Bilkent University
Department of Computer Engineering

Senior Design Project
T2330
vendAR

Detailed Design Report

22003550, Mehmet Feyyaz Küçük, feyyaz.kucuk@ug.bilkent.edu.tr
22002478, Ege Ayan, ege.ayan@ug.bilkent.edu.tr
22003271, Deniz Çelik, denizc@ug.bilkent.edu.tr
22001983, Ender Utlu, ender.utlu@ug.bilkent.edu.tr
220001422, Parsa Keihan, parsa.keihan@ug.bilkent.edu.tr

Supervisor: Uğur Güdükbay
Course Instructors: Atakan Erdem, Mert Bıçakcı

15.03.2024

1. Introduction

1.1. Purpose of the system

With the increase in the number of households in the world, the need for furniture and house effects keep increasing. The process of buying new furniture or replacing them can be tedious for most people. Most of the time, people try to pack their furniture in a way that everything barely fits in a room. But sometimes furniture might not fit as expected. Since the delivery and construction of furniture can be costly and time-consuming, seeing the furniture not fit as expected can be very frustrating. What if there was a way to see if a piece of furniture fit into a desired place, without having to buy it and deliver it to your house? vendAR is an application that lets people visualize the furniture they want to buy on their phones with the exact dimensions. It uses Augmented Reality technology with the rear camera of the mobile phones to place the furniture in their surroundings. The users can move and rotate the AR model of the furniture around the room to see if the furniture fits to their desired place. vendAR offers to solve the tediousness and frustration of purchasing furniture by giving the buyers a preview of the product as a 3D model. This report will contain information about vendAR, its constraints, professional and ethical issues, functional and non-functional requirements as well as a market analysis in detail.

1.2. Design goals

1.2.1. User Experience (UX) Design:

VendAR will offer an intuitive user interface, allowing customers to seamlessly select and visualize furniture in their own space. The design will prioritize ease of navigation, with clear calls-to-action and minimal steps to view an item in AR. The UX will be optimized for various devices, ensuring a consistent experience across platforms.

1.2.2.Technical Architecture:

The application will be built on a robust framework that supports real-time 3D rendering and AR capabilities. It will integrate with existing eCommerce platforms using APIs and leverage cloud services for scalability. The architecture will ensure low latency and high performance, crucial for real-time visualization.

1.2.3. Functionality:

Key features will include real-time furniture visualization in the user's space, customizable options for size and color, and the ability to save and share designs. The program will use advanced AR tracking and rendering technologies to provide a realistic and immersive experience.

1.2.4. Security and Privacy:

The design will incorporate end-to-end encryption for user data and adhere to privacy regulations. User interactions with the AR environment will be secure, and any data collected will be used to enhance the user experience while maintaining user confidentiality.

1.2.5. Performance Optimization:

VendAR will be optimized for fast loading times and smooth performance, even on lower-end devices. It will use efficient algorithms to reduce battery consumption and data usage without compromising the visual quality of the furniture models.

1.2.6. Scalability:

The design will account for future expansion, allowing for easy updates and the addition of new features. It will support a growing number of users and an increasing catalog of furniture and items, without degradation in performance.

1.2.7. **Accessibility:**

Accessibility features will be integrated to ensure that the program is usable by people with disabilities. This includes voice commands, screen reader support, and high-contrast visuals for better visibility.

1.3. Definitions, acronyms, and abbreviations

- **AR**: Augmented Reality - A technology that overlays digital information on the user's view of the real world.
- **UX**: User Experience - The overall experience a user has when interacting with the AR program, including ease of use and satisfaction.
- **API**: Application Programming Interface - A set of protocols for building and interacting with software applications, crucial for integrating the AR program with online platforms.
- **3D Rendering**: The process of converting 3D wireframe models into 2D images with 3D photorealistic effects.
- **SDK**: Software Development Kit - A collection of software tools that allows the creation of applications for specific platforms, like ARKit for iOS and ARCore for Android.
- **CAD**: Computer-Aided Design - The use of computer systems to aid in the creation, modification, analysis, or optimization of a design.
- **GPS**: Global Positioning System - A satellite-based navigation system used to determine the precise location of an object, which can be used in AR for location-based services.
- **HCI**: Human-Computer Interaction - The study and planned design of human and computer activities.
- **MR**: Mixed Reality - A blend of physical and digital worlds, unlocking natural and intuitive 3D human, computer, and environment interactions.

2. Current Software Architecture

- IKEA Place:
  - Allows virtual placement of any furniture item from their store.
  - Provides accurate 3D images to convey real-life sizes.
  - Enables sharing of virtual fitting results through social media.
- Houzz:
  - Offers a wide range of products and a feature called “View in My Room 3D” to visualize furniture or decor elements in space using AR.
  - Updated to place and move over 500,000 objects with improved image transfer of material texture and room lighting considerations.
- Target:
  - Features an AR symbol in the app to visualize selected items in your space.
  - Includes a “See It in Your Space” search function to find AR-enabled items.
- Wayfair:
  - Provides a “View in Room” button in the app to quickly visualize items in your home.
  - Aims to help decide if the style and color of a piece work in the room.

3. Proposed Software Architecture

3.1. Overview

In this section, the software and hardware structure of vendAR will be explained in detail. The system's different layers and modules are explained with diagrams and figures. Hardware/software mapping of vendAR is given in a diagram, presenting the used hardware/software resources and their interconnections. Persistent data management section explains our database and object structure. Access control and security will cover the precautions we took in order to protect our app and its users’ data preventing possible security threats.

3.2. Subsystem decomposition

vendAR will primarily be based on client-server architecture. The client side is the Flutter app responsible for displaying the UI and handling the routing and AR logic. It is also responsible for communicating with the server upon specific input events. The server side is the Spring Boot app responsible for storage, data flow and manipulation.

The client side will be decomposed into two subsystems (i.e. layers). These layers will be the View Layer and the Controller Layer. The View Layer is responsible for displaying the UI of our application using the Flutter framework. Each mockup (see Analysis and Requirement report, Sec. 2.5.5) we created is represented as a view component in the View Layer. The Controller Layer is responsible for handling input and sending the appropriate requests to the server side based on the received input. Every view component has its controller component associated with it that handles the input on that specific page.

The server side will also be decomposed into two subsystems. These layers will be the Logic Layer and the Data Layer. The Logic Layer is responsible for setting up endpoints for data retrieval, insertion, deletion and handling the query logic for these operations. The Data Layer is responsible for handling model classes to represent the E/R diagram we created, (see Analysis and Requirement Report, Fig. 4).

