Senior Design Project

Project short-name: Outletter

Analysis Report

Muhammad Bilal Bin Khalid, Muhammad Saboor, Mian Usman Naeem Kakakhel, Daniyal Khalil, Muhammad Arham Khan

Supervisor: Hamdi Dibeklioğlu
Jury Members: Çigdem Gündüz-Demir and Can Alkan

Progress Report
Nov 21, 2020

Contents

1 Introduction

In today's capitalistic age, shopping is one of the most time-consuming activities that people from all walks of life participate in on a regular basis. From shopping for some clothing articles, to a new pair of shoes to even buying the latest electronic gadget, people spend hours trying to validate their shopping decisions and confirming that they're getting the best possible bargain. But considering the inherent information gap in the manual store-by-store checking process [1], we spend many hours only to buy what we got the best deal on at the three stores we visited.

So seeing the potential for improvement in the process, we decided to introduce Outletter, A one-stop shopping companion mobile application that allows users to point their mobile phones towards items like fashion articles, electronics etc. Our app would automatically recognize the product in realtime and render metadata like price comparison with nearby stores, product reviews and other available variants in nearby stores, right in the users field of view using Augmented Reality. This way, whenever a user is out for shopping, they can rely on Outletter to help them make the most reliable decision about a product and be sure to get the best deal in a very easy and fast way. Considering that Outletter will be able to identify almost any product that is available online and that there aren't any steps with manual data entry or button clicks (the user can access all information just by pointing the mobile phone at a product), the application would revolutionize the shopping process for the mass user-base including the elderly.

Apart from this, Outletter will also provide the user with the latest deals and offers going on in a particular store as soon as they are near that store. This way, users will never regret missing out a good deal from a store just because they didn't watch an ad campaign and, as promised by our app, will always be making the most well-informed shopping decision possible.

2 Current System

There are various mobile applications present on the market that help their users to find a product online by image. There are other applications that provide you with reverse

image search results from Google or other search engines. Some example of such systems are:

1. Google Lens
2. Pictpicks
3. Search by Image

These applications find different attributes in the image and then find similar images present on the Internet. However, Outletter is different from such applications because it also provides useful insight to the user about the product he is searching. This includes price comparison of the product on different stores and the public opinion about the products.

3   Proposed System

3.1 Overview

Outletter will be a hybrid application developed using React-Native [2] to target the Android and iOS markets (totalling to cover almost the entire market share). The application aims to make the shopping process more well-informed, faster and easier for users from all walks of life and hence, the expected features will be as follows:

1. **Real-time product recognition**: The application will utilize the device camera's video input to identify the item frame instantaneously and then sends only the item frame to the cloud-based server for the item to be identified and the relevant data to be retrieved. The cloud-based application first tries to match the item against currently indexed products in our database (through some brand affiliations), the product elements on the internet using techniques like image comparison and text search from image. As soon as the item is found, returns all relevant data including different prices, product reviews, available variants in current and nearby stores etc. in real time. This process takes under 3 seconds and hence, provides users the liberty to view information about any product in a very intuitive and straightforward way.

2. **Product Information in field-of-view**: The application will render relevant information retrieved from the backend service in the device's field of view using ARCore. So whenever the user points their phones at an object, they will be able to see anchored data like best price, product reviews and other metadata about the object anchored to it in the user's field of view.

3. **Location map for product options**: There will be a map feature for the user to see locations of other stores where the object they're currently pointing the phone at is available and their relevant prices.

4. **Product recommendations**: Display the user with relevant options to shop alongside the currently selected product. So, the app utilizes previous shopping patterns of users to identify relevant products that can be bought alongside the current product.

5. **Product Feedback**: Users will be able to like, dislike and leave reviews on products that users in the future will be able to have a look at when they scan the relevant product.

6. **Share products**: Users will be able to share the current product's information using a custom web link using the sharing app of their choice. This includes sending the link over WhatsApp, Messengers or other social media applications.

7. **Geo-tagged deals for nearby stores**: Whenever the user enters or is near a store, the user will see a popup in their application containing the latest deals and promotions going on at that store. This would require stores that have a brand affiliation with our application and have provided us with their latest deals, or have made a deal or campaign made available over their websites for our backend engine to index. Hence, during shopping, users will never miss a great deal on articles and will be able to make more informed decisions.

3.2 Functional Requirements

3.2.1 System Requirements

The system should:

- Ask permissions to access local storage, camera, microphone and internet connection of the mobile device.
- Display a bounding box around the focused object in the camera view.
- Provide an option to capture the image in the camera view.
- Provide an option to record the product description verbally.
- Send the picture or the verbal description to the backend server for processing.
- Be able to receive the products which are present in the market and the recommended items for the user from the backend server.
- Display the top price, URL, and reviews of the selected object in the captured image on top of the current camera view.
- Provide an option to view top products in different shops with different prices.
- Display the top products in different shops with different prices as a list.
- Provide an option to view recommended items.
- Provide an option to view the location and directions of the store the current product is available at.
- Display the recommended items as a list.
- Provide an option to like/dislike an item in the camera view.
- Provide an option to share an item in the camera view to the social media.
- Provide an option to save an offer to the wish list to view the product later.
- Provide an option to open the wish list.
- Provide an option to delete an item from the wish list.
- Provide an option to open the URL of an item in the wish list.
- Provide an option to share an item in the wish list to social media.
- Provide an option to leave a review of an item in the wish list.
- Provide an option to view the reviews of an item in the wish list.
- Provide an option to like/dislike an item in the wish list.
- Provide an option to view geo-tagged deals in the camera view.

- Display the selected geo-tagged deal.

3.2.2 User Requirements

The user can:

- Allow or deny permissions to access local storage, camera, microphone and internet connection of the mobile device.
- Select an object to focus in the camera view.
- Capture the image in the camera view.
- Describe the product verbally.
- View the top price, URL, and reviews of the selected object in the captured image on top of the current camera view.
- Select the option to view top products in different shops with different prices.
- View the top products in different shops with different prices as a list.
- View the location and directions of the store where the current product is available.
- Select the option to view recommended items.
- View the recommended items as a list.
- Like/dislike an item in the camera view.
- Share an item in the camera view to social media.
- Save an offer to the wish list to view the product later.
- Open the wish list.
- Delete an item from the wish list.
- Open the URL of an item in the wish list.
- Leave a review of an item in the wish list.
- View the reviews of an item in the wish list.
- Like/dislike an item in the wish list.
- Share an item in the wish list to social media.
- Select the option to view geo-tagged deals in the camera view.
- View geo-tagged deals in the camera view.
- Allow or deny permissions to share his/her preferences for the purpose of improvement of the system.

3.3 Non-Functional Requirements

3.3.1 Availability

- **Service:** The server downtime should not be more than 1% on a monthly regulation cycle otherwise the server should be available for requests 24 hours a day, 7 days a week.
- **Location:** The Application will only be available to users in Turkey, due to the limited scope of websites that can be parsed.
- **Operating System:** The Application should be available in both Androids 7.0 or higher and iOS 11.0 or higher operating systems.

3.3.2 Response Time

The Application Response Time for user queries and searches should be less than 3 seconds. The Response Time for User Recommendations should also be less than 3 second.

3.3.3 Scalability

The Application usage is dependent upon the user demand. Hence the application should be scalable to meet with increasing user demands by increasing the number of users and database storage. For now, due to not having proper production servers, the number of concurrent users that would be able to use the application would not be more than 100.

3.3.4 Accuracy

The product results given by the application should have at least have an Accuracy of 90% and the recommendations should all be based on the personalized tastes of the user.

3.3.5 Extensibility

The application should be extendible to different areas and categories to meet the user demand if needed. Currently, as mentioned before, the application is available only in one location which is Turkey.

3.3.6 License

All the datasets used in training the models, and all the libraries used in the development should be properly licensed and accredited.

3.3.7 Privacy

The application should ask the user before using their data in improving the recommendation system to more suit their tastes.

3.3.8 Maintainability

The application should adopt a low coupling modular design to allow the modules to not affect each other during alterations.

3.3.9 Network & Server

The Application should run smoothly on 1mbps internet smoothly without any issues. The server should be able to handle 100 concurrent users concurrently. The server should be running on a minimum 2.2 GHz.

3.3.10 Usability

The Application should be user-friendly and have an intuitive design to help promote user-functionalities.

3.4 System Models

3.4.1 Scenarios

Scenario 1 – Sign In

Use Case: Sign In

Actor(s):   User

Entry Condition(s):   User launches the application and is currently signed out.

Exit Condition(s):   User is directed to the main page, if the login is successful.

Flow of Events:

1. The user enters their credentials and presses the Sign In button on the startup page of the application
2. If the Sign In is successful, the user is directed to the Home Page

Alternative Flow of Events:

A.   Sign In Unsuccessful:
          1.   The credentials entered by the user are wrong
          2.   User is asked to enter the credentials again.
B.   Internet connectivity:
          1.   Network connectivity issue during Sign In.
          2.   Redirected to the Sign In page.

Scenario 2 – Sign up as ShopOwner

Use Case: SignUp as ShopOwner

Actor(s):   ShopOwner

Entry Condition(s):   ShopOwner is at the Sign In and presses Sign Up as ShopOwner

Exit Condition(s):   ShopOwner Signs up successfully and is redirected to the Sign In page.

Flow of Events:

1. The shopOwner enters their information in the SignUp page and then presses Sign Up

2. The shopOwner signs up successfully and is redirected to the Sign In page.

Alternative Flow of Events:

A.   Sign Up Unsuccessful:

      1.   The email shopOwner entered already has an associated account.
      2.   ShopOwner is asked to enter a different email.

B.   Sign Up Unsuccessful:

      1.   Network connectivity issue during Sign Up
      2.   The shopOwner is redirected to the SignUp page.

Scenario 3 – Sign up as Customer

Use Case: SignUp as Customer

Actor(s):   Customer

Entry Condition(s): Customer is at the Sign In page and presses Sign Up as Customer

Exit Condition(s):   Customer Signs up successfully and is redirected to the Sign In page.

Flow of Events:

    1. The customer enters their information in the Sign Up page and then presses Sign Up as customer
    2. The customer signs up successfully and is redirected to the Sign In page.

Alternative Flow of Events:

A.   Sign Up Unsuccessful:

      1.   The email customer entered already has an associated account.
      2.   The customer is asked to enter a different email.

B.   Sign Up Unsuccessful:

      1.   Network connectivity issue during Sign Up
      2.   The customer is redirected to the Sign Up page.

Scenario 4 – Add new Ad

Use Case: Add new Ad

Actor(s):   ShopOwner

Entry Condition(s): ShopOwner is signed in, and is at the home page.

Exit Condition(s):   The advertisement is added to the store.

Flow of Events:

1. The ShopOwner is already signed in.
2. The ShopOwner presses Add new Ad
3. The ShopOwner Selects Image and Positions it.
4. The ShopOwner presses Save Ad.

Alternative Flow of Events:

A.   Goes back:
        1.   The ShopOwner goes back from the Add new Ad Option.
        2.   No new Ad is placed.
B.   The Image needs to be cropped:
        1.   The ShopOwner Selects Image, Crops it, and the positions it.
        2.   The ShopOwner presses Save Ad.
C.   The Save Ad is Unsuccessful.
        1.   Network connectivity issues during Save Ad.
        2.   The ShopOwner is redirected to the previous page.
D.   The Save Ad is Unsuccessful.
        1.   The Ad is being placed at a location far away from the location of the store.
        2.   The Ad is not saved, ShopOwner is asked to place it in the location of the store.

Scenario 5 – Crop Image

Use Case: Crop Image

Actor(s):   ShopOwner

Entry Condition(s): ShopOwner has selected the Image in Add new Ad

Exit Condition(s):   The image is cropped successfully.

Flow of Events:

1. The ShopOwner is Adding a new ad
2. The ShopOwner selects the image.
3. The ShopOwner crops the Image.

Scenario 6 – Position Image

Use Case: Position Image

Actor(s):   ShopOwner

Entry Condition(s):   ShopOwner has selected the Image in Add new Ad

Exit Condition(s):   The Image is positioned.

Flow of Events:

1. The ShopOwner is placing a new Ad
2. The ShopOwner selects an Image.
3. The ShopOwner positions the Image

Scenario 7 – View Ad

Use Case: View Ad

Actor(s):   ShopOwner

Entry Condition(s):   ShopOwner is signed in, and is at the home page.

Exit Condition(s):   The ShopOwner goes back to the homepage.

Flow of Events:

1. The ShopOwner is already signed in.
2. The ShopOwner chooses the ad he wants to view from the list.
3. The ShopOwner views the ad.
4. The ShopOwner goes back

Alternative Flow of Events:

A.   Remove:
1.   The ShopOwner views the Ad
2.   The ShopOwner removes the Ad.

B.   Edit:
1.   The ShopOwner views the Ad.
2.   The ShopOwner edits it.

Scenario 8 – Remove Ad

Use Case: Remove Ad

Actor(s):   ShopOwner

Entry Condition(s):   ShopOwner is viewing the Ad.

Exit Condition(s):   The Ad is removed

Flow of Events:

1. The ShopOwner is viewing the Ad.
2. The ShopOwner presses remove Ad.
3. The ShopOwner confirms it.
4. The Ad is removed.

Alternative Flow of Events:

A.   Cancels:

1.   The ShopOwner presses remove Ad
2.   The shopOwner does not confirm.
3.   The Ad is not removed.

Scenario 9 – Edit Ad

Use Case: Edit Ad

Actor(s):   ShopOwner

Entry Condition(s):   ShopOwner is viewing the Ad.

Exit Condition(s):   The ShopOwner goes back to viewing the Ad.

Flow of Events:

1. The ShopOwner is Viewing the Ad
2. The ShopOwner presses Edit.
3. The ShopOwners positions the image, or selects a new image.
4. The ShopOwner Confirms Edit.

