
Design Document

for

Lostify

Version 1.0

Prepared by

Group #6

Group Name: hAPPy Developers

Aayush Kumar	230027	aayushk23@iitk.ac.in
Aman Raj	230116	amanraj23@iitk.ac.in
Anirudh Cheriyanaseri Bijay	230140	anirudhcb23@iitk.ac.in
Ayush Patel	220269	ayushpatel22@iitk.ac.in
Krishna Kumayu	230576	kkrishna23@iitk.ac.in
Marada Teja Satvik	230636	maradateja23@iitk.ac.in
Satwik Raj Wadhwa	230937	satwikraj23@iitk.ac.in
Shaurya Johari	230959	shauryaj23@iitk.ac.in
Somaiya Narayan Aniruddh	231019	snarayana23@iitk.ac.in
Vinay Chavan	231155	vinay23@iitk.ac.in

Course: CS253

Mentor TA: Jeswaanath Gogula

Date: 06 February 2025

Contents

CONTENTS.....	II
REVISIONS.....	III
1 CONTEXT DESIGN.....	1
1.1 CONTEXT MODEL	1
1.2 HUMAN INTERFACE DESIGN	1
2 ARCHITECTURE DESIGN	10
3 OBJECT ORIENTED DESIGN	12
3.1 USE CASE DIAGRAM	12
3.2 CLASS DIAGRAM	13
3.3 SEQUENCE DIAGRAMS.....	14
3.4 STATE DIAGRAM	26
4 PROJECT PLAN.....	28
4.1 TIMELINE	28
4.2 DEVELOPMENT PHASES.....	29
APPENDIX A – GROUP LOG	31

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Aayush Kumar Aman Raj Anirudh Cheriyachanaseri Bijay Ayush Patel Krishna Kumayu Marada Teja Satvik Satwik Raj Wadhwa Shaurya Johari Somaiya Narayan Aniruddh Vinay Chavan	Preliminary draft	06/02/25

1 Context Design

1.1 Context Model

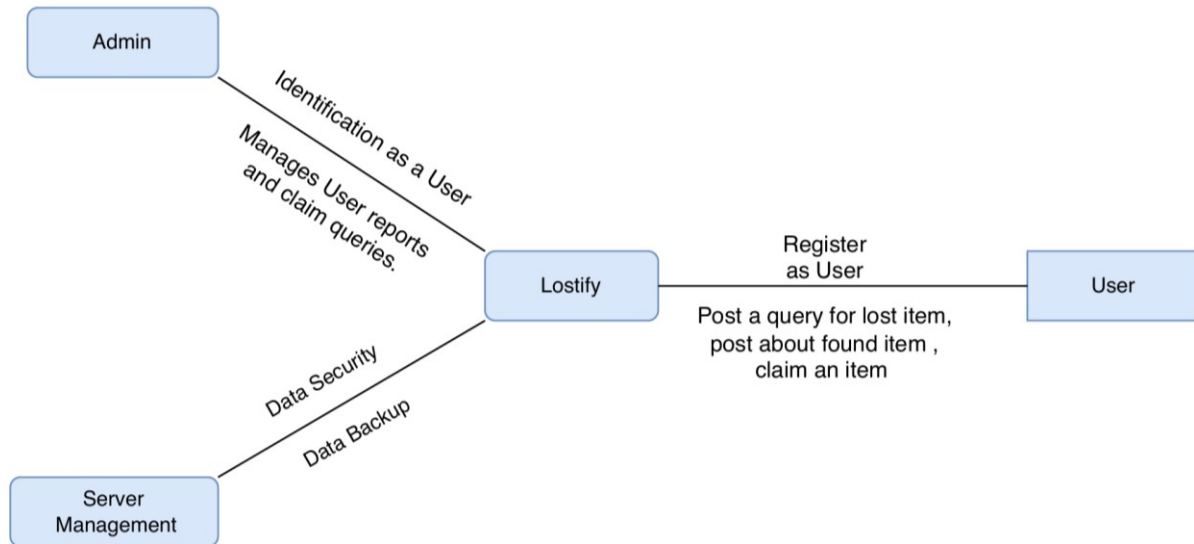


Figure 1: Context model

1.2 Human Interface Design

1.2.1 User Flow Description

The context model in the image represents the functional overview of a system named Lostify, which facilitates the management of lost and found items. The diagram consists of four main entities:

1. **Lostify (Central System):** The core platform that enables interactions between users, administrators, and server management.
2. **User:**
 - a. Registers as a user.
 - b. Can post queries for lost items.
 - c. Can post information about found items.
 - d. Can claim a found item.
3. **Admin:**
 - a. Identifies users.
 - b. Manages user reports and claim queries.

4. Server Management:

- Ensures data security.
- Handles data backup.
- The relationships between these entities define how Lostify operates, with users interacting with the platform to report and claim lost items, while administrators manage user reports.

1.2.2 User Flow Design

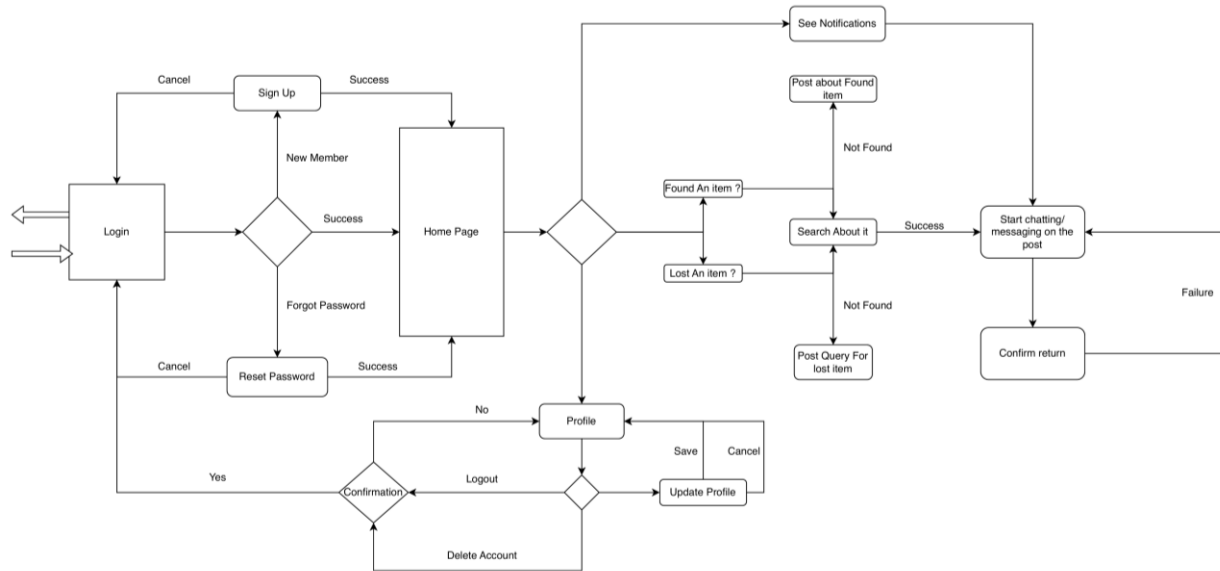


Figure 2: User flow diagram

1.2.2.1 Sign Up Page

The figure displays three sequential screens of a mobile application's sign-up process, all featuring a light blue background with abstract blue and teal geometric shapes.

Screen 1: Sign up
The title is "Sign up" with the subtitle "Create an account to get started". It contains four input fields: "Name" (with placeholder "Username"), "Email Address" (with placeholder "name@email.com"), "Password" (with placeholder "Create a password" and a strength indicator icon), and "Confirm password" (with placeholder "Confirm password" and a strength indicator icon). Below these is a checkbox labeled "I've read and agree with the [Terms and Conditions](#) and the [Privacy Policy](#)". At the bottom is a blue "Sign Up" button.

Screen 2: Create Profile
The title is "Create Profile". It contains five input fields: "Name *" (with placeholder "Display name"), "Phone number" (with placeholder "Contact number"), "Campus Address *" (with placeholder "Institute residential address" and a strength indicator icon), "Designation *" (with placeholder "Designation"), and "PF/Roll no." (with placeholder "PF/Roll no."). Below these is a "Profile Photo" section with an "Upload Image" button and an image icon. At the bottom is a blue "Get OTP" button. A red note at the very bottom states "Mandatory fields are marked with *".

Screen 3: Enter confirmation code
The title is "Enter confirmation code" with the subtitle "A 4-digit code was sent to your registered email address". It features four empty square boxes for entering the code. Below these is a blue "Resend code" button and a blue "Continue" button.