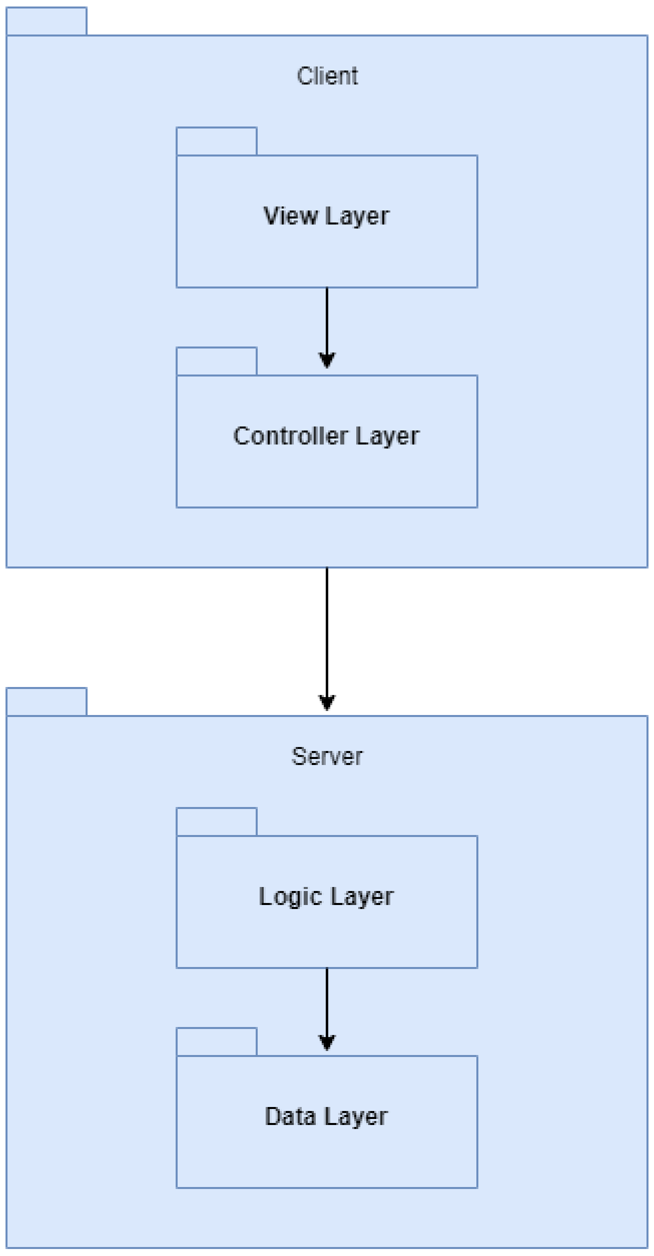

Fig. 1. Subsystem decomposition into client-server architecture

3.3. Hardware/software mapping

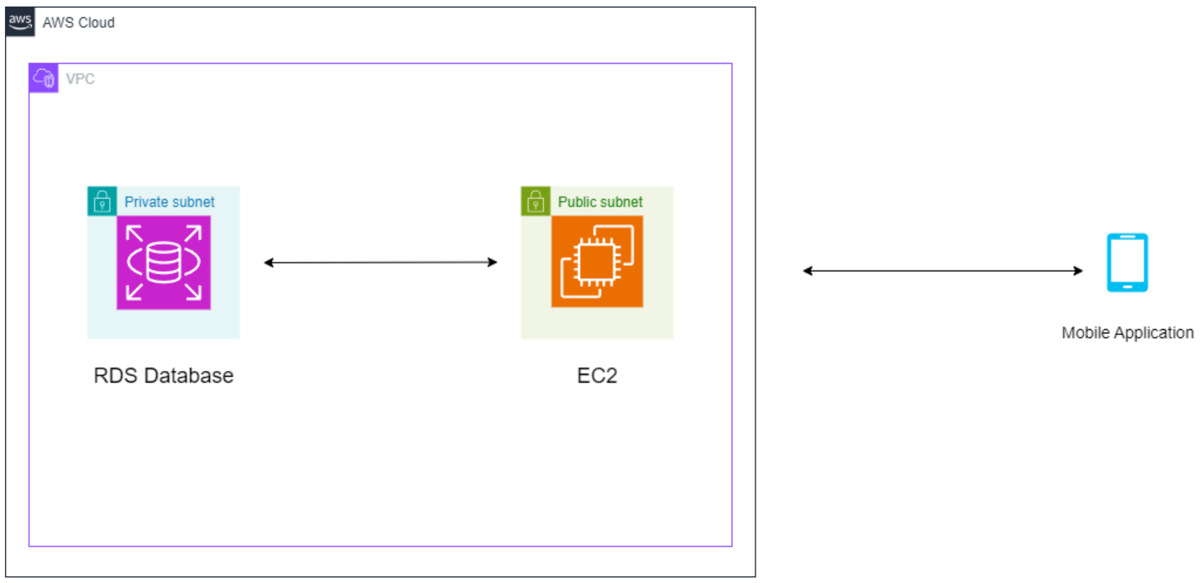

Fig. 2. Hardware/software mapping diagram

vendAR will be developed for Android and İOS with Flutter. Vendar will make use of the camera of the mobile device to present an AR service to the end users. Users will create models using the template model bases in vendAR. They will be stored in Google Drive using *NetworkImage*.

Flutter allows us to utilize the mobile device's hardware resources in order to place the models in desired locations via its camera. We will use Java Spring Boot that is deployed in the AWS EC2 machine to manage API calls to our PostgreSQL database and communicate with the database which also is hosted in AWS's RDS service. AWS machines and the client mobile device will be the hardware that is used. Flutter, spring boot and PostgreSQL will be the software parts of vendAR.

3.4. Persistent data management

vendAR needs to store user credentials, product models and product features in order to fetch and manipulate them when needed. We use GoogleDrive *NetworkImage* to store the product models. For the user credentials and the product details we use PostgreSQL in AWS cloud service. It stores our data in tables since it's a relational database and returns the queries as JSON objects to our application when requested. It is up 24/7 and we can take snapshots for backup both automatically and manually through AWS to prevent possible data losses.

3.5. Access control and security

In our application, the only sensitive information is the users' login credentials which we will hold encrypted in our database for security.

To ensure further security, we will use token-based authentication. Token-based authentication is a protocol which allows the applications to uniquely identify users and associate them with unique access tokens which hold their id in the user database. The token is invalidated when either the users log out or it expires by itself. The access tokens also lets us make requests from client to server side[2].

