HIMALAYA DARSHAN COLLEGE

Tribhuvan University

Institute of Science and Technology

A Project Report On

"Alnote"

A STUDY NOTES ARCHIVAL AND DISTRIBUTION SYSTEM

In Partial Fulfilment of Requirements for the Bachelor Degree in Computer Science and Information Technology (BSc. CSIT)

Submitted to

Department of Computer Science and Information Technology

Himalaya Darshan College

Biratnagar, Nepal

Submitted by

Dip Chandra Ojha (27319/077)

Gaurav Raj Bana (27320/077)

SaimanKatwal (27336/077)

Date: 7th Falgun, 2081

Tribhuvan University

Faculty of Computer Science and Information Technology

Himalaya Darshan College

Supervisor's Recommendation

I hereby recommend that this project prepared under my supervision by dip chandra ojha (27319/077) , Gaurav Raj Bana (27320/077) and SaimanKatwal (27336/077) entitled **"STUDY NOTES ARCHIVAL AND DISTRIBUTION SYSTEM"** in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and information Technology is recommended for the final evaluation.

SIGNATURE

Mr. Kushal Niroula

SUPERVISOR

Project Coordinator

Department of IT

Himalaya Darshan College

Tribhuvan University

Faculty of Computer Science and Information Technology

Himalaya Darshan College

LETTER OF APPROVAL

This is to certify that this project prepared by Dip Chandra Ojha, Gaurav Raj Bana and SaimanKatwal entitled **"STUDY NOTES ARCHIVAL AND DISTRIBUTION SYSTEM"** in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and information Technology has been evaluated. In our opinion, it is satisfactory in the scope and quality as a project for the required degree.

_____ **Kushal NiroulaSumit Kumar Shah** **Supervisor** Lecturer Himalaya Darshan College	_____ **HOD** Department of IT Himalaya Darshan College
_____ **Internal Examiner** Himalaya Darshan College Roadcess Chowk, Biratnagar	_____ **External Examiner**

ACKNOWLEDGEMENT

The successful completion of this project would not have been possible without the guidance and support of many individuals. We would like to express our deepest gratitude to our supervisor, Kushal Niroula, for his invaluable guidance, continuous support, and encouragement throughout the development of this project. His expertise and insightful feedback have been instrumental in shaping the project to its current form.

We also extend our sincere thanks to Er. Sumit Babu Shah, Head of the IT Department, for providing us with the necessary academic environment and resources that facilitated the successful completion of our project.

Lastly, we are grateful to our peers, friends, and family members for their constant support and motivation. This project has been a significant milestone in our academic journey, and we hope to apply the knowledge and experience gained to future achievements. We look forward to further learning and collaboration in the years to come.

ABSTRACT

Alnote is a web-based Study Notes Archival and Distribution System that provides a platform for students to upload, share, and access study materials in PDF format. Users can browse notes based on their academic needs, with Normal Users required to watch ads for access, while Premium Users enjoy direct search capabilities.

The platform ensures secure authentication and authorization, with an Admin and Moderator overseeing content moderation, user verification, and subscription management. Users can also review and rate uploaded notes, contributing to aquality-driven learning environment.

Alnote is developed using React, Tailwind CSS, and JavaScript for the front end, ASP.NET Core Web API for the back end, and PostgreSQL for database management. The system aims to enhance accessibility and organization in education by providing a structured, user-friendly,and efficient way to manage study materials.

Table of Contents

Title **Page No.**

List of Abbreviations

AbbreviationsFull Form	
UI	User Interface
VS	Visual Studio

List of Table

Table No. **Title** **Page No.**

List of Figures

Figure No. **Tittle** **Page No.**

CHAPTER 1: INTRODUCTION

1.1 Introduction

In today's digital age, students and educators require quick and efficient access to study materials. Traditional methods of note-sharing are often unorganized, time-consuming, and lack accessibility. The proposed project, Alnote, is A Study Notes Archival and DistributionSystem designed to provide students with a seamless way to upload, share, and access study materials in PDF format.

The system eliminates the inefficiencies of manual note-sharing by offering a structured, user-friendly web platform where students can browse and download educational resources. Unlike traditional methods, Alnote ensures an organized and secure environment, reducing dependency on physical copies and scattered online resources. With subscription-basedaccess and ad-supported free viewing, the platform is tailored to provide an optimal experience for all users.

The system consists of three user roles:

- Admin and Moderator – Manages uploaded content, verifies accounts, and monitors platform activities.
- Normal Users – Can upload and view PDFs but must watch ads to access content.
- Premium Users – Have additional features such as direct search and the ability to download notes.

By streamlining the process of sharing academic materials, Alnote enhances accessibility and efficiency for students worldwide.

1.2 Problem Statement

The development of an online study notes distribution system aims to address the challenges students face with traditional methods of accessing and sharing educational materials. The key issues include:

i. Unorganized note-sharing – Students often struggle to find relevant study materials due to scattered resources.

ii. Limited accessibility – Physical notes or materials stored in private folders limit access for other learners.

iii. Time-consuming process – Searching for the right study materials often requires navigating multiple sources.

iv. Lack of monetization options for content creators – Students who create high-quality notes do not have a structured way to benefit from their contributions.

The Alnote platform resolves these issues by providing an efficient, structured, and user-friendly system for students to upload, share, and access study materials.

1.3 Objectives

The primary objectives of Alnote are:

i. To provide a platform for students to share and access verified study materials.

ii. To offer a user-friendly experience with subscription and ad support.

1.4 Scope and Limitations

1.4.1 Scope

The Alnote system focuses on the efficient sharing of academic materials in PDF format. The platform provides:

- A structured note-sharing system with categorized subjects and topics.
- Subscription-based access for premium users.
- Admin moderation to prevent inappropriate content.
- Monetization for note creators via a structured buying/selling system.

1.4.2 Limitations

- The system does not provide live tutoring or interactive Q&A features.
- It does not manage handwritten notes or physical copies—only PDF uploads.
- Payment verification for subscriptions is done manually by the admin.
- Content moderation is dependent on admin reviews and user feedback.

1.5 Report Organization

This report provides a detailed explanation of the Alnote system. It includes:

- System architecture and design diagrams for a clear understanding of the platform's

structure.

- Details on the tools and technologies used, including React, ASP.NET Core, PostgreSQL, and Tailwind CSS.
- Implementation details, feasibility studies, and testing results to ensure system functionality.

This document serves as a comprehensive guide to understanding the development, implementation, and benefits of the Alnote Study Notes Archival and Distribution System.

1.6 Report Organization

The report consists of five chapters which will cover the designing and development application.

I. **Chapter One:** This chapter introduces the system and the problems, and gives an overview of the study.

II. **Chapter Two:** This chapter covers the literature review which is the previous related work that has been done before.

III. **Chapter Three:** This chapter explains the selected methodology that we are going for in this project. Planning through the methodology.

IV. **Chapter Four:** This chapter discusses the implementation and testing.

V. **Chapter Five:** This chapter discusses the conclusion, recommendations, and future works to improve this study.

CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW

2.1 Background Study

With the rapid advancement of digital technology, students and educators require more efficient methods to share and access academic materials. Traditional methods of distributing study notes—such as physical copies, scattered online resources, and social media groups—often lead to disorganization, limited accessibility, and difficulty in retrieving relevant materials.

The Alnote Study Notes Archival and Distribution System is designed to address these challenges by providing a structured online platform where students can upload, browse, and access study materials conveniently. By integrating a subscription-based model and an ad-supported free access system, Alnote ensures that both free and premium users can benefit from a seamless learning experience.

The key features of the system include:

- Easy upload and retrieval of PDF-based notes**.**
- Categorized study materials for structured browsing.
- Premium subscription model for enhanced accessibility, including direct search and download options.
- Review and rating system to maintain content quality.
- Admin moderation to ensure content security and credibility.

By shifting the focus to a digital platform, Alnote eliminates common inefficiencies, making it easier for students to find reliable study resources and for note creators to monetize their efforts.

2.2 Literature Review

i) Digital Learning Platforms for Efficient Knowledge Sharing

Author: John F. McMillan (2021)

According to McMillan [1], the evolution of digital learning platforms has significantly enhanced the accessibility of educational resources. Many universities and online learning communities have integrated cloud-based repositories where students can share, access, and review study materials. However, existing platforms often lack a structured approach to

categorize and verify uploaded content. Alnote addresses this issue by implementing categorized notes, an admin-verification system, and a rating system, ensuring high-quality study materials.

ii) A Framework for Subscription-Based Educational Resource Platforms

Author: Emily R. Stevens (2022)

Stevens [2] discusses the growing adoption of subscription-based educational platforms, which provide ad-free access, direct search functionality, and exclusive content. Many digital education services, such as Coursera, Udemy, and Khan Academy, utilize a similar model. Alnote adopts this approach, offering premium users enhanced access and ad-freeexperiences while maintaining free access with advertisements for normal users.

iii) User Engagement and Content Monetization in Online Learning

Author: Ahmed K. Alwani (2020)

Alwani [3] explores the monetization of user-generated content in educational platforms. He states that students who create high-quality notes often struggle to benefit financially from their contributions. Alnote resolves this by allowing users to sell notes via a structuredsubscriptionmodel, creating opportunities for peer-to-peer knowledge sharing and financial incentives for contributors.

By analyzing these studies, Alnote builds upon proven digital learning methodologies, integrating a structured subscription model, efficient content verification, and an organizednote-sharing system to enhance the academic experience for students worldwide.

CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

3.1 System Analysis

3.1.1Development Methodology

For the development methodology, we have selected, agile developmental methodology. Agile methodology provides several key advantages in this project. Some of those advantages are:

- The project can be developed and delivered in small increments, such as user registration, profile management and so on.
- Agile methodology allows for testing and feedback at each stage of development.
- The methodology manages evolving requirements as feedback is received from stakeholders or testers.
- Frequent feedback loops from stakeholders or users ensure the app meets user expectations.
- Agile methodology focuses on delivering high-priority features first, ensuring efficient resource utilization.

Figure 3.1: Agile Methodology

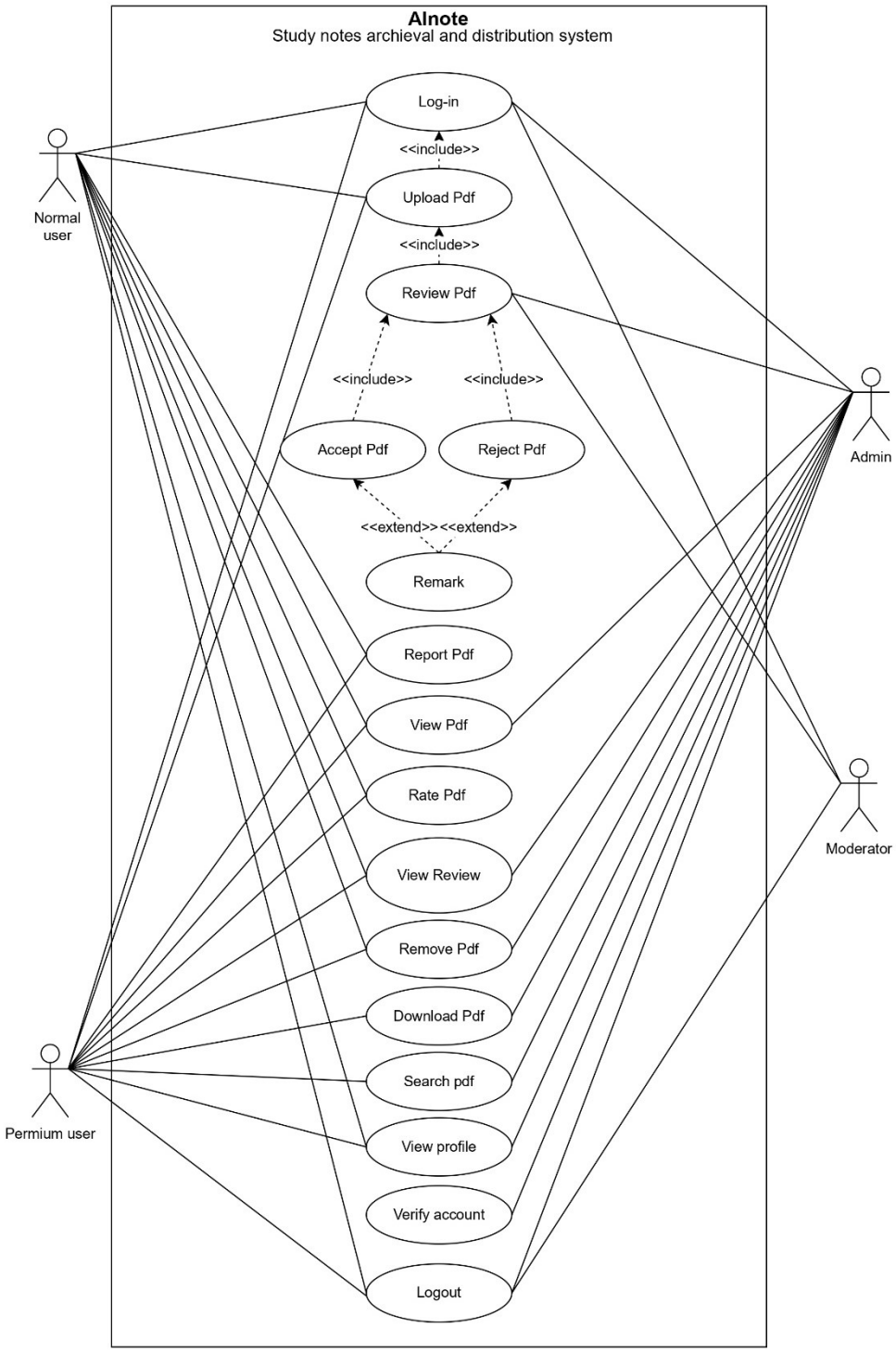


Figure 3.2: Use Case Diagram

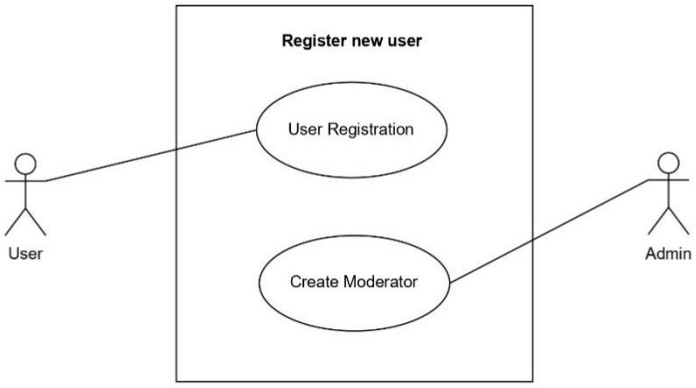


Figure 3.2.1: Use Case Diagram

Actors

- Normal User: Basic user with access to standard features.
- Premium User: Enhanced privileges (e.g., higher upload limits, exclusive features).
- Admin: Manages system-level actions (e.g., removing PDFs, user accounts).
- Moderator: Handles account verification and authentication processes.

Key Use Cases

Authentication & Account Management

- Sign-up: New users register. Likely *includes* **Verify Account** (to confirm credentials).
- Log-in: Authentication for existing users.
- Verify Account: Managed by the Moderator actor.
- Logout: Ends the session.

