A

Project Report

On

SHARE-HUB

Developed by

GOHIL KAPILBHAI S. (CE015)

Department of CE, DD University

Guided By:

Prof. Siddharth P. Shah

Department of Computer Engineering

Faculty of Technology,

Dharmsinh Desai University

College Road, Nadiad-387001

Department of Computer Engineering
Faculty of Technology,
Dharmsinh Desai University, Nadiad

CERTIFICATE

This is to certify that the practical/term work carried out in the subject of **System Design Practice** and recorded in this journal is the bonafide work of **Gohil Kapilbhai CE015 22CEUOD002** of B.Tech semester **VI** in the branch of Computer Engineering during the academic year **2023-2024.**

Prof. Siddharth P. Shah
Assistant Professor,
Dept. of Computer Engineering.,
Faculty of Technology,
Dharmsinh Desai University,
Nadiad

Dr. C.K. Bhensdadia
Head,
Dept. of Computer Engineering,
Faculty of Technology,
Dharmsinh Desai University,
Nadiad

Abstract

The purpose of this project is to streamline the process of document sharing among users, addressing the need for efficient and secure sharing while maintaining control over access rights. In various scenarios, users often find themselves in situations where they must share materials or documents with others but wish to restrict access to certain individuals or groups. Recognizing these challenges, the Share-Hub developed, an innovative online document sharing platform.

Share-Hub offers users the ability to create, upload, and share materials with friends, colleagues, or collaborators seamlessly. What sets Share-Hub apart is its robust role-based access control system, which allows creators to assign specific roles to each user. These roles dictate the level of access granted to users, ensuring that sensitive documents remain protected.

Furthermore, Share-Hub empowers document owners with granular control over permissions within these roles. For instance, users assigned the "viewer" role may be permitted to download documents for viewing purposes only, while those with the "editor" role may have the additional ability to make modifications. This flexibility enables creators to tailor access rights according to their specific requirements, promoting collaboration while safeguarding document integrity.

The primary objective of Share-Hub is to provide users with a user-friendly and intuitive platform for sharing various documents and files without encountering the complexities often associated with traditional file-sharing methods. By simplifying the sharing process and enhancing security measures, Share-Hub aims to facilitate seamless collaboration and information exchange among users.

Acknowledgements

I would like to express my sincere gratitude to all those who have contributed to the completion of this project.

First and foremost, I extend my deepest appreciation to **Prof. Siddharth P. Shah** whose guidance, support, and invaluable insights have been instrumental throughout this endeavor. Your expertise and encouragement have not only enriched the project but also contributed significantly to my personal and professional growth.

Last but not least, I would like to express my gratitude to my family for their unwavering love, support, and understanding throughout this journey. Your encouragement and belief in my abilities have been a constant source of strength, and I am truly grateful for your presence in my life.

In conclusion, I extend my heartfelt thanks to all those who have been a part of this project, directly or indirectly. Your contributions have been invaluable, and I am deeply grateful for your support.

With Sincere Regards,

Gohil Kapil

List of Figures

List of Tables

Table of Contents

1 Introduction

1.1 Brief Overview

Share-Hub is a platform where people can easily share documents. Sometimes, people need to share documents to the other peoples. Share-Hub makes it simple to share these documents while keeping them safe.

Share-Hub is designed to make sharing documents easy and safe. It's like a new and improved way of sharing files online. It's made to be easy for anyone to use, even if they're not very familiar with technology.

Overall, Share-Hub is all about making it easy for people to share and manage their documents online in a safe and simple way.

1.2 Key Features

1. **User Authentication (Login/Register):** Share-Hub provides a secure user authentication system, allowing users to create accounts (register) or log in securely. This ensures that only authorized individuals have access to the platform's features.
2. **File Upload and Download:** Users can easily upload documents and files to Share-Hub, making them accessible for collaboration. Additionally, users can download files shared by others to their local devices for offline access.
3. **Document Sharing:** Share-Hub facilitates seamless document sharing among users. Whether it's sharing files with specific individuals or groups, users can easily collaborate on projects by sharing documents with others on the platform.
4. **Access Control:** Share-Hub offers robust access control mechanisms, allowing document owners to control who can view, edit, or download their files. Users can assign different access levels and permissions to individuals or groups, ensuring that sensitive information remains protected.
5. **Notifications:** Share-Hub keeps users informed about relevant activities through notifications. Users receive notifications about file sharing requests, updates to shared documents, and other important events, helping them stay up-to-date with project developments.

6. **User-Friendly Interface:** Share-Hub features an intuitive and user-friendly interface, making it easy for users to navigate the platform and access its various features. The interface is designed to be accessible to users of all skill levels, ensuring a seamless user experience.
7. **Comment on material:** Share-Hub provides the functionality to user to comment on the material and provide the feedback.

1.3 Tools, Technology, and Platforms Used

In the development of Share-Hub, a variety of tools, technologies, and platforms were utilized to ensure the successful implementation of its features and functionalities. This section provides an overview of the key tools, technologies, and platforms employed in the development process.

1.3.1 MERN Stack

Share-Hub was built using the **MERN** stack, which is a popular technology stack for developing full-stack web applications. **MERN** stands for **MongoDB**, **Express.js**, **React.js**, and **Node.js**, each serving a specific purpose in the development process:

1. **MongoDB:** MongoDB is a NoSQL database that was chosen for its flexibility, scalability, and ease of integration with Node.js. It provides a reliable storage solution for managing document metadata and user information within Share-Hub.
2. **Express.js:** Express.js is a minimalist web application framework for Node.js that simplifies the process of building web applications and APIs. It was used in Share-Hub to handle routing, middleware integration, and server-side logic.
3. **React.js:** React.js is a JavaScript library for building user interfaces, known for its component-based architecture and efficient rendering. Share-Hub's client-side interface was developed using React.js, providing users with a responsive and interactive experience.
4. **Node.js:** Node.js is a runtime environment that allows developers to run JavaScript code on the server-side. It was used in Share-Hub to build the backend infrastructure, handle server-side logic, and interact with the MongoDB database.

1.3.2 WebStorm IDE

WebStorm IDE, developed by JetBrains, was chosen as the integrated development environment (IDE) for building Share-Hub. WebStorm provides a powerful set of features for web development, including intelligent code completion, debugging tools, version control integration, and support for popular web technologies and frameworks. Its intuitive interface and robust feature set made it an ideal choice for streamlining the development workflow and ensuring code quality and consistency throughout the project.

1.3.3 Firebase

Firebase was utilized for document storing purposes within Share-Hub. Firebase offers a scalable and flexible cloud-based platform for storing and syncing data across devices, making it an ideal choice for document management within the application.

2 Software Requirements Specifications

2.1 Introduction to Software Requirements

software requirements are like a recipe that tells us what features the Share-Hub platform needs and how they should work together to create a useful and user-friendly system.

In this section, we'll explore the different things Share-Hub should be able to do and what qualities it should have to make sure it's easy to use, safe, and helpful for everyone who uses it. Think of it as a list of all the important things Share-Hub needs to be able to do to be successful.

2.2 Product Scope

Share-Hub is a user-friendly online platform where you can easily share documents with others. You can log in or create an account, upload your files, and then share them with specific people. You can also download files, delete them when you no longer need them, and control who can access them. Plus, Share-Hub keeps you updated with notifications about any new files shared with you. It's designed to make document sharing simple and secure for everyone.

2.3 User Classes and Characteristics:

2.3.1 User Class

In Share-Hub, there is a single user class representing all users of the platform. Each user is identified by their unique login credentials and has access to the same set of features and functionalities. However, the roles assigned to a user can vary depending on the specific material being shared.

2.3.2 Roles and Abilities

When a user shares a document or file on Share-Hub, they have the ability to assign roles to other users who are granted access to that material. These roles, along with their respective abilities, are as follows:

1. **Owner:**
    - **Ability:** Share, upload, delete material, delete content, download
    - **Description:** The Owner is the creator of the material and holds sole ownership over it. The Owner has full control over the material, including the ability to share, upload, and delete both the material itself and its content. This ownership is non-editable and non-transferable, ensuring that the Owner retains complete control over the material at all times.
2. **Collaborator:**
    - **Ability:** Share, upload, delete content, download
    - **Description:** Collaborators have full access to the shared material, allowing them to view, edit, and contribute to the document. They can upload new versions, delete outdated content within the material, and download the document for offline use.
3. **Viewer:**
    - **Ability:** Download
    - **Description:** Viewers have permission to download and view the shared material but cannot make any edits or modifications. They are restricted to read-only access, ensuring that the document remains unchanged.
4. **Editor:**
    - **Ability:** Upload, delete content, download
    - **Description:** Editors have the ability to upload new versions of the shared material, make edits, and delete outdated content within the material. They can download the document for reference or offline use but may not have full access to all functionalities available to the document owner.

By default, these abilities are set by the owner of the material when sharing the document or file. The owner can customize the access permissions for each role based on their specific requirements, ensuring that the shared material remains secure and accessible only to authorized individuals.

2.4 Design and Implementation Constraints

1. **Internet Connection Requirement:** Since Share-Hub operates from the cloud, a stable internet connection is essential for users to access the system through our web interface. A reliable network connection ensures seamless interaction with Share-Hub's features and functionalities. Therefore, users should have access to a good internet connection to effectively utilize the platform.
2. **System Resources:** Share-Hub's performance may be affected by the capabilities of the user's device, such as processing power, memory, and available storage space. Users with older or less powerful devices may experience slower response times or limitations in functionality.
3. **Third-Party Dependencies:** Share-Hub relies on Firebase services for document management, which may be subject to changes or disruptions. Users should be aware of potential dependencies on Firebase services and stay informed about any updates or changes that may impact Share-Hub's availability or performance.

2.5 Functional Requirements

2.5.1 User Management

R.1 User registration

Description: Users can register themselves on the platform to access its functionalities and services.

R.1.1 Select create new account option

Input: User selects the option to create a new account.

Output: The user is prompted to enter their full name, email, and password.

R.1.2 Provide the details and submit

Input: User provides their details (full name, email, password) and submits the form.

Output: User will be asked for the opt sent to his/her provided email.

Exceptions:

1. If email format is not correct then user can’t able to submit the form.

2. If a user with the provided email already exists, they are informed about the existing account and prompted to use a different email address.
3. If the maximum number of sending OTPs is exceeded, the user is notified and prompted to wait before attempting again.

R.1.3 Enter the OTP for verification

Input: User enters the OTP received in his/her email.

Output: If the OTP is correct, the user's account is successfully created and redirected to the Successful message.

Exception: If the OTP entered by the user is incorrect, user can re-enter OTP.

R.1.4 Successful message

Input: No input.

Output: Thankful message shown upon account creation.

R.2 User Login

Description: Registered users can securely log in to their accounts to access the platform's features and services.

R.2.1 Select login option

Input: User selects the option to log in to their account.

Output: The login form is displayed, prompting users to enter their email and password.

R.2.2 Provide login credentials and submit

Input: User provides their login credentials (email and password) and submits the form.

Output: The system validates the provided credentials, and if they are correct, the user is logged in and redirected to the main page.

Exceptions: If the provided credentials are incorrect, the user is informed and can retry.

R.3 Reset password

Description: Users can reset their password if they forget it, ensuring continued access to their account.

R.3.1 Access Forgot Password option

Input: User clicks on the "Forgot Password" link on the login page.

Output: The user is directed to the password reset form.

R.3.2 Provide email address and new password

Input: Provide the email address and new password.

Output: An email containing an OTP is sent to the provided email address and user is redirected to the OTP verification page.

Exception:

1. If the provided email address is not found in the system, the user is informed.
2. If new password and confirm password do not match then user is informed.

R.3.3 Enter the OTP for verification

Input: User enters the OTP received in his/her email.

Output: If the OTP is correct, the user's password is successfully updated and redirected to the Successful page.

Exception: If the OTP entered by the user is incorrect, user can re-enter OTP.

R.3.4 Successful message

Input: No input.

Output: Message showing information regarding the password update.

R.4 Logout

Description: Users can securely log out of their accounts to end their session and protect their account security.

R.4.1 Access Logout option:

Input: User clicks on the "Logout" option in the platform's user profile dropdown.

Output: The user's session is terminated, and they are redirected to the login page.

2.5.2 Document Management

R.1 Material creation

Description: Users can create new materials by providing a name and description for the material.

R.1.1 Access Material Creation option

Input: User clicks on the "Material Create" button or option on the platform.

Output: The material creation form is displayed.

R.1.2 Enter name and description

Input: User enters the name and description for the material in the designated fields.

Output: The entered name and description are displayed in the material creation form.

Exception: If the length of the material name or description exceeds the specified character limit, the user is notified, and the creation process is halted.

R.1.3 Click on Create

Input: User clicks on the "Create" button to submit the material creation form.

Output: The material is created with the provided name and description.

R.2 File upload

Description: Users can upload files to specific materials or folders on the platform, selecting the necessary files and monitoring the upload progress.

R.2.1 Select Material and Folder to Upload Files

Input: User navigates to the desired material and folder where they want to upload files.

Output: The selected material or folder is displayed as the upload destination.

R.2.2 Select Files for Upload

Input: User clicks on the "Upload Files" button to choose files for upload.

Output: A file selection dialog is displayed, allowing users to browse and select one or multiple files.

R.2.3 Click on Upload

Input: User clicks on the "Upload Files" button to upload chosen files.

Output: Users wait for the selected files to be uploaded to the designated material and folder. A progress indicator would be displayed to show the upload progress.

Exception: If user doesn't have rights to upload file/s then the error message shown to user.

R.3 File/s download

Description: Users can select files and download from the platform.

R.3.1 Select Material and Folder to Download Files

Input: User navigates to the desired material and folder from where he/she wants to download files.

Output: The selected material and folder are opened and files shown in that material and folder.

R.3.2 Select Files for Download and click on download

Input: User selects the files they want to download and additionally provide the name of the zip file to be downloaded and clicks on download.

Output: The selected files downloaded to the user's machine.

Exception: If user doesn't have rights to download file/s then the error message shown to user.

R.4 File/s delete

Description: Users can select files and delete.

R.4.1 Select Material and Folder to delete Files

Input: User navigates to the desired material and folder from where he/she wants to download files.

Output: The selected material and folder are opened and files shown in that material and folder.

R.4.2 Select Files for delete and click on delete

Input: User selects the files they want to delete and clicks on download.

Output: The selected files deleted from cloud.

Exception: If user doesn't have rights to delete file/s then the error message shown to user.

R.5 Document share

Description: User can share the document if he/she has access to do so.

R.5.1 Click on share

Input: User clicks on share option for sharing material.

Output: Share dialog box opens which contains code for the material

If user is owner of material, then another box opens which contains the options to share using email with specific role.

R.5.2 Enter emails-roles and click on share

Input: Owner of material provides the email and roles to share material.

User can add multiple users with diff roles and click on share

Output: Notifications sends to the respective users for joining material

R.6 Join Material Via Code

Description: User can join the material via code of the material.

R.6.1 Click on the get material option

Input: User click on the join material option.

Output: Join material form shown to user.

R.6.2 Enter the code for material

Input: User enters the code for the material

Output: The material added to the user's material list.

Exception: If code is wrong then error message shown to user.

R.7 Join Material via Notification

Description: User can join material via notification they received.

R.7.1 Click on notifications

Input: User clicks on the notification's icon.

Output: Notifications shown to the user.

R.7.2 Click on join or cancel

Input: User click on join or cancel button

Output: The material added to the user's material list or upon cancelling the notification removed from the notifications of user.

R.8 Delete Document/Material

Description: User can delete material from the cloud and own list.

R.8.1 Click on the delete button

Input: User click on delete button for the material he/she wants to delete.

Output: Confirmation pop up shown to the user for confirmation.

R.8.2 Confirm the conformation

Input: Tap yes if you are sure else cancel

Output: if yes selected then material deleted from the user's material list and from the cloud.

R.9 Leave Document/Material

Description: User can leave material they had joined.

