

EduQuest Islamabad and Rawalpindi Uni Guide



Gul Muhammad

BSE203005

Eman Yaqoob

BSE211043

**Supervised by
Mr. Temour Abbas**

Spring 2024

**Software Engineering
Department of Software Engineering**

Capital University of Science & Technology, Islamabad

Project Report



Version	01
----------------	----

NUMBER OF MEMBERS	02
--------------------------	----

TITLE	Edu Quest: Islamabad and Rawalpindi UniGuide
--------------	--

SUPERVISOR NAME	Mr. Temour Abbas
------------------------	------------------

MEMBER NAME	REG. NO.	EMAIL ADDRESS
Gul Muhammad	BSE203005	Bse203005@cust.pk
Eman Yaqoob	BSE211043	Bse211043@cust.pk

MEMBERS' SIGNATURES

Supervisor's Signature

Approval Certificate

This project, entitled as “Edu Quest: Islamabad and Rawalpindi Uni Guide” has been approved for the award of

Bachelor of Science in Software Engineering

Committee Signatures:

Supervisor:

(Supervisor Name Dr / Mr. / Ms.)

Project Coordinator:

(Mr. Ibrar Arshad)

Head of Department:

(Dr. Nadeem Anjum)

Declaration

I/We, hereby, declare that “No portion of the work referred to, in this project has been submitted in support of an application for another degree or qualification of this or any other university/institute or other institution of learning”. It is further declared that this undergraduate project, neither as a whole nor as a part thereof has been copied out from any sources, wherever references have been provided.

MEMBERS' SIGNATURES

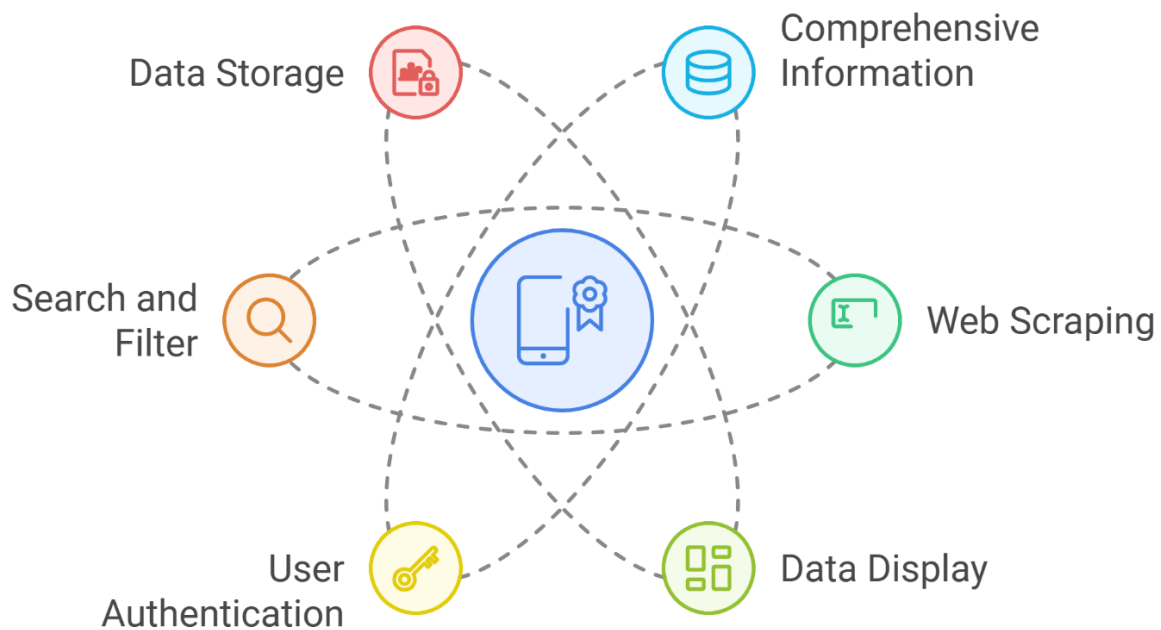
Acknowledgments

It is usual to thank those individuals who have provided particularly useful assistance, technical or otherwise, during your project. Your supervisor will be pleased to be acknowledged as he or she will have invested quite a lot of time overseeing your progress.

Executive Summary

This project proposes the development of a centralized web-based application to streamline university information access. This project involves developing a web scraper in Python to collect key data from university websites, including names, logos, About Us sections, contact details, and fee structures. The scraped data is formatted in JSON and integrated into a full-stack system using React.js for the front-end and Laravel for the back-end. React handles the display of university information, while Laravel supports user registration, login, and authentication via secure APIs. MySQL is used for storing user data. The system provides an efficient, user-friendly platform for accessing and displaying university details.

University Information Platform Overview



Key Functionalities:

- **Comprehensive Information:** Users will have access to a wealth of university information, including details on departments, academic calendars, fee structures, and more.
- **Web Scraping:** Automatically collects key information from university websites, such as name, logo, About Us, contact details, and fee structure.
- **Data Display:** Displays scraped university data in a user-friendly interface using React.js, allowing users to easily navigate and view detailed university profiles.
- **User Authentication:** Secure user registration and login functionality, ensuring personalized access and data management through Laravel's authentication system.
- **Search and Filter:** Allows users to search for universities and filter results based on specific criteria such as fee structure or contact details.
- **Data Storage:** User data and session information are stored securely in a MongoDB database for efficient management and retrieval.

Benefits:

- **Streamlined Information Access:** Users can quickly and easily access essential university information from various institutions in one place.
- **Timesaving:** Automates the process of collecting and presenting university details, saving users time compared to manually searching individual websites.
- **Personalized Experience:** The platform supports user registration, offering a personalized experience with secure login and data management.
- **Efficient Data Management:** By using structured JSON data, the platform ensures easy integration and efficient data retrieval.
- **Secure and Scalable:** The use of Laravel and MySQL ensures secure user authentication and scalability as the platform grows.

Table of Content

Chapter 1.....	14
1. Introduction	14
1.1 Project Introduction	14
1.2 Beneficiaries:	14
1.3 Existing Systems / Solutions	15
1.4 Business Scope Target Market:	16
1.5 Useful Tools and Technologies	18
1.6 Project Work Break Down.....	21
1.7 Project Timeline	21
1.8 Summary.....	22
Chapter 2.....	23
2. Requirement Specification and Analysis.....	23
2.1 Functional Requirements	23
2.2 Non-Functional Requirements.....	24
2.3 System Use Case Modeling.....	25
2.4 Use case Description.....	26
2.5. System Sequence Diagram	33
2.5.3. Search	34
2.5.4. Comparison.....	35
2.5.5. Chat	s36
2.5.6. Manage Chat	37
2.5.7. Approve Registration	38
2.6. Domain Model:.....	39
2.7. GUI:.....	40

Chapter 3.....	45
3.1. System Diagram	45
3.2. Class Diagram.....	46
3.3. Sequence Diagram.....	47
3.4. Entity Relationship Diagram	48
3.5. Database Schema.....	49
3.6. Software COTS.....	49
Chapter 4.....	50
4. Software Development	50
4.1 Coding Standards.....	50
4.2. Development Environment.....	51
4.3. Software Description	52
Chapter 5	116
5. Software Testing.....	116
5.1 Testing Methodology.....	116
5.2 Test cases.....	116
Chapter 6.....	121
6. Software Deployment.....	121
6.1 Deployment Platform	121
6.1.1. AWS (Amazon Web Service)	121
6.2 Deployment Process Description	121
6.2.1. Creating an account on AWS	121
6.2.2. Dashboard.....	122
6.2.3. Service (Elastic Beanstalk).....	122
6.2.4. Creating Configure Environment	123
6.2.5. Creating a .rar file for uploading.....	123

6.2.6. Uploading the Application Code on the AWS	124
6.2.7. Creating a Cloud MongoDB Database (cluster)	124
6.2.8. Creating a Database user	125
6.2.9. Database cluster connected with web server.....	125
References	126
Plagiarism Report.....	127
Report Approval Certificate	128

Table of Figures

Figure 1 Comparison Chart.....	15
Figure 2 React.js Development Cycle.....	18
Figure 3 Node.js Development Cycle.....	19
Figure 4 Python for Web Scraper	19
Figure 5 System Overview.....	20
Figure 6 Figma Design System.....	20
Figure 7 Project Work Break down	21
Figure 8 Project Timeline	21
Figure 9 Requirement Analysis	23
Figure 10 Use-case Modeling	25
Figure 11 Registration Sequence Diagram	33
Figure 12 Login Sequence Diagram	33
Figure 13 Search Sequence Diagram.....	34
Figure 14 Comparison Sequence Diagram	35
Figure 15 Chat Feature.....	36
Figure 16 Manage Chat.....	37
Figure 17 Approve Registration.....	38
Figure 18 Domain Model.....	39
Figure 19 Welcome Page	40
Figure 20 Registration Page.....	40
Figure 21 Login Page.....	41
Figure 22 Dashboard.....	41
Figure 23 University Details	42
Figure 24 User Chats	42
Figure 25 Comparison Page.....	43
Figure 26 Admin Dashboard.....	43
Figure 27 Admin chat management	44
Figure 28 Admin Approve Registration.....	44
Figure 29 Software Architecture Diagram.....	45
Figure 30 Class Diagram	46
Figure 31 Sequence Diagram.....	47
Figure 32 Entity Relationship Diagram	48
Figure 33 Database Schema.....	49
Figure 34 Creating an account for AWS.....	121
Figure 35 AWS Dashboard.....	122
Figure 36 Elastic Beanstalk (Service).....	122
Figure 37 Creating Configure Environment	123
Figure 38 Creating the .rar file of the Project	123
Figure 39 Uploading the Application Code	124
Figure 40 Creating a Cloud MongoDB Database (cluster).....	124
Figure 41 Database cluster connected with web server	125

List of Tables

Table 1 User Registration	116
Table 2 User Login	117
Table 3 Search University.....	117
Table 4 Compare Universities.....	118
Table 5 User Chat	118
Table 6 Admin Sign-in.....	119
Table 7 Approve Registration	119
Table 8 Delete Chat	120

Chapter 1

1. Introduction

The project aims to develop a comprehensive web-based application that provides detailed information about universities in Rawalpindi and Islamabad, including departmental offerings, fee structures, and other relevant details. The platform will serve as a centralized hub for prospective users and educators seeking information about educational institutions in the region.

1.1 Project Introduction

This project proposes a web application to be a one-stop shop for prospective Users, and educators seeking information about universities in Rawalpindi and Islamabad. Currently, the process of researching universities can be scattered and time-consuming. This application aims to solve this problem by providing a centralized platform with detailed information on universities, departments, programs, and more.

1.2 Beneficiaries:

The target beneficiaries are threefold:

- **Prospective Students:** Individuals seeking information about various universities, including their programs, fees, and contact details, to make informed decisions about their education.
- **Parents:** Those looking to explore and compare universities for their children's higher education options.
- **Education Consultants:** Professionals who advise students and families on the best university choices based on detailed institutional information.
- **University Administrators:** Institutions may use the platform for competitive analysis or to ensure their data is correctly represented online.

1.3 Existing Systems / Solutions

Websites like:

1.3.1 HEC [Higher Education Commission]

- Website link: <https://www.hec.gov.pk/english/Pages/default.aspx>

1.3.2 Top Universities:

- Website link: <https://www.topuniversities.com>

1.3.3 Morweb:

- Website link: <https://morweb.org/post/college-websites>

1.3.4 All Universities info:

- Website link: <https://www.alluniversity.info>

COMPARISON CHART

FEATURES	DEPARTMENT DETAILS	SEARCH AND FILTER	SCHOLARSHIPS DETAILS	CONTACT INFORMATION	LOCATION OF UNIVERSITIES	FEEDBACK	FACULTY DETAILS
TOP UNIVERSITY	✓	✓	✗	✗	✓	✓	✗
MORWEB	✓	✗	✗	✓	✗	✓	✗
ALL UNIVERSITY INFO	✓	✓	✗	✓	✗	✗	✗
HEC	✗	✓	✗	✓	✓	✓	✗

Figure 1 Comparison Chart

1.4 Business Scope

Target Market:

1.4.1 Prospective Users in Rawalpindi & Islamabad: Individuals seeking information about universities in the region to make informed decisions about their higher education.

1.4.2 Parents of Prospective Users: Parents looking for comprehensive and accessible university information to guide their children's educational choices.

1.4.3 Educators (High School Counselors, Career Advisors): Professionals assisting students in exploring and selecting universities based on academic interests, career goals, and financial considerations.

Value Proposition:

Centralized Hub: A one-stop platform providing comprehensive university information, including programs, fees, faculty, and more, for prospective students in Rawalpindi and Islamabad.

Informed Decision-Making: Streamlined search and filtering capabilities based on user criteria (location, program, fee, etc.), allowing users to easily find universities that match their needs.

Personalized Recommendations: User accounts and profiles provide tailored suggestions based on academic interests, goals, and preferences, enhancing the user experience.

Transparency and Accessibility: The platform digitizes university information, ensuring clear, transparent, and easily accessible admissions data for prospective students and their families.

Growth Potential:

Expand Geographically: Consider including other cities in Pakistan or neighboring regions.

Offer Additional Services: Integrate application tools, scholarship matching, or career counseling services (potentially in partnership with universities or relevant institutions).

Freemium Model: Offer a basic platform with core features for free, while premium options unlock additional benefits.

Data Analytics: Leverage user data to provide valuable insights to universities, helping them tailor their programs and offerings

Competitive Advantage:

Comprehensiveness: Provide deeper information compared to existing platforms, offering a richer user experience.

Focus on Region: Cater specifically to Users in Rawalpindi & Islamabad, addressing their local needs and preferences.

Personalization: Tailor the platform to individual user needs, enhancing decision-making and program selection.

1.5 Useful Tools and Technologies

React.js:

React.js, commonly known as React, is an open-source JavaScript library developed by Facebook. It is used for building user interfaces, especially for single-page applications. React.js is a powerful library for building user interfaces. Its innovative features make it a compelling choice for modern web and mobile app development. [1]

React.js Development Cycle

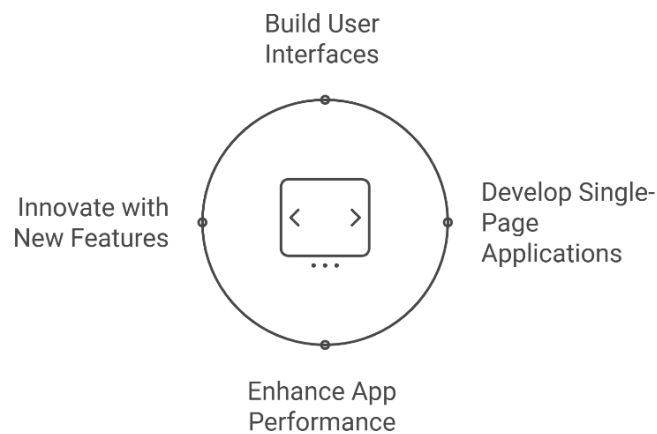


Figure 2 React.js Development Cycle

Node.js:

Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to run JavaScript code on the server side. Node.js is a runtime environment that has gained popularity for its performance using JavaScript on both the client and server sides. [2]

Node.js Development Cycle

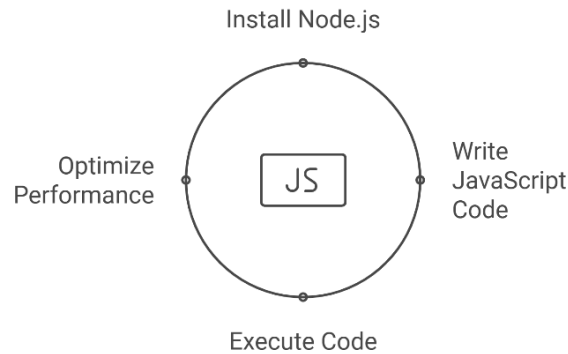


Figure 3 Node.js Development Cycle

Python:

Python is a versatile, high-level programming language used for web scraping, data analysis, automation, and more. For this project, Python is used to develop a web scraper that collects data from various university websites. Its simplicity and powerful libraries (like BeautifulSoup, lxml and requests) make it ideal for web scraping tasks.

Python for Web Scraping

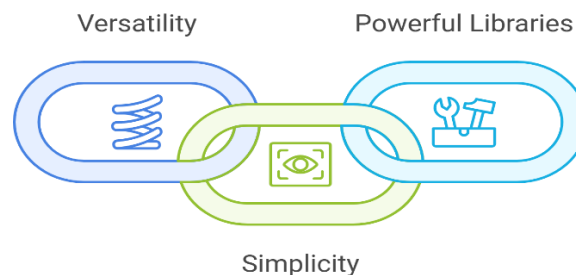


Figure 4 Python for Web Scraper

MongoDB (Database):

MongoDB is an open-source database management system used for storing and organizing data. In this project, MongoDB is used as the backend database to store user data, university information, and other system-related data. Its robust querying capabilities, security features, and scalability make it an ideal choice for managing application data.

System Architecture Overview

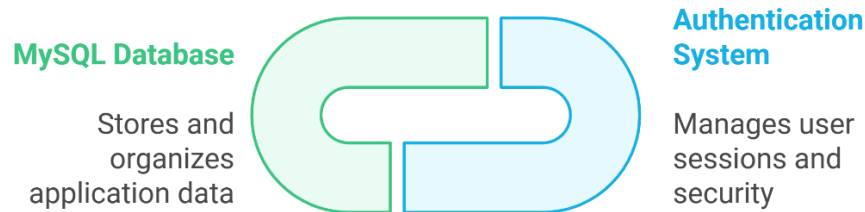


Figure 5 System Overview

Figma:

Figma is a cloud-based design and prototyping tool that has gained significant popularity among designers and teams for its collaborative features and versatility. Figma is a cloud-based, collaborative, and versatile design feature, combined with its commitment to accessibility, and justifies its widespread adoption among designers and design teams.