Also, our application requires the usage of the camera for which we will take permissions from the user's device.

4. Subsystem services

4.1. Client

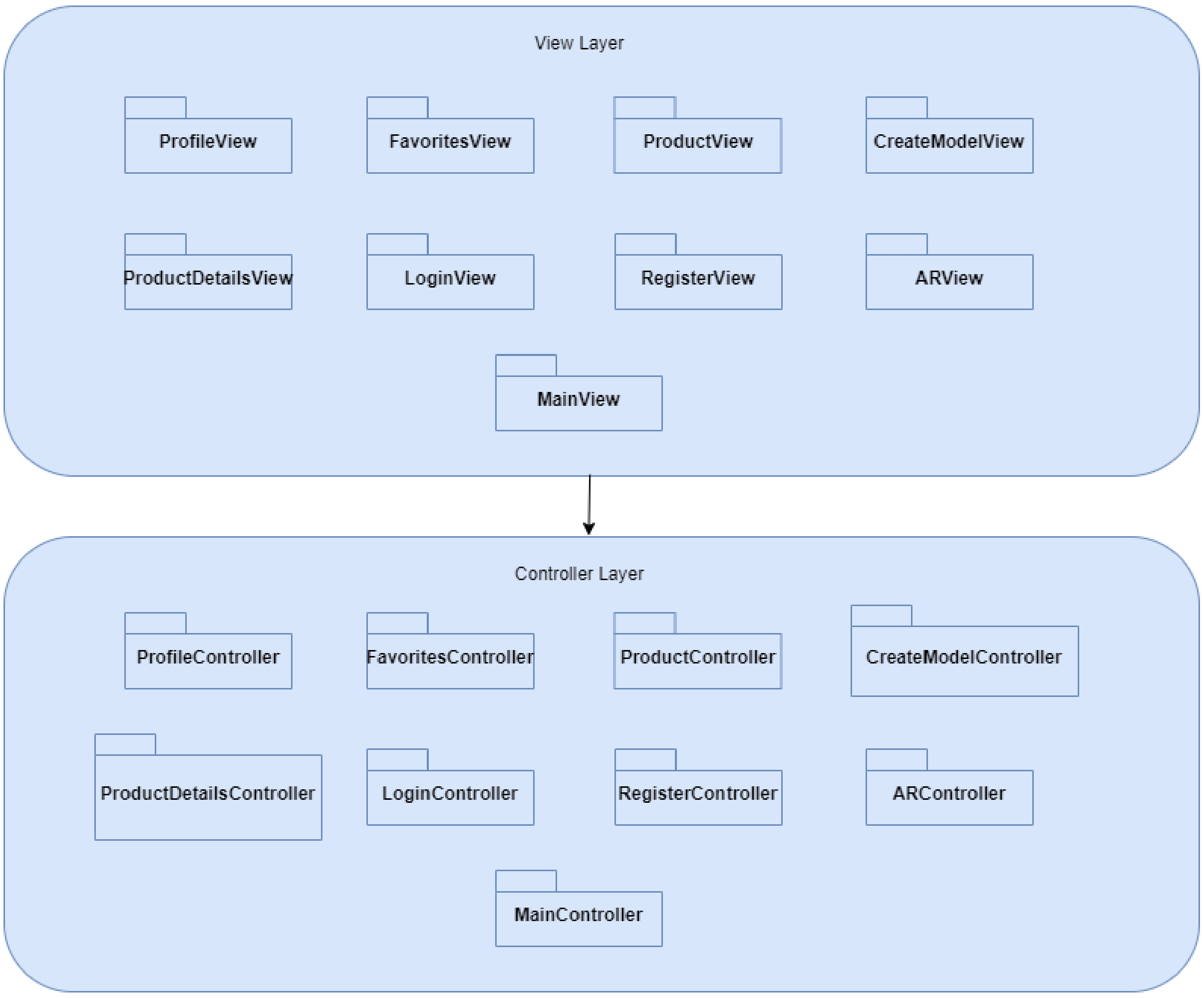

Fig. 3. Decomposition of client subsystem

4.1.1. View Layer

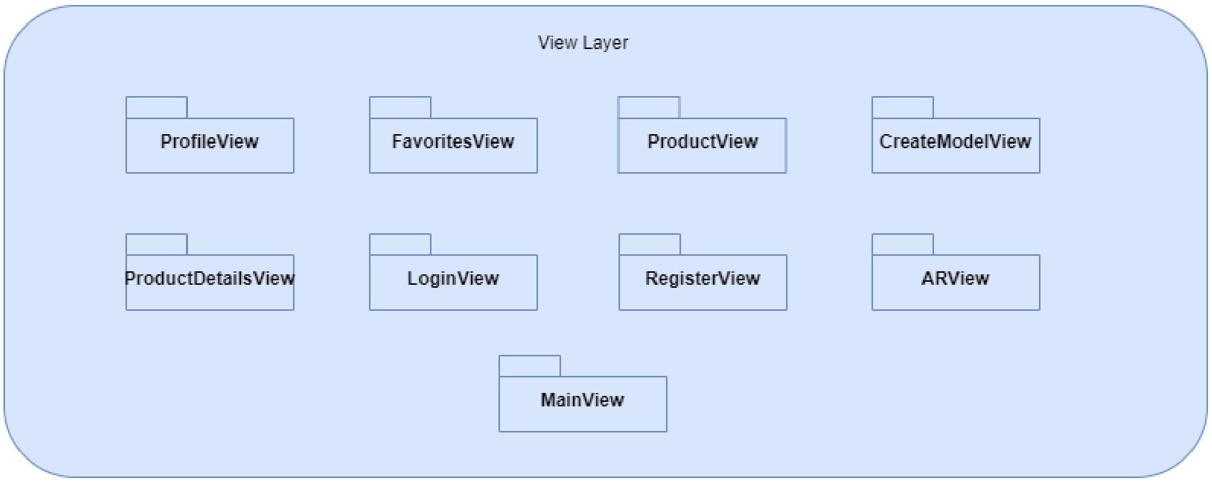

Fig. 4. View Layer diagram

➢***ProfileView:*** Responsible for rendering the profile page.
➢***FavoritesView:*** Responsible for rendering user's favorited items page.
➢***ProductView:*** Responsible for rendering a list of products after user uses the search bar.
➢***CreateModelView:*** Responsible for rendering the page for adding a new item model.
➢***ProductDetailsView:*** Responsible for rendering a specific product's details page.
➢***LoginView:*** Responsible for rendering the login page.
➢***RegisterView:*** Responsible for rendering register page.
➢***ARView:*** Responsible for rendering the page where the AR model is rendered in camera space.
➢***MainView***: Responsible for rendering the home page.