Figure 3: Sign up page

1.2.2.2 Login Page

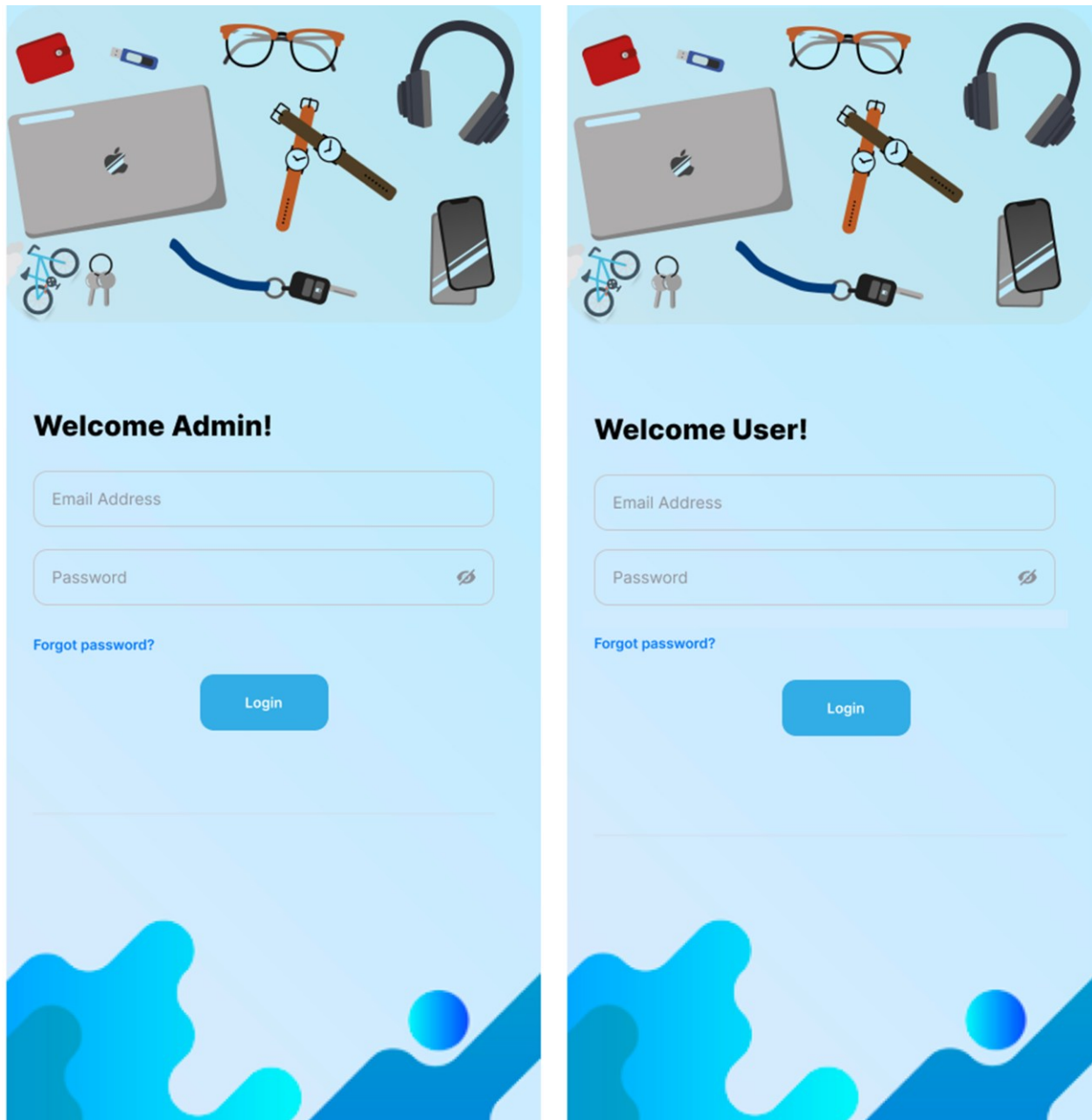


Figure 4: Login page

1.2.2.3 Homepage

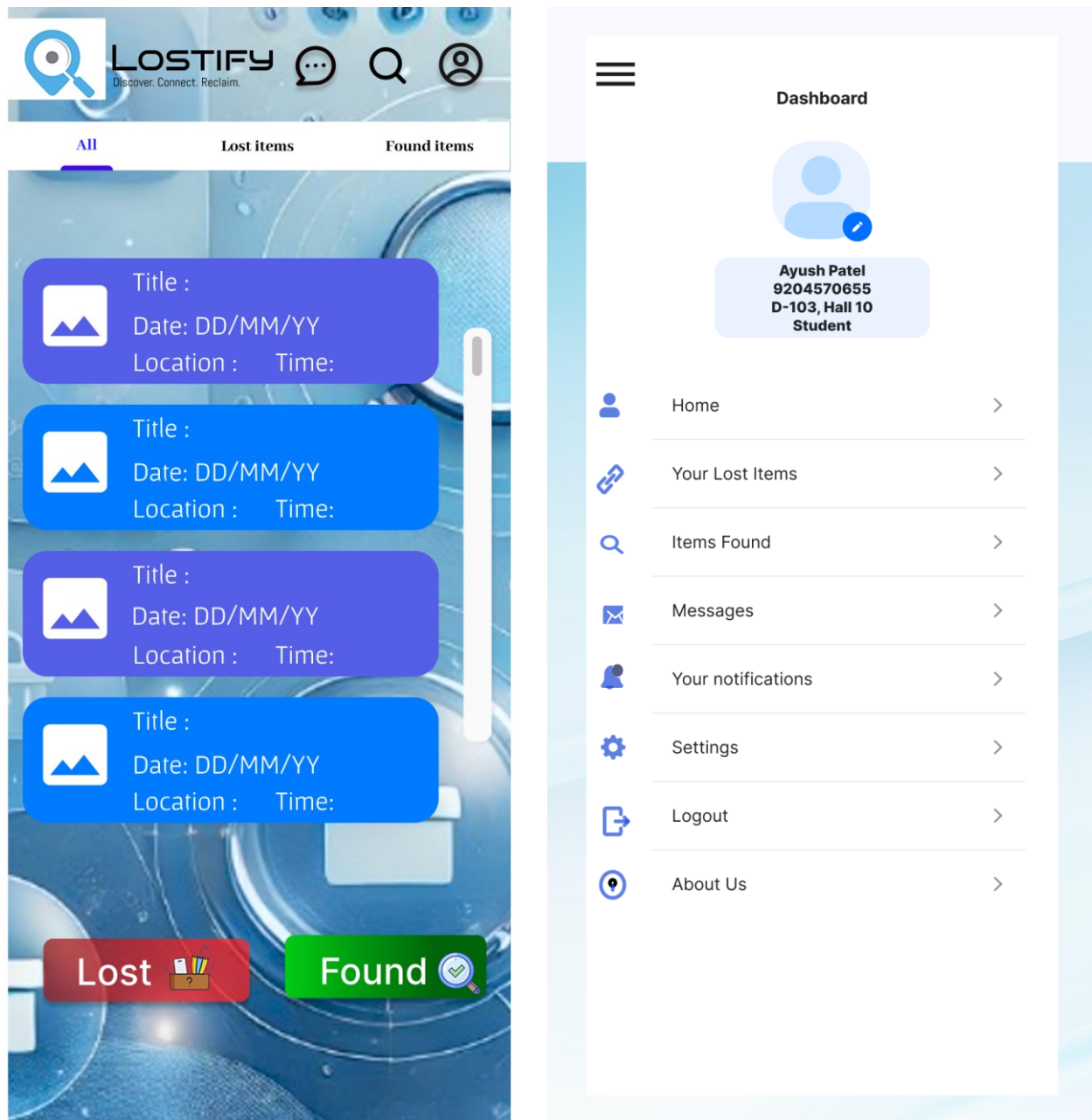


Figure 5: Homepage

1.2.2.4 Own Posts Page

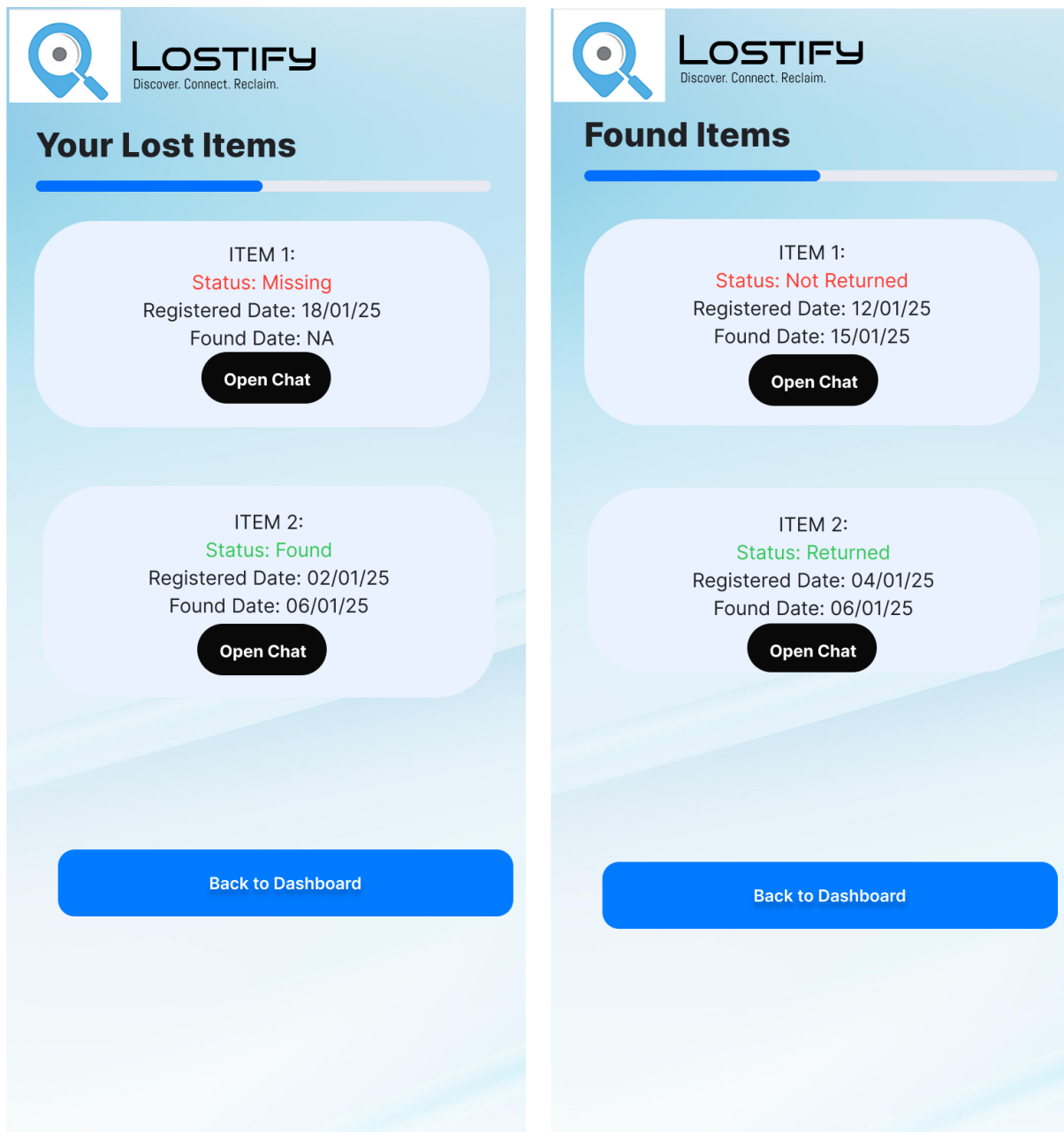
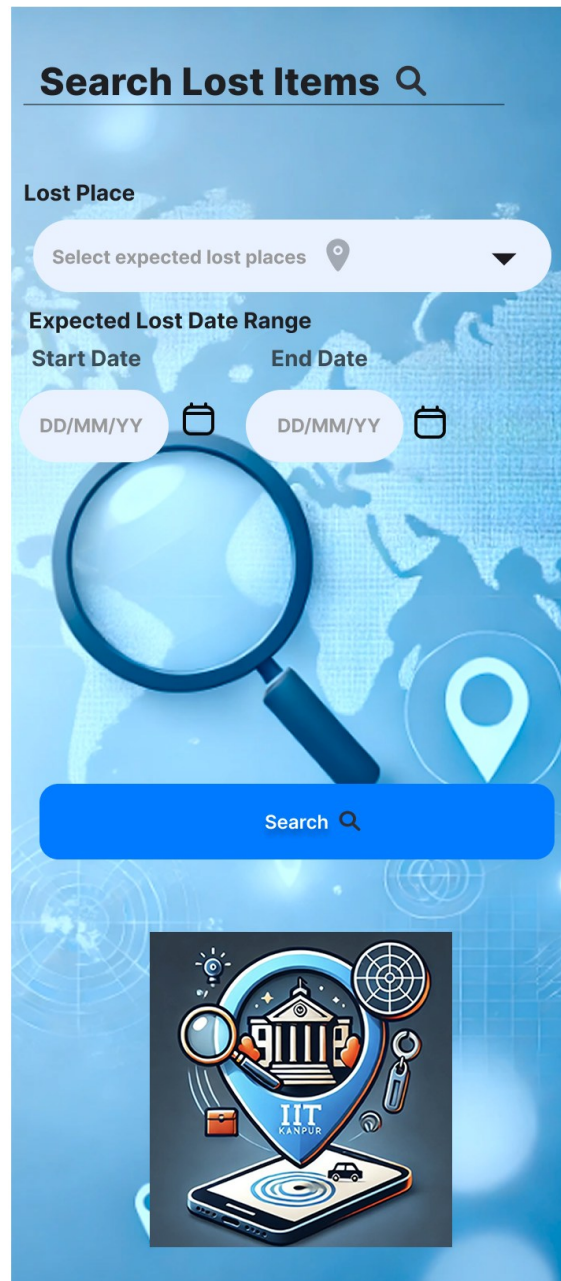


Figure 6: Posts created by the user

1.2.2.5 Search Page



The image shows a mobile application interface for searching lost items. The background is a blue world map with a magnifying glass icon. The title "Search Lost Items" is at the top with a magnifying glass icon. Below it, there is a "Lost Place" section with a dropdown menu labeled "Select expected lost places". Underneath is the "Expected Lost Date Range" section, which includes "Start Date" and "End Date" fields, each with a "DD/MM/YY" placeholder and a calendar icon. A large blue "Search" button with a magnifying glass icon is positioned below the date range fields. At the bottom, there is a square graphic featuring the IIT Kanpur logo, a magnifying glass, a key, a smartphone with a car icon, and a target symbol.