Figma's Design Ecosystem

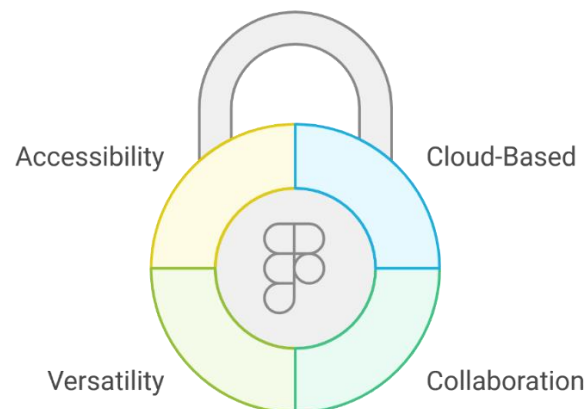


Figure 6 Figma Design System

1.6 Project Work Break Down

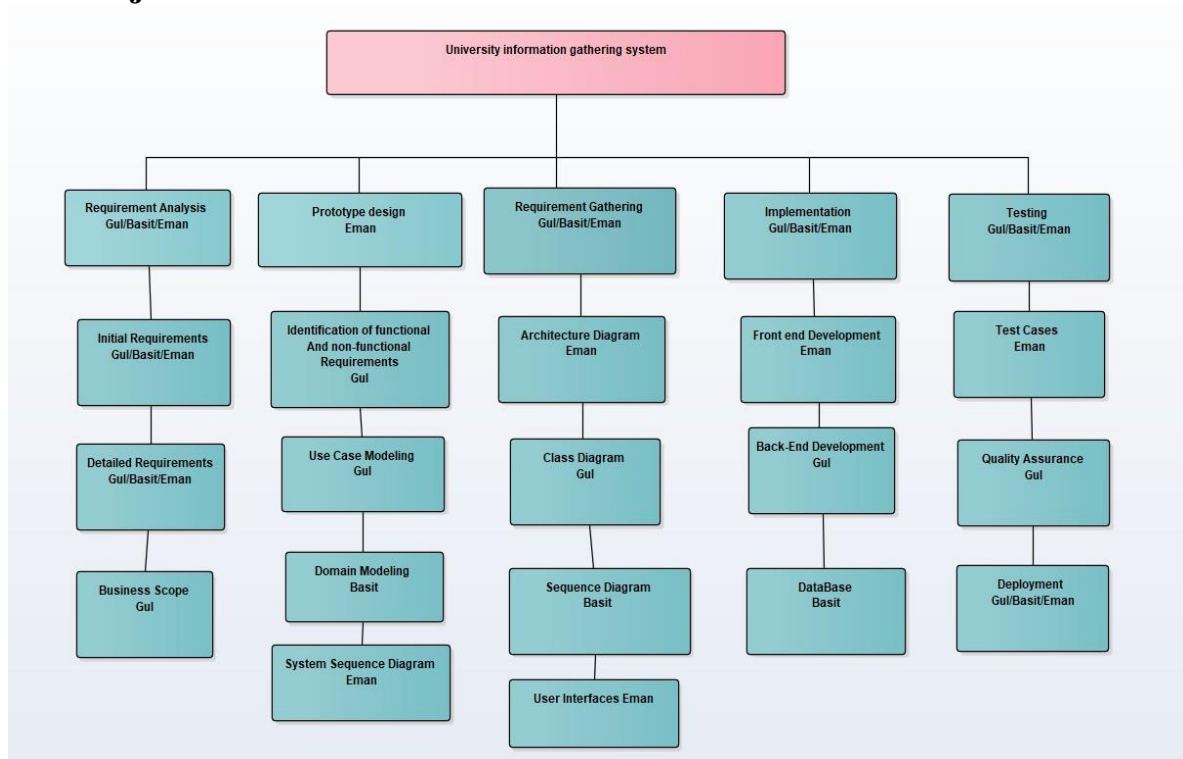


Figure 7 Project Work Break down

1.7 Project Timeline

To represent the project schedule Gantt chart is developed. Gantt chart which shows a proposed timeline for the project. The tasks are divided into two Iterations. All the phases mentioned in each iteration cover an estimated period that will lead to the completion of the project in a systematic way.

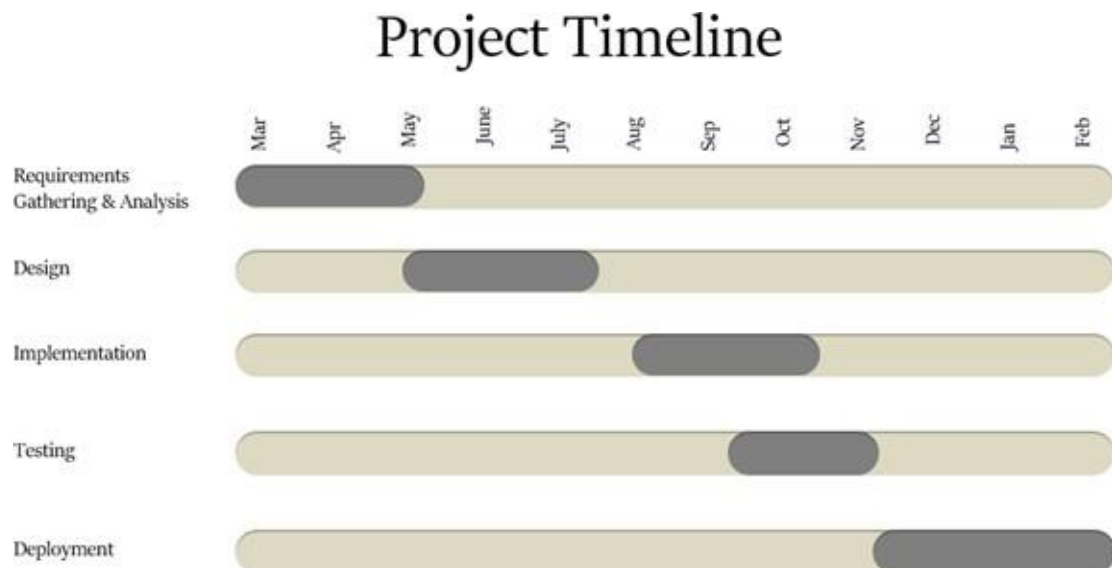


Figure 8 Project Timeline

1.8 Summary

introduces the project, providing a comprehensive overview of its objectives, relevance, and scope. The chapter outlines the problem the project aims to solve, identifying the need for a centralized platform to access university information efficiently. It also highlights existing solutions in the market and distinguishes this project by its unique features, such as its integration of web scraping, dynamic data presentation, and user-centric interface. The chapter further defines the business scope, specifying the target audience and value proposition, while emphasizing how the platform streamlines the decision-making process for prospective students. Additionally, it details the tools and technologies employed in the development process, including Python, Laravel, React.js, and MySQL. The work breakdown structure is also presented, offering a clear roadmap of the project phases and estimated timelines for completion. This chapter serves as a foundational guide to understanding the project's vision, technical approach, and anticipated outcomes.

Chapter 2

2. Requirement Specification and Analysis

Requirements Analysis is the method of characterizing the desires of the users for an application that must be built or adjusted. Requirements analysis involves all the tasks that are conducted to identify the needs of different stakeholders. Therefore, requirements analysis means to analyze, document, validate, and manage software or system requirements. This chapter includes functional and non-functional requirements use case diagrams, use case descriptions, and system sequence diagrams for the requirements selected/revised in the current iteration.

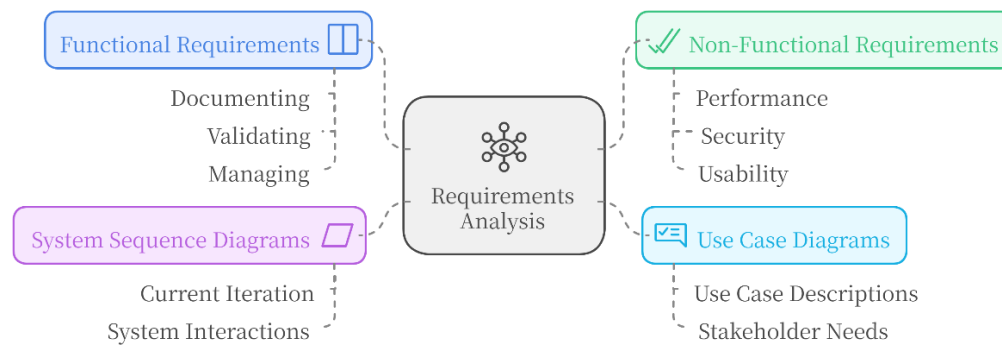


Figure 9 Requirement Analysis

2.1 Functional Requirements

Functional requirements define the functionalities of a system or its components. Functional requirements may be calculations, technical details, data manipulation and processing, and other specific functionality that define what a system is supposed to accomplish.

Sr. No	Functional Requirements	Type	Status
1	Users shall have the ability to register within the system.	Core	Completed
2	Users shall be able to log into the system securely.	Core	Completed
3	Users shall have the functionality to search for universities by name.	Core	Completed
4	Users shall be able to access and view detailed information about universities.	Core	Completed
5	Users shall have the capability to compare multiple universities	Core	Completed
6	Users shall be able to communicate via chat and upload documents within the system.	Core	Completed
7	Admin should be able to approve registrations	Core	Completer
8	Admin should be able to sign into the system	Core	Completed
9	Admin should be able to manage chats	Core	Completed

2.2 Non-Functional Requirements

Non-functional requirements cover all the remaining requirements, which are not covered by the functional requirements. They specify criteria that judge the operation of a system, rather than specific behaviors, for example: “Modified data in a database [3] shall be updated for all users accessing it within 2 seconds”. Some typical non-functional requirements [4] include performance, Data Integrity, Error Handling, availability, reliability, maintainability, usability, and security. Functional requirements describe what the system should do while non-functional requirements describe how the system works.

Sr. No	Non-Functional Requirements	Category
1	The system shall respond to user actions.	Performance
2	The system shall display error messages that are clear and helpful, with a readability score at or above grade level 8.	Usability
3	The web scraper shall be capable of handling changes in university website structures without requiring manual intervention (dynamic adaptability).	Reliability
4	The system shall ensure that the data scraped by the web scraper is up-to-date and accurate.	Reliability
5	The system shall support modifications and updates with an average effort of 4 hours or less per change.	Maintainability
6	The system code shall be documented, covering all major functions and modules.	Maintainability
7	The system shall provide secure user login, sign-up, and session management using the Laravel built-in authentication system, ensuring user data privacy.	Security

2.3 System Use Case Modeling

A Use Case depicts how actors will interact with the system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. The following use case diagrams will depict how our system works. [5]

2.3.1 Entire System Use-Case Modeling

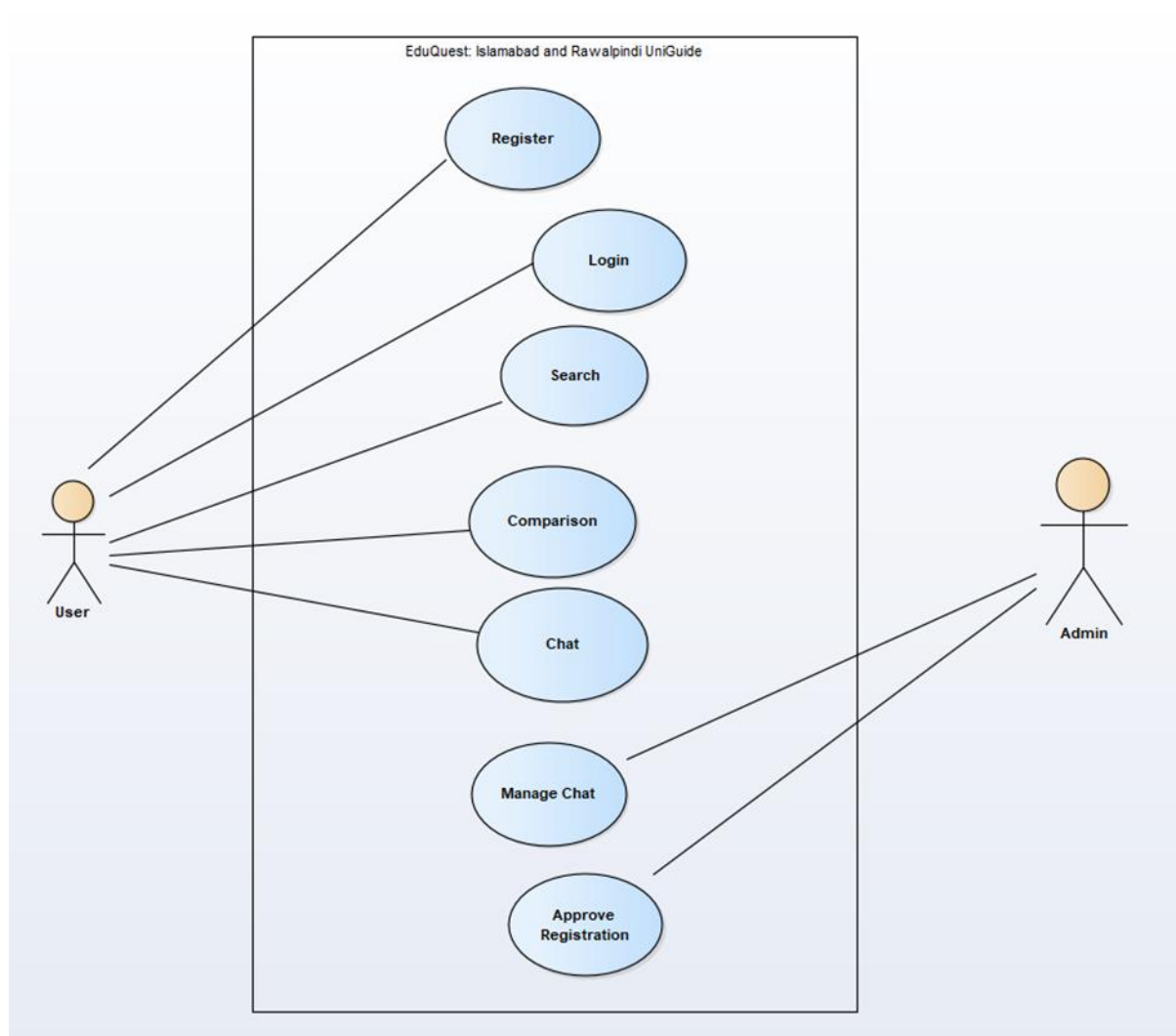


Figure 10 Use-case Modeling

2.4 Use case Description

2.4.1 Registration

Use Case ID:	Uc-01		
Use Case Name:	Registration		
Created By:	Gul Muhammad	Last Updated By:	Gul Muhammad
Date Created:	04/24/2024	Last Revision Date:	04/24/2024
Actors:	User		
Description:	The user registration process is to enable individuals to create an account within a system, platform, or application, granting them access to personalized features, content, and functionalities.		
Trigger:	Register button		
Stakeholders:	User		
Preconditions:	The system is accessible and the user has not registered before.		
Post conditions:	The user is registered with a unique identifier, gaining access to personalized features based on account type.		
Normal Flow:	User	System	
	1. Navigates to the registration page or clicks "Register."	1. The system presents a registration form for essential information.	
	2. Provides required information.	2. The system validates the entered data.	
Alternative Flows:	1. If the email already exists, prompt the user to log in or recover the password. 2. If errors occur (e.g., invalid email, weak password), provide error messages and correction guidance.		
Exceptions:	In case of system failures or technical issues, handle system failures gracefully, providing instructions to proceed.		

2.4.2 User Login

Use Case ID:	Uc-02		
Use Case Name:	User Login		
Created By:	Gul Muhammad	Last Updated By:	Gul Muhammad
Date Created:	04/24/2024	Last Revision Date:	04/24/2024
Actors:	User		
Description:	The user login process is to authenticate and authorize registered users, granting them access to their personalized accounts, features, and content within the system.		
Trigger:	Login button		
Stakeholders:	User		
Preconditions:	The system is operational and the user is registered.		
Post conditions:	The user is logged in and can access their account.		
Normal Flow:	User	System	
	1. The registered User navigates to the login page or clicks on the "Log In" button. 2. User enters login credentials (username/email and password)	1. The system presents a registration form with fields in response. 2. If successful, user is granted access and if unsuccessful, system prompts user to retry.	
Alternative Flows:	If the User forgets their password, the system may provide a "Forgot Password" link or functionality, guiding the user through a secure password recovery or reset process.		
Exceptions:	1. Invalid Credentials: User receives error message and retry option. 2. Account Lockout (Optional): After failed attempts, system locks account for security.		