4.1.2. Controller Layer

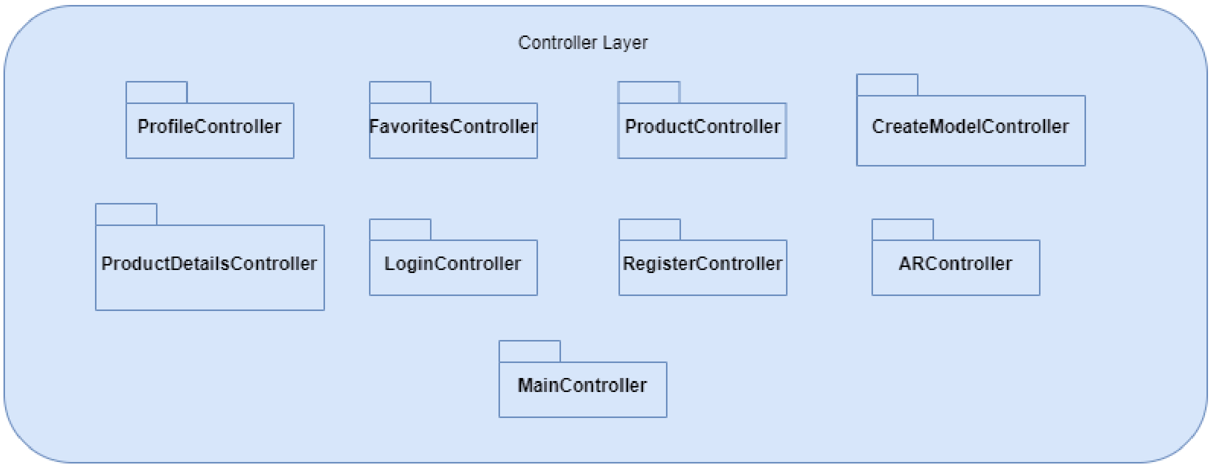

Fig. 5. Controller Layer diagram

➢***ProfileController:*** Responsible for input and request handling in the profile page.
➢***FavoritesController:*** Responsible for input and request handling in the user's favorited items page.
➢***ProductController:*** Responsible for input and request handling in the products page.
➢***CreateModelController:*** Responsible for input and request handling in the page for creating a new product model.
➢***ProductDetailsController:*** Responsible for input and request handling in the page for a specific product's details page.
➢***LoginController:*** Responsible for input and request handling in the login page.
➢***RegisterController:*** Responsible for input and request handling in the register page.
➢***ARController:*** Responsible for input and request handling in the page where the AR model is displayed in the camera space.
➢***MainController:*** Responsible for input and request handling in the home page.