Search Lost Items 🔍

Lost Place

Select expected lost places 📍 ▼

Expected Lost Date Range

Start Date **End Date**

DD/MM/YY 📅 DD/MM/YY 📅

Search 🔍

IIT KANPUR

Figure 7: Search page

1.2.2.6 Post Lost Item

The figure displays two sequential mobile application screens for reporting a lost item. Both screens have a blue header with a magnifying glass icon and the title 'Lost an Item'.

Left Screen: Add Title and Description

- Section: **Add Title and Description**
- Form: A text input field labeled 'Title'.
- Form: A larger text area labeled 'Add Description...'.
- Section: **Upload Image**
- Form: A button with an image icon and an upload arrow icon.
- Form: A blue button labeled 'Next'.

Right Screen: Add Location, Time and Date

- Section: **Add Location, Time and Date**
- Form: A dropdown menu labeled 'Location'.
- Form: A list of location options with checkboxes: 'Hall XYZ', 'Library', 'RM', and 'IME'.
- Form: A button labeled 'Time' with a clock icon.
- Form: A button labeled 'Date' with a calendar icon.
- Form: A blue button labeled 'Post'.

Figure 8: Create a post for a lost item

1.2.2.7 Post Found Item

The figure displays three sequential mobile app screens for posting a found item, all featuring a blue background with a faint circular graphic of two hands holding an object.

- Screen 1: Found an Item**
 - Header: "Found an Item" with a magnifying glass icon.
 - Section: "Upload Image" with a large white image placeholder and a small image icon.
 - Footer: "Check Found item" with four circular progress indicators (1st is active) and a blue "Next" button.
- Screen 2: Found an Item**
 - Header: "Found an Item" with a magnifying glass icon.
 - Section: "Add Title and Description" with a "Title" input field and a "Found item Description" text area.
 - Footer: "Check Found item" with four circular progress indicators (2nd is active) and a blue "Next" button.
- Screen 3: Found an Item**
 - Header: "Found an Item" with a magnifying glass icon.
 - Section: "Location where item was found" with a "Location" dropdown menu.
 - Section: "Date and time of find" with "DD/MM/YY" and "Time" dropdown menus.
 - Section: "Present location of the item" with a "Present Location" dropdown menu.
 - Footer: "Check Found item" with four circular progress indicators (3rd is active) and a blue "Next" button.

Figure 9: Create a post for a found item

2 Architecture Design

Lostify is a mobile application implementing a client-server architecture.

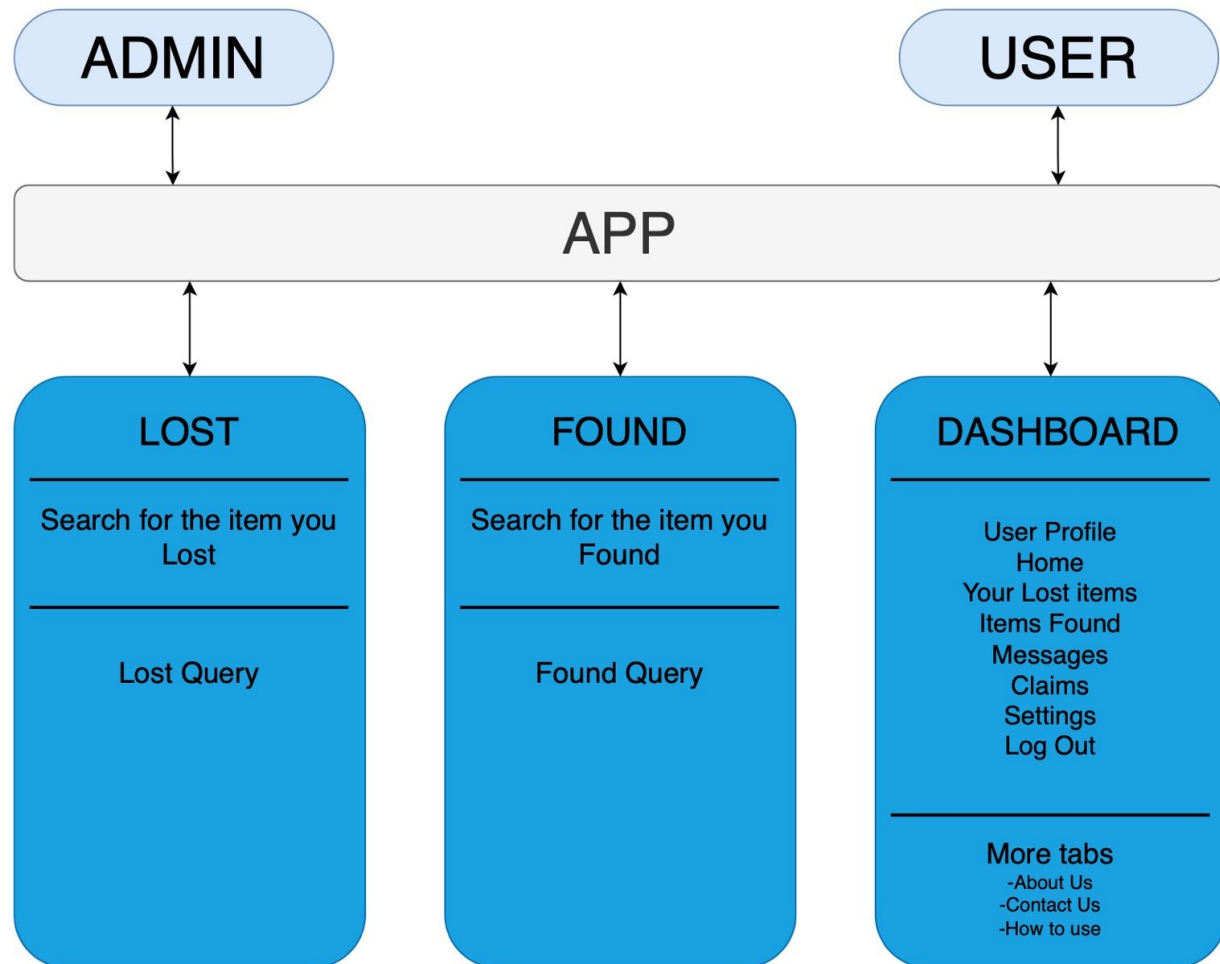


Figure 10: Architecture design diagram

The client-server architecture enables decentralised access from client devices to a centralised, shared database from multiple locations. The setup ensures that all users can access the same information since only data stored at the server needs to be kept up to date. This allows seamless searches, reporting and claim management across the platform.

Significant advantages of the client-server model include:

- **Centralised services:** Core functionality such as item posting and user authentication are centralised, mitigating the need for redundant implementations on individual client devices.
- **Real-time updates:** Since operations interact with a shared database, updates (such as new posts) are immediately reflected for all users.

Some downsides of the client-server model include:

- **Single point of failure:** If the central server goes offline, some of the core functionality of the application becomes unavailable.
- **Network dependency:** The application requires a stable internet connection to function effectively, which may prove a limitation in certain areas.

The model meets the security requirements as database access requires authentication and physical access to the server is not possible as the backend is hosted on the cloud.

However, performance requirements are limited by the performance and capacity of the server, with possible latency in response under high load. Furthermore, changes to the backend may necessitate the deployment of updates to the frontend as well, complicating maintenance and version compatibility.

3 Object Oriented Design

3.1 Use Case Diagram

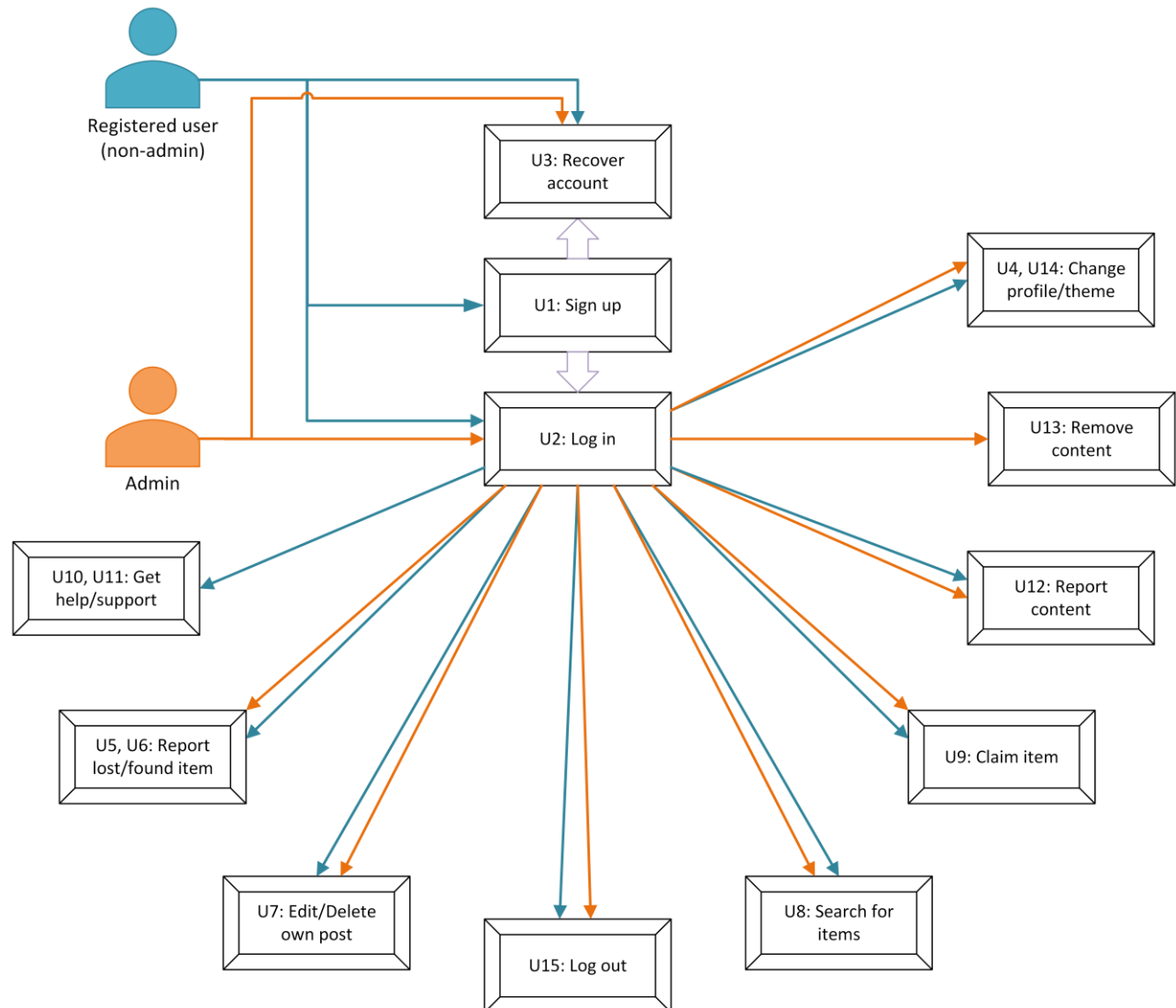


Figure 11: Use case diagram

Registered users shall fall into two roles: admin and non-admin. Both admins and non-admins can:

- Sign up and log in.
- Post lost and found items and edit them.
- Search for posts.
- Open a chat to claim an item.
- Report content.
- Customise theme and edit profile.

Admins, in addition to the above, can remove inappropriate content.