PDF Management

- Upload PDF: Users submit study notes. Requires prior authentication (<<include>> relationship with Log-in).
- Download/View/Search PDF: Basic actions for accessing notes.
- Remove PDF: Likely restricted to *Super Admin*.

Review & Moderation

- Review PDF: Base use case for evaluating submissions.
- Accept PDF: <<include>> relationship with Review PDF (approval).
- Reject PDF: <<extend>> relationship with Review PDF (conditional rejection, e.g., adding a Remark).
- Report PDF: Flag inappropriate content.

User Interaction

- Rate PDF: Users provide feedback.
- View Profile: Check user details.
- View Review: Access feedback on uploaded notes.

Relationships

1. <<include>>:
   - Prerequisite actions. For example:
     - Upload PDF requires Log-in.
     - Accept PDF requires Review PDF.
2. <<extend>>:
   - Conditional extensions. For example:
     - Reject PDF extends Review PDF when a submission is declined (may trigger Remark).

Actor-Use Case Associations

- Actor - Associated Use Cases Normal
- User - Sign-up, Log-in, Upload/View/Search/Download PDF, Rate, Report
- Premium User - All Normal User features + potential advanced privileges
- Admin Remove PDF, Review/Accept/Reject PDF, Manage users
- Authenticator Verify Account, Log-in authentication

Non-Functional Requirements

These requirements define the criteria that the Alnote Study Notes Archival and DistributionSystem must meet for optimal functionality.

Performance:

- The system efficiently handles multiple users simultaneously.
- Quick response time for searching, uploading, and downloading notes.
- Optimized database queries to ensure fast retrieval of study materials.

Security:

- Users must authenticate using email and password before accessing premium features.
- Admin moderation ensures content integrity and prevents unauthorized uploads.
- Premium users' payment information is securely processed and verified by the Admin.

Usability:

- A clean and simple UI ensures easy navigation for users.
- The platform is mobile-responsive, making it accessible on multiple devices.
- Categorized notes and a structured interface improve user experience and ease of access.

3.1.2 Feasibility Study

3.1.2.1 Technical Feasibility:

The Alnote system is a fully web-based platform. The technologies used include:

- **Frontend:** React, Tailwind CSS, JavaScript
- **Backend:** Web API with ASP.NET Core
- **Database:** PostgreSQL
- **Development Environment:** Visual Studio Code

Each of these technologies is open-source or widely used, making it technically feasible. The required technical skills are manageable, and the system's architecture is scalable to accommodate more users in the future.

3.1.2.2 Operational Feasibility:

- Alnote simplifies the study material-sharing process by providing a centralized and well-organized platform.
- The system significantly reduces the effort needed to search for study materials.

- The subscription model ensures a sustainable user experience, making the system operationally feasible.

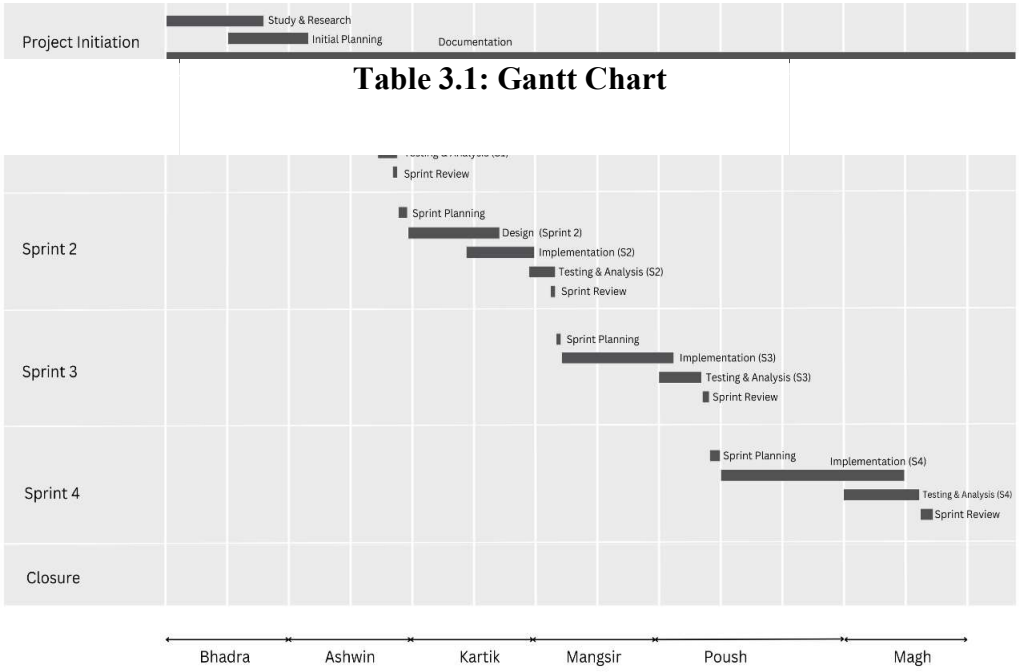
3.1.2.3 Economic Feasibility:

- Alnote is a cost-effective platform with minimal developmentcosts since most of its tools are open-source.
- The subscription-based revenue model ensures financial sustainability.
- Ad-based revenue for free users supports platform maintenance, making it economically feasible.

3.1.3Schedule Feasibility

Schedule feasibility involves creating a proper timeline for the project's initial phase i.e., from study and research to the ending phases. A project timeline will be established, outlining each phase's start and end dates.

Table 3.1: Gantt Chart



1.1.4 Class Diagram

A class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

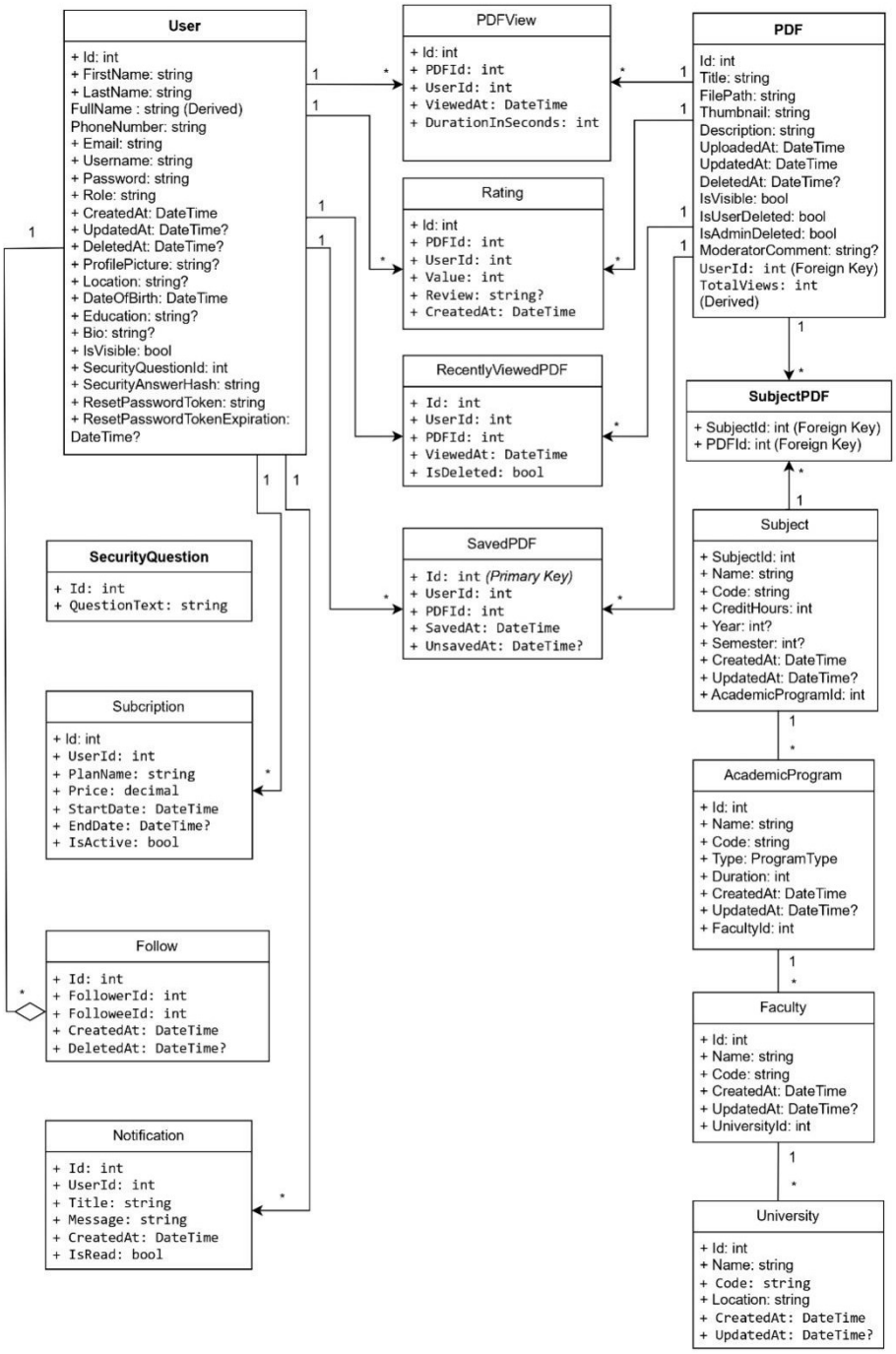


Figure 3.3: Class Diagram

3.1.4Sequence Diagram

Sequence diagrams in UML specifically focus on *lifelines*, or the processes and objects that

live simultaneously, and the messages exchanged between them to perform a function before the lifeline ends.