4.2. Server

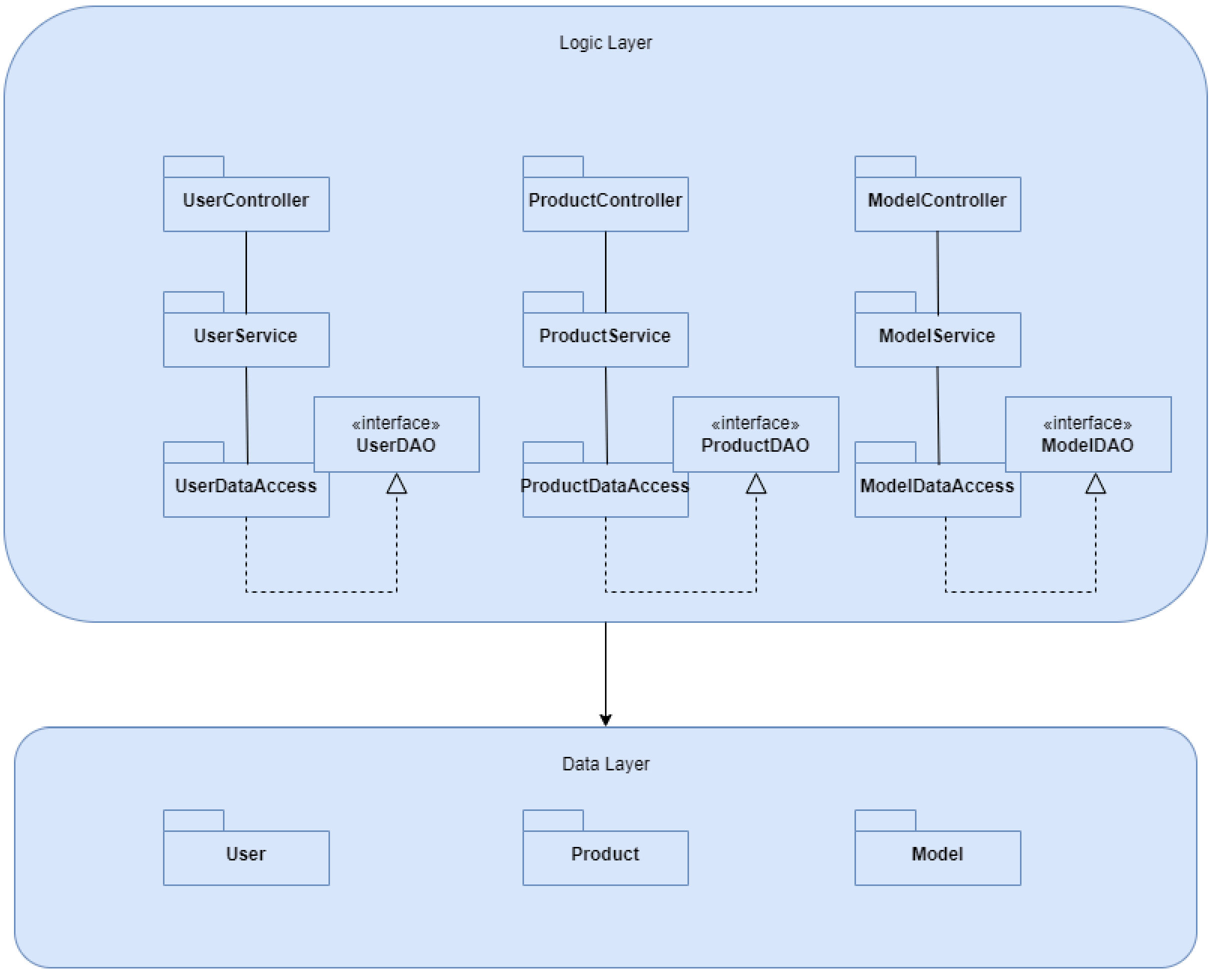

Fig. 6. Decomposition of server subsystem

4.2.1. Logic Layer

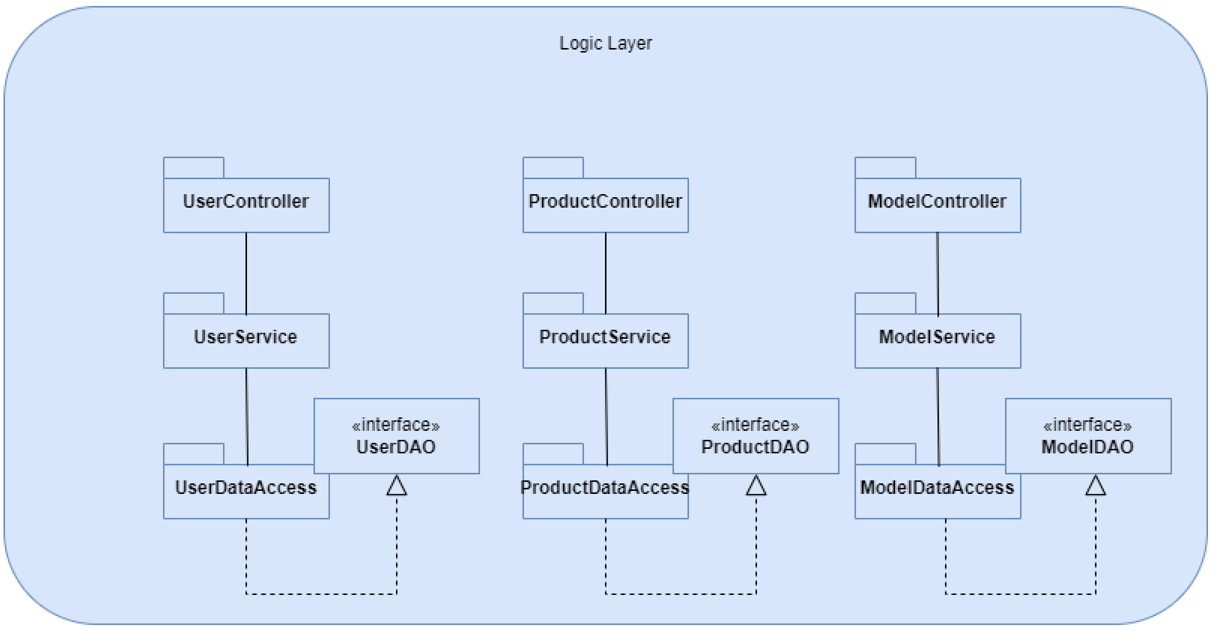

Fig. 7. Logic (Business) Layer diagram

➢***UserDAO/ProductDAO/ModelDAO:*** These interfaces contain abstract functions that correspond to query operations.

➢***UserDataAccess/ProductDataAccess/ModelDataAccess:*** These classes implement their respective DAO classes by implementing each function using actual SQL queries using *JdbcTemplate*[3].

➢***UserService/ProductService/ModelService:*** These classes contain their related DataAccess classes and use their functions to implement the desired capabilities. They constitute the business layer of the application containing most of the logic operations. Usually call some SQL query functions to perform an operation.

➢***UserController/ProductController/ModelController:*** These classes have the related Service classes and use them directly. They are the places where the API endpoints are created and the request bodies are manipulated according to the service functions. Requests enter the backend application through controllers' functions that handle the same request type.

4.2.2. Data Layer

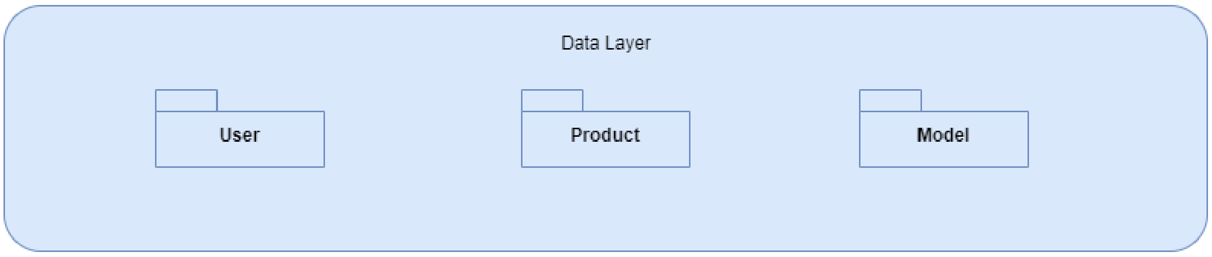

Fig. 8. Data Layer diagram

➢ ***User:*** The model class that represents the user table in our database. It has the following attributes:

id: **UUID**

name: **String**

surname: **String**

password: **String**

email: **String**

phoneNumber: **String**

➢ ***Product:*** The model class that represents the product table in our database. It has the following attributes:

id: **UUID**

userId: **UUID** (since product entity has FK for user entity)

title: **String**

description: **String**

images: **String[]**

features: **String[]**

➢ ***Model:*** The model class that represent the model table in our database. It has the following attributes:

id: **UUID**

productId: **UUID** (since model entity has FK for product entity)

dimensions: **float[]**

src: **String**

5. Test Cases

5.1 Functional Test Cases

Test Case 1: TC_Login_01

Description: To test if a successful login can be made and the home page can be reached with the account.

Preconditions: Login Page is launched. There is an account to be used for the test.

Test Steps:

1. Enter the account information.
2. Press the confirmation button.

Expected Result: Login will be successful, the home page will be launched.

Test Case 2: TC_Login_02

Description: To test if a successful login can be made without filling both fields.

Preconditions: Login page is launched.

Test Steps:

1. Enter the account information for one of the fields or don’t enter at all.
2. Press the confirmation button.

Expected Result: Login will be failed. Error message will be displayed.

Test Case 3: TC_Login_03

Description: To test if a successful login can be made with incorrect account information.

Preconditions: Login page is launched.

Test Steps:

1. Enter the wrong account information for one of the fields or both fields.
2. Press the confirmation button.

Expected Result: Login will be failed. Error message will be displayed.

Test Case 4: TC_Login_04

Description: To test if a successful login can be made using the Sign in with Google option.