5. The ShopOwner goes back to the homepage.

Alternative Flow of Events:

A.   Not Confirmed:

           1.   The ShopOwner positions the image, or selects a new image.
           2.   The ShopOwner does not confirm the edits.
           3.   The ShopOwner goes back to viewing the Ad.

Scenario 10 – Edit Store Details

Use Case: Edit Store Details

Actor(s):   ShopOwner

Entry Condition(s):   ShopOwner is signed in, and is at the home page.

Exit Condition(s):   The ShopOwner goes back to the homepage.

Flow of Events:

1. The ShopOwner is already signed in.
2. The ShopOwner presses edit shop details.
3. The ShopOwner edits the details and confirms it.
4. The ShopOwner goes back.

Alternative Flow of Events:

A.   Select Icon:

           1.   The ShopOwner edits the details.
           2.   The ShopOwner selects a new icon.
           3.   The ShopOwner confirms the edit.
           4.   The ShopOwner goes back.

B.   Goes back without editing:

           1.   The ShopOwner presses edit.
           2.   The ShopOwner goes back to the homepage.

Scenario 11 – Select Store Icon

Use Case: Select Store Icon

Actor(s):   ShopOwner

Entry Condition(s):   ShopOwner is editing the Ad.

Exit Condition(s):   The ShopOwner selects the new store icon.

Flow of Events:

      1. The ShopOwner is currently editing the Ad
      2. The ShopOwner selects a new store icon.

Scenario 12 – View Top Products

Use Case: View Top Products

Actor(s):   Customer

Entry Condition(s): The customer has scanned a clothing item and has received the results from the API.

Exit Condition(s):   The customer can press the back button to go to the main screen showing the camera view to scan a product.

Flow of Events:

      1. The customer is shown a horizontal list of top products based on the scan sorted according to their price and similarity.
      2. The customer scrolls the list right by swiping left to show more items in the list.

Alternative Flow of Event(s):

A.   Exit application:

      The customer can press back button twice or home button to exit the application

Scenario 13 – Scan Product

Use Case: Scan Product

Actor(s):   Customer

Entry Condition(s): A customer has successfully logged in as a customer and has pointed the camera of mobile towards a clothing item.

Exit Condition(s): The customer can exit the application by pressing the back button twice or home button.

Flow of Events:

1. The customer points the camera towards a clothing item.
2. The customer brings the required item in focus by tapping it.
3. With an internet connection, the customer is shown a list of top products sorted by price and similarity and another list of recommended items.

Alternative Flow of Event(s):

A.   No internet connection:

      If internet connection is lost, a message is shown to the customer telling him to reconnect to the internet.

B.   No product found:

      If the product is nowhere to be found in our database or internet, the customer is shown the message 'no product found`' in camera view.

Scenario 14 – View Reviews

Use Case: View Reviews

Actor(s):   Customer

Entry Condition(s): The customer is viewing a product and has clicked the 'view all' button under reviews section.

Exit Condition(s): The customer can press the back button to go back to the product details page.

Flow of Events:

1. The customer is shown a vertical list of reviews.
2. The customer can scroll down by swiping up to see more reviews.

Alternative Flow of Event(s):

A.   No internet connection:

If internet connection is lost, a message is shown to the customer telling him to reconnect to the internet.

B.   Exit application:

The customer can press the back button twice or home button to exit the application.

Scenario 15 – Like/Dislike Product

Use Case: Like/Dislike Product

Actor(s):   Customer

Entry Condition(s): The customer is viewing a product.

Exit Condition(s):   The customer presses the back button to go back to the page from where the customer has selected the product.

Flow of Events:

The customer presses the like button shown in the details of the product.

Alternative Flow of Event(s):

A.   No internet connection:

If internet connection is lost, a message is shown to the customer telling him to reconnect to the internet.

B.   Exit application:

The customer can press the back button twice or home button to exit the application.

C.   Product already liked:

If the product is already liked by the customer, the product is disliked.

Scenario 16 – Add/Remove from Wishlist

Use Case: Add/Remove from Wishlist

Actor(s):   Customer

Entry Condition(s): The customer is viewing a product.

Exit Condition(s):   The customer presses the back button to go back to the page from where the customer has selected the product.

Flow of Events:

> The customer presses the wishlist button shown in the details of the product.

Alternative Flow of Event(s):

A.   No internet connection:

> If internet connection is lost, a message is shown to the customer telling him to reconnect to the internet.

B.   Exit application:

> The customer can press the back button twice or home button to exit the application.

C.   Product already added to wishlist:

> If the product is already in the wishlist of the customer, the product is removed from the wishlist.

Scenario 17 – View Product Details

Use Case: View Product Details

Actor(s):   Customer

Entry Condition(s): The customer clicks on an item in the top products list, recommended products list, wishlist, or liked items.

Exit Condition(s):   The customer presses the back button to go back to the page from where the customer has selected the product.

Flow of Events:

The customer is shown the details of the product including the name, price, image, url, likes, and location of the corresponding physical store.

Alternative Flow of Event(s):

A.   No internet connection:

      If internet connection is lost, a message is shown to the customer telling him to reconnect to the internet.

B.   Exit application:

      The customer can press the back button twice or home button to exit the application.

C.   Review:

      The customer presses the review button.

D.   Location:

      The customer presses the location button.

E.   URL:

      The customer presses the URL button.

Scenario 18 – Post reviews

Use Case: Post reviews

Actor(s):   Customer

Entry Condition(s): The customer is viewing a product and has clicked the 'view all' button under reviews section.

Exit Condition(s):   The customer successfully posts a review.

Flow of Events:

1. The customer is shown a blank space to post the review followed by a vertical list of all reviews.
2. The customer can click the blank space to show the keyboard.
3. The customer types the review using the keyboard.
4. The customer clicks the 'Post' button to post the review.

Alternative Flow of Event(s):

A.   No internet connection:

        If internet connection is lost, a message is shown to the customer telling him to reconnect to the internet.

B.   Exit application:

        The customer can press the back button twice or home button to exit the application.

C.   Go back to product details page:

        The customer can press the back button to go back to the product details page.

Scenario 19 – View Recommended Products

Use Case: View Recommended Products

Actor(s):   Customer

Entry Condition(s): The customer has scanned a clothing item and has received the results from the API.

Exit Condition(s):   The customer can press the back button to go to the main screen showing the camera view to scan a product.

Flow of Events:

    1. The customer is shown a horizontal list of recommended products tailor made for the specific customer according to his previous searches and likes.
    2. The customer scrolls the list right by swiping left to show more items in the list.

Alternative Flow of Event(s):
A.   Exit application:

        The customer can press the back button twice or home button to exit the application.

Scenario 20 – Visit URL

Use Case: Visit URL

Actor(s):   Customer

Entry Condition(s): The customer is viewing a product.

Exit Condition(s):   The customer visits the URL in the default browser.

Flow of Events:

1. The customer clicks on the URL of the product.
2. The default browser opens up with the URL of the product.

Scenario 21 – Open Location

Use Case: Open Location

Actor(s):   Customer

Entry Condition(s): The customer is viewing a product.

Exit Condition(s):   The customer sees the location of the store in the default maps application.

Flow of Events:

1. The customer clicks on the view location button of the product.
2. The default maps application opens up with the store centered in the maps.

Scenario 22 – Share Product

Use Case: Share Product

Actor(s):   Customer

Entry Condition(s): The customer is viewing a product.

Exit Condition(s):   The customer shares the product on the desired platform.

Flow of Events:

1. The customer clicks on the share button of the product.
2. The customer is shown with several platforms available on his mobile on which he can share the product.
3. THe customer selects one of the platforms.
4. The customer shares the product on the desired platform using that platform's application.

Scenario 23 – View Wishlist

Use Case: View Wishlist

Actor(s):   Customer

Entry Condition(s): The customer has clicked the wishlist button from the menu.

Exit Condition(s):   The customer can press the back button to go to the page from where the customer opened the wishlist.

Flow of Events:

1. The customer is shown a vertical list of items that the customer has added to the wishlist.
2. The customer scrolls the list down by swiping up to show more items in the list.

Alternative Flow of Event(s):

A.   No internet connection:

If internet connection is lost, a message is shown to the customer telling him to reconnect to the internet.

B.   Exit application:

The customer can press the back button twice or home button to exit the application.

Scenario 24 – Open Ad

Use Case: Open Ad

Actor(s):   Customer

Entry Condition(s): A customer has successfully logged in as a customer and has pointed the camera of mobile towards an ad anchored in the shop.

Exit Condition(s):   The customer can press the back button twice or home button to exit the application.

Flow of Events:

1. The customer points the camera towards an anchored ad.
2. The customer brings the ad in focus by tapping it.

Alternative Flow of Event(s):

A.   No internet connection:

      If internet connection is lost, a message is shown to the customer telling him to reconnect to the internet.

B.   Exit application:

      The customer can press the back button twice or home button to exit the application.

3.4.2 Use Case Model
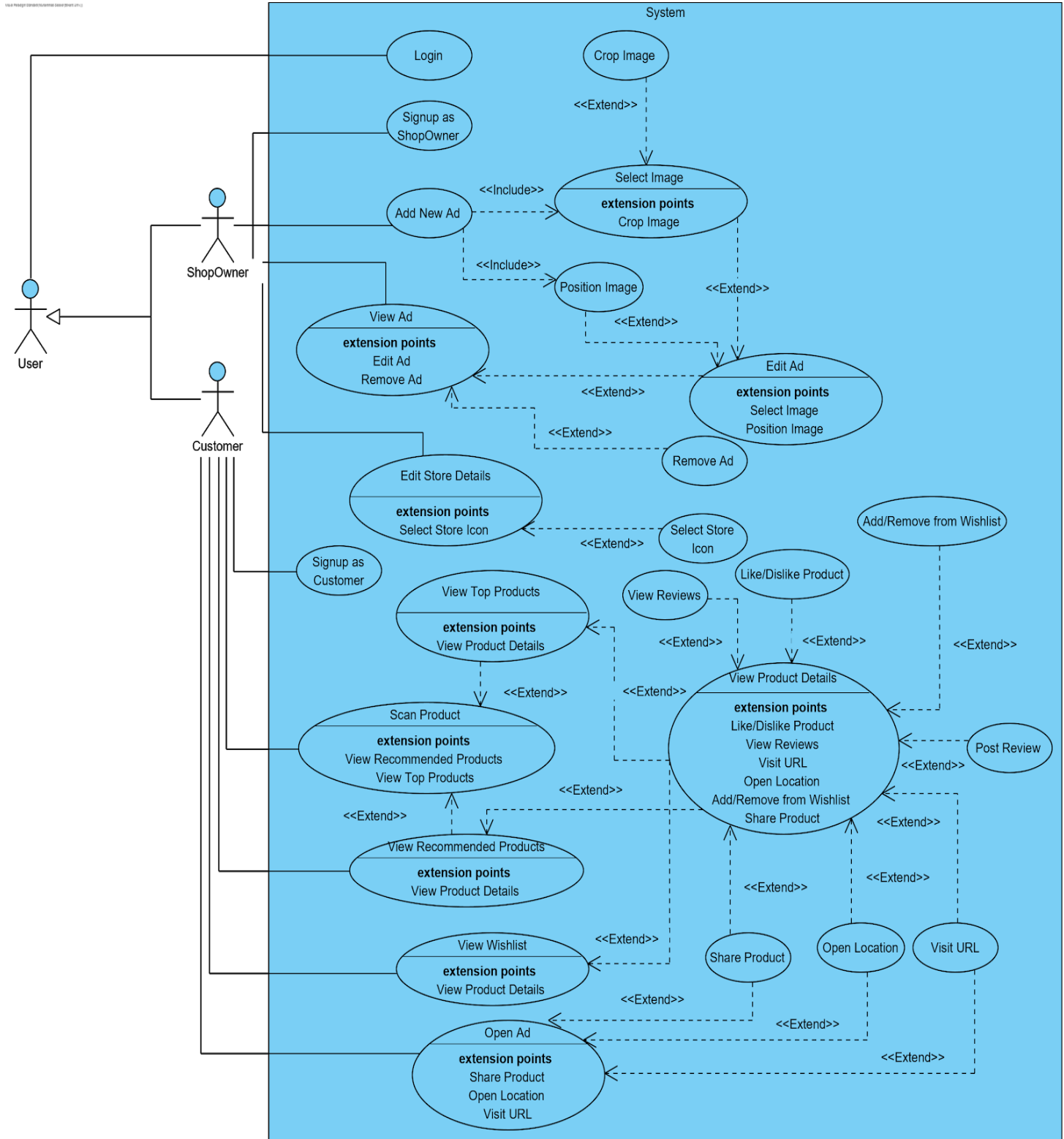
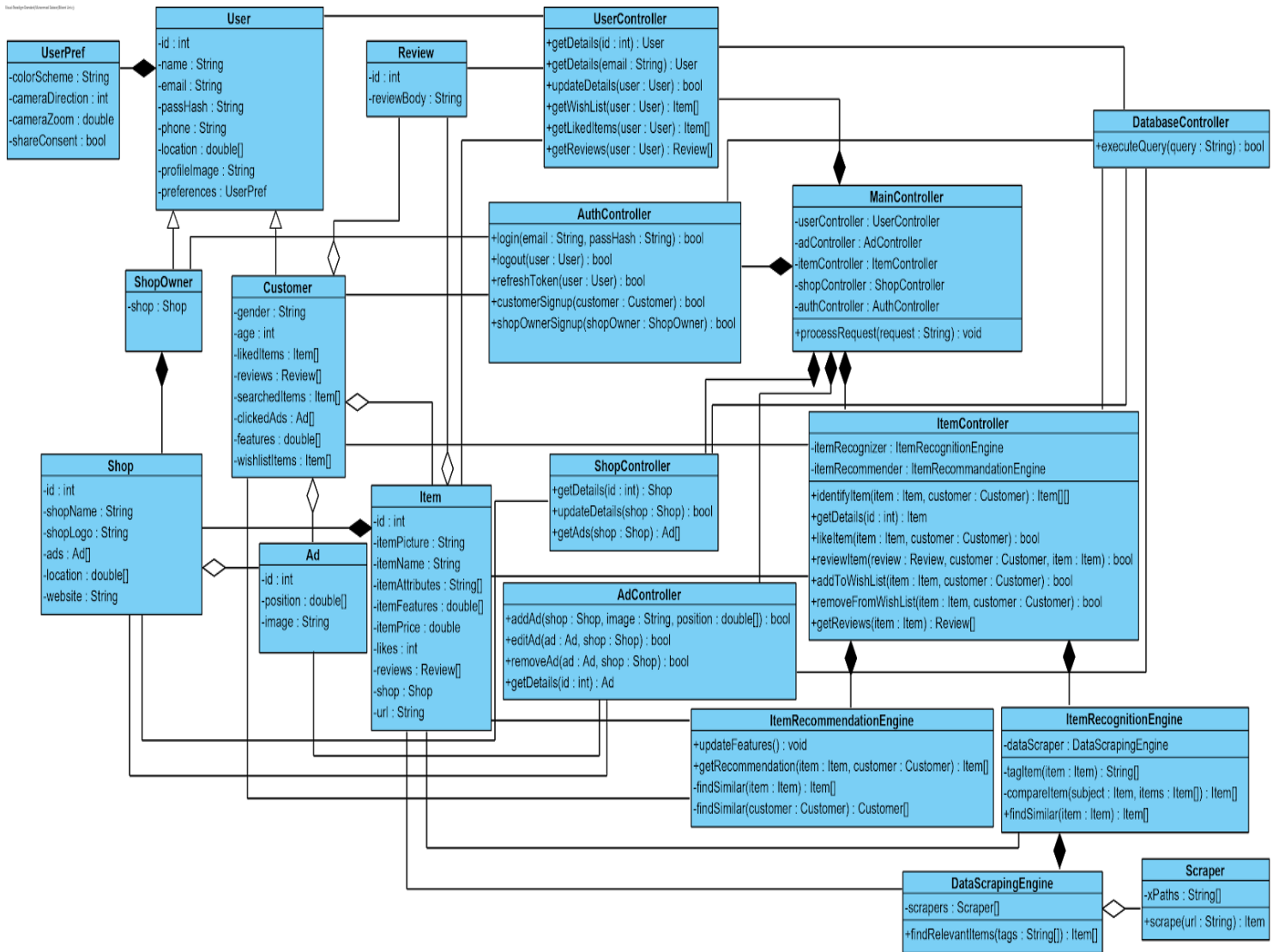Figure 1: Use Case Diagram