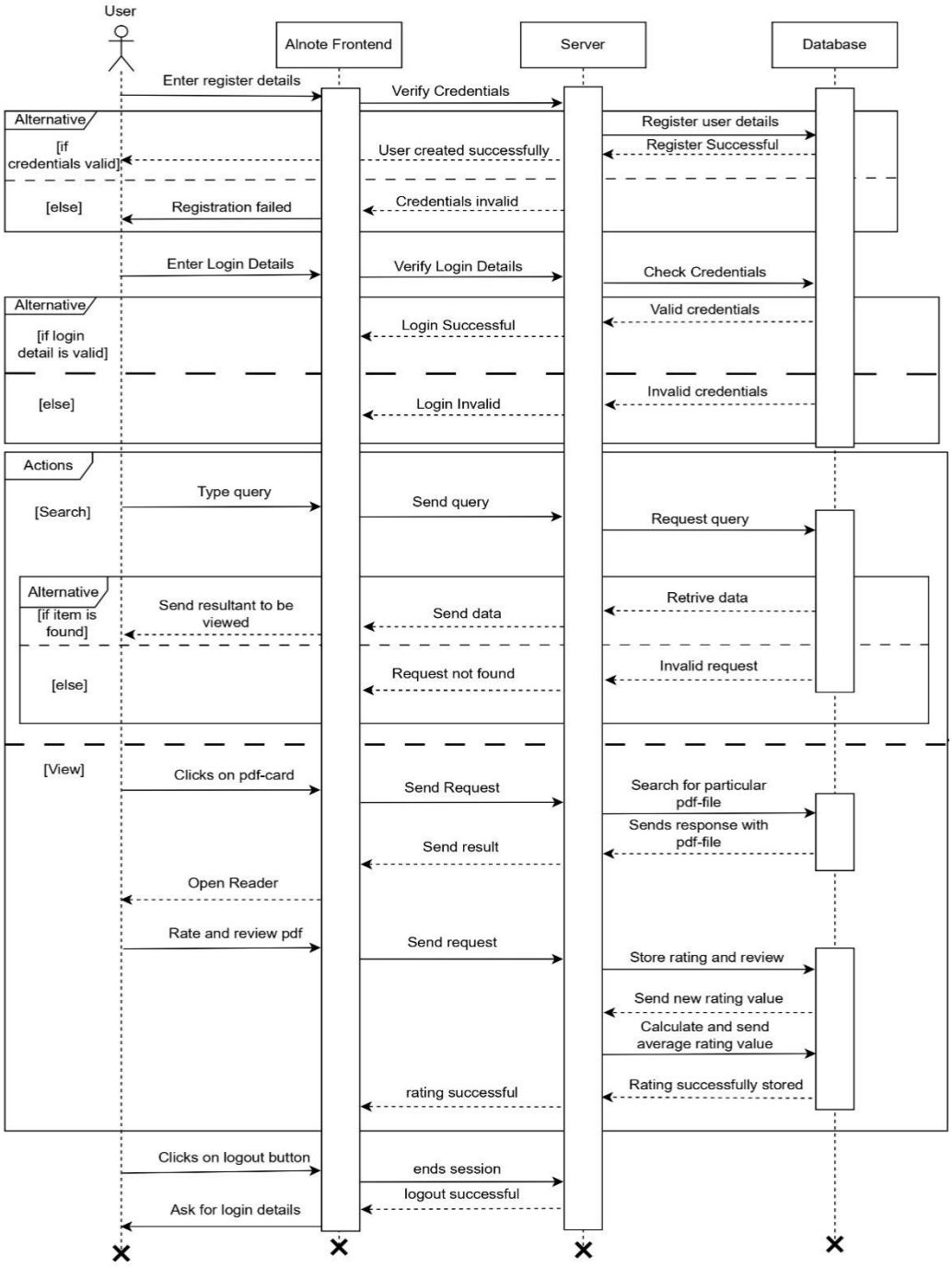


Figure 3.4: Sequences Diagram

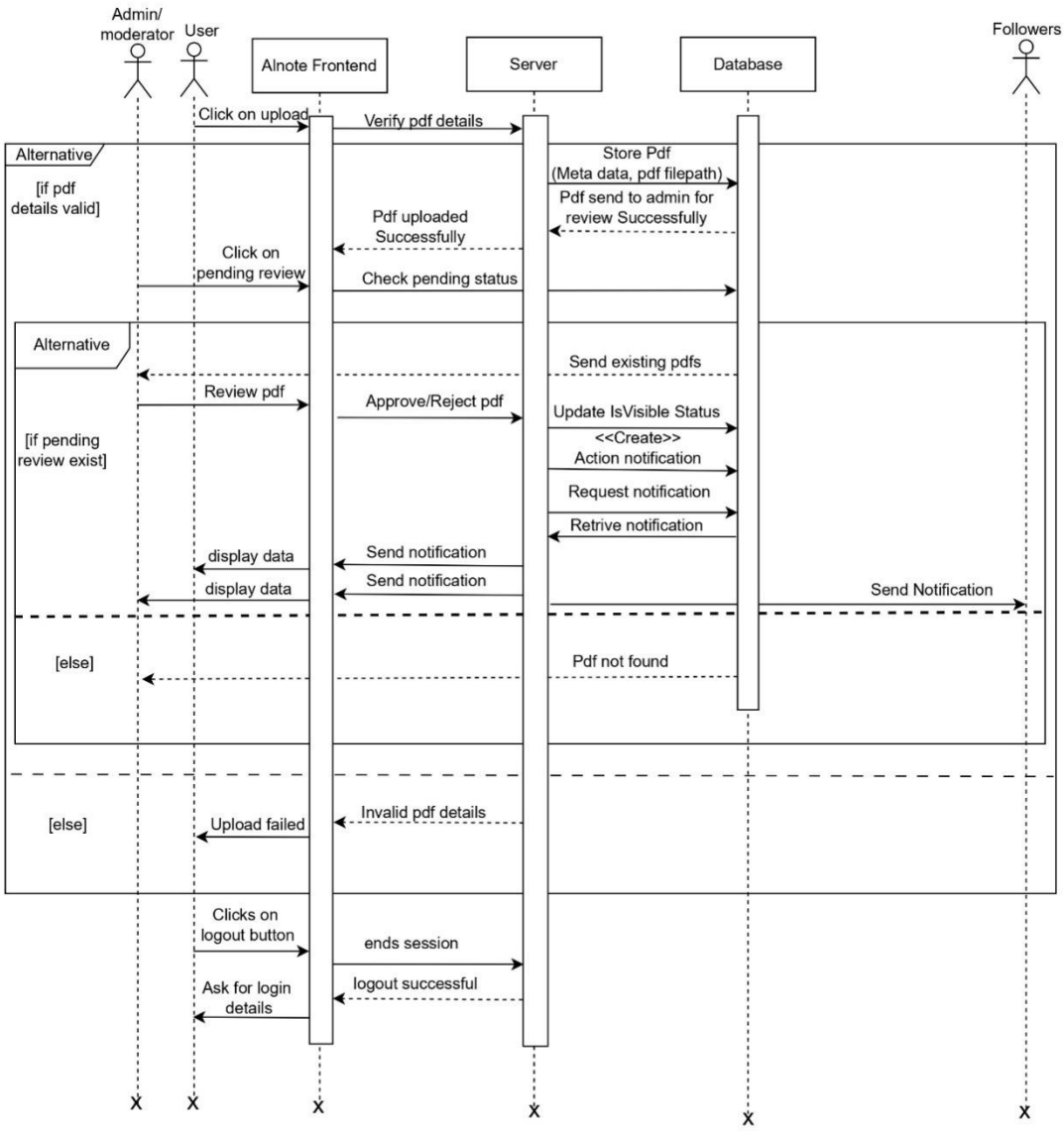


Figure 3.4.1: Sequences Diagram

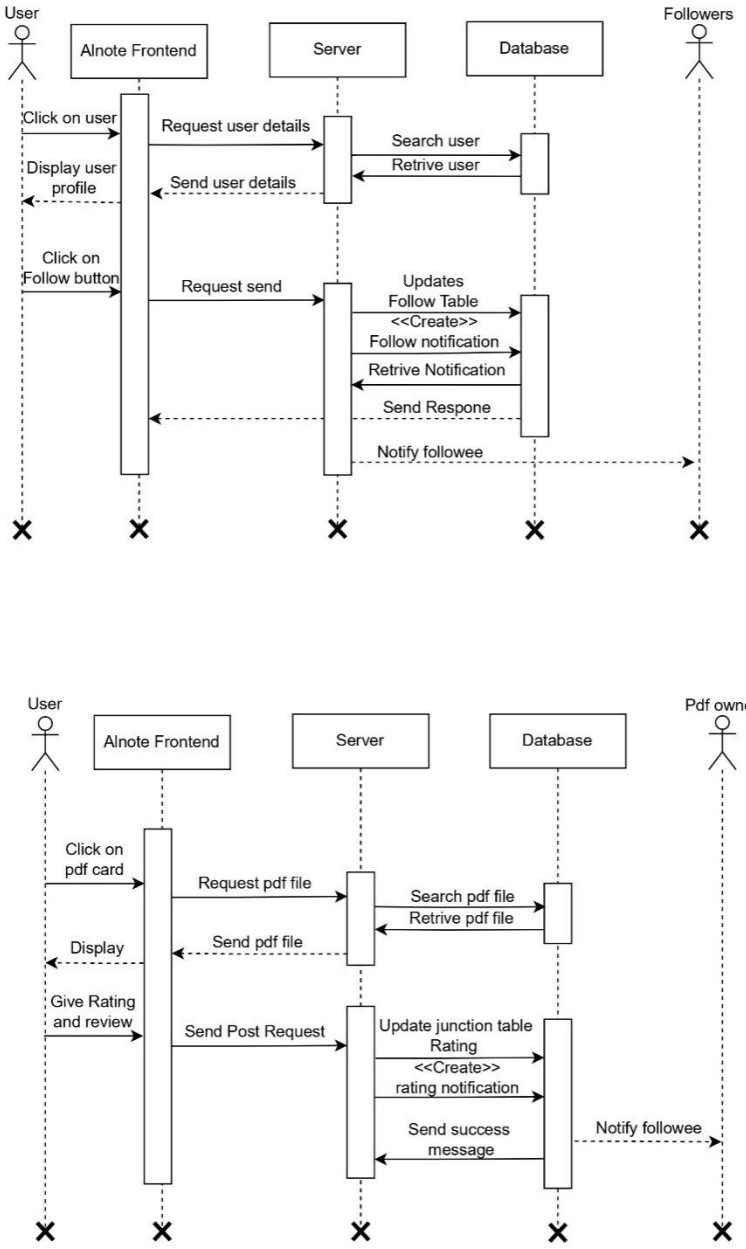


Figure 3.4.2: Sequences Diagram

3.1.5 Flowchart Diagram

A flowchart is a diagram that visually represents a process or system, illustrating the sequence of steps and decisions. It uses different shapes to signify various elements: ovals for start and end points, rectangles for actions or processes, diamonds for decision points (like yes/no questions), and parallelograms for input/output operations. Flowcharts are useful tools for simplifying complex processes, making them easier to understand, analyze, and improve by clearly showing how each step flows into the next.

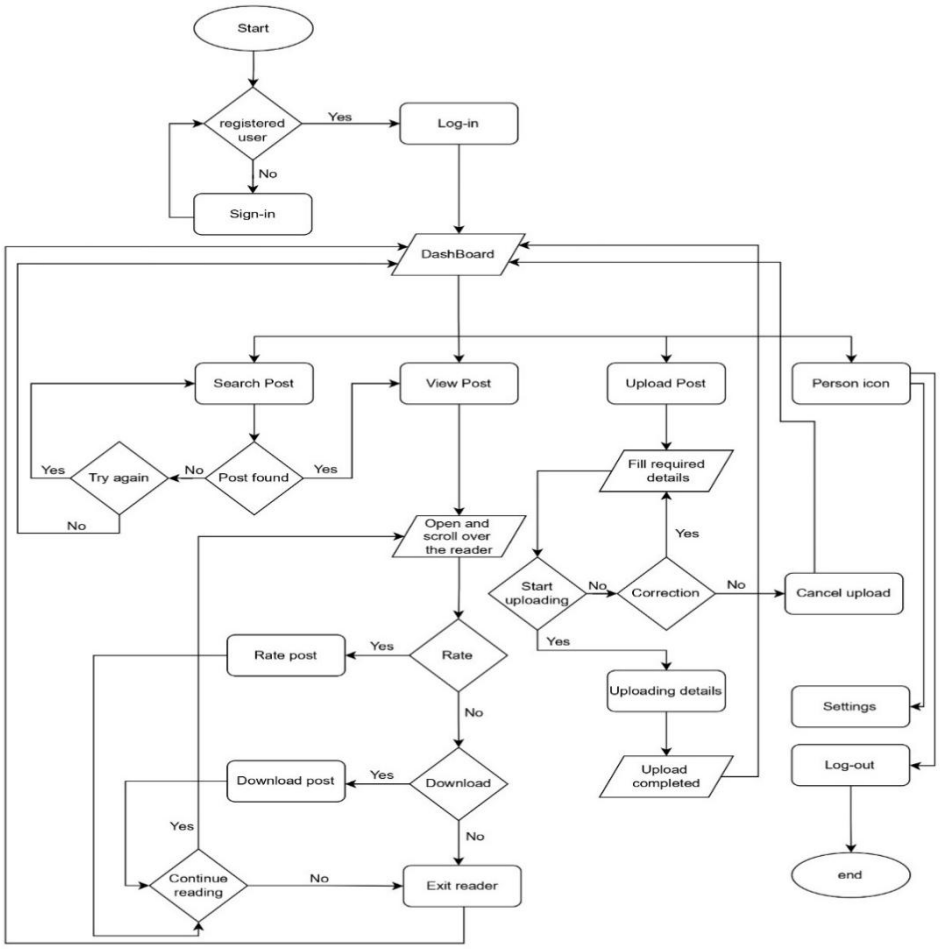


Figure 3.5: Flowchart Diagram

3.1.6 Activity Diagram

An Activity Diagram is a UML diagram that represents the flow of activities or tasks in a process. It captures the sequence of steps, decisions, and parallelism in a process. Activity diagrams are ideal for process modeling because they focus on control flow and workflow logic.

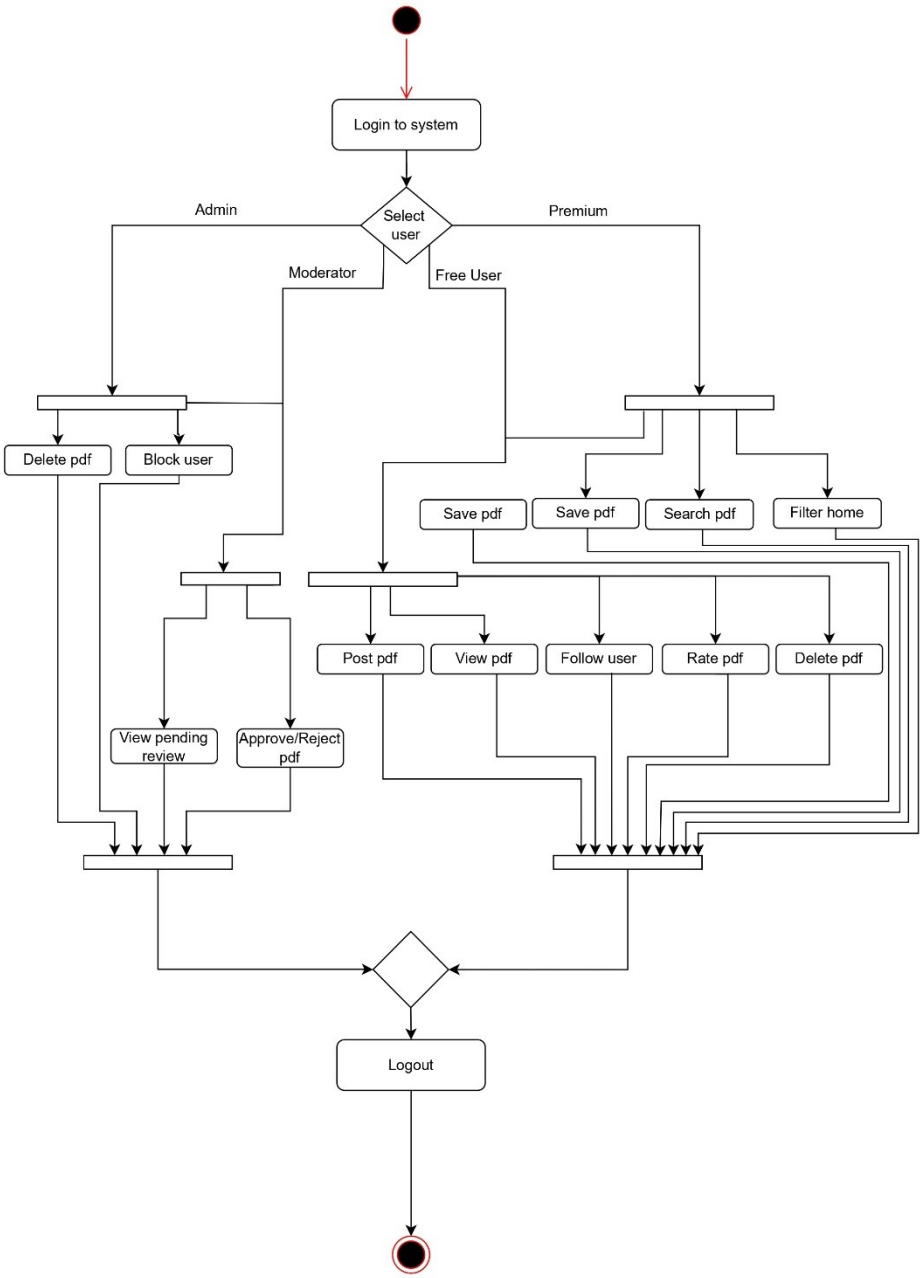


Figure 3.6: Activity Diagram

CHAPTER 4: SYSTEM DESIGN

4.1 Design

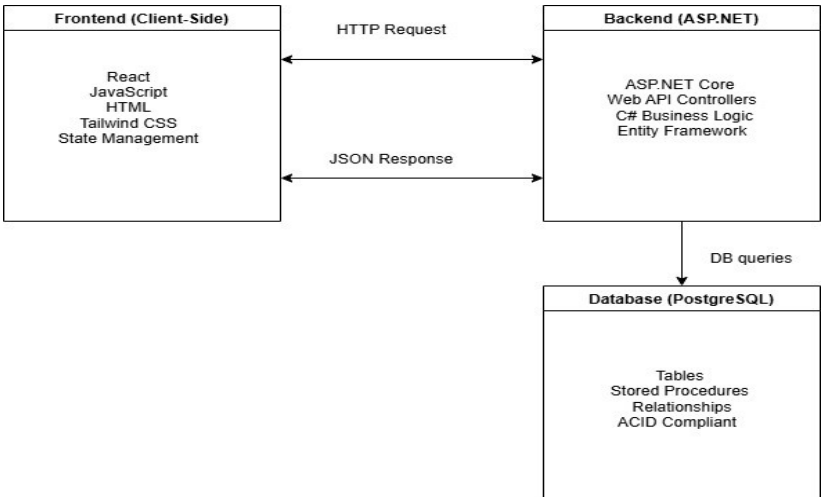


Figure 4.1: Headless Architecture

The design of the Alnote project is structured into three primary tiers: Frontend (Client-Side), Backend (ASP.NET), and Database (PostgreSQL). The Frontend is constructed with HTML, Tailwind CSS, and JavaScript, which is responsible for the user interface and interactive parts. It interacts with the Backend via HTTP requests and gets responses in JSON. The Backend, constructed with ASP.NET Core, is responsible for the business logic and request processing via Web API Controllers. The program employs C# to implement fundamental functions such as user authentication and note management and employs Entity Framework for database manipulation. The database, which is PostgreSQL-supported, stores data in tables, employs stored procedures for pre-defined special tasks, and establishes relationships among various tables. It adheres to ACID principles, hence guaranteeing data reliability and integrity. This structural design allows for an effortless flow of information among the user interface, server-side calculation, and database, thereby creating a stable and secure backbone of the Alnote system.

4.1.1 Deployment Diagram

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them. It is typically used to visualize the physical hardware and software of a system. It helps to understand how the system will be physically deployed on

the hardware.