2.4.3 Search Universities

Use Case ID:	Uc-03		
Use Case Name:	Search		
Created By:	Gul Muhammad	Last Updated By:	Gul Muhammad
Date Created:	04/24/2024	Last Revision Date:	04/24/2024
Actors:	User		
Description:	The search use case is to enable users to find relevant information, or resources efficiently and accurately within a system/platform, or application.		
Trigger:	Search button		
Stakeholders:	User		
Preconditions:	The System is operational, the user can access the search feature and the searchable data exists.		
Post-conditions:	The User has found relevant information or services based on their search query.		
Normal Flow:	User	System	
	<div>1. The user accesses the search through a designated search bar or page within the system.</div> <div>2. The user enters their search query into the search input field, expressing their intent or the specific information, product, or service they seek.</div>	<div>1. The system processes the user's query and initiates a search across content within its database.</div> <div>2. The System Displays relevant results based on the search query entered.</div>	
Alternative Flows:	<div>1. Advanced search: The User can refine search with filters.</div> <div>2. Autocomplete: The System suggests terms as user types.</div>		
Exceptions:	<div>1. No results found: System displays message and suggests alternatives.</div> <div>2. System error: System gracefully handles error and offers support options.</div>		

2.4.4 Compare Universities

Use Case ID:	Uc-04		
Use Case Name:	Compare Universities		
Created By:	Gul Muhammad	Last Updated By:	Gul Muhammad
Date Created:	04/24/2024	Last Revision Date:	04/24/2024
Actors:	User		
Description:	The Comparison use-case helps users to make a comparison between universities based on fee Structures, Departments Based, and Offered Program to make an informed decision.		
Trigger:	Compare University button		
Stakeholders:	User		
Preconditions:	The system is operational, the user is logged in and can access the account.		
Post-conditions:	The User can successfully compare two universities based on their fee structure, Department Based and Offered program.		
Normal Flow:	User	System	
	1. The user navigates through the Homepage and clicks on the Compare button. 2. The User easily compares two universities based on the search made.	1. The system provides a comparison Page with given search fields in response. 2. The system shows the universities related to the interest.	
Alternative Flows:	1. Invalid university search: System suggests alternatives. 2. Autocomplete suggestions: System assists with search terms		
Exceptions:	1. No results found: The System displays a message and suggests alternatives. 2. System error: The System gracefully handles error and offers support options.		

2.4.5 Chat

Use Case ID:	Uc-05		
Use Case Name:	Chat		
Created By:	Gul Muhammad	Last Updated By:	Gul Muhammad
Date Created:	04/24/2024	Last Revision Date:	04/24/2024
Actors:	User		
Description:	The "Chat" use case is to facilitate real-time communication between users within a system, platform, or application.		
Trigger:	Chat button		
Stakeholders:	User		
Preconditions:	The User is logged into the system and user clicks the Chat button to start a new chat. A stable internet connection must be available.		
Post conditions:	The user receives messages sent by other Users in the chat.		
Normal Flow:	User	System	
	a. The user selects the option to start a new chat or joins an existing chat. b. The user types a message and sends it.	a. The system creates a new chat session. b. The user receives messages from other Users.	
Alternative Flows:	If the user is not authenticated, the system prompts the user to log in.		
Exceptions:	If there is an error in sending or receiving the chat, the system should display an appropriate error message and guide the user to try again.		

2.4.6 Manage Chat

Use Case ID:	Uc-06		
Use Case Name:	Manage Chat		
Created By:	Gul Muhammad	Last Updated By:	Gul Muhammad
Date Created:	04/24/2024	Last Revision Date:	04/24/2024
Actors:	Admin		
Description:	The "Manage Chat" use case is to facilitate and manage a real-time communication between users within a system, platform, or application		
Trigger:	Chat button		
Stakeholders:	Admin		
Preconditions:	The system is operational, the admin has logged into the system (if required).		
Post conditions:	The Admin manages the chat by Viewing and Deleting a Specific chat if required.		
Normal Flow:	Admin	System	
	a. The Admin views and accesses the users chat to make necessary	a. The System applies the Changes made by the Admin	
Alternative Flows:	1. User list management: Admin can add/remove users from group chats 2. Chat settings: Admin can adjust chat settings like user permissions.		
Exceptions:	1. Permission error: System informs admin if they lack permission for a specific action 2. Chat access error: System informs admin if the chat cannot be accessed due to technical issues. 3. Data retrieval error: System displays an error message if chat details cannot be retrieved.		

2.4.7 Approve Registration

Use Case ID:	Uc-07		
Use Case Name:	Approve Registration		
Created By:	Gul Muhammad	Last Updated By:	Gul Muhammad
Date Created:	04/24/2024	Last Revision Date:	04/24/2024
Actors:	Admin		
Description:	The "Approve Registration" use case is to enable administrators to oversee and manage user accounts within a system/platform.		
Trigger:	Approve registration button.		
Stakeholders:	Admin		
Preconditions:	The admin is logged into the system and the system is operational.		
Post conditions:	The admin has successfully approved the user’s account applied necessary changes, and the system reflects the updated status and details of the user account.		
Normal Flow:	Admin	System	
	a. The admin accesses the user management Option. b. The admin selects a specific user account for a review c. The admin approves or manages the accounts	a. The system displays a list of user accounts. b. The system displays the user’s details and information.	
Alternative Flows:	The admin is unable to make required changes due to a system error. The admin faces a database errors e.g. unable to fetch required data.		
Exceptions:	1. System error: Admin is informed and appropriate measures are taken. 2. Unauthorized access attempt: System denies access and provides error messages.		