R.9.1 Click on the Leave button

Input: User click on leave button for the material he/she wants to leave.

Output: Confirmation pop up shown to the user for confirmation.

R.9.2 Confirm the conformation

Input: Tap yes if you are sure else cancel

Output: if yes selected then material removed from the user's material list.

R.10 Comment on Material

Description: User can a comment on material.

R.10.1 Select comment option

Input: click on the comment icon of material you want to comment.

Output: Form for the comment shown.

R.10.2 Write comment and submit

Input: Write your comment and hit submit.

Output: Comment submitted to the material's comment section

R.11 Like material

Description: User can like a material.

R.11.1 Click like button

Input: Click on the like button of the material you want to like or remove like.

Output: Upon clicking button if material is already liked then like removed else like added to the material.

R.12 View Comments

Description: User can view comments for the material.

R.12.1 Select the comment option

Input: click on the comment icon of material you want to see comments.

Output: Comments are displayed on the screen for that material.

2.6 Non-Functional Requirements

2.6.1 Performance

NFR 1: Share-Hub should work fast so you don't have to wait long to use it. It should load pages quickly and let you upload and download files without any delays.

NFR 2: Share-Hub should work well even when lots of people are using it at the same time. It should stay fast and not slow down, even when many people are using it together.

2.6.2 Security

NFR 3: Share-Hub keeps your passwords in the encrypted format in database so it will be hard for attacker to access your account even if he/she has access to database.

NFR 4: Share-Hub has special roles to make sure that only the people you trust can access your important documents and files. It makes sure that your information stays private and protected.

2.6.3 Usability

NFR 5: Share-Hub is easy to use. It has clear buttons and menus so you can find what you need quickly. It's made for everyone, even if you're not very good with computers.

2.6.4 Reliability

NFR 7: Share-Hub is always available for you to use. It hardly ever goes offline, so you can access your documents and files whenever you need them.

NFR 8: If something goes wrong while you're using Share-Hub, The developer quickly fixes the problem so you can keep using it without any trouble.

2.6.5 Scalability

NFR 9: Share-Hub grows with you. It can handle more users and documents as more people start using it. Even if lots of people join, it stays fast and reliable.

2.6.6 Compatibility

NFR 10: Share-Hub works on all devices. You can use it on your computer, tablet, or phone, no matter what type you have. It works with all kinds of web browsers, so you can use it with the one you like best.

