Bilkent University

Department of Computer Engineering

Senior Design Project

Team ID: T2331

RecoModa

Design Project Final Report

Güven Gergerli 21803393 guven.gergerli@ug.bilkent.edu.tr

Hakan Gülcü 21702275 hakan.gulcu@ug.bilkent.edu.tr

Nasuh Dinçer 21702933 nasuh.dincer@ug.bilkent.edu.tr

Tarık Buğra Karali 21703937 bugra.karali@ug.bilkent.edu.tr

Zülal Nur Hıdıroğlu 21903125 nur.hidiroglu@ug.bilkent.edu.tr

Supervisor: Shervin Rahimzadeh Arashloo

Innovation Expert: Muhammed Naci Dalkıran

19.05.2023

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

1. Introduction

Searching for outfit combinations and finding the best-fitting clothes online can be time-consuming and challenging. According to research that investigates people's shopping behavior, almost %42.8 of the UK spend more than 7 hours online shopping, and shoppers browse different websites to find the best product. [1] RecoModa wants to optimize time spent during online shopping by a more accurate recommendation system of outfit combinations according to the user's taste, allowing the shopper to see the products of different brands. RecoModa is a mobile-based shopping application that acts like social media. It is a platform where shoppers can follow other users and buy their combinations through links. Overall, it aims to decrease the waste of time as much as possible by offering every product with the correct recommendation system using machine learning..

The innovation in this application is to assist shoppers who want to browse different products of different brands with an accurate recommendation system in less time spent. The application suggests the products to the user according to the user taste. In this offering, combining social media and shopping applications to ease finding the best product is intended. The innovation type of RecoModa is incremental innovation because RecoModa adds new functionalities to the services currently in the market.

In this report, the design goals of RecoModa are listed below. In the following sections, the current software architecture and proposed software architecture of RecoModa are discussed. Later, subsystem services and test cases are explained in a detailed way. Lastly, consideration of various factors in engineering design and teamwork details are discussed.

2. Requirements Details

2.1 Functional Requirements

2.1.1 Registration Functionalities

- The application displays unique recommended posts to each user according to the recommendation model, which is determined by the likings of the user.
- The application lists the recommended posts according to the recommendation model below the given categories.
- The application mandatorily requires a username, password, and e-mail.
- The application activates the register button if and only if the mandatory information is correctly given.
- The application requires a unique username, a valid e-mail address, and a minimum 6-character password which includes an uppercase, lowercase, and

special character. Otherwise, the application gives an error message and does not activate the register button.
- The application requires email confirmation for successful registration.

2.1.2 Search Functionalities

- The user can search by typing a string in the text field.
- The user can search by choosing given categories of garments, RGB color values, and sizes.

2.1.3 Post & Profile Functionalities

- The user is able to post outfits into the system, where users can attach the shopping link to each garment..
- The user can see the previous post of another user or self by the profile.
- The user is able to “like” or “comment” on a post that is visible to other users.
- The user can “save” the post or linked garments on the post, which is available to the user in the “saves” section.
- The user can “follow” or be “followed” by another user, which alters the recommendation model.

2.1.4 Information Functionalities

- The user needs to measure their own body in a specified given way. The information includes: weight, height, gender, shoe size, cloth size.

2.1.5 Server Functionalities

- The server recommends posts according to user interactions and returns a post list.
- The server saves and updates posts and user information in the database.
- The server encrypts the user information and then saves them in the database.

2.2 Non-Functional Requirements

2.2.1 Usability:

- It should be easy for a new user to adapt to the application interface
- Users should operate all functionalities such as creating a post, saving a post,, commenting, etc. through their mobile application without any technical knowledge
- The application should suggest the products which match with user’s taste
- It is essential that the application runs smoothly and efficiently in order.

2.2.2 Performance:

- The processes that need high computing power should be handled on the server side so that the program does not drain the battery.

2.2.3 Reliability:

- There should not be any program crash or data loss or any failure with payment
- If a failure occurs, the mean time to fix the failure should be minimum
- The average time between two failures should be maximum

2.2.4 Marketability:

- RecoModa should have a user-friendly user interface
- The application acts like social media, and it gives users to follow other people's combines. If influencers use the application, the range of people using this app should increase.

2.2.5 Extendability:

- The implementation of the application should allow adding new features and functionalities in the future, such as other distributing services and messaging.

2.2.6 Security:

- The sensitive information of the user, such as passwords and credit card information, should be saved in an encrypted format.
- By using a stateless session (JSON Web Token), all data does not need to be saved in a database on the server side like cookies. It only exists on the client side. It eliminates the CSRF attacks [2].

2.2.7 Scalability:

- The prototype of the application should be able to handle 50.000 users and 2000 concurrent users.

2.2.8 Maintainability:

- The RecoModa should have %90 maintainabilities in a day which means the component should be repaired maximum within a day.

2.2.9 Flexibility:

- The application should adapt to future changes and requirements such as messaging etc.

2.2.10 Modularity:

- The application should be divided into smaller modules which ensures software development manageability.

2.2.11 Aesthetics:

- The application should create attractiveness for customers and positive emotions through the design of the front end.

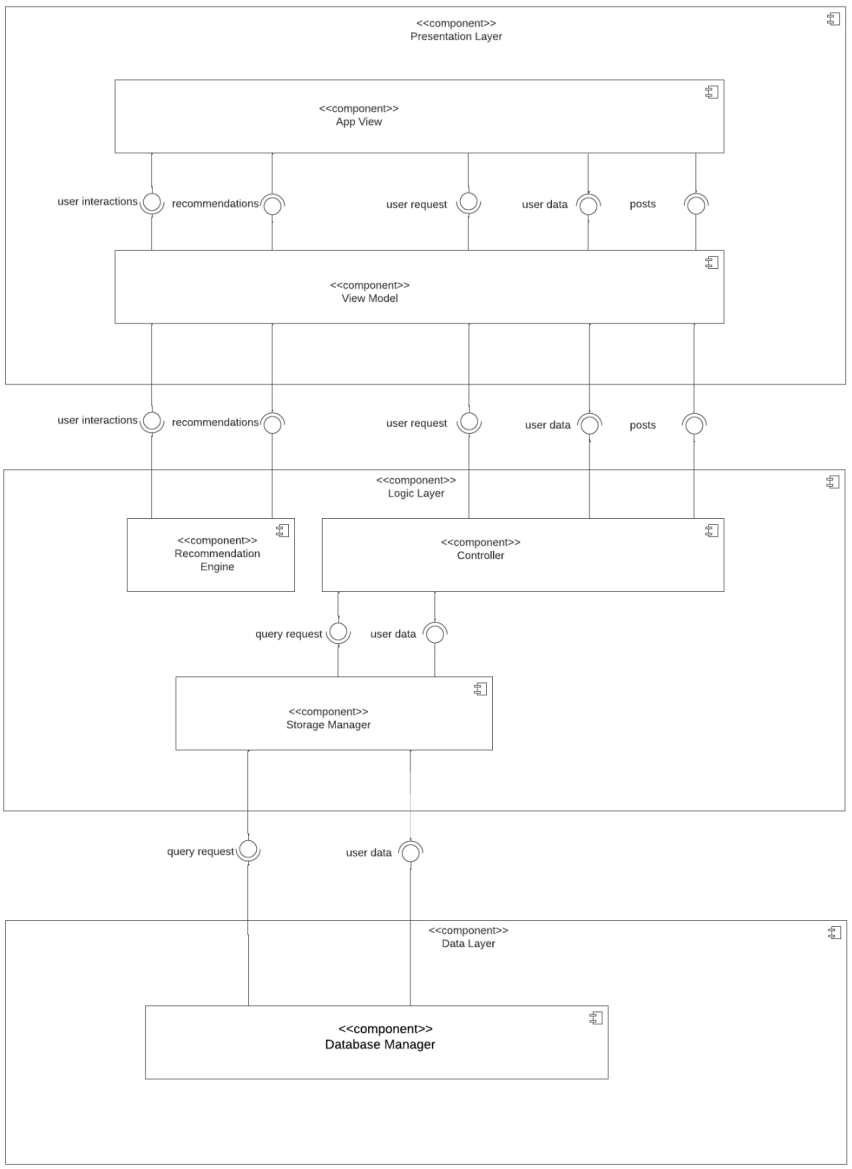
3. Final Architecture and Design Details

3.1 Overview

The system's software architecture and subsystem decomposition are extensively detailed in this section. We required a software architecture that is compatible with our design aims in order to achieve our design objectives. So we decided to split up the project's architecture into various layers, and it depends on these layers communicating with one another.

3.2 Subsystem Decomposition

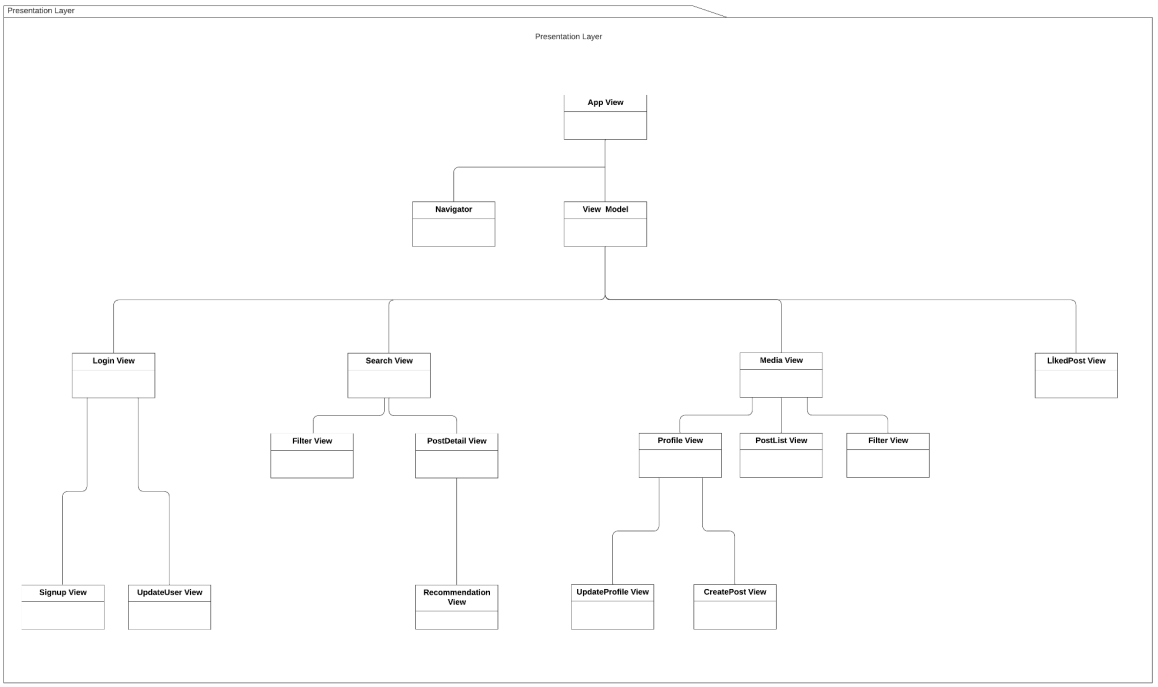
According to the goals of our application's design, the RecoModa system follows the 3-Tier architecture that includes the Presentation Tier, Logic Tier, and Data Tier. Since RecoModa will be a mobile application, the 3-Tier architectural style is appropriate to reduce the interaction between subsystems. This style provides flexibility to develop and update each layer with minimum dependency on other layers. In this architectural style, the client does not have direct access to the database, so it is more difficult for a client to obtain unauthorized data, and it provides security. Because of the increased modularity, one tier can be changed or replaced more easily without affecting the other tier.