3.2 Class Diagram

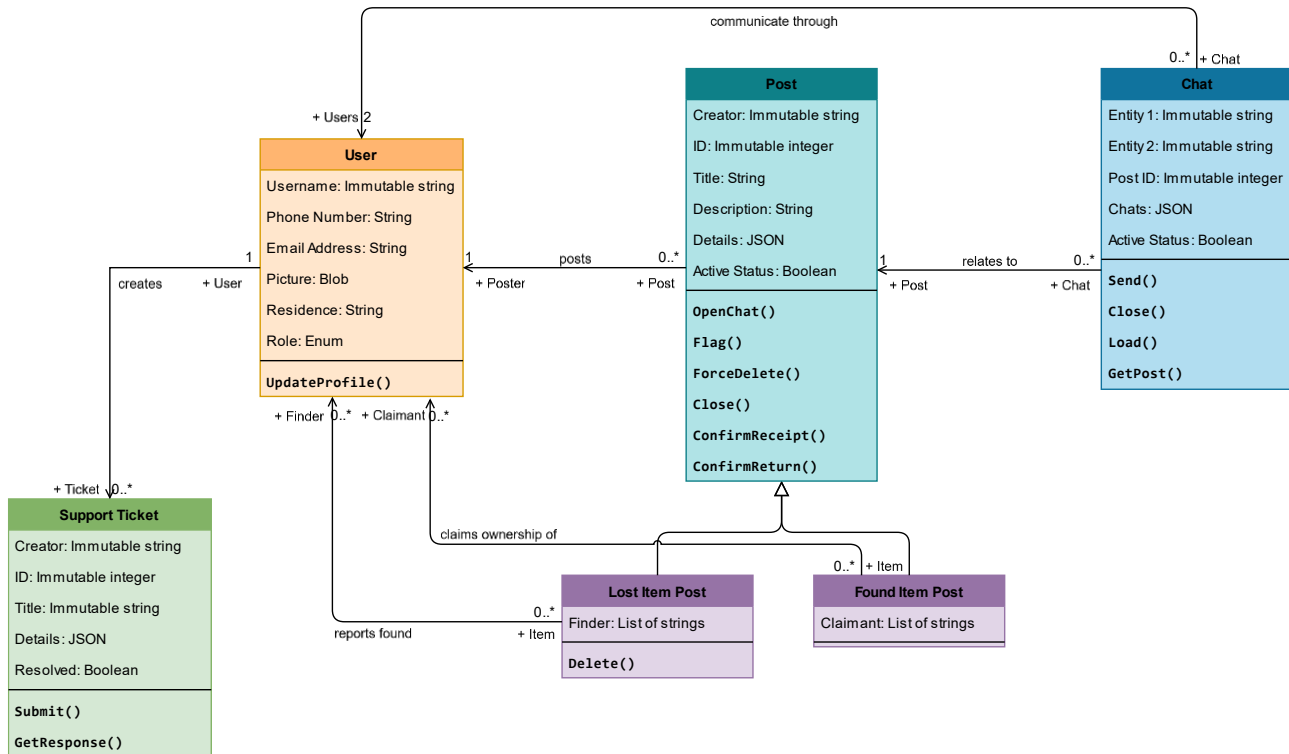


Figure 12: Class diagram

The application follows an object-oriented design paradigm. The following entities are represented as classes:

- Registered users.
- Posts for lost items.
- Posts for found items.
- Chats.
- Tickets opened to request support.

Posts for lost and found items inherit from a common base class.

Data members of each class are accessed through getters and setters, with the members themselves being private, thus enabling information hiding. Public methods are used to modify state, with requests to the server performed by the methods.

3.3 Sequence Diagrams

3.3.1 Sign Up

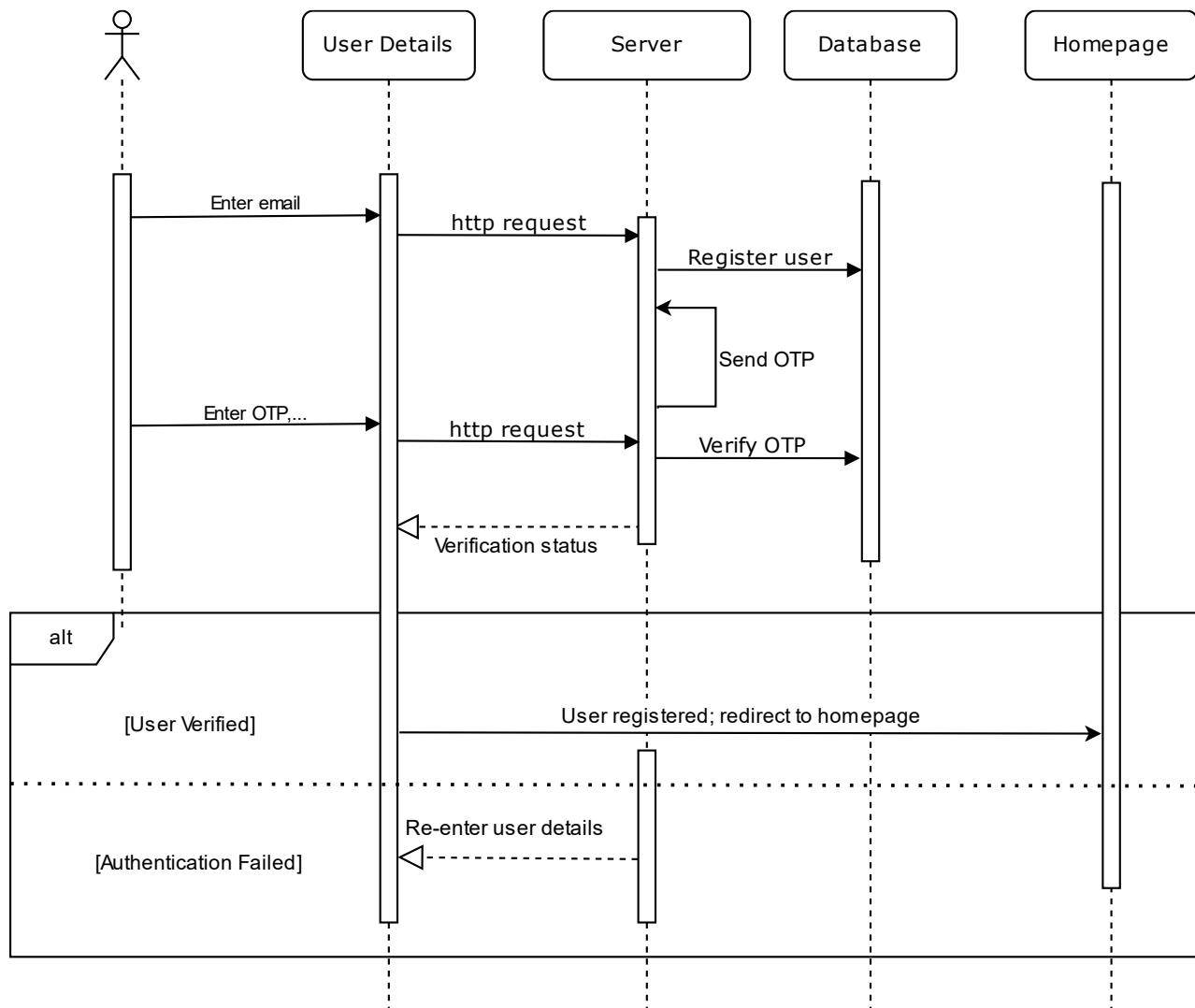


Figure 13: Sequence diagram for signing up

Purpose:

The signup page is designed to facilitate seamless access to Lostify for users affiliated with IIT Kanpur, including students, faculty, and staff. It ensures that only authorised individuals can report, track, and manage lost and found items within the IITK community.

User authentication, role-based access of admins and users, grants appropriate permissions for managing lost and found queries. **Community-driven security** ensures that only IITK-affiliated individuals can participate, enhancing trust and reliability in the platform.

3.3.2 Login

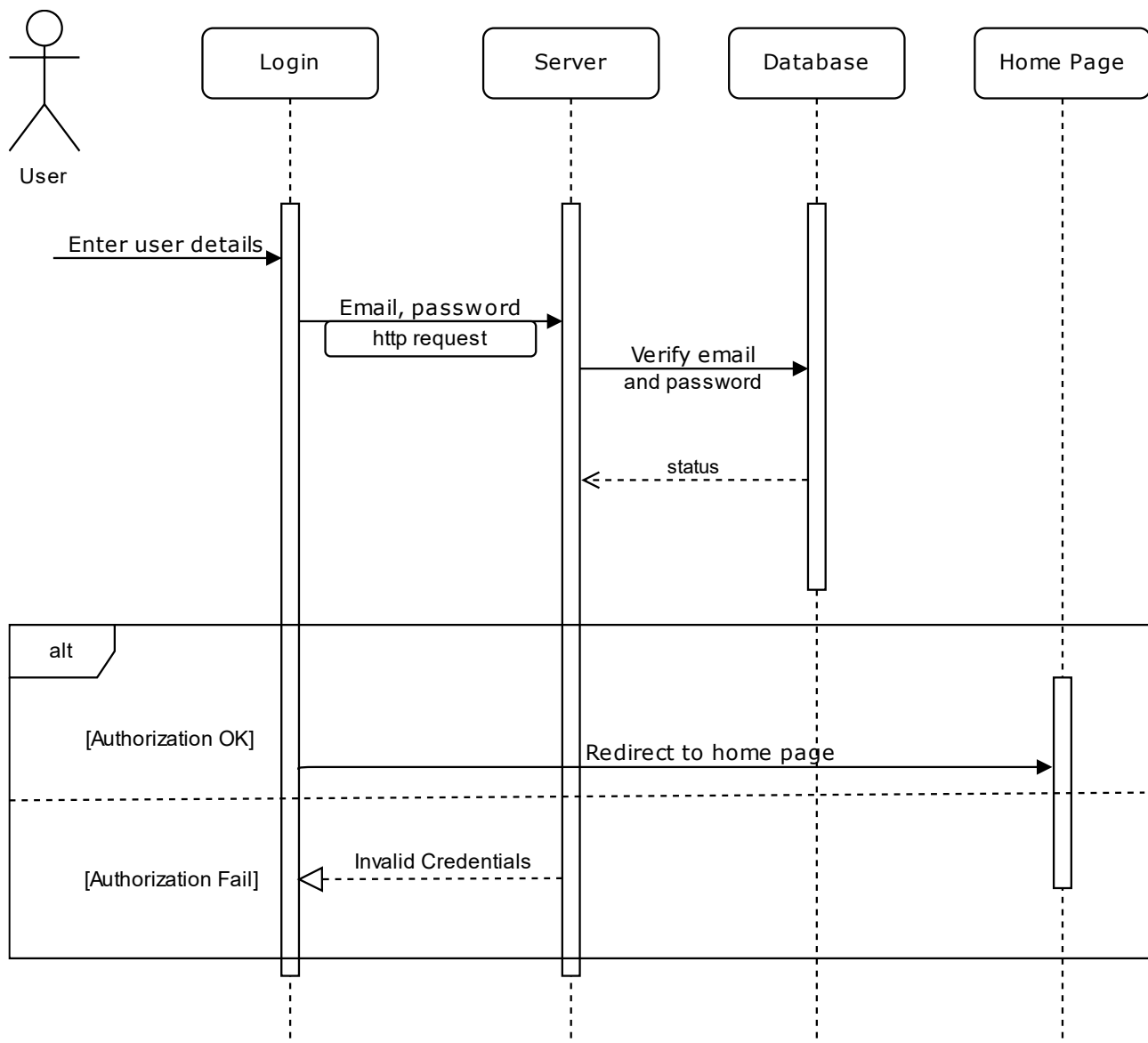


Figure 14: Sequence diagram for logging in

Purpose:

The login page serves as the gateway for authenticated users to access Lostify, ensuring secure and role-based entry into the system.

- **Secure authentication:** Restricts access to registered IITK users.
- **Role-based access:** Admins manage the system; users report and track items.
- **Session management:** Allows users to continue activities and track queries.
- **Efficient navigation:** Directs users to Dashboard, Lost, or Found sections.