3 Design

3.1 Use-case diagram

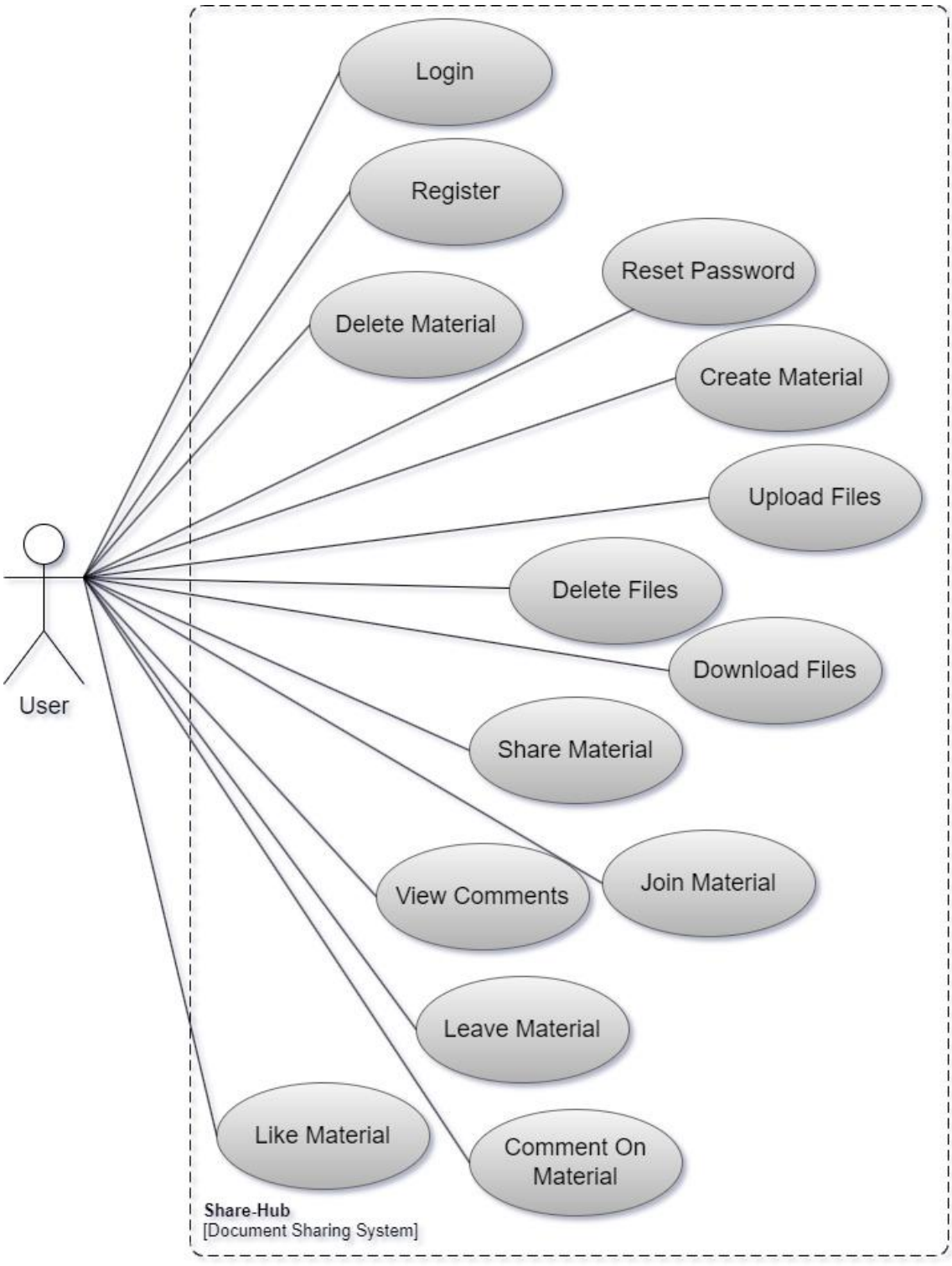

Figure 3.1 Use case Diagram

3.2 E-R Diagram

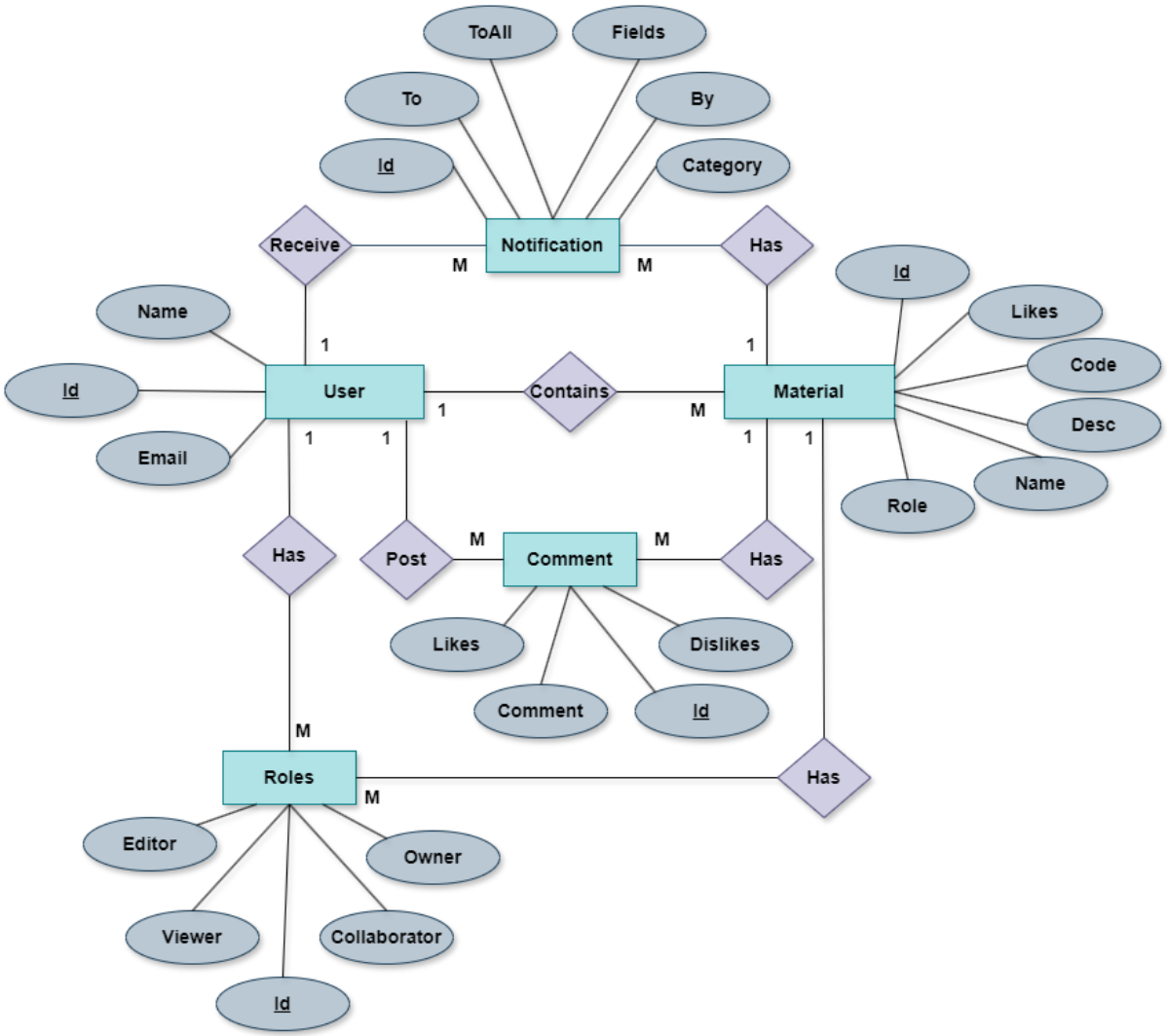

Figure 3.2 E-R Diagram

3.3 Sequence Diagrams

3.3.1 Register

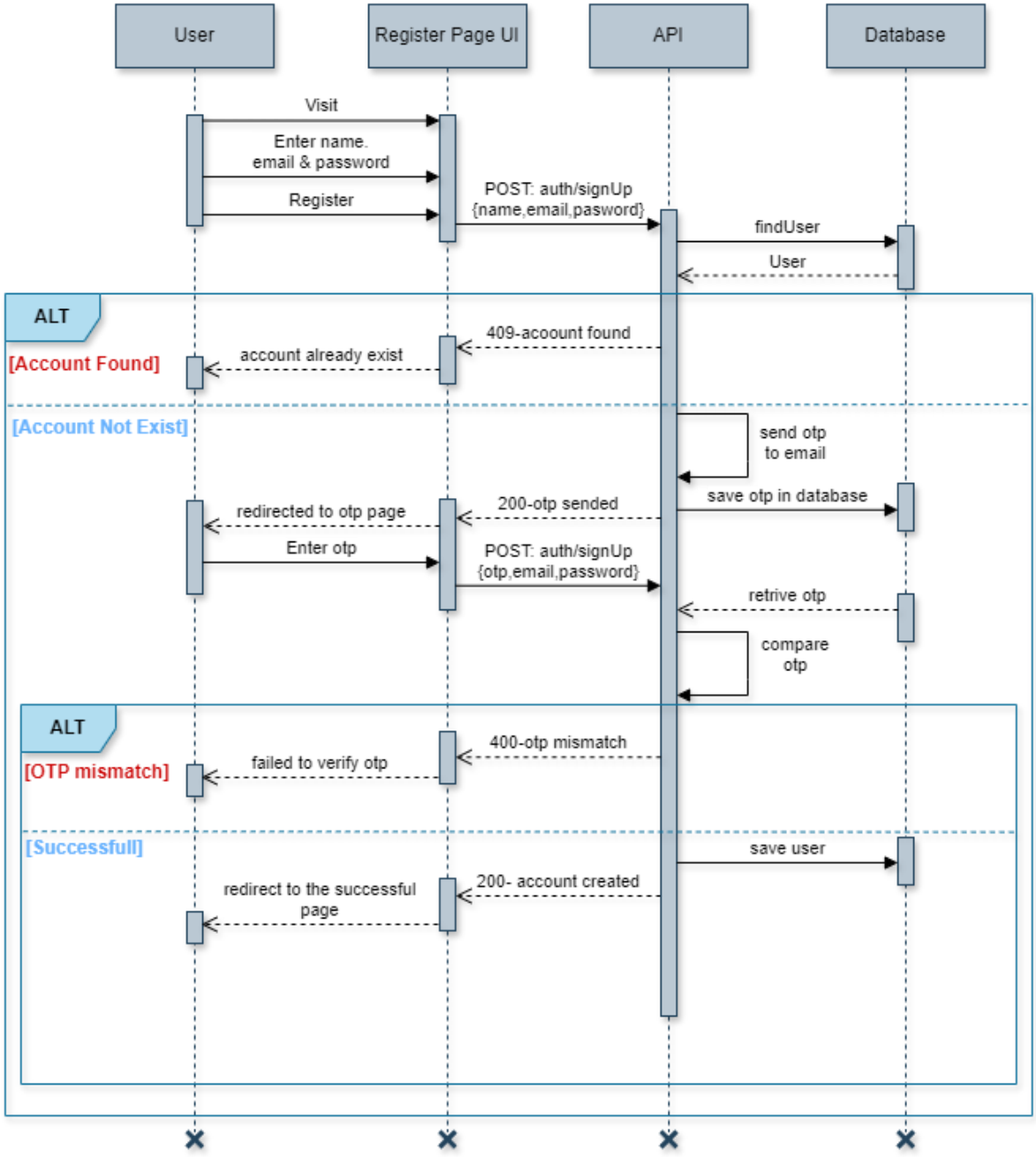

Figure 3.3 Register Sequence Diagram

3.3.2 Login

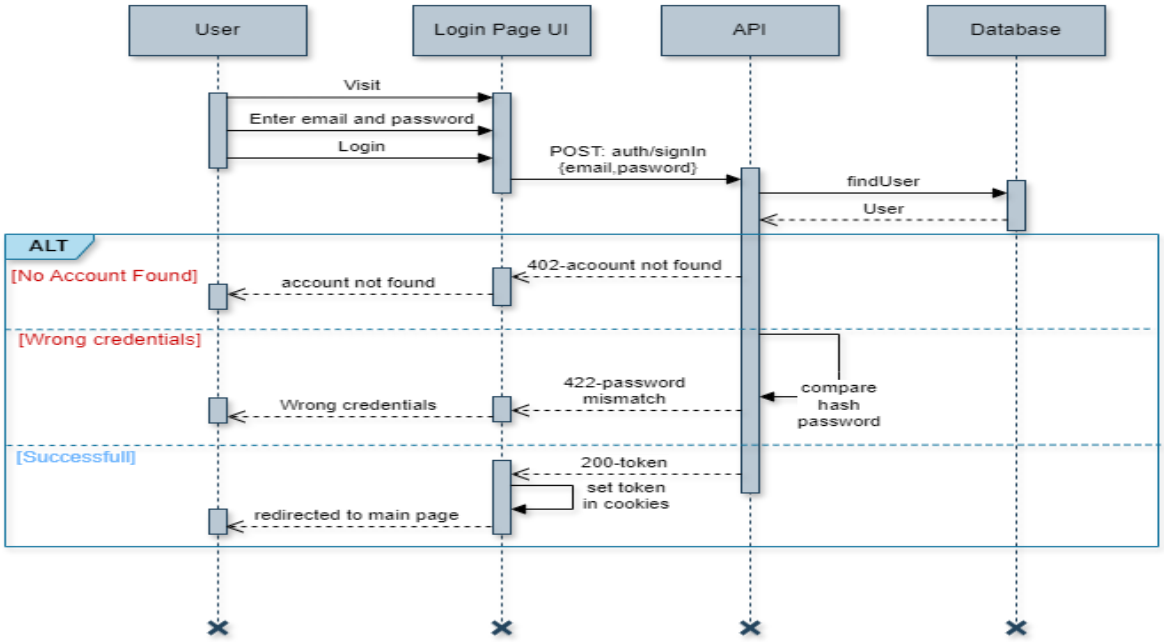

Figure 3.4 Login Sequence Diagram

3.3.3 Create Material

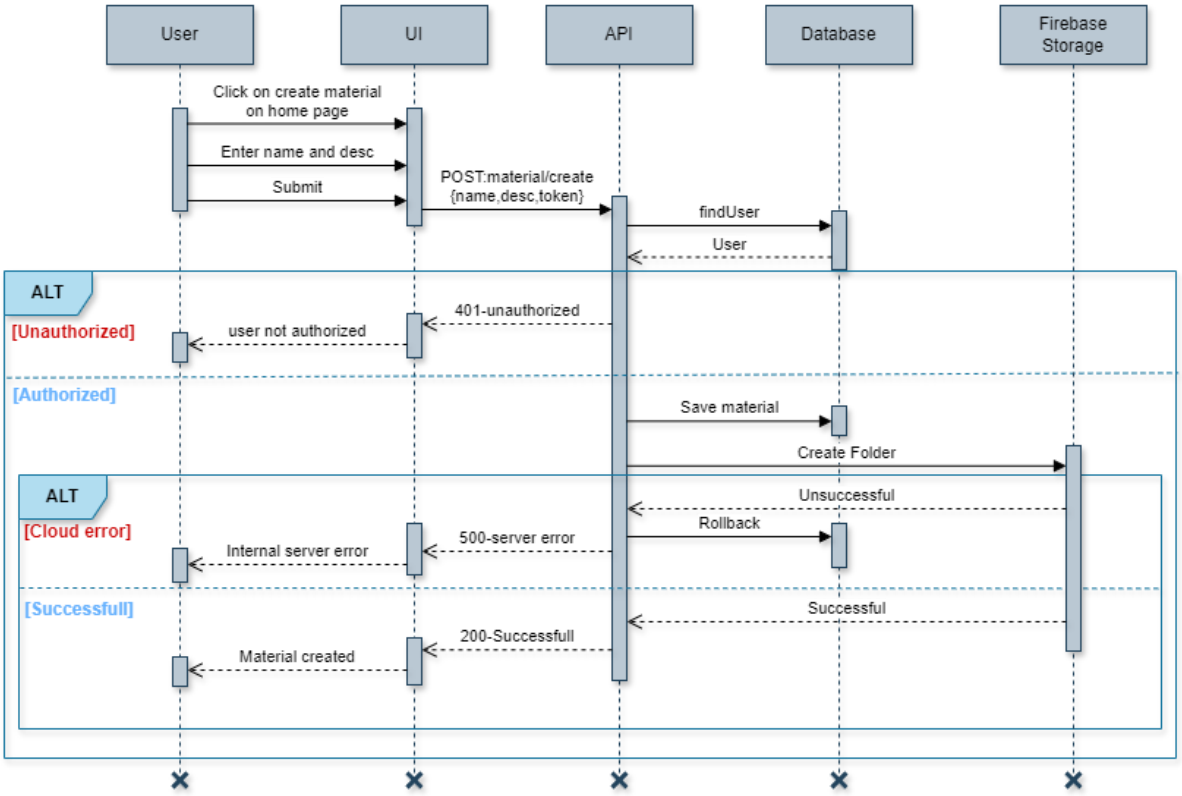

Figure 3.5 Create Material Sequence Diagram

3.3.4 Delete Material

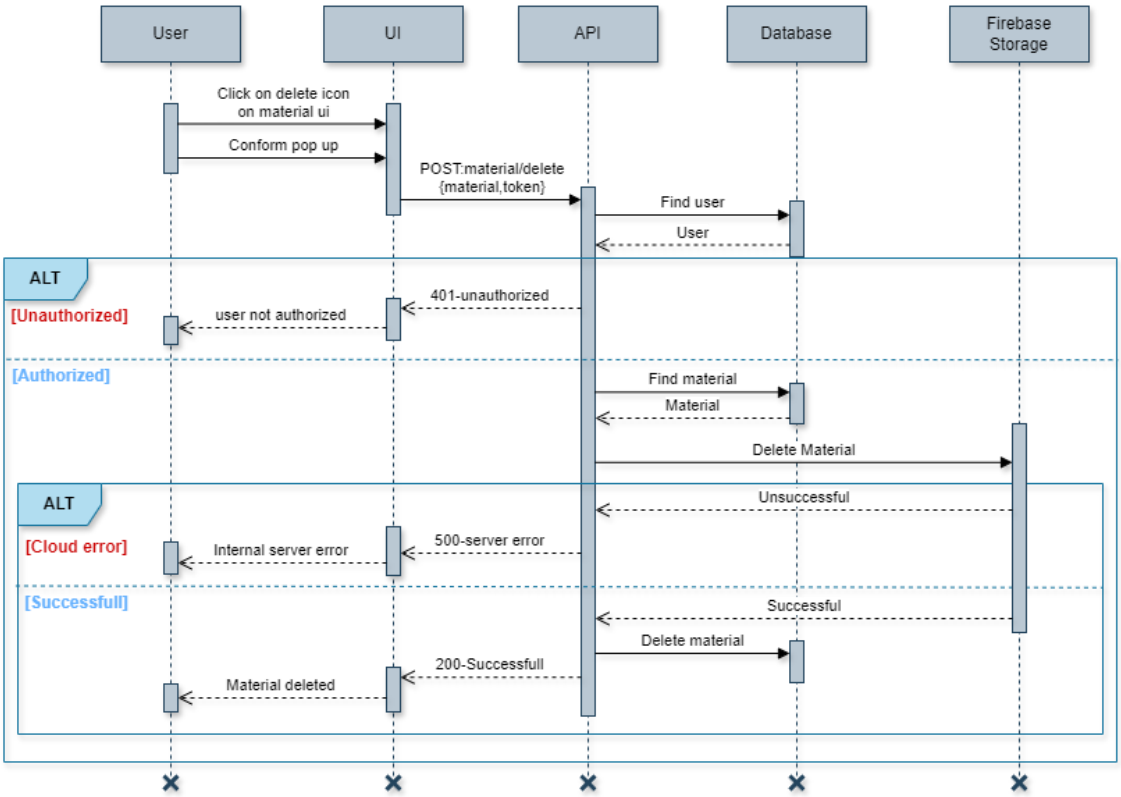

Figure 3.6 Delete Material Sequence Diagram

3.3.5 Share Material

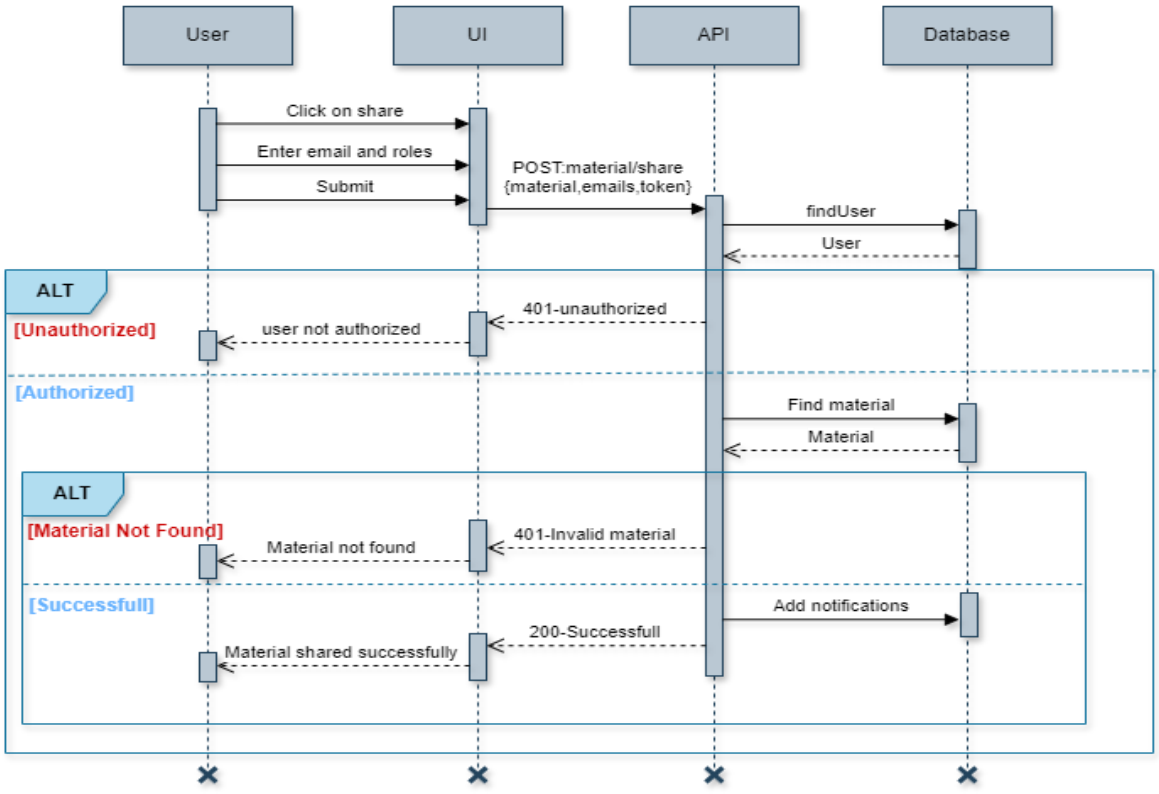

Figure 3.7 Share Material Sequence Diagram

3.3.6 File/s upload

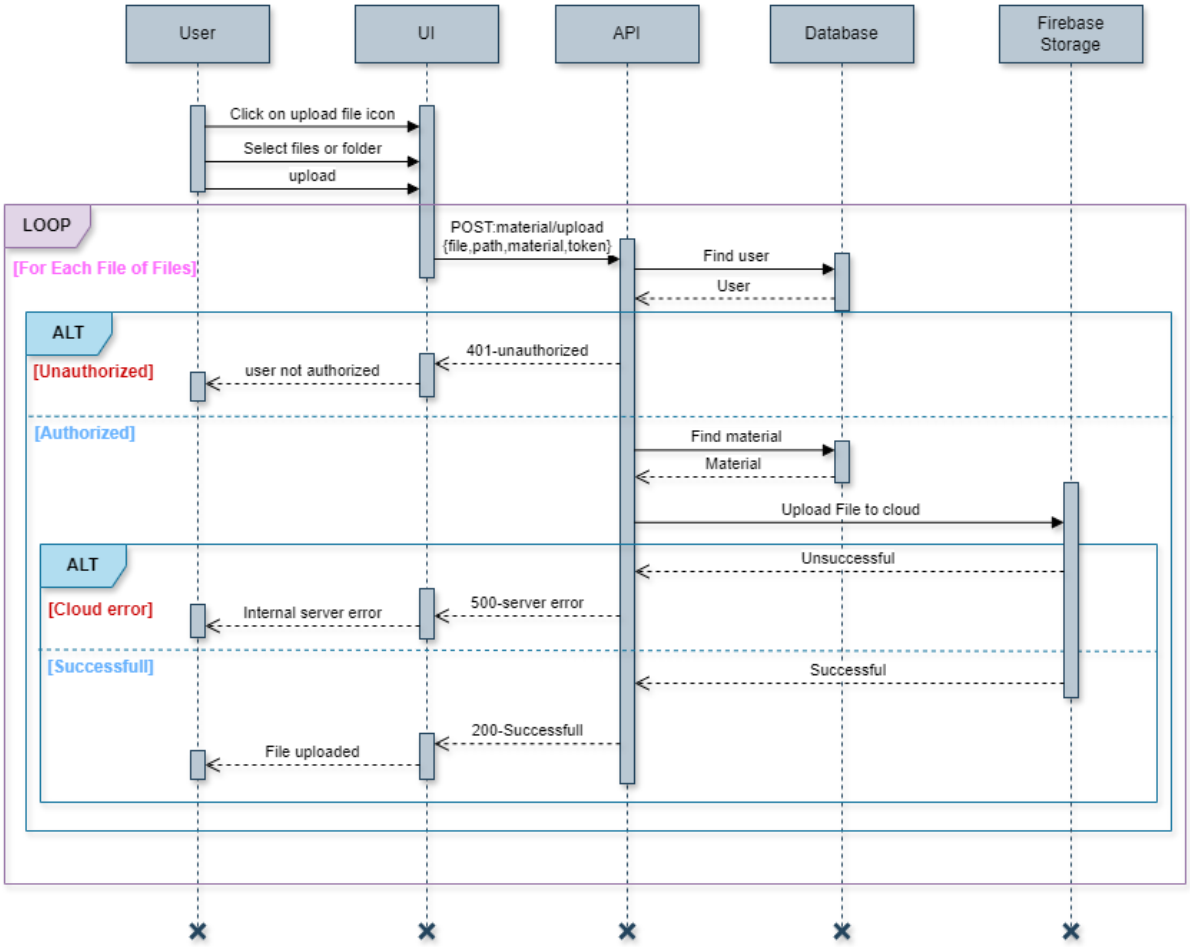

Figure 3.8 File/s Upload Sequence Diagram

3.3.7 File/s download

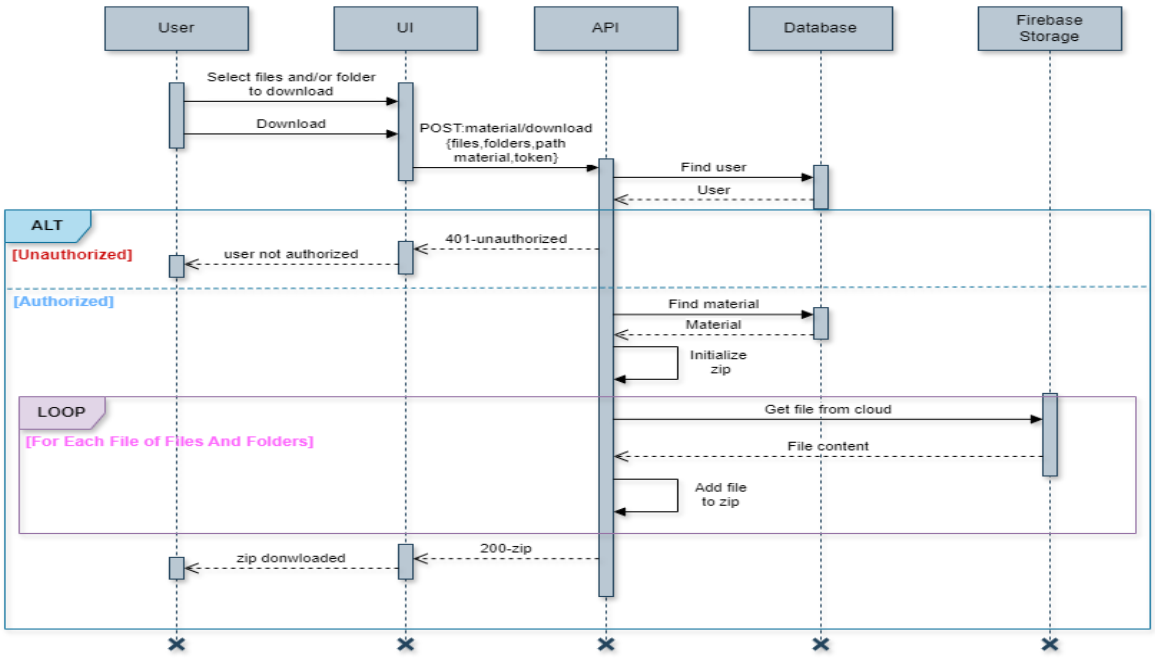

Figure 3.9 File/s Download Sequence Diagram

3.3.8 File/s Delete

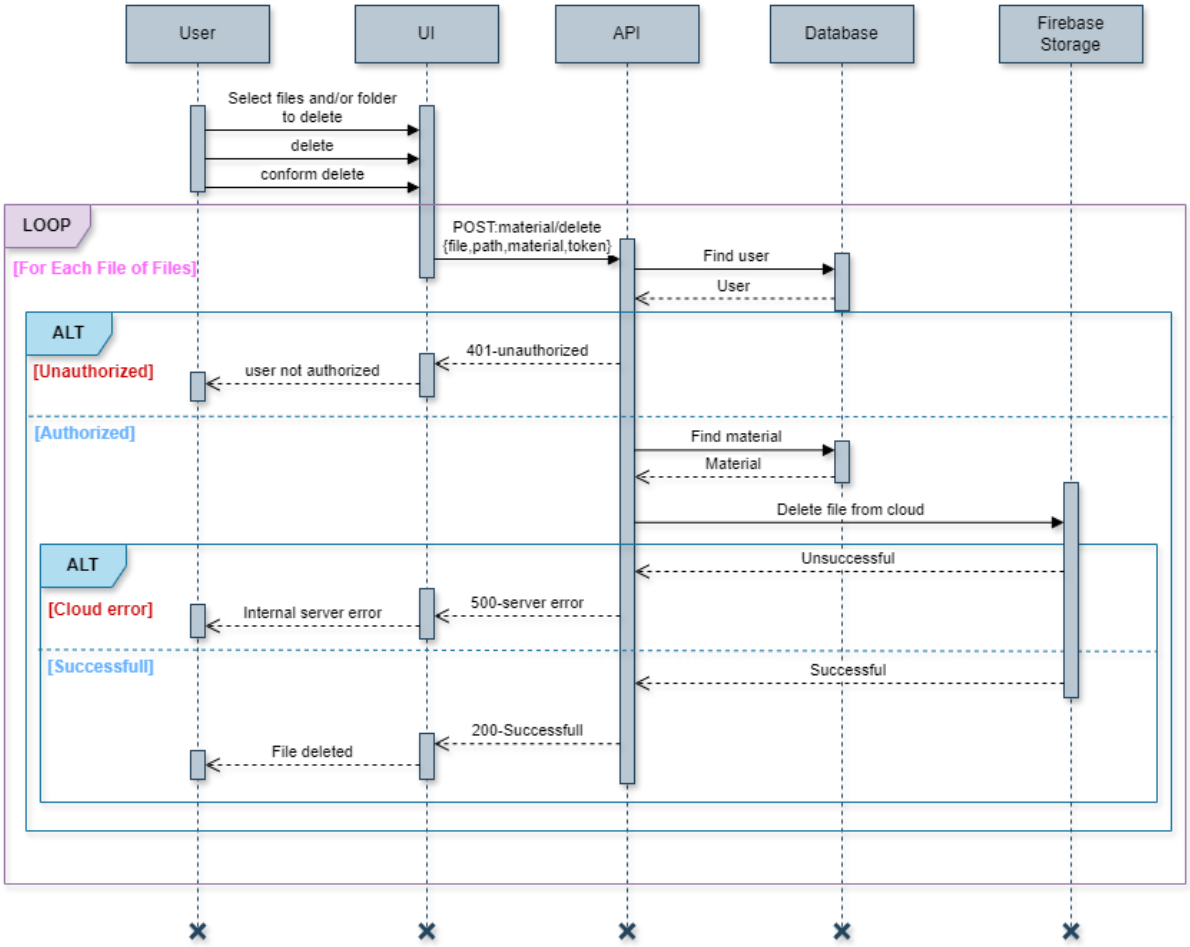

Figure 3.10 File/s Delete Sequence Diagram

3.3.9 Comment

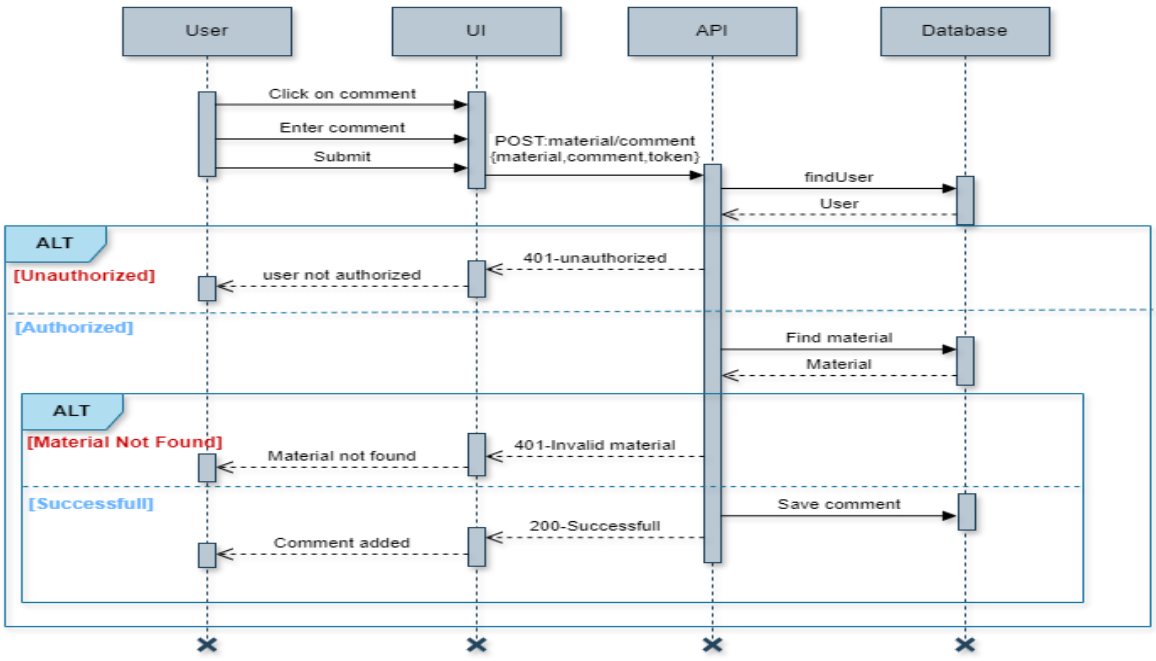

Figure 3.11 Comment Sequence Diagram

3.3.10 Forgot Passoword

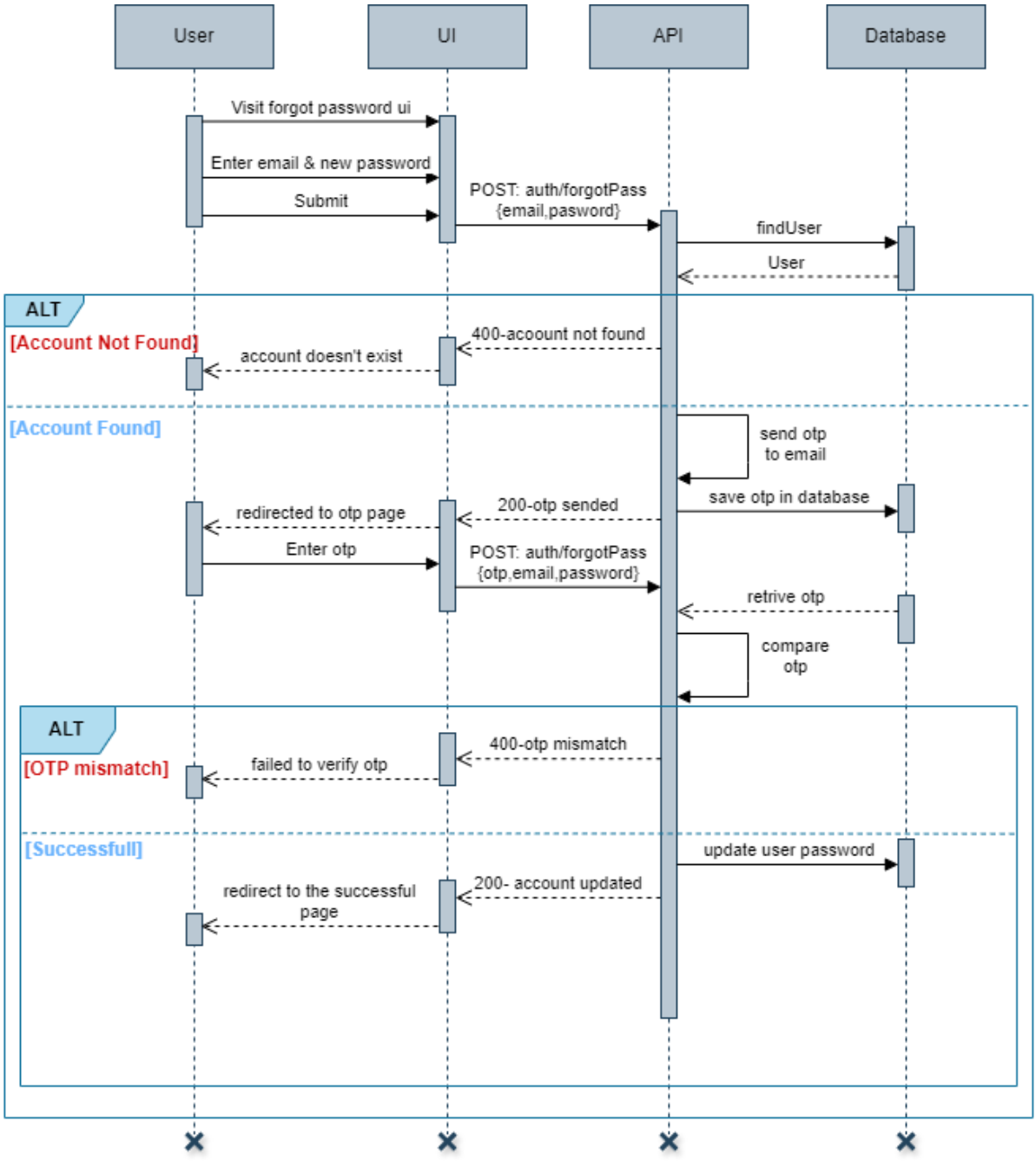

Figure 3.12 Forgot Password Sequence Diagram

3.3.11 Join material

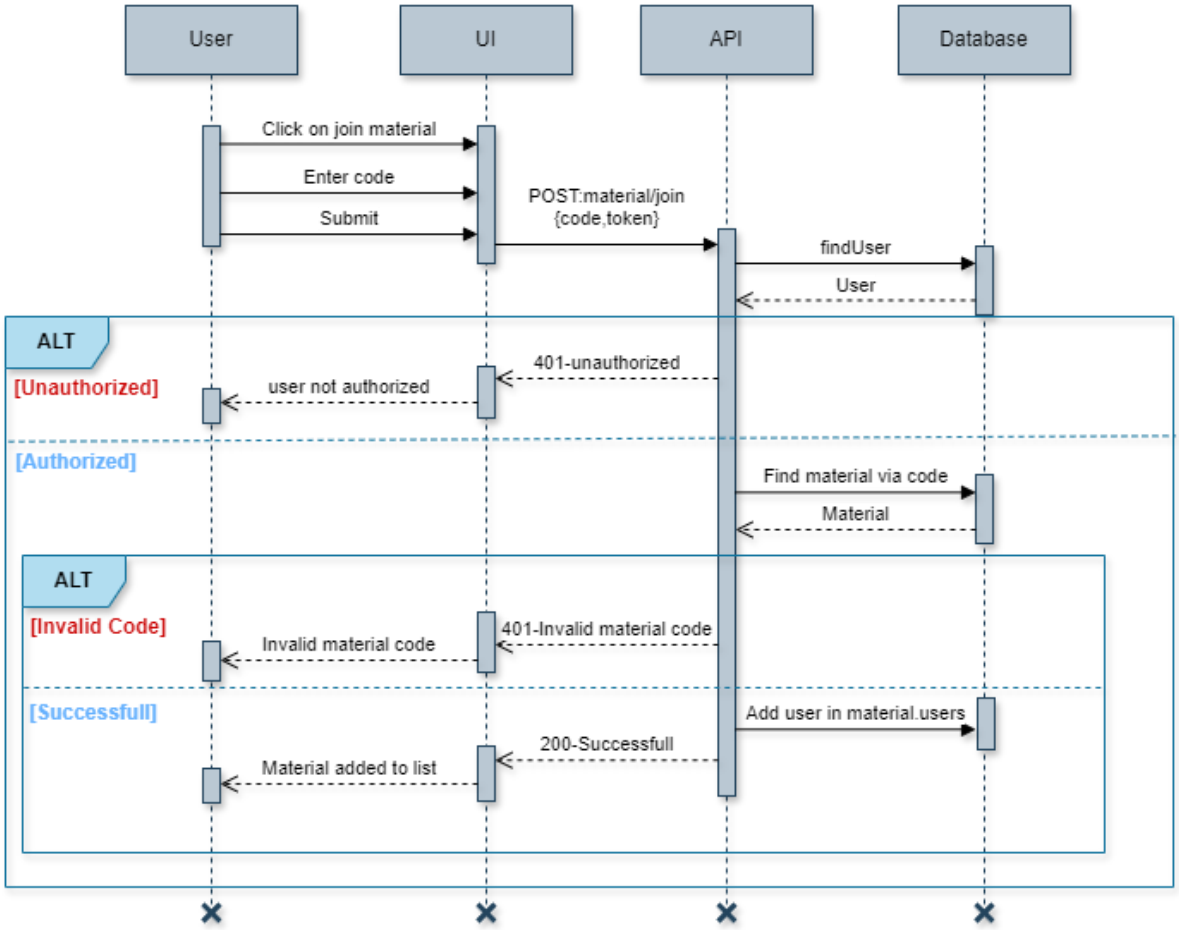

Figure 3.13 Join Material Sequence Diagram

3.4 Authentication Database Design

3.4.1 User Schema

```
const userSchema = new mongoose.Schema(
    {
        name:{type:String,required:true,trim:true,minLength:10,maxLength:50},
        email:{type:String,required:true,unique:true,trim:true},
        password:{type:String,required:true,trim:true,
            minLength:10,maxLength:50},
    },
    {
        timestamps:true
    }
);
```

3.4.2 OTP Schema

```
const otpSchema = new mongoose.Schema(
    {
        email:{type:String,required:true},
```

```
        otp:{type:String,required:true},
        times:{type:Number,required:true},
        createdAt:{type:Date,default:()=>Date.now(),expires:600}
    }
)
```

3.5 Share-Hub Database Design

3.5.1 Material Schema

```
const materialSchema = new mongoose.Schema(
    {
        creator:{type:mongoose.Schema.Types.ObjectId,ref:'User',re-
quired:true},
        name:{type:String,required:true,trim:true,minLength:4,maxLength:50},
        desc:{type:String,trim:true,required:true,minLength:10,maxLength:500},
        likes:[{type:mongoose.Schema.Types.ObjectId,ref:'User'}],
