# COMSATS University Islamabad Abbottabad, Pakistan

**EASY BUY WITH 3D AUGMENTED REALITY**

***By***

**Muhammad Akhlaq CIIT/SP21-BSE-056/ATD Huzaifa Sajjad CIIT/SP21-BSE-012/ATD Rimsha Muneer CIIT/SP21-BSE-082/ATD**

***Supervisor***

**Mr. Umair Mujtaba**

***Bachelor of Science in Computer Science (2021-2025)***

**The candidate confirms that the work submitted is their own and appropriate credit has been given where reference has been made to the work of others**.

.

# COMSATS University, Islamabad Abbottabad Campus

**EASY BUY WITH 3D AUGMENTED REALITY**

## A project presented to

**COMSATS University Islamabad, Pakistan**

## In partial fulfillment

**of the requirement for the degree of**

***Bachelor of Science in Computer Science (2021-2025)***

## By

**Muhammad Akhlaq CIIT/SP21-BSE-056/ATD Huzaifa Sajjad CIIT/SP21-BSE-012/ATD Rimsha Muneer CIIT/SP21-BSE-082/ATD**

i

**DECLARATION**

We hereby declare that this software, neither whole nor as a part, has been copied out from any source. It is further declared that we have developed this software and accompanied the report entirely based on our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other. We will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

**Muhammad Akhlaq Huzaifa Sajjad Rimsha Muneer**

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (SE) “Easy Buy with 3D Augmented Reality” was developed by **Muhammad Akhlaq(CIIT/SP21-BSE-056), Huzaifa Sajjad (CIIT/SP21-BSE-012), and Rimsha Muneer (CIIT/SP21-BSE-082),** under the supervision of “Mr. Umair Mujtaba” and that in (their/his/her) opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Computer Sciences.

**Supervisor**

**External Examiner**

**Head of Department (Department of Computer Science)**

**EXECUTIVE SUMMARY**

RIn the home decor industry, one of the main challenges for customers is envisioning how items will look and fit in their own spaces, leading to lower purchase confidence, higher return rates, and less overall satisfaction. Traditional online shopping lacks the interactive, personalized experience that customers need to make informed decisions. To bridge this gap, we developed the "Easy Buy with 3D Augmented Reality" app, a solution that brings immersive AR technology into the home decor shopping experience

.

Key Features of Roomy include:

The "Easy Buy with 3D Augmented Reality" app revolutionizes home decor shopping by enabling users to virtually place decor items within their real environments. Using AR technology, customers can view 3D models of products such as furniture, wall art, and decorative items, assessing their style, color, and fit within their personal spaces before making a purchase. This feature greatly enhances buyer confidence, reduces return rates, and provides a visually engaging and user-friendly shopping experience.

The app combines a cross-platform mobile interface developed with Flutter and a robust back end powered by Django. These technologies enable smooth, responsive navigation, accurate AR rendering, and secure data handling. The app’s architecture leverages Object-Oriented Programming (OOP) principles and follows a hybrid 3-tier MVC structure, ensuring modularity, scalability, and maintainability. Adopting an Agile development methodology has allowed us to refine the app iteratively based on stakeholder feedback, aligning with user needs and industry standard.

.

**ACKNOWLEDGEMENT**

All praise is due to Almighty Allah who granted us a small portion of his infinite wisdom, enabling us to successfully complete this demanding task.

We would like to extend our heartfelt gratitude to our project leader 'professor' “Mr. Umair Mujtaba”. absence from the project would have made its completion uncertain without his personal supervision, advice, and valuable guidance. We are immensely grateful to them for their unwavering support and constant assistance throughout this project.

And we are immensely grateful to our parents and family members who have consistently supported us and instilled in us the values of honesty and hard work.

**Muhammad Akhlaq Huzaifa Sajjad Rimsha Muneer**

**ABBREVIATIONS**

|  |  |
| --- | --- |
| **SRS** | Software Requirement Specification |
| **PC** | Personal Computer |
| **FC** | Functional Requirements |
| **NFC** | Non-Functional Requirements |
| **UC** | Use Case |
| **DFD** | Data Flow Diagram |

**TABLE OF CONTENTS**

1. [Introduction 1](#_bookmark0)
   1. [Brief 1](#_bookmark1)
   2. [Relevance to Course Modules 1](#_bookmark2)
   3. [Project Background 2](#_bookmark3)
   4. [Literature Review 3](#_bookmark4)
   5. [Analysis from Literature Review (in the context of your project) 3](#_bookmark5)
   6. [Methodology and Software Lifecycle for This Project 3](#_bookmark6)
      1. [Rationale behind Selected Methodology 3](#_bookmark7)
2. [Problem Definition 5](#_bookmark8)
   1. [Problem Statement 5](#_bookmark9)
   2. [Deliverables and Development Requirements 5](#_bookmark10)
   3. [Current System (if applicable to your project) 6](#_bookmark11)
3. [Requirement Analysis 8](#_bookmark12)
   1. [Use Cases Diagram(s) 8](#_bookmark13)
   2. [Detailed Use Case 9](#_bookmark14)
   3. [Functional Requirements 20](#_bookmark15)
   4. [Non-Functional Requirements 21](#_bookmark16)
4. [Design and Architecture 23](#_bookmark17)
   1. [System Architecture 23](#_bookmark18)
   2. [Data Representation [Diagram + Description] 26](#_bookmark19)
   3. [Process Flow/Representation 36](#_bookmark20)
   4. [Design Models [along with descriptions] 37](#_bookmark21)
5. [Implementation 37](#_bookmark22)
   1. [Algorithm 39](#_bookmark23)
   2. [External APIs 42](#_bookmark24)
   3. [User Interface 42](#_bookmark25)
6. [Testing and Evaluation 54](#_bookmark26)
   1. [Manual Testing 54](#_bookmark27)
      1. [System testing 54](#_bookmark28)
      2. [Unit Testing 54](#_bookmark29)
      3. [Functional Testing 61](#_bookmark30)
      4. [Integration Testing 61](#_bookmark31)
   2. [Automated Testing: 66](#_bookmark32)
      1. [Tools used: 66](#_bookmark33)
7. [Conclusion and Future Work 67](#_bookmark34)
   1. [Conclusion 67](#_bookmark35)
   2. [Future Work 67](#_bookmark36)
8. [References 68](#_bookmark37)

**LIST OF FIGURES**

Fig 3.1: Use Case Diagram 8

Fig 4.1: Architecture diagram 25

Fig 4.2 System Sequance Diagram 26

Fig 4.3 Sequence diagram for Registration 27

Fig 4.4 Sequence diagram for Sign in 28

Fig 4.5 Sequence diagram for Manage Profile 29

Fig 4.6 Sequence diagram for create Chat Rooms 30

Fig 4.7 Sequence diagram for invite members 31

Fig 4.8 Sequence diagram for join chat room 32

Fig 4.9 Sequence diagram for search Rooms 33

Fig 4.10 Sequence diagram for update room 34

Fig 4.11 Sequence diagram for delete chat room 35

Fig 4.12 Process Flow/Representation 36

Fig 4.13 Data Flow Diagram level 0 37

Fig 4.14 Data Flow Diagram level 1 37

Fig 4.15 Data Flow Diagram level 2 38

Fig 5.1 Home Page 42

Fig 5.2 Profile page 43

Fig 5.3 Registration page 43

Fig 5.4 Login page 44

Fig 5.5 forget password screen 44

Fig 5.6 Reset password screen 45

Fig 5.7 messages page 45

Fig 5.8 List of Rooms page 46

Fig 5.9 List of users in group 47

Fig 5.10 Video call joining screen 48

Fig 5.11 Video calling screen 48

Fig 5.12 Screen sharing window 49

Fig 5.13 Messaging in video call screen 49

Fig 5.14 Private Room screen 50

Fig 5.15 Temporary Room Screen 50

Fig 5.16 Login Screen 51

Fig 5.17 Registration Screen 51

Fig 5.18 Forget password Screen 52

Fig 5.19 Create Room Screen 52

Fig 5.20 Profile Management Screen 53

Fig 5.21 Profile Management Screen 53

**LIST OF TABLES**

Table 3.1 Register 9

Table 3.2 Sign In 10

Table 3.3 Manage Profile 11

Table 3.4 Create Chat Room 12

Table 3.5 Invite Members 13

Table 3.6 Search Room 14

Table 3.7 Join Room 15

Table 3.8 Schedule Chat Room 16

Table 3.9 Update Chat Room 17

Table 3.10 Leave Room 17

Table 3.11 Delete chatrooms 18

Table 3.12 log Out 19

Table 5.1 Details of APIs used in the project 42

Table 6.1 System Testing Testcase 54

Table 6.2 Login Unit Testcase 56

Table 6.3 Profile Management Module Unit Testcase 57

Table 6.4 Chat Room Creation Module Test case 57

Table 6.5 temporary chat room creation Module Test case 58

Table 6.6 invitation module Test case 58

Table 6.7 search module Test case 59

Table 6.8 join chat rooms module Test case 59

Table 6.9 Room Scheduling module Test case 59

Table 6.10 Real time communication Test case 60

Table 6.11 Private rooms accessibility module Test case 60

Table 6.12 User Registration Functional Testcase 61

Table 6.13 User Login Functional Testcase 62

Table 6.14: User Profile management Functional Testcase 62

Table 6.15: User invitation management Functional Testcase 63

Table 6.16: User view list of users Functional Testcase 63

Table 6.17: Room creation Functional Testcase 63

Table 6.18: Search Rooms Functional Testcase 63

Table 6.19: Delete Rooms Functional Testcase 64

Table 6.20: Temporary Room creation Functional Testcase 64

Table 6.21: Private Room creation Functional Testcase 64

Table 6.22: Schedule Rooms Functional Testcase 64

Table 6.23: Video calling Functional Testcase 65

Table 6.24: Messaging audio/Text Functional Testcase 65

Table 6.25: Logout Functional Testcase 65

Table 6.26: Integration Testcase 65

Table 6.26: *Table 6.27: Automated testing* 66

# Introduction

Roomy is dynamic web application build for communication in the real time with themed chat rooms. This application helps us to connect with people having same interests. The key feature of roomy app is to create temporary chat rooms. Users can send audio messages and video calling features. The unique feature is temporary chat rooms. User can be able to discuss various topics in the rooms.

## Brief

The "Easy Buy with 3D Augmented Reality" app project aims to revolutionize the home decor shopping experience by integrating AR technology, allowing users to virtually place decor items in their own spaces. This approach enhances the shopping experience, helping users make more confident purchasing decisions and reducing the likelihood of returns. The outcome of this project is a feature-rich, user-centric mobile application with a visually engaging and intuitive interface that leverages 3D visualization for improved product exploration.

## Relevance to Course Modules

Relevance to Course Modules in the Context of the Roomy Proposal.

### Object-Oriented Programming (OOP):

The project heavily relies on OOP principles to create modular, reusable components for user profiles, AR models, and products. Courses like Object-Oriented Programming and Advanced Object-Oriented Design provided the foundational knowledge of classes, inheritance, polymorphism, and encapsulation, which are applied in the app's design and implementation

### Mobile Application Development****:****

This project integrates skills from *Mobile Application Development*, particularly in Flutter, which enables cross-platform mobile development. Understanding platform-specific UI design and optimization techniques directly contributed to creating a responsive, user-friendly interface for iOS and Android.

### Project Management and Agile Methodologies:

This project integrates skills from *Mobile Application Development*, particularly in Flutter, which enables cross-platform mobile development. Understanding platform-specific UI design and optimization techniques directly contributed to creating a responsive, user-friendly interface for iOS and Android.

1

### Web Technologies and Programming:

Utilizing Django for back-end development and database management connects with skills acquired in *Web Technologies* and *Web Application Development* courses. Knowledge of frameworks, server-side scripting, and CRUD operations has been essential in building the logic and data abilities.

### Augmented Reality (AR) Fundamentals:

The project applies AR technology, allowing users to visualize products in their own space. Courses in *Computer Graphics* and *Human-Computer Interaction* (HCI) provided insight into visual rendering, 3D modeling, and user experience design, all of which were crucial for building engaging AR experiences.

### Software Engineering:

The use of Agile methodology for iterative development, requirements gathering, and testing aligns with concepts from Software Engineering and Software Project Management courses. These courses introduced essential practices in version control, project management, and documentation, ensuring that the project follows a structured approach.

## Project Background

The idea behind the "Easy Buy with 3D Augmented Reality" app is rooted in the need to transform the traditional online shopping experience for home decor by leveraging Augmented Reality (AR) technology. In the home decor market, users often struggle to visualize how products like furniture, wall art, and other decorative items will look and fit in their actual spaces. This uncertainty can lead to lower buyer confidence, higher return rates, and reduced customer satisfaction.

AR technology bridges this gap by allowing users to place virtual representations of decor items in their real-world environments through their mobile devices. By integrating 3D models of decor items, the app enables users to assess product dimensions, colors, and overall fit within their space before making a purchase

Augmented Reality overlays digital information—such as images, 3D models, and animations—onto the real world through the device’s camera. In the context of home decor shopping, the user’s mobile device scans the room or selected space, and the AR engine processes this visual data to place a virtual object within that physical environment. The app then aligns the virtual decor item with the spatial dimensions captured by the camera, enabling the user to walk around, view from different angles, and adjust parameters (e.g., size, color) to see how the item would look in real life.

The AR component is complemented by a user-friendly interface, developed with Flutter, and a robust backend using Django. This app structure provides smooth navigation, accurate AR rendering, and secure data handling, offering a seamless shopping experience

## Literature Review

Here is the literature review of the project Roomy.

The "Easy Buy with 3D Augmented Reality" app aligns with several current trends and ongoing research in AR, e-commerce, and home decor. This literature review explores key developments, trends, and existing products that contribute to the evolving landscape of AR-integrated shopping experiences.

* 1. **Current Trends in Augmented Reality in E-commerce:**

Here is the current trends of the system

* **Augmented Reality in Retail:**

AR technology has gained significant traction in retail, particularly in e-commerce. According to a Deloitte study, retailers leveraging AR have seen improvements in customer engagement, with shoppers spending up to 40% more time on AR-enabled products than on standard listings. This shows the potential of AR to boost engagement and support more confident buying decisions by allowing customers to visualize products realistically.

* **Personalized Shopping Experiences:**

Consumers increasingly seek personalized shopping experiences that cater to their preferences. AR offers a unique way to achieve this by tailoring the shopping experience to the customer’s physical environment, enabling them to make informed decisions based on their personalized home layout.

* **Reducing Return Rates:**

A significant trend in e-commerce is the use of AR to help reduce return rates. By allowing users to “try before they buy” virtually, AR mitigates uncertainty, leading to a decrease in returns due to incorrect product fit or appearance, a common issue in the decor market.

* 1. **Research in Augmented Reality for Home Décor:**
* **Spatial Computing and Object Placement:**

Research in spatial computing has focused on enhancing AR’s ability to detect surfaces, assess room dimensions, and accurately position virtual objects in real-world spaces. Studies, such as those from Stanford's Virtual Human Interaction Lab, highlight the importance of spatial computing for accurately placing decor items within different room layouts, enhancing realism and user satisfaction.