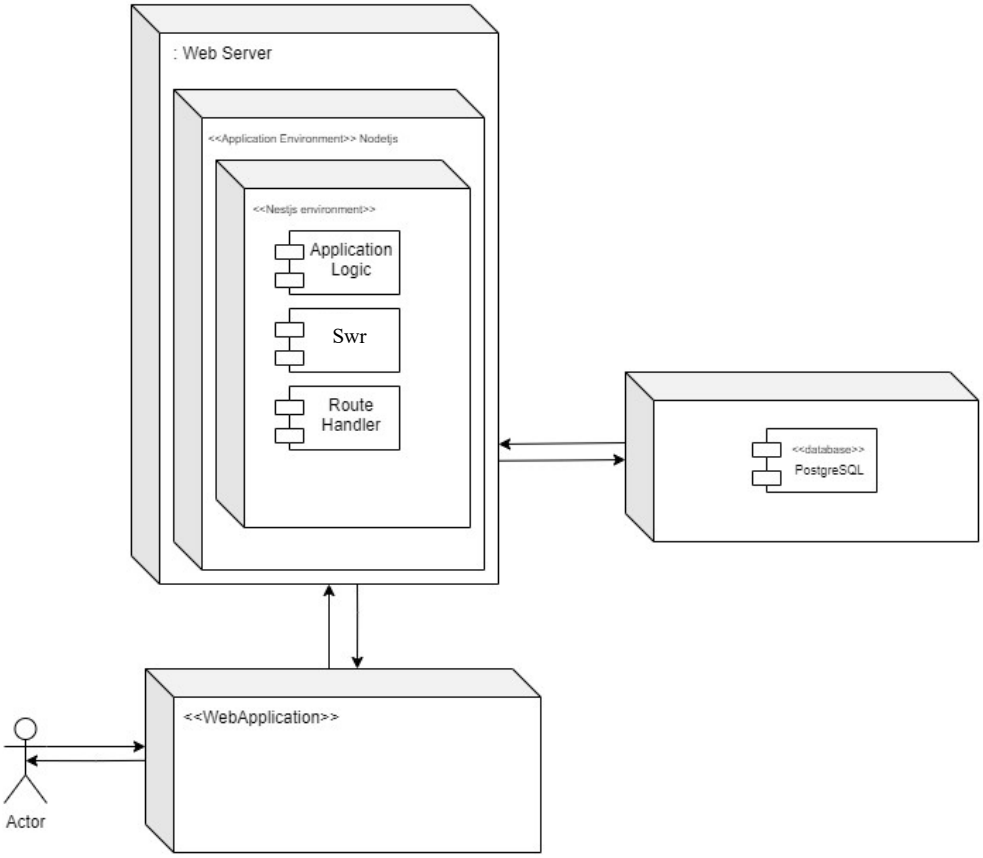


Figure 4.2: Deployment Diagram

4.2 Algorithm Details

The **Substring Search Algorithm** used in the system follows a Brute Force approach to check whether a given search query exists within a dataset of text records. This algorithm is straightforward but inefficient for large datasets, as it checks for substring matches sequentially.

The process involves the following key steps:

- Token Matching – Each record is checked to see if it contains the search term.
- Sequential Search – The algorithm scans each text entry from start to end, looking for a match.

- Filtering and Ranking – Records containing the search term are retrieved.

The algorithm ensures that users receive results where the search term is present, maintaining simplicity and usability.

Workflow of the Algorithm

1. Fetch User Query – Retrieve the search string input by the user.
2. Retrieve Data – Fetch all relevant records from the database.
3. Perform Substring Search:
    - Scan each record and check if the search term exists within it.
    - If found, mark the record as a match.
4. Sort Results – Order results based on their natural order in the dataset.
5. Return Results – Display matching records to the user.

Substring Matching Formula

The substring search algorithm follows this formula:

$$\mathbf{M(T,P)} = \begin{cases} \mathbf{1}, & \boldsymbol{if pexistsinT} \\ \mathbf{0}, & \boldsymbol{otherwise} \end{cases}$$

Where:

- **T** = The text (record) being searched.
- **P** = The pattern (user query).
- **M(T, P)** = 1 if the pattern exists in the text, otherwise 0.

Brute Force Substring Search Steps

For a text **T** of length **n** and a pattern **P** of length **m**, the brute-force approach works as follows:

1. Start at the first character of **T**.
2. Compare it with the first character of **P**.

3. If they match, compare the next character in **T** with the next character in **P**.
4. Continue until:
    - All characters of P match → Pattern found.
    - A mismatch occurs → Move to the next character in **T** and restart the comparison.
5. Repeat the process until the end of **T**.

The worst-case time complexity of this algorithm is O(n * m), meaning it performs poorly for large datasets.

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1 Implementation

Software implementation is the process of integrating application into organization workflow structure. Implementation is very important because it provide access to businesses towards latest technology.

5.1.1 Tools used:

1.PostgreSQL: PostgreSQL is an object-relational database management system (ORDBMS) that is known for its robustness, scalability, and SQL compliance. It supports advanced data types like JSON, XML, and arrays, along with ACID (Atomicity, Consistency, Isolation, Durability) compliance, making it reliable for handling complex transactions and large datasets.

2. GIT for version control: Git allows multiple developers to work on the same codebase simultaneously, even if they are not connected to a central server. This means that developers can work on their own local copies of the code, making changes and committing them to their local repository. These changes can then be merged into the main repository when they are ready.

3. NPM packages: Annpm package is a collection of code, resources, and metadata that can be installed using the NPM. These packages can contain any kind of code or resource that can be used by NodeJS applications or other npm packages. It is typically hosted on the npm registry, a public repository where developers can publish and share their packages with others. Developers can also create and host their own private registries to share packages within their organization or with selected users.

4. Draw.io: Draw.io is the web-based diagramming tools that allow users to create a wide range of diagrams, including flowcharts, network diagrams, mind maps, and more. While this tools have similar features and capabilities, there are some differences between them.

5. Swagger: Swagger is a widely adopted framework for designing, building, and documenting APIs. It allows developers to easily define API endpoints, perform testing, and generate documentation, ensuring seamless integration and interaction with APIs. With its user-friendly interface, Swagger helps developers to send HTTP requests, view responses,

and automate testing, making it an essential tool for API development and testing.

6. ASP.NET:open-source web framework developed by Microsoft for building dynamic web applications and APIs. It allows developers to create scalable, high-performance web solutions using .NET languages like C#. ASP.NET simplifies the development process by providing built-in tools for routing, authentication, data access, and more, enabling rapid development of robust web applications. Whether you're building simple websites or complex enterprise-level solutions, ASP.NET provides flexibility and efficiency for building modern, secure, and high-performance web applications.

5.2 Testing

Testing strategies are the approach for testing the system or application. It is the process to get some information about the key issues of the developed system and verify the application functions correctly. By running tests against our application consistently, we can verify our application's correctness, functional behavior, and usability before we release it publicly. Testing is the process of running a system with the intention of finding errors. Testing can be done using varieties of levels. Some of the levels of testing we carried out are:

5.2.1 Unit Testing

A unit is the smallest testable part of the software. It usually has one or a few inputs and usually a single output. Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently checked for proper operation. Unit testing increases confidence in changing/maintaining code. We have tested the module in an attempt to discover any error in the code.

Table 5.1: Registration Testing

Test Case ID	**S.N.**	**Case Name**	Registration
TC_01	**1.**	**Purpose of Testing**	Register users
TC_02	**2.**	**Test Attribute**	Filling out the required information asked during registration.

TC_03	3.	**Test Unit**	**Test Case**	**Test Result**
	4.	Register Button	Invalid format or missing values	Display appropriate message
TC_05	5.	Register Button	Valid Format and no missing values	Redirect to the Login Page and store the details in the database

Table 5.2: Login Testing

Test Case ID	**S.N.**	**Case Name**	Login	
TC_01	1.	**Purpose of Testing**	Login users	
TC_02	2.	**Test Attribute**	Filling out username and password.	
TC_03	3.	**Test Unit**	**Test Case**	**Test Result**
	4.	Login Button	Incorrect username and password	Display login unsuccessful message
TC_05	5.	Login Button	Correct username and password	Redirect to the Homepage

Table 5.2: View and Edit User's Details Testing

Test Case ID	**S.N.**	**Case Name**	View and Edit User's Details	
TC_01	**1.**	**Purpose of Testing**	Retrieval and Update of the database	
TC_02	**2.**	**Test Attribute**	Click on the profile and edit button	
TC_03	**3.**	**Test Unit**	**Test Case**	**Test Result**
	4.	Profile Display	Click on the profile	Display the detail of the user
TC_05	**5.**	Edit Button	Change the bio, location, education and save.	Display the save changes message and update the database.

Table 5.3: Search and Algorithm Testing

Test Case ID	**Case Name**	Search and Algorithm	
TC_01	**Purpose of Testing**	Retrieval and Update of the database	
TC_02	**Test Attribute**	Click on the profile and edit button	
TC_03	**Test Unit**	**Test Case**	**Test Result**
	Search	Type your suitable pdf	Display the list of pdf and allow them to choose one

4			
TC_05	Algorithm	Select the pdf required	Display the pdf based on the search and rating.

5.2.2 System Testing

System Testing is testing on a complete, integrated system to evaluate the system's compliance with its specific requirements. This testing is done to ensure that the system meets the requirements. System testing involves testing the software code for the following:

- Testing the fully integrated applications in order to check how components interact with one another and with the system as a whole.
- Verifying through testing every input in the application to check for desired outputs.
- Testing of the user's experience with the application.

Table 5.4: System Testing

Test Strategy	System Test			
Test Module	Complete application test includes registration, login, profile view and edit, pdf search, pdf recommendation based on search and rating.			
Test Case ID	**Conditional Test**	**Expected Outcome**	**Actual Outcome**	**Remarks**
TC_01	Register as a user	The details provided are stored in the database.	As per the expected result.	Pass
TC_02	Log in as a user	Redirect to the homepage of the application	As per the expected result	Pass

TC_03	Search for a pdf or select the pdf posted by user	Display a list of pdfs on the basis of the required pdf.	As per the expected result	Pass
TC_04	Log in as Admin	Redirect to the homepage of Admin.	As per the expected result	Pass
TC_05	Pdf upload request accept/ reject	Can accept or reject the request of pdf upload.	As per the expected result	Pass
TC_06	Ad management	Ad upload and display	As per the expected result	Pass
TC_07	User management	Admin can block the user.	As per the expected result	Pass
TC_08	Academic structure management	Admin can add, edit and delete academic structure	As per the expected result	Pass
TC_09	Log in as Moderator	Redirect to the homepage of Moderator.	As per the expected result	Pass
TC_10	Pdf upload request accept/ reject	Can accept or reject the request of pdf upload.	As per the expected result	Pass
TC_11	Ad management	Ad upload and display	As per the expected result	Pass

CHAPTER 6 : CONCLUSION AND RECOMMENDATION

6.1 Conclusion

The users that are involved in our application tremendously benefit from our system. Our proposed application is an effective, dependable, and easy-to-use. Accuracy, efficiency, usability, security, and scalability were all taken into consideration when developing the system. Our application provides a platform which enables students to **upload, share, and access** study materials in **PDF format** Also, the system is web-based, thus users can access the web application's content at any time, day or night.

6.2 Future Recommendation

The project can be further developed and improved by working in the following mentioned areas:

1.Payment integrations.

2.Improving data security by implementing two-factor authentication.

3.Enhancing the user interface to provide better user-experience.

4.Introducing better mechanisms to verify notes.

5.Real time chat.

6. To generate revenue using subscription based and ad model.

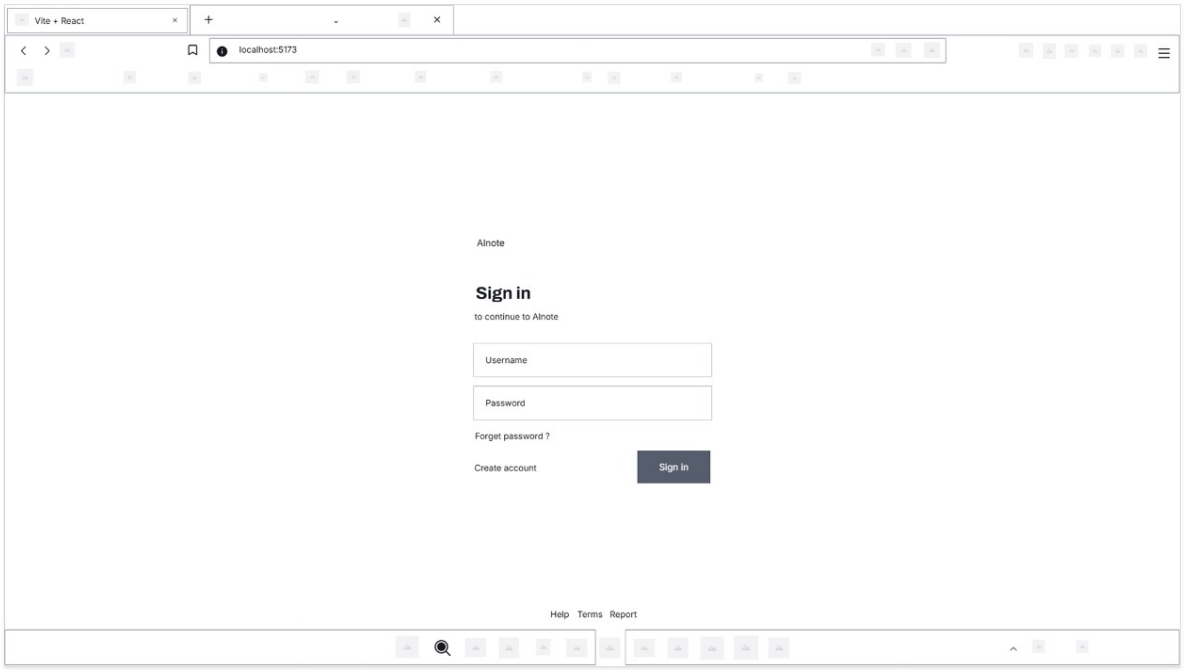
References

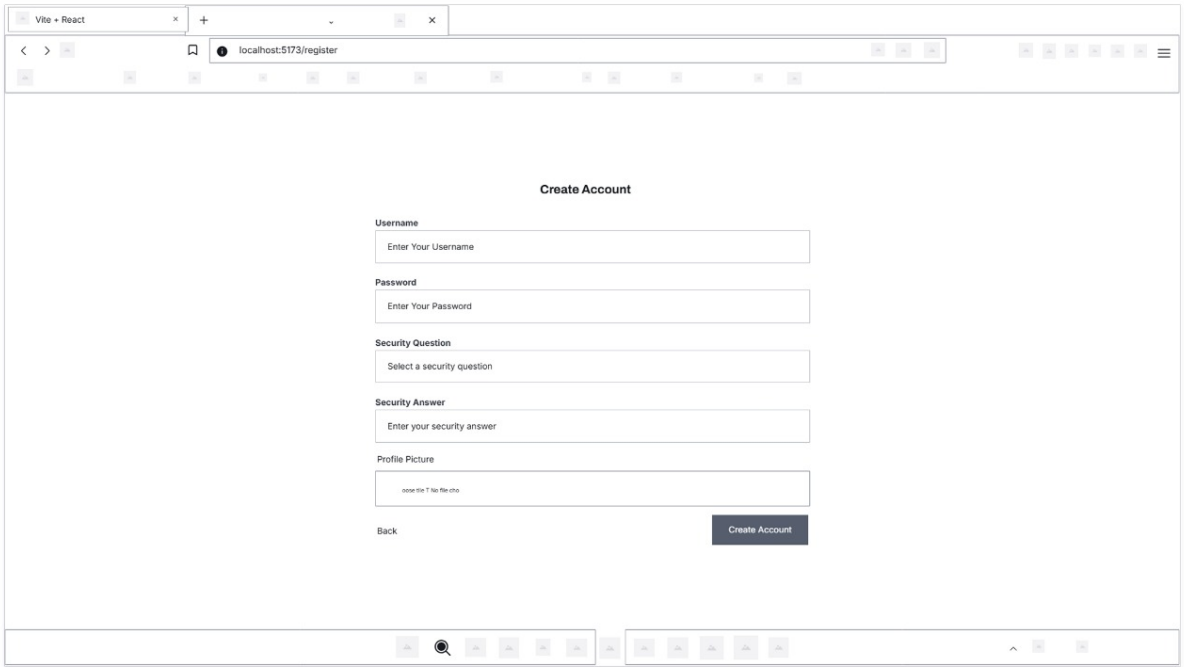
[1] J. F. McMillan, "Digital Learning Platforms for Efficient Knowledge Sharing," in *Proceedings of the International Conference on Educational Technology*, 2021.