3.4.3 Object and Class Model

Figure 2: Class Diagram

3.4.4 Dynamic Model

3.4.4.1 Activity Diagram

The activity diagram shows the main flow of the application. The diagram shows how the overall application will work. After signing in depending on the type of user the application will follow a completely different set of activities.
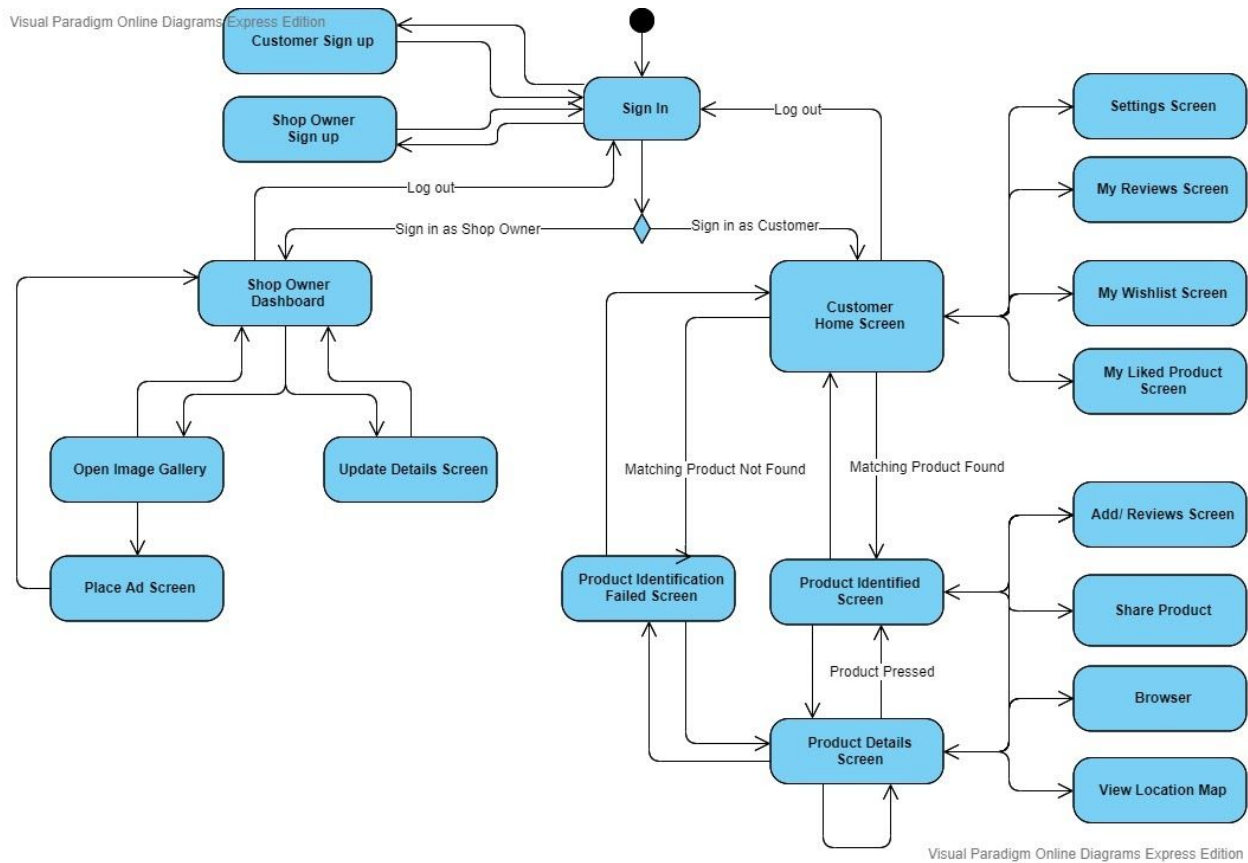

Figure 3: Main Flow Activity Diagram

3.4.4.2 State Diagram

There are three states in the mobile application. When the user logs in (ie: entry point), the app is in the scanning for product state. When a product is found, the app moves to the product found state where it displays the product's information and other relevant functions like recommended products etc. When a product is not found, the app moves to the product not found state where it only shows the top product hits. The user can go back to the scanning state from either of the two states. If a user logs out, the app ends.
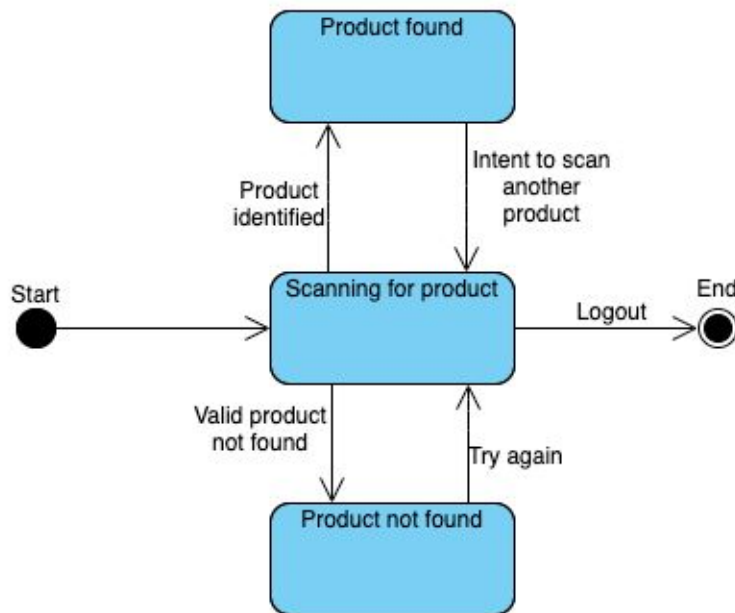

Figure 4: State diagram of the mobile application

The Web scraper microservice has two states where id starts in the idle state. Once a request to scrape the websites for certains data is received, it moves to the "scraping for data" state. Once the request has been served, it moves back to the idle state. Once the admin quits the scraper, it ends.
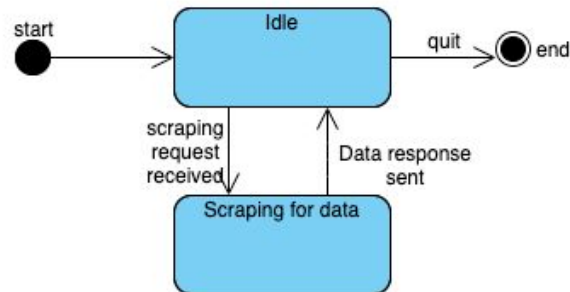

Figure 5: State diagram of web scrapers

The backend application will have two states. It will start from idle state. When it receives an identification request, it goes to the identifying product image state, where it tags the image and then finds similar images from the internet and local database. After it collects the result, it sorts them in descending order according to their similarity and sends all the results back. After sending all the results back, the application goes back to the idle state.
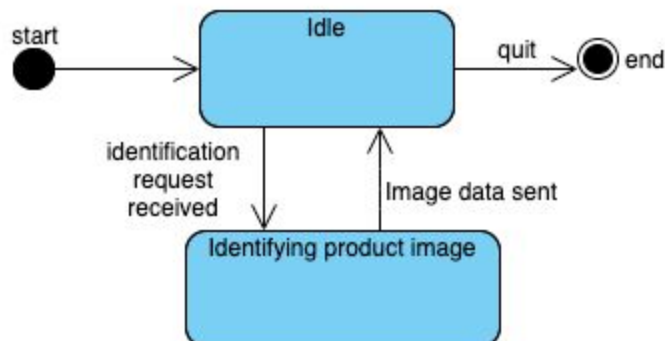

Figure 6: State diagram of the product identification microservice

3.4.4.3 Sequence Diagrams

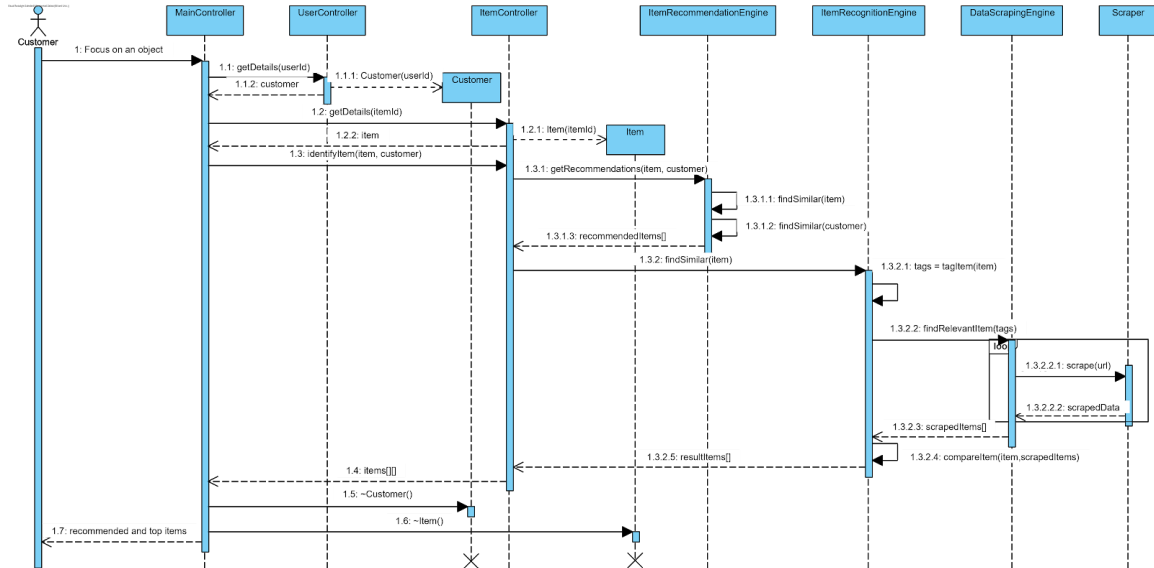### 3.4.4.3.1 Customer gets top match and recommendations

Figure 7: Sequence Diagram - Customer gets top match and recommendations

When the customer is on the main screen of the application, he will have to point the camera in the direction of a clothing item. Then upon keeping the camera stable for a while, the application will trigger an event of sending a request to the backend API with the image and customer information. The image will be passed to the ItemRecognitionEngine and in turn, the ItemRecognitionEngine will tag the image. The ItemRecognitionEngine will pass these tags to DataScrapingEngine which will return a list of similar items scraped from the web according to the tags. ItemRecognitionEngine will rank these images according to the similarity level between them and the original image received from the customer. Meanwhile, the ItemRecommendationEngine will generate tailor-made recommendations for the customer. Finally, these sets of images will be returned and displayed to the customer on mobile application.