Preconditions: Login page is launched. There is a Google account that is logged in on the phone.

Test Steps:

1. Press the Sign in with Google button.
2. Select the Google account and give the necessary permissions Google asks for.

Expected Result: Login will be successful after the Google account is verified.

Test Case 5: TC_Login_05

Description: To test if a successful login can be made using the Sign in with Apple option.

Preconditions: Login page is launched. There is an Apple account that is logged in on the phone. The phone is an IOS phone.

Test Steps:

1. Press the Sign in with Apple button.
2. Select the Apple account and give the necessary permissions Apple asks for.

Expected Result: Login will be successful after the Apple account is verified.

Test Case 6: TC_SignUp_01

Description: To test if a successful sign up can be made by creating an account.

Preconditions: Login page is launched. The email that will be used for new account creation isn’t already used in an already existing account.

Test Steps:

1. Press the Sign Up button.
2. Enter the account information.
3. Press the confirm button.
4. After the Login Page is launched enter the account information to login
5. Press the confirm button.

Expected Result: A new account will be created and login will be successful.

Test Case 7: TC_SignUp_02

Description: To test if a successful sign up can be made by creating an account with an already used email.

Preconditions: Login page is launched. The email that will be used for new account creation is already used in an already existing account.

Test Steps:

1. Press the Sign Up button.
2. Enter the account information.
3. Press the confirm button.

Expected Result: A new account won't be created and an error message will be displayed for the email field saying the email is already used.

Test Case 8: TC_SignUp_03

Description: To test if a successful sign up can be made by creating an account when the name or the surname fields are empty.

Preconditions: Login page is launched. The email that will be used for new account creation isn't already used in an already existing account. A unique phone number is used for the testing.

Test Steps:

1. Press the Sign Up button.
2. Enter the account information leaving one or both of the name fields empty.
3. Press the confirm button.

Expected Result: A new account won't be created and an error message for the empty field will be displayed.

Test Case 9: TC_SignUp_04

Description: To test if a successful sign up can be made by creating an account when the phone number field is empty.

Preconditions: Login page is launched. The email that will be used for new account creation isn't already used in an already existing account. A unique name and surname is used for the testing.

Test Steps:

1. Press the Sign Up button.
2. Enter the account information leaving the phone number field empty.
3. Press the confirm button.

Expected Result: A new account won't be created and an error message for the empty field will be displayed.

Test Case 10: TC_SignUp_05

Description: To test if a successful sign up can be made by creating an account when the phone number of the account is used before.

Preconditions: Login page is launched. The email that will be used for new account creation isn't already used in an already existing account. The phone number that will be used is already used in another account.

Test Steps:

1. Press the Sign Up button.
2. Enter the account information using the phone number that was already used for another account.
3. Press the confirm button.

Expected Result: A new account won't be created and an error message for the phone number field will be displayed saying the number is already used.

Test Case 11: TC_HomePage_01

Description: To test if the Home Page can be used to interact with the categories.

Preconditions: A successful login has been made. The Home Page is launched. There are some products created that have specified categories.

Test Steps:

1. Select a category from the top part of the Home Page
2. Press on the selected category panel.

Expected Result: The items within the category will be displayed on the page.

Test Case 12: TC_HomePage_02

Description: To test if the Home Page can be used to interact with the products.

Preconditions: A successful login has been made. The Home Page is launched. There are some products created.

Test Steps:

1. Select a product from the top part of the Home Page
2. Press on the selected products panel.

Expected Result: The product’s page will be launched and the information of the product will be displayed.

Test Case 13: TC_HomePage_03

Description: To test if the Home Page can be used to open the Product Creation Page.

Preconditions: A successful login has been made. The Home Page is launched.

Test Steps:

1. Press the Product Creation button. (Bottom right button with the plus sign)

Expected Result: The Product Creation page will be launched.

Test Case 14: TC_HomePage_04

Description: To test if the Home Page can be used to use the search function.

Preconditions: A successful login has been made. The Home Page is launched. There some products created so that they can be searched.

Test Steps:

1. Press the search icon on the top right.
2. Write existing tags or product names that already exist so they can be searched.
3. Press the confirm button to finalize the search.

Expected Result: The products that have the most relevance with the searched terms will be displayed.

Test Case 15: TC_HomePage_05

Description: To test if the Home Page can be used to toggle dark mode.

Preconditions: A successful login has been made. The Home Page is launched.

Test Steps:

1. Press the kebab menu (3 dots) on the top right.
2. Press the dark mode switch on the drop down menu that is displayed.

Expected Result: The dark mode will be activated or deactivated depending on the mode prior to the change.

Test Case 16: TC_HomePage_06

Description: To test if the Home Page can be used to change language.

Preconditions: A successful login has been made. The Home Page is launched.

Test Steps:

1. Press the kebab menu (3 dots) on the top right.
2. Press the language button on the drop down menu that is displayed.
3. Pick the language you want from the drop down menu that is displayed.

Expected Result: Then language selected will be used after the app reloads itself after the setting change.

Test Case 17: TC_HomePage_07

Description: To test if the Home Page can be used to view Terms of Service and Privacy Policy.

Preconditions: A successful login has been made. The Home Page is launched.

Test Steps:

1. Press the kebab menu (3 dots) on the top right.
2. Press the Terms of Service or Privacy Policy button on the drop down menu that is displayed.

Expected Result: The page for the selected option will be displayed.

Test Case 18: TC_HomePage_08

Description: To test if the Home Page can be used to traverse the application using the navigation bar at the bottom.

Preconditions: A successful login has been made. The Home Page is launched.

Test Steps:

1. Press an option from the navigation menu.
2. Press the option that wasn’t pressed on the previous step.
3. Press on the home option.

Expected Result: The Profile and Favorites pages will be displayed in the order they are pressed with all the user’s information. After that the Home Page will be displayed after step 3.

Test Case 19: TC_ProductCreation_01

Description: To test if the Product Creation can be successfully completed.

Preconditions: A successful login has been made. The Product Creation Page is launched from the Home Page.

Test Steps:

1. Fill out all the information for the product.
2. Import a model or select a template.
3. Press the confirm button to complete the creation.

Expected Result: After the confirmation button is pressed, the Profile page will be launched with the product visible under the My Models section. Another account can be used to see if the creation was successful also by using the search function.

Test Case 20: TC_ProductCreation_02

Description: To test if the Product Creation can be successfully completed without all the fields completely filled out.

Preconditions: A successful login has been made. The Product Creation Page is launched from the Home Page.