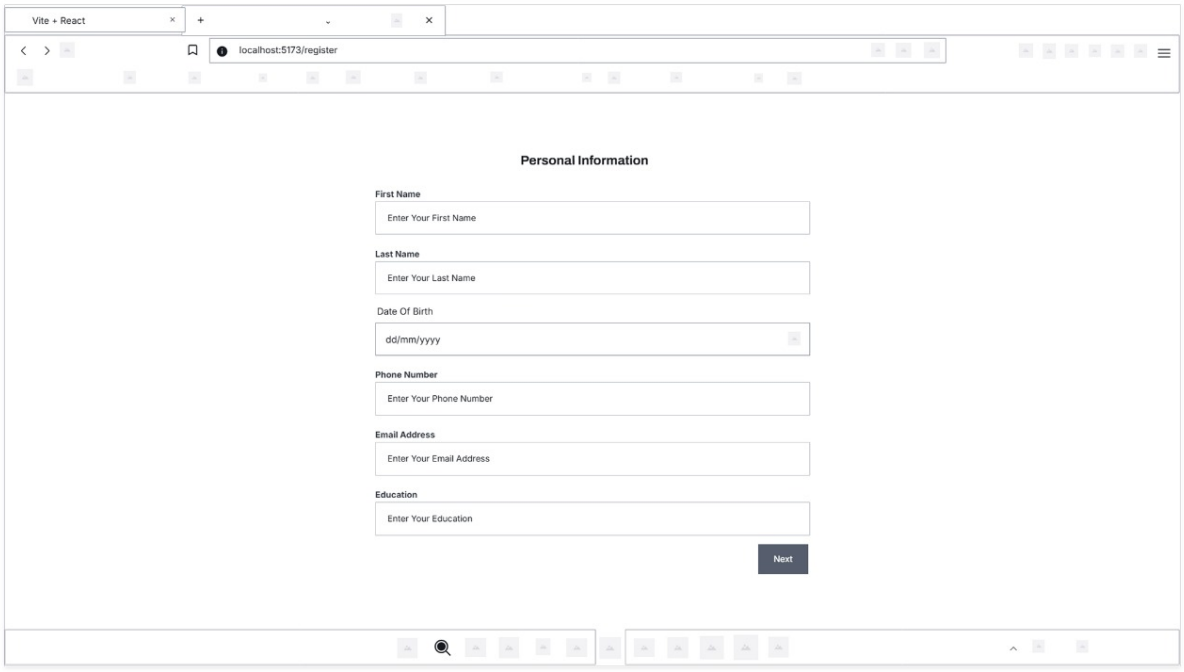
[2] E. R. Stevens, "A Framework for Subscription-Based Educational Resource Platforms," in *Journal of Digital Learning and Educational Innovation*, vol. 15, no. 3, pp. 45-58, 2022.

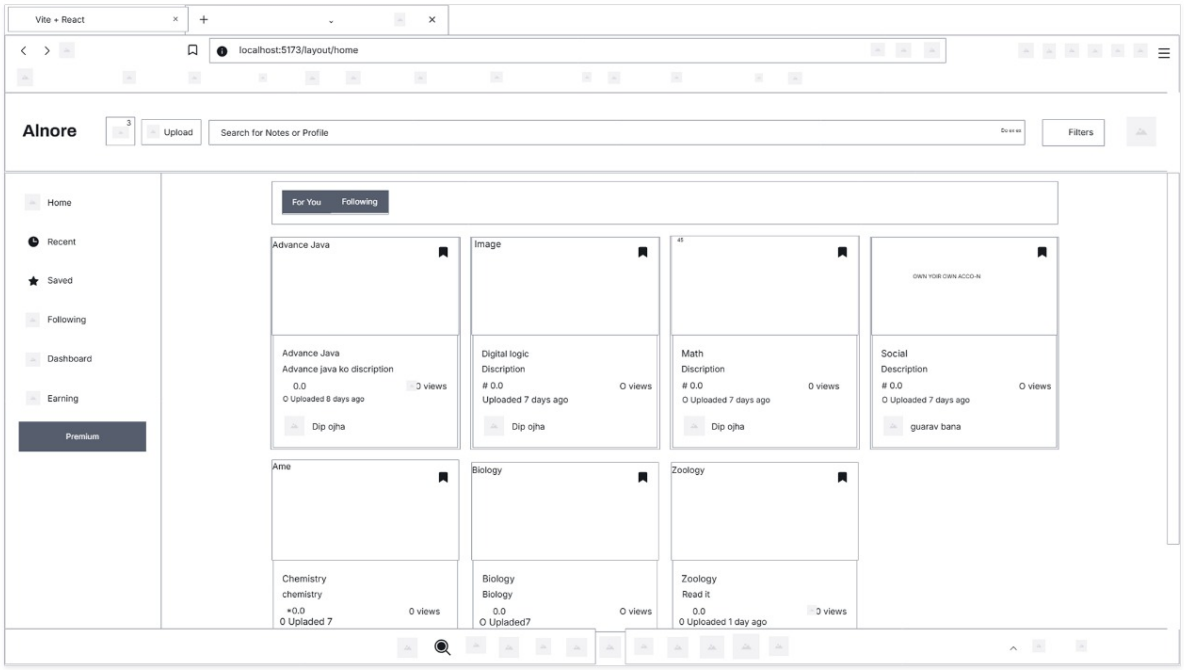
[3] A. K. Alwani, "User Engagement and Content Monetization in Online Learning," in *International Conference on Online Learning and Content Monetization*, 2020.

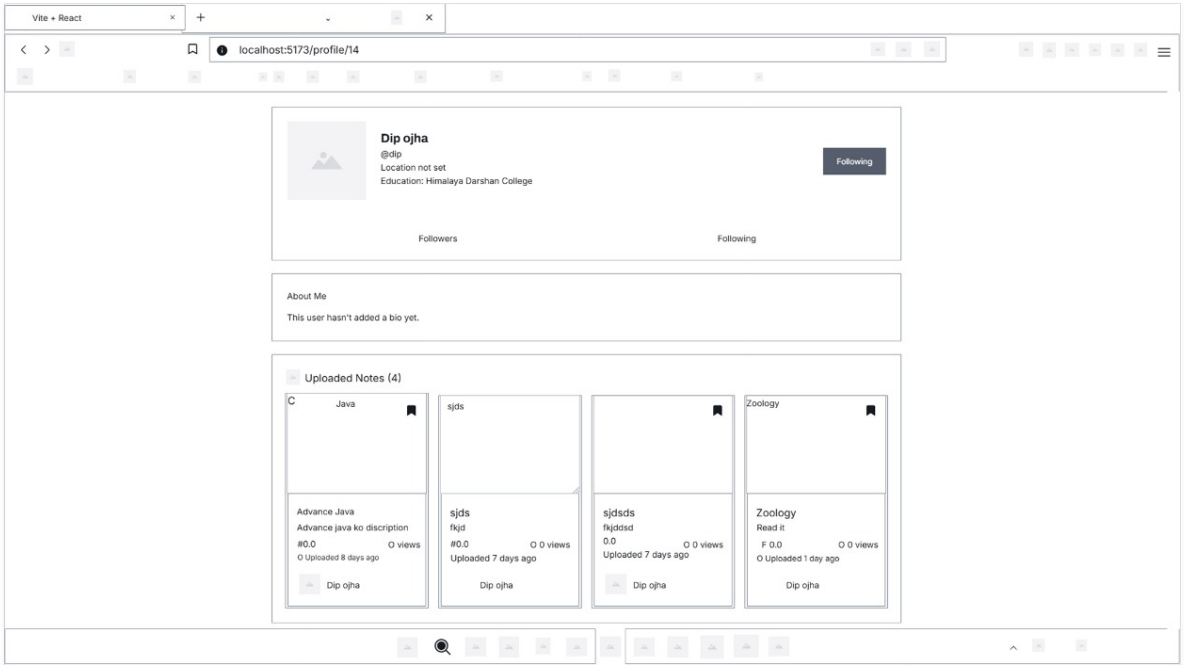
APPENDIX

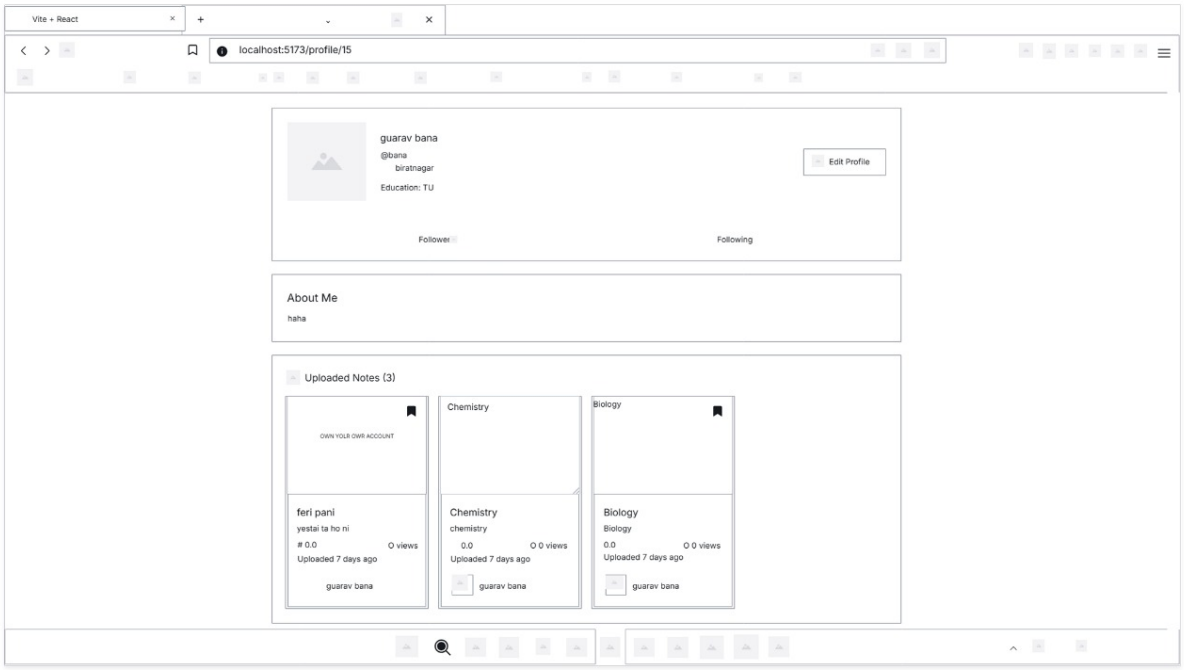

Vite + React
localhost:5173
Alnote
Sign in
to continue to Alnote
Username
Password
Forget password ?
Create account
Sign in
Help Terms Report

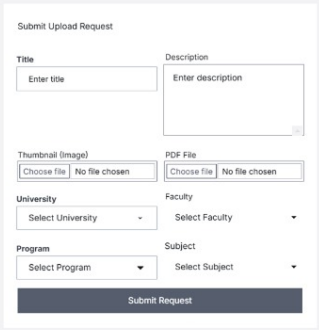

Vite + React
localhost:5173/register
Create Account
Username
Enter Your Username
Password
Enter Your Password
Security Question
Select a security question
Security Answer
Enter your security answer
Profile Picture
Back
Create Account

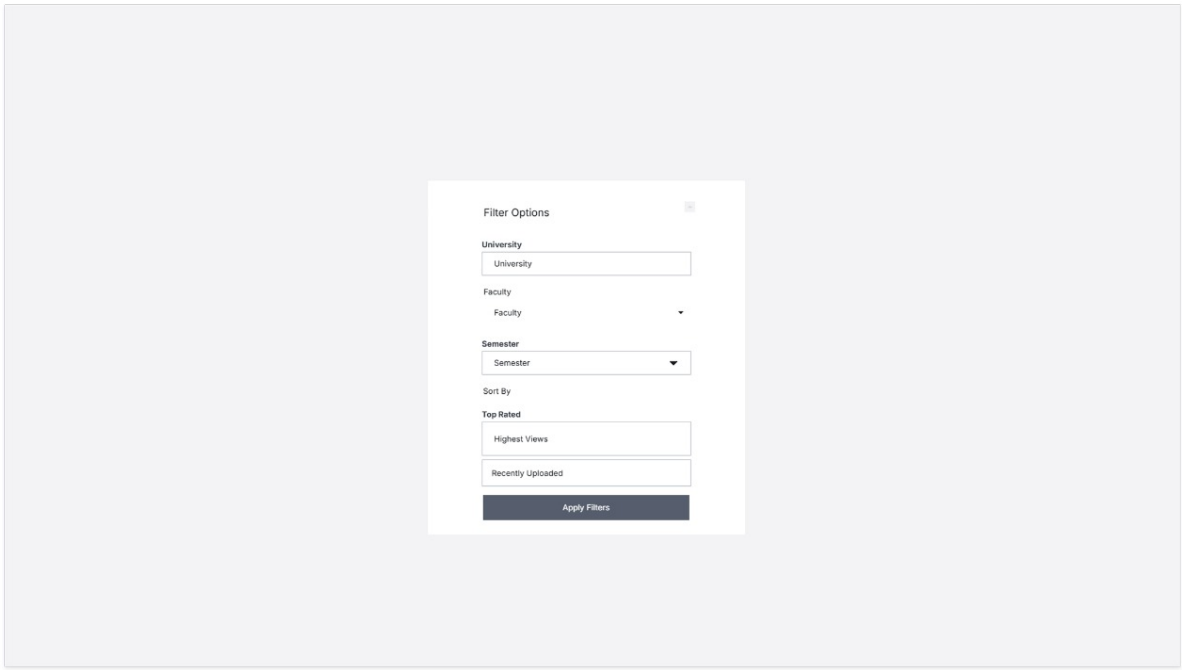

Vite + React
localhost:5173/register
Personal Information
First Name
Enter Your First Name
Last Name
Enter Your Last Name
Date Of Birth
dd/mm/yyyy
Phone Number
Enter Your Phone Number
Email Address
Enter Your Email Address
Education
Enter Your Education
Next