        comments:[{type:mongoose.Schema.Types.ObjectId,ref:'Comment'}],
        code:{type:String,required:true,unique:true,trim:true},
        users:[{type:mongoose.Schema.Types.ObjectId,ref:'User'}]
    }
)
materialSchema.index({code:1});
```

3.5.2 User Schema

```
const category = ['Owner','Editor','Viewer','Collaborator'];
const userSchema = new mongoose.Schema(
    {
        email:{type:String,required:true,unique:true,trim:true},
        materials:[{material:{type:mongoose.Schema.Types.ObjectId,ref:'Materi-
al'},role:{type:String,enum:category,defalut:'Viewer'}}],
        name:{type:String,required:true,minLength:10,maxLength:50},
        accessInfo:{
            Editor:{share:{type:Boolean,default:false},upload:{type:Bool-
ean,default:true},delete:{type:Boolean,default:true},download:{type:Bool-
ean,default:true}},
            Viewer:{share:{type:Boolean,default:false},upload:{type:Bool-
ean,default:false},delete:{type:Boolean,default:false},download:{type:Bool-
ean,default:true}},
            Collaborator:{share:{type:Boolean,default:true},upload:{type:Bool-
ean,default:true},delete:{type:Boolean,default:true},download:{type:Bool-
ean,default:true}},
            Owner:{share:{type:Boolean,default:true},upload:{type:Boolean,de-
fault:true},delete:{type:Boolean,default:true},download:{type:Boolean,de-
fault:true}},
        }
    },
);
```

3.5.3 Notification Schema

```
const categories = ['Invitation','News','Html'];
const notificationSchema = new mongoose.Schema(
    {
        by:{type:mongoose.Schema.Types.ObjectId,ref:'User',required:true},
        to:{type:mongoose.Schema.Types.ObjectId,ref:'User'},
        category:{type:String,enum:categories,required:true},
        fields:{type:mongoose.Schema.Types.Mixed,required:true},
        toAll:{type:Boolean,default:false},
        seen:{type:Boolean,default:false}
    }
)
```

3.5.4 Comment Schema

```
const commentSchema = new mongoose.Schema(
    {
        by:{type:mongoose.Schema.Types.ObjectId,required:true,ref:'User'},
        likes:[{type:mongoose.Schema.Types.ObjectId,ref:'User'}],
        disLikes:[{type:mongoose.Schema.Types.ObjectId,ref:'User'}],
        comment:{type:String,required:true}
    },
    {
      timestamps:true
    }
)
```

4 Implementation Details

4.1 Technology Stack

Share-Hub is developed using the MERN (MongoDB, Express.js, React.js, Node.js) stack. MongoDB serves as the database, Express.js handles server-side logic, React.js powers the frontend, and Node.js supports backend operations.