* **User Experience and Interaction Design:**

Recent studies underscore the importance of intuitive interfaces in AR applications. Research published in *The Journal of Visual Languages and Computing* highlights that an intuitive user interface (UI) and user experience (UX) design in AR applications are critical for user adoption and engagement, especially for non-technical users. This influences the development of user-friendly interfaces, as seen in popular AR-based decor apps like IKEA Place and Wayfair.

* **Rendering and Realism in AR:**

Research in AR has advanced realistic rendering, allowing users to see lifelike textures and shadows on virtual items in real time. Enhanced rendering techniques, as discussed in publications by ACM SIGGRAPH, focus on lighting and material accuracy, both of which are essential for decor apps to provide an authentic visualization experience.

* 1. **Existing Products and Applications**
* **IKEA Place:**

One of the pioneers in AR decor applications, IKEA Place, allows users to place virtual IKEA furniture in their homes. By using ARKit, the app accurately scans room dimensions and overlays scaled 3D models of furniture, showcasing the potential of AR in creating a seamless, life-like product experience. IKEA Place demonstrates that AR can effectively support large inventories and complex 3D assets.

* **Wayfair’s AR Feature:**

Wayfair, another leader in home decor, integrates AR capabilities to allow users to place virtual decor items in their homes before purchasing. Its emphasis on simplicity and user-friendliness shows that easy-to-use AR functions can enhance the shopping experience, even for users who may not be familiar with advanced technology.

## Analysis from Literature Review

The insights from the literature review provide a strong foundation for the development of the "Easy Buy with 3D Augmented Reality" app, enabling us to build upon existing AR solutions while addressing their limitations. The following analysis draws comparisons with the existing research, trends, and products discussed in the literature review, highlighting how our project innovates and improves upon them.

### Advancing the User Experience and Interaction Design:

IKEA Place and Wayfair's AR features are known for their user-friendly design and simplicity, which is essential for attracting a broad audience. However, these apps often have limited interaction options, such as restricted ability to adjust colors, sizes, or styles of products in the AR view.

IKEA Place and Wayfair's AR features are known for their user-friendly design and simplicity, which is essential for attracting a broad audience. However, these apps often have limited interaction options, such as restricted ability to adjust colors, sizes, or styles of products in the AR view.

Our app aims to provide a more customizable and interactive AR experience. Users will be able to adjust product parameters like color, material, and size within the AR view, offering a more dynamic experience than current solutions. This customization enhances the user’s decision-making process, addressing a significant limitation identified in existing AR decor applications.

* + 1. **Improved Accuracy in Object Placement and Spatial Scaling**

One of the main challenges identified in existing AR solutions, such as Amazon’s AR View, is the occasional lack of scaling accuracy and placement precision. Users often find that virtual items appear inaccurately scaled, affecting their ability to make confident purchasing decisions.

The "Easy Buy" app addresses this challenge by incorporating advanced spatial computing algorithms and high-quality rendering to ensure accurate placement and scaling. By implementing the latest developments in spatial detection, our app reduces the chance of scaling errors, providing a more realistic and reliable AR shopping experience that instills confidence in users’ buying decisions.

* + 1. **Enhancing Rendering Quality for Realism:**

Current products like Wayfair and IKEA Place have made strides in rendering but still face occasional discrepancies between virtual representations and real-life products, especially regarding textures and lighting.

To address this, "Easy Buy" prioritizes advanced rendering techniques, including realistic lighting and shadow effects that align with the room’s natural lighting. Drawing on insights from research in ACM SIGGRAPH, our app seeks to produce life-like AR representations by accurately simulating textures and lighting, which is crucial for an authentic visualization experience.

* + 1. **Customization and Flexibility in Product Visualization:**

Many current AR home decor apps have limited product variety and customization in the AR view, which can restrict users’ ability to find the perfect match for their unique spaces.

Our app addresses this by offering a more extensive catalog of customizable decor items. Users can tailor product aspects (e.g., size, color) and preview different configurations in real-time, a feature inspired by the personalized shopping trend. This enhanced flexibility makes "Easy Buy" stand out as a more adaptable solution in AR shopping for home decor.

## Methodology and Software Lifecycle for This Project

A For the development of the "Easy Buy with 3D Augmented Reality" app, the **Agile methodology** has been selected due to its flexibility, iterative structure, and user-centered approach. This methodology, combined with the **Object-Oriented Programming (OOP)** paradigm, ensures a modular and scalable codebase while promoting continuous improvement and adaptation throughout the project lifecycle.

The **Software Development Life Cycle (SDLC) model** selected is Agile, which allows for incremental development, regular feedback, and faster delivery of functional components. Each development cycle, known as a sprint, will deliver part of the application, allowing for immediate testing and feedback from stakeholders. This iterative process helps in addressing complex requirements like 3D rendering, AR integration, and a fluid user interface by focusing on small, manageable tasks

### Agile Software Development Methodology

Agile methodology is best for Roomy due to its force on iterative development, collaboration, and adaptability. Using agile methodology, the project remains

flexible and responsive to user feedback and changing requirements throughout its lifecycle

Characteristics of Agile Software Development Methodology

* + - * **Modularity:** OOP enables breaking down complex systems, like AR-based applications, into self-contained, reusable classes. For example, each AR decor item can be represented as an object, encapsulating all properties (size, colour, style) and behaviours
      * **Reusability:** OOP’s reusable classes simplify adding new features or modifying existing ones. The AR feature, product management, and user profile can be coded as separate classes and reused across different parts of the application.
      * **Maintainability:** The encapsulation of data and behaviors in OOP improves code readability and maintainability, making it easier for developers to debug and update components as the app evolves.
      * **Design Patterns:**

OOP supports essential design patterns like MVC (Model-View-Controller), ensuring a structured and maintainable codebase

### Rationale behind Selected Methodology:

Here is the selected methodology:

* + - * **Flexibility:** Agile allows the team to adapt to changes in requirements or technology, which is critical in an evolving field like AR.

Frequent Deliverables: Through Agile’s sprint cycles, new features are released incrementally, ensuring continuous progress and early detection of issues.

* + - * **User Feedback:** Agile emphasizes collaboration with stakeholders, allowing user feedback to shape the product in each iteration etc.
      * **Risk Management:** Agile’s iterative approach enables early identification and mitigation of potential risks, which is crucial for a technically complex project like AR integration.

# Problem Definition

This chapter details the precise problem that the "Easy Buy with 3D Augmented Reality" app is designed to solve, outlining the gap in current online shopping experiences for home decor and highlighting the expected outcome of the solution. This app aims to tackle the challenges users face in visualizing how decor items will appear in their actual spaces, addressing uncertainty in online shopping by enabling realistic, interactive product visualization.

## Problem Statement

The rapid growth of online shopping in the home decor market has highlighted a significant challenge for consumers: the inability to visualize how items will look in their unique spaces. Traditional product images fail to convey a sense of scale, style compatibility, and spatial fit, resulting in frequent returns and reduced customer satisfaction. This problem creates a barrier for customers seeking confidence in their purchasing decisions, especially when buying high-investment items like furniture and decor.

## Deliverable and Development Requirements

The Deliverables for the Roomy.

### Mobile Application (iOS & Android):

Create A cross-platform app that enables users to browse, customize, and visualize decor items in AR before purchasing

### Augmented Reality (AR) Feature:

Implement An interactive AR interface allowing users to place, adjust, and view decor items in real-time within their physical space.

* + 1. **User Authentication and Profiles:**

A system for user registration, social media login, and profile management to enable personalized recommendations and purchase history.

* + 1. **Product Catalog and Customization Options:**

A robust catalog of decor items with customizable features for users to tailor each item to their preferences

* + 1. **Shopping Cart and Checkout Process:**

A seamless, secure process for adding items to a cart and completing purchases with various payment options.

* + 1. **Backend and Database:**

A server infrastructure to manage product data, user accounts, order history, and other dynamic content.

* + 1. **Admin Panel:**

A web-based dashboard for managing product listings, updating inventory, and processing orders.

## Current System

Many platforms now support real-time communication, each with its own features. Two of the most popular are Slack and Discord, which help teams collaborate and build communities using chat rooms or channels.

### Slack:

Slack is a business communication tool that helps team members stay connected and share important information. It allows people to work together more smoothly

Discord is a free communication app where millions of users connect and used messages, audios, videos with each other it’s a gaming platform. Discord can be used on every operating system like window, Linux, MacOS, iOS etc and used in systems like web, Android etc

### Limitations of Current Systems:

Slack and Discord are good for communication, but they have some limits that Roomy wants to fix. Slack’s free version has restrictions that make it hard for small groups. It’s better for long-term projects than for quick chat rooms. Discord, on the other hand, isn’t great for professional settings. It focuses on voice chats and is made for gamers. Plus, neither platform makes it easy to set up and schedule temporary chat rooms for specific talks or events, which is important for some users.

### Roomy's Improvement:

Roomy is made to fix these issues. It lets users create temporary chat rooms. They can also schedule talks using a built-in calendar. Users can chat with text, audio, or video easily. Roomy is good for both work and casual chats. It’s a better option for people who want quick discussions based on interests. Plus, it gives more control over how the rooms are set up.

# Requirement Analysis

Here is a Use case Diagram of Roomy.

## Use Cases Diagram(s)

A diagram of a customer

AI-generated content may be incorrect.

*Fig 3.1 Use Case Diagram*

A diagram of a product

AI-generated content may be incorrect.

*Fig 3.2 Use Case Diagram*

## Detailed Use Case

Here is the Detailed Use Case Diagram of Roomy.

### Use Case Description for Register:

*Table 3.1 Register*

|  |  |
| --- | --- |
| **Use Case ID:** | UC-1 |
| **Use Case Name:** | Login to the Application |
| **Actors:** | **Primary Actor:** User, Admin **Secondary Actors:** Authentication System |
| **Description:** | A user or admin logs into the application using their email and password. This use case ensures that only registered users and admins can access personalized features and their respective dashboards. |
| **Trigger:** | A user or admin wants to access the app's features and chooses the login option from the app's welcome screen. |
| **Preconditions:** | PRE-1: The user or admin has previously registered an account.  PRE-2: The user or admin has the application installed on their device. |
| **Postconditions:** | POST-1: The user or admin is logged into the app and can access personalized features. POST-2: The system records the login event for security and analytics purposes. |
| **Normal Flow:** | 1. The user or admin selects the "Login" option from the welcome screen.  2. The system displays the login form requesting email and password.  3. The user or admin enters their email and password and submits the form.  4. The system verifies the email and password with the authentication system.  5. The system logs the user or admin into the application.  6. The system redirects the user to the home screen or the admin to the admin dashboard.  7. The system displays a welcome message with the user's or admin's name. |
| **Alternative Flows:** | **AF-1: Incorrect Password or Email**  1. User or admin enters incorrect email or password.  2. The system displays an error message indicating invalid login credentials.  3. User or admin can retry logging in or use the "Forgot Password" option. **AF-2: Forgot Password**  1. User or admin selects "Forgot Password" option.  2. System prompts user or admin to enter their registered email.  3. User or admin enters email and submits the form.  4. System sends password reset instructions to the entered email. |
| **Exceptions:** | EX-1: Authentication System Unavailable  1. System detects that the authentication service is down.  2. System displays a message to the user or admin indicating that login is currently unavailable and to try again later. |
| **Business Rules** | BR-1: Users and admins must provide a valid email address and password to log in.  BR-2: Passwords must meet the security requirements (e.g., minimum length, complexity).  BR-3: Users and admins will be locked out after 5 consecutive failed login attempts for 15 minutes to prevent brute-force attacks. |
| **Assumptions:** | A-1: It is assumed that users and admins will remember their login credentials or use the "Forgot Password" option if needed.  A-2: It is assumed that users and admins have a stable internet connection to interact with the authentication system. |
| **Use Case ID:** | UC-1 |

### Use Case Description for Sign in:

*Table 3.2 Sign In*

|  |  |
| --- | --- |
| **Use Case ID:** | UC-2 |
| **Use Case Name:** | Signup for an Account |
| **Actors:** | **Primary Actor:** User,  **Secondary Actors:** Authentication System, Email Service |
| **Description:** | A user or admin signs up for a new account by providing necessary details and verifying their email. This use case ensures that only valid users and admins can create accounts and access the application's features. |
| **Trigger:** | A user or admin wants to create a new account and selects the signup option from the app's welcome screen. |
| **Preconditions:** | PRE-1: The user has not previously registered with the same email.  PRE-2: The user has the application installed on their device. |
| **Postconditions:** | POST-1: The user account is created and stored in the authentication system.  POST-2: A verification email is sent to the user provided email address.  POST-3: The system logs the signup event for security and analytics purposes. |
| **Normal Flow:** | 1. The user selects the "Signup" option from the welcome screen.  2. The system displays the signup form requesting necessary details (e.g., name, email, password, and role).  3. The user fills in the form with the required details and submits it.  4. The system checks if the email is already registered.  5. The system stores the new account details in the authentication system.  6. The system sends a verification email to the provided email address.  7. The user clicks the verification link in the email.  8. The system verifies the email and activates the account.  9. The system displays a confirmation message indicating successful signup and email verification. |
| **Alternative Flows:** | **AF-1: Email Already Registered**  1. The system detects that the email is already registered.  2. The system displays an error message indicating the email is already in use.  3. The user can choose to log in instead or use a different email to sign up.  **AF-2: Email Verification Failed**  1. The user or admin does not receive the verification email.  2. The user or admin requests the system to resend the verification email.  3. The system resends the verification email. |
| **Exceptions:** | **EX-1: Authentication System Unavailable**  1. The system detects that the authentication service is down.  2. The system displays a message indicating that signup is currently unavailable and to try again later.  **EX-2: Invalid Email Format**  1. The system detects that the entered email format is invalid.  2. The system prompts the user or admin to enter a valid email address. |
| **Business Rules** | BR-1: Users must provide a valid email address that is not already registered.  BR-2: Passwords must meet the security requirements (e.g., minimum length, complexity).  BR-3: Users must verify their email address before they can access full features of the application. |
| **Assumptions:** | A-1: It is assumed that users will provide accurate information during signup.  A-2: It is assumed that users have a stable internet connection to interact with the authentication system and receive emails. |

### Use Case Description for view Product:

*Table 3.3 view Product*

|  |  |
| --- | --- |
| **Use Case ID:** | UC-3 |
| **Use Case Name:** | View Product |
| **Actors:** | **Primary Actor:** User **Secondary Actors:** Application, Device Camera |
| **Description:** | A user can view products in 3D augmented reality (AR) to visualize how they will look in a real-world environment. This use case enhances the shopping experience by providing a realistic view of the products. |
| **Trigger:** | A user wants to see how a product will look in their environment and selects the 3D AR option |
| **Preconditions:** | PRE-1: The user has a device capable of AR and has granted camera permissions. |
| **Postconditions:** | POST-1: The user views the product in 3D AR.  POST-2: The system logs the AR view event for analytics. |
| **Normal Flow:** | 1. The user navigates to the product details page. 2. The system displays the "View in 3D AR" option. 3. The user selects the "View in 3D AR" option. 4. The system activates the device's camera and requests necessary permissions. 5. The user grants the necessary permissions. 6. The system loads the 3D model of the product. 7. The user points the device's camera to a surface where they want to place the product. 8. The system renders the 3D model of the product in the real-world environment through the device's camera.   The user views and interacts with the 3D model. |
| **Alternative Flows:** | **AF-1: Camera Permissions Denied**   1. The user denies the camera permissions. 2. The system displays a message indicating that camera permissions are required for AR.   The user can retry and grant permissions. |
| **Exceptions:** | **EX-1: AR System Unavailable**  The system detects that the AR service is down.  The system displays a message to the user indicating that AR is currently unavailable and to try again later. |
| **Business Rules** | BR-1: Users must grant camera permissions to use AR.  BR-2: The 3D models must be optimized for real-time rendering. |
| **Assumptions:** | A-1: It is assumed that users will have compatible devices to use AR.  A-2: It is assumed that users have a stable internet connection for downloading 3D models. |