Figure[1] subsystems interact

The diagram in figure 1 shows how the subsystems interact with each other. The general overview of the system is given in figure 1. The Presentation Layer represents the client-side. Every interaction which the user performs belongs to this layer (UI). The Logic Layer represents the server-side. Every backend operation belongs to this layer. The Data Layer represents the storage side of the application.

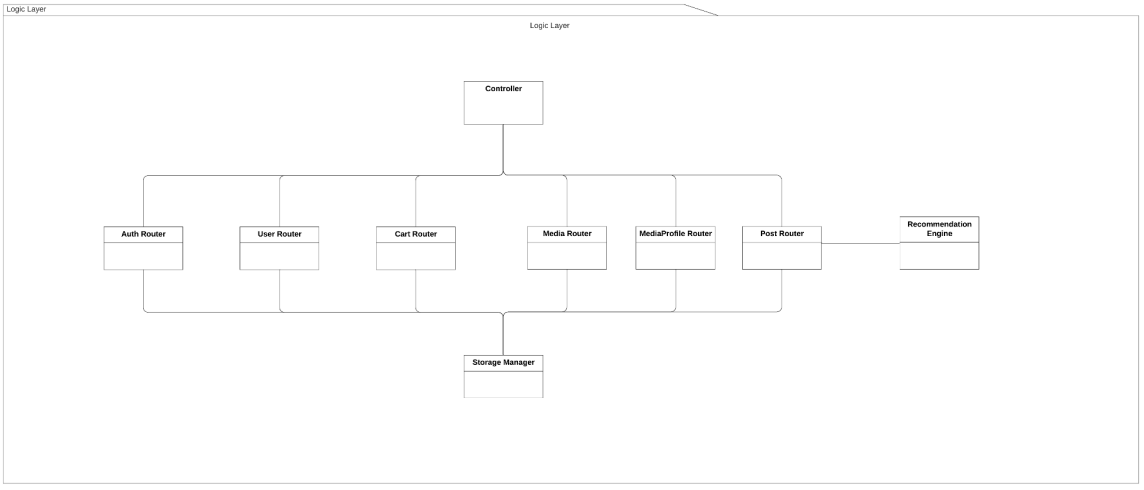
3.2.1 Presentation Layer



Figure[2] presentation layer

The Interface of the application is represented by the presentation layer, as seen in Figure 2. This layer handles all interactions with the user. With API calls, all user requests are passed directly to the logic layer. Then the Logic Layer passes the results of these calls back to the Presentation Layer.

3.2.2 Logic Layer



Figure[3] Logic Layer

The Logic Layer which is seen in Figure 3, includes all of the Presentation Layer's backend functionality. The connection between the presentation layer and the logic layer is made through API endpoints. The machine learning model for recommendation is also placed here. The data that is sent from the Logic Layer to the Data Layer via the Storage Manager.

3.2.3 Data Layer

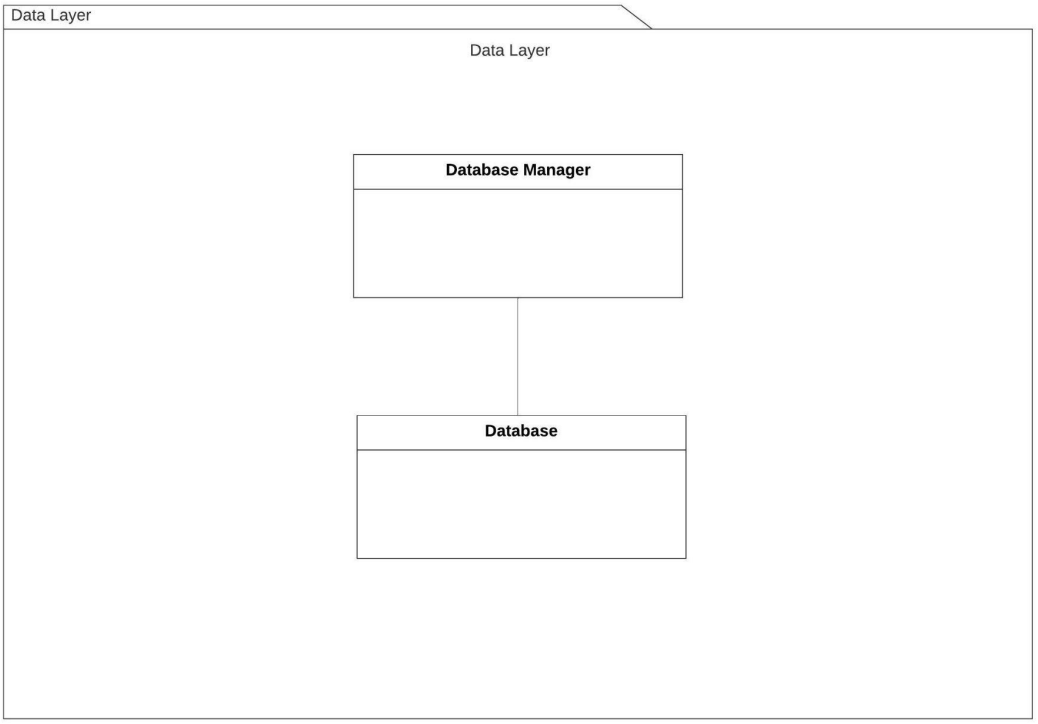
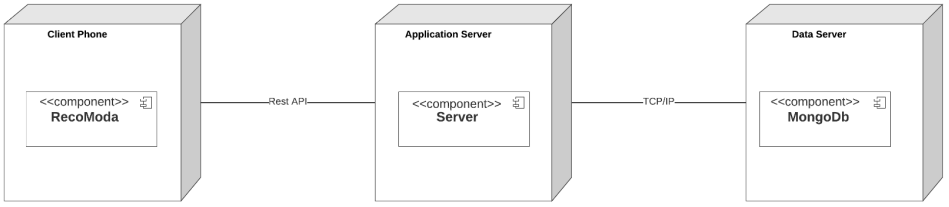


Figure [4] Data Layer

The Data Layer which is seen in Figure 4, represents the database of the application. The database contains the user information, posts and social media information. Since MongoDb is a real time database, the data can be retrieved or updated instantly.

3.3 Hardware/Software Mapping



Figure[5] Hardware/Software Mapping

RecoModa is a react native mobile application that works with Android-powered devices. Users can use their Android smartphones or tablets to communicate with the application's servers. Via their Android devices, users can execute the application’s features.

The deployment diagram given in Figure 5 illustrates the relationship between the application's subsystems. Server subsystem includes the REST API that makes the HTTP request processing with Client Phone. The application backend functionalities are executed in the Server subsystem. The server subsystem accesses the necessary data from the database and sends a response back to the presentation layer in order to meet the requests from the client phone.

3.4 Persistent Data Management

In the RecoModa, users’ personal data and information will be stored in MongoDB. Since MongoDB is a real-time database, users can update their information, and changes are updated instantly. User information should also be persistent after a successful login until the user logs out. This part of the persistent data management will be done using JSON web tokens and redux libraries. Both of the libraries will use the local file system of the user machine. After a successful login, token information will be stored in local storage. Then, it sends this token information in all requests that request authorization (get, post, put, delete…). For every incoming request, the server verifies the token and checks whether the user can access it. Additionally, thanks to the "Redux" library, the user information, including social media information, will be saved locally on users' devices and can be instantly accessed as needed. This will improve the application's operational effectiveness. Since the products are fetched from external APIs, product information is not stored in the database, so the required field for data storage is minimized.

3.5 Access Control and Security

3.5.1 Authentication

In the RecoModa, there are two types of users which are regular users and admin. Since there are different types of users, access to the pages should be controlled. The admin pages should be accessible only to admins. An open standard called JSON Web Token (JWT) outlines a compact and self-contained method for securely transferring data between parties as a JSON object. The fact that this information is digitally signed allows for verification and security. JWTs can be signed using a public/private key pair using RSA or ECDSA or a secret (with the HMAC algorithm) [3]. In order to prevent undesired access, JSON Web Tokens are used. The JWT will be included in each request after the user logs in, enabling access to the routes, services, and resources that are authorized with that token.

3.5.2 Deep Linking

Deep linking presents a special vulnerability for mobile apps. Deep linking allows data to be sent from an external source directly to a native program [4]. Nothing prevents a malicious application from joining the same scheme and hijacking your deep link in order to gain access to the information it holds. While sending app:/products/1 is not harmful, sending tokens raises security issues. To avoid this problem, we set up a universal link (HTTP or HTTPS) login interface and use a random identifier to locally authenticate the incoming login token.

3.5.3 Sensitive Information

Anyone inspecting the app bundle might access anything contained in the code in plain text. Using sensitive information in the code can be insecure. In order to solve this problem, the dotenv library is used. Sensitive informations are defined in the .env file and their instances are used in the code. Also, before personal information is saved to the database, the information is encrypted by using the bcrypt library.

4. Development/Implementation Details

While developing Recomoda, we used React-Native, an open source UI software framework, to develop Frontend and implemented it through the Visual Studio Code IDE. Since React-Native is only for apps suitable for Android platforms, our app can run on Android platforms. Thanks to React-Native's wide variety of modules, we have developed our User Interface mainly using them.

We used MongoDB, which is the NoSQL database management system, as the database system because it has a scalable and flexible structure. It was also suitable for high-volume data storage and since we have a lot of repetitive data, it prevents repetition and redundancy. Also, we store all our data with MongoDB, we don't use cloud services like AWS. Also, the photos were stored in base64 format. Node.js, a fast and reliable framework, was used for the backend.

Python was used as the programming language while developing the Machine Learning model because it is the most widely used language in this field with its wide variety of modules. The keras and scikit-learn libraries are the main modules used in the model. Arrays of size 2048 created using Resnet50 and GlobalMaxPooling2D are saved in the database and then similarities are found using the Cosine similarity method. For the deployment phase, we provided the database-model connection using PythonShell provided by Node.js.

Github was used to keep the project up-to-date and everyone pushed what they did very regularly to the main branch, which is the only branch we have. It also pulled current code before implementing it.