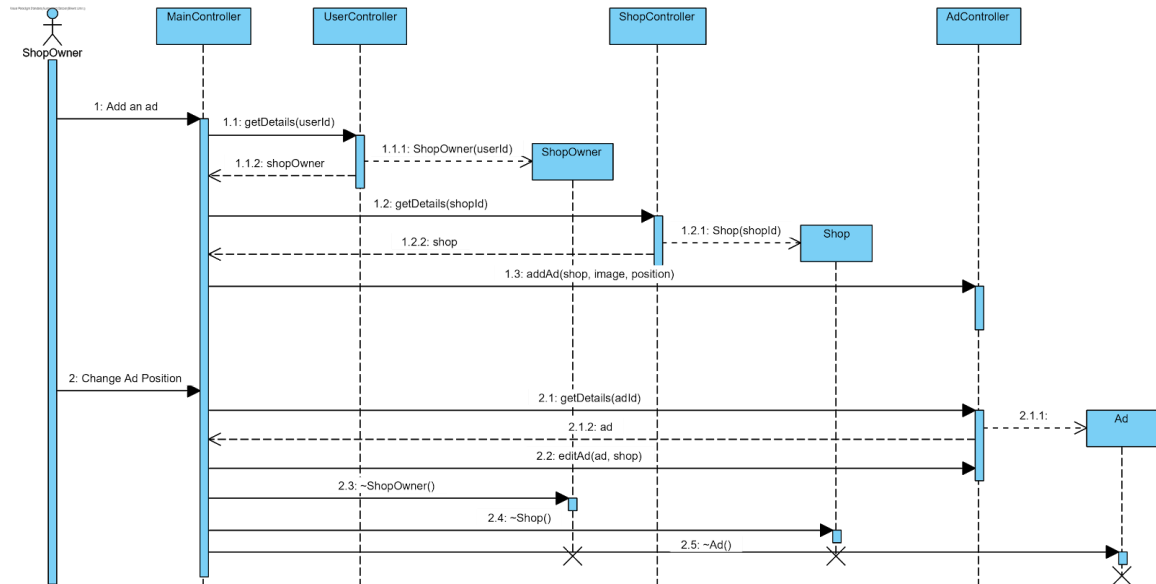### 3.4.4.3.2 Shop Owner adds an ad and updates its position

Figure 8: Sequence Diagram - Shop Owner adds an ad and updates its position

When the shop owner creates a new ad, he will have to add an image to the AR view and position the image to his liking. He will then press the 'Add ad' button which will send a request to the backend API with this information. This information will be added to the database. When the shop owner changes the position of an existing ad, he repositions the image in AR view and confirms his choice. This triggers an API request to the backend which will retrieve the existing ad and update its position in the database.

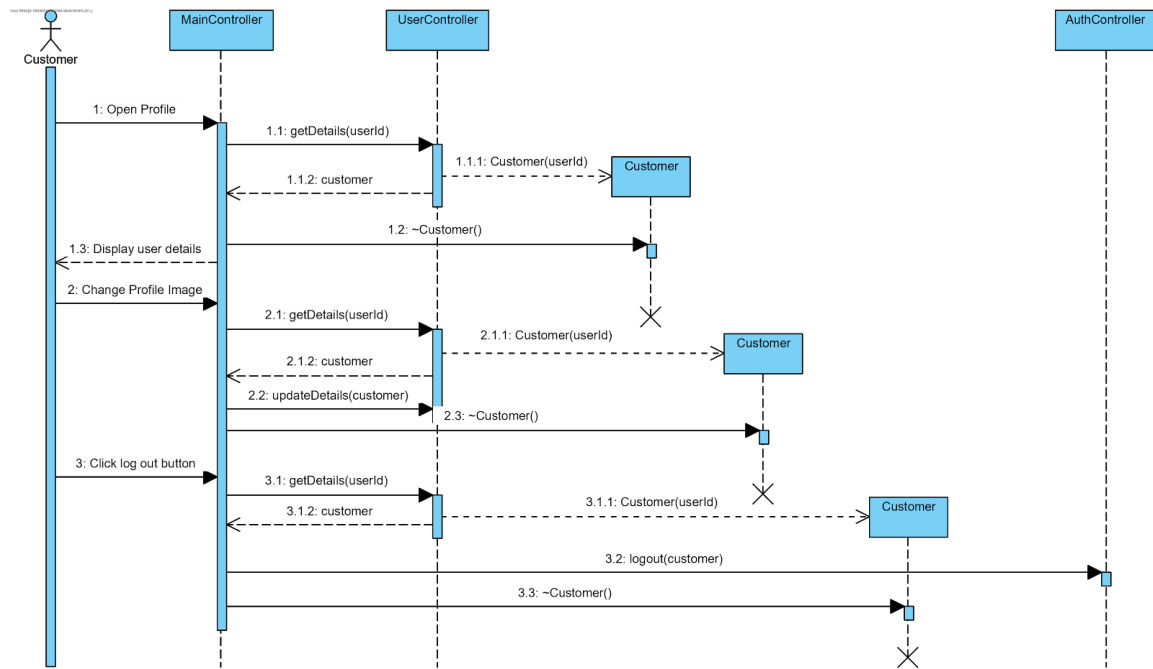### 3.4.4.3.3 Customer changes user details and logout

Figure 9: Sequence Diagram - Customer changes user details and logs out

When the customer goes to his profile page in the mobile application, his details will be retrieved from the backend through an API request. When the customer changes his profile image, he chooses a new image and confirms his choice. This triggers an API request again with the new profile image data in it. This image data will be updated in the backend database. The customer then clicks the logout button which will send an API request changing his loggedIn status in the database system.

3.4.4.3.4 Customer opens wishlist, checks the reviews, and post a review
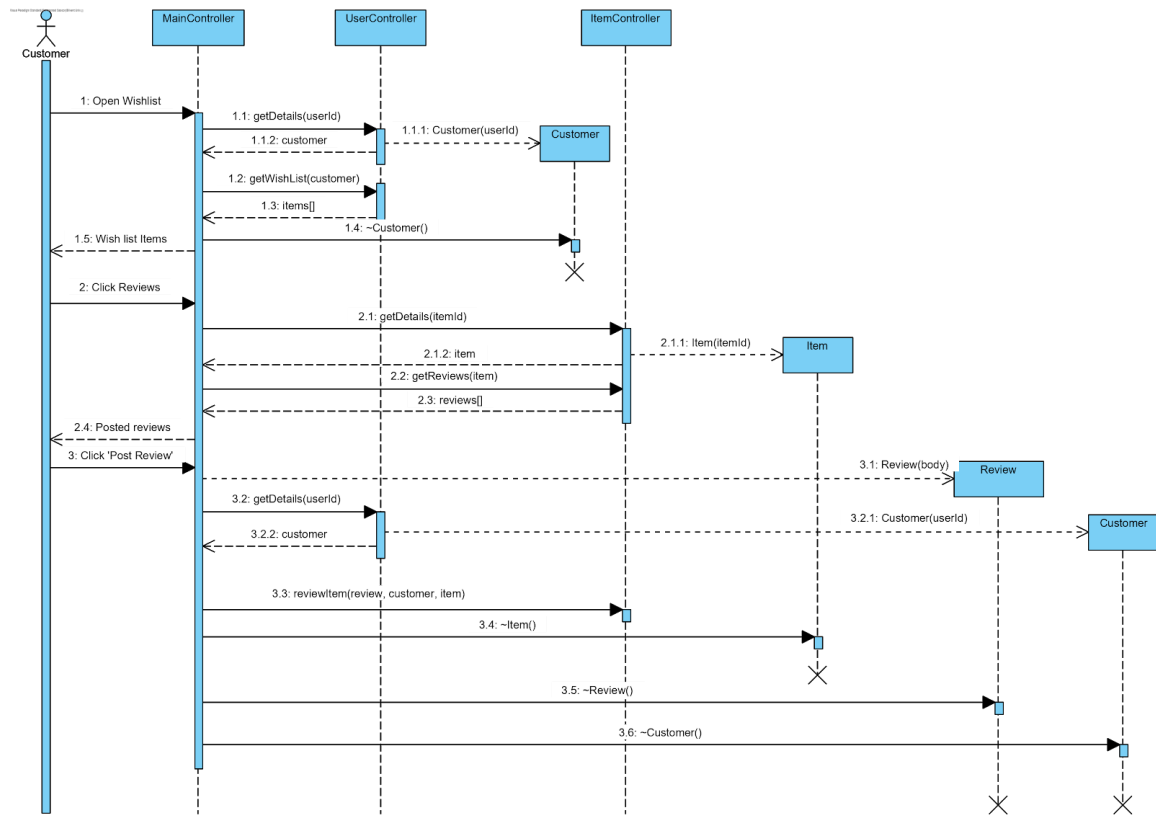

Figure 10: Sequence Diagram - Customer opens wishlist, checks the reviews, and post a review

When the customer opens the wishlist from the menu in the application, an API request is generated which finds the wishlist of the current customer from the database. The customer then clicks the 'view all' button under the 'Reviews' section of the item of his choosing, generating an API request which finds the item and retrieves its reviews from the database. These reviews are displayed in front of the customer and a choice is given to the customer to enter his own review. When the user enters his review and clicks the 'add review' button, an API request is generated which sends the review and item to the backend and the review is added in the record of the item in question.

3.4.5 User Interface

Login Page

This will be the welcome page of our application where the user can choose to login if they have an account or choose to create a new account as a Shop Owner or as a Customer
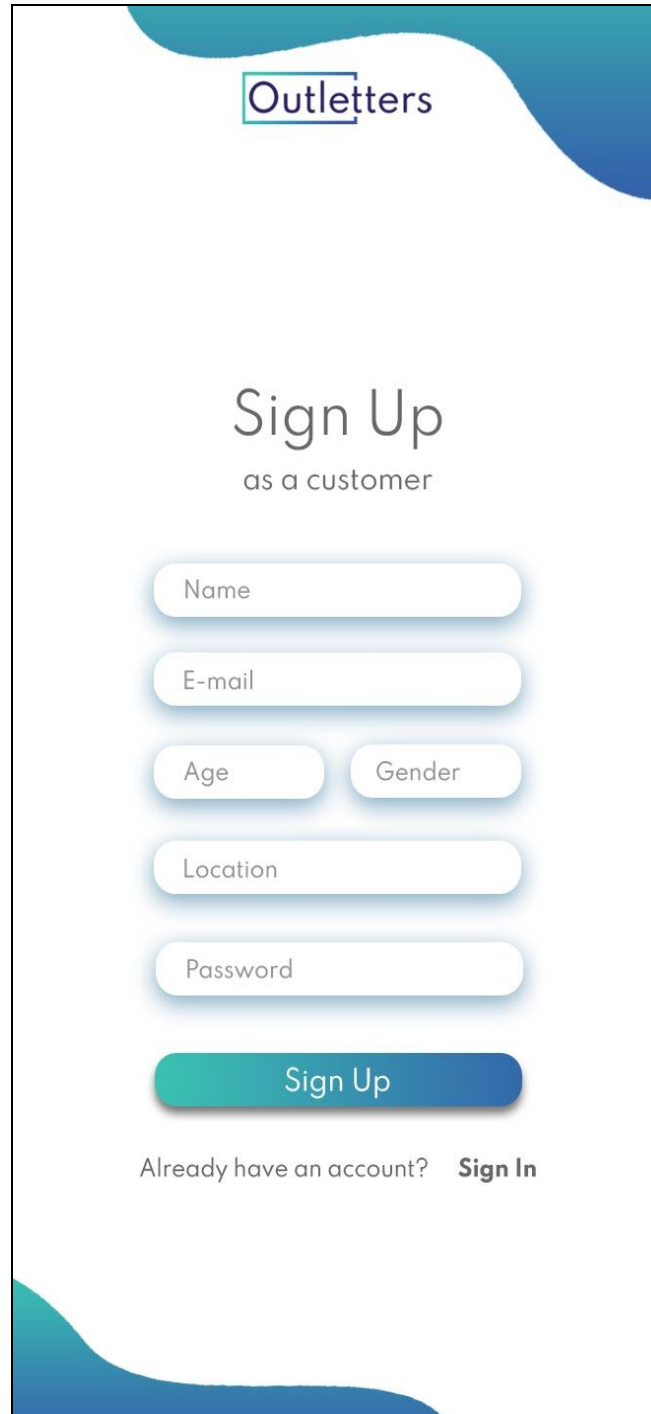
Figure 11: Login page

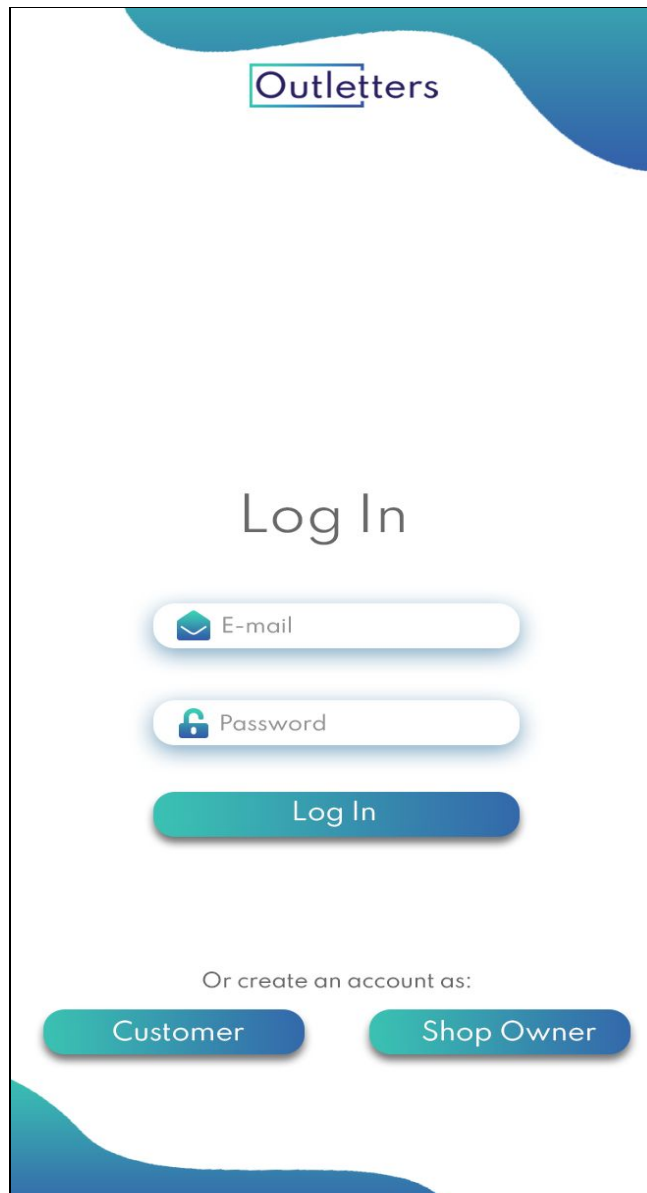**Signup Page**

The user will be able to create a new account. The mockup below shows the signup screen for the Customer users.