Test Steps:

1. Fill out all the information for the product except one field. This can be any field from the product name, price, images or the model.
2. Press the confirm button to complete the creation.

Expected Result: After the confirmation button is pressed, there will be two different results depending on the field that was left empty. If the images or product features are left empty, the product creation will be completed successfully, albeit with missing information, which can be seen under the My Models section . If any of the other fields are left empty, error messages will be displayed saying the field should be filled.

Test Case 21: TC_ProductCreation_03

Description: To test if the Product Creation can be successfully canceled.

Preconditions: A successful login has been made. The Product Creation Page is launched from the Home Page.

Test Steps:

1. Fill out some fields of the page.
2. Press the cross on the top left of the Product Creation Page.

Expected Result: The Home Page will be launched with none of the information saved.

Test Case 22: TC_ProductPage_01

Description: To test if the Product Model can be successfully viewed from the Product page and if it can be viewed in AR and in 3D.

Preconditions: A successful login has been made. The Home Page is launched. There is at least one product that exists.

Test Steps:

1. Select and press on a product.
2. In the product page press on the View Model button.
3. After the model page is launched press on the model switch to view in AR.
4. Press the cross on the top left to return to the product page.

Expected Result: After step 2, the 3D model will be displayed. After step 3, the AR Model will be displayed using the phone’s camera. The model page will be exited with step 4.

Test Case 23: TC_ProductPage_02

Description: To test if the product can be bought by using the Order Here button.

Preconditions: A successful login has been made. The Home Page is launched. There is at least one product that exists.

Test Steps:

1. Select and press on a product.
2. In the product page press on the Order Here button.
3. Choose the browser you will use from the menu that will be displayed.

Expected Result: The product will be displayed with the shopping site link provided by the seller on the selected browser.

Test Case 24: TC_ProductPage_03

Description: To test if the product can be marked as favorite.

Preconditions: A successful login has been made. The Home Page is launched. There is at least one product that exists.

Test Steps:

1. Select and press on a product.
2. In the product page press on the star button on the top right.

Expected Result: The product will be displayed on the favorites page.

Test Case 25: TC_ProductPage_04

Description: To test if the Product Page can be exited.

Preconditions: A successful login has been made. The Home Page is launched. There is at least one product that exists.

Test Steps:

1. Select and press on a product.
2. In the product page press on the cross on the top left.

Expected Result: The page that was used to open the product page will be launched. This could be the Home Page, the Favorites Page or the Profile Page.

Test Case 26: TC_FavoritesPage_01

Description: To test if the Favorites Page can be used for opening a Product.

Preconditions: A successful login has been made. The Favorites Page is launched. There is at least one product that is added to the favorites.

Test Steps:

1. Select and press on a product.
2. In the product page press on the cross on the top left.

Expected Result: The favorite product’s Product Page will be launched. After step 2 the Favorites Page will be relaunched.

Test Case 27: TC_FavoritesPage_02

Description: To test if the Favorites Page can be used for opening a Product to unfavorite the product.

Preconditions: A successful login has been made. The Favorites Page is launched. There is at least one product that is added to the favorites.

Test Steps:

1. Select and press on a product.
2. In the product page press on the star icon on the top right.
3. In the product page press on the cross on the top left.

Expected Result: The favorite product’s Product Page will be launched. After step 2, the favorite will be removed. After step 3, the product won’t be visible on the Favorite Page that was launched.

Test Case 28: TC_ProfilePage_01

Description: To test if the Profile Page can be used for editing account information.

Preconditions: A successful login has been made. The Profile Page is launched. The new information for the email and phone number is unique.

Test Steps:

1. Select an information field to change.
2. Press the edit button and update the information.

Expected Result: The profile information will be updated permanently.

Test Case 29: TC_ProfilePage_02

Description: To test if the Profile Page can be used for editing account information that is used in other accounts.

Preconditions: A successful login has been made. The Profile Page is launched. There is another account that uses the updated information.

Test Steps:

1. Select an information field to change.
2. Press the edit button and update the information.

Expected Result: The profile information won’t be updated and an error message will be displayed saying the information is in use.

Test Case 30: TC_ProfilePage_03

Description: To test if the Profile Page can be used changing the password.

Preconditions: A successful login has been made. The Profile Page is launched.

Test Steps:

1. Press the Change Password button.
2. Enter your old password.
3. Enter your new password.
4. Enter your new password again and press confirm.

Expected Result: The password will be updated if it is strong enough. An error message will be displayed if the password isn't strong enough, explaining what to change.

Test Case 31: TC_ProfilePage_04

Description: To test if the Profile Page can be used for removing a product.

Preconditions: A successful login has been made. The Profile Page is launched. User has a product they created.

Test Steps:

1. Press on the desired product from the My Models section.
2. Press the Remove Model button.

Expected Result: The model will be removed and won't be visible under the My Models section.

5.2 Non-Functional Test Cases

Test Case 32: TC_Usability_01

Description: To test the ease of installation from the mobile application store.

Preconditions: The application is available on the mobile application store.

Test Steps:

1. Search for the application on the mobile application store.
2. Download and install the application.

Expected Result: The installation process should be straightforward and completed without errors.

Test Case 33: TC_Usability_02

Description: To test the user interface's friendliness for users of all ages.

Preconditions: The application is launched.

Test Steps:

1. Use the application interface across different age groups, including children and elderly users.
2. Refer to the user manual that will be provided for guidance.

Expected Result: Users of all ages should find the interface intuitive and easy to navigate, with clear instructions provided in the user manual.

Test Case 34: TC_Usability_03

Description: To test the clarity of the application concept for non-professionals.

Preconditions: The application is launched.

Test Steps:

1. Present the application concept to individuals not familiar with the field.
2. Observe their understanding and gather feedback.

Expected Result: Non-professional users should grasp the application's concept easily without extensive explanation.

Test Case 35: TC_Reliability_01

Description: To test the application's stability during installation and running.

Preconditions: The application is installed and running.

Test Steps:

1. Monitor for any crashes or unexpected terminations during installation.
2. Continuously use the application and monitor for any crashes during runtime.

Expected Result: The application should not crash or terminate unexpectedly during installation or usage.

Test Case 36: TC_Reliability_02

Description: To test the responsiveness of the application during camera usage.

Preconditions: The application is running, and the camera feature is accessed.

Test Steps:

1. Utilize the camera feature extensively within the application.
2. Observe for any delays or lags in camera responsiveness.