In general, everyone tried to solve problems with various websites such as StackOverflow, ChatGPT, Kaggle and Youtube, as we were not very familiar with what they were doing at the beginning. Many problems were also solved using Pair programming. For example, all model implementation done with pair programming.

5. Test Cases and Results

5.1 Unit Test Results

5.1.1 User tries to write invalid username in username textfield.

Test Case id: 5121
Test Type: Security
Test Case Objective: Checks the username textfield.
Test Case Procedure:

- The username field requires a minimum of 6 characters, a maximum of 16 characters, numbers(0-9), letters(a-z, A-z), and special characters (only underscore, period, hyphen allowed). It cannot be left blank. The username must begin with a character. It cannot include special characters.
- Enter an invalid username to the username textfield.

Expected Result: A warning message that contains the reason why the given username is invalid should be displayed.

Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Username Textfield Tested

5.1.2 User tries to write valid username in username textfield.

Test Case id: 5122
Test Type: Security
Test Case Objective: Checks the username textfield.
Test Case Procedure:

- The username field requires a minimum of 6 characters, a maximum of 16 characters, numbers(0-9), letters(a-z, A-z), and special characters (only underscore, period, hyphen allowed). It cannot be left blank. The username must begin with a character. It cannot include special characters.
- Enter a valid username to the username textfield.

Expected Result: Any warning message should not be displayed.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Username Textfield Tested

5.1.3 User tries to write invalid password in password textfield.

Test Case id: 5123
Test Type: Security
Test Case Objective: Checks the password textfield.
Test Case Procedure:

- The password field requires a minimum of 6 characters, a maximum of 16 characters, numbers (0-9), letters (a-z, A-Z), and all special characters. It cannot be empty.
- Enter an invalid password to the password textfield.

Expected Result: A warning message that contains the reason why the given password is invalid should be displayed.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Password Textfield Tested

5.1.4 User tries to write valid password in password textfield.

Test Case id: 5124

Test Type: Security
Test Case Objective: Checks the password textfield.
Test Case Procedure:

- The password field requires a minimum of 6 characters, a maximum of 16 characters, numbers (0-9), letters (a-z, A-Z), and all special characters. It cannot be empty.
- Enter a valid password to the password textfield.

Expected Result: Any warning should not be displayed.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Password Textfield Tested

5.1.5 User tries to write valid credentials in login template.

Test Case id: 5125
Test Type: Security
Test Case Objective: Checks the login button.
Test Case Procedure:

- Enter a valid username.
- Enter a valid password.

Expected Result: Login button should be enabled.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Credentials Textfield Tested

5.1.6 User tries to write invalid credentials in login template.

Test Case id: 5126
Test Type: Security
Test Case Objective: Checks the login button.
Test Case Procedure:

- Enter an invalid username.
- Enter an invalid password.

Expected Result: Login button should be disabled.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Credentials Textfield Tested

5.1.7 User tries to reset password with an invalid email address.

Test Case id: 5127
Test Type: Security
Test Case Objective: Checks the password reset functionality.
Test Case Procedure:

- Enter an invalid email address.
- Click the reset password button.

Expected Result: A warning message should be displayed.
Priority: Critical
Test Date: 19.05.2023
Test Result: Unsuccess
Service Tested: Reset Password Textfield Tested

5.1.8 User tries to reset password with a valid email address.

Test Case id: 5128
Test Type: Security
Test Case Objective: Checks the password reset functionality.
Test Case Procedure:

- Enter a valid email address.
- Click the reset password button.

Expected Result: “Reset link is sent” message should be displayed .
Priority: Critical
Test Date: 19.05.2023
Test Result: Unsuccess
Service Tested: Reset Password Textfield Tested

5.1.9 User tries to create a post without an image.

Test Case id: 5129
Test Type: Functionality
Test Case Objective: Checks the create post functionality.
Test Case Procedure:

- Login to the application.
- Click the upload post button.
- Enter a valid description.
- Enter a valid link or links.
- Click share button.

Expected Result: A warning message beside the image box should be displayed. Share button should be disabled.
Priority: Minor
Test Date: 19.05.2023

Test Result: Success
Service Tested: Create Post Tested

5.1.10 User tries to create a post without a description.

Test Case id: 51210
Test Type: Functionality
Test Case Objective: Checks the create post functionality.
Test Case Procedure:

- Login to the application.
- Click the upload post button.
- Enter valid images or an image.
- Enter a valid link or links.
- Click share button.

Expected Result: A warning message beside the description box should be displayed. Share button should be disabled.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: Create Post Tested

5.1.11 User tries to create a post with valid credentials.

Test Case id: 51211
Test Type: Functionality
Test Case Objective: Checks the create post functionality.
Test Case Procedure:

- Login to the application.
- Click the upload post button.
- Enter valid images or an image.
- Enter a valid description.
- Click share button.

Expected Result: Share button should be enabled.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: Create Post Tested

5.1.12 User tries to navigate detailed post.

Test Case id: 51212

Test Type: Functionality
Test Case Objective: Checks the detailed post functionality.
Test Case Procedure:

- Login to the application.
- Click the home button.
- Click any image.

Expected Result: Navigate to detailed post page.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Navigate Detailed Post Tested

5.1.13 User tries to like posts at home page.

Test Case id: 51213
Test Type: Functionality
Test Case Objective: Checks the detailed post functionality.
Test Case Procedure:

- Login to the application.
- Click the home button.
- Click like button for any post.

Expected Result: Like number is incremented.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Like Button Tested

5.1.14 User tries to comment posts at home page.

Test Case id: 51214
Test Type: Functionality
Test Case Objective: Checks the detailed post functionality.
Test Case Procedure:

- Login to the application.
- Click the home button.
- Click like comment for posts.
- Enter the text for comment

Expected Result: Comment is displayed.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Comment Button Tested

5.2 Integration Test

5.2.1 User tries to log in to RecoModa with valid credentials.

Test Case id: 5221
Test Type: Security
Test Case Objective: Check the interface link between the Login and other pages.
Test Case Procedure:

- Enter login credentials and click on the Login button.
- Verify that you have navigated to the main page.

Expected Result: If successful, user information should be placedprofile page. Otherwise, a warning message should be displayed.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Login Tested

5.2.2 User tries to like a post on the home page.

Test Case id: 5222
Test Type: Functionality
Test Case Objective: Check the interface of Home Page
Test Case Procedure:

- Login to the application.
- Click the like button.

Expected Result: If successful, the like will be counted as a like number.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: LikeTested

5.2.3 User tries to comment a post.

Test Case id: 5223
Test Type: Functionality
Test Case Objective: Check the interface link between the Detailed Post and Home Page modules.
Test Case Procedure:

- Login to the application.
- Click on a post to see detailed post information
- Click to comment button

- Navigate to detailed post page

Expected Result: If successful, user navigate to detailed post page and comment for the post.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: Comment Tested

5.2.4 User tries to follow a user.

Test Case id: 5224
Test Type: Functionality
Test Case Objective: Check the interface link between the Home Page and User Profile modules.
Test Case Procedure:

- Login to the application.
- Click on a post.
- Click the follow user button placed on the user's media profile.
- Click on your own user media profile.
- Verify that the user's media profile is added to the following list on your own user media profile page.

Expected Result: If successful, the user should be added to the follower list. Otherwise, there will be a warning to the user that the post has not been removed due to a connection problem.
Priority: Minor
Test Date: 19.05.2023
Test Result: Unsuccess
Service Tested: Follow Tested

5.2.5 User tries to unfollow a user.

Test Case id: 5225
Test Type: Functionality
Test Case Objective: Check the interface link between the Home Page and User Profile modules.
Test Case Procedure:

- Login to the application.
- Click on profile page.
- Navigate to following page
- Click the unfollow user button placed on the user's media profile that the user already follows.

Expected Result: That user should be deleted from the follower list.
Priority: Minor
Test Date: 19.05.2023

Test Result: Unsuccess
Service Tested: Unfollow Tested

5.2.6 User adds posts to user’s whistlist.

Test Case id: 5226
Test Type: Functionality
Test Case Objective: Check the interface link between the Post and Whistlist Pages
Test Case Procedure:

- Login to the application.
- Click on the home page.
- Click on the bookmark for any post.
- Verify that the post is shown in the user’s wishlist.

Expected Result: The post should be added to the wishlist page.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: Add Whislist Tested

5.2.7 User tries to post a new post.

Test Case id: 5227
Test Type: Functionality
Test Case Objective: Check the interface link between the Navbar and Upload Post.
Test Case Procedure:

- Login to the application.
- Click on the new post button in the navbar.
- Fill in the required areas to create a post.
- Upload the media to the post.
- Click the upload button.

Expected Result: The post should be added to the user's media profile. If some of the required areas are not filled, or the image is not uploaded, the user will be given an error message.
Priority: Moderate
Test Date: 19.05.2023
Test Result: Success
Service Tested: Add Post Tested

5.2.8 User tries to delete a post from my posts page

Test Case id: 5228
Test Type: Functionality

Test Case Objective: Check the interface link between the Profile and Detailed Post modules.
Test Case Procedure:

- Login to the application.
- Navigate to profile page
- Click on a post to see detailed post information
- Click on the delete button
- Verify that deleted posts should be deleted from My Posts page.

Expected Result: If successful, deleted posts should be deleted from the my posts page. Otherwise, there will be a warning to the user that the post has not been removed due to a connection problem.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: Delete Post Tested

5.2.9 User tries to update a post in the selected post's details page

Test Case id: 5229
Test Type: Functionality
Test Case Objective: Check the interface link between the Post and Media Profile modules.
Test Case Procedure:

- Login to the application.
- Navigate to profile page
- Click on a post to see detailed post information
- Click on the update post button
- Verify that updated posts should be updated user's media profile.

Expected Result: If the post is updated successfully, the post should be updated in the user's media profile. Otherwise, the system should give a warning message.
Priority: Minor
Test Date: 19.05.2023
Test Result: Unsuccess
Service Tested: Update Post Tested

5.2.10 User clicks the search button on the Filter page.

Test Case id: 52210
Test Type: Functionality
Test Case Objective: Check the interface link between the Search and Filter modules.
Test Case Procedure:

- Login to the application.
- Click on the search page.
- Click on the filter button.

- Select some filters available.
- Click on the search button to see filtered results.
- Verify that the results shown in the search page are filtered correctly.

Expected Result: The Posts that suit with given filters should be listed in the posts page.
Priority: Major
Test Date: 19.05.2023
Test Result: Success
Service Tested: Search Page Tested

5.2.11 User enters a keyword in the search bar and click search button.

Test Case id: 52211
Test Type: Functionality
Test Case Objective: Verify that the user can search for a post they want or want to reach by entering a keyword.
Test Case Procedure:

- Login to the application.
- Click on the search page.
- Click on the search button.
- Enter the keyword wanted to search on the search bar.
- Click on the search button to see filtered results.
- Verify that the results shown in the search page are filtered correctly.