2.5. System Sequence Diagram

2.5.1. Registration

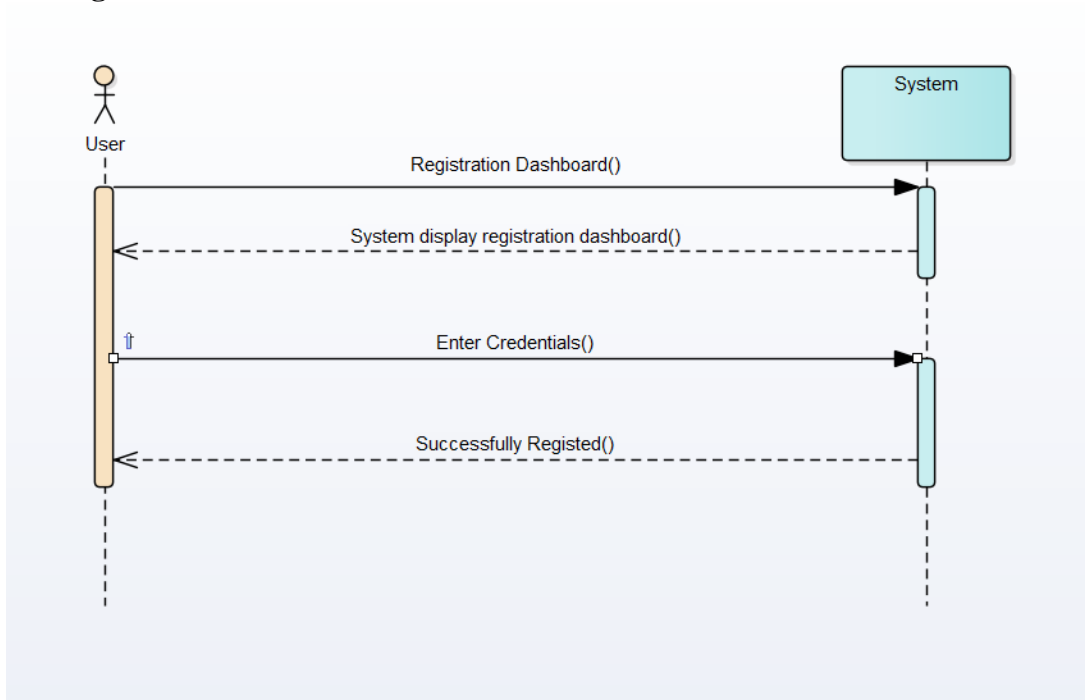


Figure 11 Registration Sequence Diagram

2.5.2. Login

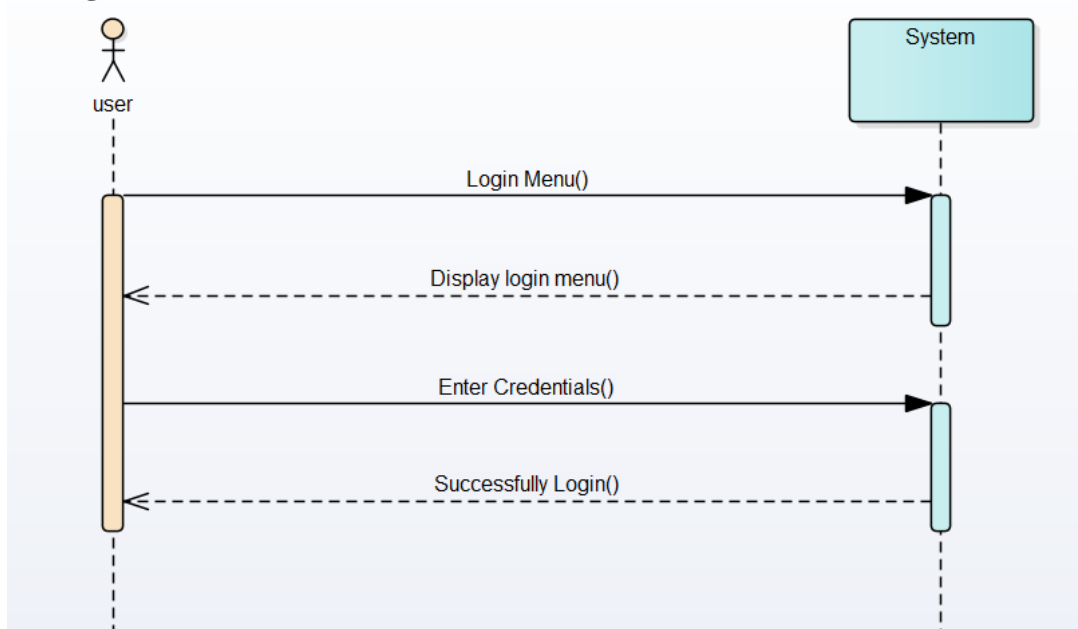


Figure 12 Login Sequence Diagram

2.5.3. Search

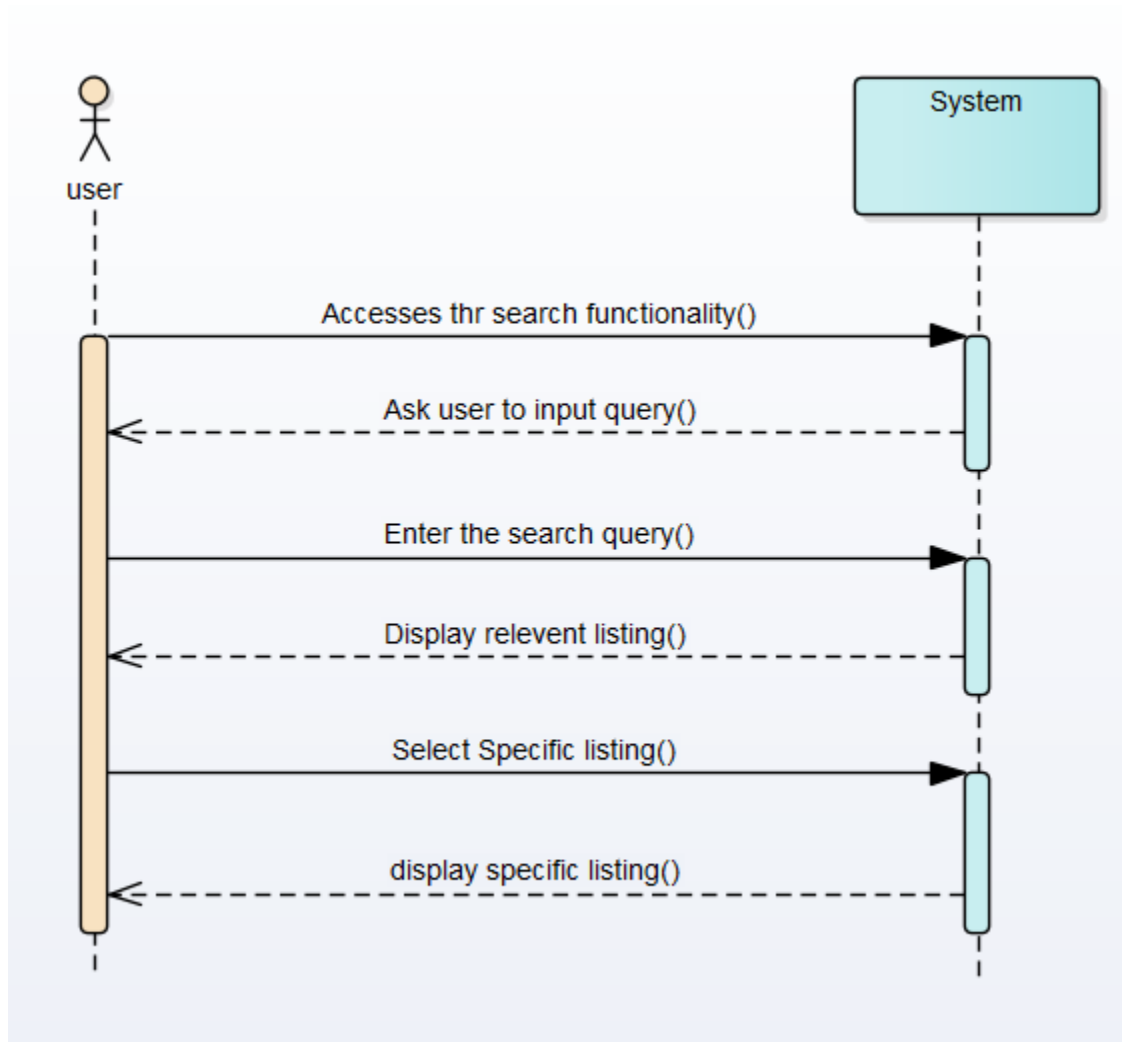


Figure 13 Search Sequence Diagram

2.5.4. Comparison

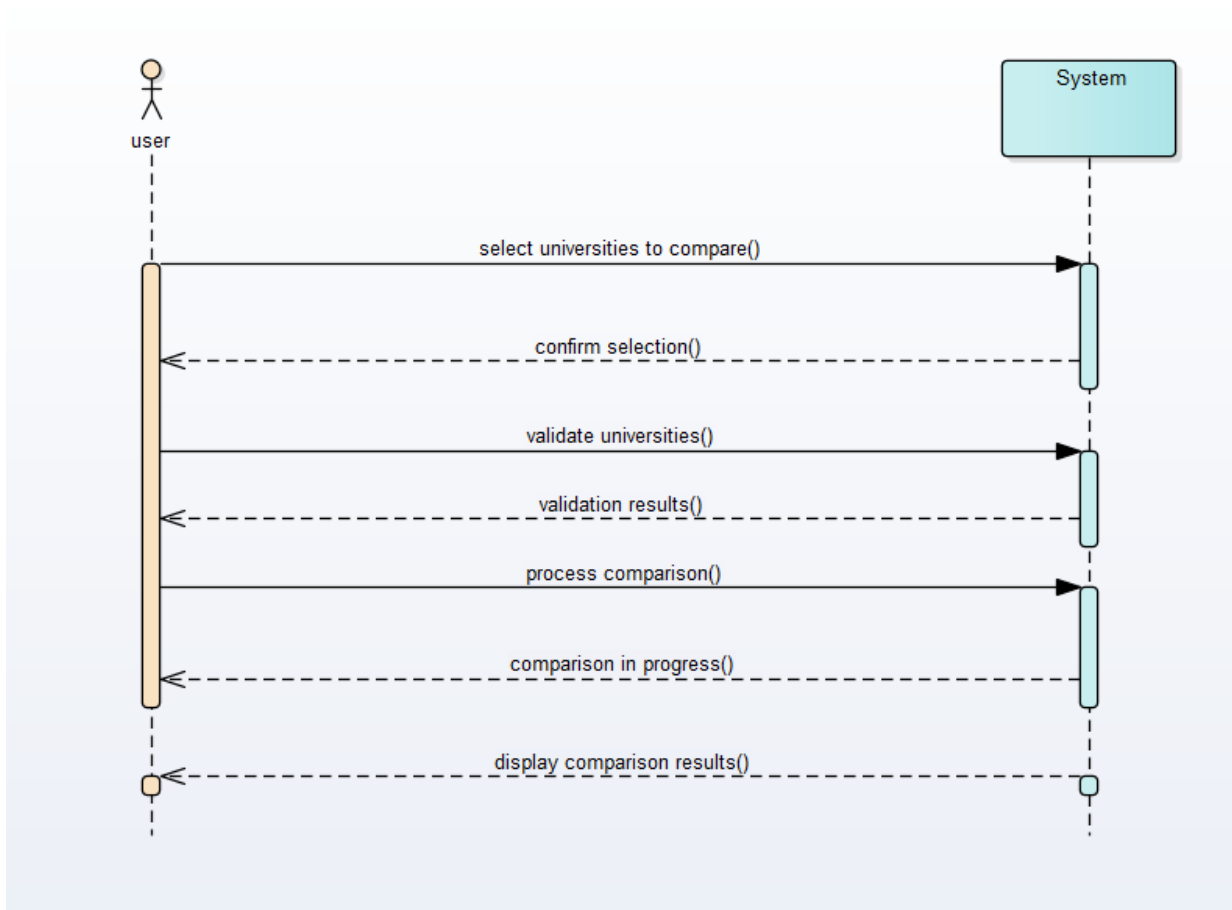


Figure 14 Comparison Sequence Diagram

2.5.5. Chat

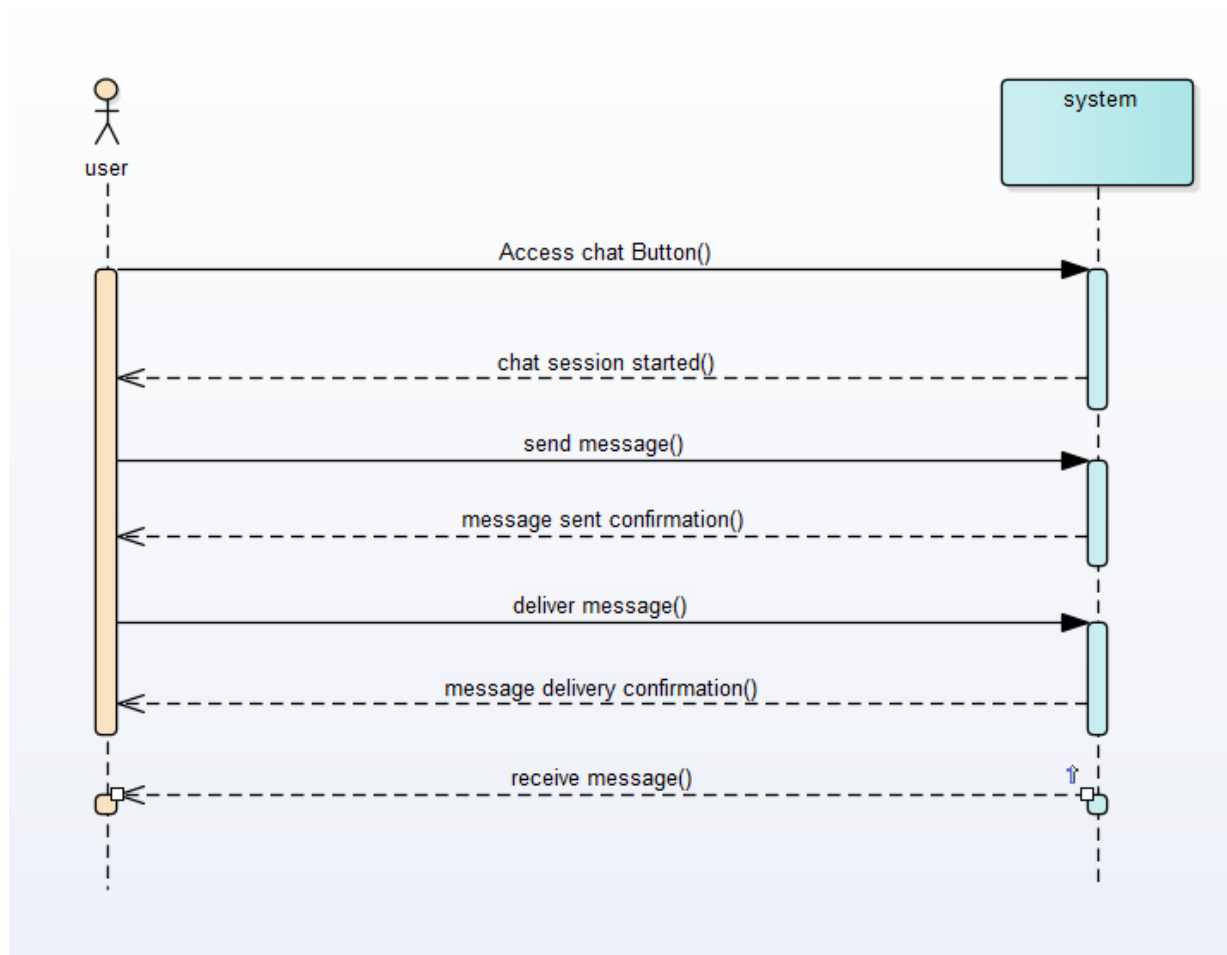


Figure 15 Chat Feature

2.5.6. Manage Chat

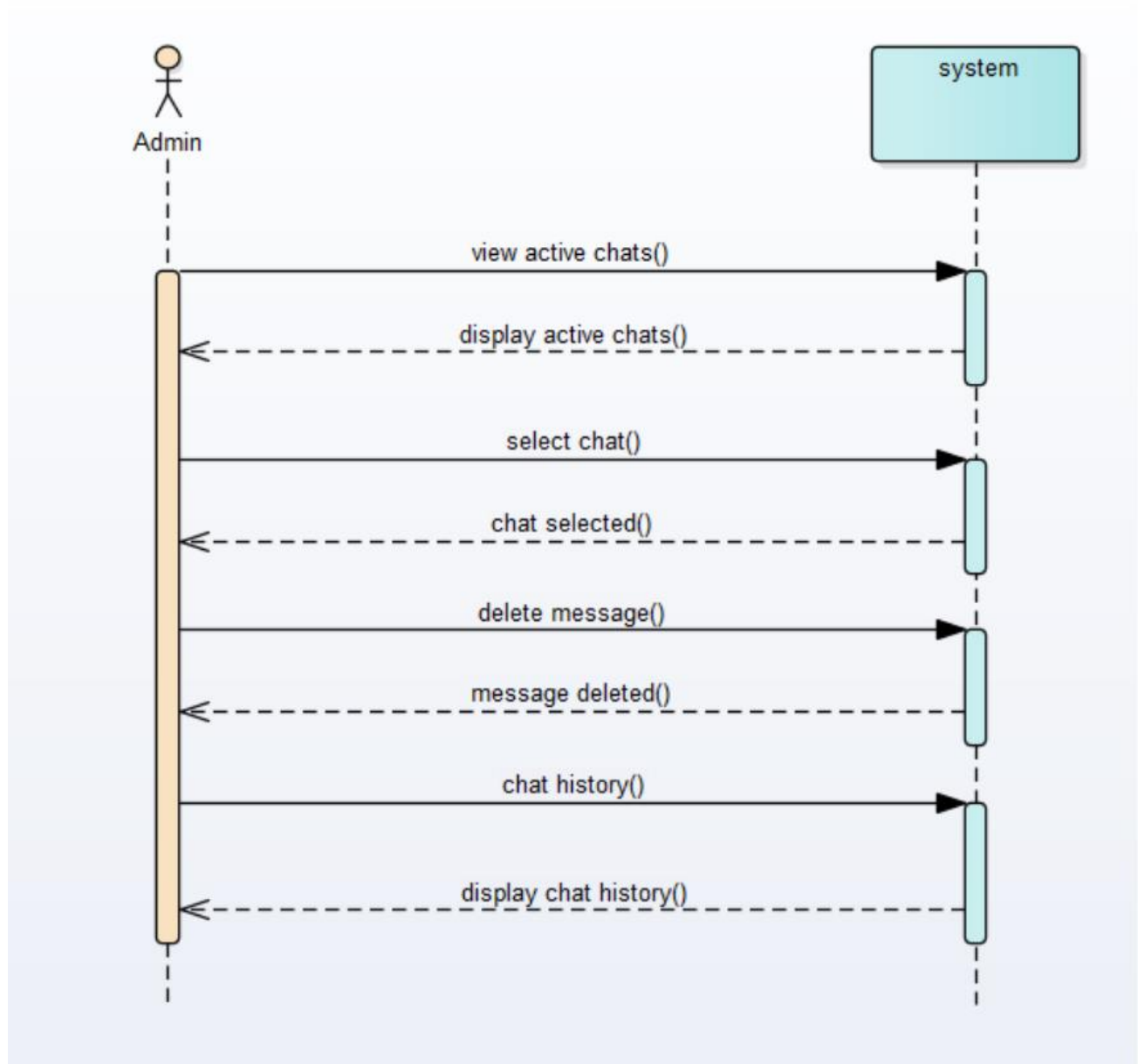


Figure 16 Manage Chat

2.5.7. Approve Registration

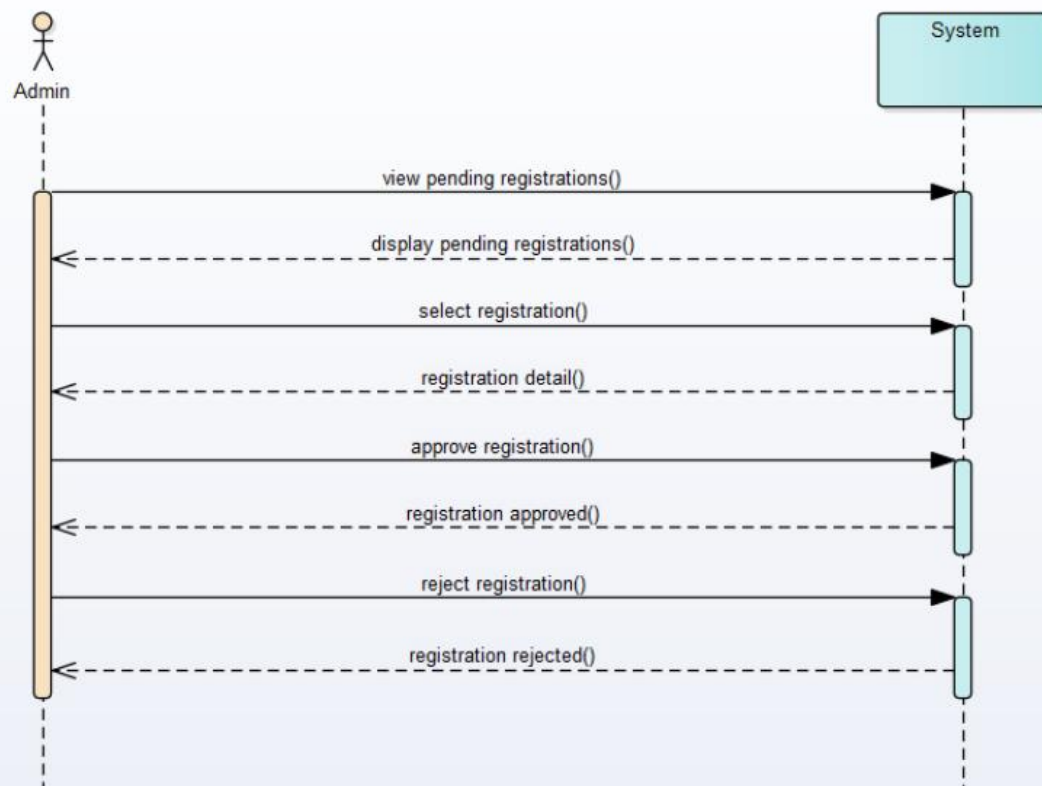


Figure 17 Approve Registration

2.6. Domain Model:

The Domain Model is your organized and structured knowledge of the problem. The Domain Model should represent the vocabulary and key concepts of the problem domain and it should identify the relationships among all of the entities within the scope of the domain.