### Use Case Description for Add to cart:

*Table 3.4 Add to cart*

|  |  |
| --- | --- |
| **Use Case ID:** | UC-4 |
| **Use Case Name:** | Add to Cart |
| **Actors:** | **Primary Actor:** Customer **Secondary Actors:** Easy buy with 3D AR, Product Database |
| **Description:** | A customer can add a product to their shopping cart to proceed with the purchasing process at a later time. |
| **Trigger:** | A customer decides to purchase a product and selects the option to add it to their cart. |
| **Preconditions:** | PRE**-1:** The customer is using the Easy buy with 3D AR app.  PRE**-2:** The customer has selected a product that is available in the product database. |
| **Postconditions:** | POST**-1:** The product is added to the customer's shopping cart.  POST**-2:** The system updates the cart and reflects the new total. |
| **Normal Flow:** | The customer opens the Easy buy with 3D AR app.  The customer navigates to the product details page for a desired product.  The system displays the product details, including the "Add to Cart" option.  The customer selects the "Add to Cart" option.  The system checks the product availability in the database.  The system adds the product to the customer's shopping cart.  The system updates the cart and displays a confirmation message, including the updated cart total and a link to view the cart.  The customer can continue shopping or proceed to the cart. |
| **Alternative Flows:** | The customer selects the "Add to Cart" option.  The system checks the product availability in the database.  The system finds that the product is out of stock.  The system displays a message indicating that the product is currently out of stock and cannot be added to the cart. |
| **Exceptions:** | The customer selects the "Add to Cart" option.  The system fails to add the product to the cart due to a network error.  The system displays an error message and suggests the customer try again later. |
| **Business Rules** | BR**-1:** Only products that are in stock can be added to the cart.  BR**-2:** The system must update the cart total and display the updated information to the customer |
| **Assumptions:** | A**-1:** It is assumed that customers have a stable internet connection while using the app.  A**-2:** It is assumed that the product database is regularly updated with current stock information. |

### Use Case Description for Edit Cart

*Table 3.5 Edit Cart*

|  |  |
| --- | --- |
| **Use Case ID:** | UC-5 |
| **Use Case Name:** | Edit Cart |
| **Actors:** | **Primary Actor:** Customer **Secondary Actors:** Easy buy with 3D AR, Product Database |
| **Description:** | A customer can edit the items in their shopping cart by updating quantities or removing items before proceeding to checkout. |
| **Trigger:** | A customer wants to review and modify the contents of their shopping cart. |
| **Preconditions:** | PRE**-1:** The customer is using the Easy buy with 3D AR App  PRE**-2:** The customer has added items to their shopping cart. |
| **Postconditions:** | POST**-1:** The shopping cart is updated with the customer's modifications.  POST**-2:** The system updates the cart and reflects the new total. |
| **Normal Flow:** | The customer opens the e-commerce app.  The customer navigates to their shopping cart.  The system displays the current contents of the chart, including product names, quantities, prices, and the total amount.  The customer selects an item to edit.  The system displays the updated cart contents and new total.  The customer continues shopping or proceeds to checkout. |
| **Alternative Flows:** | The customer selects an item to edit and increases the quantity.  The system checks the product availability in the database.  The system finds that the additional quantity is not available.  The system displays a message indicating the available quantity and adjusts the item quantity accordingly. |
| **Exceptions:** | The customer selects an item to edit.  The system fails to update the cart due to a network error. |
| **Business Rules** | BR**-1:** The system must accurately reflect current stock levels when updating item quantities.  BR**-2:** The system must display the updated cart total after any modifications. |
| **Assumptions:** | **A-1:** It is assumed that customers have a stable internet connection while using the app.  **A-2:** It is assumed that the product database is regularly updated with current stock information. |

### Use Case Description for Make payment:

*Table 3.6 Make Payment*

|  |  |
| --- | --- |
| **Use Case ID:** | UC-6 |
| **Use Case Name:** | Make Payment |
| **Actors:** | **Primary Actor:** Customer **Secondary Actors:** Easy buy with 3D AR, Stripe Payment Gateway |
| **Description:** | A customer can make a payment for their order using Stripe as the payment gateway. This use case handles the process of securely processing the payment and completing the order. |
| **Trigger:** | A customer proceeds to checkout and decides to pay for their order using Stripe. |
| **Preconditions:** | **PRE-1:** The customer is using the Easy buy with 3D AR App.  **PRE-2:** The customer has items in their shopping cart.  **PRE-3:** The customer has entered valid shipping information. |
| **Postconditions:** | POST**-1:** The payment is successfully processed.  POST**-2:** The order is created and confirmed.  POST**-3:** The system updates the inventory and logs the transaction. |
| **Normal Flow:** | The customer opens the Easy buy with 3D AR App and navigates to their shopping cart.  The customer reviews the cart contents and proceeds to checkout.  The system prompts the customer to enter shipping information if not already provided.  The customer enters the required shipping information.  The system displays the order summary and the option to choose a payment method.  The customer selects Stripe as the payment method.  The system redirects the customer to the Stripe payment interface.  The customer enters their payment details (credit card information) on the Stripe interface.  The system sends the payment details to Stripe for processing.  Stripe processes the payment and returns a response to the e-commerce system.  The system receives the payment confirmation from Stripe.  The system confirms the order and displays a confirmation message to the customer, including the order number and details.  The system updates the inventory and logs the transaction for analytics. |
| **Alternative Flows:** | Stripe processes the payment and returns a declined response.  The system displays a message to the customer indicating that the payment was declined and suggests checking the payment details or using a different payment method.  The customer can retry with a different payment method or correct the payment details and try again. |
| **Exceptions:** | The customer attempts to proceed with payment.  The system fails to communicate with Stripe due to a network error.  The system displays an error message and suggests the customer try again later. |
| **Business Rules** | BR**-1:** Payment details must be securely transmitted to Stripe.  BR**-2:** The order is not confirmed until payment is successfully processed. |
| **Assumptions:** | A**-1:** It is assumed that customers have a stable internet connection while making the payment.  A**-2:** It is assumed that Stripe's services are operational and available. |

### Use Case Description for Manage Product:

*Table 3.7 Manage Product*

|  |  |
| --- | --- |
| **Use Case ID:** | UC-7 |
| **Use Case Name:** | Manage Product |
| **Actors:** | **Primary Actor:** Admin **Secondary Actors:** Easy buy with 3D AR, Product Database |
| **Description:** | An admin can manage products in the e-commerce app, including adding new products, updating existing products, and deleting products. |
| **Trigger:** | The admin wants to perform product management tasks such as adding, updating, or deleting a product. |
| **Preconditions:** | PRE**-1:** The admin is logged into the e-commerce app with the appropriate permissions.  PRE**-2:** The product database is accessible. |
| **Postconditions:** | POST**-1:** The product database is updated with the changes made by the admin.  POST**-2:** The system logs the product management actions for audit purposes. |
| **Normal Flow:** | The admin logs into the e-commerce app with their credentials.  The admin navigates to the product management section.  The system displays options to add, update, or delete products.  The admin selects the option to add a new product.  The system displays a form for entering product details (name, description, price, category, images, stock quantity, etc.).  The admin fills in the required product details.  The admin submits the form to add the product.  The system validates the entered data.  The system saves the new product to the product database.  The system displays a confirmation message indicating that the product has been added successfully. |
| **Alternative Flows:** | The admin submits the form to add or update a product.  The system detects invalid or incomplete product details.  The system displays an error message indicating the issues with the entered data.  The admin corrects the details and resubmits the form. |
| **Exceptions:** | The admin attempts to add, update, or delete a product.  The system cannot access the product database due to a database error.  The system displays an error message indicating that the action cannot be completed at this time and suggests trying again later. |
| **Business Rules** | BR**-1:** All product details must be validated before being saved to the database.  BR**-2:** Only admins with appropriate permissions can perform product management tasks.  BR**-3:** The system must log all product management actions for audit purposes. |
| **Assumptions:** | A**-1:** It is assumed that admins have a stable internet connection while performing product management tasks.  A**-2:** It is assumed that the product database is regularly backed up and maintained |

## Functional Requirements

Here is the list of functional requirements.

### User Registration:

User shall be able to register for an account by providing their email, password, and additional required details

Allow The system shall validate the uniqueness of the provided email address.

Upon successful registration, the system shall send a verification email to the registered email address

### User Login:

Users shall be able to log in using their registered email and password. The system shall authenticate user credentials against the stored database. If authentication is successful, the system shall grant access to personalized features and dashboards. If authentication fails, the system shall display an error message indicating invalid login credentials

### Password Recovery:

Users shall be able to recover their password by providing their registered email address. The system shall send a password reset link to the provided email address. Upon clicking the password reset link, users shall be redirected to a page where they can reset their password

### Add New Product:

Admins shall be able to add new products to the system by providing product details such as name, description, price, and category. The system shall validate the completeness and correctness of the provided product information.

### Update Product Information:

Admins shall be able to update existing product details, including name, description, price, and category. The system shall reflect the updated product information immediately in the product catalog.

### Delete Product:

Admins shall be able to delete products from the system. The system shall prompt for confirmation before permanently deleting a product

### View Product in 3D AR:

Users shall be able to view products in 3D augmented reality to visualize them in their environment. The system shall support AR-enabled devices and provide an intuitive interface for interacting with AR content.

### Place Order

User shall be able to add products to their cart and proceed to checkout to place an order. The system shall calculate the total order amount, including taxes and shipping fees. Upon order placement, the system shall generate an order confirmation with a unique order ID

### View Reviews:

Users shall be able to view reviews and ratings submitted by other users for products.The system shall display reviews in a user-friendly format, sorted by relevance or recency

## Non-Functional Requirements

Nonfunctional requirements describe how Easy Buy should perform and the quality it must maintain. These focus on things like speed, security, ease of use, and scalability. Below are the non-functional requirements for Easy Buy.

### Performance

* + - * **Response Time:** Thesystem should respond to user actions within 2 seconds for 85% of requests.
      * **Scalability:** The system should support up to 10,000 concurrent users without performance issues.
      * **High Uptime:** The system should have 80% uptime, excluding scheduled maintenance**.**
      * **Daily Backups:** The system should perform daily backups of critical data, stored securely for 30 days.
      * **Data Encryption:** The system should encrypt all sensitive data both in transit and at rest
      * **Multi Factor Authentication:** The system should use multi-factor authentication (MFA) for admin users.
      * **Consistent Interface:** The system should provide a consistent user interface across all pages and functions
      * **Accessibility Compliance:** The system should comply with WCAG 2.1 AA accessibility standards.
      * **Scalable Architecture:** The system should support horizontal scaling to handle increasing loads.
      * **Modular Code:** The system should be developed using modular code to ease maintenance and updates
      * **Monitoring and Alerts:** The system should have monitoring and alert mechanisms for detecting and notifying people about issues

### Usability

* + - * **Easy Product Search:** The system shall allow users to search for products using a search bar and filter options with minimal effort.
      * **Simple Registration Process:** The system shall provide a straightforward registration process that requires minimal user input and offers clear instructions.
      * **Help and Support**: Provide user guides, tutorials, and troubleshooting help so users can easily understand how to use Roomy.

### Reliability

* + - * **Error Handling**: Roomy should be able to handle small errors without crashing, so users can keep chatting without disruptions.
      * **High Availability**: The system should be up and running most of the time, with only minimal downtime for updates or maintenance.
      * **Intuitive Navigation:** The system shall provide an intuitive navigation structure that allows users to easily move between different sections of the app.
      * **Clear Error Messages:** The system shall display clear and informative error messages that help users understand what went wrong and how to fix it.
      * **Quick Checkout Process:** The system shall provide a streamlined checkout process that allows users to complete their purchases with minimal steps.
      * **Accessible Design:** The system shall comply with WCAG 2.1 AA standards to ensure accessibility for users with disabilities.
      * **Consistent User Interface:** The system shall provide a consistent user interface design across all pages and functions.
      * **Efficient Order Management:** The system shall allow users to easily view, edit, and track their orders from a centralized order management page.
      * **Personalized User Dashboard:** The system shall provide a personalized dashboard for users that displays relevant information such as order history, recommended products, and account settings.
      * **Help and Support:** The system shall provide easy access to help and support resources, including FAQs, user guides, and contact options

### Compatibility

* + - * **Works on All Browsers**: Easy Buy should work smoothly across different web browsers like Chrome, Firefox, Safari, and Edge.
      * **Device-Friendly**: Easy Buy should work well on various devices, including computers, tablets, and smartphones, with consistent experience on each.

# Design and Architecture

Here is detailed Design and Architecture of Easy Buy.

## System Architecture

Here is the System Architecture of Romy.

### Frontend (React.js)

* + - * **Description:** This module represents the user interface of the Roomy chat application. It is developed using React.js, a powerful JavaScript library known for building dynamic and interactive web applications. The frontend module provides the platform for users to access all features of Roomy through a modern and user- friendly interface.
      * **Functionality:** The front end handles the user interaction with Roomy's key features, such as registering new users, logging in, creating or joining chat rooms, and engaging in text, audio, and video communications. It allows users to manage their profiles, search for chat rooms, schedule future chats, and receive notifications. Designed to be responsive, the interface adapts to various screen sizes, offering a seamless experience across desktop and mobile devices.

### Backend (Node.js + Express.js)

* + - * **Description:** This module constitutes the server-side logic of the Roomy application. Built using Node.js and Express.js, it handles the data flow between the frontend and the database, manages requests, and executes business logic. It ensures secure communication and data processing behind the scenes.
      * **Functionality:** The backend is responsible for handling CRUD operations for user accounts, chat rooms, and messaging data. It processes requests from the frontend, ensuring that actions like creating rooms, sending messages, or updating profiles are completed in real-time. Additionally, the backend orchestrates secure user authentication and data retrieval, integrating with MongoDB to store and retrieve all user and chat room information.

### Authentication Service

* + - * **Description:** This module handles the authentication of the system. It ensures that only authorized users can log in and access the system's features.
      * **Functionality:** The authentication service manages user login and registration, verifies credentials using token-based authentication (such as JWT), and enforces secure access to the platform. It allows users to log in, log out, and securely manage their accounts, ensuring that sensitive user data is encrypted, and access is role-based to maintain system integrity.

### Chat Room Service

* + - * **Description:** This module is responsible for managing chat rooms within Roomy. It handles the creation, modification, and deletion of chat rooms, both temporary and permanent.
      * **Functionality**: The chat room service allows users to create chat rooms for text, audio, and video communication. Users can also schedule future chat rooms and invite others to join. This service ensures users can manage chat room settings, search for specific rooms, update room policies, and delete rooms when they are no longer needed. It provides a dynamic environment for real-time collaboration and conversation.