3.3.3 Post Lost/Found Item

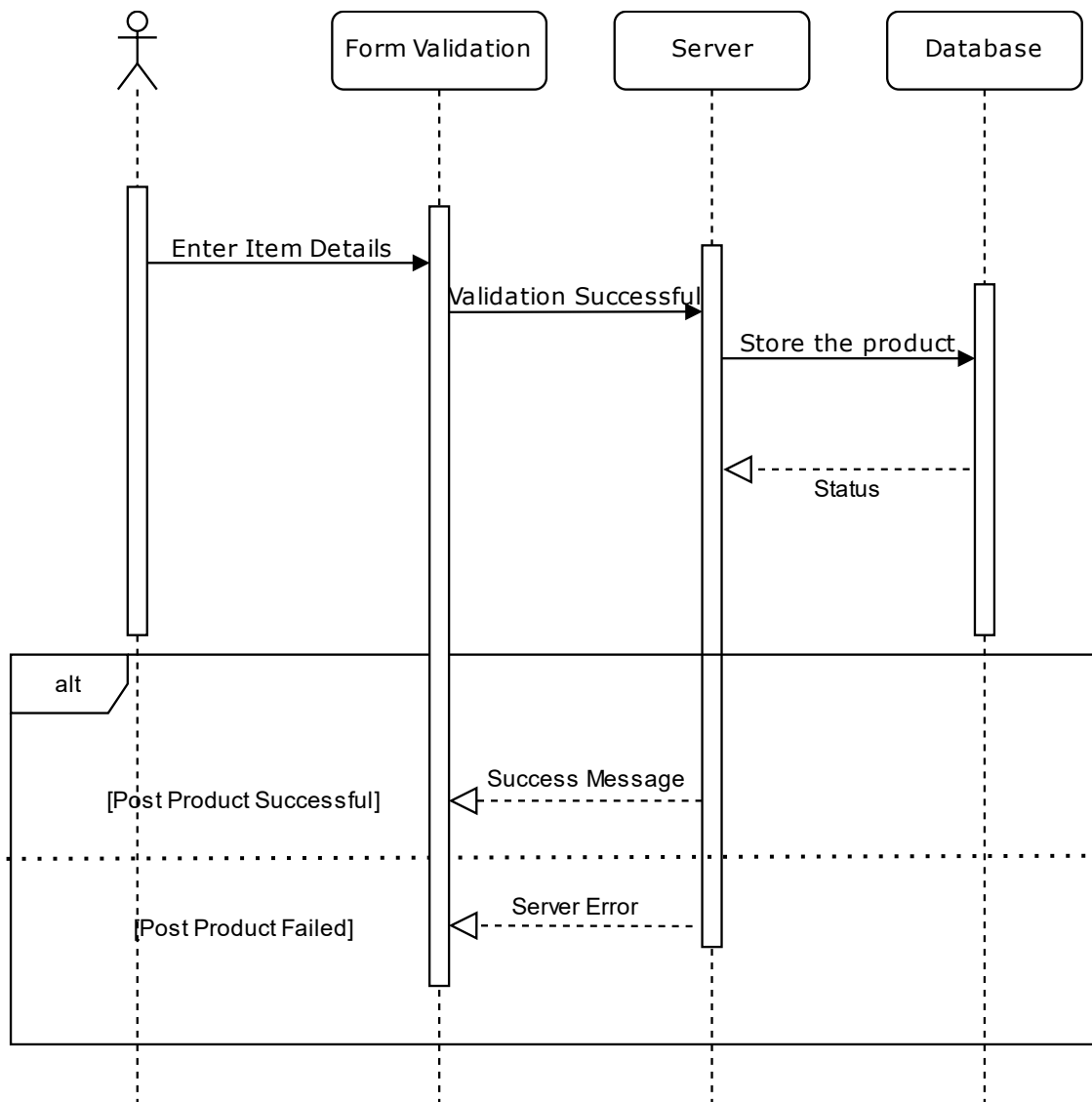


Figure 15: Sequence diagram for posting lost/found items

3.3.4 Update Profile

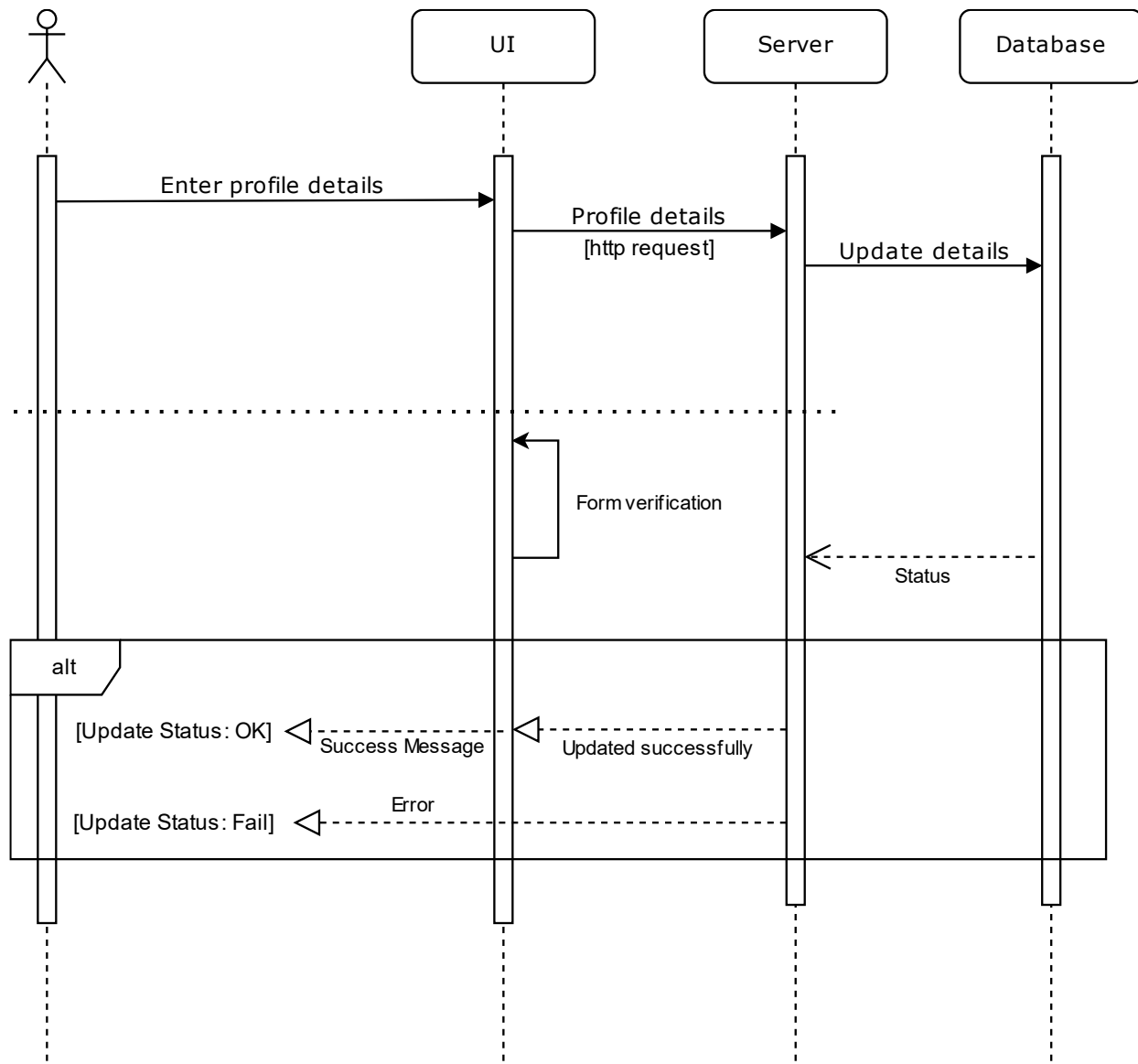


Figure 16: Sequence diagram for profile update

Purpose:

The 'Update Profile' feature allows IITK users to modify their personal details, ensuring accurate and up-to-date information. It helps in better identification, communication, and query management within Lostify.

- **Profile Accuracy:** Enables users to keep their information up to date.
- **Better Identification:** Helps in verifying lost/found item claims.
- **Enhanced Communication:** Ensures correct contact details for queries.

3.3.5 Open Chat

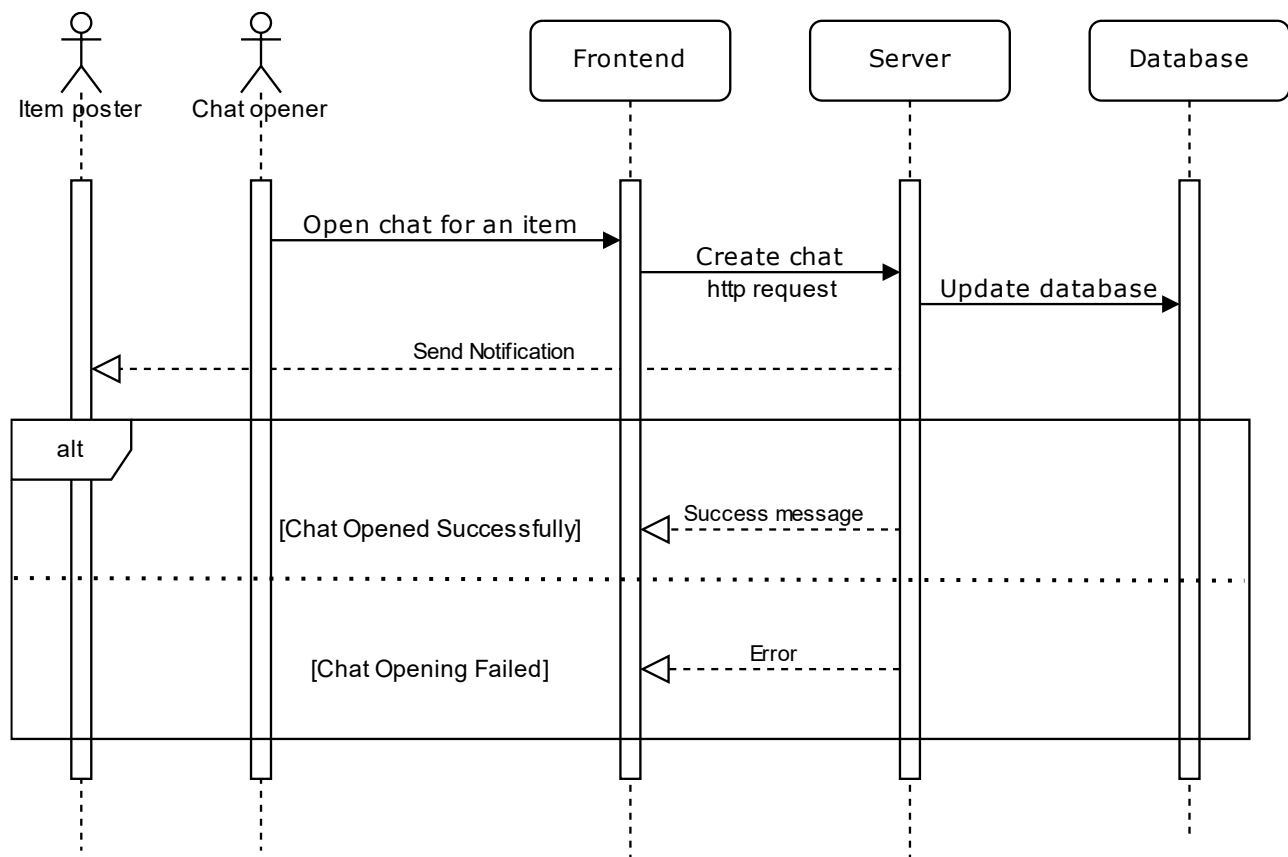


Figure 17: Sequence diagram for opening chat

Purpose:

The chat feature allows users to communicate directly for faster verification and coordination of lost and found items. Additionally, an admin monitors chats to ensure proper communication, resolve disputes, and update information if needed, enhancing the reliability of the platform.