Expected Result: Posts related to the keyword written to the application by the user should be displayed on the posts page.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: Search Page Tested

5.2.12 User selects a category from the filter menu.

Test Case id: 52212
Test Type: Functionality
Test Case Objective: Verify that the user can filter posts by category.
Test Case Procedure:

- Login to the application.
- Click on the search page.
- Click on the filter button.
- Select the category wanted to be displayed.
- Click on the search button to see filtered results.
- Verify that the results shown in the search page are filtered correctly.

Expected Result: Posts related to the selected category should be displayed on the post page.
Priority: Minor

Test Date: 19.05.2023
Test Result: Success
Service Tested: Filter Page Tested

5.2.13 User searches price range from the filter menu.

Test Case id: 52213
Test Type: Functionality
Test Case Objective: Verify that the user can filter posts by price range.
Test Case Procedure:

- Login to the application.
- Click on the search page.
- Click on the filter button.
- Select the price range wanted to be displayed.
- Click on the search button to see filtered results.
- Verify that the results shown in the search page are filtered correctly.

Expected Result: Posts within the selected price range should be displayed on the post page.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: Filter Page Tested

5.2.14 User sorts posts with price sorting

Test Case id: 52214
Test Type: Functionality
Test Case Objective: Verify that the user can sort posts by price.
Test Case Procedure:

- Login to the application.
- Click on the search page.
- Click on the filter button.
- Click on the sort button.
- Verify that the results shown in the search page are sorted based on the selected sorting option correctly.

Expected Result: Posts should be displayed in ascending or descending order based on the selected sorting option.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: Filter Page Tested

5.2.15 User adds posts to wishlist

Test Case id: 52215

Test Type: Functionality
Test Case Objective: Verify that the user can add a post to their wishlist.
Test Case Procedure:

- Login to the application.
- Click on the home page.
- Choose a post.
- Click the Add WishList button.
- Verify that the post added to the user's wish list.

Expected Result: The post should be added to the user's wish list.
Priority: Minor
Test Date: 19.05.2023
Test Result: Success
Service Tested: Add Whislist Tested

5.2.16 User remove posts from wishlist

Test Case id: 52216
Test Type: Functionality
Test Case Objective: Verify that the user can remove a posts from the wishlist. **Test Case Procedure**:

- Login to the application.
- Click on the wishlist page.
- Identify a post that needs to be removed.
- Click on the remove button or icon next to the post..
- Verify that the post is no longer present in the user's wish list.

Expected Result: The specified post should be successfully removed from the user's wish list.
Priority: Minor.
Test Date: 19.05.2023
Test Result: Unsuccess
Service Tested: Remove Whislist Tested

5.2.17 User enter description on upload page

Test Case id: 52217
Test Type: Functionality
Test Case Objective: Verify that the user enter description on upload page.
Test Case Procedure:

- Login to the application.
- Click on the upload page
- Upload image
- Enter a description

Expected Result: The user should be able to enter a description on post
Priority: Major.
Test Date: 19.05.2023

Test Result: Success
Service Tested: Description Add Tested

5.2.18 User comment for a postt

Test Case id: 52218
Test Type: Functionality
Test Case Objective: Verify that the user can leave a review for a posts .
Test Case Procedure:

- Sign in to the app.
- Click on the home page.
- Choose a post to comment.
- Click on the comments tab or section for that post
- Click on the add review button.
- Enter the required information.
- Verify that the review is successfully submitted and displayed on the post's comment section.

Expected Result: The user should be able to leave a review for a post and the review should be successfully submitted and displayed on the post's reviews section.
Priority: Moderate.
Test Date: 19.05.2023
Test Result: Success
Service Tested: Comment Add Tested

5.2.19 User delete comment for a post

Test Case id: 52219
Test Type: Functionality
Test Case Objective: Verify that the user can remove a review for a post.
Test Case Procedure:

- Sign in to the app.
- Click on the posts page.
- Choose a produpost ct that the user already has a comment on.
- Click on the comments tab or section for that posts.
- Click on the edit button.
- Click on the Delete Review button.
- Verify that the review is successfully deleted and no longer displayed on the posts comment section.

Expected Result: The review should be removed from the post page and the user should receive a confirmation message.
Priority: Moderate.
Test Date: 19.05.2023
Test Result: Unsuccess
Service Tested: Comment Delete Tested

5.2.20 User view profile information

Test Case id: 52220
Test Type: Functionality
Test Case Objective: Verify that the user can view their profile information.
Test Case Procedure:

- Sign in to the app.
- Click on the profile page.
- Verify that the user's profile information is displayed.
- Verify that the user can edit their profile information.

Expected Result: The user should be able to view their profile information and have the option to edit it if necessary.
Priority: Minor.
Test Date: 19.05.2023
Test Result: Success
Service Tested: View Profile Tested

5.2.21 User edits and updates their own profile page

Test Case id: 52221
Test Type: Functionality
Test Case Objective: Check the interface link between the Profile and Edit Profile modules.
Test Case Procedure:

- Sign in to the app.
- Click on the Profile page.
- Verify that the user's profile information is displayed.
- Verify that the user can edit their profile information.

Expected Result: The profile information should be updated and reflected in the user's profile page.
Priority: Moderate.
Test Date: 19.05.2023
Test Result: Success
Service Tested: Update Profile Tested

5.2.22 User checks interface link profile to settings

Test Case id: 52222
Test Type: Functionality
Test Case Objective: Check the interface link between the Profile and Settings modules.
Test Case Procedure:

- Sign in to the app.
- Click on the Profile page.
- Verify that there is a link or button to access the Settings page.

- Click on the Settings link or button.
- Verify that the user is redirected to the Settings page without any errors.
- Verify that the Settings page contains relevant options to manage the user's account settings.
- Verify that there is a link or button to access the Profile page from the Settings page.
- Click on the Profile link or button.
- Verify that the user is redirected to the Profile page without any errors or issues.
- Verify that the Profile page displays the user's profile information accurately.

Expected Result: The interface link between the Profile and Settings modules should work smoothly without any errors or issues. The user should be able to access both pages easily and navigate between them. The "Settings" page should contain relevant options to manage the user's account settings, and the "Profile" page should display the user's profile information accurately.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Settings Tested

5.2.23 User checks interface link profile to measurement

Test Case id: 52223
Test Type: Functionality
Test Case Objective: Check the interface link between the Profile and Measurement modules.
Test Case Procedure:

- Sign in to the app.
- Click on the profile page.
- Verify that there is a link or button to access the "Measurement " page.
- Click on the Measurement link or button.
- Verify that the user is redirected to the "Measurement " page without any errors or issues.
- Verify that the Measurement page contains relevant options to manage the user's Measurement preferences, such as body sizes
- Verify that the profile page displays the user's profile information accurately.

Expected Result: The interface link between the Profile and Measurement modules should work smoothly without any errors or issues. The user should be able to access both pages easily and navigate between them.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Measurement Tested

5.2.24 User checks interface link profile to account

Test Case id: 52224
Test Type: Functionality
Test Case Objective: Check the interface link between the Profile and Account modules.
Test Case Procedure:

- Sign in to the app.
- Click on the profile page.
- Verify that there is a link or button to access the "Account " page.
- Click on the Account link or button.
- Verify that the user is redirected to the "Account " page without any errors or issues.
- Verify that the Account page contains relevant options to manage the user's Measurement preferences, such as body sizes
- Verify that the profile page displays the user's profile information accurately.

Expected Result: The interface link between the Profile and Account modules should work smoothly without any errors or issues. The user should be able to access both pages easily and navigate between them.
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Account Tested

5.2.25 User checks interface link search to post

Test Case id: 52226
Test Type: Functionality
Test Case Objective: Check the interface link between the Search and Post modules.
Test Case Procedure:

- Sign in to the app.
- Navigate to search page
- Click on the search bar or icon in the app.
- Enter a keyword or phrase to search for a post.
- Verify that the app displays a post list that matches the search query.
- Click on a post from the search results.
- Verify that the user is redirected to the Post page without any errors or issues.
- Verify that the Post page displays the details of the selected post , such as the name, description, price, and images.

Expected Result: The interface link between the Search and Post modules should work smoothly without any errors or issues. The user should be able to search for a post using keywords or phrases and view a list of posts that match the search query. The user should be able to click on a post from the search results and be redirected to the detailed post page without errors or issues.

Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Search Tested

5.2.26 User tries to get a recommendation.

Test Case id: 5221
Test Type: Compatibility
Test Case Objective: Check the interface link between the Home page and other pages.
Test Case Procedure:

- Enter login credentials and click on the Login button.
- Verify that you have navigated to the main page.

Expected Result: If successful, recommendation list upload in 5 seconds..
Priority: Critical
Test Date: 19.05.2023
Test Result: Success
Service Tested: Recommendation Tested

5.3 End-to-End Test

5.3.1 Users registers and logins into the system

Test Case id: 5321
Test Type: Security
Test Case Objective: Verify users can successfully register and log in to the mobile application.
Test Case Procedure:

- Click and enter the application.
- Verify that the Login page is on.
- Click on the "Register" button.
- Verify that the Registration page is on.
- Fill in the required areas in the registration page.
- Click on the “Submit” button.
- Verify that a success message that the account is created is displayed.
- Verify that the Login page is on.
- Enter the registered username and password into the login and click on the Login account button.
- Verify that the user is redirected to the main page and logs in successfully.

Expected Result: The user should register into the application and then use their credentials to Login into the system. The user then should be redirected to the main page.
Priority: Critical.

Test Date: 19.05.2023
Test Result: Success
Service Tested: Register Tested

5.3.2 User views and adds posts to whislist.

Test Case id: 5322
Test Type: Functionality
Test Case Objective: Verify users can view and select posts .
Test Case Procedure:

- Login to the application.
- Click on the Search or Home page.
- Verify that a list of available post is displayed.
- Click on a post details page.
- Verify that the post details page is displayed.
- Click on the Add wishlist (bookmark) button.
- Verify that a success message is displayed and that the post is added to the shopping cart.

Expected Result: The user should view and select posts with no errors.
Priority: Critical.
Test Date: 19.05.2023
Test Result: Success
Service Tested: Add Whistlist Tested

5.3.3 Users can view wishlist history

Test Case id: 5324
Test Type: Functionality
Test Case Objective: Verify users can view order history.
Test Case Procedure:

- Login the application.
- Click on the wishlist button.
- Verify that the list of previous orders is displayed.
- Click on a random order to view its details.
- Verify that the order details page is displayed.