### Communication Service

* + - * **Description:** This module facilitates real-time communication between users in Roomy. It supports text messaging, audio, and video communication across various chat rooms.
      * **Functionality:** The communication service ensures that messages are delivered instantly between users, whether they are sending text, audio, or video messages. It integrates WebSocket technology for real-time, two-way communication, providing fast and efficient interaction. Notifications and reminders for scheduled rooms are also managed by this service, keeping users informed and engaged.

### User Profile Service

* + - * **Description:** This module manages user profiles, allowing users to customize and update their personal information on the Roomy platform.
      * **Functionality:** The user profile service enables users to create and edit profiles by adding details such as names, bios, and profile pictures. Users can update their preferences, manage privacy settings, and view their activity history. This service ensures that personal information is securely stored and accessible for modification whenever necessary.

.

### Group Management Service

* + - * **Description:** This module manages the creation and maintenance of user groups, including open and private groups within Roomy.
      * **Functionality:** The group management service allows users to create both public and private groups for discussions. Group creators can manage membership, set privacy levels, and invite others to join. This service ensures that users can leave or join groups, update group settings, and maintain the exclusivity of private groups by controlling access.

### Database (MongoDB)

* + - * **Description:** This module handles the storage of all data within the Roomy system. Built using MongoDB, it provides a NoSQL database that stores user profiles, chat rooms, messages, and other relevant information.
      * **Functionality:** The database stores and retrieves all system data, including chat logs, user details, and room information. The MongoDB database is integrated with the backend to provide fast access to real-time data across all aspects of the platform.

### 4.1.9 Architecture Diagram

*Fig 4.1: Architecture diagram*

## Data Representation

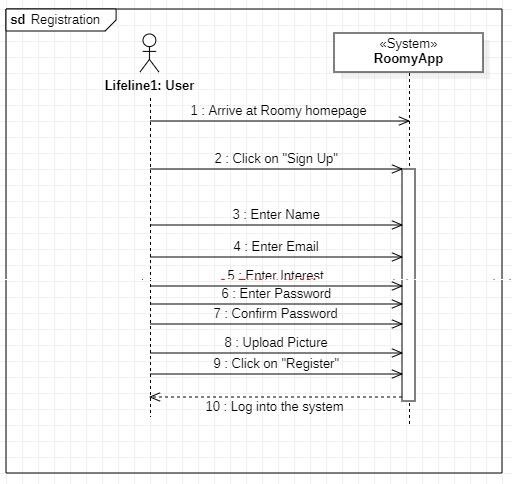
Here is detailed Design and Architecture of Roomy.

### System Sequence Diagram

*Fig 4.2: System Sequence diagram*

### Sequence Diagram for Registration

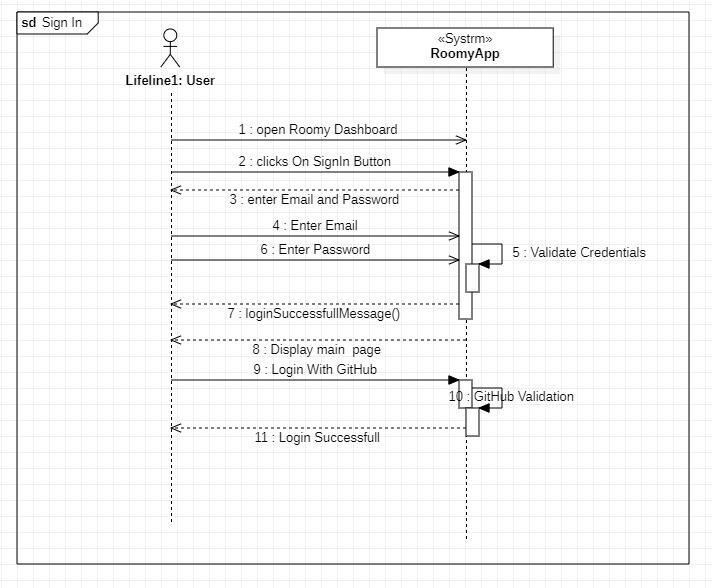
This SSD illustrates the process by which a new user signs up for Roomy by providing necessary information such as an email address and password, which the system then validates before creating a new user account and sending a confirmation response



*Fig 4.3: Sequence diagram for Registration*

### Sequence Diagram for Sign in

This SSD outlines how an existing user logs into their Roomy account by entering their email address and password, with the system validating the credentials and granting access upon successful validation

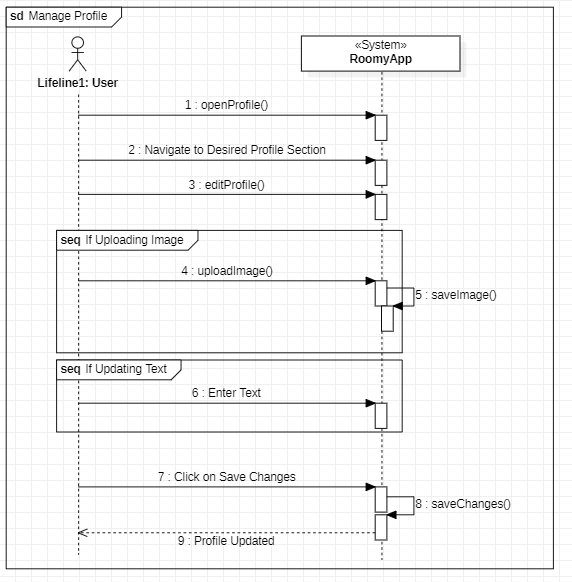


*Fig 4.4: Sequence diagram for Sign in*

.

### Sequence Diagram for Manage Profile

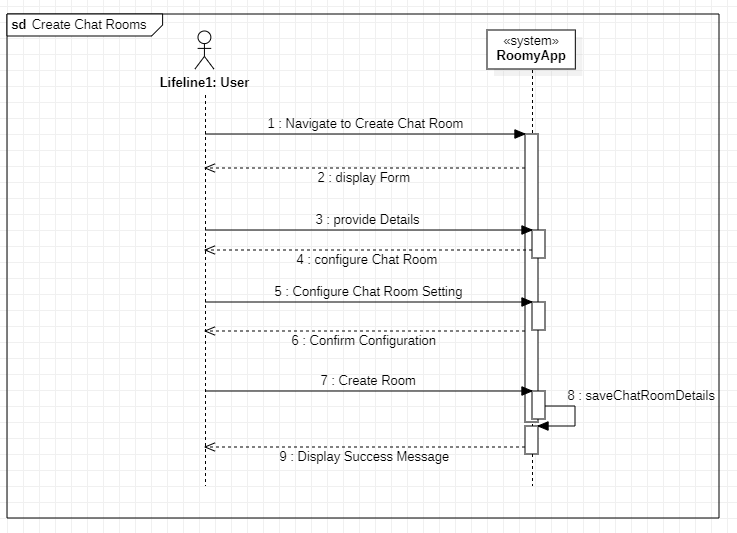
This SSD details the steps a user takes to customize their profile by updating their interests, bio, and profile picture, which the system then validates and saves before confirming the update to the user.



*Fig 4.5: Sequence diagram for Manage Profile*

### Sequence Diagram for Create Chat Rooms

This SSD shows how a user creates a new chat room by providing a room name, which the system then uses to generate the chat room and notify the user of its successful creation

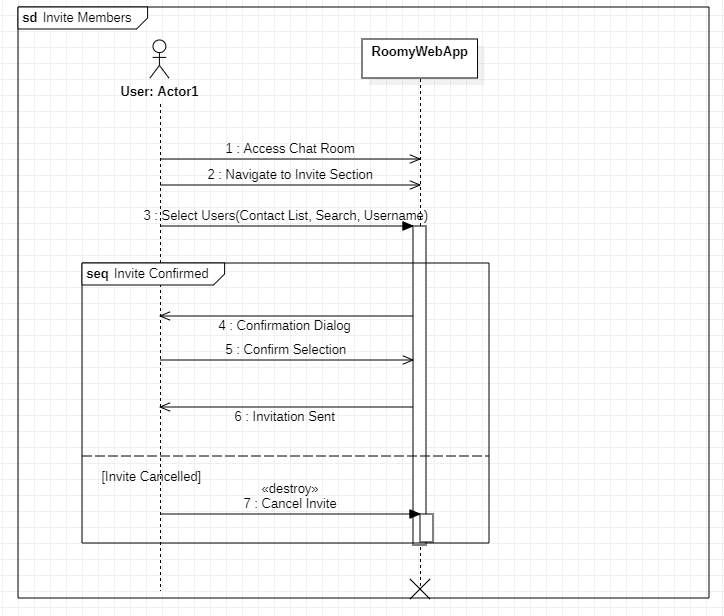


*Fig 4.6: Sequence diagram for create Chat Rooms*

.

### Sequence Diagram Invite members

This SSD depicts how a user invites other members to a chat room by generating an invitation link or sending an email invite, which the system processes and sends to the invited users

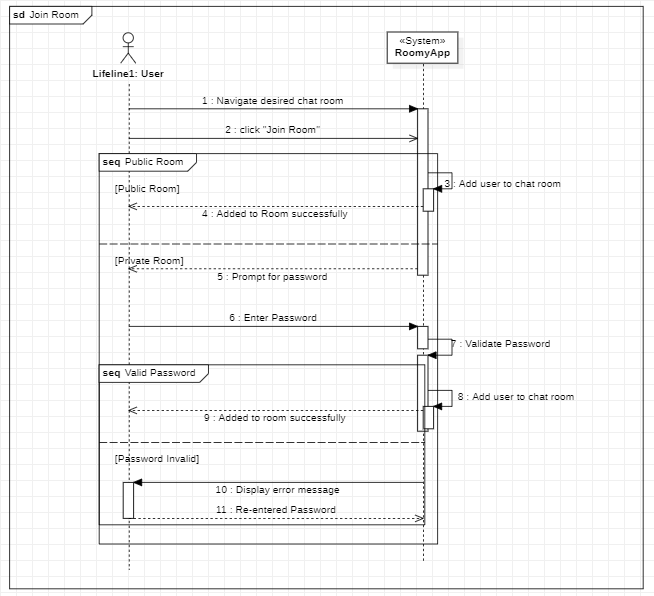


*Fig 4.7: Sequence diagram for invite members*

.

### Sequence Diagram for join chat Room

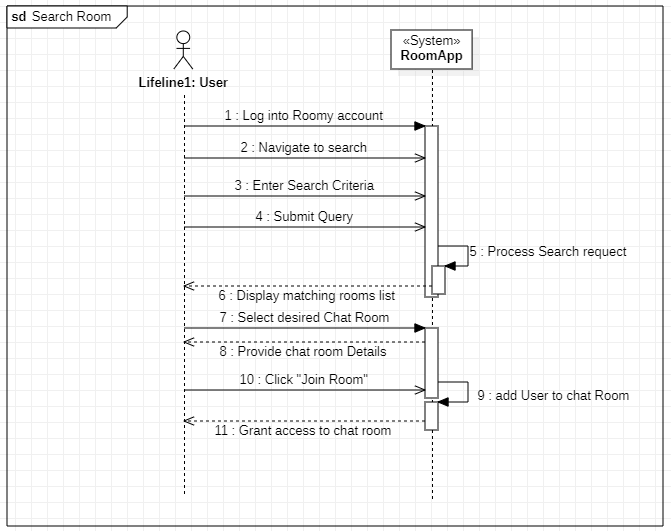
This SSD illustrates how a user joins a chat room via an invitation link, with the system validating the link and granting access to the chat room.



*Fig 4.8: Sequence diagram for join chat room*

### Sequence Diagram for Search Rooms

This SSD outlines the process a user follows to search for chat rooms by name, with the system retrieving and displaying relevant chat rooms based on the search query.



*Fig 4.9: Sequence diagram for search Rooms*

### A diagram of a system Description automatically generatedSequence Diagram for update Room

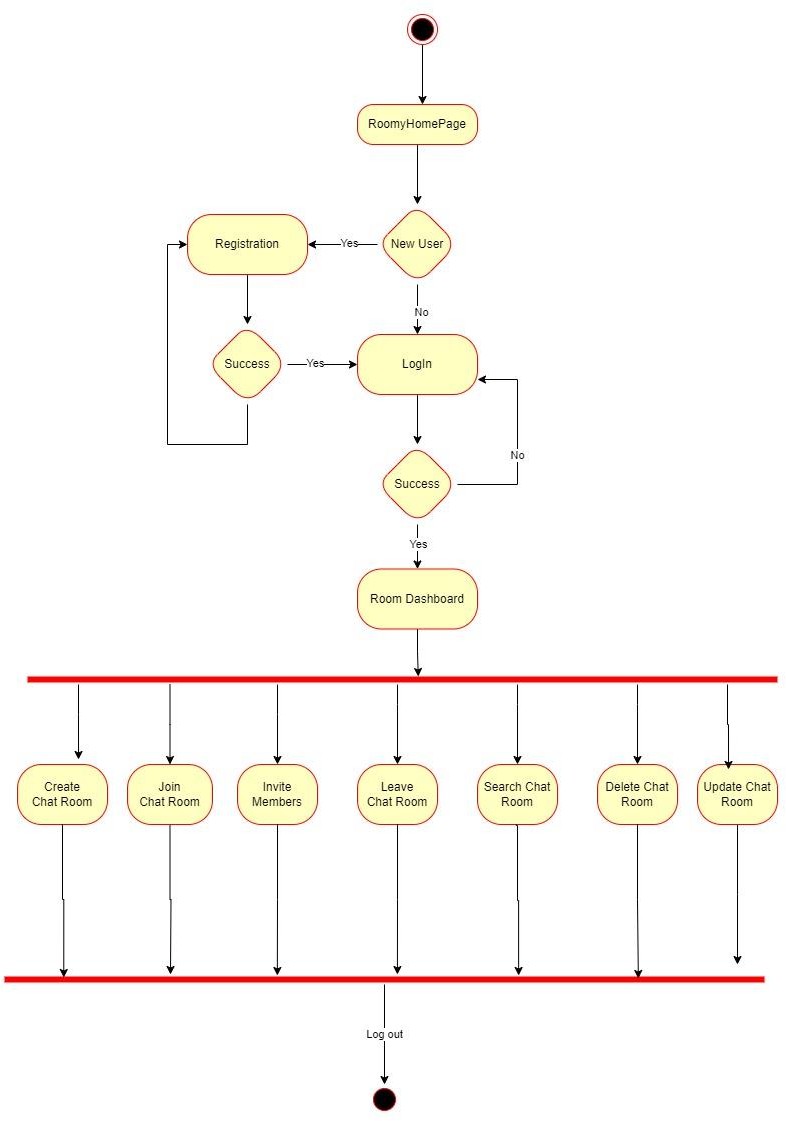
*Fig 4.10: Sequence diagram for update room*

### A screenshot of a chat room Description automatically generatedSequence Diagram for Delete Chat Room

*Fig 4.11: Sequence diagram for delete chat room*

## Process Flow/Representation

Here is a process flow diagram for Roomy.