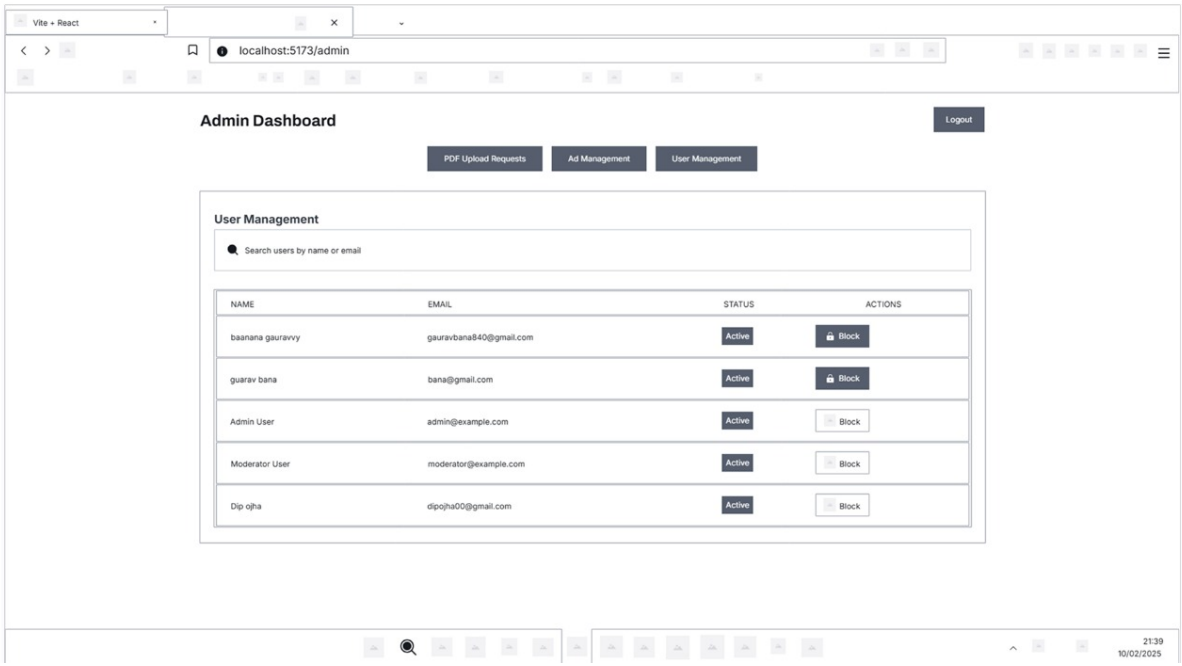

Vite + React
localhost:5173/layout/home
Alnore
Upload
Search for Notes or Profile
Filters
Home
Recent
Saved
Following
Dashboard
Earning
Premium
For You
Following
Advance Java
Advance Java
Advance java ko discription
Dip ojha
Image
Digital logic
Discription
Uploaded 7 days ago
Dip ojha
Math
Discription
Uploaded 7 days ago
Dip ojha
Social
Description
Uploaded 7 days ago
guarav bana
Ame
Chemistry
chemistry
Biology
Biology
Biology
Zoology
Read it
Zoology

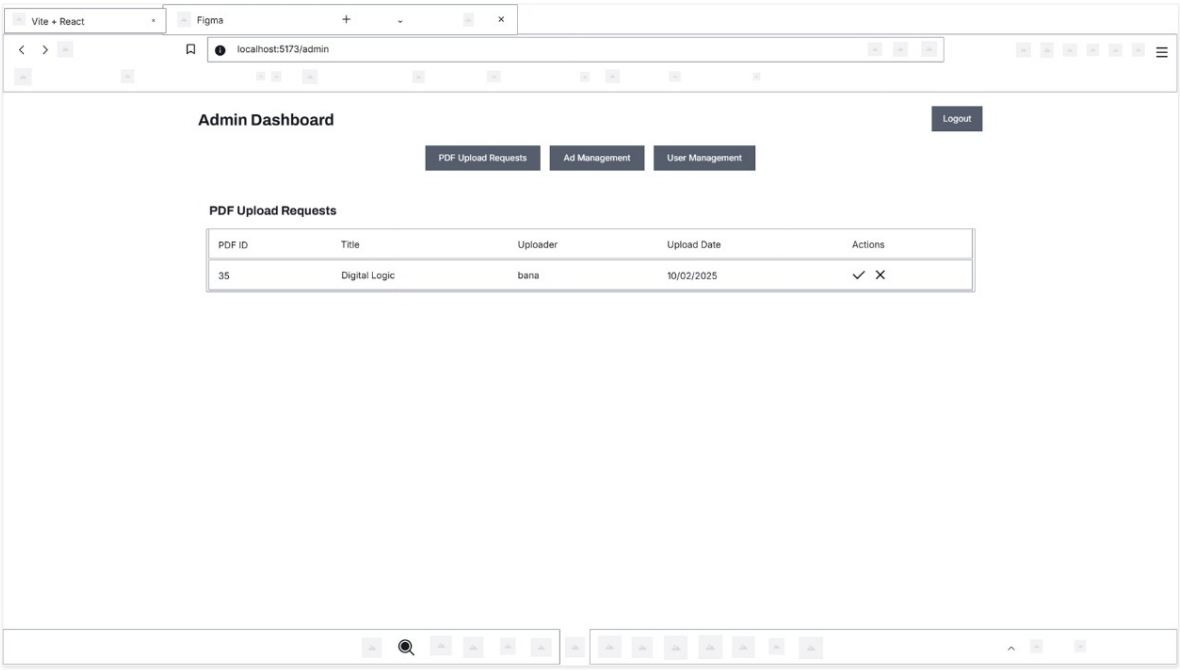

Vite + React
localhost:5173/profile/14
Dip ojha
@dip
Location not set
Education: Himalaya Darshan College
Following
Followers
Following
About Me
This user hasn't added a bio yet.
Uploaded Notes (4)
Java
Advance Java
Advance java ko discription
Dip ojha
sjds
sjds
fkjd
Uploaded 7 days ago
Dip ojha
sjdsds
fkjddsd
Uploaded 7 days ago
Dip ojha
Zoology
Zoology
Read it
Uploaded 1 day ago
Dip ojha

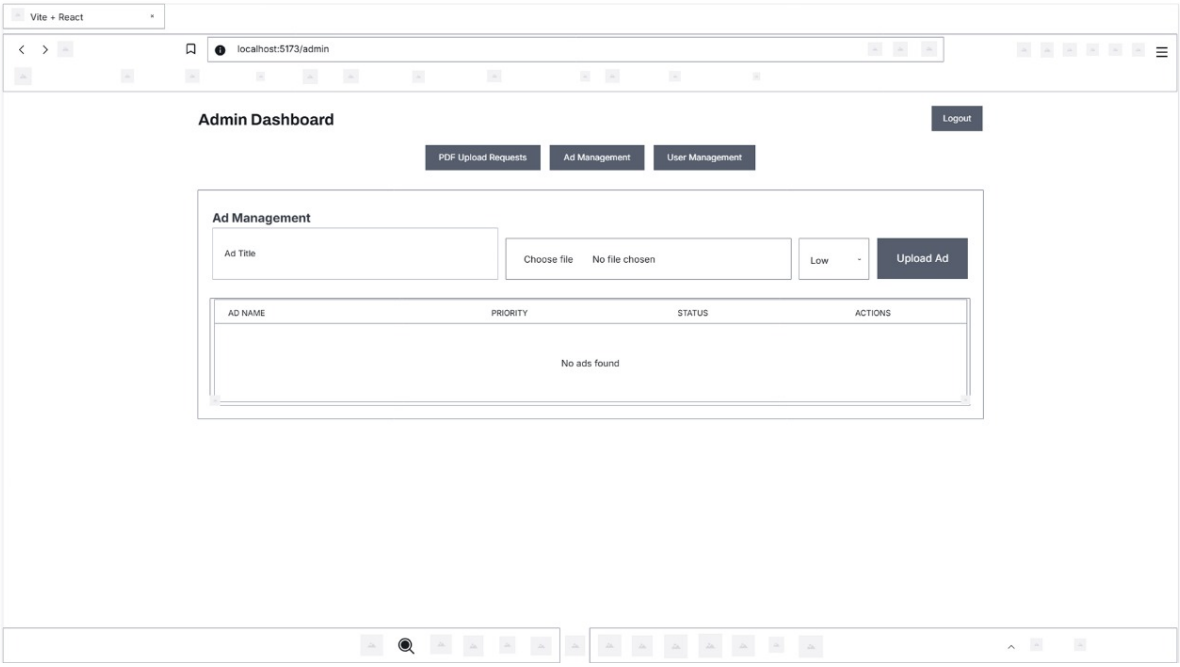

Vite + React
localhost:5173/profile/15
guarav bana
@bana
biratnagar
Education: TU
Edit Profile
Follower
Following
About Me
haha
Uploaded Notes (3)
OWN YOLR OWR ACCOUNT
feri pani
yestai ta ho ni
0.0
O views
Uploaded 7 days ago
guarav bana
Chemistry
Chemistry
chemistry
0.0
O 0 views
Uploaded 7 days ago
guarav bana
Biology
Biology
Biology
0.0
O 0 views
Uploaded 7 days ago
guarav bana

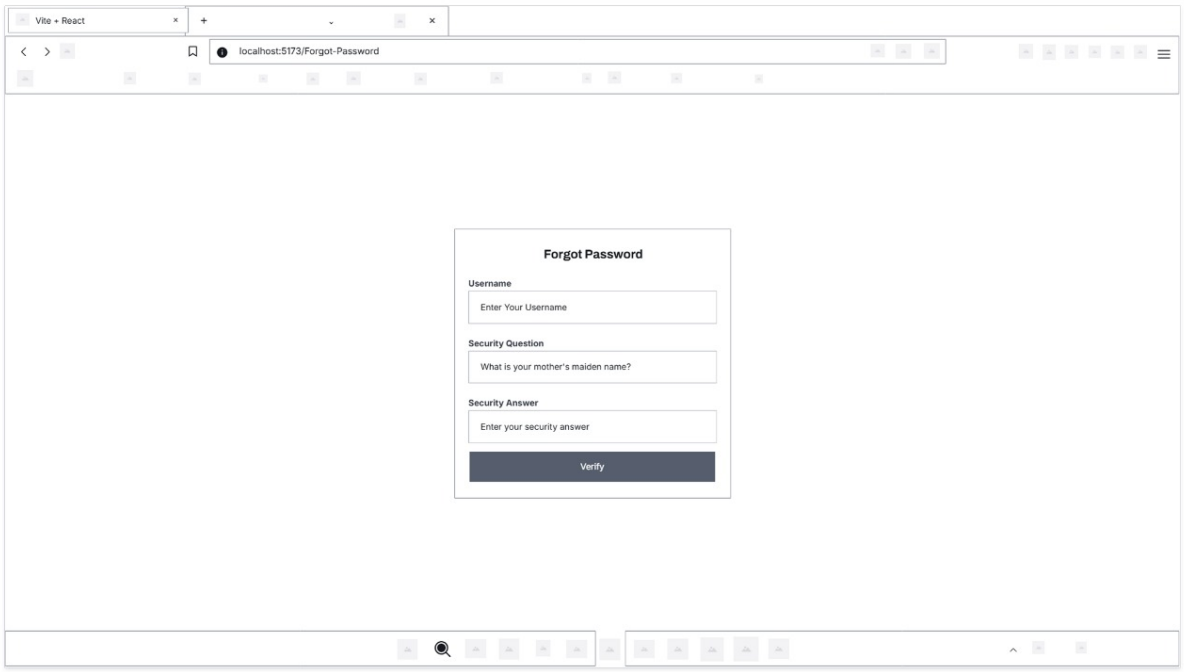

Submit Upload Request
Title
Enter title
Description
Enter description
Thumbnail (Image)
Choose file
No file chosen
PDF File
Choose file
No file chosen
University
Select University
Faculty
Select Faculty
Program
Select Program
Subject
Select Subject
Submit Request

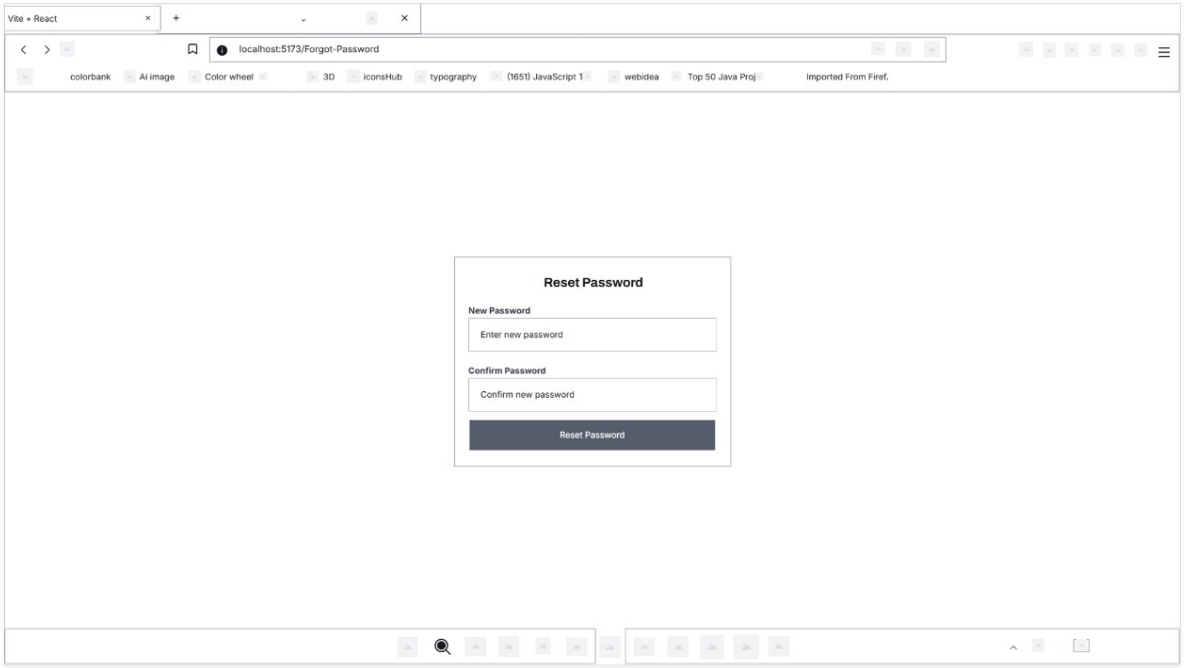

Filter Options
University
University
Faculty
Faculty
Semester
Semester
Sort By
Top Rated
Highest Views
Recently Uploaded
Apply Filters


Vite + React
localhost:5173/admin
Admin Dashboard
Logout
PDF Upload Requests
Ad Management
User Management
User Management
Search users by name or email
NAME
EMAIL
STATUS
ACTIONS
baanana gauravvy
gauravbana840@gmail.com
Active
Block
guarav bana
bana@gmail.com
Active
Block
Admin User
admin@example.com
Active
Block
Moderator User
moderator@example.com
Active
Block
Dip ojha
dipojha00@gmail.com
Active
Block
21:39
10/02/2025