Expected Result: The user should view and select posts with no errors.
Priority: Major.
Test Date: 19.05.2023
Test Result: Unsuccess
Service Tested: Add Whistlist Tested

6. Maintenance Plan and Details

RecoModa uses MongoDB to store every user's data, information, and images in the system. Since market prices vary often, it is critical to maintain the database up to date so that our designs can be demonstrated to be reliable and resilient. As a result, in the further development plans, we are going to pay attention to database maintenance best practices such as frequent backups, defragmentation, optimized queries, data integrity, and efficient indexing.
[https://www.getsecuretech.com/best-database-maintenance-practices/]

The Recomoda system uses state-of-the-art visual machine learning algorithms and color traction algorithms in the recommendation system. Because of that, having the models work as efficiently and accurately as possible is our main concern. To achieve this, our machine learning developers will be checking the health and efficiency of our models regularly and apply further models and tweakings in need.

7. Other Project Elements

In this section, the other project elements that are various factors in our design, ethics/responsibilities, teamwork details, and new knowledge that is acquired and applied will be discussed.

7.1. Consideration of Various Factors in Engineering Design

Several elements related to the engineering design of RecoModa were explored throughout the final phase of our project.

	Effect Level	Effect
Public Health	1/10	No substantial influence adversely affects the health of the person using RecoModa. The application, on the other hand, allows users to use it as much as they want. This might result in extensive and unbalanced use of RecoModa, which may result in a slight decline in users' physical and mental health. To avoid this type of undesirable circumstance, we advise users of our application not to stare at the screen of their mobile phone for too long because looking at the screen for a long time has a negative effect on mental and eye health. In rare cases, if excessive program use is

		reported, we may consider sending notifications to users to provide short breaks while using the application.
Public Safety	8/10	The information obtained from users for usage in the application is crucial, and its secrecy is one of our top priorities. The body measurement information collected from users will be encrypted and safeguarded and will not be shared with any third parties at any cost. Regardless, due to the significance of this material, we will take all required safeguards against any cyber assault that may seek to steal it. We place a high value on data privacy in our practice, and we are well aware that any data breach could land us in legal problems. As a result, we are working hard to implement innovative approaches that have shown to be effective in data protection.
Public Welfare	5/10	RecoModa does not engage in any practices that are harmful to the well-being of its users or put them under financial strain. RecoModa, on the other hand, as an application, makes promoting, influencing, and purchasing much more accessible. Although it saves users time, many other users with uncontrollable consumption behaviors may suffer financially. To avoid situations like these, we urge that consumers spend within their means and keep track of their financial status when shopping. If this does not solve the problem, we may implement a configurable shopping limitation for those who want to be prevented from overspending.
Global Factors	8/10	The RecoModa app can be used anywhere in the world. Since RecoModa only promotes and does not interfere with selling directly, 3rd parties will handle sales and shipping, and our application will be actively provided in all countries where these parties are available. Using the application for fashion tracking rather than for shopping is also an option. We will release our application in English so that a huge number of people

		may actively utilize it. We will explore delivering the RecoModa program in other languages if demand exists.
Cultural Factors	7/10	RecoModa is both a social and a fashion application. Because fashion can change with culture, we anticipate that users will progressively develop diverse communities based on their cultural dress trends. However, we welcome cultural and fashion distinctions; we try not to overlook culture and to become a multicultural practice for everyone. We will continue to provide unprejudiced recommendations that are appropriate for the users' cultural clothing styles, thanks to RecoModa's recommendation system. As a result, each user can freely shape their preferences in the application.
Social Factors	9/10	Recomoda is an app that aims to unite all its users under fashion's banner. Our application is both a social media and a promotion application. People can follow each other, watch, enjoy, and create material, and even convert it into a source of money. We set out to give users an accessible experience in a social setting. We want people to be able to speak with one another and grow in the fashion industry as long as the norms we set in this social environment are followed.
Environmental Factors	3/10	RecoModa application utilizes cutting-edge application development technologies to enhance efficiency and reduce server workload, making it more accessible to users worldwide. As a result, we intend to make our application more environmentally friendly and to need fewer resources to execute it. However, we do not have any large-scale applications that attempt to center the environmental elements. Thus, our efforts could be more effective.

Economic Factors	5/10	The RecoModa application features a mechanism that promotes and simplifies shopping, resulting in increased shopping due to the extensive recommendation algorithms. We anticipate that our practice will have a good long-term impact on the significant economy because it stimulates the market and increases promotion and purchasing.

7.2. Ethics and Professional Responsibilities

Because the RecoModa program is a recommendation application and for its users, it will gather and evaluate various information from them. The application's goal is to deliver a decent user experience as well as data privacy so that users feel comfortable while using it. All data received from the user during the data-collecting phase (gender, weight, height, clothing size, and shoe size) will not be shared with any other third-party apps or procedures. A user name, password, location, user email, and images shot by camera devices, photos submitted by the users themselves, and other user information are examples of data that cannot be shared. Users will face terms and restrictions while signing up for the application and will be notified appropriately. Our system will operate in accordance with KVKK guidelines. As a result, there should be no concerns about data breaches or ethical difficulties in our RecoModa app.

Another area in which we are concerned about being ethical is the issue of copyright. RecoModa will do a thorough examination of labor and ethics before developing using open-source libraries. When external code is introduced to the program while it is being created, an ethical attitude will be demonstrated by purchasing and contacting the code owner.

7.3. Teamwork Details

During the RecoModa project's implementation, the group members were separated into three groups to work on the front-end, back-end, and data science. After the development phase proceed, the groups became more interactive, and many members worked in several groups. This made the communication between group members persist in the areas that required synchronization, where we regularly switched groups to enhance efficiency. The remainder of this section contains detailed information regarding teamwork.

7.3.1. Contributing and Functioning Effectively on the Team

Güven Gergerli: Güven has played a role in the development of the machine learning model and data science of the project. He has worked on visual machine learning algorithms, exploring data using various tools, setting the data for the implementation of various machine learning models, and connecting it to the application. He has also assisted in the front-end development and designing the application. He also helped in the debugging process on the backend and database parts of the project.

Hakan Gülcü: Hakan has played a role in the development of the machine learning model and data science of the project. He conducted research on which machine-learning models could be used for the project and then worked on implementing and tweaking the visual machine-learning model. He has also contributed to the back-end part of the project. He also played a role in debugging and connecting the back-end and front-end of the project.

Nasuh Dinçer: Nasuh has played a role in the development of the front-end part of the RecoModa application. He has developed the required pages for the application to function. He has also played the role of merging connections between the front-end and back-end sides of the application. He also contributed to front-end debugging and tweaking performance on the front end. He worked on UI design improvement in the front end as well.

Tarık Buğra Karali: Tarık has played a role in the development of the back-end part of the RecoModa application. He has developed the required functions for the application to function. He has handled the server-client connection of the application and created queries for the database. He has also played the role of merging connections between the front-end and back-end sides of the application. He also contributed to the debugging process of the front-end and back-end of the application.

Zülal Nur Hıdıroğlu: Zülal has played a role in the development of the front end of the RecoModa application. She has helped with the development of the required pages and functions for the application. She worked on UI design improvement in the front end as well. She has contributed to both back-end and front-end debugging. She has also played the role of merging connections between the back-end and machine-learning sides of the application.

7.3.2. Helping Creating a Collaborative and Inclusive Environment

We each have our own set of responsibilities, and we all have strictly followed our obligations. We support the person in charge of a task if they have other responsibilities or find the work allocated to them needing help to do. When someone is hesitant about a task, we always help each other precisely describe

what must be done. We discuss the status of the tasks for which we are accountable frequently to keep each other informed all the time. We tried to create a consensus on every significant decision and applied it accordingly. We provide each other feedback on the tasks for which we are responsible and make changes as appropriate. Even when we assigned different duties to each other, we always solicited input from other teammates while working on a task or after finishing it to ensure that the tasks were completed correctly. We had many situations where we did not plan, yet we tried to put out our best efforts to overcome them by discussing and aiding each other.

7.3.3. Taking the Lead Role and Sharing Leadership on the Team

We always shared leadership duties. When we wanted to schedule a meeting, set a timeline, or complete a task as a group, whoever was the most accessible at the moment took the lead. With our obligations, we attempted to guide people in need and placed a high value on it. The idea is for both of us to lighten each other's load while also contributing new perspectives to our advancement. Furthermore, having different leaders at different times does not imply that the leader makes choices independently. We attempted to get an agreement on our thoughts before moving on to reduce the mistakes. Our leader's responsibility is to unite people, combine different ideas, and establish a plan by combining them.

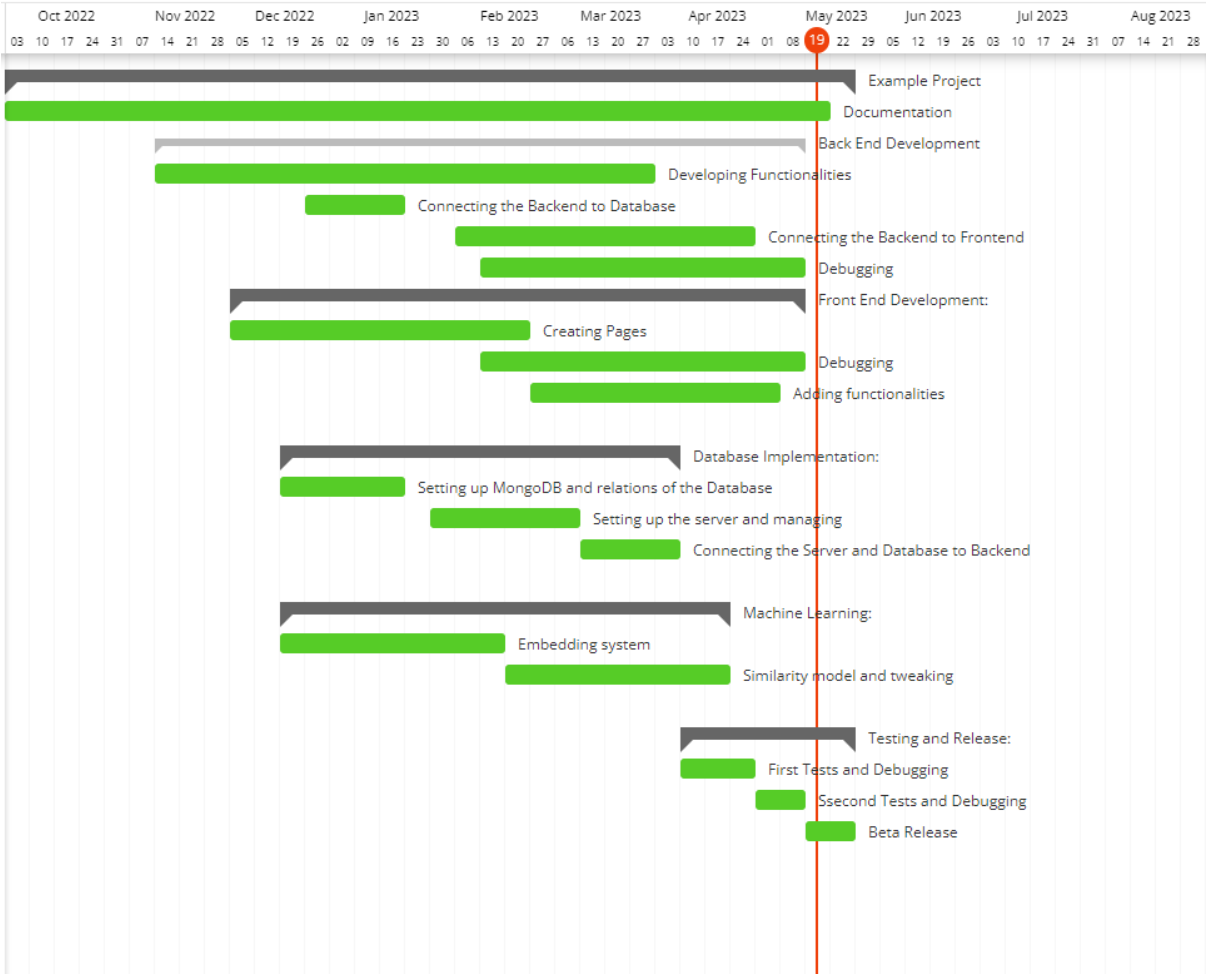
7.3.4. Meeting Objectives

Our project planning was initially set as work packages in the analysis requirements report, which can be seen below.