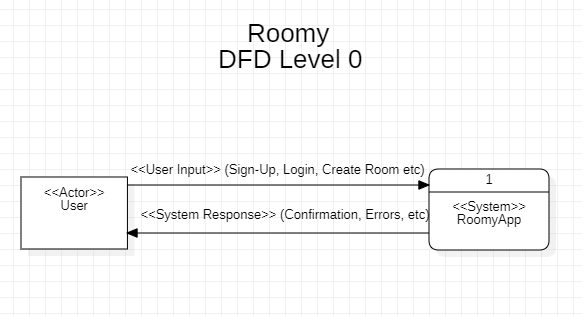


*Fig 4.12 process flow diagram*

## Design Models [along with descriptions]

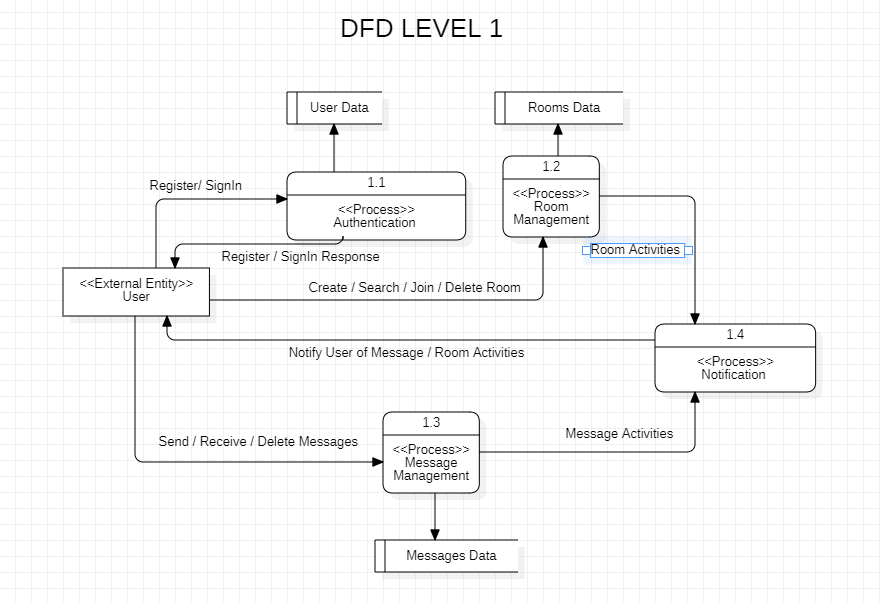
Here is Design Models with different levels of diagram

### Data Flow Diagram Level 0:

****

*Fig 4.13: Data Flow Diagram level 0*

### Data Flow Diagram Level 1:

****

*Fig 4.14: Data Flow Diagram level 1*

### Data Flow Diagram Level 2:

*Fig 4.15: Data Flow Diagram level 2*

# Implementation

This chapter will discuss implementation details.

## Algorithm

### Pseudo Code for Register User:

// Frontend (React.js)

// Function triggered when user submits the registration form function registerUser(userData) {

// Prepare data for submission formData.append('email', userData.email); formData.append('password', userData.password); formData.append('interest', userData.interest.join(',')); if (userData.image) { formData.append('profilePicture', userData.image);

}

// Send data to backend fetch(['http://localhost:5000/api/auth/createUser',](http://localhost:5000/api/auth/createUser%27) { method: 'POST',

body: formData,

})

.then((response) => response.json())

.then((data) => {

// Update UI based on response if (data.success) {

saveUserData(data.token, data.user); navigateTo('/create\_room'); displaySuccessMessage('Registration successful!');

} else {

displayErrorMessage(data.message || 'Registration failed.');

}

})

.catch(() => {

displayErrorMessage('An error occurred while processing your request.');

});

}

// Backend (Node.js with Express.js)

// Endpoint for registering a user app.post('/api/auth/createUser', (req, res) => {

const { name, email, password, interest } = req.body;

// Validate data

if (!name || !email || !password) {

return res.json({ success: false, message: 'Missing required fields.' });

}

### Pseudo Code For Login:

// Frontend (React.js)

// Function triggered when user submits login form function loginUser(credentials) {

// Send data to backend fetch('/api/auth/login', { method: 'POST', headers: {

'Content-Type': 'application/json',

},

body: JSON.stringify(credentials),

})

.then(response => response.json())

.then(data => {

// Update UI based on response if (data.token && data.user) {

storeTokenAndUserData(data.token, data.user); navigateToDashboard(); showLoginSuccessMessage();

} else {

displayLoginError(data.error || 'Login failed.');

}

})

.catch(error => {

displayLoginError('An unexpected error occurred during login.');

});

}

### Pseudo Code For Register User:

// Frontend (React.js)

// Function triggered when user submits profile update function submitProfileUpdate(userData) {

// Prepare data to send to backend

Create FormData object and append user details (name, email, password, bio, interest).

If a new profile picture exists, append it to FormData.

// Send data to backend fetch('/api/auth/updateuser', { method: 'PATCH',

headers: {

'Authorization': `Bearer ${token}`,

},

body: FormData,

})

.then(response => response.json())

.then(data => {

// Update UI based on response if (data.user) {

Display success message to the user.

Update profile picture and other details in the UI. Save updated user data to local storage.

} else {

Display error message with details from the response.

}

})

.catch(error => {

Display generic error message to the user.

});

}

// Validate incoming data

Validate fields (name, email, password, bio, interest). Handle file upload if a profile picture is included.

// Update profile in the database if (valid data) {

Update user data in the database using userId from the token. Return success response with updated user data.

} else {

Return failure response with validation error message.

}

});

## External APIs

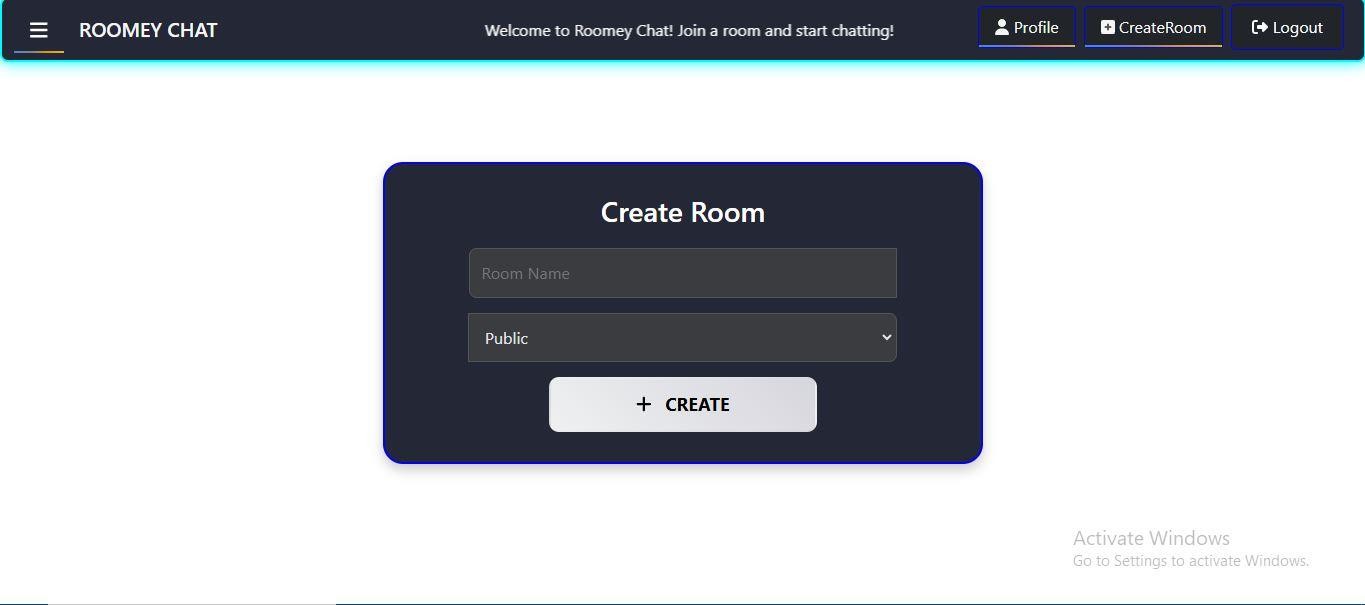
Table 5.1 shows that the Api list.

*Table 5.1: Details of APIs used in the project*

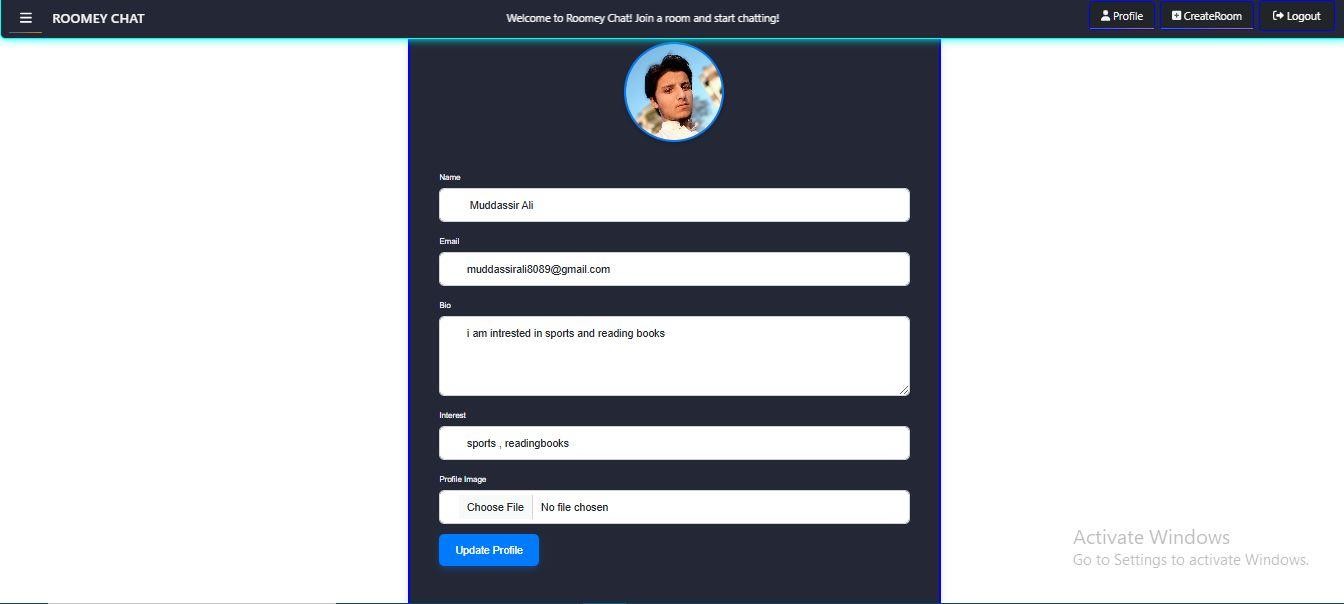
|  |  |  |  |
| --- | --- | --- | --- |
| **Name of API** | **Description of API** | **Purpose of usage** | **List down the function/class name in which it is used** |
| Zegocloud. | API integration platform that allows developers to add real-time communication, video calling, and live streaming features to their  applications | I am using the Api for video calling feature | Video.js |

## User Interface

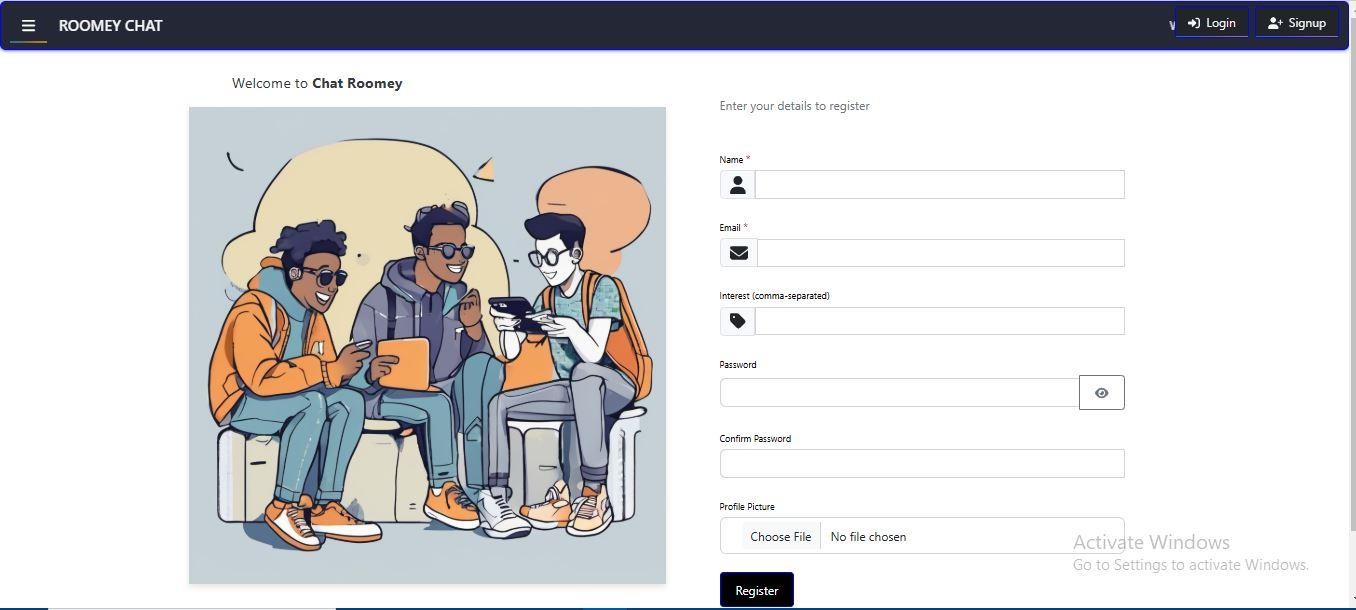
Details about user interface with descriptions.