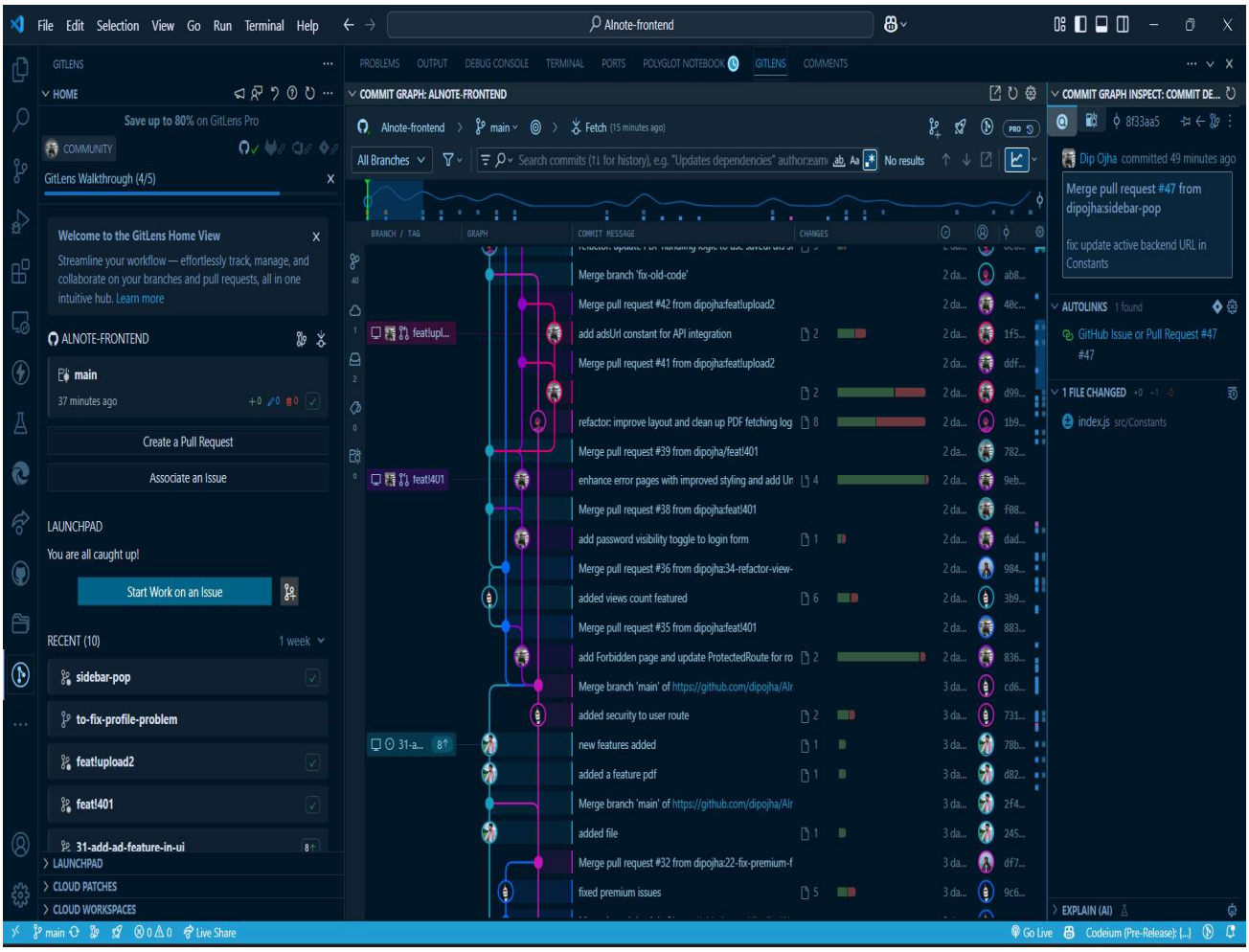

Vite + React
Figma
localhost:5173/admin
Admin Dashboard
Logout
PDF Upload Requests
Ad Management
User Management
PDF Upload Requests
PDF ID
Title
Uploader
Upload Date
Actions
35
Digital Logic
bana
10/02/2025


Vite + React
localhost:5173/admin
Admin Dashboard
Logout
PDF Upload Requests
Ad Management
User Management
Ad Management
Ad Title
Choose file
No file chosen
Low
Upload Ad
AD NAME
PRIORITY
STATUS
ACTIONS
No ads found

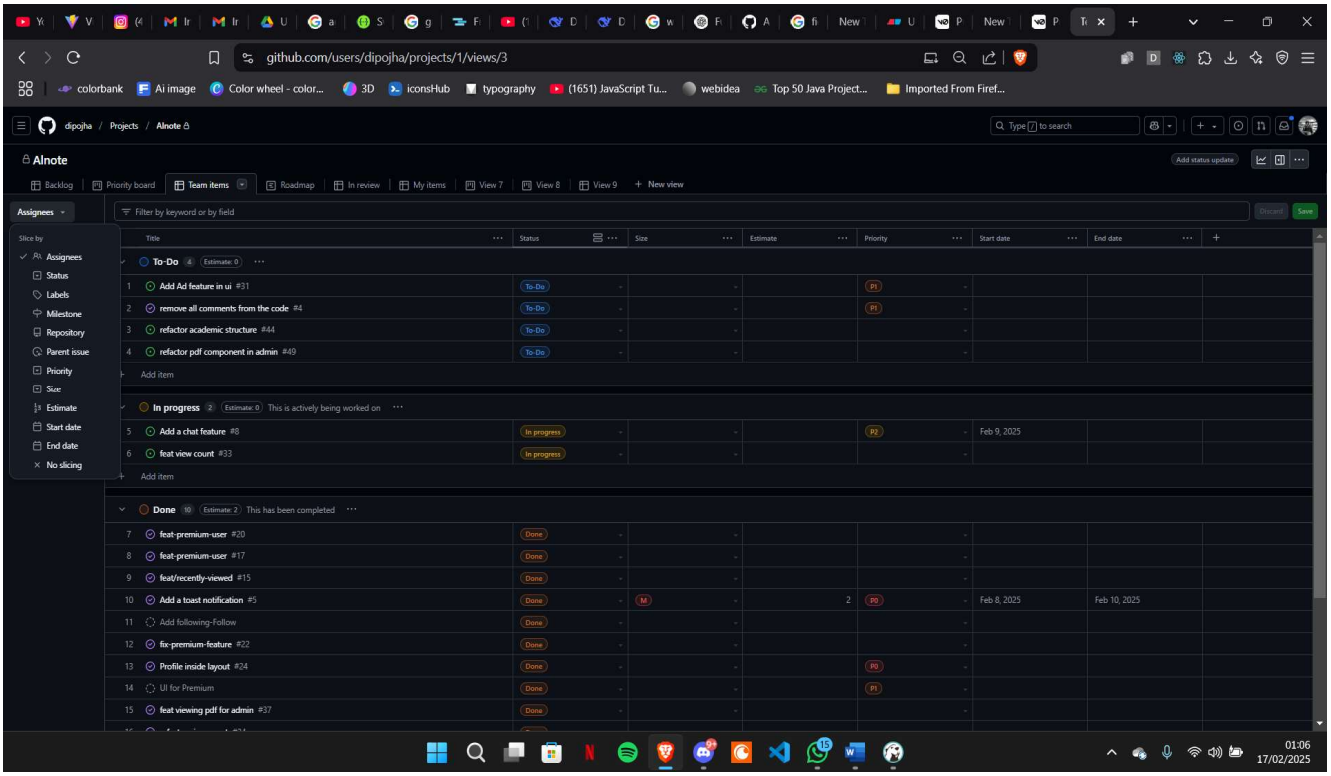

Vite + React
localhost:5173/Forgot-Password
Forgot Password
Username
Enter Your Username
Security Question
What is your mother's maiden name?
Security Answer
Enter your security answer
Verify

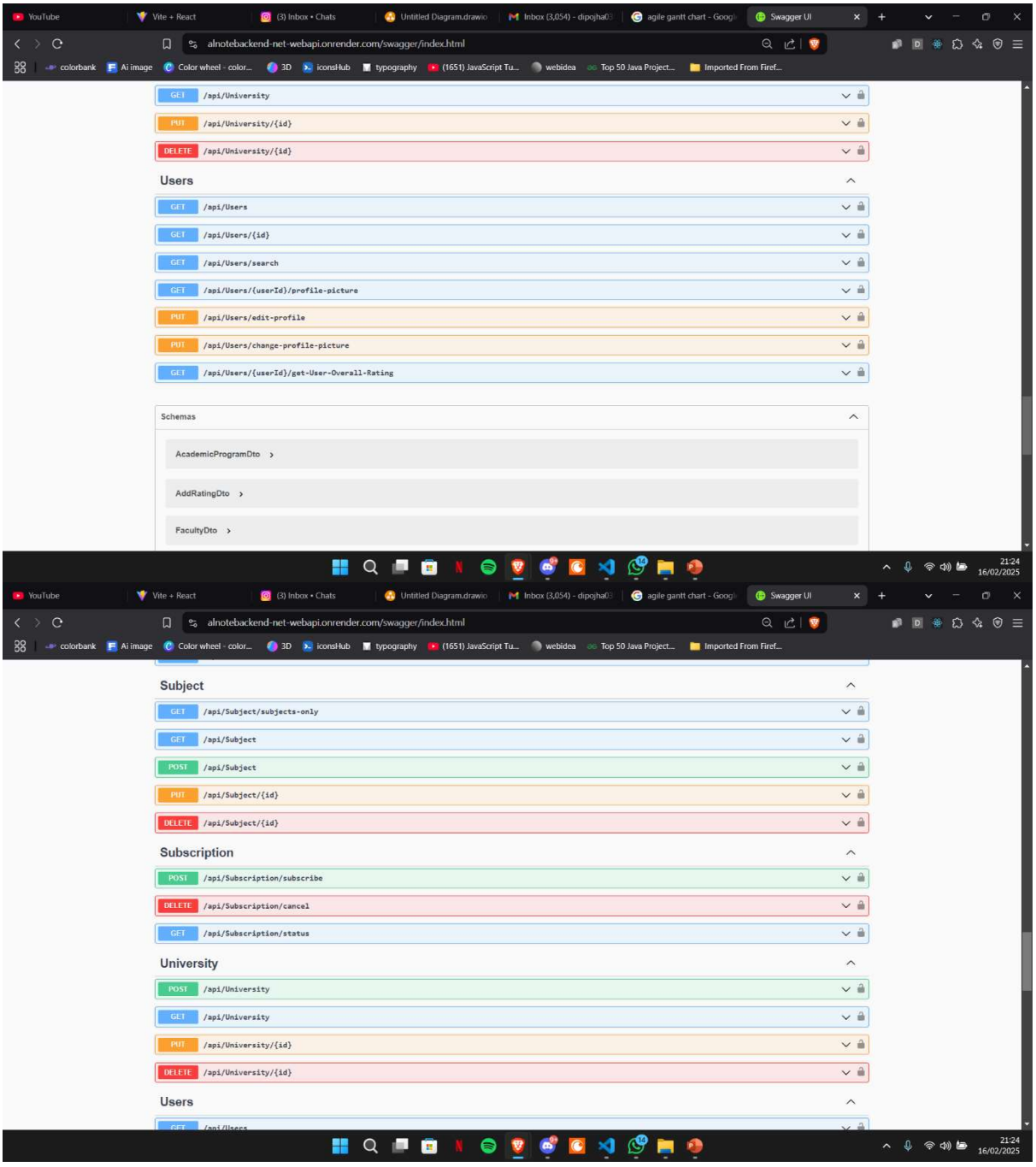

Vite + React
localhost:5173/Forgot-Password
colorbank
Ai image
Color wheel
3D
iconsHub
typography
(1651) JavaScript 1
webidea
Top 50 Java Proj
Imported From Firef.
Reset Password
New Password
Enter new password
Confirm Password
Confirm new password
Reset Password