Figure 12: Sign up for Customer page

Customer Screens:

The following screens will be available for user who logged in as Customer

Camera Screen

The user will be able to move around the camera and take pictures of a product.

Figure 13: Camera screen

Product Found Screen 1

If the product scan is successful, the best deal for the product will pop up on the screen as well as a scrollable menu below which will show more details regarding the product as well as other recommended products.
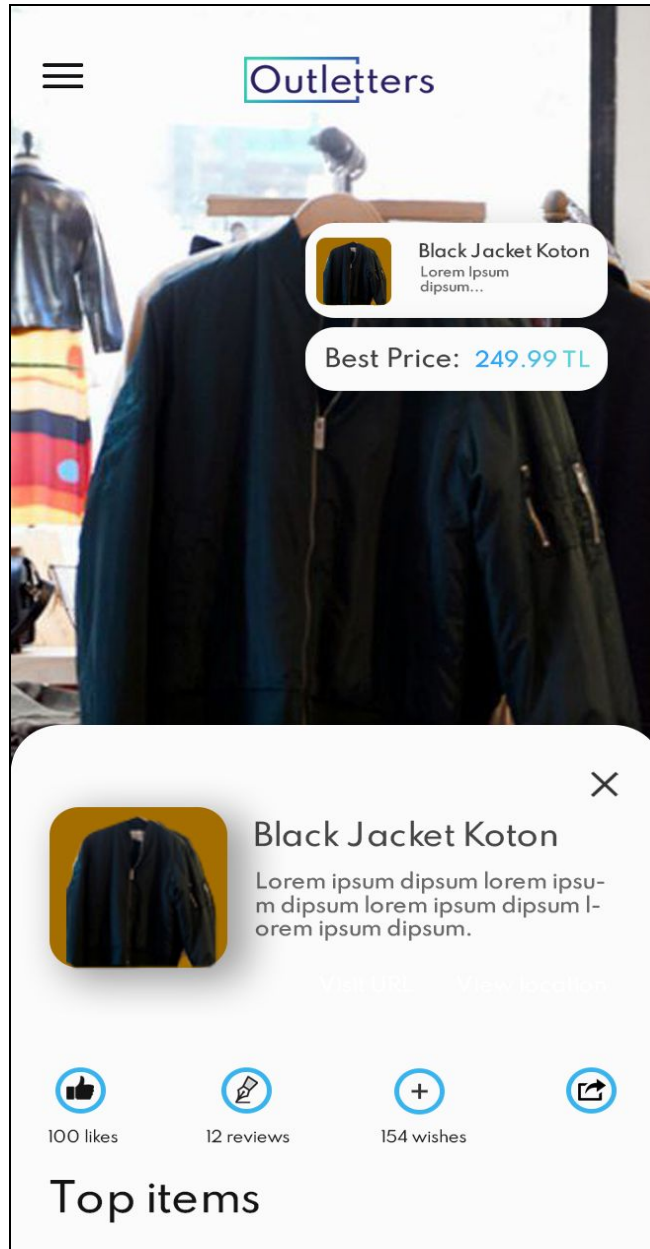
Figure 14: Product found page

Product Found Screen 2

If the user slides up the menu at the bottom, the menu will cover up the screen and show details such as number of likes, reviews, location etc. As well as options such as add to wishlist and like the product. There are also Top Items which are products similar to the one scanned and Recommended Items which are products people bought along with the scanned object.
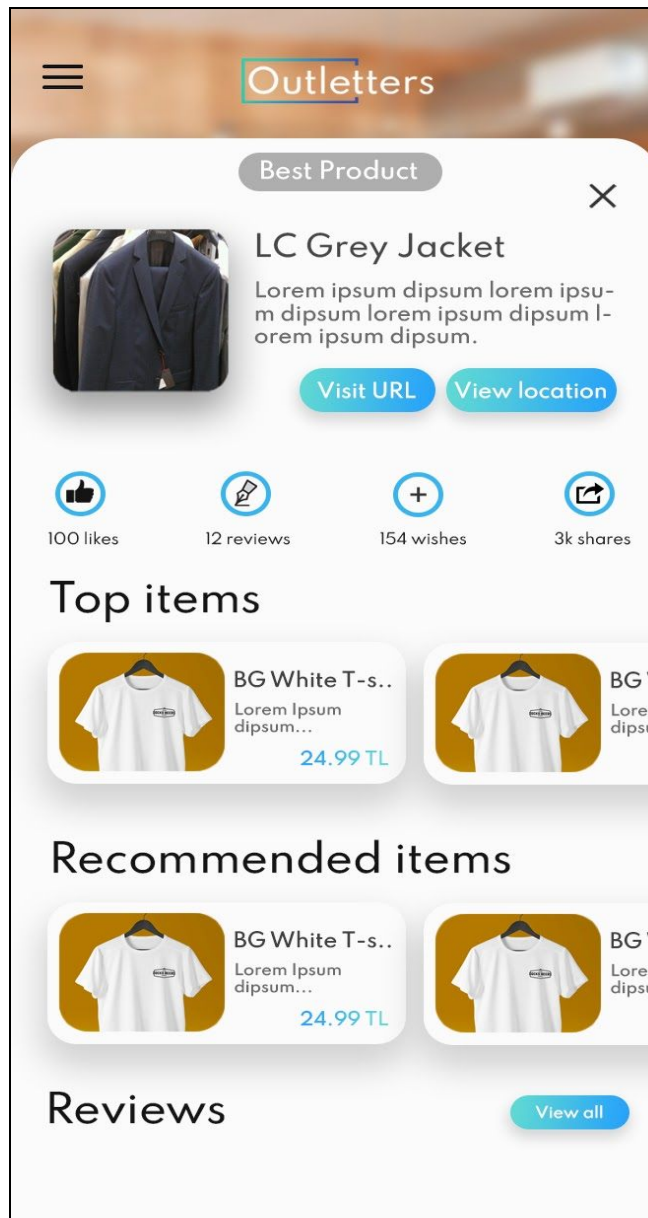
Figure 15: Product results page

Product Not Found Screen

If the product is not found, the user has the option to scan it again by pressing the refresh button. The slidable menu will also appear showing Top Items, which are products similar to the one that was scanned.

Figure 16: Product not found page

Product Screen

When the user selects a product by tapping on it on any of the screens, the product screen will open showing details of the particular product.
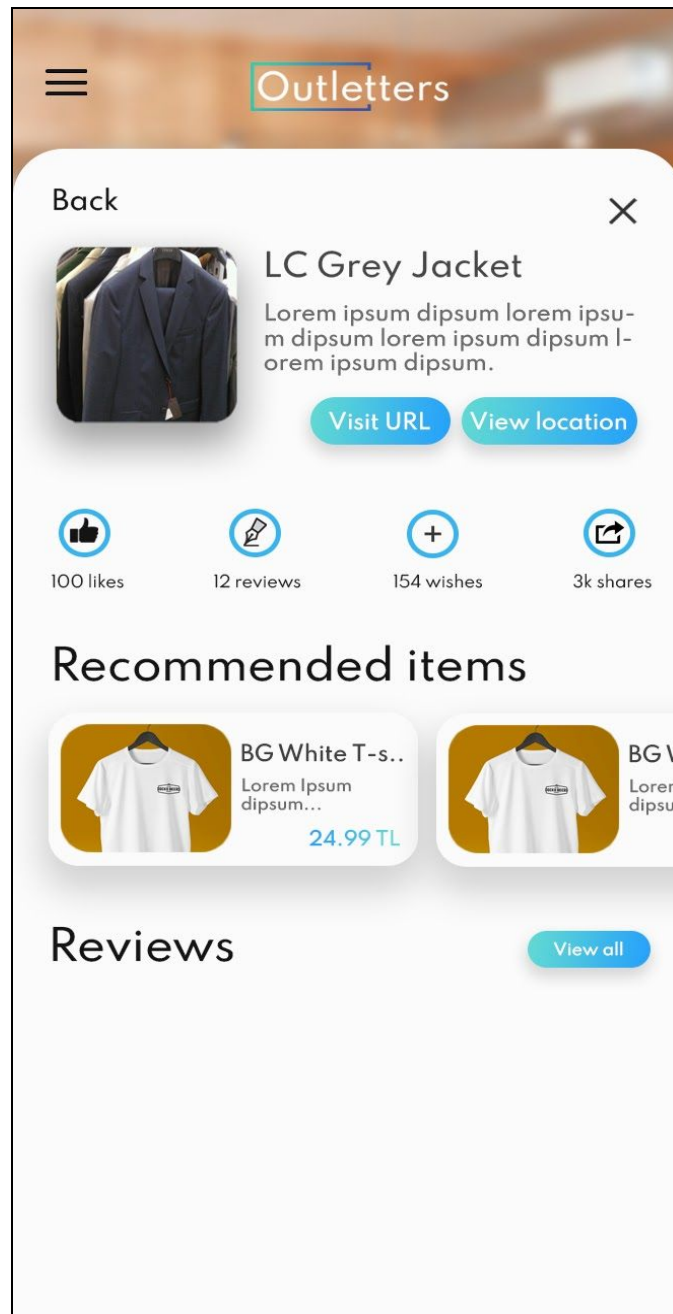
Figure 17: Product Details page

Review Screen

When the user presses the review button of a product, the following screen opens. The users can write their review and/or view other reviews written for this product.
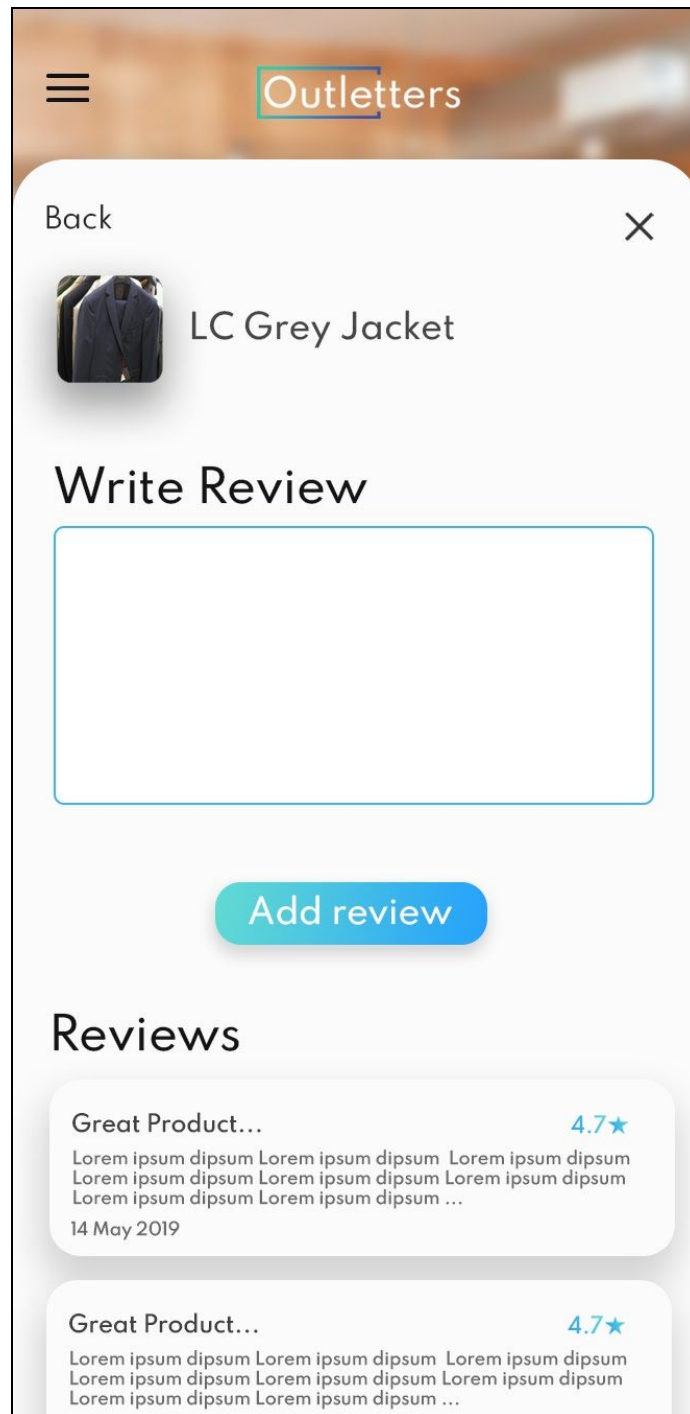
Figure 18: Review screen

Location Screen

If the user presses the View Location button for any product, a map opens in the background showing the locations where the product is available. Tapping on the pin will open the native map application of that phone so the user can find directions for it.

