my project idea.
So What? My success consisted of uploading the newest project proposal and finally getting to finalise my idea with my relevant supervisor and begin to work towards the midpoint presentation and implementation of my project. This was a relief to get another submission under my belt in this challenging year, but I had to amend my idea before the midpoint presentation due to my supervisor not being able to keep in touch due to high working demand.
Now What? Planning and prioritising my time are key for the next few months implementing this project. It will take a lot of hard work and research, but the end goal is in sight. I will meet to my Gant chart I have set out to this project stay on schedule and try to complete each task. Every week I will build on my project and tweak it together to make the final project I seek. I will also be meeting with my academic supervisor once every two weeks to keep in touch and seek guidance on this project. Between that time, I will continue to work in the manner I am doing and ask any relevant questions to the supervisor.

Student Signature	Adam Condon

December Reflection

Student Name	Adam Condon
Student Number	X18392911
Course	BHS in Computing (Software Development)

Month: December

What? This month was packed full of project Submissions the main one being the midpoint submission which was a halfway point submission of our project process I was very happy of the progress my project was making as I implemented three main parts of Advocate and was on track to receiving the grade I wanted.
So What? I will receive my result in January where I will chop and change things from my app and report according to my feedback and my supervisor and from here, I will continue to work towards the next goal which is the final submission in May
Now What? I will continue to complete my last semester of college attempting to maintain my 2:1 overall average and get the best grades I can possible.

Student Signature	Adam Condon

January Reflection

Student Name	Adam Condon
Student Number	X18392911
Course	BHS in Computing (Software Development)

Month: January

What? This month mainly consisted of projects and exam I had to complete such as the “Midpoint Submission” for Software project where I submitted a current stage of code from my android app “Advocate”. I also submitted a current stage of my project documentation and presented a demo of my app via Microsoft teams to lecturers for marking. I will receive the feedback of this submission in February and will began to fix whatever parts I need to accordingly and progress as much as I can.
So What? My success consisted of uploading all of my projects from semester 1 and maintaining a 2:1 average overall and hopefully this can be continued until I finish college. I will receive ten days off between semester 1 and 2 which will give me time to recover and reflect on the last semester and how I can improve going into semester 2.
Now What? A large priority of my time will be spent by adding to my project’s documentation and implementation. In semester 2 having only 4 modules now rather then 6 which is a big

relief considering the weighting of the software project in comparison to two modules so this will be a really big chunk of my final grade and will require a lot of work.	
Student Signature	Adam Condon

February Reflection

Student Name	Adam Condon
Student Number	X18392911
Course	BHS in Computing (Software Development)

Month: February

What? This month was spent clearing up mistakes mainly based on my project midpoint feedback. A lot of the corrections needed were for the report. Such as Use case diagrams and certain sections highlighted by my project supervisor during are fortnightly meetings. I made these changes accordingly throughout the month keeping other CAs in mind.	
So What? My success consisted of uploading new monthly journals and continue to expand in my final year project work and meeting with my project supervisor to correct and guide me with the work I produced. Setting micro and macro tasks has really helped me progress in software project too allow me to have goals for the next meeting myself and project have so that we can criticise the work if needed and improve it in preparation to the final submission.	
Now What? March will be a busy month with other modules consisting strict deadlines so some time may need to prioritise to these rather the final year project. Otherwise, I have planned to implement a google maps API into my application which gather local cinemas to a user's location and successfully deploy this.	
Student Signature	Adam Condon

March Reflection

Student Name	Adam Condon

Student Number	X18392911
Course	BHS in Computing (Software Developement)

Month: March

What? This month consisted of me adding more advanced implementation to the project and enabling me to test more features of my app as I progressed. Once again, I met with my supervisor on a fortnight basis, and we discussed possible improvements to the code and documentation. The main changes were arranged to be fixed for the next meeting starting on the 4th of April where I expect to have majority of the documentation complete.
So What? This month consisted of me progressing more with my software project in both documentation and implementation constantly thinking of the feedback I received from my midnight submission and aiming higher for the final submission where ill need to complete my implementation in due time before the submission where I can critically test the application.
Now What? Next month I aim to complete my applications implementation side, where I will the n produce automated tests and record the results and document all the results as well as add or clean up any remaining aspects of the document. This will be done once all the other remaining apps are submitted successfully.