4.2 Development Environment

WebStorm IDE is employed for Share-Hub's development, offering a suite of tools for efficient coding. Git and GitHub are utilized for version control and collaboration management.

4.3 Implementation

To ensure the efficient and scalable implementation of Share-Hub, **two separate projects** have been developed: **Authentication and Share-Hub.**

1. Authentication

The Authentication project is responsible for managing user accounts and authentication-related tasks such as login, registration, and password recovery. It provides a robust and secure framework for user management operations, ensuring that user data is handled with most confidentiality and integrity.

2. Share-Hub

Share-Hub is a standalone project solely dedicated to document management functionalities. It serves as the core platform for users to upload, share, and collaborate on documents seamlessly. By focusing exclusively on document management, Share-Hub aims to streamline the sharing process and enhance collaboration efficiency.

Both projects, Authentication, and Share-Hub, share the same database server, ensuring consistency and integrity of user data across the system. However, they utilize different backend servers to handle their respective functionalities. This architecture allows for efficient data management while maintaining separation between user management and document management operations.

Let's take a closer look at each component, exploring their implementations, one by one and take Overview of some important functions.

4.4 Directory Structure

4.4.1 Backend

Below is the directory structure of the backend

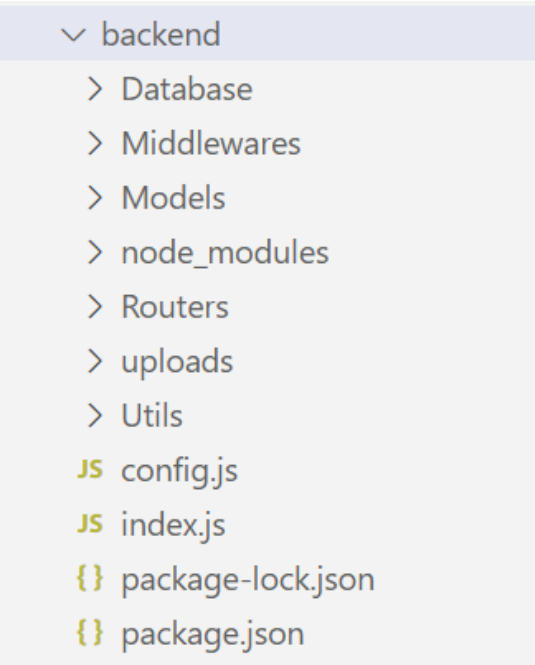

Figure 4.1 Backend Directory Structure

1. Database:

This directory contains the file defining the method for establishing a connection to the database.

2. Middlewares:

Within the Middlewares directory, middleware functions are stored. These functions intercept incoming HTTP requests and perform tasks such as authentication, error handling, or data validation before passing control to the route handlers.

3. Models:

The Models directory houses the schema definitions for the data models used in the application.

4.Routers:

The Routers directory contains route handler functions for processing incoming HTTP requests. Each router file typically handles requests for a specific resource or endpoint in the application, defining the corresponding operations.

5.Uploads:

The Uploads directory contains a files which should be uploaded to cloud while uploading it first stored in this directory. After successfully uploading it should be removed from here.

6.Utils:

Various utility implementations, such as email sending and photo uploading to the cloud, are stored in this directory.

7. config.js:

The config directory contains configuration files for the backend application. This includes environment-specific configurations, such as database connection strings, API keys, or other settings that vary between development, testing, and production environments.

8. index.js:

The index file serves as the entry point of the backend application. It initializes the server, sets up middleware, connects to the database, and defines the routing logic. This file orchestrates the execution of the backend application.

4.4.2 Frontend

Below is the directory structure of the frontend

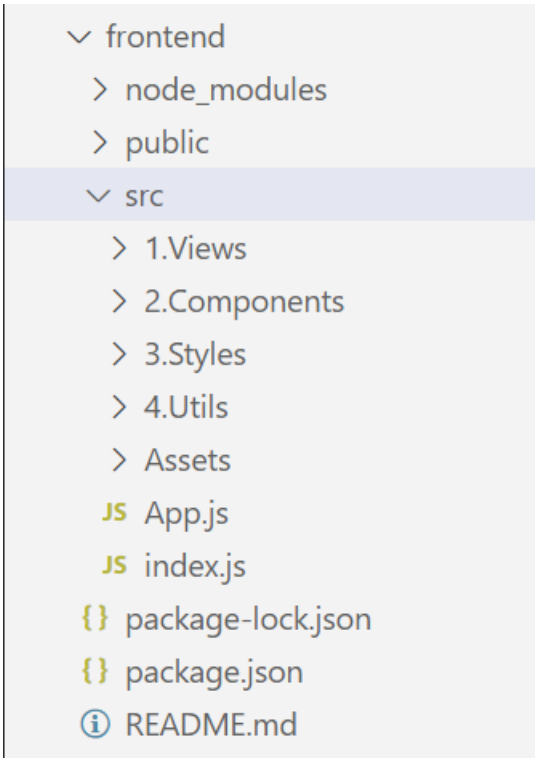

Figure 4.2 Frontend Directory Structure

1. Views:

The views directory contains the various view files or components that represent different pages or sections of the application. These files typically contain the markup and structure of the user interface.

2.Components:

Within the components directory, reusable UI components are stored. These components encapsulate specific UI elements or functionalities, making them easily reusable across different parts of the application.

3.Styles:

The styles directory houses the CSS or styling files for the frontend. This includes stylesheets, CSS modules, or any other styling-related files used to define the appearance and layout of the UI components.

4.Utils:

In the utils directory, utility functions or helper methods are stored. These utilities provide common functionalities or operations that are used across the application, such as formatting dates, handling API requests, or performing data manipulation.

5.Assets:

The assets directory contains static assets such as images, icons, fonts, or any other media files used in the frontend application. These assets are typically imported into components or views as needed.

6.App.js:

App.js serves as the main entry point of the frontend application. It is responsible for rendering the root component of the application and managing the overall application state.

7.index.js:

index.js is the entry point of the frontend application, where the ReactDOM render function is invoked to render the root component (usually App.js) into the HTML document. It initializes the frontend application and sets up any necessary configurations.

Now let's understand some modules and its functionalities.

4.5 Module Information

Throughout the implementation various modules and files designed and implemented. Among of them some important module's information given below

4.5.1 Authentication Module Information

1. **Middlewares**
   Within the authentication module, a middleware named "authenticate" is employed. Its primary function is to validate user requests, ensuring their authenticity. This process involves verifying whether a token is associated with the user request and subsequently determining its validity.

2. **Routers**

Two routers used in this module 1. authRouter and 2. userRouter

- **authRouter:** The authRouter encompasses a range of methods pertinent to authentication procedures. Presently, it facilitates functionalities such as new user registration, password recovery, user login, retrieval of the currently logged-in user, and token removal from the browser, consequently effectuating logout functionality.
- **userRouter:** In the authentication module, the userRouter is dedicated to managing various operations related to regular user activities. Its current functionalities encompass retrieving a user by their token, conducting searches based on email prefixes and returning users whose email prefixes match the specified criteria, as well as retrieving user information based on their unique user ID.

3. **Utils**

The utilities module, at present, houses essential functions designed to fulfill two primary tasks:

- **Sending Email Functionality:** This utility function orchestrates the sending of emails to users' specified email addresses, facilitating crucial communication channels for account-related notifications, verifications, and updates.

- **Profile Picture Upload to Firebase Storage:** This utility function manages the seamless uploading of user profile pictures to Firebase Storage, ensuring efficient storage, accessibility, and management within the application environment.

4.5.2 Share-Hub Module Information

1. **Middlewares**

Within the Share-Hub module, the middleware is responsible for the following functionalities:

- Verification of user tokens by making requests to the authentication module to ensure the validity of the user's request.
- Initialization of multer for storing files locally in the system for uploading to cloud services.

2. **Routers**

This module contains two routers 1. materialRouter and 2. userRouter

- **materialRouter:** The materialRouter in the Share-Hub module provides functionalities for material management, including creating, liking, sharing, and accessing materials. It also handles file-related tasks such as uploading to Firebase, managing folders, downloading files, and deleting files. Additionally, it supports commenting, sending invitations, managing notifications, and user list management.
- **userRouter:** The userRouter within the Share-Hub module handles various user management functionalities. These include searching for users based on email prefixes from the authentication module, retrieving access rights for specific users, and fetching and setting access rights for users.
- **Utils:** The utilities module within this module implements several key functionalities to facilitate smooth operations. These include uploading files using Firebase, retrieving download URLs for files, listing all directories and files within a given folder, and checking the existence of files in Firebase.

5 Testing

5.1 Testing Methodology Used

5.1.1 BlackBox Testing

In this section, we'll focus on the testing methodologies utilized in Share-Hub development, with a particular emphasis on black-box testing. Black-box testing is a software testing technique where the internal structure, design, or implementation details of the system are not known to the tester. Instead, the tester interacts with the system's external interfaces and observes its behaviour based on predefined inputs and expected outputs.

5.1.2 Advantages of Black-Box Testing

1. Independence from implementation details: Testers do not need access to the source code or knowledge of the system's internal design, making black-box testing suitable for testing third-party software or components.

2.Encourages thorough testing: Black-box testing encourages testers to explore various input combinations and scenarios, ensuring comprehensive test coverage and uncovering potential defects or inconsistencies in the system.

3.Emphasizes user perspective: By focusing on inputs and outputs from a user's perspective, black-box testing helps ensure that the system meets user requirements and expectations.

5.2 Test Suites

5.2.1 Register Functionality Testing

Test Case Id	Test Case Objective	Pre requisite	Steps	Expected Output	Actual Output	Status
TC_1.	Test for the user registration. (For the user not having account.)	User should not have the account.	1.Select the create account button. 2.Enter full name, email and password 3.Submit.	If mail is valid then OTP should be sent to the mail and user redirected to OTP page.	OTP received on the mail and redirected to the OTP page.	PASS

TC_2.	Test for invalid OTP for the registration	OTP should be sent to the mail.	1.Enter any OTP other than one received on the mail.	The error message should be shown indicating the invalid OTP.	The error message shown indicating invalid OTP.	PASS
TC_3.	Test for valid OTP for the registration	OTP should be sent to the mail.	1.Enter valid OTP received on the mail.	User account should be created and redirected to the Successful page.	User account created and redirect to the successful page.	PASS
TC_4.	Test for enter OTP after the 10 minutes (after expiration of OTP)	Ten minutes should have passed since the OTP was sent.	1.Enter valid OTP received on the mail.	The error message should be shown indicating the invalid OTP.	The error message shown indicating invalid OTP.	PASS
TC_5.	Test for sending more than 5 OTPs for the same email.	The user has already sent OTP five times.	1.Try to send OTP one more time by filling the name, email and password field in the registration form.	The error message should be shown indicates that the maximum OTP sending attempt limit has been exceeded. Please try again after some time.	The error message shown indicating maximum OTP sending attempt limit has been exceeded.	PASS
TC_6.	Test for the user registration. (For the user having existing account)	User should have account.	1.Select the create account button. 2.Enter full name, email and password 3.Submit.	The error message should be shown indicating user account already exist.	The error message shown indicating user account already exist.	PASS
TC_7.	Test for register with long name or too short name and/or password (<5 or >50 characters)	User should not have account.	1.Select the create account button. 2.Enter full name, email and password 3.Submit.	The error message should be shown indicating too small or big name and/or password.	The error message indicating the too small or big name and/or password.	PASS

Table 5.1 Test Suite for Testing Register Functionality

5.2.2 Login Functionality Testing

Test Case Id	Test Case Objective	Pre requisite	Steps	Expected Output	Actual Output	Status
TC_1.	Test for the login with	No pre-requisite	1.Go to login page	The error message	The error message	PASS

	incorrect mail and correct password.		2.Enter invalid email and valid password.	should be shown regarding invalid credentials.	shown indicating invalid credentials.	
TC_2.	Test for the login with incorrect mail and incorrect password.	No pre-requisite	1.Go to login page 2.Enter invalid email and valid password.	The error message should be shown regarding invalid credentials.	The error message shown indicating invalid credentials.	PASS
TC_3.	Test for the login with correct mail and incorrect password.	No pre-requisite	1.Go to login page 2.Enter invalid email and valid password.	The error message should be shown regarding invalid credentials.	The error message shown indicating invalid credentials.	PASS
TC_4.	Test for the login with correct mail and correct password.	User should already have account.	1.Go to login page 2.Enter valid email and valid password.	The user should be logged in and redirected to the main page.	Successful login and user redirected to the main page.	PASS

Table 5.2 Test Suite for Testing Login Functionality

5.2.3 Password Recovery Testing

Test Case Id	Test Case Objective	Pre requisite	Steps	Expected Output	Actual Output	Status
TC_1.	Test for forgot password. (For the user who doesn't have account)	User should not have account.	1.Go to forgot password page. 2.Enter email, new password and confirm new password field. 3.Submit	The error message should be shown indicating account not found.	The error message shown indicating account not found.	PASS
TC_2.	Test for forgot password. (Password and confirm password not matching.)	User should have account.	1.Go to forgot password page. 2.Enter email, new password and confirm new password field. 3.Submit	The error message should be shown indicating password and confirm password doesn't match.	The error message shown indicating password and confirm password doesn't match.	PASS
TC_3.	Test for forgot password. (For genuine user.)	User should have account.	1.Go to forgot password page. 2.Enter email, new password and confirm new password field. 3.Submit 4.Enter OTP received on mail and verify mail.	The user's password should be updated and redirected to the successful page.	The user's password changed and redirected to the successful page.	PASS