WP#	Work package title	Leader	Members involved
WP1	Design Analysis and Specifications	Güven Gergerli	All Members
WP2	Application Front-End	Nasuh Dinçer	Nasuh Dinçer, Tarık Buğra Karalı, Zülal Nur Hıdıroğlu
WP3	Application Back-End	Tarık Buğra Karalı	All Members

WP4	Data Privacy	Hakan Gülcü	Hakan Gülcü, Nasuh Dinçer, Tarık Buğra Karalı, Güven Gergerli
WP5	Recommendation Model	Güven Gergerli	Güven Gergerli, Hakan Gülcü
WP6	API Bot and Communication	Hakan Gülcü	Hakan Gülcü, Güven Gergerli, Nasuh Dinçer
WP7	General Test and Release	Zülal Nur Hıdıroğlu	All Members

However, as our progress developed and detailed even more, we as a group wanted to migrate into a well-organized Gantt chart to set up our objectives more precisely. The project plan on the Gantt chart we created before and during our implementation is shown below.



The initial task was to create the deliverables and documentation of CS 491/2 well detailed. All members contributed to the documentation equally, and our group did not have any issues, and deliverables were prepared before the deadline, meeting every goal we set.

The second task was the backend development. In this development group, the functions that are required are created for the system to work healthily. The connections to the database and front end also is made in the group. Also, the system is debugged to work perform correctly and efficiently.

The third task is front-end development. In this development group, the pages that we have discussed and designed before are implemented. Also, we added front-end functionalities and debugged the visuals of the application in this group. The main technology used in this development group is React Native.

The fourth task is database implementation. To store and retrieve data successfully, we have implemented relational tables in this development group. Also, server-related processes are done under the group as well. The main technology used in this development group is MongoDB.

The fifth task is Machine Learning development. We created an efficient and accurate visual machine-learning model that finds the most similar pictures depending on the provided visual data. The main technology used in this development group is Tensorflow and Python.

The sixth and final task is the Testing and Release group. We conducted various tests and debugged the system as a whole. We run two testing and debugging cycles which led to the final Beta Release of our application.

In overall, many members of our group ended up working on a different development group that they started with in which our working habit was much more fluid compared to a hierarchic business model. Therefore, we did not choose to assign a leader to our development groups. The details of who mainly worked on which group can be found in section 7.3.1.

7.4 New Knowledge Acquired and Applied

Throughout the production of this project, we used technologies with which we had some understanding but needed to gain experience. These technologies include the following:

- Node.js
- Express.js
- Keras
- Scikit-learn
- React Native
- NoSQL Databases (MongoDB)
- PythonShell
- REST API
- Expo

We were continually looking for better approaches to integrate the functionality in order to increase the complexity of this project while minimizing errors. We completed several literature reviews before deciding on the applied technologies and spent a significant amount of time and effort learning brand-new technologies. Despite the fact that time constraints forced us to work harder all the time, we were glad to learn about many disciplines of our future occupation.

8. Conclusion and Future Work

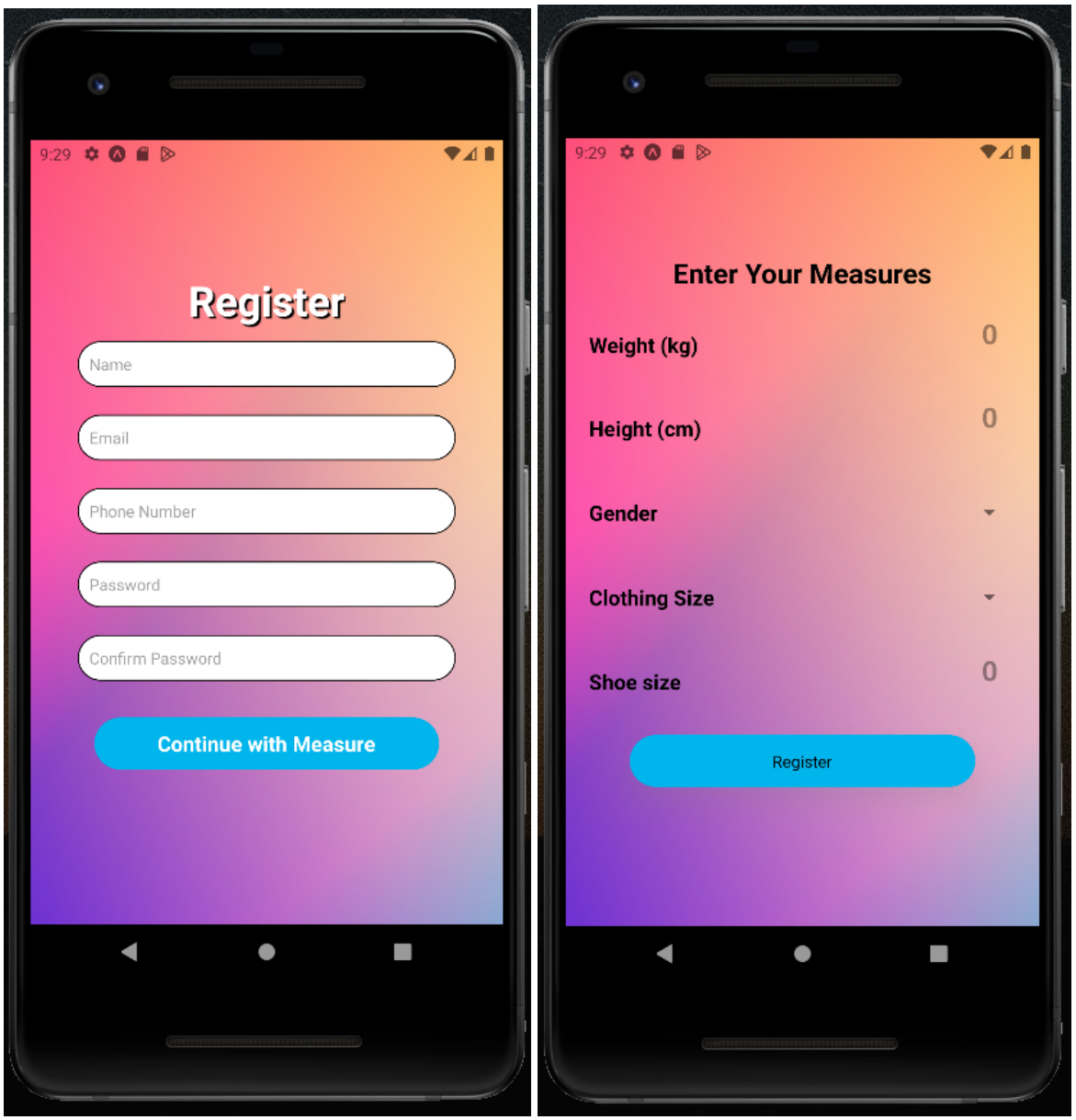
RecoModa is a fashion, promotion, and social media application. It is designed to help users search, share, follow, and stay updated on different types of fashion across many categories. With state-of-the-art visual machine learning algorithms in RecoModa, it is possible to find similar products in a few seconds with a picture reference. RecoModa offers users a powerful filtering and search feature, including color scanning technology, to decrease the time spent finding particular products. RecoModa also allows users to store their body measurements in the system to recommend products that are suitable to their preferences precisely. Recomoda aims to give small businesses and influencers the opportunity to promote their own brands by advertising many products in it. At the moment, we did not develop every functionality we determined at the start, such as presenting actual shopping in our system or gathering the Amazon and Trendyol products into our system with API due to challenges we faced during development face. However, we are looking forward to progress and fulfillment afterward.

9. User Manual

9.1 Sign-in Page

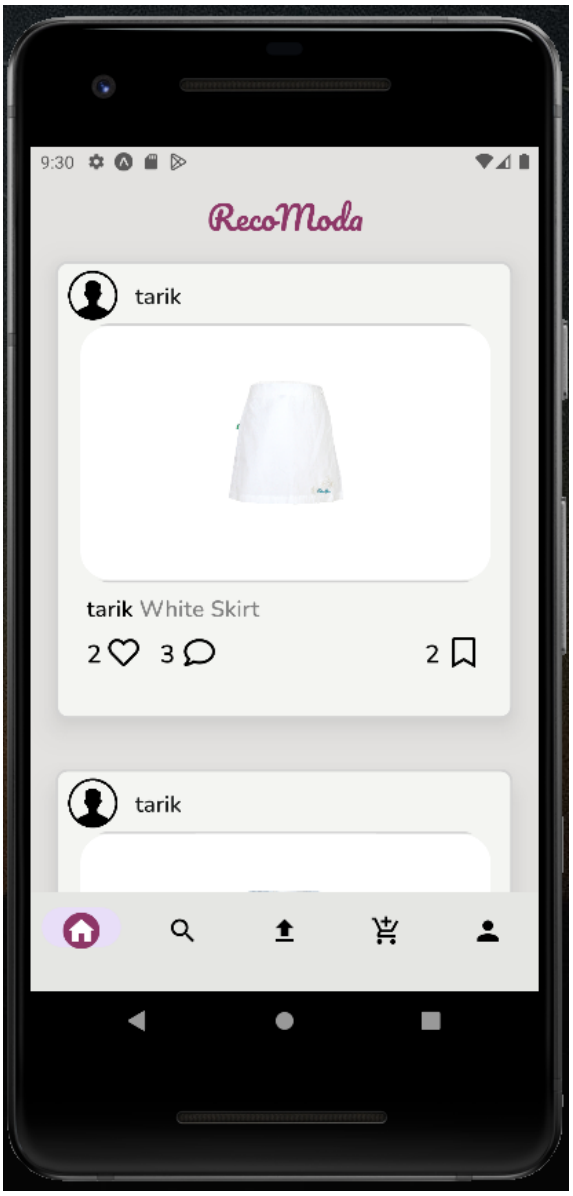
The first page that welcomes users is the sign-in page. If the user has an account, after entering their e-mail and password information, they are directed to the Home (Main Page) page by clicking the Login button. If incorrect information is entered, a warning appears and prompted to enter it again. Users who forget their passwords can change their passwords from the Forgot Password page. Users who do not have an account can create a new account with the register button.