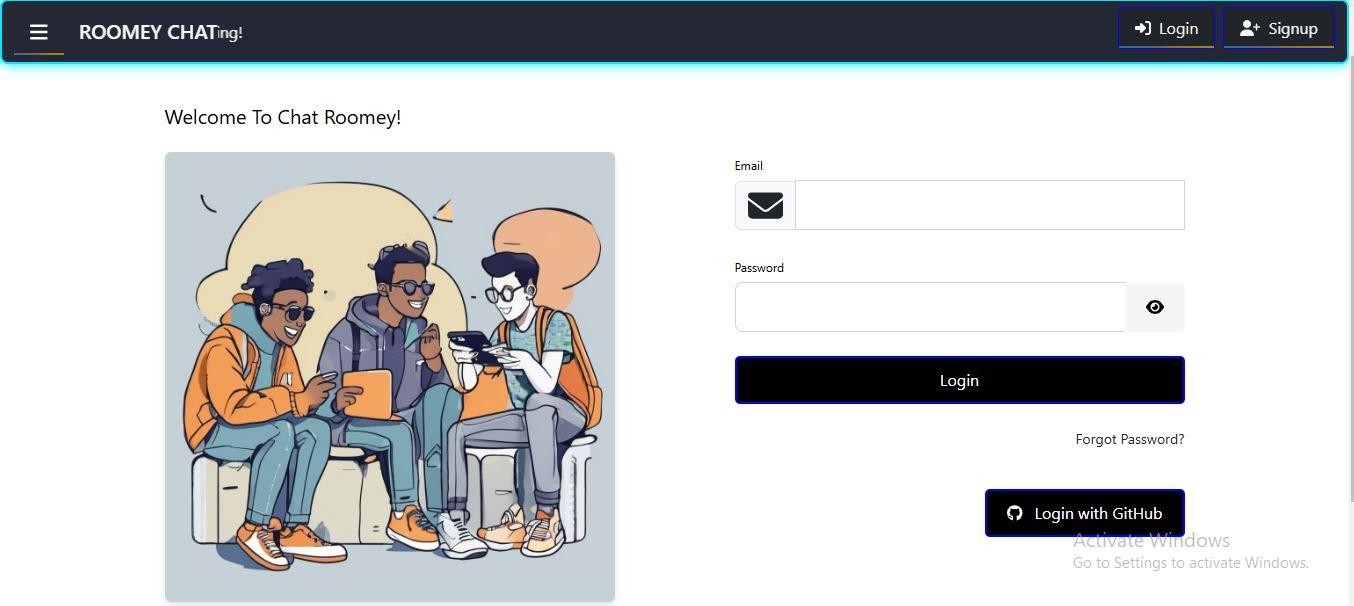
*Fig 5.1 Home Page*



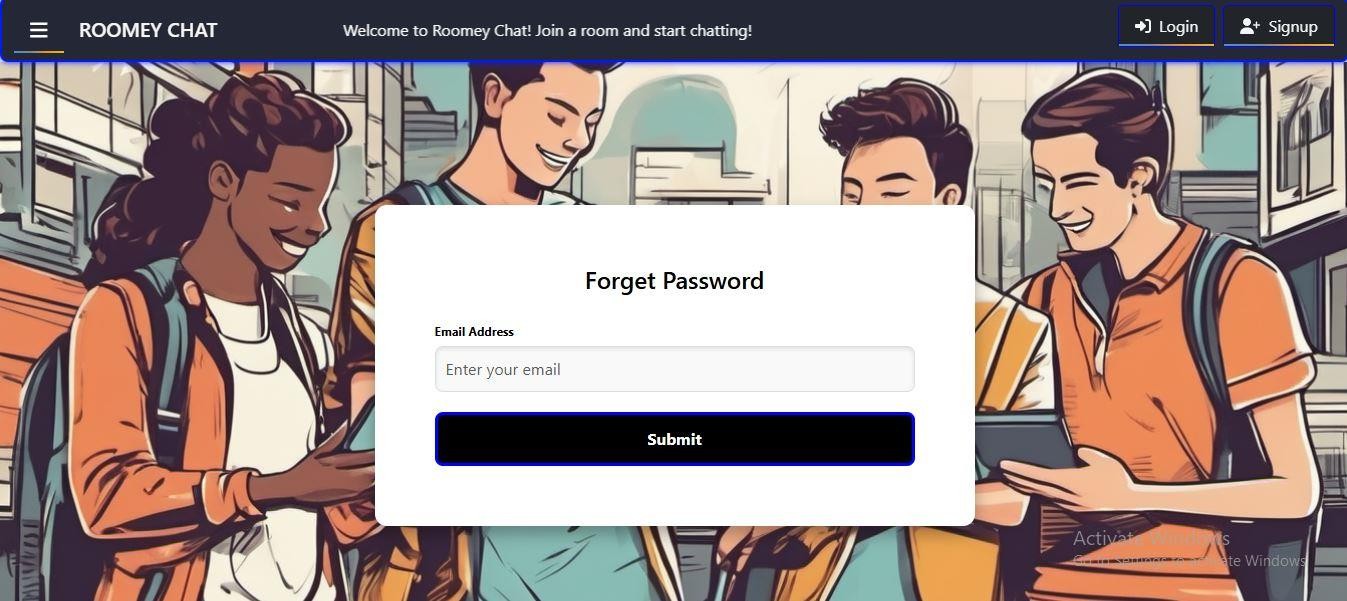
*Fig 5.2 Profile page*

**

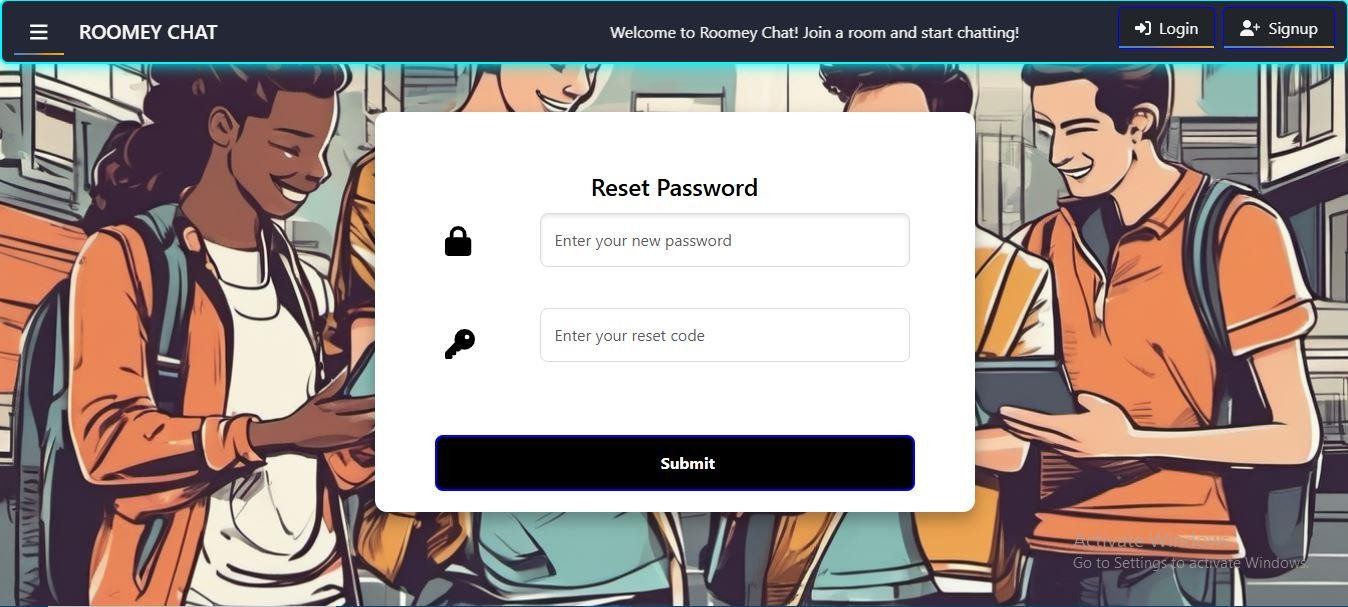
*Fig 5.3 Registration page*



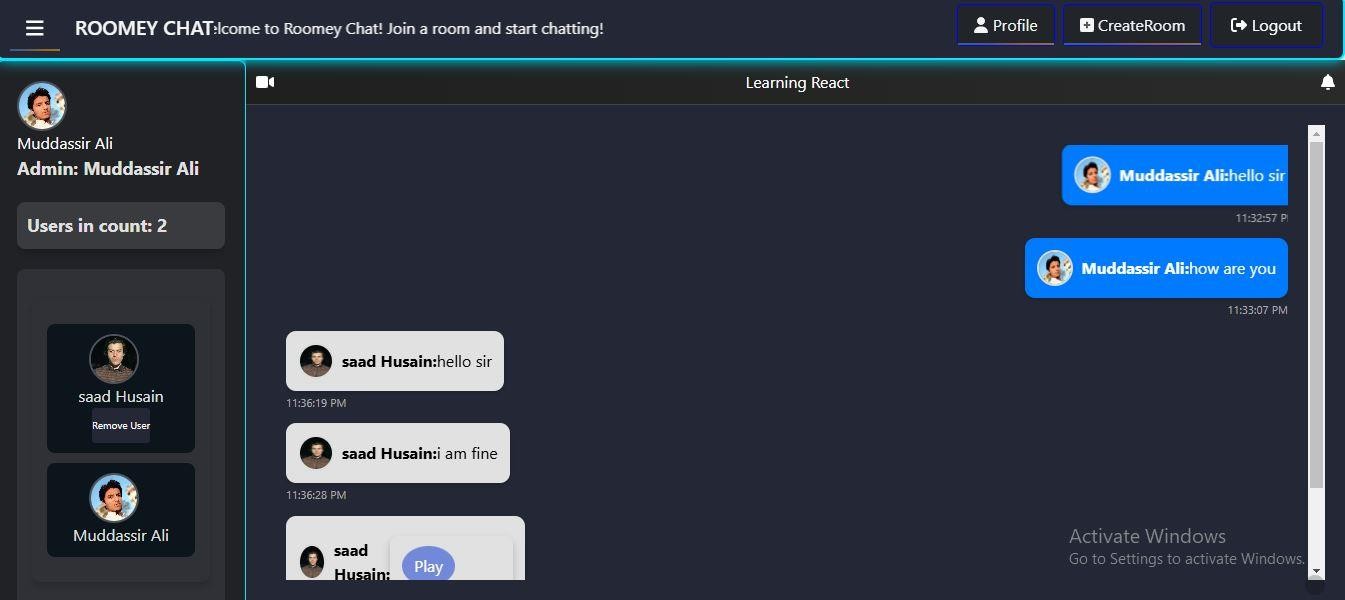
*Fig 5.4 Login page*

**

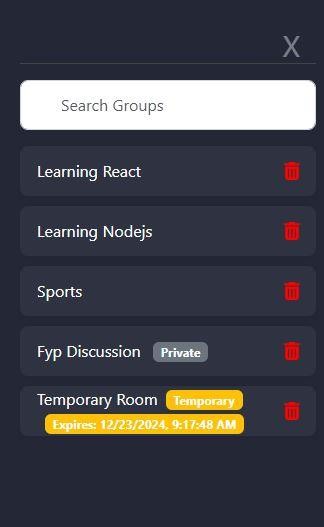
*Fig 5.5 Forget password screen*



*Fig 5.6 Reset password screen*

**

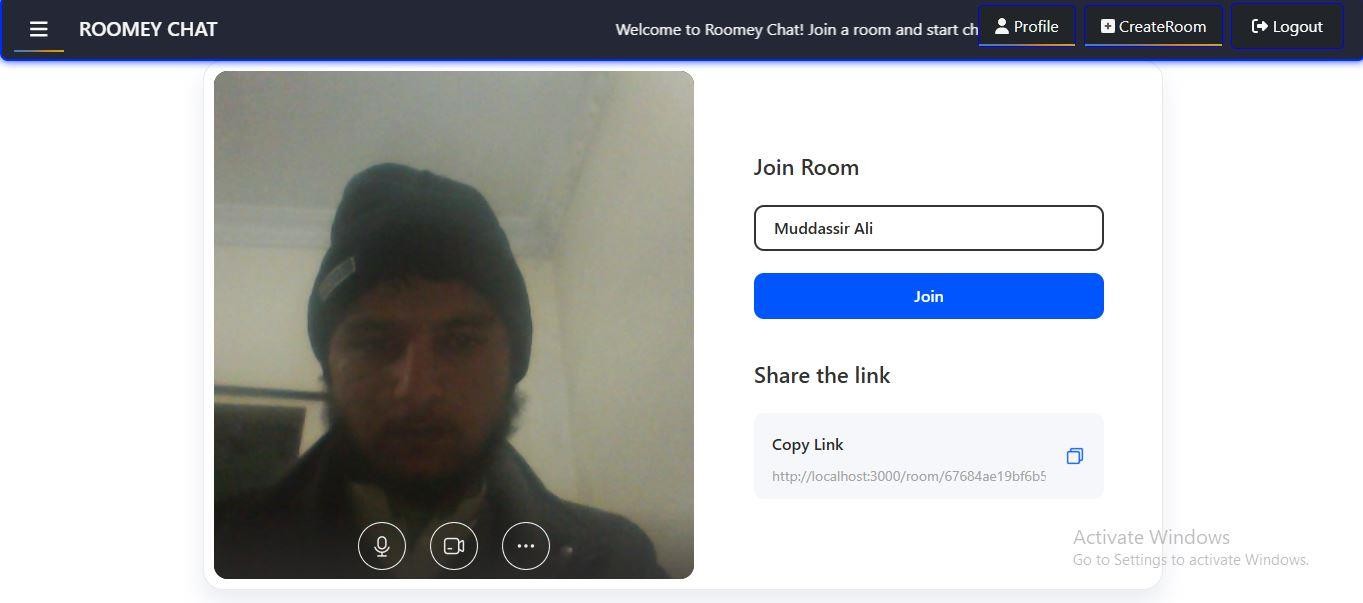
*Fig 5.7 messages page*



*Fig 5.8 List of Rooms page*



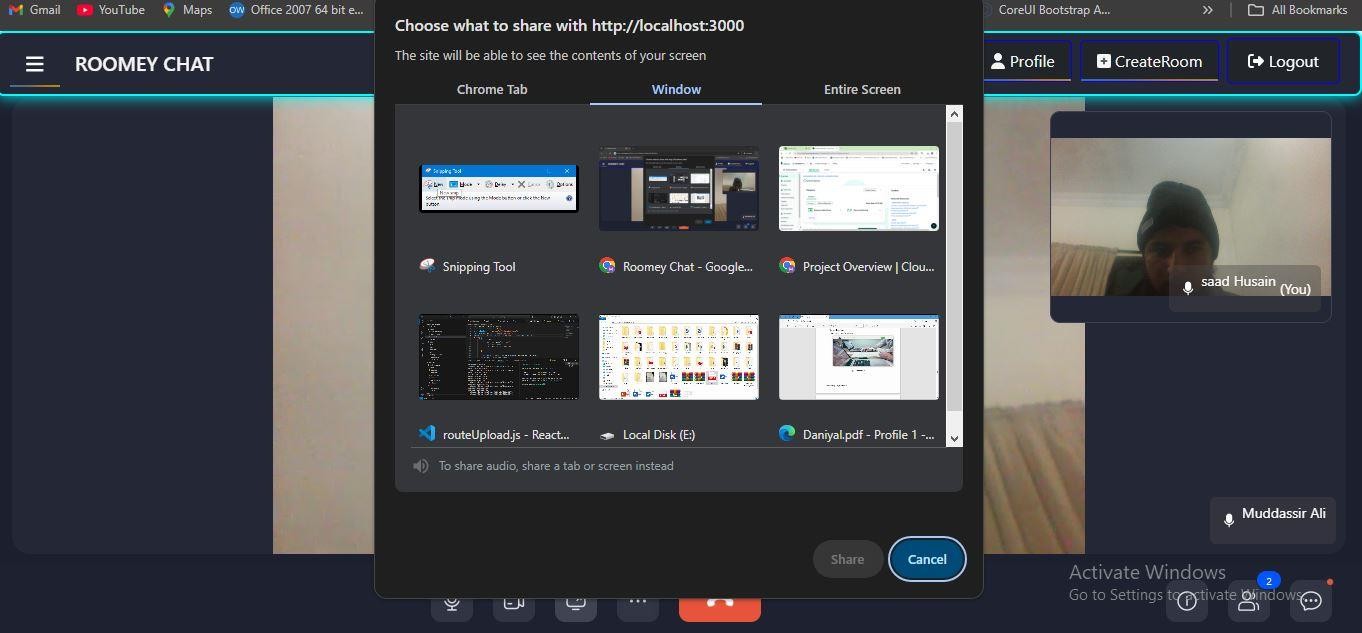
*Fig 5.9 List of users in group*



*Fig 5.10 Video call joining screen*

**

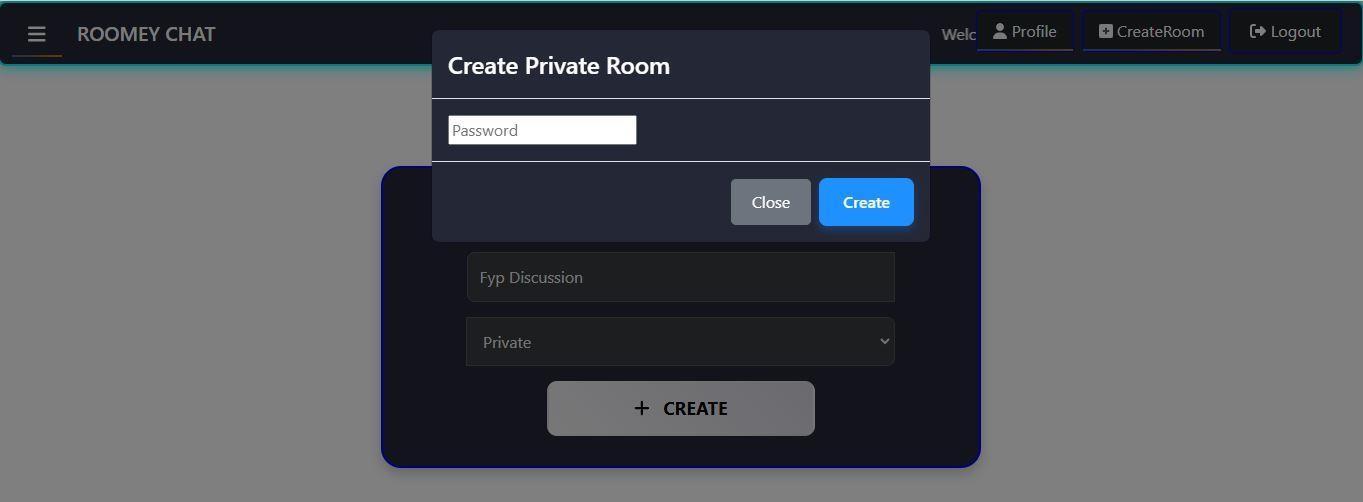
*Fig 5.11 Video calling screen*



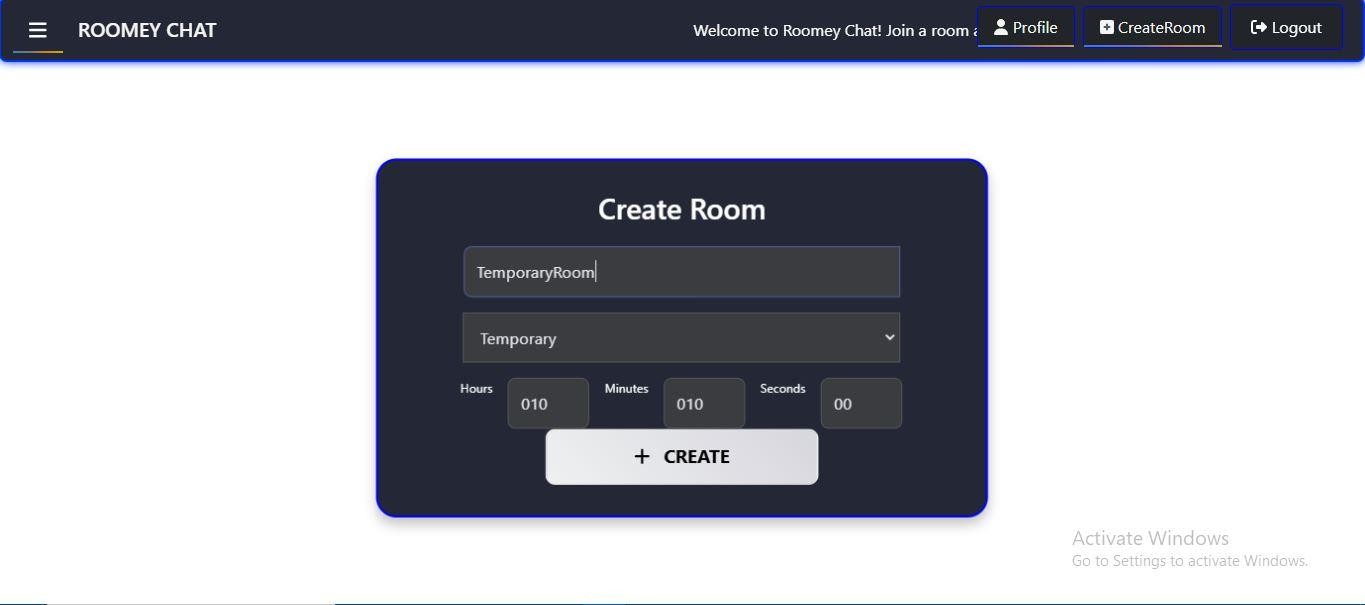
*Fig 5.12 Screen sharing window*

**

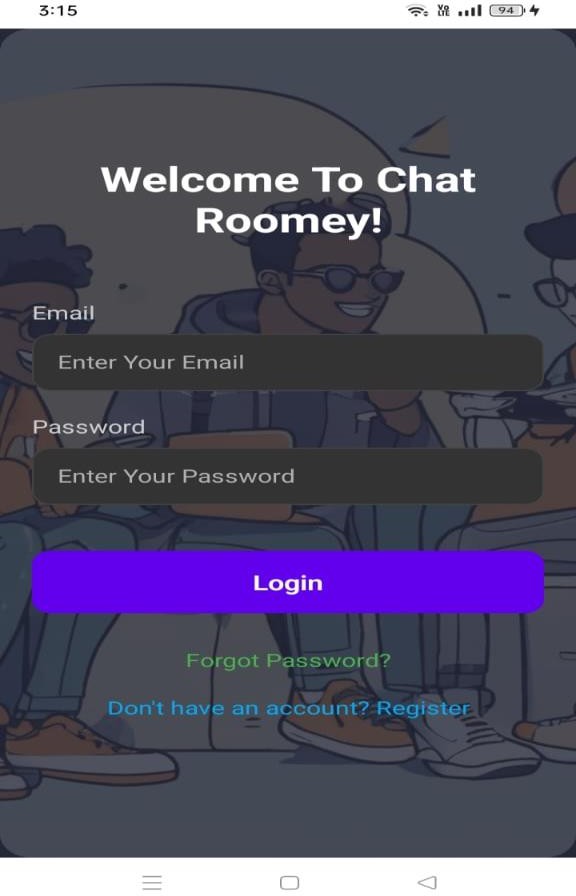
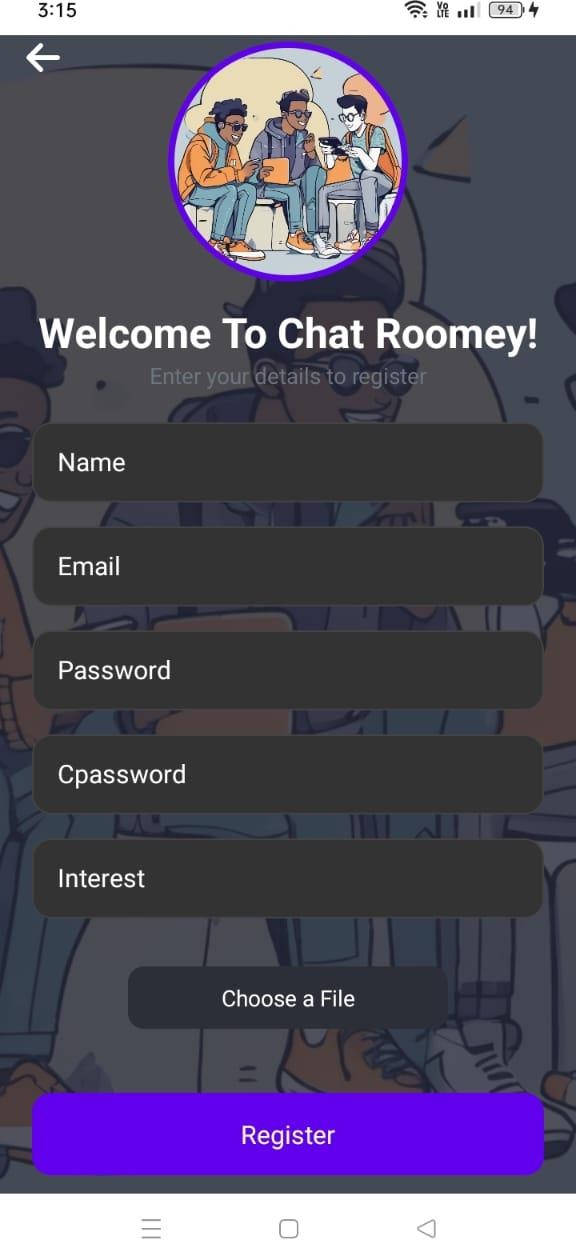
*Fig 5.13 Messaging in video call screen.*



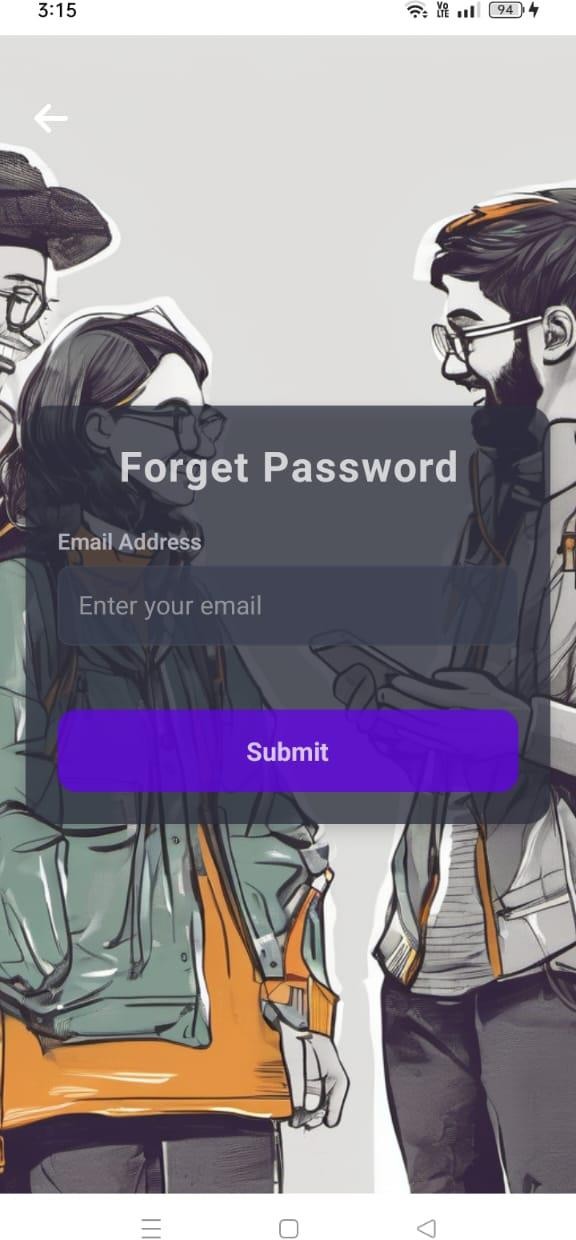
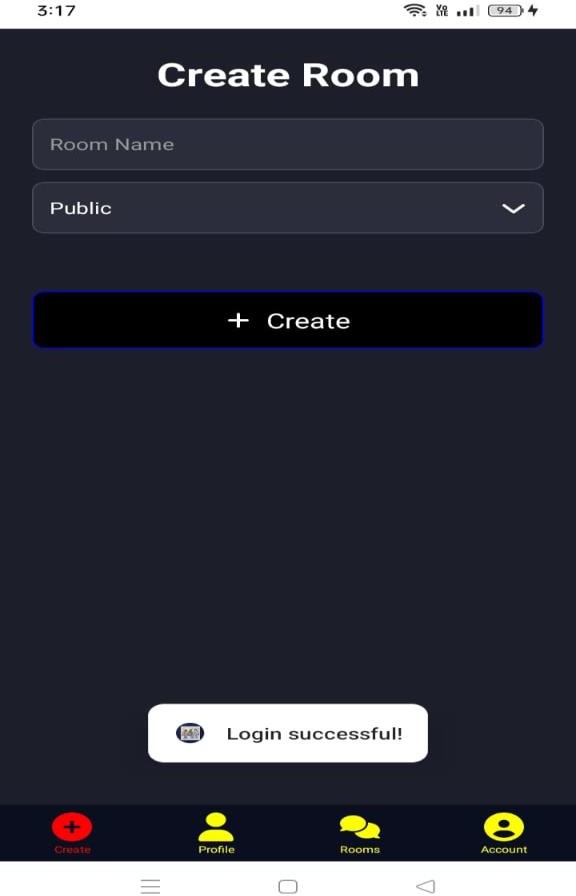
*Fig 5.14 Private Room screen*

**

*Fig 5.15 Temporary Room Screen*

*Fig 5.16 Login Screen Fig 5.17 Registration Screen*

*Fig 5.18 Forget password Screen Fig 5.19 Create Room Screen*

*Fig 5.20 Profile Management Screen Fig 5.21 Profile Management Screen*

# Testing and Evaluation

During software development, testing ensures that each component functions correctly, while evaluation assesses the software's overall performance and user satisfaction. Various types of testing, such as unit and integration testing, verify functionality, while usability testing and performance metrics gauge user experience and system efficiency

## Manual Testing

It ensures that software functions as expected from a user's perspective, identifying user interface issues, usability concerns, and functional bugs that might be missed by automated tests

### System testing

This table outlines the various aspects and stages of system testing conducted in the software development lifecycle.