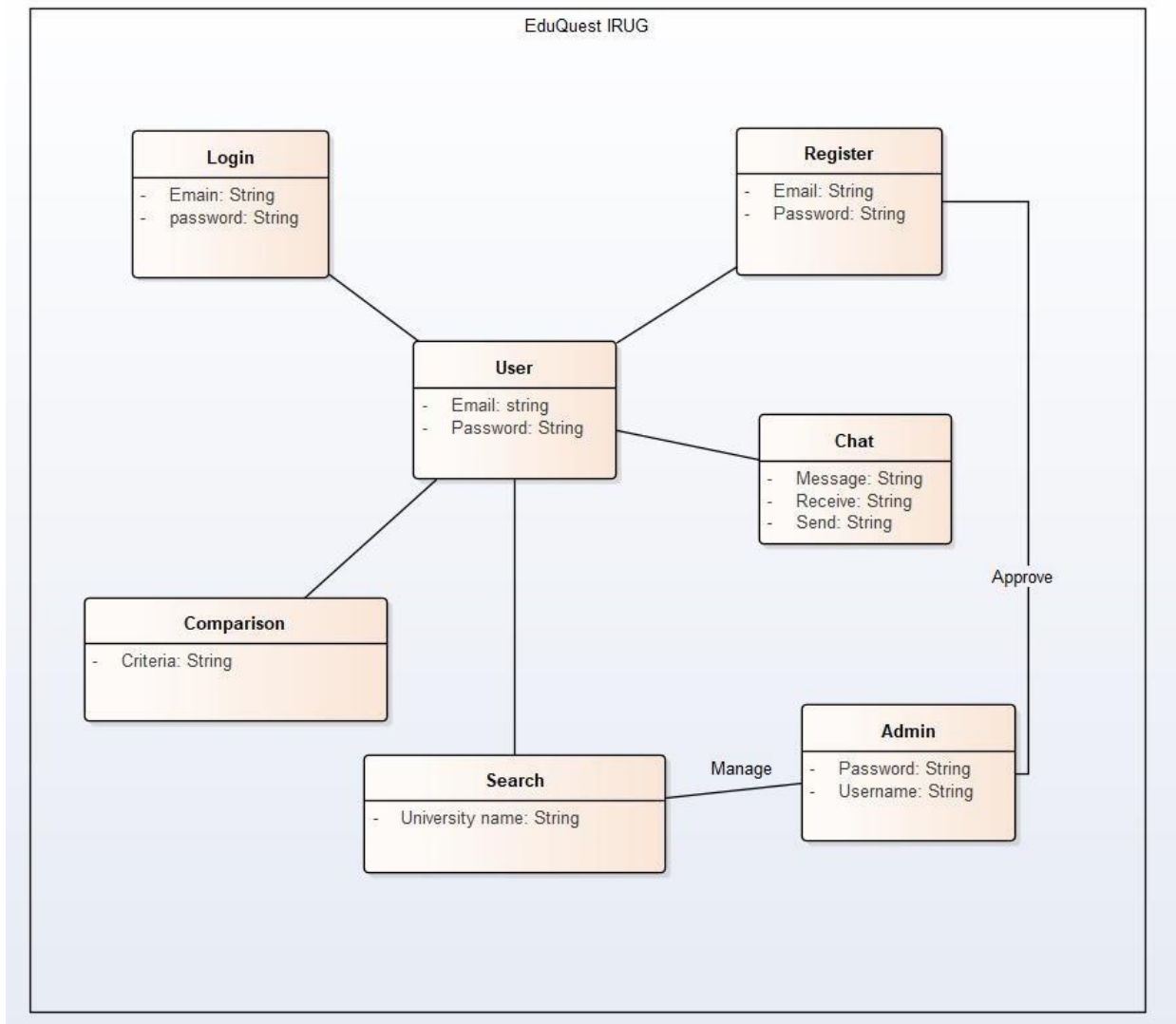


Figure 18 Domain Model

2.7. GUI:

2.7.1. Website Welcome Page

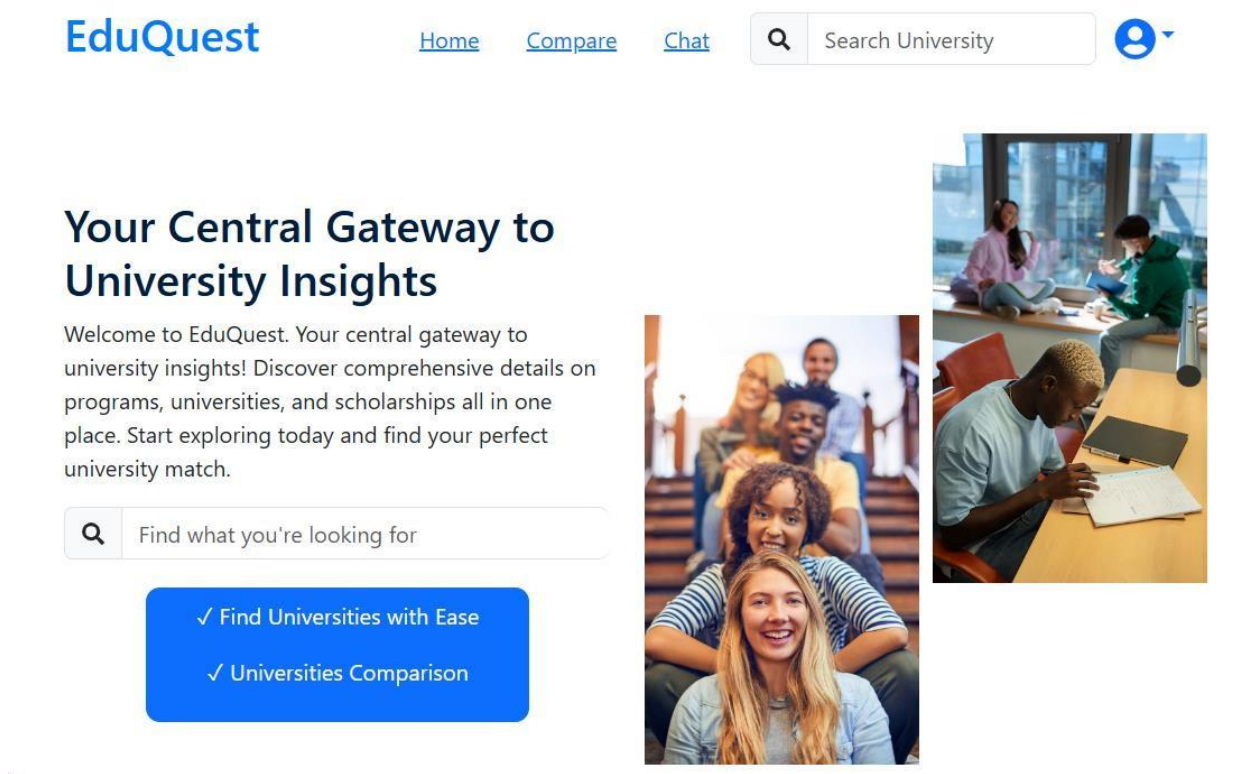


Figure 19 Welcome Page

2.7.2. Registration Page

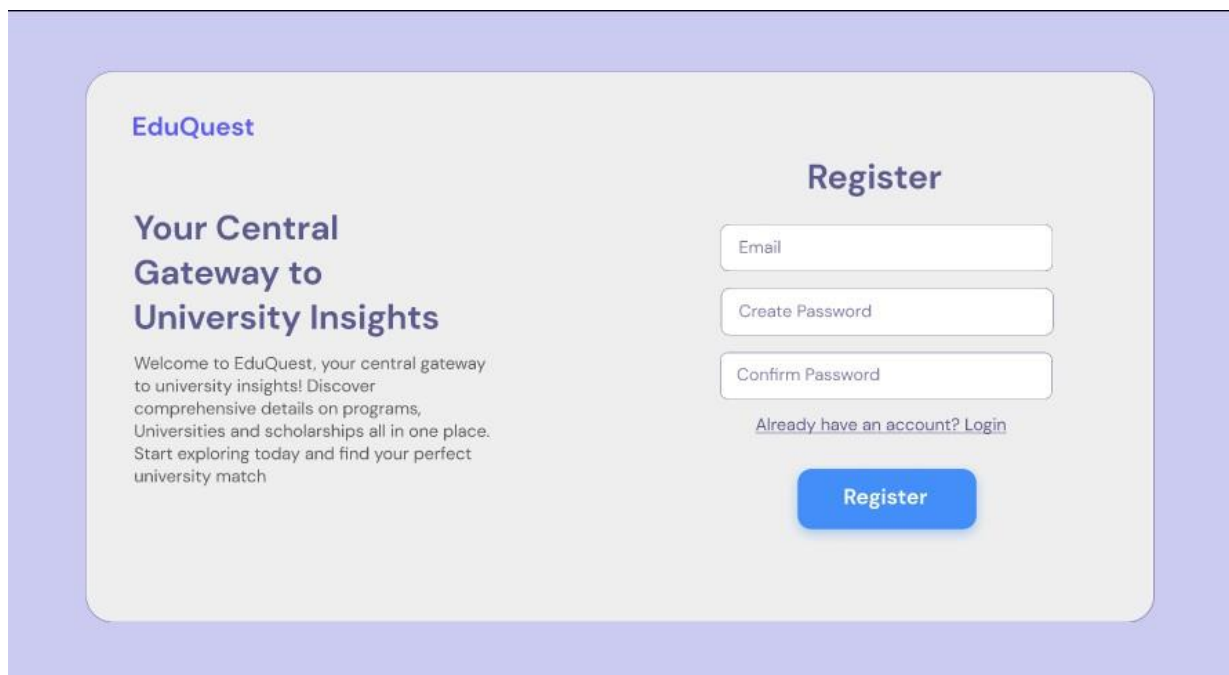
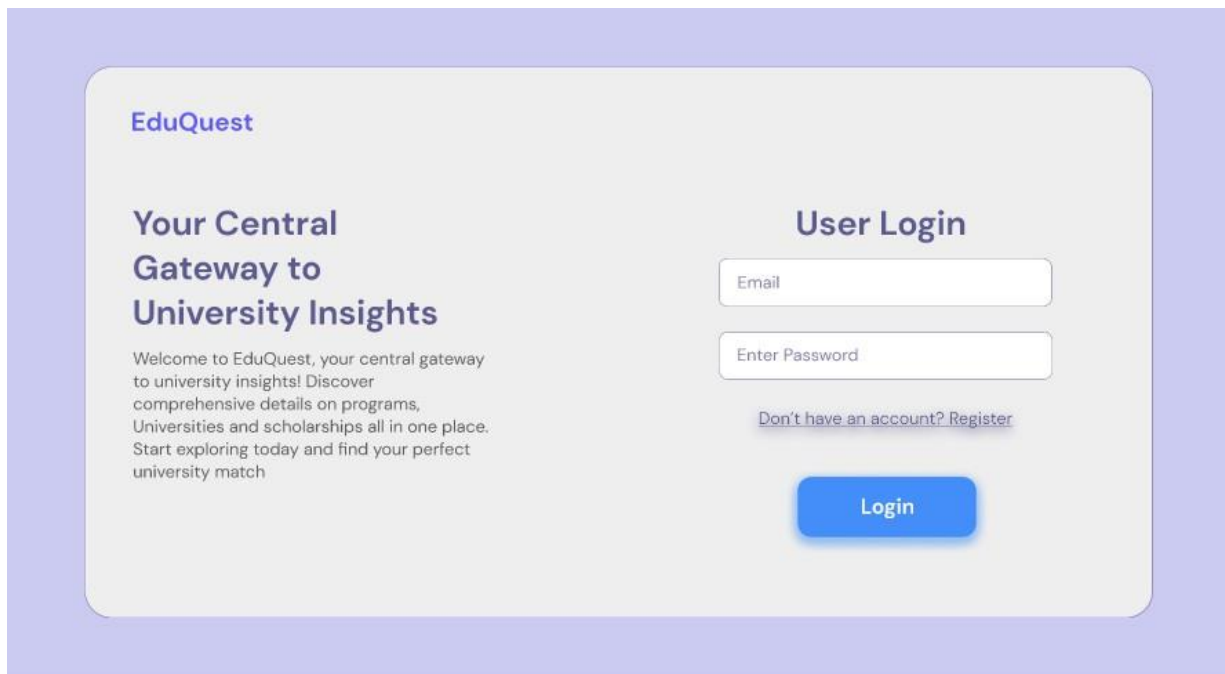


Figure 20 Registration Page

2.7.3. Login Page



The login page features a light purple background. On the left, the 'EduQuest' logo is at the top, followed by the heading 'Your Central Gateway to University Insights'. Below this is a welcome message: 'Welcome to EduQuest, your central gateway to university insights! Discover comprehensive details on programs, Universities and scholarships all in one place. Start exploring today and find your perfect university match'. On the right, the 'User Login' section contains two input fields for 'Email' and 'Enter Password', a link for 'Don't have an account? Register', and a blue 'Login' button.

Figure 21 Login Page

2.7.4. Dashboard

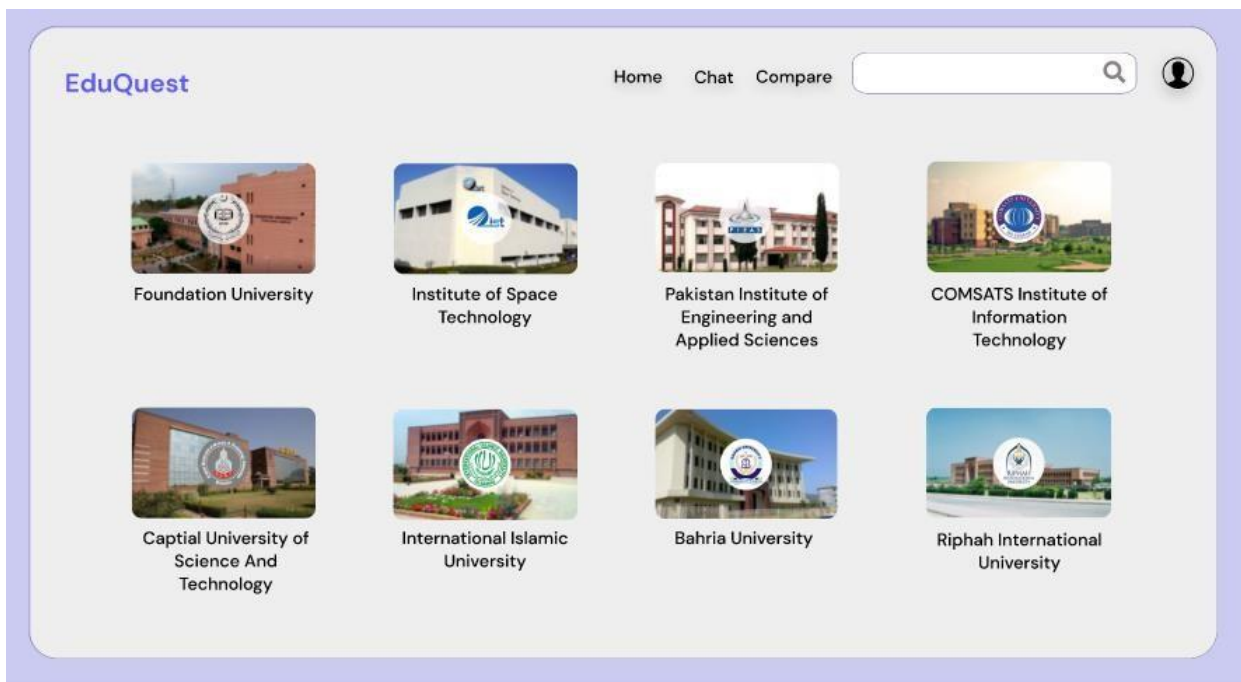


Figure 22 Dashboard

2.7.5. University Details

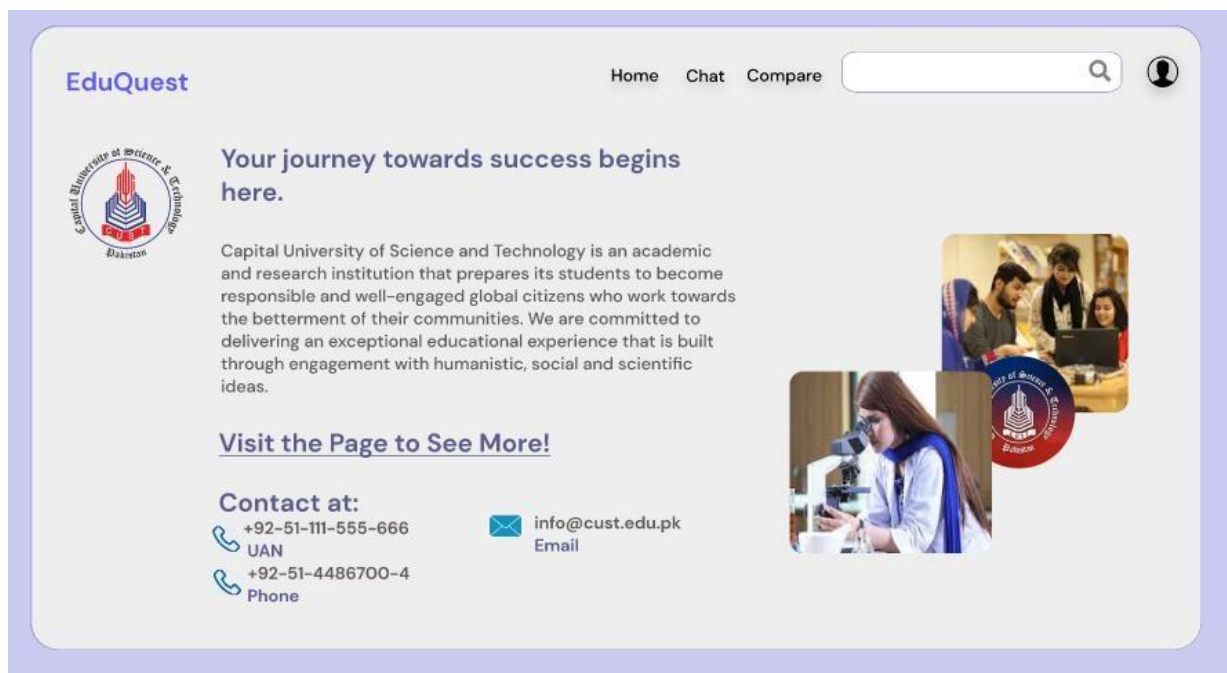


Figure 23 University Details

2.7.6. User Chats

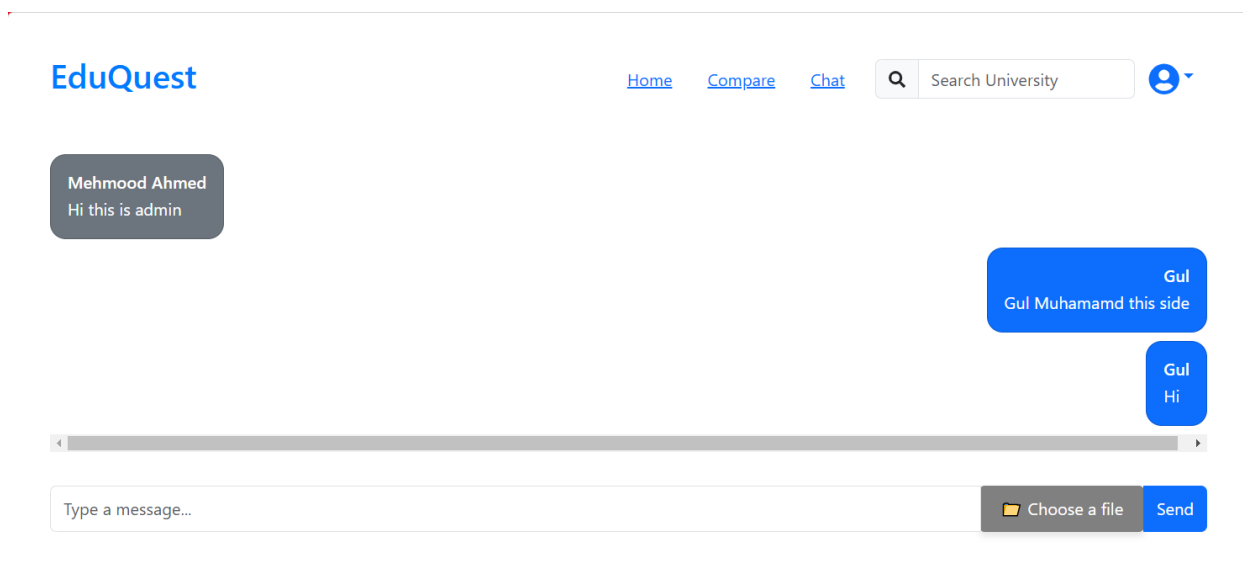
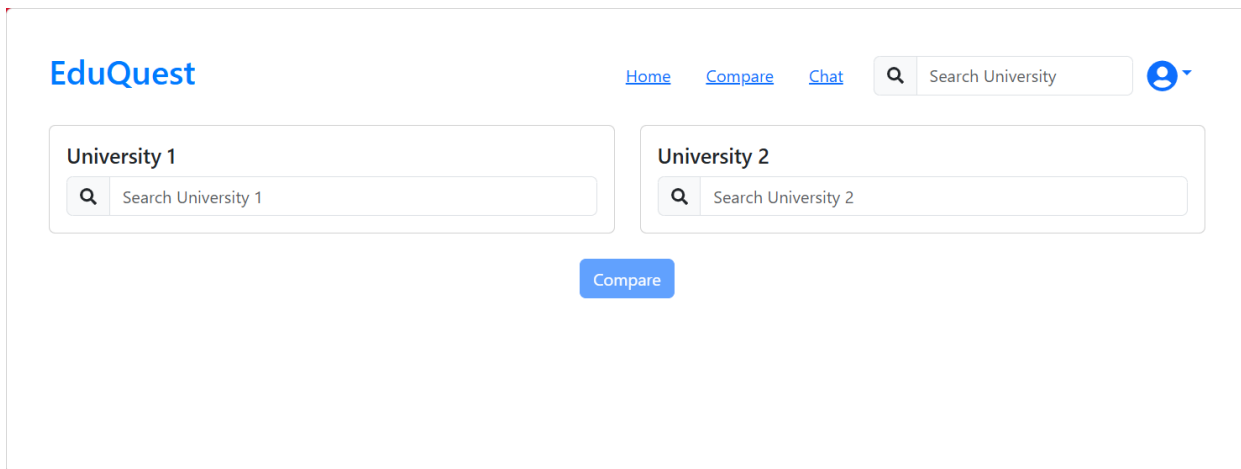


Figure 24 User Chats

2.7.7. Comparison Page



The screenshot shows the 'Comparison Page' of the EduQuest application. At the top left is the 'EduQuest' logo. To its right are navigation links: 'Home', 'Compare' (which is highlighted), and 'Chat'. Further right is a search bar labeled 'Search University' with a magnifying glass icon and a user profile icon. Below the navigation bar, there are two input fields for university selection. The first is labeled 'University 1' and contains a search bar with the placeholder text 'Search University 1'. The second is labeled 'University 2' and contains a search bar with the placeholder text 'Search University 2'. Below these two fields is a blue button labeled 'Compare'.