9.2 Register Page



Users must choose an unused username to become a member, and then enter their email and phone number, respectively. Then the user must specify a password and confirm it. Afterwards, click the Continue With Measure button and enter the Weight, Height, Gender, Clothing Size and Shoe Size information, but these are not mandatory. Afterwards, registration is completed with the Register button.

9.3 Home Page

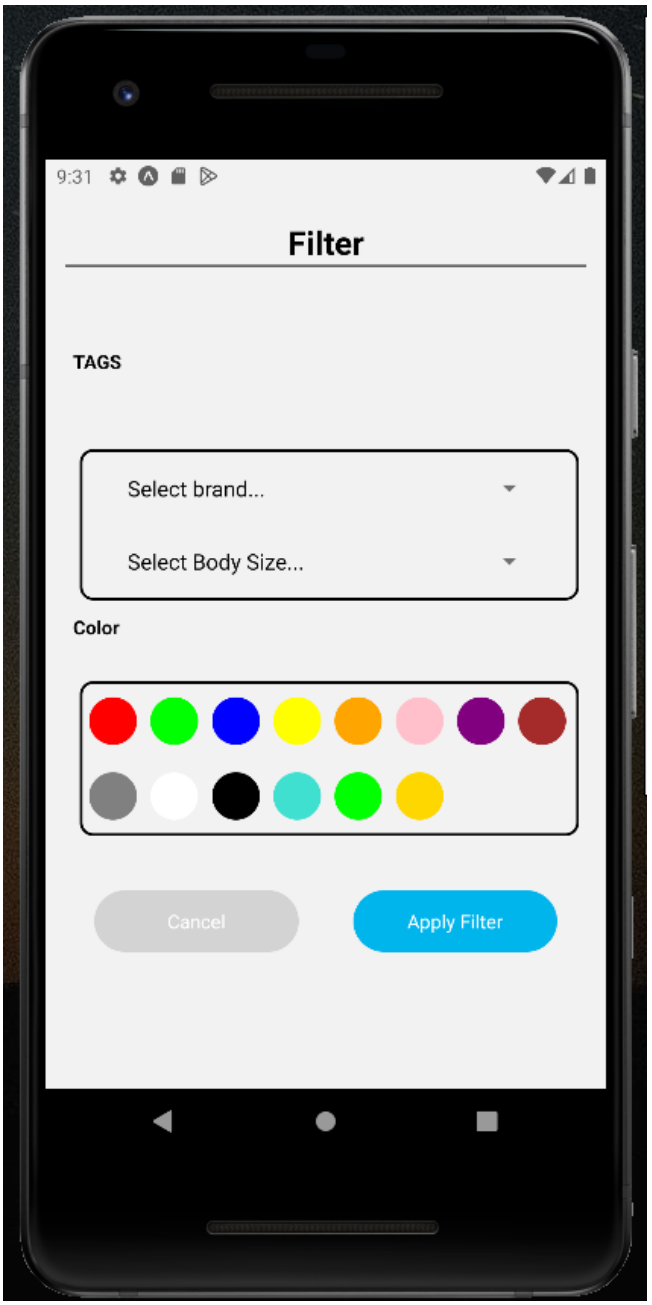


The home page is the page where the posts shared by other users that the user follows. Posts are sorted from most recent to oldest. Users can like posts by clicking the like button. By clicking the comment button, you can see the detailed post and read other comments. Also, get redirected to detailed page and there you can see similar products under the post.

9.4 Search Page

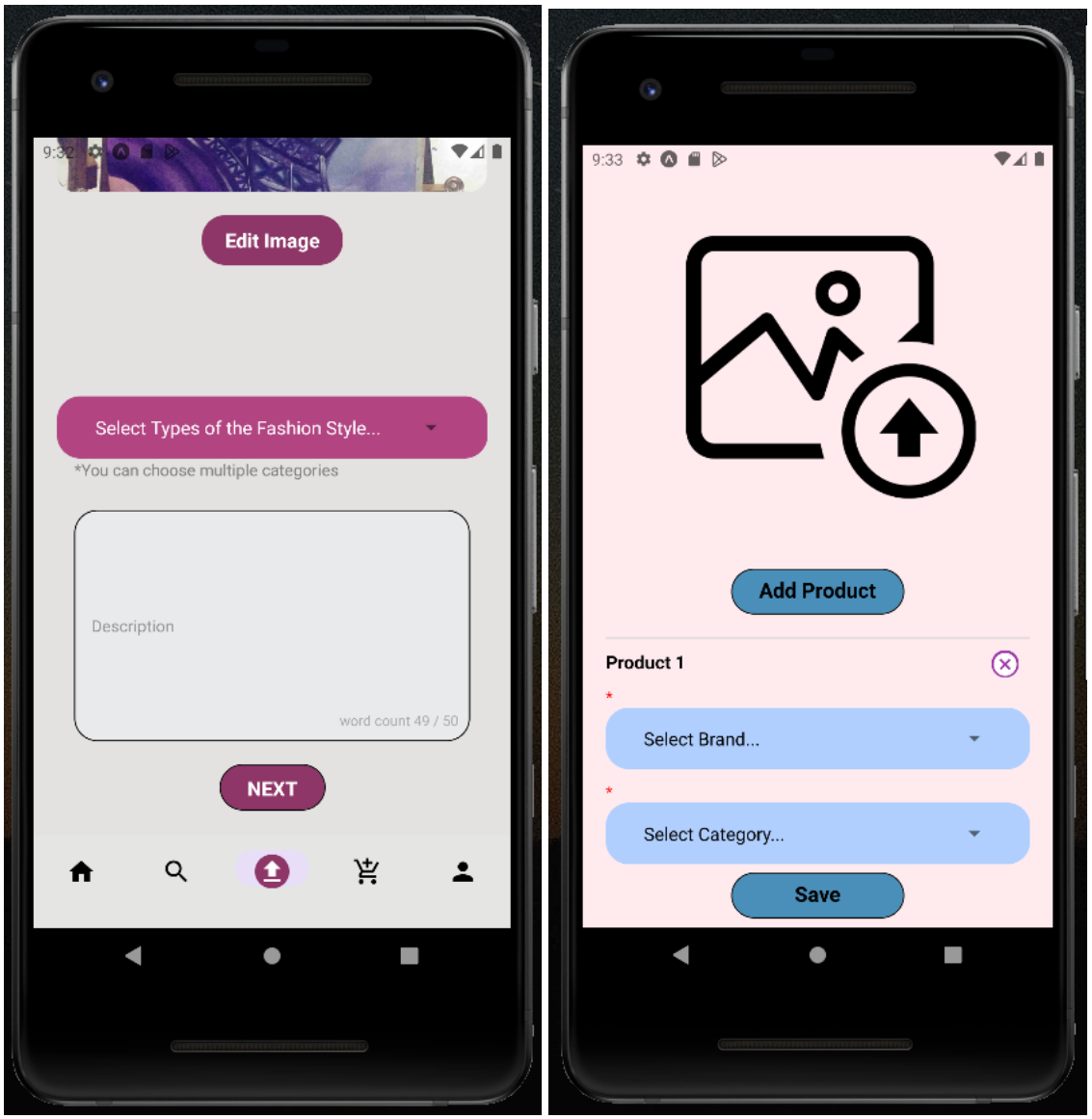
In Search Page, the user can only see the photos of the posts shared by the users they do not follow. Click on a photo to see the detailed post and similar posts to that post. You can search for products by name by typing in the Search button or click on the suggested categories and sort them. Also, the user clicks on the filter button and is directed to the Filter page, where they can set the filters and then see the appropriate posts.

9.5 Filter Page



In the Filter page, the user can select multiple brands, select multiple body sizes, and then select multiple colors. In addition, user does not have to fill the criteria they does not want. Then user can start to see the appropriate photos by clicking the Apply filter button. With the Cancel button, they cancel their selection and are directed to the previous page.

9.6 Upload Image Page

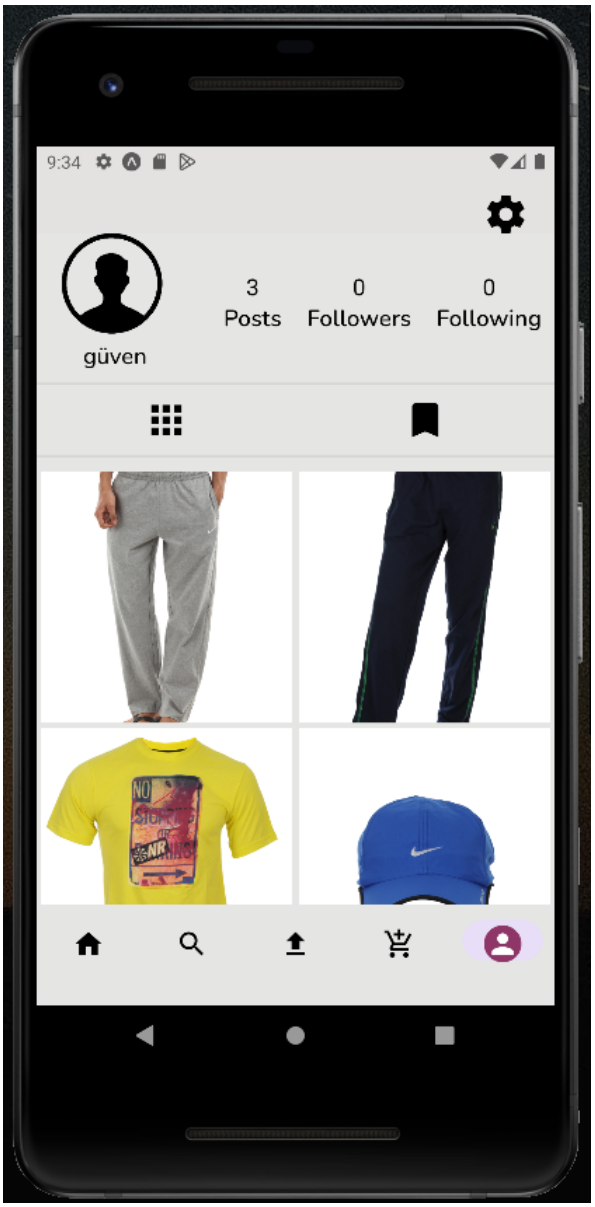


The user who navigates to the Upload page can upload any photo they want from here. The user, who first selects a photo in the edit image section, then filters according to the type of photo he has uploaded. Then continues to the next page by writing a description. On the other page, it ends the photo upload process by choosing a brand and category.