Table 5.3 Test Suite for Testing Password Recovery Functionality

5.2.4 Material create, delete, leave testing

Test Case Id	Test Case Objective	Pre requisite	Steps	Expected Output	Actual Output	Status
TC_1.	Test for create material with too small and or too big name (<4 or >50 characters) and/or description. (<10 or >500 characters)	User should be logged in.	1.Click on the create material in home page. 2.Enter name and description for the material. 3.Submit	The error message should be shown indicating the too small or too big length of the name and/or description.	The error message shown indicating too small name and/or description	PASS
TC_2.	Test for create material with the appropriate name and description.	User should be logged in.	1.Click on the create material in home page. 2.Enter name and description for the material. 3.Submit	Material should be created and available at the home page.	Material created and available at home page.	PASS
TC_3.	Test for delete material when user is owner of the material.	User should be owner of the material and logged in.	1.Click on the delete material icon. 2.confirm the conformation.	Material should be deleted from the user's list and all content also removed from cloud.	Material deleted from the user's list and it's all content also removed from cloud.	PASS
TC_4.	Test for delete material when user is not owner (delete request through postman tool)	User should not be owner of the material and logged in.	1.send post request to material/deleteMaterial with fields material id and JWT token.	The error message should be shown indicating not having access.	The error message shown indicating not having access.	PASS
TC_5.	Test for leave material through postman.	User has not joined the material.	1.send post request to material/leaveMaterial with fields material id and JWT token.	The error message should be shown indicating material is not in your list.	The error message shown indicating the material is not in your list.	PASS
TC_6.	Test for leave material.	User has joined material.	1.send post request to material/leaveMaterial with fields material id and JWT token.	The material should be removed from the user's materials list.	The material is removed from the user's material list.	PASS

Table 5.4 Test Suite for The Material Create, Delete, Leave Functionality

5.2.5 Share material and like material testing

Test Case Id	Test Case Objective	Pre requisite	Steps	Expected Output	Actual Output	Status
TC_1.	Test the sharing of material done by user using postman.	User should not be owner of material.	1.send post request to material/sendInvitation with fields material id, peoples and JWT token.	The error message should be displayed indicating the fact that only owner can send invitation.	The error message shown indicating that only owner can send invitation.	PASS
TC_2.	Test the sharing of material done by user.	User should be owner of the material and logged in to system.	1.Click on share. 2.Enter email and select role. 3.Send invitation for joining material.	The notification for joining the material should be sent to specified users.	The notification for joining material sent to the specified users.	PASS
TC_3.	Test the sharing of material done by user.	User should be owner of the material and already sent the notification to the specified user and logged in to the system.	1.Click on share. 2.Enter email and select role. 3.Send invitation for joining material.	The old notification should be replaced by the newer one.	The old notification updated and new notification shown to user.	PASS
TC_4.	Test for the like material.	User should be logged in and not already liked material.	1.Click on the heart icon.	The like should be added to the material and like count should be increased.	The like added to the material and like count increased.	PASS
TC_5.	Test for unlike material.	User should be logged in and already liked material.	1.Click on the heart icon.	The like should be removed from the material and like count should be decreased.	The like removed from the material and like count decreased.	PASS

Table 5.5 Test Suite for Share Material and Like Material Functionality

5.2.6 Join material via code and via notification testing

Test Case Id	Test Case Objective	Pre requisite	Steps	Expected Output	Actual Output	Status
TC_1.	Test for joining	User should be logged in.	1.Click on join material. 2.Enter invalid material code.	The error message should be	The error message shown	PASS

	material via invalid code.			shown indicating invalid material code.	indicating invalid material code.	
TC_2.	Test for joining material via valid code.	User should be logged in.	1.Click on join material. 2.Enter valid material code.	The material should be added to the user's material list.	The material added to the user's material list.	PASS
TC_3.	Test for joining material via notification	User should be logged in.	1.Click on the notification icon 2.Click on the join button.	The material should be added to the user's material list.	The material added to the user's material list.	PASS
TC_4.	Test for rejecting the invitation for the joining material.	User should be logged in.	1.Click on the notification icon 2.Click on the join button.	The notification should be removed and no change in the user's material list.	The notification removed and no change in user's material list.	PASS

Table 5.6 Test Suite for Join Material via Code and Via Notification Functionality

5.2.7 File/s download and delete testing

Test Case Id	Test Case Objective	Pre requisite	Steps	Expected Output	Actual Output	Status
TC_1.	Test for the download file when user doesn't have access to download files Done using postman tool.	User should not have the access to download the file.	1.send post request to material/download with fields material id, files, folder and JWT token.	The error message should be shown indicating that user doesn't have access to download files.	The error message shown specifying user doesn't have access to download file.	PASS
TC_2.	Test for the material when user has access to download files and select only one file.	User should have right to download file and should be logged in.	1.Go to material page and select one file. 2.Click on the download button.	The file should be downloaded to the user's machine.	The file downloaded to the user's machine.	PASS
TC_3.	Test for the material when user has access to download files and user selects multiple files.	User should have right to download file and should be logged in.	1.Go to material page and select multiple files. 3.Click on the download button.	The zip file should be download-ed with name Unknown Study Files.	The zip file downloaded with name Unknown Study Files.	PASS

TC_4.	Test for the material when user has access to download files and user selects multiple files with the zip name.	User should have right to download file and should be logged in.	1.Go to material page and select multiple files. 2.Provide zip file name. 3.Click on the download button.	The zip file should be download-ed with provided name.	The zip file download-ed with provided name.	PASS
TC_5.	Test for the delete file when user doesn't have access to delete files. Done using postman tool.	User should not have the access to delete the file.	1.send post request to material/deletefile with fields material id, file, and JWT token.	The error message should be shown indicating that user doesn't have access to delete files.	The error message shown specifying user doesn't have access to delete file.	PASS
TC_6.	Test for the material when user has access to delete files.	User should have right to delete file and should be logged in.	1.Go to material page and select one/multiple files. 3.Click on the delete button.	The file/s should be removed from the cloud.	The file/s removed from the cloud.	PASS

Table 5.7 Test Suite for File Download and Delete Functionality

5.2.8 Upload file/s testing

Test Case Id	Test Case Objective	Pre requisite	Steps	Expected Output	Actual Output	Status
TC_1.	Test for the uploading file when user does not have the access to upload. Done using postman tool.	User should not have the access to upload the file.	1.send post request to material/upload with fields material id, file, path, and JWT token.	The error message should be shown indicating that user doesn't have access to upload files.	The error message shown specifying user doesn't have access to upload file.	PASS
TC_2.	Test for the uploading file when user has access to upload file and upload multiple files.	User should be logged in and has access to upload file.	1.Go to material page and select upload file button. 2.Select file/s from file picker. 3.Click on upload.	The file/s should be uploaded to cloud and at last total uploaded and failed files shown (if any).	The file/s uploaded to cloud and at last total uploaded and failed files shown (if any).	PASS
TC_3.	Test for the uploading file when user has access to upload file and upload folder and files.	User should be logged in and has access to upload file.	1.Go to material page and select upload file button. 2.Select folder and file/s from file picker. 3.Click on upload	All the files within the folder should be uploaded and relative path should be maintained in cloud.	All the files within the folder were uploaded and relative path also maintained in cloud.	PASS

TC_4.	Test for selecting file again using file picker which are already selected in first time.	User should be logged in and has access to upload file.	1.Go to material page and select upload file button. 2.Select file/s from file picker. 3.Close file picker and again select same file. 3.Click on upload	Files which are selected twice should be considered as once and uploaded.	Files which are selected twice were considered and uploaded. (means same file uploaded twice as it is selected two time.)	FAIL