Figure 25 Comparison Page

2.7.8. Admin Dashboard

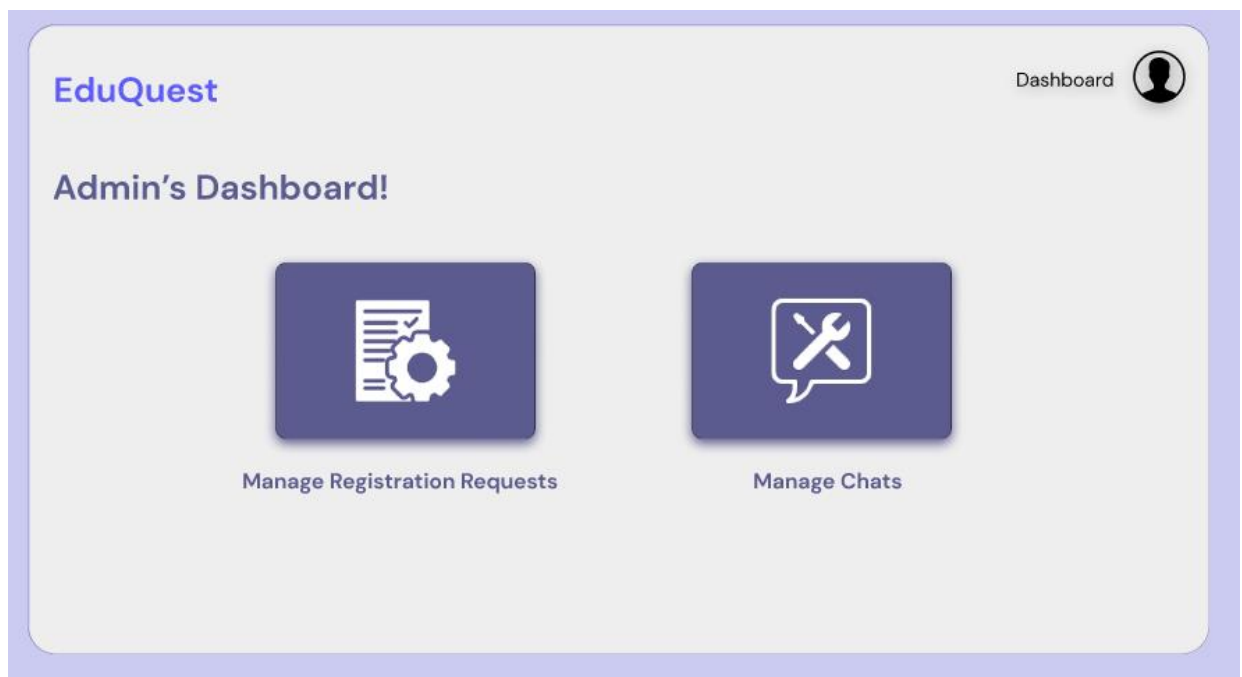


Figure 26 Admin Dashboard

2.7.9. Admin Chat Management

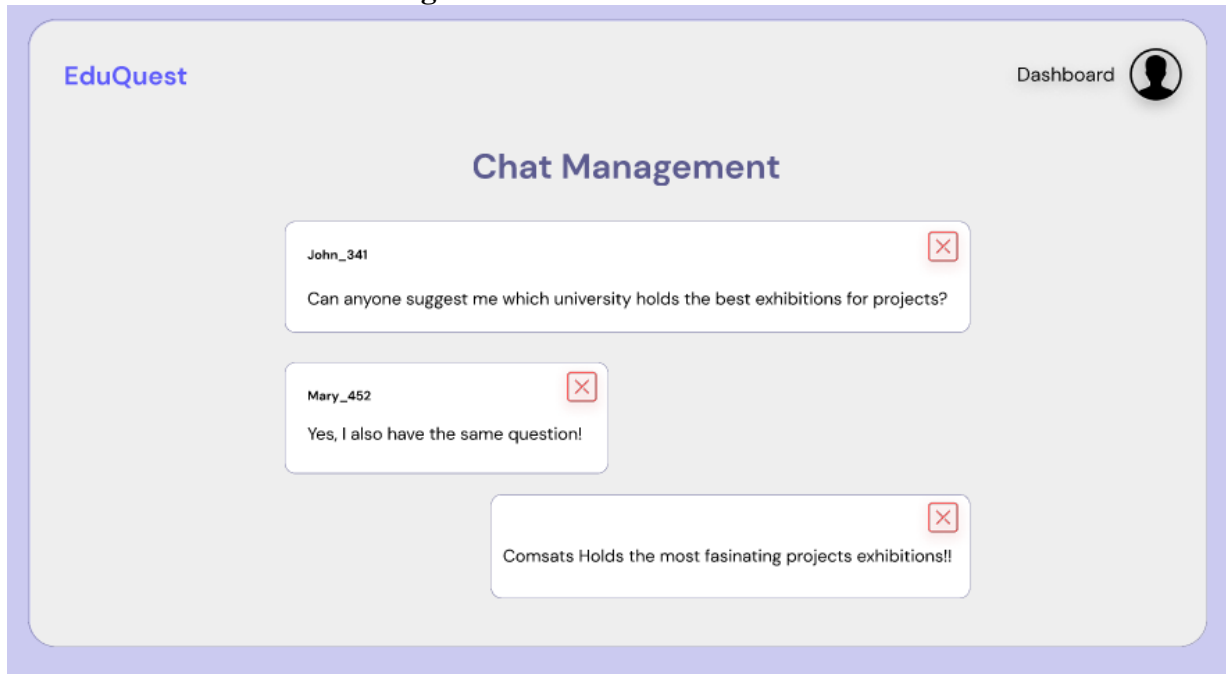


Figure 27 Admin chat management

2.7.10. Admin Approve Registrations



Figure 28 Admin Approve Registration

Chapter 3

3.1. System Diagram

This chapter aims to provide information complementary to the development phase. The project will fail without an adequate design that delivers the required function and quality attributes. However, communicating architecture to its Stakeholders are as important a job as creating it in the first place.

3.1.1. Software Architecture

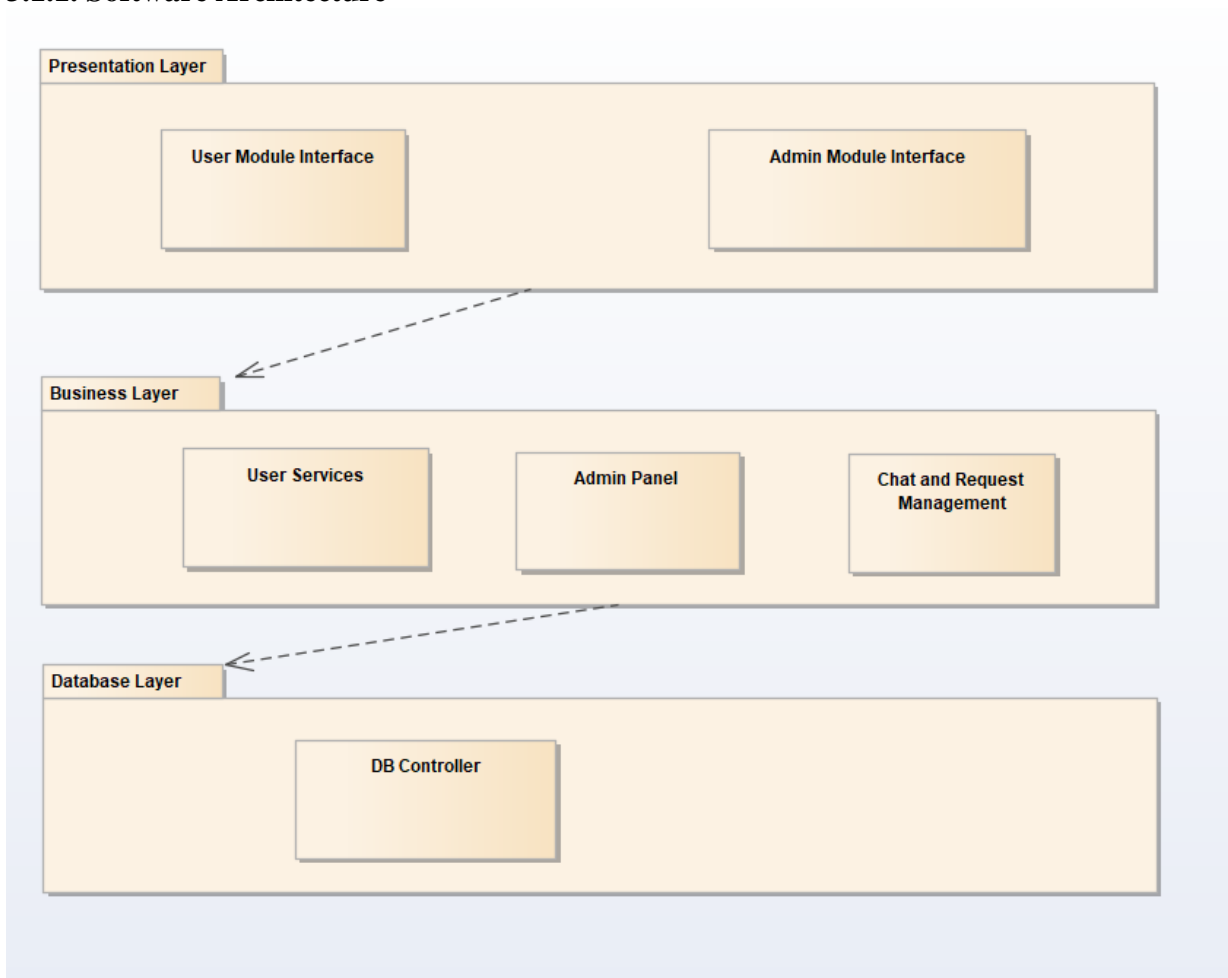


Figure 29 Software Architecture Diagram

3.2. Class Diagram

Class Diagram provides an overview of the target system by describing the objects and classes inside the system and the relationships between them. It provides a wide variety of usages; From modeling the domain-specific data structure to detailed design of the target system. [6]