3.3.6 Confirm Return

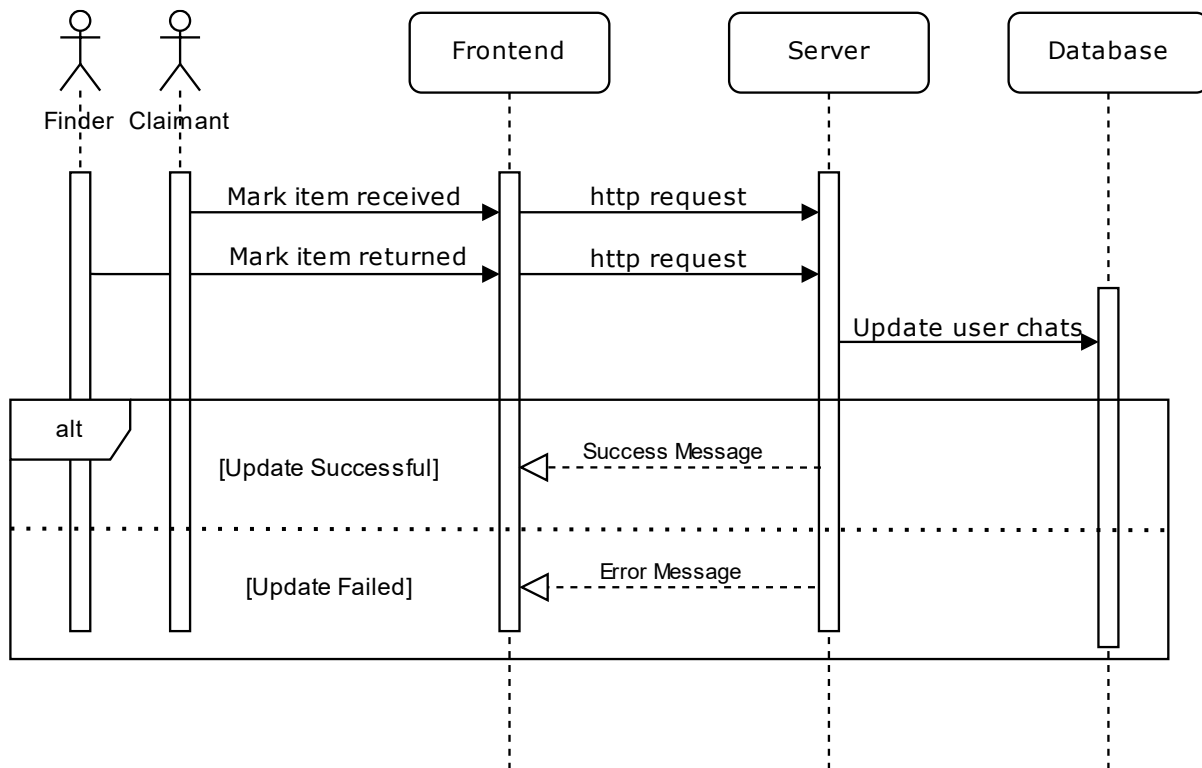


Figure 18: Sequence diagram to confirm return of lost/found item

Purpose:

Suppose a user who has found an article and has posted it under 'found items' successfully establishes the identity of the owner. The two parties shall then confirm on their respective instances of the application the successful return and receipt of the item. This ensures non-repudiation of item handover records stored for future reference in case of dispute.

3.3.7 Reset Password

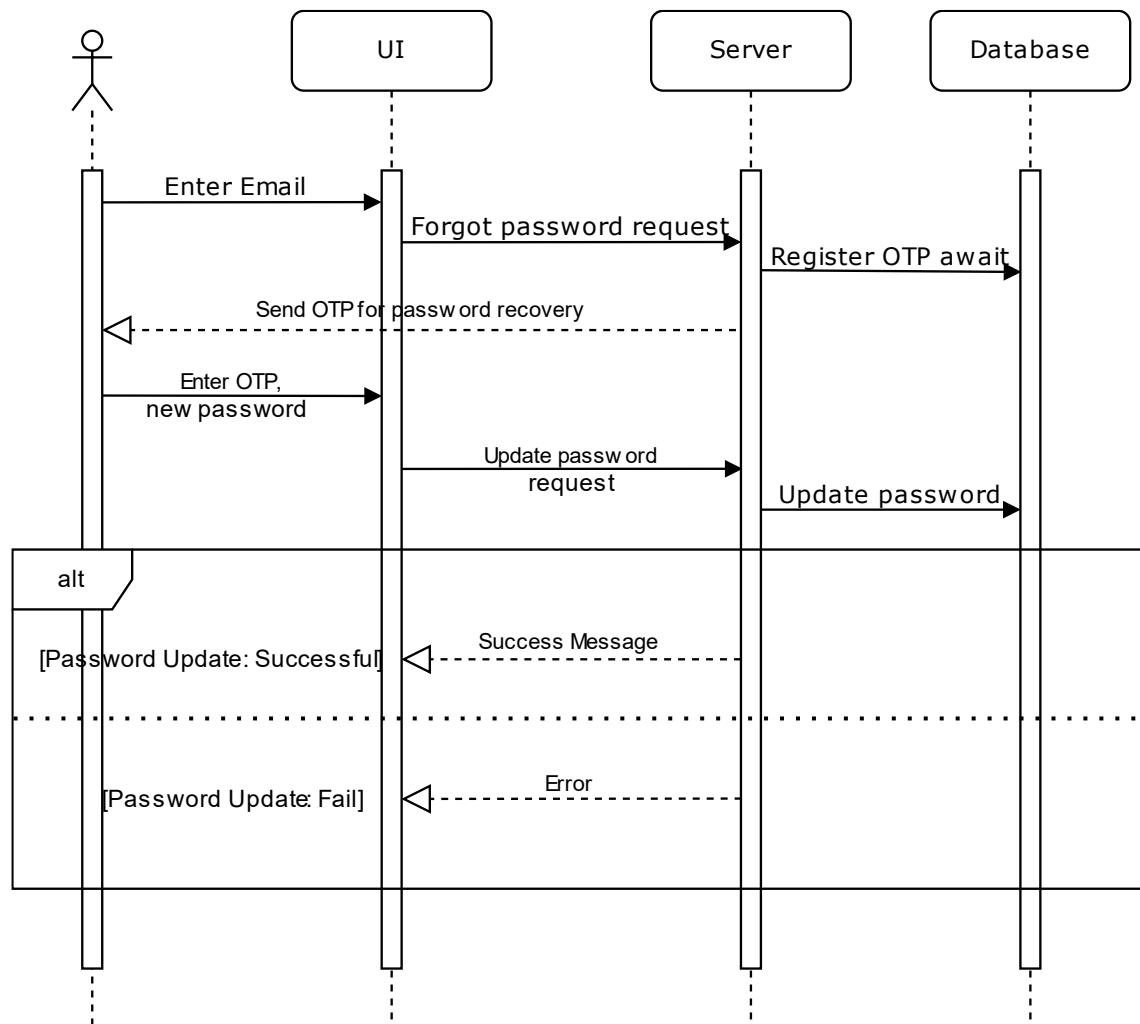


Figure 19: Sequence diagram to reset password

Purpose:

This use case ensures that users can change their password to securely reestablish access to their account.

3.3.8 Edit/Delete Own Post

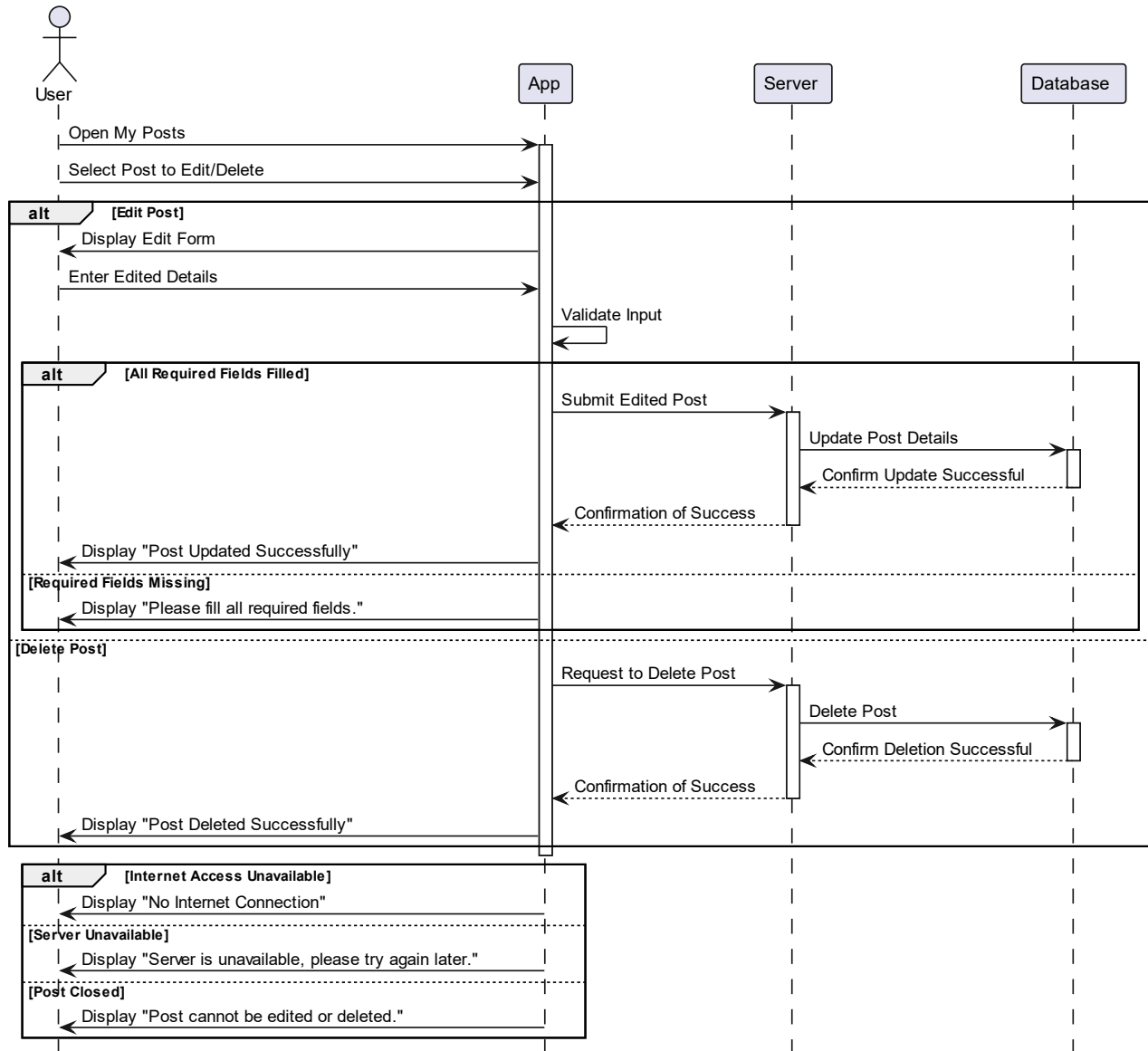


Figure 20: Sequence diagram for editing/deleting post

Purpose:

If a user has made some error or wants to change some details of his/her query already posted then s/he can edit the post via the edit/delete feature provided in Profile → My Lost items/Found items → Select the post → Edit/Delete.