Table 5.8 Test Suite for Upload File Functionality

6 Screenshots

6.1 Register page UI

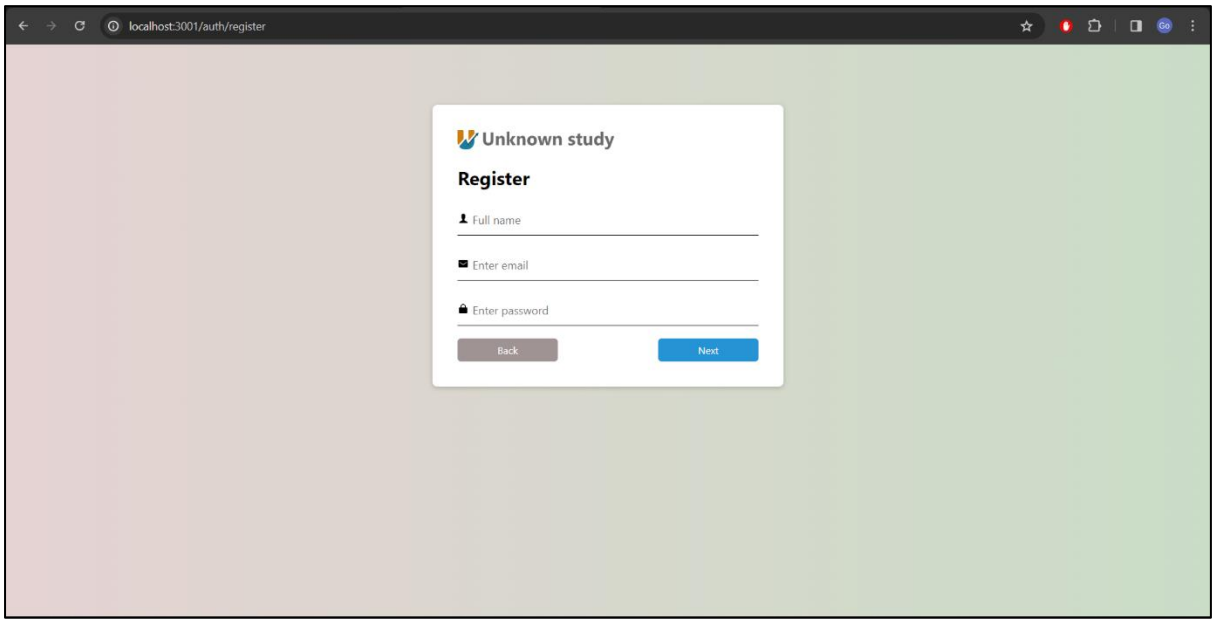

Figure 6.1 Register Page UI

6.2 Login page UI

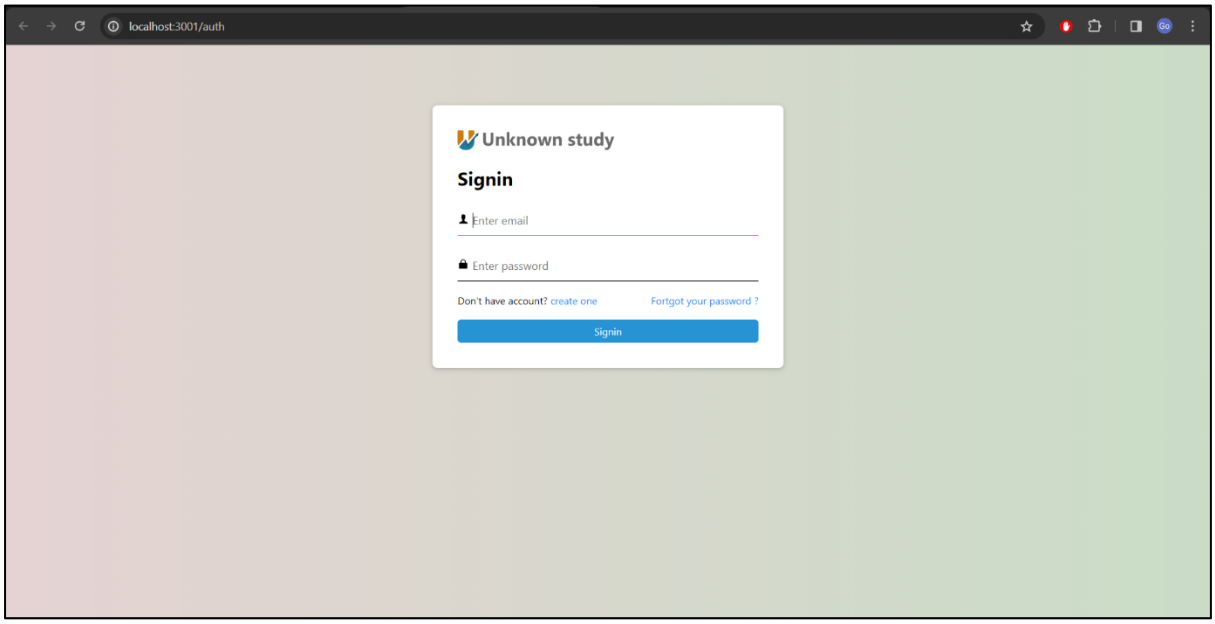

Figure 6.2 Login Page UI

6.3 OTP page UI

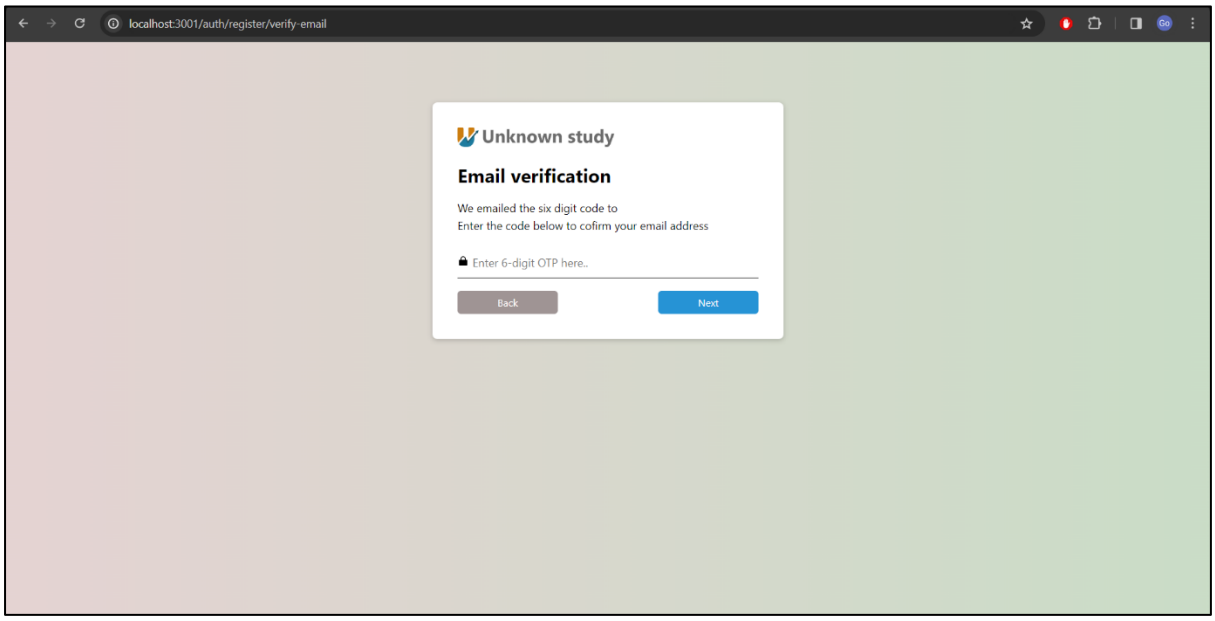

Figure 6.3 OTP Page UI

6.4 Forgot password page UI

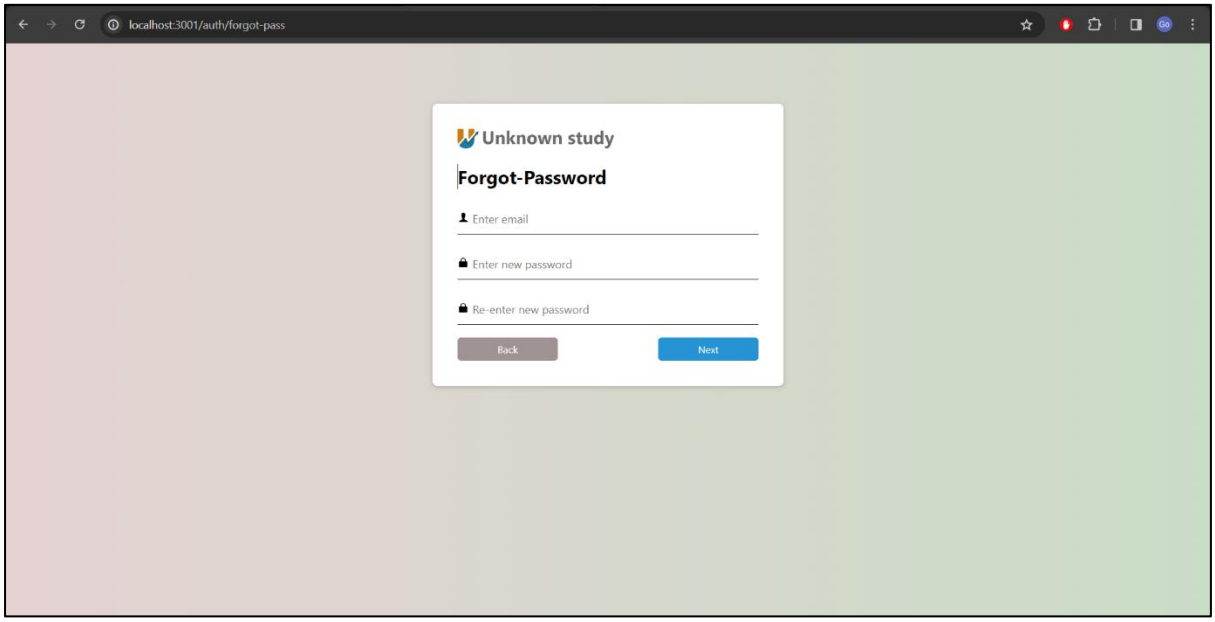

Figure 6.4 Forgot Password UI

6.5 Home page UI

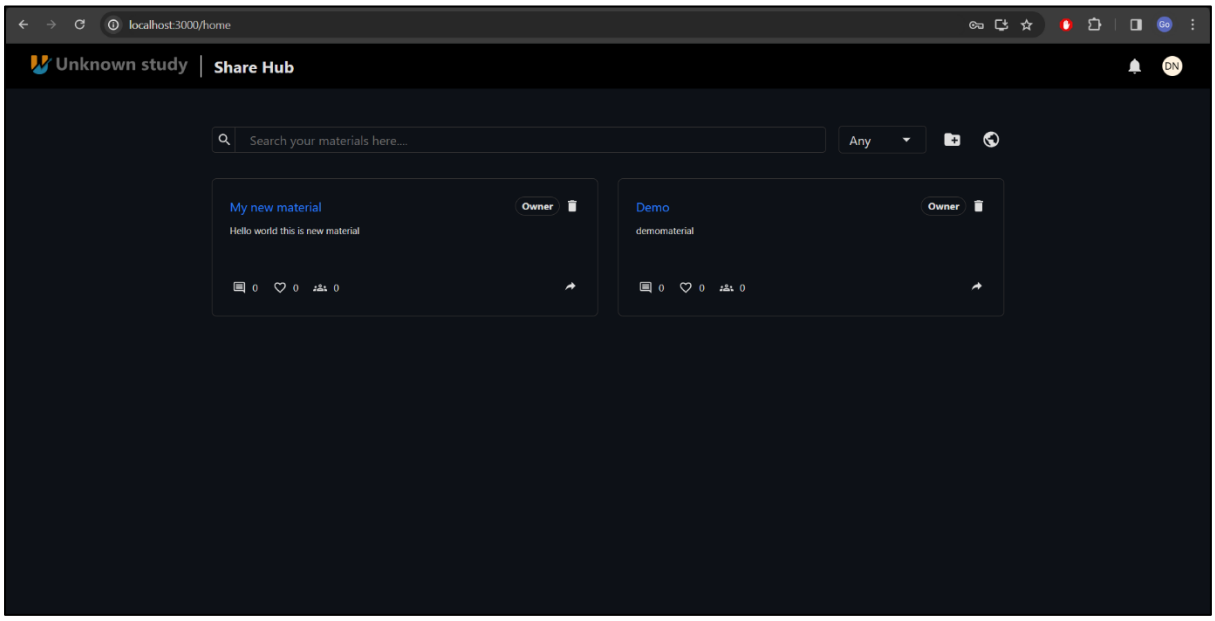

Figure 6.5 Home Page UI

6.6 Material Create UI

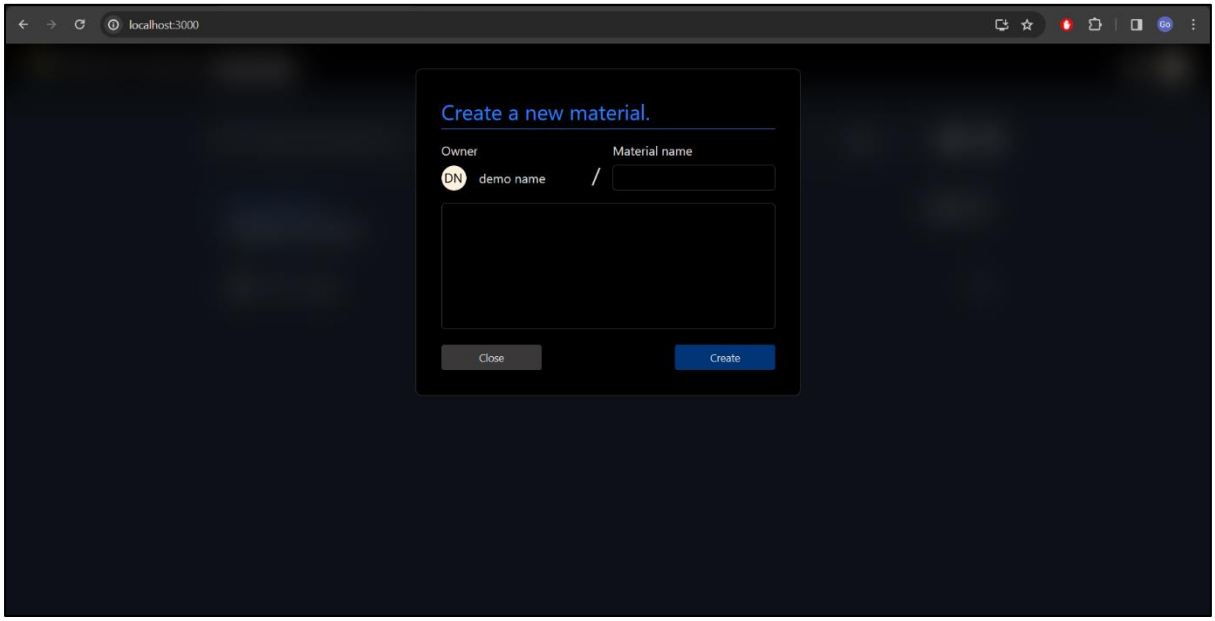

Figure 6.6 Material Create UI

6.7 Join Material UI

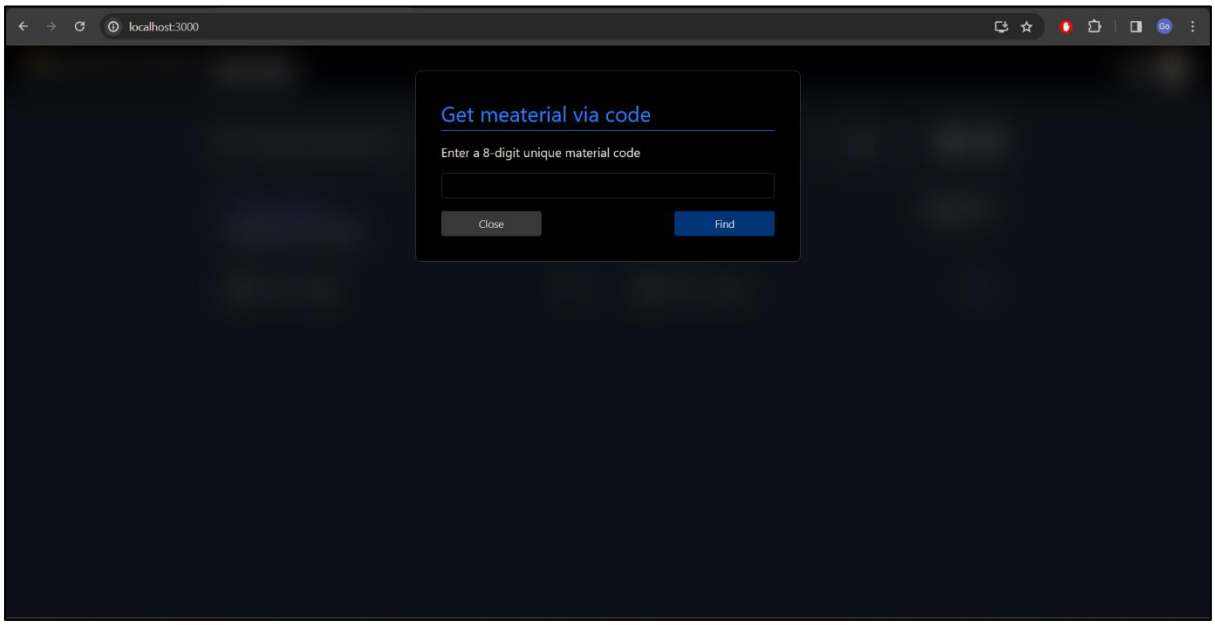

Figure 6.7 Join Material UI

6.8 Material Page UI

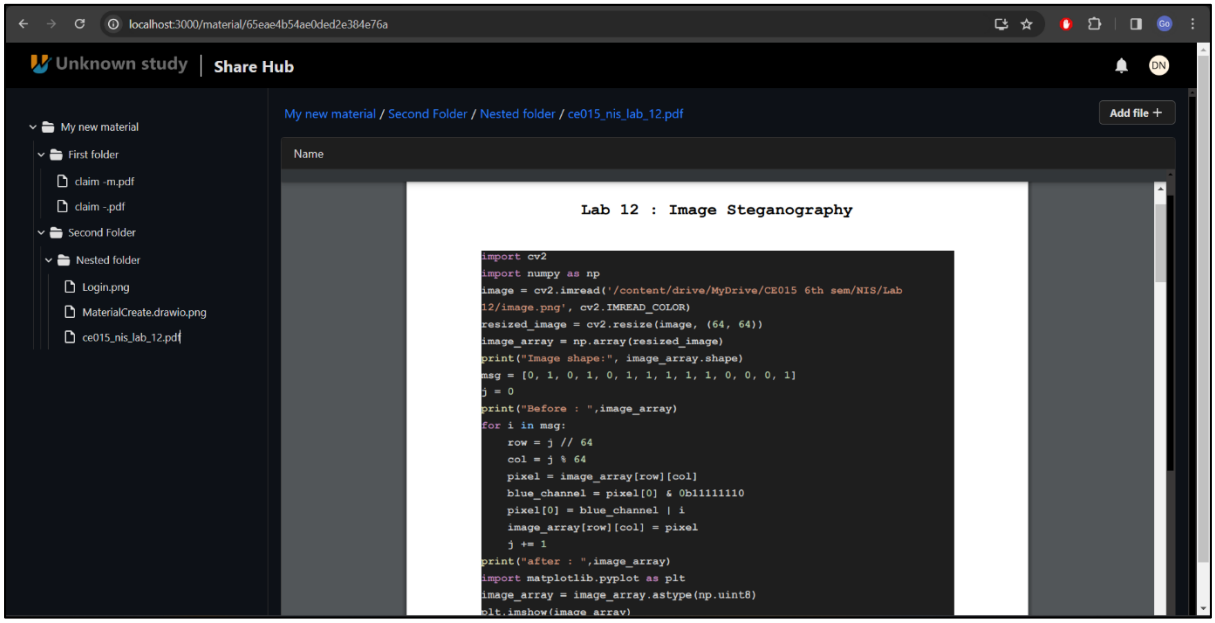

Figure 6.8 Material Page UI-I

6.9 Material page – 2 UI

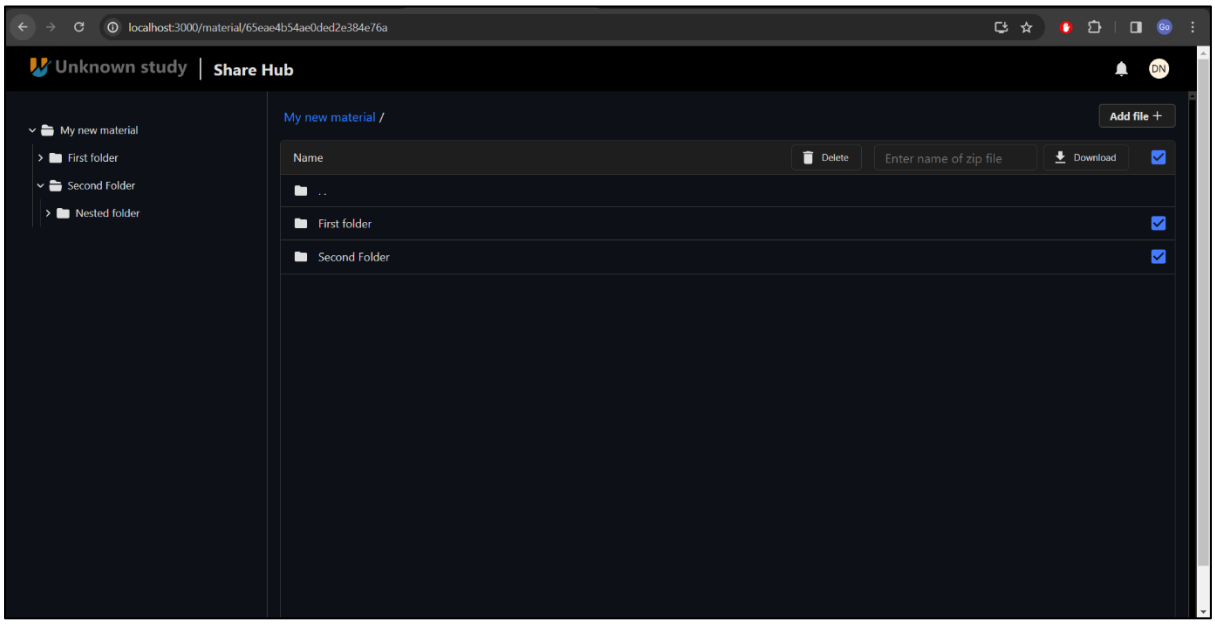

Figure 6.9 Material Page UI-II

6.10 Upload screen UI

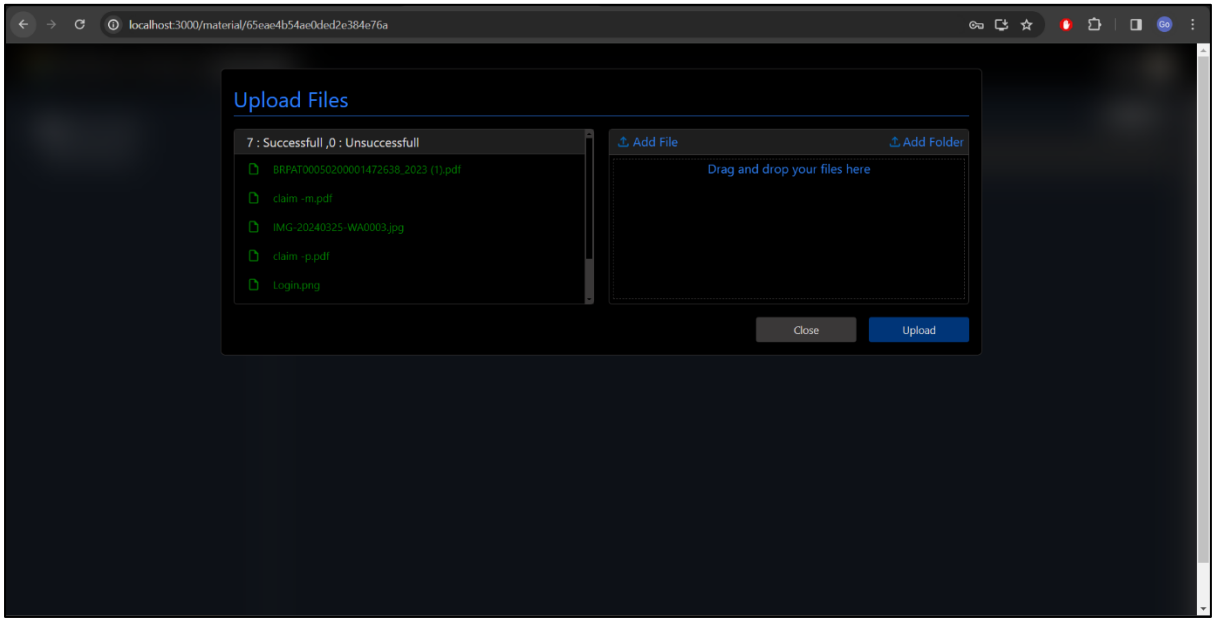

Figure 6.10 Upload Screen UI

6.11 Comment Screen UI

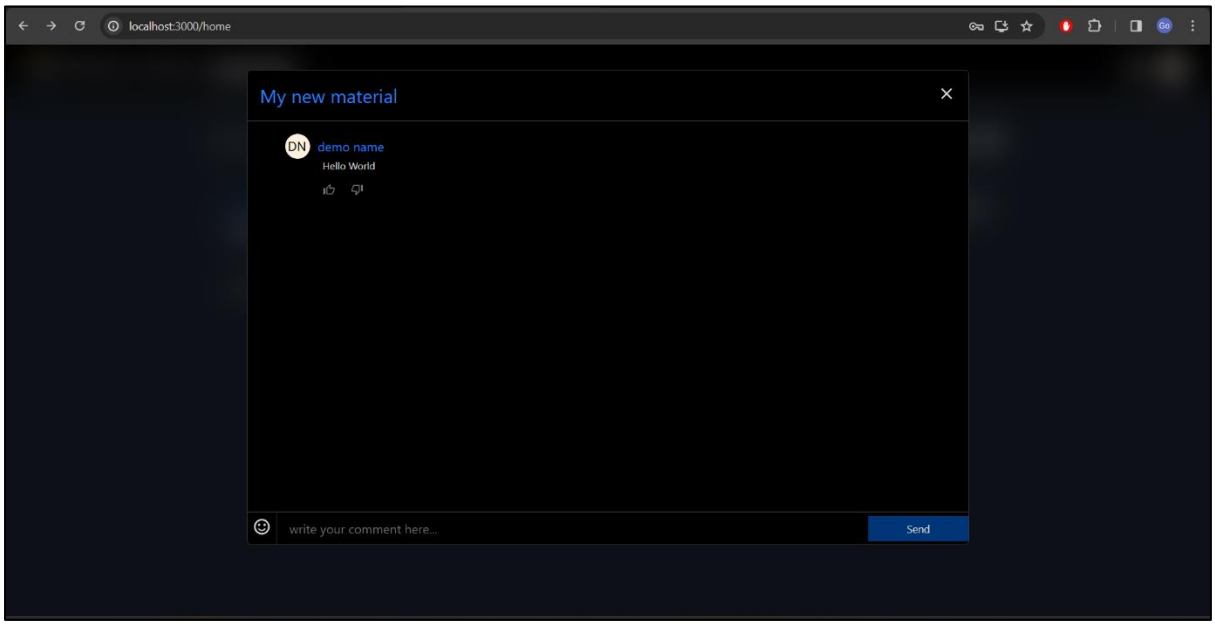

Figure 6.11 Comment Scree UI

6.12 Share Material UI

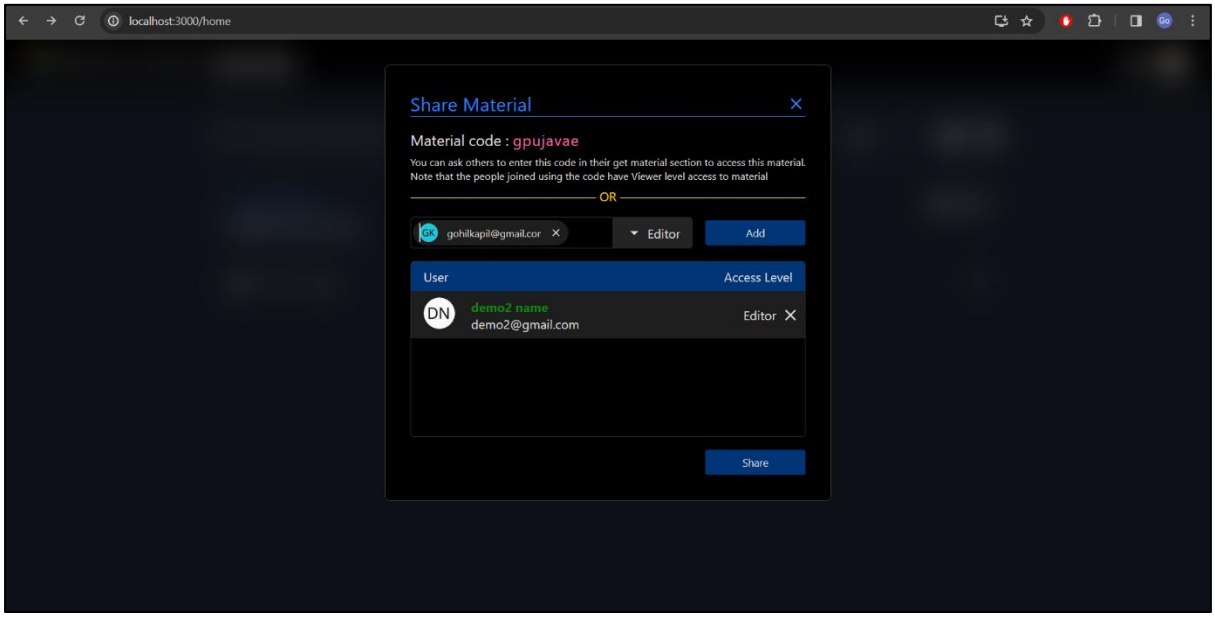

Figure 6.12 Share Material UI

6.13 Setting page

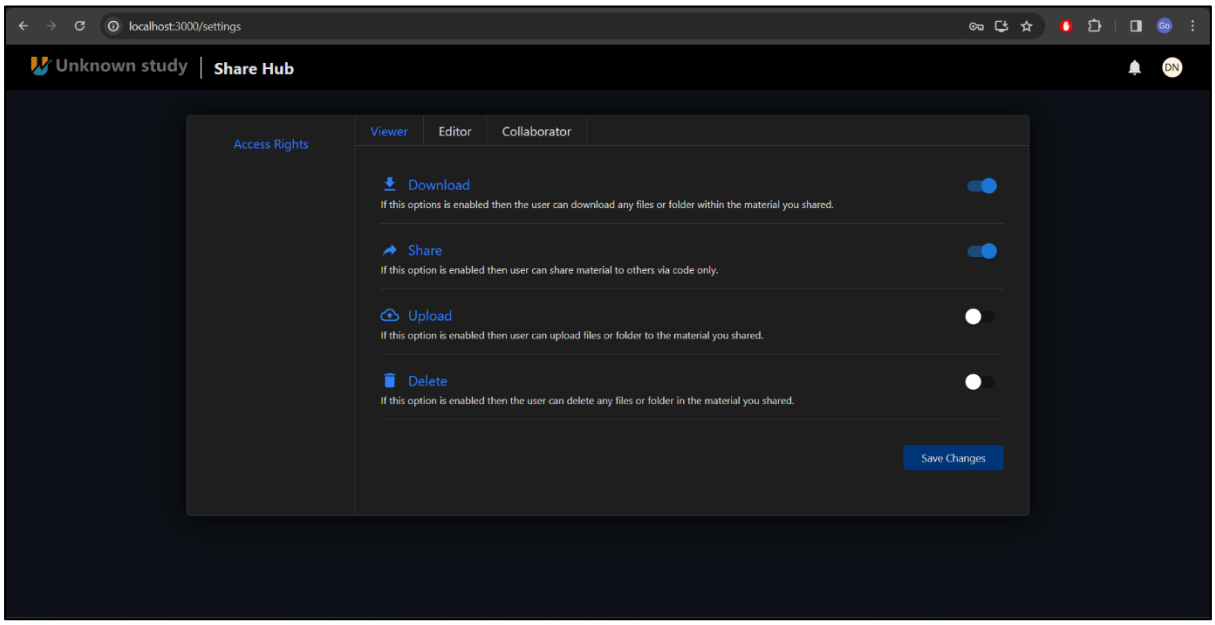

Figure 6.13 Setting Page UI

7 Conclusion

The primary objective of the Share Hub project was to create a robust and user-friendly platform for seamless document sharing. Leveraging the advantages of the MERN stack technology facilitated rapid development, ensuring that our system met the requirements for speed, efficiency, and scalability. By utilizing Firebase Storage, a secure and reliable solution for document storage was established, guaranteeing the safekeeping and accessibility of files as needed.

Throughout the development process, range of essential features implemented, including authentication functionalities such as login, registration, and password recovery. Additionally, key document management features such as file upload, download, and deletion were seamlessly integrated. Moreover, advanced functionalities like material creation, sharing with access control, chat options, like buttons, and notifications were incorporated to enhance user experience and interaction.

Following thorough testing of the project's functionalities, it is confirmed that the system performs optimally across all specified features. However, acknowledging that continuous improvement is essential to stay competitive in the market, the commitment remains to refine and enhance the platform to meet evolving user needs and emerging industry standards. This project has provided invaluable insights into aspects such as time management, critical thinking, and cloud integration for storage solutions, resembling a real-world content management system experience.

Looking ahead, our focus will be on further upgrades and enhancements to transform Share Hub into an even more robust and efficient application, ensuring its continued success and relevance in the dynamic landscape of document sharing platforms.

8 Limitation and Future Extension

8.1 Limitation

While considerable effort has been devoted to enhancing the Share Hub platform, several limitations persist, as detailed below:

1. **Lack of Move Option:** Share Hub currently lacks a move option, hindering users from relocating materials between folders without resorting to re-uploading. The absence of a cut-and-paste functionality diminishes user efficiency.
2. **Absence of Rename Option:** Users are unable to rename materials they have created, limiting customization and organizational capabilities within the platform.
3. **Insufficient File Information Display:** Share Hub does not display comprehensive file information such as upload date, file size, or last modified date, potentially impeding users' ability to manage and track their documents effectively.
4. **Missing Favourite Marking Option**: The platform lacks a feature to mark materials as favourites, depriving users of the ability to prioritize or easily access frequently used documents.
5. **No Recycle Bin Functionality:** Share Hub does not include a recycle bin feature, meaning deleted materials are permanently removed from the system, posing a risk of data loss in the event of accidental deletion.
6. **Limited File Preview Support:** While Share Hub supports file previews for certain formats such as PDF, additional file types lack preview functionality, potentially limiting users' ability to assess content before opening. Supported file extensions include PDF, C, CPP, Java, TXT, C++, JS, IN, OUT, SCSS, JSON, CSV, TSV, JPEG, AVIF, JPG, JPEG, JFIF, PNG, SVG etc. Most of code file extensions supported but it lacks at Microsoft Office documents (e.g., Word, Excel, PowerPoint) and multimedia files (e.g., MP4, MP3).
7. **Restricted Storage in Firebase Free Tier**: Share Hub's reliance on the Firebase free tier imposes a storage limit of 5 GB, potentially restricting users' capacity to store and manage large volumes of documents.
8. **Lack of User Storage Usage Information:** Share Hub does not provide users with visibility into their storage usage, preventing them from monitoring and managing their allocated storage effectively.

9. **No Storage Limit Management:** The platform does not enforce limits on individual user storage usage, leading to potential overuse and resource allocation issues.