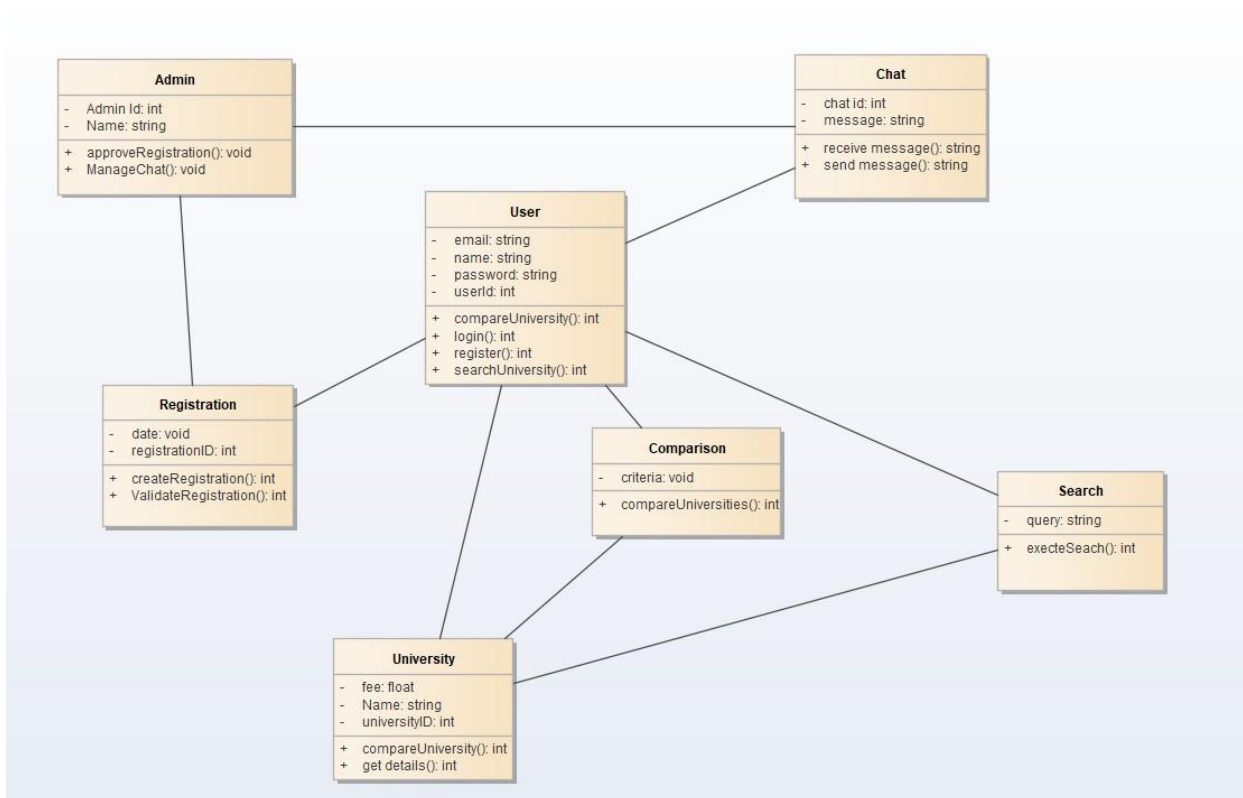


Figure 30 Class Diagram

3.3. Sequence Diagram

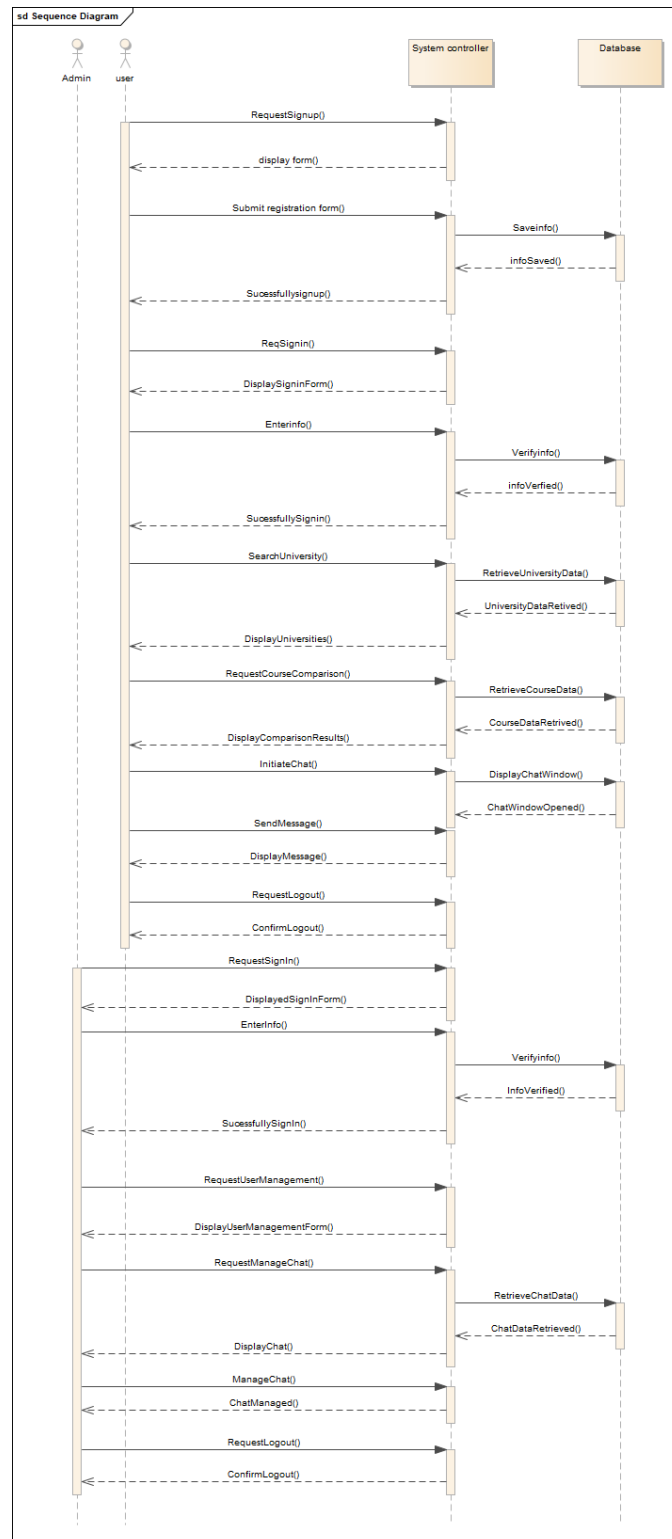


Figure 31 Sequence Diagram

3.4. Entity Relationship Diagram

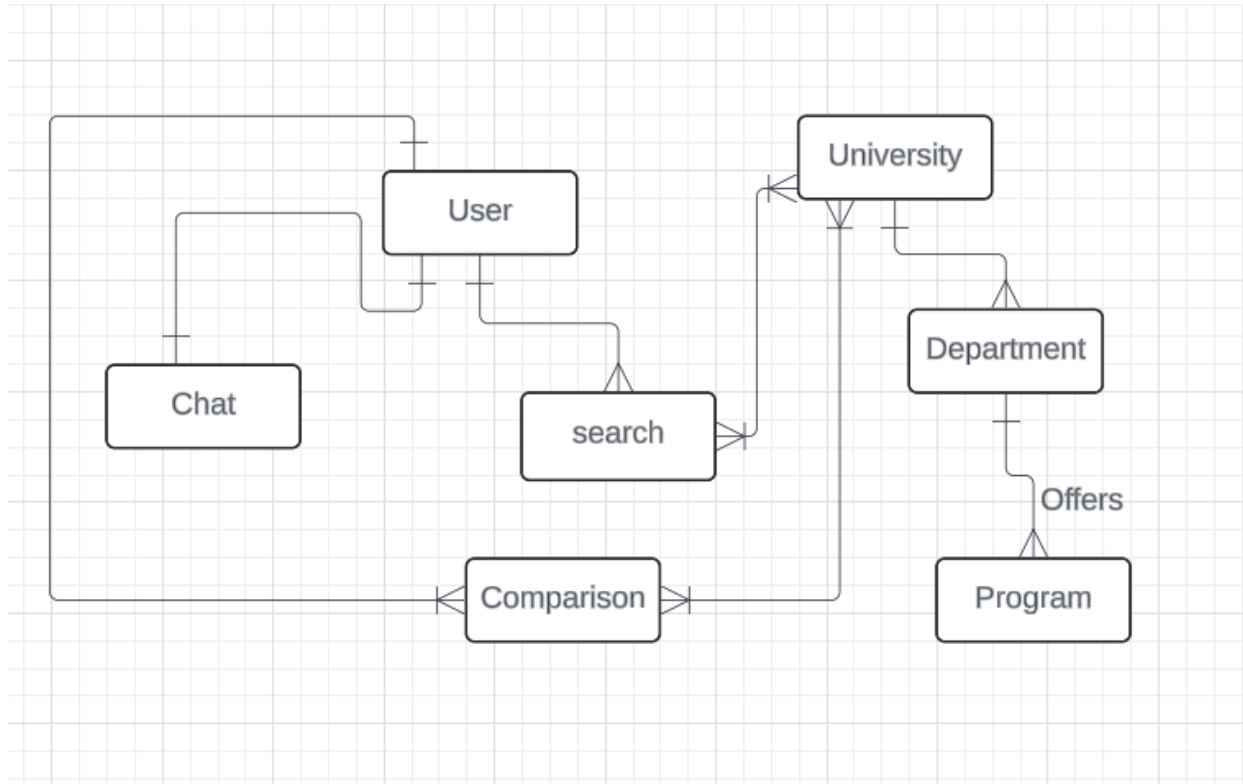


Figure 32 Entity Relationship Diagram

3.5. Database Schema

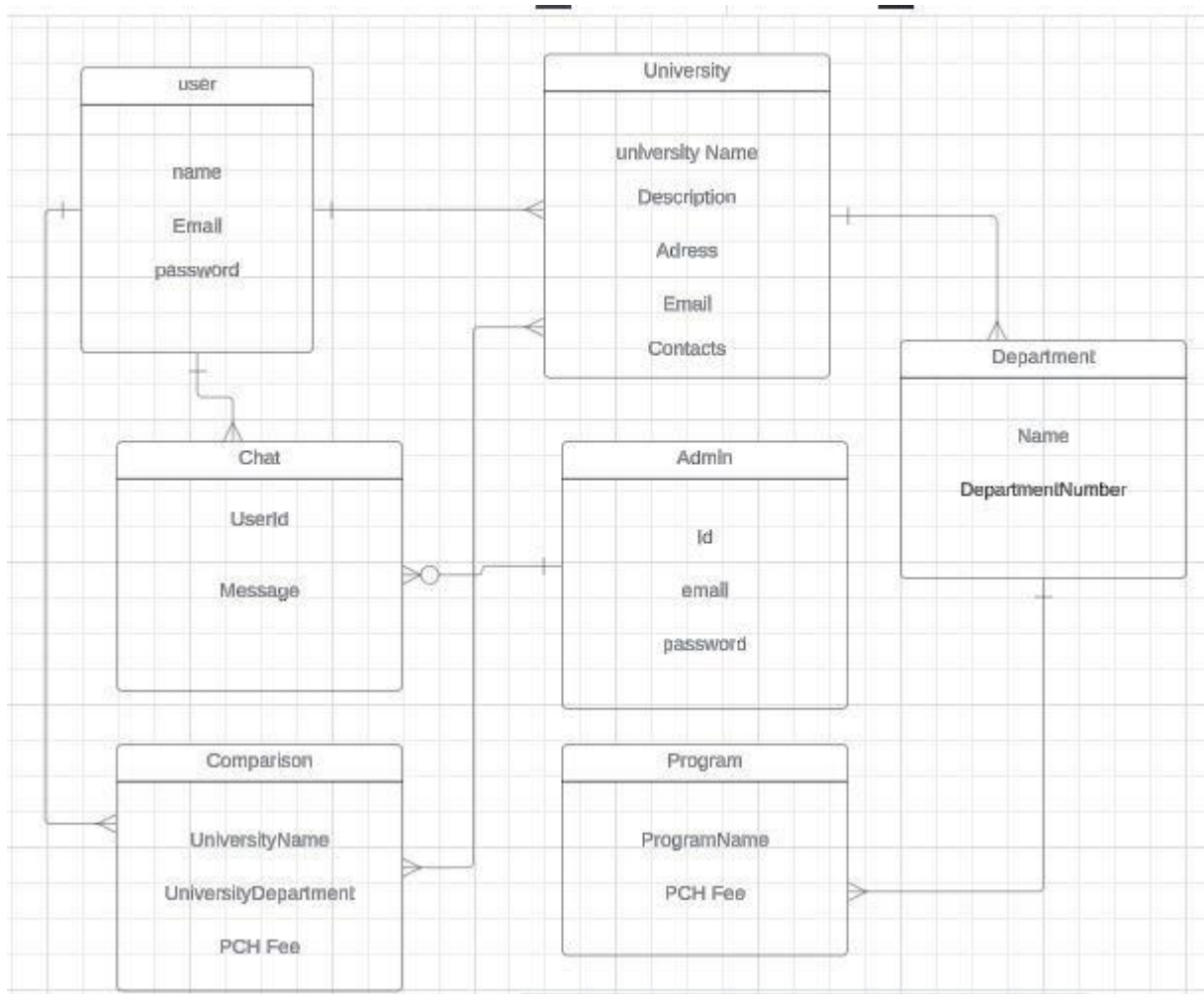


Figure 33 Database Schema

3.6. Software COTS

We are not using any type of Software COTS

Chapter 4

4. Software Development

The **Software Development** chapter describes the coding standards, environment setup, and development process for the system. It covers the implementation details of the system based on the design specifications, outlines the tools used, and discusses the issues encountered during development along with their resolutions.

4.1 Coding Standards

4.1.1 Indentation

Proper code indentation is essential for maintaining readability, understandability, and a clear hierarchy in the codebase. The system follows standard indentation practices, ensuring that nested code blocks are indented consistently with 2-4 spaces. This practice helps developers understand the structure of the code and improve maintainability.

4.1.2 Declaration

Frontend (React):

- React uses **functional components** for the front end, where components' names begin with uppercase letters, following standard React conventions.
- **State variables** are declared at the top of the component to keep the code organized and easy to follow. Variables are named descriptively using **camelCase**.
- Local variables in JavaScript are declared with `const`, `let`, or `var`, based on their mutability.

Backend (Laravel + Python Scrapers):

- In **Laravel**, backend code adheres to **snake case** naming conventions for variables, methods, and database field names, while **Pascal Case** is used for class names.
- **Python-based scrapers** are used for extracting data from university websites. Python variables and functions follow **PEP 8** guidelines, using snake case for variables and function names, and Capitalized Words for class names.

4.1.3. Statement Standards

Compound statements (such as if-else, for, try-catch) are written with clear indentation to show their hierarchical structure. **Braces** {} are used to enclose blocks of code, with each statement properly indented inside its respective block.

4.1.4. Naming Conventions

Proper naming convention rules are followed while implementation of this project which make programs more reasonable by making them simpler to peruse. While implementing this project, we have used words from Natural Language (English) to properly assign understandable names to classes, variables and methods. Such as Requests, Document Collection, Basic Information etc. instead of un-understandable names like my method, a1, b1 etc. Terminologies applicable to the domain of project are used. Implying that if user refers to Email as Registration Number then term Registration.

4.2. Development Environment

4.2.1. Frontend Development Environment

The frontend is developed using **React**, a JavaScript library for building dynamic user interfaces.

The development environment setup includes:

- **VS Code**: An open-source code editor used for writing and editing frontend code, with features like syntax highlighting and IntelliSense.
- **Command Prompt (CMD)**: Used for running commands to install necessary dependencies, such as React and other libraries.
- **Node Package Manager (NPM)**: Used for managing dependencies and packages for the frontend application.

Key frontend tools:

- **React.js** for building the user interface.
- **React Router** for handling navigation.

4.2.2. Backend Development Environment

The backend of the system consists of two main parts: the **Laravel** framework for API management and **Python** for web scraping.

- **VS Code**: Used for writing the PHP code for Laravel as well as Python scripts for scraping.
- **Command Prompt (CMD)**: For managing both Laravel and Python environments and running development servers.

- **Composer:** A n p m dependency manager used for managing Nodejs and backend libraries.
- **Python (for Scraping):** Python 3 is used for building scrapers to collect data from university websites.

Key backend tools:

- **JWT (JSON Web Tokens)** for secure user authentication.
- **MongoDB** for the database system.
- **Python** for the scraping functionality

4.3. Software Description

4.3.1. Frontend: React JS

The application's front end is developed with **React JS**, utilizing **Material UI** for styling and **Axioms** for API integration. It offers a seamless, user-friendly interface for features like university details, chat systems, and scholarship offerings. The design is responsive, accessible, and scalable, adhering to modern web standards.

Core Features

1. Routing and Navigation:

- Uses react-router-dom for managing routes such as /, /about, /contact, /scholarship, and dynamic pages like /Castles.

2. Material UI:

- Provides reusable components like Button, Typography, Grid, and Container for a polished, responsive UI.

3. Universities Page:

- Displays a responsive grid of university cards with hover effects and links to detailed pages (e.g., /Castles).

4. Footer:

- Consistent across pages with quick links to home, about, contact, and privacy sections.

5. API Integration:

- Fetches data dynamically using Axioms and use Effect. State variables store and display data such as university details.

6. **Search Functionality:**

- A search bar on the Universities page allows users to find specific universities.

7. **Responsive Design:**

- Material UI's grid system ensures adaptability across devices.

8. **Styling:**

- Modern color schemes and interactive hover effects enhance the user experience.

4.3.2. Expected User Experience

- **University Listings:** Displays a grid of university cards with names, logos, and links to detailed pages.
- **Dynamic Pages:** Displays detailed university data fetched from backend APIs.
- **Chat and Scholarship:** Accessed via the navigation bar for real-time interaction and scholarship information.
- **Loading States:** Displays "Loading..." while fetching data to ensure smooth transitions.

4.3.3. Backend: Web Scraper (Flask)

The backend is built using **Flask**, **Requests**, and **Beautiful Soup** to scrape public data from university websites. Data is exposed via RESTful APIs for integration with the frontend.

Key Features

1. **Data Extraction:**

- Scrapes university names, logos, fees, contact info, mission, and vision from official sites.
- Supports pages like "About Us," "Contact Us," and "Fee Structure."

2. **API Endpoints:**

- `/api/cuts-data`: University name and logo.
- `/api/about-us`: Vision and mission statements.
- `/api/contact-us`: Contact details.
- Additional endpoints for universities like FUUAST and Bahira.

3. **Error Handling:**

- Manages failed requests and missing data with helpful error messages.

4. **Scalability:**

- Modular design allows easy addition of scraping routes.

5. **CORS Support:**

- Enables cross-origin requests for seamless frontend-backend integration.

6. **Flexible JSON Data:**

- Outputs structured data for easy integration into other platforms.

Key Features

1. **User Registration:**

- Validates inputs (e.g., unique emails), supports profile images and stores user data.

2. **User Login:**

- Validates credentials and generates Bearer tokens for secure API access.

3. **Error Handling:**

- Validation errors return 422 and authentication errors return 401.

4. **Security:**

- Token-based authentication using Laravel Sanctum.
- Passwords are ideally hashed (currently stored as plain text for simplicity).

Technologies Used

- **Frontend:** React JS, Material UI, Axioms, React Router.
- **Backend:** Flask, Beautiful Soup, Requests, lamp, Flask-CORS.
- **Authentication:** Laravel, Laravel Sanctum, Validator, Hash.

Potential Use Cases

- Aggregating university data for research or administrative purposes.
- User-friendly platforms for exploring university details and scholarships.
- Secure user authentication for APIs in mobile or web apps.

4.4.4 Code Snippets

React.js/Registration

```
import React,{use State} from 'react';
import axios from 'axios';
import { Button, TextField, Container, Box, Typography, InputAdornment,Alert } from '@mui/material';
import SearchIcon from '@mui/icons-material/Search';
import yourImage from './Img11.png'; // Adjust the path as needed
import Snackbar from '@mui/material/Snackbar';
import { useNavigate,BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';
const Footer = () => {
  return (
    <Box
      sx={ {
        width: '100%',
        backgroundColor: '#5e5bfb',
        color: 'white',
        padding: '20px 0',
        mt: 'auto',
        textAlign: 'center',
      }}
    >
      <Typography variant="h6" sx={{ mb: 1 }}>
        EduQuest
      </Typography>
      <Typography variant="body1" sx={{ mb: 1 }}>
        &copy; { new Date().getFullYear() } EduQuest. All Rights Reserved.
      </Typography>
      <Box sx={{ display: 'flex', justifyContent: 'center', gap: 2 }}>
        <Button component={Link} to="/" sx={{ color: 'white' }}>
          Home
        </Button>
        <Button component={Link} to="/about" sx={{ color: 'white' }}>
          About Us
        </Button>
        <Button component={Link} to="/contact" sx={{ color: 'white' }}>
          Contact
        </Button>
        <Button component={Link} to="/privacy" sx={{ color: 'white' }}>
          Privacy Policy
        </Button>
      </Box>
    </Box>
  );
};

const Register = () => {
  const [error, setError] = useState("");
  const navigate = useNavigate();
```

```

const [name, setName] = useState("");

const [email, setEmail] = useState("");
const [password, setPassword] = useState("");
const [message, setMessage] = useState("");

const [snackbarOpen, setSnackbarOpen] = useState(false); // Snackbar visibility
const [snackbarMessage, setSnackbarMessage] = useState(""); // Snackbar message
const [snackbarSeverity, setSnackbarSeverity] = useState('success'); // Snackbar type ('success' or 'error')
const handleRegister = async () => {
  try {
    const response = await axios.post(
      'http://127.0.0.1:8000/api/register',
      {
        name: name.trim(),
        email: email.trim(),
        password: password.trim(),
      },
      {
        headers: {
          Accept: 'application/json',
        },
      }
    );

    // If successful
    console.log('DONE', response.data);
    navigate('/registersuccess');
    setSnackbarSeverity('success');
    setSnackbarMessage('User registered successfully!');
    setSnackbarOpen(true);
  } catch (error) {
    // Handle validation and other errors
    if (error.response) {
      console.error('Error Response:', error.response.data);

      if (error.response.data.errors) {
        // Collect validation error messages
        const errors = Object.values(error.response.data.errors)
          .flat()
          .join(', ');
        setSnackbarMessage(`Validation Errors: ${errors}`);
      } else {
        setSnackbarMessage(`Error: ${error.response.data.message || 'An error occurred.'}`);
      }
    } else {
      console.error('Error Details:', error);
      setSnackbarMessage('An unexpected error occurred.');
    }

    setSnackbarSeverity('error');
  }
}

```

```
setSnackbarOpen(true); // Show Snackbar
```

```

    }
  };

  // Close Snackbar
  const handleCloseSnackbar = () => {
    setSnackbarOpen(false);
  };

  return (
    <
      <Container
        maxWidth={false} // Remove max width restriction
        sx={{
          minHeight: '100vh', // Set full viewport height
          backgroundColor: '#cbcaf0',
          display: 'flex',
          justifyContent: 'center',
          alignItems: 'center',
          padding: '50px', // Add some padding for spacing
        }}
      >
        <Box
          sx={{
            width: '100%', // Adjust width as needed
            minHeight: 'calc(100vh - 100px)', // Ensure it takes almost the full screen height, accounting for padding
            padding: '20px',
            backgroundColor: '#eeeeee',
            borderRadius: '16px',
            display: 'flex',
            flexDirection: 'column',
            alignItems: 'center',
            justifyContent: 'space-between',
          }}
        >
          <Box
            sx={{
              width: '100%',
              display: 'flex',
              justifyContent: 'space-between',
              alignItems: 'center',
              mb: 2
            }}
          >
            <Typography variant="h4" sx={{ color: '#5e5bfb',fontWeight:'bold' }}>
              EduQuest
            </Typography>
            <Box>

```



```

</Box>
</Box>
<Box
  sx={{
    display: 'flex',
    width: '100%',
    justifyContent: 'space-between',
    alignItems: 'center',
    padding: '150px',
  }}
>
  <Box sx={{ width: '50%' }}>
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold' }}>
      Your Central
    </Typography>
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold' }}>
      Gateway to
    </Typography>
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold' }}>
      University Insights
    </Typography>
    <Typography variant="subtitle1" sx={{ mb: 2 }}>
      Welcome to EduQuest, your central gateway to university insights! Discover comprehensive details
on programs, universities, and scholarships all in one place. Start exploring today and find your perfect
university match.
    </Typography>
  </Box>

  <Box
    component="form"
    sx={{
      display: 'flex',
      flexDirection: 'column',
      gap: 2,
      width: '40%', // Adjusted width of the form
      padding: '20px', // Added padding

      justifyContent: 'center',
      alignItems: 'center',
      paddingRight: '40px'
    }}
  >
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold', textAlign: 'center' }}>
      Register
    </Typography>
    <TextField
      variant="outlined"

```

```

        label="Name"
        type="Name"
        onChange={ (e) => setName(e.target.value) }
        fullWidth

        sx={{ '& .MuiOutlinedInput-input': { color: '#438ef8' }, '& .MuiOutlinedInput-root': {
backgroundColor: 'white', borderRadius: '8px' } }}
    />
    <TextField
        variant="outlined"
        label="Email"
        type="email"
        onChange={ (e) => setEmail(e.target.value) }
        fullWidth

        sx={{ '& .MuiOutlinedInput-input': { color: '#438ef8' }, '& .MuiOutlinedInput-root': {
backgroundColor: 'white', borderRadius: '8px' } }}
    />
    <TextField
        variant="outlined"
        label="Create Password"
        type="password"
        onChange={ (e) => setPassword(e.target.value) }
        fullWidth

        sx={{ '& .MuiOutlinedInput-input': { color: '#438ef8' }, '& .MuiOutlinedInput-root': {
backgroundColor: 'white', borderRadius: '8px' } }}
    />

    <Typography variant="body2">
        Already have an account? <a href="/login">Login</a>
    </Typography>
    <Button
        variant="contained"
        sx={{
            backgroundColor: '#438ef8',
            height: '50px',
            color: 'ffffff',
            boxShadow: '0px 0px 10px rgba(67, 142, 248, 0.8), 0px 0px 20px rgba(67, 142, 248, 0.6), 0px 0px
30px rgba(67, 142, 248, 0.4)',
            '&:hover': {
                backgroundColor: '#438ef8',
            },
        }}

        onClick={handleRegister}
    >
        Register
    </Button>
</Box>

</Box>

```

```
    <Snackbar
      open={snackbarOpen}
      autoHideDuration={4000} // Automatically hide after 4 seconds
      onClose={handleCloseSnackbar}
    >
      <Alert
        onClose={handleCloseSnackbar}
        severity={snackbarSeverity} // 'success' or 'error'
        sx={{ width: '100%' }}
      >
        {snackbarMessage}
      </Alert>
    </Snackbar>
    <Footer />
  </Box>
</Container>
</>
);
};

export default Register;
```