Student Signature	Adam Condon

April Reflection

Student Name	Adam Condon
Student Number	X18392911
Course	BHS in Computing (Software Development)

Month: April

What? This month consisted of me adding more advanced implementation to the project and enabling me to test even more features to advocate. Also meeting with my supervisor more regularly to ensure everything was going to plan and was to a high standard.
So What? This month consisted of me progressing more with my software project in both documentation and implementation constantly thinking of the feedback I received from my supervisor and adjusting changes to my app it was time to also start cleaning up my report as well as my GitHub repo which will be the repository, I will use to deploy my app.
Now What? It is now time to submit my project

Student Signature	Adam Condon

May Reflection

Student Name	Adam Condon
Student Number	X18392911
Course	BHS in Computing (Software Development)

Month: May

What? This month was last month of college where I proofread, tested, and cleaned up my code and report, I also made my project poster, booking appointment availably in preparation for my project showcase where I will be pitching my project idea to other companies. I have a graduate role lined up for after college, but it is no harm demonstrating my project
So What? The main purpose of this month is ensuring my presentation and app are fully functioning and ready to demo to potential investors or managers. This means I follow my strict Gant chart and Trello cards to fulfil each macro and micro task weekly.
Now What? It is now time to submit my project and present my innovation to my lecturers and companies.

Student Signature	Adam Condon

16.1 Invention Disclosure Form (Remove if not completed)

Please fill in the following sections if you think your idea is innovative:

1. Title of Invention

ADVOCATE

2. Inventors

Name	School/Research Institute	Affiliation with Institute (i.e. department, student, staff, visitor)	Address, contact phone no., e-mail	% Contribution to the Invention
ADAM CONDON	NCI	STUDENT	W91NC89, 0851318226, X18392911@student.ncirl.ie	100%

3. Contribution to the Invention

Each contributor/potential inventor should write a paragraph relating to his/her contribution and include a signature and date at the end of the paragraph.

100% contribution to Adam Condon on the app Advocate.

Adam Condon

4. Description of Invention

Advocate is a user-friendly movie recommendation system that allows users who are unsure or indecisive of what movie they should watch to gain access to recommended movies based on hundreds of options. The user can like any movies at their own leisure and depending on what they like/watch they will be recommended similar movies. Advocate will also allow the user to find out where the nearest cinema/shop to buy movies of their choice are with its integrated Google API system and their eventual payment system.

5. Why is this invention more advantageous than present technology?

The problem Advocate solves is accessing recommendations for movies depending on what you like all for free with the click of a button. Advocate has a huge selection of movies ranging from newer to old movies which some online movie subscriptions do not have with a large variety of great films. You can access advocate for free with to without Wi-Fi connection and still receive the same results.

Advocate is great for Dates, family nights or even individual sittings if the user is unsure of what to watch. Adding the maps feature where the user also can see where they can view the movies locally without having to travel far eases the stress of organizing and planning and allow the user to access the movies in external sources within a few seconds. Advocate is quick, easy, and fun. If you are a movie lover like me it is a must have.

6. What is the current stage of development / testing of the invention?

The current stage is 50% I have begun developing the application and am half through the process the main algorithm's/aspects of the app should be fully developed by May 2022 with the highest working rates around February/March. I have also began testing the app manually and through different devices as explained for responsiveness but due to the fact the app and core parts are not fully complete more testing will be done in the later stages of development.

7. List the names of companies which you think would be interested in using, developing, or marketing this invention

Netflix, Disney Plus, Amazon and Sony.

8. Funding Partner(s)

Government Agency & Department	n/a
% Support	n/a
Contract/Grant No.	n/a
Contact Name	n/a
Phone No.	n/a
Address	n/a

Industry or other Sponsor	n/a
% Support	n/a
Contract/Grant No.	n/a
Contact Name	n/a
Phone No.	n/a
Address	n/a

9. Where was the research carried out?

All my research and development were carried out in my house remotely with restricted access to college/working environments due to a global pandemic.

10. What is the potential commercial application of this invention?

The Movie industry

11. Was there transfer of any materials/information to or from other institutions regarding this invention?

If so please give details and provide signed agreements where relevant.

N/a

12. Have any third parties any rights to this invention?

If yes, give names and addresses and a brief explanation of involvement.

N/A

13. Are there any existing or planned disclosures regarding this invention?

Please give details.

N/A

14. Has any patent application been made? Yes/No

If yes, give date: ________________ Application No.: ________________

Name of patent agent: ____

Please supply copy of specification.

15. Is a model or prototype available? Has the invention been demonstrated practically?

Yes a prototype of the project is available and presentable to be installed on any android device above android APK 5.

I/we acknowledge that I/we have read, understood, and agree with this form and the Institute's *Intellectual Property and Procedures* and that all the information provided in this disclosure is complete and correct.

I/we shall take all reasonable precautions to protect the integrity and confidentiality of the IP in question.

Inventor: Adam Condon

Signature Adam Condon Date: 12/12/2022

1.1. Other materials used

N/A