While Share Hub continues to evolve, addressing these identified limitations will be crucial to enhancing user experience and maximizing the platform's utility. Further exploration may uncover additional limitations that warrant attention and resolution.

8.2 Future Extension

As Share Hub evolves, several future extensions could be considered to enrich its features and capabilities:

1. **Move Option Enhancement**: Implement a more robust move option that allows users to seamlessly transfer materials between folders with ease, including the ability to drag and drop files.
2. **Dynamic Renaming:** Introduce dynamic renaming functionality that enables users to rename materials directly within the platform, improving organizational flexibility and user experience.
3. **Comprehensive File Information Display:** Enhance the file information display to include additional details such as file type, author, version history, and access permissions, providing users with a more comprehensive overview of their documents.
4. **Favorites Management:** Develop a favorites management system that enables users to create, organize, and access favorite materials conveniently, facilitating quick access to frequently used documents.
5. **Recycle Bin Feature:** Implement a recycle bin feature that allows deleted materials to be temporarily stored in a designated folder, enabling users to recover accidentally deleted documents and reduce the risk of data loss.
6. **Expanded File Preview Support:** Extend file preview support to include a wider range of file formats, such as Microsoft Office documents (e.g., Word, Excel, PowerPoint) and multimedia files (e.g., MP4, MP3) for enhancing users' ability to preview and interact with various types of content.
7. **Enhanced Storage Options:** Introduce additional storage options beyond the Firebase free tier, such as premium storage plans with increased capacity and advanced features, catering to the needs of users with larger document repositories.

8. **User Storage Usage Analytics:** Provide users with detailed analytics and insights into their storage usage, including usage trends, space allocation breakdowns, and recommendations for optimizing storage efficiency.
9. **Storage Limit Management Tools:** Develop tools and controls for administrators to set and enforce storage limits for individual users or user groups, ensuring fair resource allocation and preventing storage overuse.
10. **Collaborative Editing Features:** Explore collaborative editing features that enable multiple users to simultaneously edit and collaborate on documents in real-time, fostering teamwork and productivity.
11. **Advanced Search Functionality:** Enhance the search functionality to support advanced search criteria, filters, and operators, enabling users to quickly locate specific documents based on various attributes and keywords.
12. **Integration with Third-Party Services:** Integrate Share Hub with popular third-party services and platforms such as Google Drive, Dropbox, and Microsoft OneDrive, allowing users to seamlessly access and manage documents across multiple cloud storage providers.

These future extensions aim to enhance Share Hub's functionality, usability, and versatility, empowering users to manage, collaborate on, and access documents more efficiently and effectively. As Share Hub evolves, prioritizing these extensions based on user feedback and market trends will be essential to ensuring its continued success and relevance in the document management landscape.

9 Bibliography

The following resources were consulted throughout the development process of Share-Hub.

1. Sandy Dev. "How to Implement OTP Verification in Authentication System with Express.js and MongoDB." Medium, 2022. [Online]. Available: https://sandydev.medium.com/how-to-implement-otp-verification-in-authentication-system-with-express-js-and-mongodb-c4f1c1314aed.
2. Devmhn07. "Connect MongoDB Database to Express Server: Step-by-Step." Medium, 2022. [Online]. Available: https://medium.com/featurepreneur/connect-mongodb-database-to-express-server-step-by-step-53e548bb4967.
3. Devmhn07. "JWT Authentication in Node.js: A Practical Guide." Medium, 2022. [Online]. Available: https://dvmhn07.medium.com/jwt-authentication-in-node-js-a-practical-guide-c8ab1b432a49.
4. MongoDB. "Express MongoDB REST API Tutorial." MongoDB, 2022. [Online]. Available: https://www.mongodb.com/languages/express-mongodb-rest-api-tutorial.
5. MongoDB. "Query Arrays." MongoDB Documentation, 2022. [Online]. Available: https://www.mongodb.com/docs/manual/tutorial/query-arrays/.
6. OTP Generator. "OTP Generator." npm, 2022. [Online]. Available: https://www.npmjs.com/package/otp-generator.
7. Firebase. "Firebase Storage Documentation." Google Firebase, 2022. [Online]. Available: https://firebase.google.com/docs/storage/web/start.
8. React. "React Documentation." React, 2022. [Online]. Available: https://react.dev/.
9. W3Schools. "React useContext Hook." W3Schools, 2022. [Online]. Available: https://www.w3schools.com/react/react_usecontext.asp.
10. freeCodeCamp. "Simplify Your File Upload Process in Express.js." freeCodeCamp, 2022. [Online]. Available: https://www.freecodecamp.org/news/simplify-your-file-upload-process-in-express-js/.
11. Google Cloud. "Google App Engine Documentation." Google Cloud, 2022. [Online]. Available: https://cloud.google.com/appengine/docs/.