3.3.9 Search Post

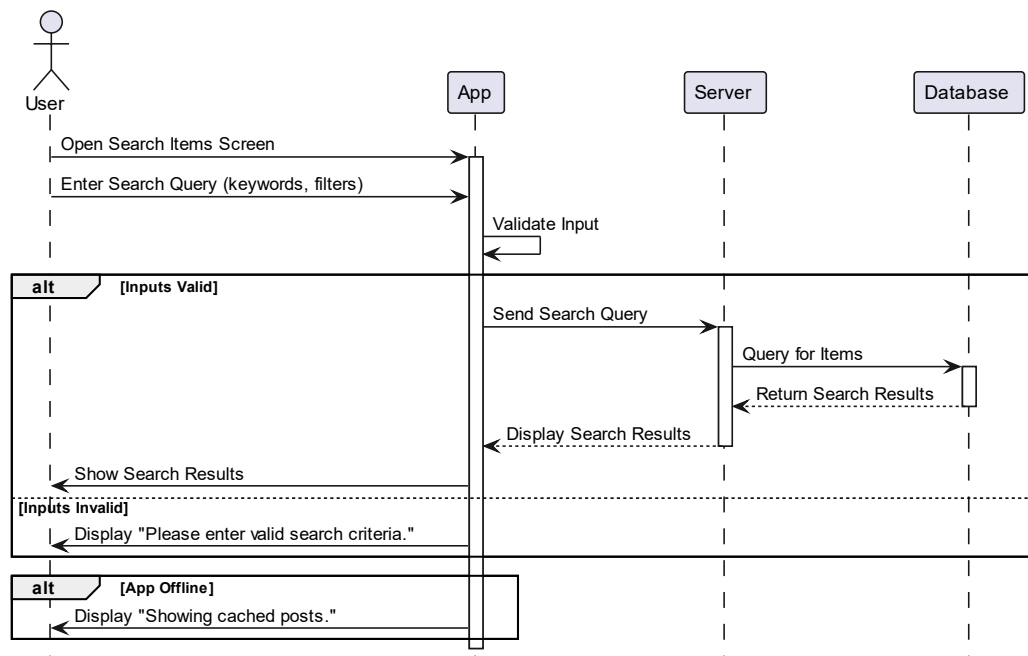


Figure 21: Sequence diagram for running a search query

Purpose:

If a specific query a user is trying to find a post, then s/he can use the search functionality and input the details about the item.

3.3.10 Access Help

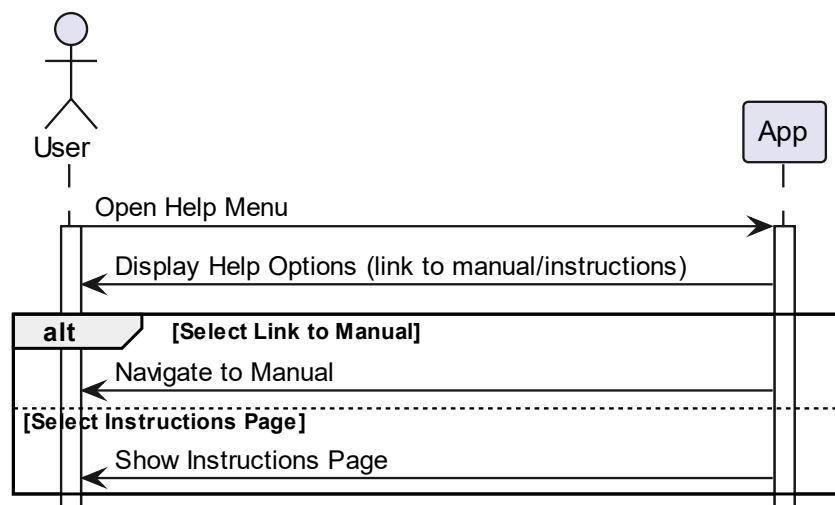


Figure 22: Sequence diagram to access help

Purpose:

Users can access a user manual.

3.3.11 Request Support

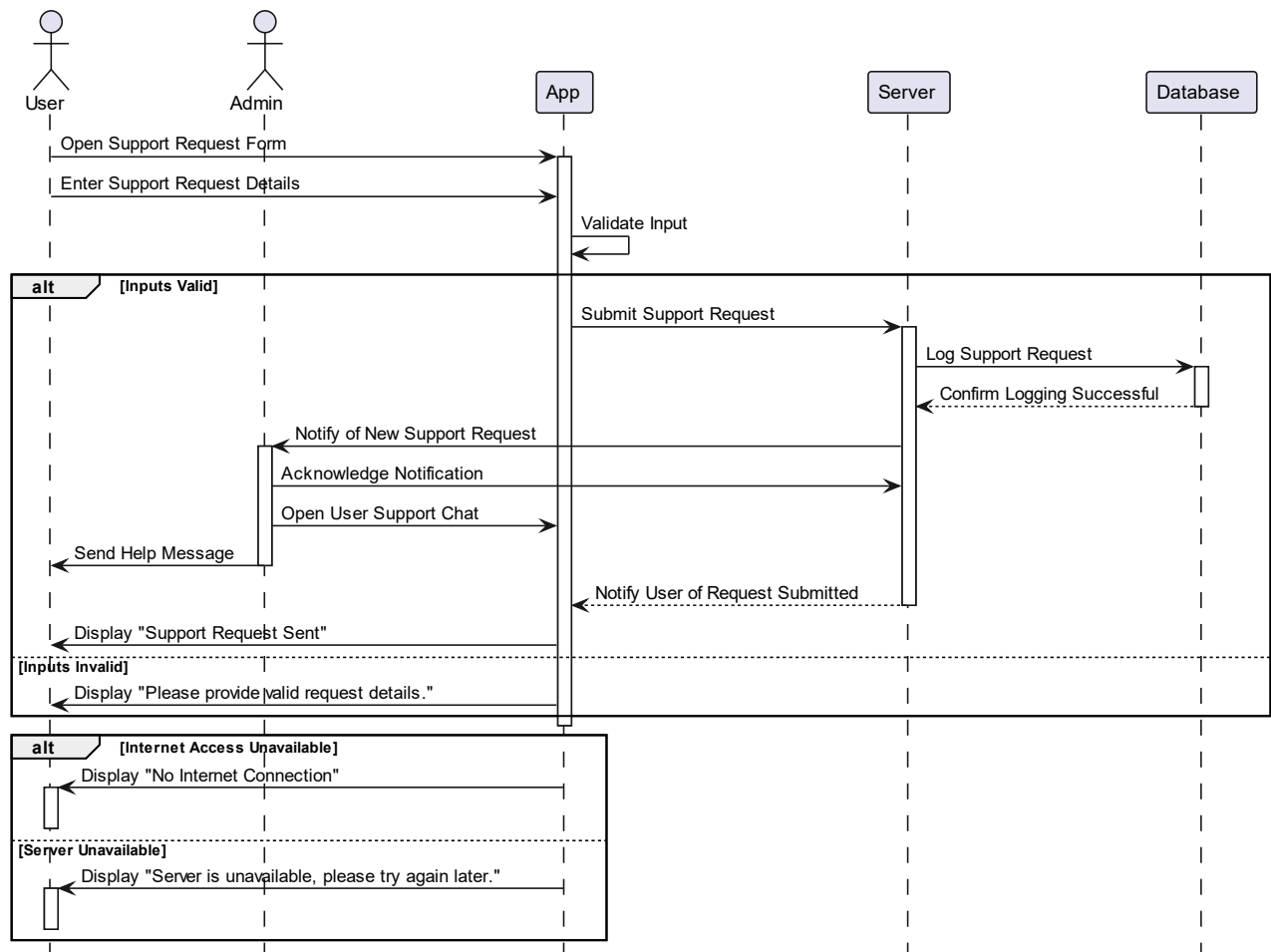


Figure 23: Sequence diagram to request support

Purpose:

Users can request admin support in case of malfunction or account-related issues through tickets.

3.3.12 Customise Theme

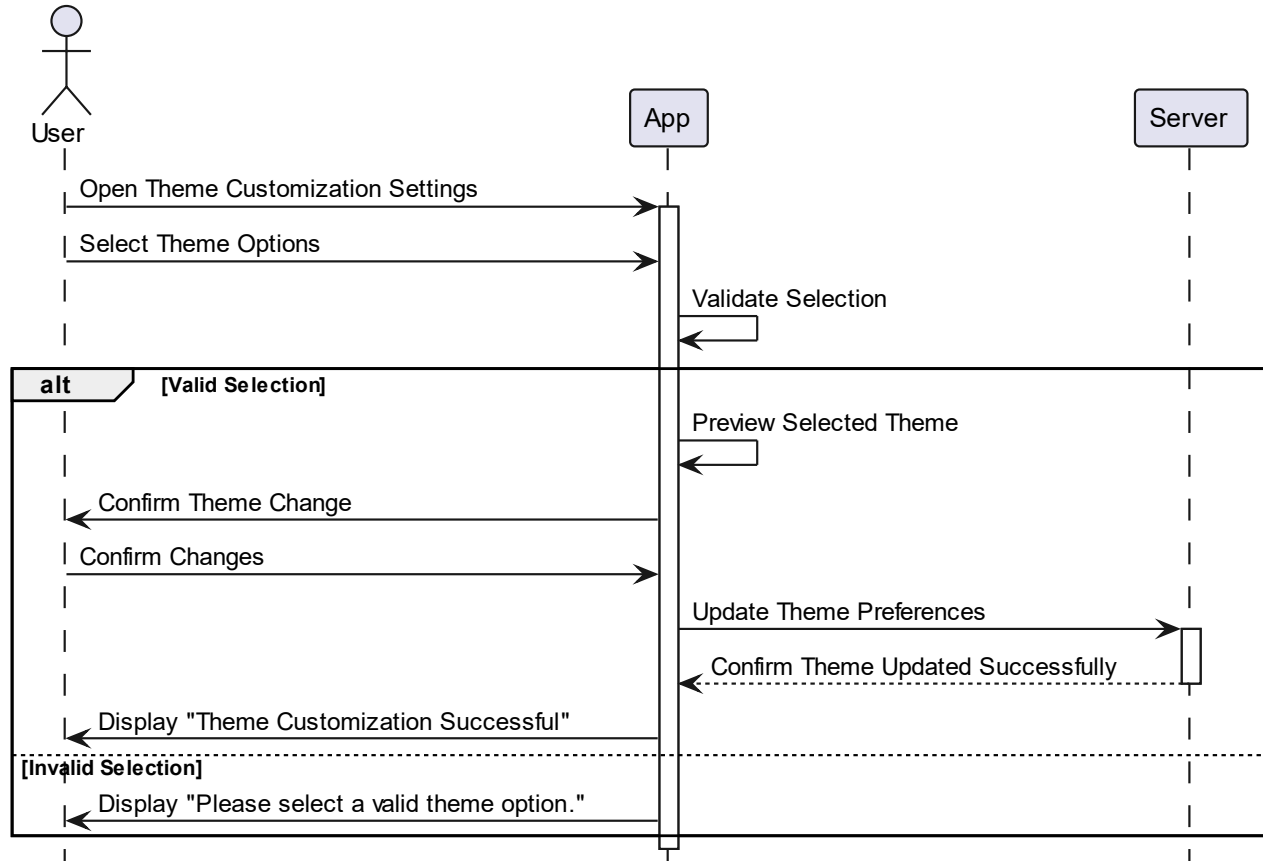


Figure 24: Sequence diagram to customise theme

Purpose:

User can toggle between dark theme and light theme.