Figure 19: Location screen

Hamburger Menu

Tapping on the hamburger menu option located on the top left side of the screen will open the hamburger menu. The users can view their wishlist, reviews and liked items. They can access settings and choose to log out from the application.
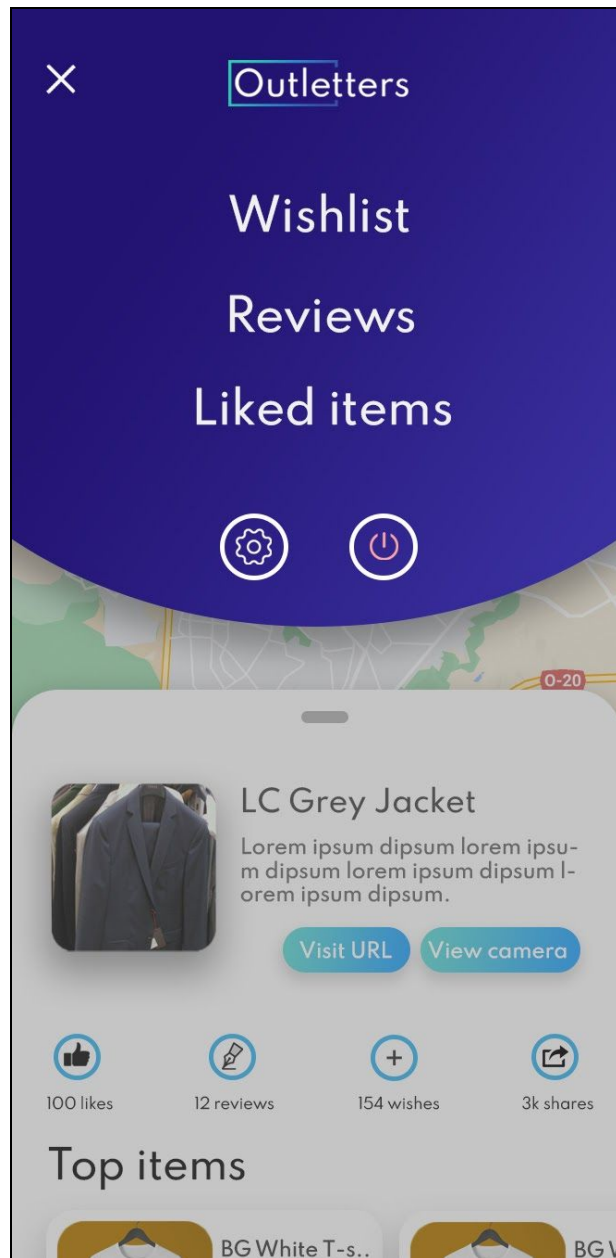
Figure 20: Hamburger menu

Wishlist Screen

From the hamburger menu the user can reach the wishlist screen which shows all the products the user added to the wishlist. The user can tap on a product to open the product screen.
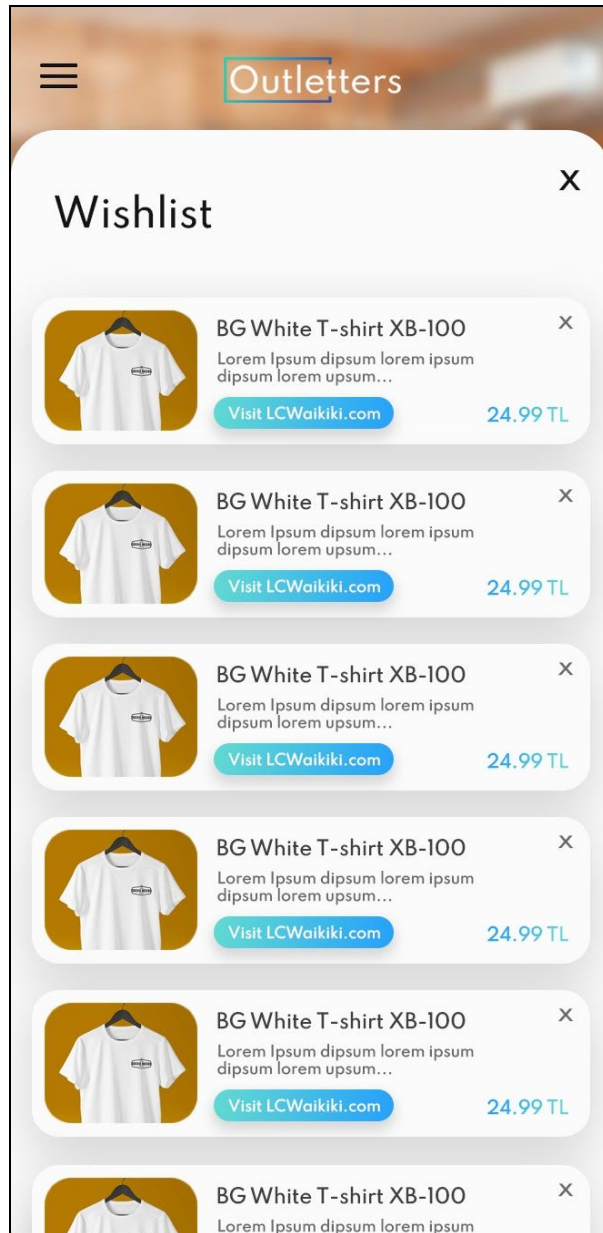
Figure 21: Wishlist screen

Shop Owner Screens:

The following screens will be available for users who logged in as a Shop Owner.

Shop Owner Dashboard Screen

The home screen for the Shop Owner where the user can edit details about their shop and add new ads.
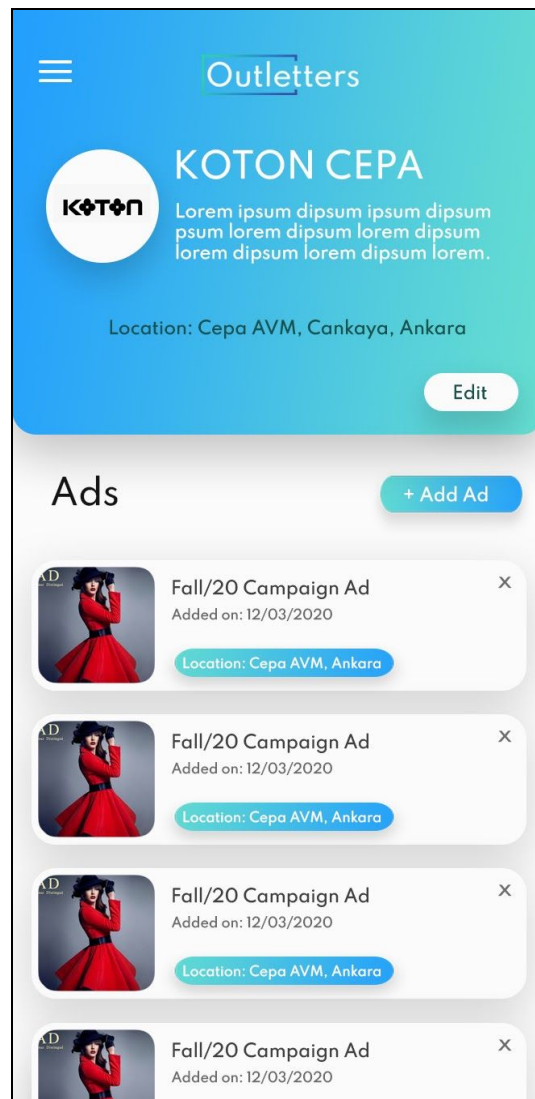
Figure 22: Shop owner dashboard page

Ad Placement Screen

After choosing the ad image from their gallery, the user can place the ad in AR. The top arrow rotates the image and the four corner dots distort the image. Pressing on the place ad button will fix the ad to the place and Customer users will be able to see the ad on their screens if they are at the same place.

Figure 23: Ad placement page

4 Other Analysis Elements

4.1 Considerations of Various Factors

Our project happens to have a wide scope, because shopping is one of the fundamentals of life, that comes right after eating, drinking, and earning. This can lead to adverse effects belonging to certain important categories. These effects are discussed in detail down below.

Public Health

This application does not have any effect directly on the physical health of our users, but it might lead to some undesirable consequences. The application also provides links for online shopping of the scanned products; hence it will promote the 'online' part of shopping. Which will in turn contribute to the unhealthy, 'stay-at-home' mindset in the world [3]. But to counter this effect the application also provides the locations of where the item is physically available as well, so people who preferred the way of online shopping will prefer it more, and the ones who preferred the physical aspect of shopping will prefer that more. In a way this application is just making it convenient for people to shop the way they prefer to. Aside from the effects that the application will have on people physically it might end up having a bigger impact mentally. The application promotes shopping in a way for users to be able to get something that they see. This will in turn promote the 'anxiety' that social media brings along with it. People would want things like other people have and would try to find it on our platform. This will contribute to the depression that is caused by social media in regards to the feeling of self-unfulfillment and which is one of the fundamentals of identity crisis. This is an adverse psychological effect that this application might have on a few of the users.

Public Safety

The application will be very safe, we will be regulating all the shop owners that will be signing up on our platform and verifying the existence of that shop. The only issue that can be in the form of fake reviews for genuine products to decrease sales, but to control that to some extent users can only sign up with one email address, things will help to control the number of scam accounts that can be made on the application. Other than this the application will be fairly safe for public use, free from any dire consequences.

Public Welfare

The application provides users with the best available price for any sort object that they scan, this might make people more prone to spending money on things, and buying items more if they believe that they are getting it at the best price. This is a common aspect seen in basic consumer behavior that people tend to buy things that are on discount or are considered more cheaper than their actual prices [4]. This might lead to people spending more and more money on items that they do not actually need, which in turn would make them have less money on things they actually need. To counter this

issue, the application also provides the users with a recommended list for them to be able to go through in which they are able to find items they require or were searching for previously so the users do not have too much money on a lot of irrelevant items.

Global Factors

The application will be limited to the basic global standards set for every application hosted on Google play, our application will follow Google Play Terms of Service's Developer Program Policy which is used to promote the concept of building trust-worthy applications [5]. Since our application uses user data and stores it as well, we will also limit our development to the General Data Protection Regulations set by the EU to promote security of user data [6]. The guidelines are a must for us to follow and be incorporated in our application design.

Cultural Factors

We do not happen to have any cultural limitation affecting our application because the data received by the users will be personalized for them. The application will not contain any bias towards anyone from any other culture and will based its recommendation on user's likes/dislikes only. Hence our application and Data Model will have no cultural limitations.

Social Factors

The social factors that affect some applications can be considered as religion, age, race, sexual orientation. But again, as stated above, the application will have personalized data for each user, hence the application will also be free from any social bias. The social factor will not be a limitation for our application's development in any way.

4.2 Risks and Alternatives

Since our application relies heavily on mobile phone cameras and internet scraping, there are certain risks involved with using them. These risks are evaluated and their effects on our application are detailed below.

4.2.1. Mobile phone Camera quality and positioning

Since our application relies heavily on the mobile phone camera usage, any disturbance that may be caused to the camera affects our application directly. The application takes the image from the camera as its main input and then preprocesses it to find the relevant shopping products. If that input were to be blurred, or tampered with in any way, the

result would not be what the user expected and would defeat the purpose of our application. Thus having the correct positioning and the best possible camera quality is a must in our application.

4.2.2. Ban from shopping websites while scraping

Since we do not have any data to begin with, we have to search and scrape the internet to populate our database with clothing items. These clothing items might be or might not be what the user would search but a record is still necessary for generating the personalized recommendations for the user. To get these shopping items, we have to scrape many shopping websites which are expected to have some sort of security against automated scraping.This might result in the ban of our IP from those websites, resulting in us getting incomplete shopping data from those websites. This will affect the performance of our application since our machine learning models might fail due to incomplete data.

4.2.3. Failure to acquire dataset for text extraction

While searching for the required clothing item on the internet, we have to infer certain attributes of the clothing item at hand including, color, type, and text written on top of the item. To extract this text, we are planning to use a machine learning model which would require a certain dataset to train it. If we are unable to find such a dataset, our model would not work as expected and we will not get the complete data to search for the clothing item on the internet thus reducing the overall performance of our application.

4.2.4. The change of front-end design of shopping websites

When our user captures an image of a clothing item, it is searched for matches in our database as well as on the internet. If we are unable to find the item in the database, we rely only on the internet scraping. The scrapers are designed according to the front end designs of several shopping websites. If these designs are changed, our scraper would not work properly and thus not get us any viable results. This would result in the decrease of the performance of the application.

4.2.5. The change of url of items in database.

Usually when the user searches a clothing item, we search it in our local database as well. Since the database contains products from already scraped shopping websites,

they might have the queried clothing item. In case the item is found, we have to visit the website again to confirm the updated price and availability of that product. If the URL of the item were to be changed, we would not be able to reach the product details page and would thus end up with outdated data regarding that clothing item.

4.2.6. Plan B

Considering all of these factors, the plan B must consider eliminating the risks of bad mobile phone Camera quality and positioning, ban from shopping websites while scraping, failure to acquire dataset for text extraction, the change of front-end design of shopping websites, and the change of url of items in database. We have come up with the following alternative methods as a plan B to reduce these risks.

Since our input is an image which could be blurred or of low quality, we can reduce the risk of it generating bad results by preprocessing the image. This preprocessing includes noise reduction of the image which can be achieved by using an image processing toolkit e.g opencv.

1. Since we have a chance of getting banned from the shopping websites due to excessive requests, we can use certain tricks to avoid being detected including setting manual rate limits of our scrapers and having a wide pool of IPs from which we can send the requests.

2. Since we require the text on the clothing item to search more efficiently, if our text extraction model fails, we will have to remove the text extraction feature and increase the number of search results from each website to increase the probability of matching the correct clothing item.

3. Since we require the application to scrape every mentioned website for a clothing item, if a website front end were to change, we would not be able to get any results from there. Our short term solution would be to not scrape from that website and our long term solution would be scheduling a validator which would verify that we can scrape the website correctly.

4. Since we require our application to search our local database for the existing items, if our items' URL were to be incorrect, we would simply mark it unusable and later on reindex the item from the internet search. We would extract the features of the image in the database and scrape the internet for any best matching item to replace the outdated item.

The risks and their effects on our application are summarized in the table below along with an alternative to reduce the risk.