Expected Result: The application should maintain smooth camera functionality without any noticeable lags or delays.

Test Case 37: TC_Reliability_03

Description: To test the application's compatibility across mobile-based operating systems.

Preconditions: The application is installed on different mobile-based operating systems.

Test Steps:

1. Run the application on various mobile-based operating systems.

Expected Result: The application should run smoothly without any crashes across different mobile-based operating systems.

Test Case 38: TC_Performance_01

Description: To test the efficiency of resource allocation and execution time.

Preconditions: The application is running.

Test Steps:

1. Monitor the application's resource usage and execution time during various tasks.

Expected Result: The application should effectively utilize system resources and execute tasks within a reasonable time frame.

Test Case 39: TC_Performance_02

Description: To test the application's compatibility with older generation smartphones.

Preconditions: The application is installed on older generation smartphones.

Test Steps:

1. Run the application on older generation smartphones with camera functionality.

Expected Result: The application should run smoothly on older generation smartphones without compromising performance.

Test Case 40: TC_Performance_03

Description: To test the speed of data transactions within the application.

Preconditions: The application is running and performing data transactions.

Test Steps:

1. Measure the time taken for various data transactions within the application.

Expected Result: Data transactions within the application should be fast and not hinder overall performance.

Test Case 41: TC_PrivacySecurity_01

Description: To test the compliance with PDPA regarding data collection.

Preconditions: The application is running and collecting user data.

Test Steps:

1. Review the data collected by the application and compare it with PDPA guidelines.

Expected Result: The application should only collect data as needed and comply with PDPA regulations.

Test Case 42: TC_PrivacySecurity_02

Description: To test the camera usage privacy controls.

Preconditions: The application is running and accessing the device's camera.

Test Steps:

1. Monitor camera usage and permissions granted by the application.

Expected Result: The application should only use the device's camera as needed and respect user privacy settings.

Test Case 43: TC_PrivacySecurity_03

Description: To test the privacy of saved images or videos.

Preconditions: The application is running and saving images or videos.

Test Steps:

1. Review the storage location and access permissions for saved images or videos.

Expected Result: Images or videos should be saved privately in the database as needed and inaccessible to unauthorized users.

Test Case 44: TC_Scalability_01

Description: To test the application's ability to handle concurrent user access.

Preconditions: The application is running with multiple users accessing it simultaneously.

Test Steps:

1. Simulate high user traffic accessing the application concurrently.

Expected Result: The application should remain responsive and stable even with a large number of simultaneous users.

6. Consideration Of Various Factors In Engineering Design

6.1. Constraints

Implementation Constraints

The implementation will be performed on two different platforms of IOS and Android using ARCore library of Google for Flutter as a wrapper class in order to run Augmented Reality Technology on our Flutter application. So having a built-in library for that utility for both platforms is very pivotal since it would be very difficult to implement the project without it. We also need a proper hosting service for the backend of our application. For the frontend side there won't be any necessary cloud hosting since the application will run on the clients' mobile devices.

Economic Constraints

Our project depends mainly on free resources and libraries for the software side. For the cloud services that we will use to host our backend application, we might have a monthly outcome. However, due to not having a very large user base and demand initially, these costs will likely be neglicible.

Ethical Constraints

It is important for our project to not violate any software license right through pirated content or any other illegal method. Also, the program should be easy to learn and understand by a vast range of users with various backgrounds. It should not have content that is discriminatory to some groups with different religion, nation, ethnicity or culture

6.2. Standards

We use IEEE 830 and UML 2.4.1 standards standards to ensure quality in all of our reports, diagrams and deliverables. We also use semantic commit comments for our Github commits to make the version control more traceable and readable [1]. If we see an issue related with the current version, we can trace back easily thanks to our commit standards and ease the process. By sticking with the IEEE, UML and Github semantic commit standards, we can ensure the quality and efficiency in our work.

7. Teamwork Details

7.1. Contributing and functioning effectively on the team

We have shared the workload among ourselves such that each team member can work with the part that he is interested in and can bring the most contribution to the team and to the project. While assigning tasks and forming teams which are front-end, back-end and AR, we seek the past experiences such as internships and part-time working experiences of each team member. Ege has hands-on experience with the front-end development through various projects, so he takes the lead for the mobile application development side of the project. For the AR implementation and its integration with the application, Deniz is responsible since he took courses related to computer graphics before and is familiar with 3D modeling with Blender. For the backend implementation, Feyyaz, the leader of our team, also takes the lead with back-end development due to his past experiences along with Ender. Ender is also responsible for creating and maintaining the hosting environment through AWS due to the cloud computing course he is taking. Parsa is responsible for ensuring Android and IOS compatibility of our application along with testing. By having this kind of sharing of workload, we get the most from each team member to our project which increases our overall efficiency.

7.2. Helping creating a collaborative and inclusive environment

Despite the fact that all of us have our individual tasks, we never hesitate to ask each other for help. Ege, a frontend developer, might assist Feyyaz and Ender with some issues related with the backend development or database queries if he can. Deniz, originally responsible for AR, might contribute to the overall architecture of the software as well as designing how the API calls will take place in the middleware. Parsa might also assist with the front-end development by contributing to creation of some pages whenever needed. Ender can also require help from Ege who is also familiar with hosting services. Since all of us have knowledge about multiple concepts, we can easily get assistance from each other. Since we have created a very inclusive environment from the beginning, we can ask for help from each other without any hesitation. By helping to create a more collaborative and inclusive environment, we contribute not only to our tasks but also each other’s task to improve the quality of the project and efficiency.

7.3. Taking lead role and sharing leadership on the team

Even though our team leader is Feyyaz, each of us can take the lead whenever necessary for some specific topic or issue. A democratic, inclusive environment is ensured whenever we need to make a decision about a very general concept that concerns all aspects of the project. If the issue is too specific, anyone who has the knowledge and experience about it can lead the team through the process of handling it. Our team leader, Feyyaz, is responsible for the delivery of reports, assigning meetings and process tracking of each member where each member can lead the subtasks and take responsibility for it. Overall, the leadership is shared among the team and each member is capable of leading the whole team whenever it is necessary.

8. Glossary

UML: Unified Modeling Language

AWS: Amazon Web Services

AR: Augmented Reality

DAO: Data Access Object

9. References

[1] https://gist.github.com/joshbuchea/6f47e86d2510bce28f8e7f42ae84c716
[2] https://www.okta.com/identity-101/what-is-token-based-authentication/
[3] https://www.baeldung.com/spring-jdbc-jdbctemplate