*Table 6.1: System Testing Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Testcase/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | User Login - Valid Credentials | Email:  [muddassirali8089@gmail.com](mailto:muddassirali8089@gmail.com) Password: alsdfhadsf | Successfully log onto the main page of Roomy | Pass |
| 2 | Invalid Login Attempt | Email: [wrong@gmail.com](mailto:wrong@gmail.com)  Password: wrong password | Display error message incorrect username or password. | pass |
| 3 | Create Chat Room | Room name :Sports Select type : permanent | New chat room created and display in the list | Pass |
| 4 | Create Temporary Chat Room | User logged in. Room name : DSA  Select type : temporary  Enter time 10hours 10 min 5 sec | Temporary Room will be created and show in the list | Pass |
| 5 | Auto delete temporary chat room | Completion of the time | Room will automatically deleted and deleted from list | pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | Schedule Chat Room | Select the schedule date using calendar  19 dec 2024 | New schedule room created and display in the list | pass |
| 7 | Private Chat Room | User logged in. Room name : DSA Select type : Private  password:”1234” | New private room created and display in the list | pass |
| 8 | Join private room | Valid password | Private room join successfully | pass |
| 9 | Send Message in Chat Room | User logged in, Room: "Sports", Message: "Hello everyone" | Message sent and displayed in chat room. | Pass |
| 10 | Join Chat Room | User logged in, click on the room select the room | Successfully join the chat room. | Pass |
| 11 | Invite User to Chat Room | User logged in, and click on copy link button and send it the user | Invitation sent to the user, and they can join the room. | Pass |
| 12 | Search Chat Rooms | User logged in Search : "sports" | Display a list of rooms matching "sports". | Pass |
| 13 | Profile Customization | User logged in, Update interests and profile picture | Profile updated successfully with changes reflected on the profile page. | Pass |
| 14 | Video Call in Room | User logged in,  Click on the video icon and gave room Id video call is starting and other user click on join room | Successfully start and end the call with video enabled. | Pass |
| 15 | Room Deletion by Creator | User logged in, Room: "sports", Creator of room | Room deleted successfully. | Pass |
| 16 | User Logout | User logged in | Successfully log out and | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | return to the login screen. |  |
| 17 | Notification for New Message | User logged in, Room: "sports",  New message: "Welcome" | User receives a notification for a new message in the chat room. | Pass |
| 18 | Multiple Users in Room | Multiple users logged in, Room: “sports" | Multiple users can join and participate in the room without issues. | Pass |
| 19 | Display the list of join users in the room | User login and go to the group page | Group Join person names display in the list | Pass |

### Unit Testing

1. **User Authentication Module:**
   * **Testing Objective:** To ensure the edit profile form is working

*Table 6.2: Login Unit Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| UT- 001. | Registration Valid Input | Username: Muddassir Ali Email:  [muddassirali8089@gmail.co](mailto:muddassirali8089@gmail.com) [m](mailto:muddassirali8089@gmail.com)  Password: 123Abc@  confirm password: 123Abc@ | Registration successful and user navigate to login page | pass |
| UT- 002. | After selecting the "Login" button on the login form and entering the relevant information, confirm the user's login. | Email: [muddassirali8089@gmail.co](mailto:muddassirali8089@gmail.com) [m](mailto:muddassirali8089@gmail.com)  Password: 123Abc@ | successfully access the system roomy's main page | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UT- 003. | Invalid Login Attempt | Email: [wrong@gmail.com](mailto:wrong@gmail.com)  Password: wrong password | Display error: "Incorrect username or  password" | Pass |
| UT- 004. | User Login - Empty Fields | Email:“” Password: “” | Display error: "Fields cannot  be empty" | Pass |

### Profile Management Module

* + **Testing Objective:** To ensure the edit profile form is working properly

*Table 6.3: Profile Management Module Unit Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| UT- 001. | Profile Customization  - Valid | Update name and upload profile picture | Profile updated successfully | Pass |
| UT- 002. | Profile Customization  - Invalid  Picture | Upload unsupported file format (e.g., .exe) | Only png , jpg format accepted | pass |
| UT- 003. | Profile Customization  - Empty Bio | Leave bio blank | Donot leave the bio expty | pass |

### Chat Room Creation Module

* + **Testing Objective:** To ensure that chatroom creation module works properly

*Table 6.4: Chat Room Creation Module Test case*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| UT- 001. | Create Chat Room - Valid | Room name: "Sports", Type: Permanent | Chat room created and displayed in the list | Pass |
| UT- 003. | Create Chat Room - Empty  Name | Room name: “”, Type: Permanent | Display error:  "Room name cannot be empty" | pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UT- 004. | Schedule Chat Room | Select the schedule date using calendar  19 dec 2024 | New schedule  created and display in the list | pass |
| UT- 005 | Video Call in Room | User logged in,  Click on the video icon and gave room Id video call is starting and other user click on join room | Successfully start and end the call with video enabled. | Pass |

### Temporary Chat Room Module

* + **Testing Objective:** To ensure that the temporary chat room creation module works properly

*Table 6.5: temporary chat room creation Module Test case*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| UT- 001. | Create tempory Chat Room | Select: "temporary", Ending time  10 hours 10 min 10 sec | Rooms will we remove after the completion of time | pass |
| UT- 002. | Create Temporary Chat Room -  Invalid Time | Time: "0 hours" | Display error: "Invalid time duration" | Pass |
| UT- 003. | Auto Delete Temporary Chat Room | Completion of set time | Room automatically deleted from the  list | Pass |

### Invitation Management Module

* + **Testing Objective:** To ensure that the temporary chat room creation module working properly

*Table 6.6: invitation module Test case*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| UT- 001. | Inviting User to Chat Room | User logged in, and clicked on copy link button and send it the  user | Invitation sent to the user, and they can join the room. | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UT-  002. | Invite User -  Invalid Link | Use a corrupted or  incomplete link | Display error: "Invalid  invitation link | pass |

### Search Rooms module

* + **Testing Objective:** To ensure that the search room’s functionality works properly.

*Table 6.7: search module Test case*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| UT- 001. | Search Chat Rooms - Valid Input | Search term: “Sports” | Display list of rooms matching the term. | Pass |
| UT- 002. | Search Chat Rooms - Invalid Input | Search term: "wrong name" | Display message: "No rooms found | pass |

### join Chat Rooms Module

* + **Testing Objective:** To ensure that the join functionality is working properly.

*Table 6.8: join chat rooms module Test case*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| UT-  001. | Join Chat Room  - Valid | Select room:  "Sports" | Successfully join the  selected chat room | Pass |
| UT- 002. | Search Chat Rooms - Invalid  Input | Search term: "wrong name" | Display message: "No rooms found | pass |

### Room Scheduling

* + **Testing Objective:** To ensure that the room scheduling is working properly.

*Table 6.9: Room Scheduling module Test case*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UT- 001. | Schedule Chat Room - Valid | Schedule date: 19  Dec 2024 | New scheduled  room created and displayed | Pass |
| UT- 002. | Schedule Chat Room - Invalid  Date | Schedule date: "Past date" | Display error:  "Invalid date selected" | pass |

### Real-Time Communication Module

* + **Testing Objective:** To ensure that real-time communication is properly.

*Table 6.10:* real time communication *Test case*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| UT- 001. | Send Message in Chat Room | Room: “Sports”,  Message: "Hello everyone" | Display error:  "Message cannot be empty | Pass |
| UT- 002. | Send Message - Empty Input | Room: “Sports”, Message: "" | Display error:  "Invalid date selected" | pass |
| UT- 003. | Video Call - Valid | Start and join video call | Successfully start and end the video  call | Pass |
| UT- 004. | Video Call -  Invalid Room ID | Room ID: "Wrong123" | Display error: "Room not found" | Pass |
| UT- 005. | Notification for New Message | Room: "Sports",  New message: "Welcome" | User receives notification | Pass |
| UT- 006. | Multiple Users in Room | Multiple users  join and  participate | Room handles multiple users  without issues | Pass |

### Private chat rooms accessibility module

* + **Testing Objective:** To ensure that the private rooms are accessible properly.

*Table 6.11: private rooms accessibility module Test case*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| UT-  001. | Create private  chat rooms | Select rooms type | Private chat  rooms created | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Private  room name:” sports”  Password:”123” |  |  |
| UT-  002. | Join private chat  rooms | Enter valid  password:”1234” | Room joins  successfully | pass |

### Functional Testing

This step checks if each module works correctly and makes sure the system meets all the required features and goals.

### User Registration:

*Table 6.12: User Registration Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Register with Valid Details | Username: Muddassir Ali Email: muddassirali8089 @gmail.com Password: 123Abc@ Confirm Password: 123Abc@ | User successfully registers and navigates to login page | Pass |
| FT- 002 | Register with Invalid Email Format | Username: Muddassir Ali Email: muddassir@wron g  Password: 123Abc@ Confirm Password: 123Abc@ | Error: 'Invalid email format' | Pass |
| FT- 003 | Register with Empty Fields | Username:“” Email:“” Password:”” Confirm  Password: “” | Error: 'Fields cannot be empty' | Pass |
| FT- 004 | Register with Weak Password | Username: Muddassir Ali Email:”” muddassirali8089 @gmail.com Password: 123 Confirm Password: 123 | Error: 'Password is too weak' | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FT- 005 | Register with Existing Email | Username: Muddassir Ali Email: muddassirali8089 @gmail.com Password: 123Abc@ Confirm  Password:123Abc @ | Error: 'Email already register' | Pass |
| FT- 006 | Password and Confirm Password Mismatch | Password:123Abc @  Confirm Password:  1234Abc | Error: 'Passwords do not match' | Pass |

### User Login

*Table 6.13: User Login Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Enter your valid credentials to log in. | Email: [muddassirali8089@gmail.com](mailto:muddassirali8089@gmail.com) Password: 123Abc@ | Successfully logged into the main page | Pass |
| FT- 002 | Enter an invalid email  address to log in. | Email: [mu@wronggmail.com](mailto:mu@wronggmail.com) Password: 123Abc@ | Error: Invalid email format | Pass |
| FT- 003 | Enter an incorrect password to log in. | Email: [muddassirali8089@gmail.com](mailto:muddassirali8089@gmail.com) Password: wrongpassword | Error: 'Incorrect username or password' | Pass |
| FT- 004 | Login with Empty Fields | Email: “”  Password: ‘’” | Error: Fields cannot be empty | Pass |

### Profile management

*Table 6.14: User Profile management Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test**  **Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Update Profile with Valid  Details | Bio: “interest in coding” Profile Picture: Uploaded | Profile updated successfully | Pass |
| FT- 002 | Update Profile  with Empty Fields | Bio:“”  Profile Picture: Not uploaded | Error: Fields  cannot be empty | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FT- 003 | Upload Invalid Profile Picture  Format | Profile Picture: invalidfile.txt | Error: 'Invalid file format' | Pass |

### Invitation management

*Table 6.15: User invitation management Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Send group invitation to the user | Click on copy link and send it to the user | Copy link successfully and invitation send | Pass |
| UT- 002 | Accept invitation | Users click on the link and user join the group | User successfully join the room | Pass |

### User list in the group

*Table 6.16: User view list of users Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | View User List | Group: "Project A" | List of users displayed | Pass |

### Room creation

*Table 6.17: Room creation Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Create Room with Valid Details | Room Name: Sports Type: Public | Room created successfully | Pass |
| FT- 002 | Create Room with Empty Name | Room Name: "" | Error: Room name cannot be empty | Pass |

### Search Rooms

*Table 6.18: Search Rooms Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Search Rooms | User logged in  Search : "sports" | Display a list of rooms matching "sports". | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FT- 002 | Search Rooms | User logged in  Search : “” | Room not found | pass |

### Delete Rooms

*Table 6.19: Delete Rooms Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Room Deletion by Creator | User logged in, Room: "sports", Creator of room  Click on delete icon | Room deleted successfully. | Pass |

### Temporary Room creation

*Table 6.20: Temporary Room creation Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Create Temporary Room | Room Name: "Temporary Room" Duration: 2 hours 2  min 30 sec | Temporary room created successfully | Pass |
| FT- 002 | Auto Delete Temporary Room | Time expires for temporary room | Room automatically deleted | Pass |

### Private Room creation

*Table 6.21: Private Room creation Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Create Private Room | Room Name:  "Private Chat" Type private | Room created successfully | Pass |
| UT- 026 | Access with wrong password | Access without password | Error: ‘enter a valid password’ | Pass |

### Schedule Room

*Table 6.22: Schedule Rooms Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FT-001 | Schedule Room | Room Name: "Scheduled Room"  Date: 19 dec  2024 | Room scheduled successfully | Pass |
| FT-002 | Schedule Room with Past Date | Date: previous date | Error: "Invalid date" | Pass |

### Video calling

*Table 6.23: Video calling Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT-001 | Start Video Call | Click on "Start Call icon" | Video call started | Pass |
| FT-002 | Join Video Call | Click on "Start Call icon and  join " | Successfully joined the video  call | Pass |

### Messaging audio/text

*Table 6.24: Messaging audio/Text Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT-001 | Send Text  Message | Message:  “Hello” | Message sent  successfully | Pass |
| FT-002 | Send Audio Message | Record and send audio message | Audio message sent successfully | Pass |

### Logout functionality

*Table 6.25: Logout Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| FT- 001 | Logout from Main Page | Click on "Logout" | User logged out and redirected to login page | Pass |

### 6.1.3. Integration Testing

Table 6.23 shows the integration testing

*Table 6.26: Integration Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| IT- 001 | Successful Registration and Login | Registration details: Email: muddassirali80  [89@gmail.com](mailto:89@gmail.com) | Registration and login succeed, user | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | , Password: 123Abc@, Confirm Password:  123Abc@ | navigated to the dashboard |  |
| IT- 002 | Registration and Login  with Invalid Email | Email:  muddassirali@ | Registration  fails; error: | Pass |
|  |  | invalid, | 'Invalid email |  |
|  |  | Password: | format' |  |
|  |  | 123Abc@ |  |  |
| IT- 003 | Registration with Valid Details but Login with Wrong Password | Registration succeeds.  Login details: Email:  muddassirali80 | Login fails; error: 'Incorrect username or password' | Pass |
|  |  | [89@gmail.com](mailto:89@gmail.com) |  |  |
|  |  | Password: |  |  |
|  |  | wrong |  |  |
|  |  | password |  |  |

## Automated Testing:

Here is the automated test for Roomy

### Tools used:

Table 6.27 shows the automated testing.

*Table 6.27: automated testing*

|  |  |  |  |
| --- | --- | --- | --- |
| **Tool Name** | **Tool Description** | **Applied on [list of related tests cases / FR / NFR]** | **Results** |
| Selenium WebDriver | A framework for automating web applications for testing purposes. | login functionality,  Form submissions, Navigation testing. | Pass |
| Postman | A collaboration  platform for API development. | API endpoints testing. | Pass |

# Conclusion and Future Work

This chapter concludes the project and highlights future work.

## Conclusion

In conclusion, the development of the Roomy platform provides a versatile solution to enhance communication, foster collaboration, and facilitate efficient interaction through text, audio, and video chat rooms. By utilizing modern technologies and following software engineering best practices, Roomy addresses the limitations of traditional communication platforms, delivering a user-friendly and feature-rich experience for individuals and groups. Throughout the development process, the project incorporated key concepts from various course modules, such as web development, database management, and secure authentication. Rigorous testing and evaluation ensured the system’s reliability, functionality, and security, establishing Roomy as a robust platform for dynamic and engaging online interactions

## Future Work

While Roomy offers an innovative and efficient platform for communication and collaboration, there are several avenues for future work and enhancements: Integrate real-time language translation features to support multilingual communication, making Roomy accessible to a global audience. Develop a dedicated mobile application to provide users with seamless access to Roomy’s features on-the-go, enhancing convenience and usability. Implement AI tools for real-time content moderation and user behavior analysis to ensure a safe and productive communication environment. Enable Roomy to support large-scale virtual conferences and events with features like attendee management and breakout rooms. Incorporate advanced scheduling options with third-party calendar integration to streamline event planning and notifications. Establish a feedback loop to gather user input and perform iterative system updates, ensuring Roomy evolves to meet changing user needs. By exploring these enhancements, Roomy can continue to grow as a leading communication platform, offering versatile solutions for individuals and organizations alike.

# References

References to any book, journal paper or website should properly be acknowledged. Please consistently follow the style. The following are few examples of different resources i.e. journal article, book, and website.

1. John Smith, Building Chat Applications with WebSocket’s and Node.js, [https://developer.mozilla.org/en-US/docs/Web/API/WebSockets\_API,](https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API) Last date accessed: December 21, 2024.
2. Page Author, WebSocket Basics for Real-Time Applications, https://socket.io/docs, Last date accessed: December 21, 2024.
3. Pooja Sharma, An Overview of RESTful APIs in Chat Applications, (2021), International Journal of Software Development, Vol. 10, Issue 3, pp 50-55.

Brian Cooke, Scaling Real-Time Chat Systems, (2020), Real-Time Systems Engineering Journal, pp 120-130

71