Risk Factor	Likelihood	Effect on the Project	Plan B
Mobile phone Camera quality and positioning	60%	Reduction in performance and usability of the application	Noise reduction using an image processing toolkit
Ban from shopping websites while scraping	60%	Failure of Machine Learning models due to incomplete data from shopping websites	Setting Manual rate limits in our scraper and increasing IP pool of scraper
Failure to acquire dataset for text extraction	50%	Text extraction model not being available and thus, reducing the performance of application	Remove the text extraction feature and increase the size web results in case text is detected on the clothing item
The change of front-end design of shopping websites	30%	Reduced performance of the application due to unavailability of data from shopping websites	Do not scrape the website (short term) and schedule a validator to verify that scraper does not fail (long term)
The change of url of items in database	40%	Wrong results will be obtained from the database	Schedule a validator to verify the availability of the webpages of all items in the database and reindex the item to update the URL.

4.3   Project Plan

To ensure a smooth development process and timely design/ development of the components, we have decided to approach this project in multiple subsections that each tackle a part of the larger application infrastructure. As a summary, we will be following the following project plan and dividing the deliverable work-load accordingly:

WP #	Work package title	Leader	Members Involved
1	Analysis	Muhammad Saboor	All members
2	Design	Mian Usman Naeem	All members

		Kakakhel	
3	Scraper Development	Muhammad Bilal Bin Khalid	All members
4	Computer Vision and Machine Learning Models Development	Mian Usman Naeem Kakakhel	Mian Usman Naeem Kakakhel Daniyal Khalil Muhammad Saboor
5	Application Development	Muhammad Arham Khan	Muhammad Bilal Bin Khalid Daniyal Khalil Muhammad Arham Khan
6	Testing	Daniyal Khalil	All members

4.3.1 Work Package 1

Work Package 1: Analysis			
Dates: Week 2 -Week 8			
Leader:	Muhammad Saboor	**Members:**	All members

Objectives:
In this subsection of the project plan, multiple preliminary factors like project requirements and development necessities will be analyzed. Hence, the team will be having a look at the development plan of the project, the requirements, research topics and estimated component requirements and application architecture

Tasks:
- **Task 1.1 - Requirement Evaluation:** Understanding the project requirements and relevant specifications required to develop a working application with the planned features.
- **Task 1.2 - Stakeholder input:** Communicating with the target stakeholders (including the users) to determine and involve their needs in the application as much as possible. This would also allow the team to add/ remove any relevant/ irrelevant features and deliver a product more suited to the target audiences' needs.
- **Task 1.3 - Specification Report:** Analysing the project and determining the specifications for the planned application. This would ensure that the teams follow a feature plan strictly and the final product fulfills all planned initial specifications.
- **Task 1.4 - Analysis Report:** Analyze the project and come up with the user interface designs, use cases, dynamic application models and development structure and plan for the project.

- **Task 1.5 - Requirement Evaluation:** Analyze the project deliverables and the estimated workload in each work package to divide the works appropriately among the team. This will allow proper contributions from all members of the team according to their relevant skill sets

Deliverables:
- **Specifications Report**
- **Analysis Report**

4.3.2 Work Package 2

Work Package 2: Design

Dates: Week 5 - Week 12

Leader:	Mian Usman Naeem Kakakhel	Members:	All members

Objectives:
This subsection of the project plan will help the team come up with relevant project requirements and development factors important for the project's design. Hence, in this package, crucial elements like interface design, high level design of the project architecture, low level design of the application and its plan will be finalized. Considering how important having a good design foundation is for a robust application, this stage will consider all factors including but not limited to project specifications, requirements, use cases and project management packages.

Tasks:
- **Task 2.1 - Analysis Evaluation:** Analysing the project analysis for the design stage to ensure coherence and relevance.
- **Task 2.2 - User Interface Design:** Designing the user interface for the applications front-end.
- **Task 2.3 - High-Level Design:** The project's design goals will be disintegrated into smaller modules that will be handled by individual teams. This stage will also require the team to define the project goals and the high-level requirements from the development phases including the core application architecture.
- **Task 2.4 - Low-level design:** Analyse and generate the low level design requirements for the project. This stage will also help us create the design principles and underlying architecture requirements in detail.

Deliverables:
- **User Interface Report**
- **High-level Design Report**

● **Low-level Design Report**

4.3.3 Work Package 3

Work Package 3.1: Search service development

Dates: Week 8 - Week 10

Leader:	Muhammad Arham Khan	**Members:**	Muhammad Bilal Bin Khalid Muhammad Saboor Muhammad Arham Khan

Objectives: Creating a fully functional cloud-based microservice capable of handling advanced Google search querying using the tags provided to it (tags generated from product identification models discussed later).

Tasks: ● **Determining Google search query techniques:** compiling a list of useful Google search query techniques to enforce clauses like must-have, OR, keyword priority etc. ● **Developing a search scraper:** Developing a scraper using relevant Python libraries that searches Google using a constructed query and returns the Google Shopping tab results in a parsable format (JSON/ XML). ● **Testing/ Demo:** Testing the developed scraper against relevant search queries and in edge cases.

Deliverables: ● **A functional scraper that returns Google Shopping Search results using the provided tags in a parsable format.**

Work Package 3.2: Web scraper development

Dates: Week 10 - Week 20

Leader:	Bilal Bin Khalid	**Members:**	Mian Usman Naeem Kakakhel Daniyal Khalil Bilal Bin Khalid

Objectives: Developing Web Scrapers for commonly listed Shopping websites in the Google

Shopping search results to parse product details if their details show up in the Google Shopping search results from the previously mentioned scraper.

Tasks:
- **Determining top-listed websites on Google Shopping:** Compiling a list of multiple (upto 20) websites that are commonly listed in Google Shopping search results for clothing items.
- **Developing Scrapers:** Developing Web scrapers for these websites to parse the product details such a product image, link, price, and available locations etc.
- **Testing/ Demo:** Testing the developed scraper against relevant product links and in edge cases.

Deliverables:
- **A functional scraper that returns product details from the respective website link in a parsable format.**

4.3.4 Work Package 4

Work Package 4.1: Developing Attribute Extraction Model			
Dates: Week 8 - Week 10			
Leader:	Muhammad Saboor	**Members:**	Mian Usman Naeem Kakakhel Daniyal Khalil Muhammad Saboor

Objectives:
Statement of Work: To find the clothing item on the internet, the application has to know what type of product it is, e.g. shirt, pant, coat, skirt, etc. We also need to know the texture and the patterns of the clothing item. This model will extract these attributes from the image for us.

Tasks:
- **Demo**
- **Model Performance Report**

Deliverables:
- **Model performance report containing performance metrics to indicate the effectiveness of the final model.**

Work Package 4.2: Developing Clothing Item Extraction Model

Dates: Week 8 - Week 10			
Leader:	Mian Usman Naeem Kakakhel	**Members:**	Muhammad Saboor Daniyal Khalil Mian Usman Naeem Kakakhel

Objectives:
After knowing the type of the clothing item, we need to further process the clothing item. Thus, we will need to extract the clothing item from the image. This model will crop the clothing item from the given image.

Tasks:
- **Demo**
- **Model Performance Report**

Deliverables:
- **Model performance report containing performance metrics to indicate the effectiveness of the final model.**

Work Package 4.3: Developing Text Extraction Model			
Dates: Week 10 - Week 12			
Leader:	Daniyal Khalil	**Members:**	Mian Usman Naeem Kakakhel Muhammad Saboor Daniyal Khalil

Objectives:
To find the more similar items on the internet, we also need to know the text written on the clothing items, if any. This model will extract any text from the cropped image from model in work package 4.2

Tasks:
- **Demo**
- **Model Performance Report**

Deliverables:
- **Model performance report containing performance metrics to indicate the effectiveness of the final model.**

Work Package 4.4: Developing Clothing Item Similarity Model
Dates: Week 12 - Week 14

Leader:	Muhammad Saboor	Members:	Mian Usman Naeem Kakakhel Daniyal Khalil Muhammad Saboor

Objectives:
After getting some results from the internet and the database, the application needs to evaluate the similarity of these items to the query item. This model will give a similarity percentage between two images. We will use the result from this model to sort the results according to similarity of items to the query item.

Tasks:
- **Demo**
- **Model Performance Report**

Deliverables:
- **Model performance report containing performance metrics to indicate the effectiveness of the final model.**

Work Package 4.5: Developing Recommendation System Model			
Dates: Week 15 - Week 20			
Leader:	Mian Usman Naeem Kakakhel	**Members:**	Muhammad Saboor Daniyal Khalil Mian Usman Naeem

Objectives:
Our application needs to suggest clothing items which complement the choice and the personality of the customer. We will develop a model based on collaborative and content based filtering to personalize the overall shopping experience for the customer.

Tasks:
- **Demo**
- **Model Performance Report**

Deliverables:
- **Model performance report containing performance metrics to indicate the effectiveness of the final model.**

4.3.5 Work Package 5

Work Package 5.1: Database Design/ Development			
Dates: Week 18 - Week 20			
Leader:	Daniyal Khalil	**Members:**	Muhammad Arham Khan Muhammad Bilal Bin Khalid Daniyal Khalil

Objectives:
Designing/ developing a functional and fully normalized PostgreSQL database.

Tasks:
- **Mapping Object classes to database tables:** Creating relevant tables to avoid redundant data and necessary cross links between the data objects
- **Creating pivot tables and foreign links:** Creating relevant pivot tables for many-many relationships and other foreign id relations for other data relations.
- **Securing user access to database:** Creating limited users with strong passwords to secure the database and limit access to uninvited parties.

Deliverables:
- **A functional and normalized database system with data tables compliant with the proposed class and data models.**

Work Package 5.2: Backend Development			
Dates: Week 20 - Week 30			
Leader:	Muhammad Arham Khan	**Members:**	Muhammad Bilal Bin Khalid Daniyal Khalil Muhammad Arham Khan

Objectives:
Developing a cloud-based backend application using Django that acts as the entry point for all API calls from the mobile application and accesses the relevant web-services to analyze, generate and respond data objects for the user. This includes accessing the models to get product descriptor tags, accessing scrapers to get similar products from websites etc.

Tasks:
- **Develop a barebone backend:** Create a backend cloud-based backend application with relevant controller paths data models necessary for the application.
- **Connect database to the backend application:** Connect the earlier created

database to the backend application and testing features like migration and data seeding to ensure a valid connectivity and data class parallelism.
- **Add relevant integrations for the scraper/ model microservices:** Include necessary integrations (using CURL or similar methods) between the backend and other microservices like web scraping service/ object identification model to analyze and generate relevant data.
- **Create API endpoints in the backend app:** Add necessary API endpoints and their controller routes for the mobile application to be able to communicate with the web backend.
- **Testing/ Demo:** Deploying the backend on a cloud-based compute instance and testing the API calls against test data and edge cases.

Deliverables:
- **A functional backend handling all necessary API calls from the mobile application by performing further microservice invokes or data manipulation.**

Work Package 5.3: Mobile Application Skeleton Development		
Dates: Week 12 - Week 25		
Leader:	Bilal Bin Khalid	**Members:**	Muhammad Arham Khan Daniyal Khalil Bilal Bin Khalid

Objectives: Developing a barebone mobile application to allow development and testing of backend, product identification models, web scrapers and database etc.
Tasks: ● **Initialize React-Native application:** Initialize a basic application using React Native CLI. ● **Add core functionality:** Create basic screens like camera and result views and add Axios integrations to access the backend service.
Deliverables: ● **A skeleton mobile application that allows development and testing of the backend and model validity.**

Work Package 5.4: Mobile Application Development

Dates: Week 12 - Week 25			
Leader:	Bilal Bin Khalid	**Members:**	Muhammad Arham Khan Daniyal Khalil Bilal Bin Khalid

Objectives:
Developing the full mobile application according to the planned UI Design in earlier stages.

Tasks:
- **Creating Login Stack:** Adding the relevant screens and service code for the app entry-point, login verification and signup.
- **Creating Navigation:** Adding relevant app navigators for multiple navigation types in the application design. This is necessary to allow different functionality and rules when navigating between screens inside the app. Some examples of the different types of navigators needed include: SwitchNavigator for Login Stack, Drawer Navigator for app menu access etc.
- **Creating App-wide state management:** Adding redux and redux-saga middleware integration to maintain app-wide state of screens, local data and other core variables. This ensures a cohesive application usage experience across the multiple screens and allows for state persistence even when the app is in background.
- **Adding advanced network access volleys:** Adding advanced Axios integration to allow network access for multiple requests and adding form upload functionalities too. This will also be crucial in adding app-wide request fail handling and network state assessment for optimization purposes.
- **Adding platform specific code:** Since the application will be developed in a hybrid app development framework, to provide better application experience and access native components like camera with more flexibility, native modules will be developed. This will ensure the hybrid app provides an optimized performance similar to those of native applications.

Deliverables:
- **A ReactNative-based mobile application with all the proposed screens and interface functionalities (except for AR integration and API call access).**

Work Package 5.5: ARCore/ Google Maps Integration			
Dates: Week 20 - Week 30			
Leader:	Daniyal Khalil	**Members:**	Muhammad Arham Khan Muhammad Bilal Bin Khalid Daniyal Khalil

Objectives:
Integration of ARCore and GoogleMaps with the existing application

Tasks:
- **Developing Native wrappers for the library:** Since there is a platform native variant of the library available, development of integration wrappers will be necessary to access the functionalities of ARCore in the application.
- **Adding the AR component to relevant Camera views:** Adding the Custom components developed using Native code and integrations wrappers over a camera view to enable the proposed AR functionalities of the app. This includes rendering product details and best prices of the product over the app
- **Creating a Maps view to be used in product locations screen:** Creating a custom view with the required layers of the maps library and locations integration to be used inside the app.
- **Demo/ Testing:** Testing all AR/ location and Maps functionalities of the application.