9.7 Your Wishlist Page

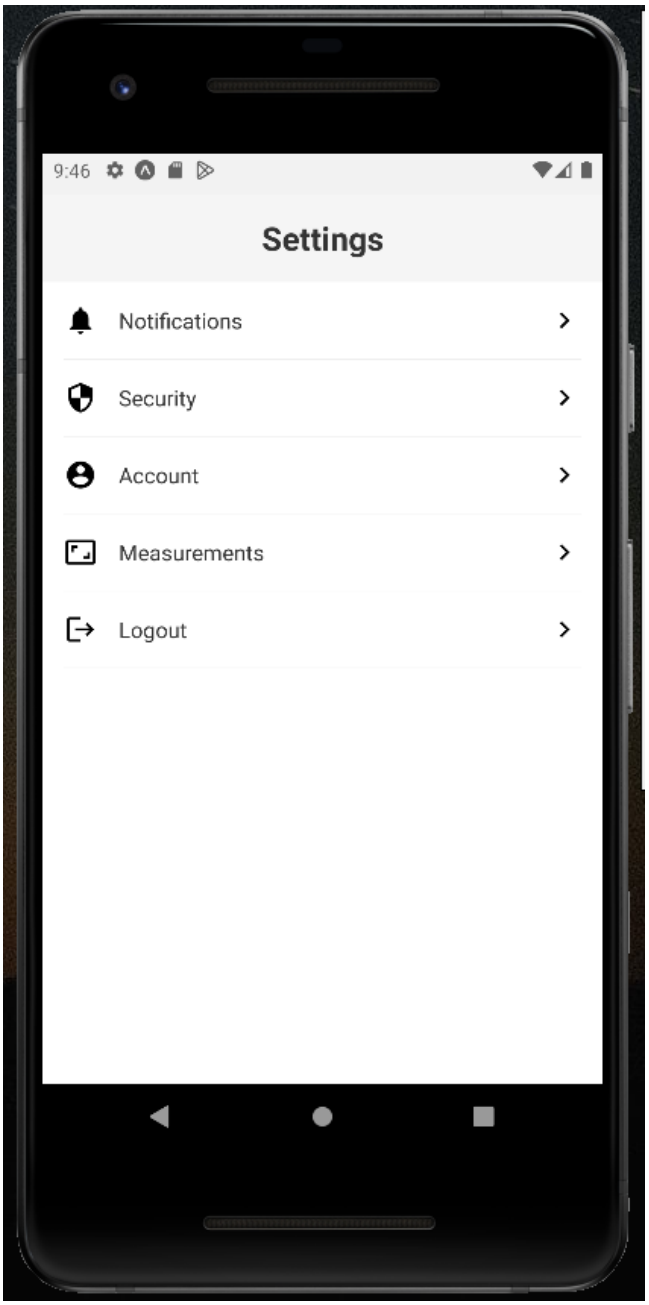
This is how the user's wishlist page is displayed. Photos saved by the user appear here. The user can remove any photo from this page later.

9.8 Profile Page



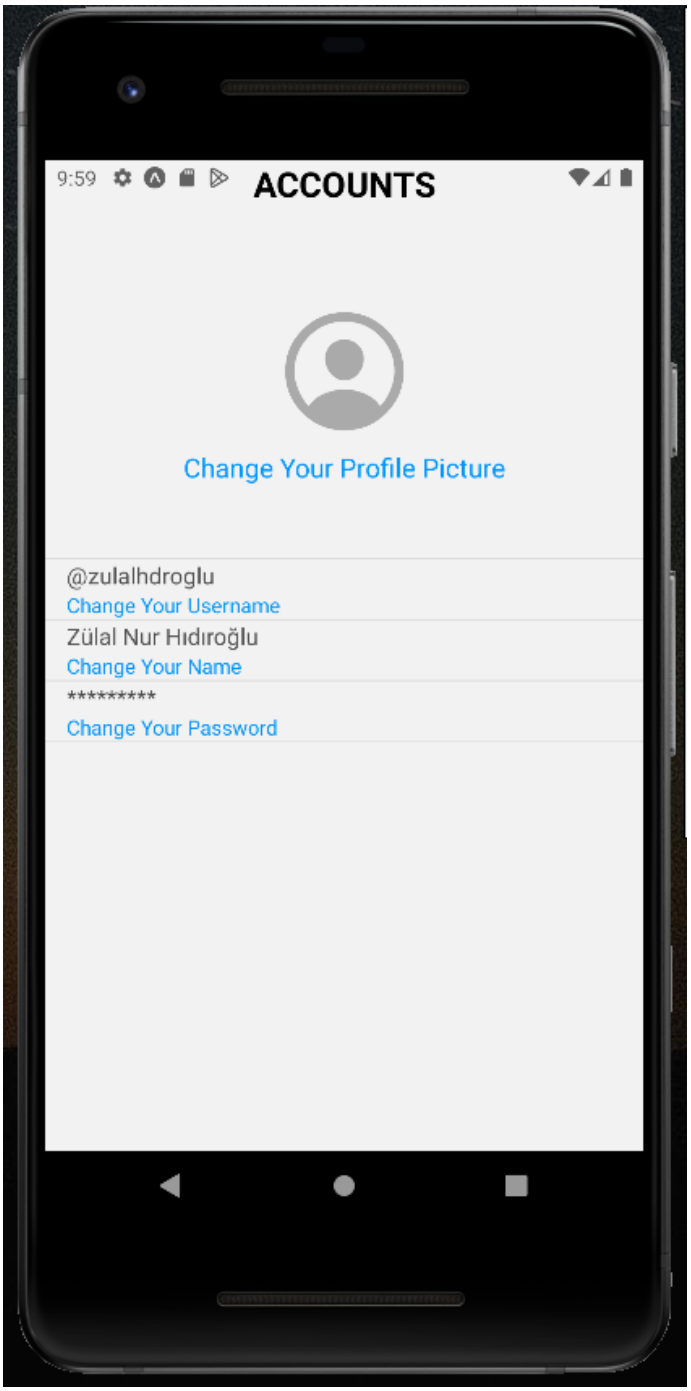
This page is the profile page. The posts found here are the photos of the user. You can click on the follower and following sections to see the users they follow or by whom they are followed. It is also written here whether it is followed by how many people. With the settings button in the upper right corner, you can go to the settings section on the next page.

9.9 Settings Page



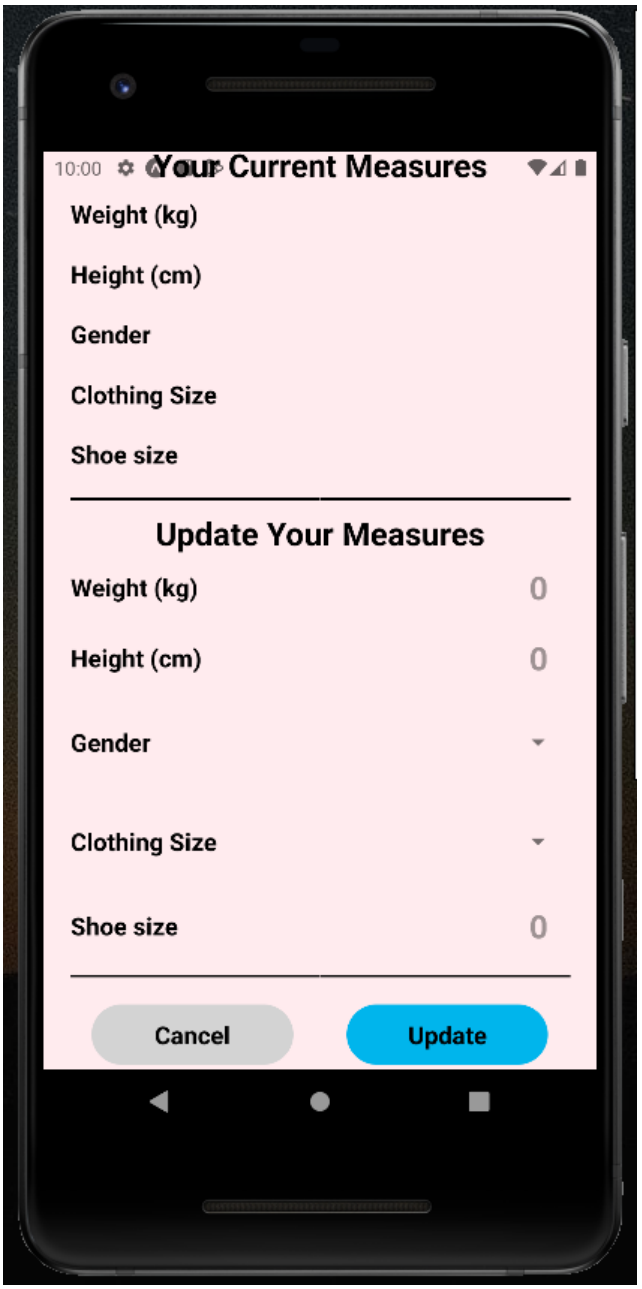
On this page, 5 different options appear before the user. The user can access and edit the information in his profile according to the options available here.You can choose which notifications will come from the application from the notifications tab at the top. here you can receive notifications of people's fortograph likes or follow-up requests.From the security tab in the second tab, arrangements can be made for the security in the application.User information can be edited from the Account tab. Here you can change your e-mail address or password.In the 4 tabs, there is a section of measures. Here, the user can enter the dimensions he has and get suggestions accordingly.

9.10 Account Page



On the account page, the user can update their current profile photo by clicking change your profile picture. It can also change user information with other change buttons. When changing the username, they cannot take the name of another user.

9.11 Measurements Page



On the Measurement page, the user enters his own measurements into the system. Here, there is an information entry according to categories such as weight, height, gender. Later, the user can edit this information if he/she wishes.

10. References

[1]Hashim, N. A., Janor, H., Sidek, F., & Nor, S. M. (2018). Online shopping: A potential of herding behavior symptom? *Business and Management Horizons*, *6*(2), 105. https://doi.org/10.5296/bmh.v6i2.14197

[2] Wikimedia Foundation. (2022, November 2). *React native*. Wikipedia. Retrieved November 13, 2022, from https://en.wikipedia.org/wiki/React_Native

[3] Wikimedia Foundation. (2022, November 2). React native. Wikipedia. Retrieved November 13, 2022, from https://en.wikipedia.org/wiki/React_Native

[4] Wikimedia Foundation. (2022, November 9). Node.js. Wikipedia. Retrieved November 13, 2022, from https://en.wikipedia.org/wiki/Node.js