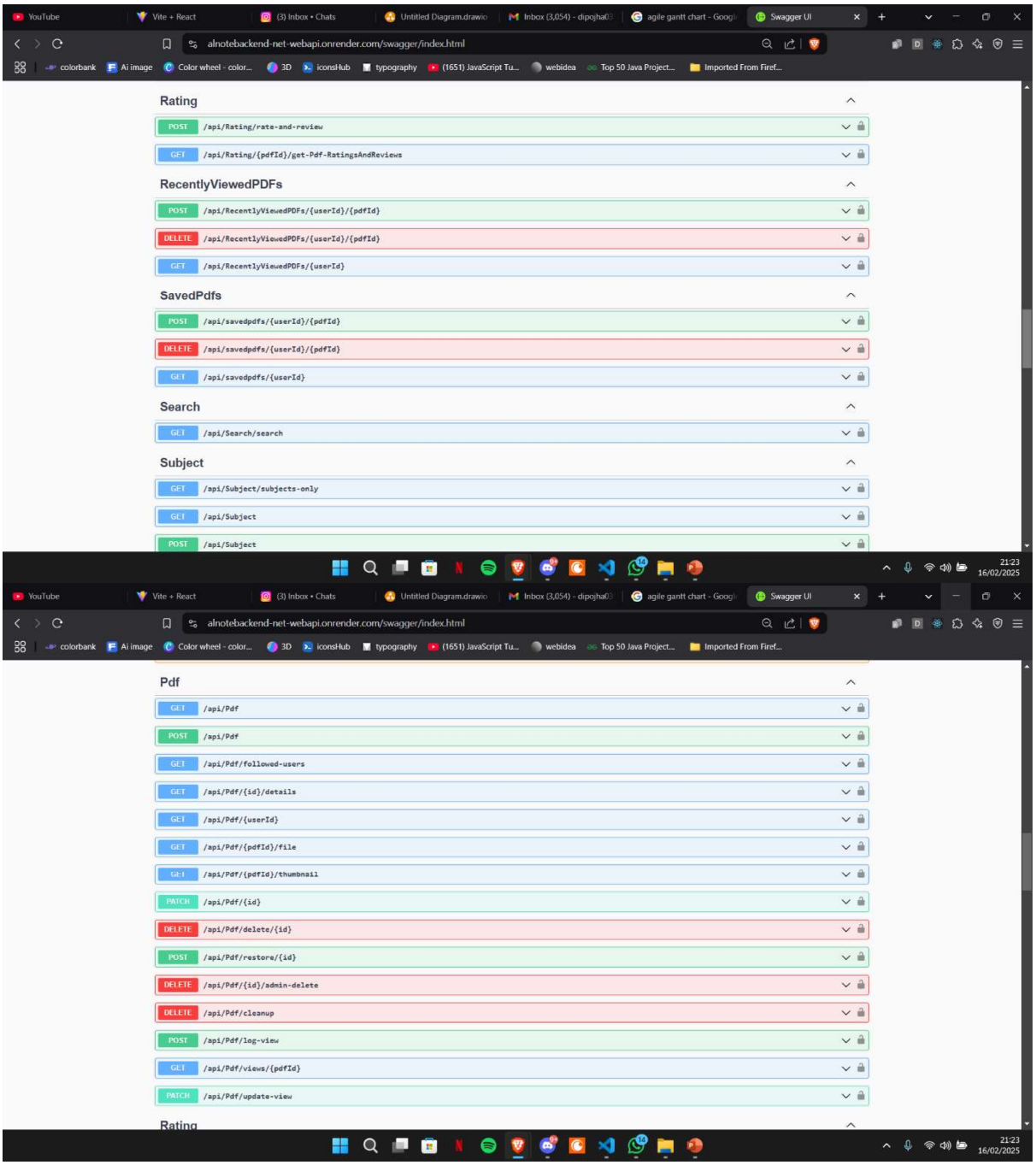
GIT LOG

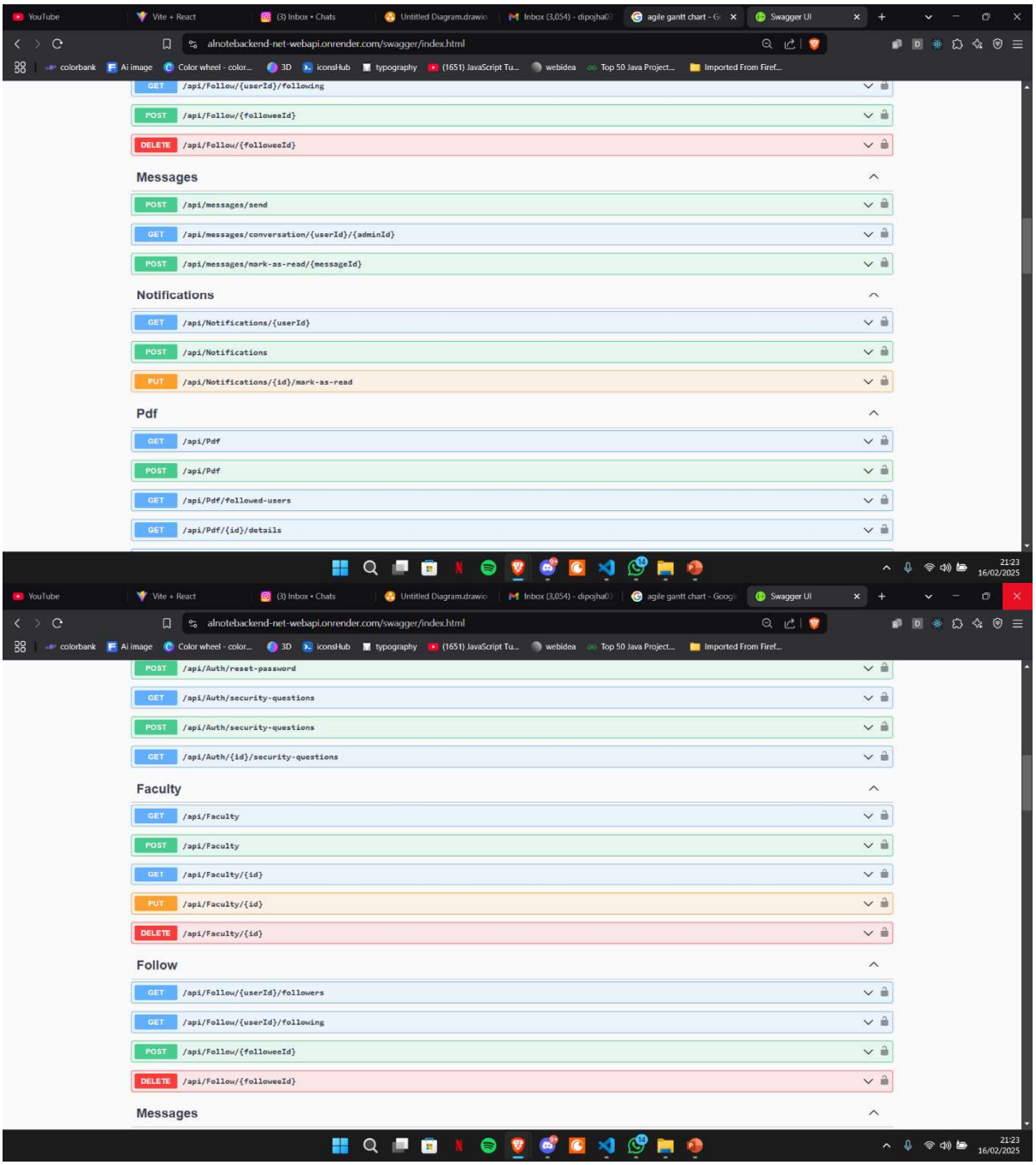

File Edit Selection View Go Run Terminal Help
Alnote-frontend
GITLENS
HOME
Save up to 80% on GitLens Pro
COMMUNITY
GitLens Walkthrough (4/5)
Welcome to the GitLens Home View
Streamline your workflow — effortlessly track, manage, and collaborate on your branches and pull requests, all in one intuitive hub. Learn more
ALNOTE-FRONTEND
main
37 minutes ago
Create a Pull Request
Associate an Issue
LAUNCHPAD
You are all caught up!
Start Work on an Issue
RECENT (10)
1 week
sidebar-pop
to-fix-profile-problem
feat!upload2
feat!401
31-add-ad-feature-in-ui
LAUNCHPAD
CLOUD PATCHES
CLOUD WORKSPACES
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS POLYGLOT NOTEBOOK GITLENS COMMENTS
COMMIT GRAPH: ALNOTE-FRONTEND
Alnote-frontend main Fetch (15 minutes ago)
All Branches
Search commits (↑↓ for history), e.g. "Updates dependencies" author:eam
No results
BRANCH / TAG GRAPH COMMIT MESSAGE CHANGES
Merge branch 'fix-old-code'
Merge pull request #42 from dipojha:feat!upload2
add adsUrl constant for API integration
Merge pull request #41 from dipojha:feat!upload2
refactor: improve layout and clean up PDF fetching log
Merge pull request #39 from dipojha/feat!401
enhance error pages with improved styling and add Ur
Merge pull request #38 from dipojha:feat!401
add password visibility toggle to login form
Merge pull request #36 from dipojha:34-refactor-view-
added views count featured
Merge pull request #35 from dipojha:feat!401
add Forbidden page and update ProtectedRoute for ro
Merge branch 'main' of https://github.com/dipojha/Alr
added security to user route
new features added
added a feature pdf
Merge branch 'main' of https://github.com/dipojha/Alr
added file
Merge pull request #32 from dipojha:22-fix-premium-f
fixed premium issues
COMMIT GRAPH INSPECT: COMMIT DE...
8f33aa5
Dip Ojha committed 49 minutes ago
Merge pull request #47 from dipojha:sidebar-pop
fix: update active backend URL in Constants
AUTOLINKS 1 found
GitHub Issue or Pull Request #47
#47
1 FILE CHANGED
index.js src/Constants
EXPLAIN (AI)
main
Live Share
Go Live
Codeium (Pre-Release): (...)

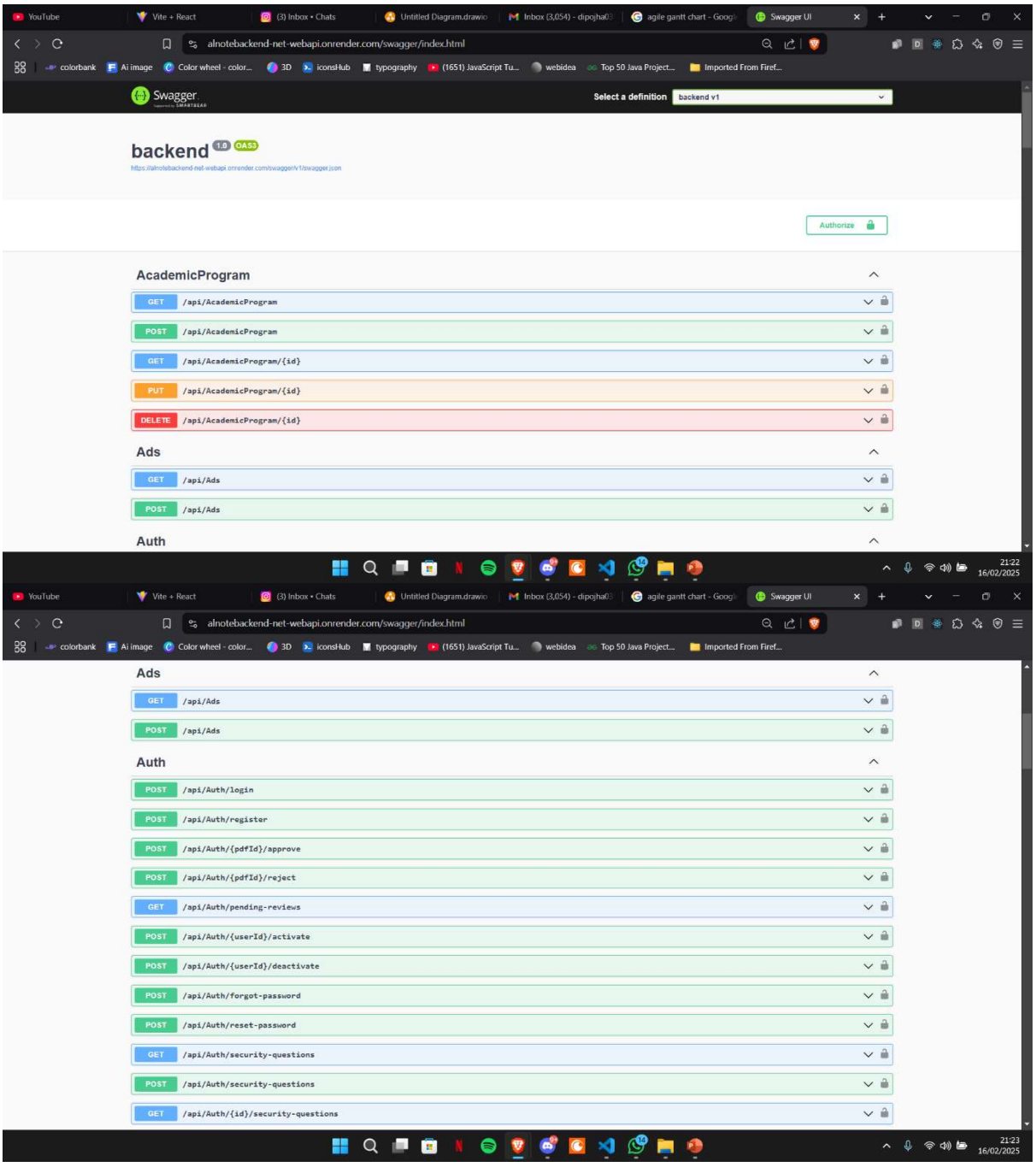
GIT ACTIVITY LOG




alnotebackend-net-webapi.onrender.com/swagger/index.html
GET /api/University
PUT /api/University/{id}
DELETE /api/University/{id}
Users
GET /api/Users
GET /api/Users/{id}
GET /api/Users/search
GET /api/Users/{userId}/profile-picture
PUT /api/Users/edit-profile
PUT /api/Users/change-profile-picture
GET /api/Users/{userId}/get-User-Overall-Rating
Schemas
AcademicProgramDto
AddRatingDto
FacultyDto
Subject
GET /api/Subject/subjects-only
GET /api/Subject
POST /api/Subject
PUT /api/Subject/{id}
DELETE /api/Subject/{id}
Subscription
POST /api/Subscription/subscribe
DELETE /api/Subscription/cancel
GET /api/Subscription/status
University
POST /api/University
GET /api/University
PUT /api/University/{id}
DELETE /api/University/{id}
Users


alnotebackend-net-webapi.onrender.com/swagger/index.html
Rating
POST /api/Rating/rate-and-review
GET /api/Rating/{pdfId}/get-Pdf-RatingsAndReviews
RecentlyViewedPDFs
POST /api/RecentlyViewedPDFs/{userId}/{pdfId}
DELETE /api/RecentlyViewedPDFs/{userId}/{pdfId}
GET /api/RecentlyViewedPDFs/{userId}
SavedPdfs
POST /api/savedpdfs/{userId}/{pdfId}
DELETE /api/savedpdfs/{userId}/{pdfId}
GET /api/savedpdfs/{userId}
Search
GET /api/Search/search
Subject
GET /api/Subject/subjects-only
GET /api/Subject
POST /api/Subject
Pdf
GET /api/Pdf
POST /api/Pdf
GET /api/Pdf/followed-users
GET /api/Pdf/{id}/details
GET /api/Pdf/{userId}
GET /api/Pdf/{pdfId}/file
GET /api/Pdf/{pdfId}/thumbnail
PATCH /api/Pdf/{id}
DELETE /api/Pdf/delete/{id}
POST /api/Pdf/restore/{id}
DELETE /api/Pdf/{id}/admin-delete
DELETE /api/Pdf/cleanup
POST /api/Pdf/log-view
GET /api/Pdf/views/{pdfId}
PATCH /api/Pdf/update-view
Rating


alnotebackend-net-webapi.onrender.com/swagger/index.html
GET /api/Follow/{userId}/following
POST /api/Follow/{followeeId}
DELETE /api/Follow/{followeeId}
Messages
POST /api/messages/send
GET /api/messages/conversation/{userId}/{adminId}
POST /api/messages/mark-as-read/{messageId}
Notifications
GET /api/Notifications/{userId}
POST /api/Notifications
PUT /api/Notifications/{id}/mark-as-read
Pdf
GET /api/Pdf
POST /api/Pdf
GET /api/Pdf/followed-users
GET /api/Pdf/{id}/details
alnotebackend-net-webapi.onrender.com/swagger/index.html
POST /api/Auth/reset-password
GET /api/Auth/security-questions
POST /api/Auth/security-questions
GET /api/Auth/{id}/security-questions
Faculty
GET /api/Faculty
POST /api/Faculty
GET /api/Faculty/{id}
PUT /api/Faculty/{id}
DELETE /api/Faculty/{id}
Follow
GET /api/Follow/{userId}/followers
GET /api/Follow/{userId}/following
POST /api/Follow/{followeeId}
DELETE /api/Follow/{followeeId}
Messages


alnotebackend-net-webapi.onrender.com/swagger/index.html
Swagger
Select a definition
backend v1
backend
1.0
OAS3
https://alnotebackend-net-webapi.onrender.com/swagger/v1/swagger.json
Authorize
AcademicProgram
GET /api/AcademicProgram
POST /api/AcademicProgram
GET /api/AcademicProgram/{id}
PUT /api/AcademicProgram/{id}
DELETE /api/AcademicProgram/{id}
Ads
GET /api/Ads
POST /api/Ads
Auth
21:22
16/02/2025
alnotebackend-net-webapi.onrender.com/swagger/index.html
Ads
GET /api/Ads
POST /api/Ads
Auth
POST /api/Auth/login
POST /api/Auth/register
POST /api/Auth/{pdfId}/approve
POST /api/Auth/{pdfId}/reject
GET /api/Auth/pending-reviews
POST /api/Auth/{userId}/activate
POST /api/Auth/{userId}/deactivate
POST /api/Auth/forgot-password
POST /api/Auth/reset-password
GET /api/Auth/security-questions
POST /api/Auth/security-questions
GET /api/Auth/{id}/security-questions
21:23
16/02/2025