3.3.13 Log out

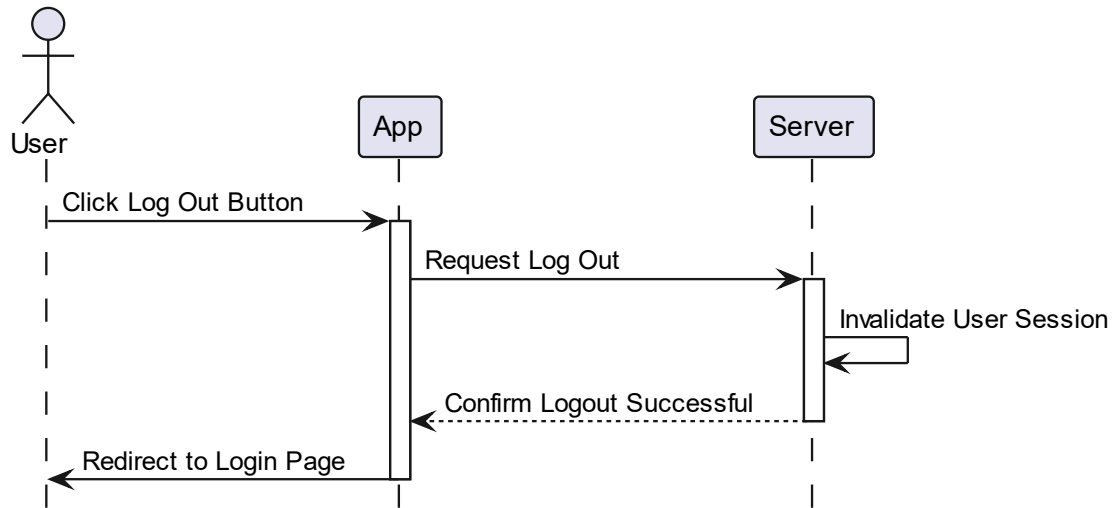


Figure 25: Sequence diagram to log out

Purpose:

The user can log out of the app on the current device.

3.4 State Diagram

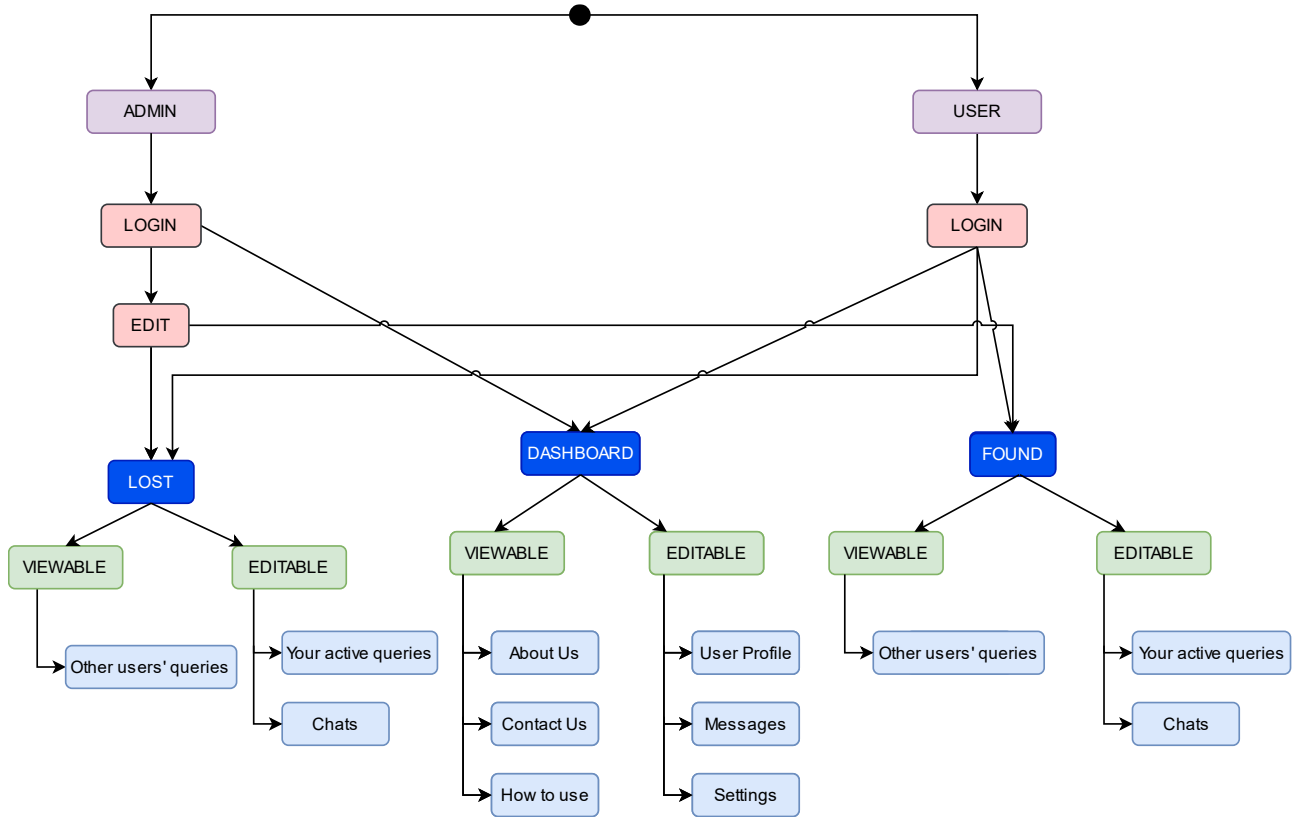


Figure 26: State diagram

The state diagram illustrates the various states and transitions within Lostify. It highlights the interaction flow between different user roles, namely **Admin** and **User**, as they navigate through the system.

3.4.1 Overview

The system begins in an initial state, from which users can either log in as an admin or a regular user. Upon logging in, the admin can access the **Edit** state, allowing modifications to various sections of the application. Users have access to two primary states: **Lost** and **Found**, where they can report and manage lost or found items. A central dashboard provides essential information and navigation options for both users and admins.

3.4.2 State Descriptions

3.4.2.1 Admin State

Admins must log in before accessing the **Edit** state. They can manage lost and found reports and edit system-wide settings.

3.4.2.2 User State

Users log in to enter the system and can navigate to the Lost or Found sections. They can create, view, and manage lost or found item queries.

3.4.2.3 Lost and Found States

Both sections allow users to:

- View other users' queries (read-only access).
- Manage their active queries, including editing details and engaging in chats.

3.4.2.4 Dashboard

The dashboard provides access to various viewable sections like 'About Us', 'Contact Us', and 'How to Use'. Editable sections include User Profile, Messages, and Settings for personalisation.

4 Project Plan

4.1 Timeline

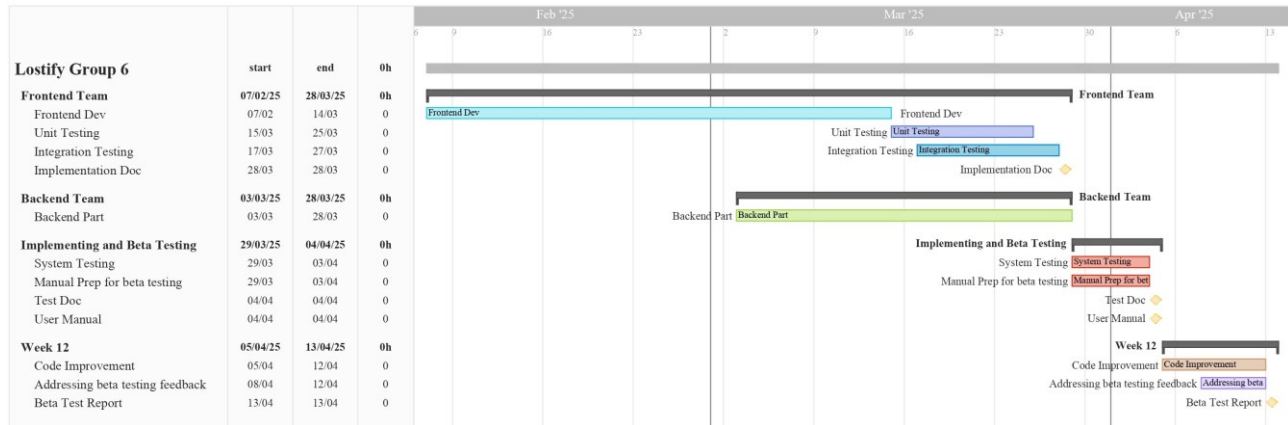


Figure 27: Gantt chart for development and testing

- 15 January 2025** Documentation of Software Requirements Specification (SRS).
- 24 January 2025**
- 25 January 2025** Documentation of Software Design Document (SDS).
- 7 February 2025**
- 8 February 2025** Implementation and testing: Member-wise classification of tasks as well as the required software is given below. Major tasks including writing the source code, designing the web page, making and testing unit as well as integrated test cases and implementing a secure Firewall would be done during this time. Testing would be done by members of respective functions.
- 28 March 2025**
- 29 March 2025** System and manual beta testing: Further necessary testing would be done during this week. This would include testing the system as a developer as well as (beta) testing as an end-user.
- 4 April 2025**
- 5 April 2025** Code improvement and beta testing: With the results of the previous testing sessions, improvements on the software would be done to increase speed, security and its overall quality.
- 13 April 2025**
- 14 April 2025** Addressing Beta Testing Feedback and Delivering the Final Project Report
- 23 April 2025**

4.2 Development Phases

This section gives a broad classification of the tasks along with the names of team members responsible for it.

4.2.1 Frontend Design

A frontend designer not only creates the aesthetics and wireframes for the product but also turns them into reality via code. Any component of the website which the end user is able to manipulate can be classified as frontend.

The members responsible for frontend development are:

- a. Ayush Patel
- b. Somaiya Narayan Aniruddh
- c. Shaurya Johari
- d. Teja Satvik
- e. Vinay

The following software might be useful for frontend development:

- **Flutter:** Flutter is an open-source UI software development kit developed by Google, enabling developers to build natively compiled applications for mobile, web, and desktop from a single codebase. It features a rich set of customizable widgets for creating expressive UIs, along with a 'hot reload' feature for instant UI updates during development. Flutter's engine, built with C++ and Dart, compiles code to native machine code, ensuring high performance and smooth graphics rendering. This allows for cross-platform development, saving time and effort while maintaining a consistent look and feel across different operating systems.

4.2.2 Backend Development

It is everything that the users don't see and contains behind-the-scenes activities that occur when performing any action on a website. It focuses primarily on databases, backend logic, APIs, and servers.

The members responsible for backend development are:

- a. Ayush Patel
- b. Shaurya Johari
- c. Vinay
- d. Anirudh Cheriyanaseri Bijay
- e. Aayush Kumar
- f. Krishna Kumayu
- g. Aman Raj

The following software might be useful for backend development:

- **Flask:** Flask is a lightweight Python web framework that offers simplicity and flexibility for building web applications. Considered a microframework, Flask doesn't force specific tools or libraries, allowing developers to choose their preferred components. It includes a built-in development server, debugger, Jinja2 templating, RESTful request dispatching, and supports secure cookies. Flask is designed to be easily extended with various extensions, providing developers control over their project's structure and the freedom to build web applications of varying complexity.

Appendix A – Group Log

Date & Time	Venue	Minutes
23/01/25 3:00 pm	4 th floor, Rajeev Motwani Building	Commenced interface design with rough outlines developed. Assigned the design of wireframes in Figma to Ayush Patel, Marada Teja Satvik, Somaiya Narayan Aniruddh, and Shaurya Johari.
31/01/25 3:00 pm	4 th floor, Rajeev Motwani Building	Reviewed the designs developed and gathered feedback. Discussed code organisation and implementation. Established collaboration and repository management policies.
02/02/25 2:00 pm	1 st floor, Rajeev Motwani Building	Planned and commenced work on design of state, class, and sequence diagrams. Collaborated on further refinements for UI design on Figma.
04/02/25 10:00 am	4 th floor, Rajeev Motwani Building	Preparation of individual diagrams assigned to members. Charted out the project plan and timeline for development and testing.
05/02/25 8:00 pm	1 st floor, Rajeev Motwani Building	Reviewed and made final edits to all diagrams and designs. Proceeded for inclusion in final document.