Deliverables:
- **Functional ARCore integration, GoogleMaps views and location access in the application.**

Work Package 5.6: API Endpoint integration			
Dates: Week 30 - Week 35			
Leader:	Muhammad Arham Khan	**Members:**	Muhammad Bilal Bin Khalid Daniyal Khalil Muhammad Arham Khan

Objectives:
Integrating the API endpoints and relevant data requests in the application to make it functional with the computing and data available from the backend system.

Tasks:
- **Adding relevant environment files for development and production:** Adding environment files for API routes in development and production mode to allow flexibility.
- **Adding Axios data callbacks:** Adding relevant Axios data callbacks including (POST, GET and image form upload) to access the API routes from the backend.
- **Testing:** Testing the API calls to verify their functionalities

Deliverables:
- **A fully functional Mobile Application with all necessary screens, integrations, data access capabilities and navigation routes.**

4.3.6 Work Package 6

Work Package 6.1: Testing Backend			
Dates: Week 35 - Week 40			
Leader:	Muhammad Saboor	**Members:**	All members
Objectives: Testing the functionality of the backend and all its API calls.			
Tasks: ● **Testing database connection and data operation:** Using dummy data to test the database operation for the data classes. ● **Testing API endpoints:** Using Postman to access and test various API endpoints of the backend applications ● **Testing microservice integrations:** Testing applications integration with all microservices including product identification model, web scrapers etc.			
Deliverables: ● **Relevant bug fixes**			

Work Package 6.2: Testing Models			
Dates: Week 38 - Week 42			
Leader:	Daniyal Khalil	**Members:**	All members
Objectives: Testing the product identification and comparison models against sample data			
Tasks: ● **Testing the product identification model:** Testing the model using test product images against expected tags. This will ensure that the model doesn't have any irrelevant outputs ● **Testing the product comparison model:** Testing the model's product comparison percentages and seeing their accuracy. The aspect of comparison speed may also be tested to ensure real-time results			
Deliverables: ● **A functional product identification and comparison model**			

Work Package 6.3: Testing Mobile Application			
Dates: Week 40 - Week 45			
Leader:	Muhammad Arham Khan	**Members:**	All members

Objectives:
Testing all aspects of the mobile application including navigation workflow, edge data cases, API call access, application life-time performance and UI updates.

Tasks:
- **Testing Navigation workflow:** Testing the navigate actions and navigators in the application to ensure a proper user experience and no invalid navigation routes.
- **Testing API calls:** Testing all API calls made by the application and their data bodies and response parsing to ensure that there are no invalid data calls and all exceptions are handled in case of a request failure.
- **Testing application life-time performance:** Testing the applications performance strategy incase of various life-cycle events like onPause, on Stop and app killed in background.
- **Testing application with erroneous or edge data:** Testing app's error handling and exception catching capabilities in case of errored data being transmitted from the backend. This will be tested with intentional false data to ensure the app doesn't crash in such a case.

Deliverables:
- **Bug fixes**
- **A fully functional mobile application for the project**

4.3.7 Gantt Chart

The time measure of weeksis chosen for the time estimates for each work package as it is easier to represent. The Gantt chart prepared for the specified work packages in the previous parts is given below
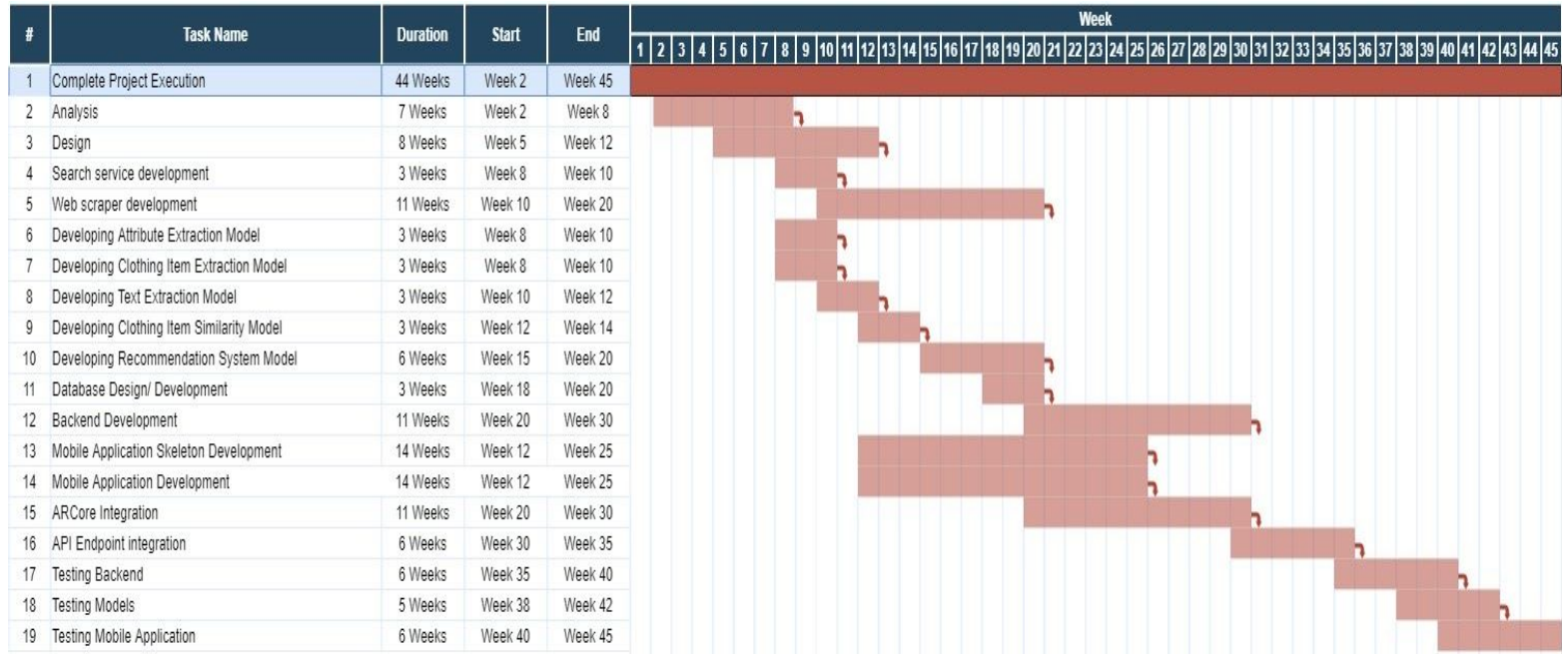
#	Task Name	Duration	Start	End	Week
1	Complete Project Execution	44 Weeks	Week 2	Week 45	
2	Analysis	7 Weeks	Week 2	Week 8	
3	Design	8 Weeks	Week 5	Week 12	
4	Search service development	3 Weeks	Week 8	Week 10	
5	Web scraper development	11 Weeks	Week 10	Week 20	
6	Developing Attribute Extraction Model	3 Weeks	Week 8	Week 10	
7	Developing Clothing Item Extraction Model	3 Weeks	Week 8	Week 10	
8	Developing Text Extraction Model	3 Weeks	Week 10	Week 12	
9	Developing Clothing Item Similarity Model	3 Weeks	Week 12	Week 14	
10	Developing Recommendation System Model	6 Weeks	Week 15	Week 20	
11	Database Design/ Development	3 Weeks	Week 18	Week 20	
12	Backend Development	11 Weeks	Week 20	Week 30	
13	Mobile Application Skeleton Development	14 Weeks	Week 12	Week 25	
14	Mobile Application Development	14 Weeks	Week 12	Week 25	
15	ARCore Integration	11 Weeks	Week 20	Week 30	
16	API Endpoint integration	6 Weeks	Week 30	Week 35	
17	Testing Backend	6 Weeks	Week 35	Week 40	
18	Testing Models	5 Weeks	Week 38	Week 42	
19	Testing Mobile Application	6 Weeks	Week 40	Week 45	

Figure 21: Gantt Chart for Project Outletter

4.4 Ensuring Proper Team-work

Teamwork is the most essential quality that affects every project's development and achievability. In our application, since we are all very eager to learn, each one of us agrees on having a collaborative development environment. So, each member can have an equal learning experience from this project. To make sure that each of us performs our responsibilities on time and provide our level best to make this product go beyond, we set a certain number of factors which will determine the effort each member is putting in this project.

· No. of GitHub Commits.

· No. of GitHub Issues Resolved.

· Activeness is Whatsapp Group.

· Activeness in Teams Meetings.

· Number of Bugs solved.

· Number of Bugs reported.

· Availability in real life Meetings.

To ensure proper teamwork, we all need to have a balanced scale for all the factors provided above. Members that fail to uphold the above given criteria will be asked for their reasons and will be motivated and helped by other members to be able to achieve more than they already have. Since we all believe that blaming each other and undermining the efforts of others will always end up being less productive in the long run.

4.5   Ethics and Professional Responsibilities

The ethical and professional responsibilities of Outletter will be discussed in regards to the potential global impact it can have. The goal of Outletter is to provide its users a quick and convenient way to find out the best offers on a product they are buying so that they can make informed decisions regarding their shopping. This application will not be restricted to any group of people and will be available for download on any application market. Due to this unrestricted accessibility, it is expected that the impact of this application will be to an integration of online and physical shopping worldwide. The global impact of this application can be divided in terms of economy, environment and society.

In terms of economy we have the possible economic impacts the application can bring about and the possible economic constraints it might face during the development phase. Outletter has the potential to bring about an increase in shopping, which in turn will increase the sales of products like clothes. At its launch, Outletter will be available for free with no immediate plans of generating profits by acting as a middleman. Other than economic impacts, there are some economic constraints that may affect the development of the app. These are mostly the costs for using services; some estimates are listed below:

- Publishing the android version of the app on Google Play Store will require a one-time payment of 27 USD.
- Publishing an application on Apple App Store has a subscription fee of 99 USD.

- Google's Custom Search JSON API provides 100 search queries per day for free. Any additional requests will cost $5 per 1000 queries and are limited up to 10k queries per day.

These costs are affordable and not expected to cause any restrictions. However, in the case where we come across unexpected costs, it can potentially limit us in some aspects.

The next impact that Outletter has is Environmental. As stated above, it is expected that Outletter can bring about an increase in shopping, both online and physical. Increase in online shopping may lead to more pollution and production of waste [7]. With more online orders, there will be an increase in transportation for the products and it will cause more air pollution. Additionally, the packaging for the will contribute to the increasing amount of plastic waste, as well as other forms of waste. As for physical shopping, there may be an increase in the usage of vehicles which contributes to air pollution.

The last impact is the Societal impact of Outletter. The biggest issue regarding ethics is privacy. Users of this application will constantly be "scanning" or taking pictures of products and it is possible for the users to take unconsented photos of people who are in the background. Outletter counters this issue by filtering the noise and unwanted parts of the photos. Regarding data privacy of users, the application will be in accordance with the Code of Ethics and General Data Protection Regulation (GDPR) [8]. The application will not store any photos which are not consented by the user.

4.6   New Knowledge and Learning Strategies

Currently our team has a diverse surface knowledge on most of the technologies we will be using. So, throughout the course of the development of this application, we will be acquiring new skills and plan learning strategies in order to do so. Some of the technologies we will use are:

- Machine Learning
- Augmented Reality
- Image Processing

- Application Design
- Android and iOS Development

There are courses available at Bilkent for topics such as machine learning and image processing, however, we will require more in-depth and practical knowledge for these. Augmented reality is a new field for our team and we will need to learn how to integrate it in our project. We also need to have a good understanding of android and iOS development as well as application design as we are going to deploy this as a commercial application.

In order to learn these new technologies, the following methods will be used:

- Literature Review
- Online Learning
- Applied Learning

Literature review will help us find and learn the latest/most useful technologies being currently used. We will apply the most ideal technologies for our machine learning model and use of augmented reality. Online learning will provide a vast variety of tutorials which will aid us in many parts of our project. Finally, applied learning will be the most important as we will gain the most knowledge by using these technologies for ourselves, learning from our mistakes and using our creativity to solve the problems.

5   References

1.   Mohsin, Maryam. "10 Online Shopping Statistics You Need to Know in 2020 [Infographic]." [Infographic]. Oberlo, September 29, 2020. https://www.oberlo.com/blog/online-shopping-statistics.

2.   "React Native · A Framework for Building Native Apps Using React." Accessed October 12, 2020. https://reactnative.dev/.

3.   J. Carter, "Convenience-craving shoppers becoming ever-more lazy," *MyCustomer*, 04-May-2014. [Online]. Available: https://www.mycustomer.com/selling/ecommerce/convenience-craving-shoppers-becoming-ever-more-lazy. [Accessed: 21-Nov-2020].

4.   "Chapter 3 Consumer Behavior: How People Make Buying Decisions," *Consumer Behavior: How People Make Buying Decisions*. [Online]. Available: https://2012books.lardbucket.org/books/marketing-principles-v1.0/s06-consumer-behavior-how-people-m.html. [Accessed: 21-Nov-2020].

5.   "Developer Program Policy (effective October 21, 2020) - Play Console Help," *Google*. [Online]. Available: https://support.google.com/googleplay/android-developer/answer/10177647?hl=en%5D. [Accessed: 21-Nov-2020].

6.   "Official Legal Text," *General Data Protection Regulation (GDPR)*, 02-Sep-2019. [Online]. Available: https://gdpr-info.eu/. [Accessed: 21-Nov-2020].

7.   Jaller, Miguel, and Anmol Pahwa. "Evaluating the Environmental Impacts of Online Shopping: A Behavioral and Transportation Approach." Transportation Research Part D: Transport and Environment. Pergamon, January 24, 2020. https://www.sciencedirect.com/science/article/pii/S1361920919302639.

8.   "Code of Ethics." Accessed October 12, 2020. https://www.nspe.org/resources/ethics/code-ethics.