React.js/Login.js

```
import React, { useState, useContext } from 'react';
import { TextField, Container, Box, Typography, InputAdornment } from '@mui/material';
import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';
import axios from 'axios';
import { Snackbar, Alert, Button } from '@mui/material';
import { useNavigate } from 'react-router-dom';
import { AuthContext } from '../AuthContext'; // Import the AuthContext
import SearchIcon from '@mui/icons-material/Search';

const Footer = () => {
  return (
    <Box
      sx={{
        width: '100%',
        backgroundColor: '#5e5bfb',
        color: 'white',
        padding: '20px 0',
        mt: 'auto',
        textAlign: 'center',
      }}
    >
      <Typography variant="h6" sx={{ mb: 1 }}>
        EduQuest
      </Typography>
      <Typography variant="body1" sx={{ mb: 1 }}>
        &copy; { new Date().getFullYear() } EduQuest. All Rights Reserved.
      </Typography>
      <Box sx={{ display: 'flex', justifyContent: 'center', gap: 2 }}>
        <Button component={Link} to="/" sx={{ color: 'white' }}>
          Home
        </Button>
        <Button component={Link} to="/about" sx={{ color: 'white' }}>
          About Us
        </Button>
        <Button component={Link} to="/contact" sx={{ color: 'white' }}>
          Contact
        </Button>
        <Button component={Link} to="/privacy" sx={{ color: 'white' }}>
          Privacy Policy
        </Button>
      </Box>
    </Box>
  );
};
```

```

const Login = () => {
  const [snackbar, setSnackbar] = useState({ open: false, message: '' });
  const navigate = useNavigate();

  const { saveToken } = useContext(AuthContext); // Get the saveToken function from context
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [error, setError] = useState('');

  const handleLogin = async () => {
    if (!email || !password) {
      setSnackbar({ open: true, message: 'Email and password are required.' });
      return;
    }

    try {
      const response = await axios.post(
        'http://127.0.0.1:8000/api/login',
        { email, password },
        {
          headers: {
            Accept: 'application/json', // Accept JSON responses
          },
        }
      );

      if (response.status === 200) {
        const token = response.data.token;
        localStorage.setItem('token', token); // Save token
        saveToken(token); // Use AuthContext
        navigate('/Unis'); // Navigate to another page
      } else {
        setSnackbar({ open: true, message: 'Invalid credentials. Please try again.' });
      }
    } catch (err) {
      console.error(err);
      setSnackbar({ open: true, message: 'Login failed. Please check your credentials or try again later.' });
    }
  };

  return (
    <
    <Container
      maxWidth={false} // Remove max width restriction
      sx={{
        minHeight: '100vh', // Set full viewport height
        backgroundColor: '#cbcaf0',
        display: 'flex',
        justifyContent: 'center',
        alignItems: 'center',
        padding: '50px', // Add some padding for spacing
      }}
    >

```

```

    }}
  >
  <Box
    sx={{
      width: '100%', // Adjust width as needed
      minHeight: 'calc(100vh - 100px)', // Ensure it takes almost the full screen height, accounting for padding
      padding: '20px',
      backgroundColor: '#eeeeee',
      borderRadius: '16px',
      display: 'flex',
      flexDirection: 'column',
      alignItems: 'center',
      justifyContent: 'space-between',
    }}
  >
  <Box
    sx={{
      width: '100%',
      display: 'flex',
      justifyContent: 'space-between',
      alignItems: 'center',
      mb: 2,
    }}
  >
  <Typography variant="h4" sx={{ color: '#5e5bfb', fontWeight: 'bold' }}>
    EduQuest
  </Typography>
  <Box>

    <Button
      variant="contained"
      sx={{
        mr: 1,
        backgroundColor: 'white',
        color: '#5c5b8f',
        fontWeight: 'bold',
        boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
        '&:hover': {
          backgroundColor: 'white',
        },
      }}
      component={Link}
      to="/login"
    >
      Login
    </Button>
    <Button
      variant="contained"
      sx={{

```

```

        backgroundColor: '#4e94f7',
        color: 'ffffff',
        fontWeight: 'bold',
        boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
        '&:hover': {
            backgroundColor: '#4e94f7',
        },
    }}

    component={Link}
    to="/register"

  >
    Register
  </Button>
</Box>
</Box>
<Box
  sx={{
    display: 'flex',
    width: '100%',
    justifyContent: 'space-between',
    alignItems: 'center',
    paddingTop: '150px',
  }}
>
  <Box sx={{ width: '50%' }}>
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold' }}>
      Your Central
    </Typography>
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold' }}>
      Gateway to
    </Typography>
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold' }}>
      University Insights
    </Typography>
    <Typography variant="subtitle1" sx={{ mb: 2 }}>
      Welcome to EduQuest, your central gateway to university insights! Discover comprehensive details
      on programs, universities, and scholarships all in one place. Start exploring today and find your perfect
      university match.
    </Typography>
  </Box>
<Box
  component="form"
  sx={{
    display: 'flex',
    flexDirection: 'column',
    gap: 2,
    width: '40%', // Adjusted width of the form
    padding: '20px', // Added padding
    justifyContent: 'center',
  }}

```

```

        alignItems: 'center',
        paddingRight: '40px',
      }}
    >
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold', textAlign: 'center' }}>
      Login
    </Typography>

    <TextField
      variant="outlined"
      label="Email"
      type="email"
      onChange={ (e) => setEmail(e.target.value) }
      fullWidth
      sx={{ '& .MuiOutlinedInput-input': { color: '#438ef8' }, '& .MuiOutlinedInput-root': {
background-color: 'white', borderRadius: '8px' } }}
    />
    <TextField
      variant="outlined"
      label="Password"
      type="password"
      onChange={ (e) => setPassword(e.target.value) }
      fullWidth
      sx={{ '& .MuiOutlinedInput-input': { color: '#438ef8' }, '& .MuiOutlinedInput-root': {
background-color: 'white', borderRadius: '8px' } }}
    />
    <Typography variant="body2">
      Don't have an account? <a href="/register">Register</a>
    </Typography>
    <Button
      variant="contained"
      sx={{
        backgroundColor: '#438ef8',
        height: '50px',
        color: 'ffffff',
        boxShadow: '0px 0px 10px rgba(67, 142, 248, 0.8), 0px 0px 20px rgba(67, 142, 248, 0.6), 0px 0px
30px rgba(67, 142, 248, 0.4)',
        '&:hover': {
          backgroundColor: '#438ef8',
        },
      }}
      onClick={handleLogin}
    >
      Login
    </Button>
  </Box>
</Box>
<Footer />

```



```

    </Box>
    <Snackbar
      open={snackbar.open}
      autoHideDuration={4000}
      onClose={() => setSnackbar({ open: false, message: " })}
      message={snackbar.message}
    />
  </Container>
</>
);
};

export default Login;

```

React.js/Regsuccess.js

```

import React from 'react';
import { Button, TextField, Container, Box, Typography, InputAdornment } from '@mui/material';
import SearchIcon from '@mui/icons-material/Search';
import { Link } from 'react-router-dom';
const Footer = () => {
  return (
    <Box
      sx={{
        width: '100%',
        backgroundColor: '#5e5bfb',
        color: 'white',
        padding: '20px 0',
        mt: 'auto',
        textAlign: 'center',
      }}
    >
      <Typography variant="h6" sx={{ mb: 1 }}>
        EduQuest
      </Typography>
      <Typography variant="body1" sx={{ mb: 1 }}>
        &copy; {new Date().getFullYear()} EduQuest. All Rights Reserved.
      </Typography>
      <Box sx={{ display: 'flex', justifyContent: 'center', gap: 2 }}>
        <Button component={Link} to="/" sx={{ color: 'white' }}>
          Home
        </Button>
        <Button component={Link} to="/about" sx={{ color: 'white' }}>
          About Us
        </Button>
        <Button component={Link} to="/contact" sx={{ color: 'white' }}>
          Contact

```

```

    </Button>
    <Button component={Link} to="/privacy" sx={{ color: 'white' }}>
      Privacy Policy
    </Button>
  </Box>
</Box>
);
};

const Regsuccess = () => {
  return (
    <Container
      maxWidth={false} // Remove max width restriction
      sx={{
        minHeight: '100vh', // Set full viewport height
        backgroundColor: '#cbcaf0',
        display: 'flex',
        justifyContent: 'center',
        alignItems: 'center',
        padding: '50px', // Add some padding for spacing
      }}
    >
      <Box
        sx={{
          width: '100%', // Adjust width as needed
          minHeight: 'calc(100vh - 100px)', // Ensure it takes almost the full screen height, accounting for padding
          padding: '20px',
          backgroundColor: '#eeeeee',
          borderRadius: '16px',
          display: 'flex',
          flexDirection: 'column',
          alignItems: 'center',
          justifyContent: 'space-between',
        }}
      >
        <Box
          sx={{
            width: '100%',
            display: 'flex',
            justifyContent: 'space-between',
            alignItems: 'center',
            mb: 2,
          }}
        >
          <Typography variant="h4" sx={{ color: '#5e5bfb', fontWeight: 'bold' }}>
            EduQuest
          </Typography>
          <Box sx={{ display: 'flex', alignItems: 'center', gap: 1 }}>
            <Button
              variant="contained"

```

```

sx={{
  backgroundColor: 'white',
  color: '#5c5b8f',
  fontWeight: 'bold',
  boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
  '&:hover': {
    backgroundColor: 'white',
  },
}}
component={Link}
to="/"
>
  Home
</Button>
<Button
  variant="contained"
  sx={{
    backgroundColor: 'white',
    color: '#5c5b8f',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: 'white',
    },
  }}
  component={Link}
  to="/chatscren"
>
  Chat
</Button>
<Button
  variant="contained"
  sx={{
    backgroundColor: 'white',
    color: '#5c5b8f',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: 'white',
    },
  }}
  component={Link}
  to="/compareunis"
>
  Compare
</Button>
<TextField
  variant="outlined"
  placeholder="Search"

```

```

size="small"
sx={{ backgroundColor: 'white' }}
InputProps={{
  startAdornment: (
    <InputAdornment position="start">
      <SearchIcon />
    </InputAdornment>
  ),
}}
/>
<Button
  variant="contained"
  sx={{
    backgroundColor: 'white',
    color: '#5c5b8f',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: 'white',
    },
  }}
  component={Link}
  to="/login"
>
  Login
</Button>
<Button
  variant="contained"
  sx={{
    backgroundColor: '#4e94f7',
    color: 'ffffff',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: '#4e94f7',
    },
  }}
  component={Link}
  to="/register"
>
  Register
</Button>
</Box>
</Box>
<Box
  sx={{
    width: '100%',
    display: 'flex',
    flexDirection: 'column',

```

```

      alignItems: 'center',
      justifyContent: 'center',
      flexGrow: 1, // Ensure it takes up the remaining space
    })
  >
  <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold', textAlign: 'center' }}>
    Registration successful!
  </Typography>
  <Typography variant="h5" sx={{ color: '#5c5b8f', fontWeight: 'bold', textAlign: 'center' }}>
    <Link to="/login" style={{ textDecoration: 'none', color: '#4e94f7' }}>
      Login here
    </Link>
  </Typography>
</Box>
<Footer />
</Box>
</Container>
);
};

export default Regsuccess;

```

React.js/Regsuccess.js

```

import React from 'react';
import { Button, TextField, Container, Box, Typography, InputAdornment } from '@mui/material';
import SearchIcon from '@mui/icons-material/Search';
import yourImage from './Img11.png'; // Adjust the path as needed
import Register from './Register';
import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';
import Regsuccess from './Regsuccess';
import Login from './Login';
import Universities from './Universities';
import Unidetls from './Unidetls';
import Chatscreen from './Chatt';
import CompareUniversities from './CompareUniversities';
import ComparisonScreen from './ComparisonScreen';
import AdminLogin from './AdminLogin';
import Adminprofile from './Adminprofile';
import AdminMain from './AdminMain';
import RegistrationRequestManagement from './RegistrationRequestManagement';
import Chatadminmngmnt from './Chatadminmngmnt';
import Scholarships from './Scholarships';
import FeeStructurePage from './FeeStructurePage';
import Custdtls from './Custdtls';
import { AuthProvider } from './AuthContext'; // Import your AuthContext
import Fuastdtls from './Fuastdtls';
import FUUASTFeeStructure from './FUUASTFeeStructure';

```

```

import BahriaFeeStructure from './BahriaFeeStructure';
import Bahriadtls from './Bahriadtls';

const Footer = () => {
  return (
    <Box
      sx={ {
        width: '100%',
        backgroundColor: '#5e5bf9',
        color: 'white',
        padding: '20px 0',
        mt: 'auto',
        textAlign: 'center',
      } }
    >
    <Typography variant="h6" sx={ { mb: 1 } }>
      EduQuest
    </Typography>
    <Typography variant="body1" sx={ { mb: 1 } }>
      &copy; { new Date().getFullYear() } EduQuest. All Rights Reserved.
    </Typography>
    <Box sx={ { display: 'flex', justifyContent: 'center', gap: 2 } }>
      <Button component={Link} to="/" sx={ { color: 'white' } }>
        Home
      </Button>
      <Button component={Link} to="/about" sx={ { color: 'white' } }>
        About Us
      </Button>
      <Button component={Link} to="/contact" sx={ { color: 'white' } }>
        Contact
      </Button>
      <Button component={Link} to="/privacy" sx={ { color: 'white' } }>
        Privacy Policy
      </Button>
    </Box>
  </Box>
  );
};

const Home = () => {
  return (
    <
      <Container
        maxWidth={false} // Remove max width restriction
        sx={ {
          minHeight: '100vh', // Set full viewport height
          backgroundColor: '#cbcaf0',

```

```

display: 'flex',
justifyContent: 'center',
alignItems: 'center',
padding: '50px', // Add some padding for spacing
}}
>
<Box
  sx={{
    width: '100%', // Adjust width as needed
    minHeight: 'calc(100vh - 100px)', // Ensure it takes almost the full screen height, accounting for padding
    padding: '20px',
    backgroundColor: '#eeeeee',
    borderRadius: '16px',
    display: 'flex',
    flexDirection: 'column',
    alignItems: 'center',
    justifyContent: 'space-between',
  }}
>
  <Box
    sx={{
      width: '100%',
      display: 'flex',
      justifyContent: 'space-between',
      alignItems: 'center',
      mb: 2,
    }}
    >
      <Typography variant="h4" sx={{ color: '#5e5bfb', fontWeight: 'bold' }}>
        EduQuest
      </Typography>
      <Box>
        <Button
          variant="contained"
          sx={{
            mr: 1,
            backgroundColor: 'white',
            color: '#5c5b8f',
            fontWeight: 'bold',
            boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
            '&:hover': {
              backgroundColor: 'white',
            },
          }}
          component={Link}
          to="/login"
        >
          Login
        </Button>

```

```

<Button
  variant="contained"
  sx={{
    backgroundColor: '#4e94f7',
    color: 'ffffff',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: '#4e94f7',
    },
  }}
  component={Link}
  to="/register"
>
  Register
</Button>
</Box>
</Box>
<Box
  sx={{
    display: 'flex',
    width: '100%',
    justifyContent: 'space-between',
    alignItems: 'center',
    padding: '40px', // Reduced padding top
    margin: '20px 0', // Add margin top to push content up
  }}
>
  <Box sx={{ width: '55%' }}>
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold' }}>
      Your Central
    </Typography>
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold' }}>
      Gateway to
    </Typography>
    <Typography variant="h3" sx={{ mb: 1, color: '#5c5b8f', fontWeight: 'bold' }}>
      University Insights
    </Typography>
    <Typography variant="subtitle1" sx={{ mb: 2 }}>
      Welcome to EduQuest, your central gateway to university insights! Discover comprehensive details
on programs, Universities, and scholarships all in one place. Start exploring today and find your perfect
university match.
    </Typography>
  </Box>
  <Box sx={{ display: 'flex', alignItems: 'center' }}>
    <TextField
      variant="outlined"
      placeholder="Find what you're looking for"
      sx={{ width: '270px', borderTopRightRadius: 0, borderBottomRightRadius: 0 }}
    </TextField>
  </Box>
</Box>

```



```

      InputProps={ {
        startAdornment: (
          <InputAdornment position="start">
            <SearchIcon />
          </InputAdornment>
        ),
      } }
    />

    <Button
      variant="contained"
      sx={ {
        backgroundColor: '#438ef8',
        height: '50px',
        color: 'ffffff',
        boxShadow: '0px 0px 10px rgba(67, 142, 248, 0.8), 0px 0px 20px rgba(67, 142, 248, 0.6), 0px
0px 30px rgba(67, 142, 248, 0.4)',
        borderTopLeftRadius: 0,
        borderBottomLeftRadius: 0,
        '&:hover': {
          backgroundColor: 'purple',
        },
      } }
    >
      Search University
    </Button>
  </Box>
</Box>

</Box>
<Box
  component="img"
  sx={ {
    width: '40%',
    height: 'auto',
  } }
  alt="Your image description"
  src={ yourImage }
/>
</Box>
<Footer />
</Box>

</Container>
</>
);
};

const App = () => {
  return (
    <AuthProvider>

```

```

<Router>
  <Routes>
    <Route path="/" element={<Home />} />
    <Route path="/register" element={<Register />} />
    <Route path="/registersuccess" element={<Regsuccess />} />
    <Route path="/login" element={<Login />} />
    <Route path="/Unis" element={<Universities />} />
    <Route path="/Custdtls" element={<Custdtls />} />
    <Route path="/Fuastdtls" element={<Fuastdtls />} />
    <Route path="/Unidtl" element={<Unidetls />} />
    <Route path="/chatscren" element={<Chatscreen />} />
    <Route path="/compareunis" element={<CompareUniversities />} />
    <Route path="/compareScreen" element={<ComparisonScreen />} />
    <Route path="/Adminlogin" element={<AdminLogin />} />
    <Route path="/welcomeAdmin" element={<Adminprofile />} />
    <Route path="/AdminMain" element={<AdminMain />} />
    <Route path="/Registermngmnt" element={<RegistrationRequestManagement />} />
    <Route path="/chatmngmnt" element={<Chatadminmngmnt />} />
    <Route path="/scholarship" element={<Scholarships />} />
    <Route path="/fee" element={<FeeStructurePage />} />
    <Route path="/Fuuastfee" element={<FUUASTFeeStructure />} />
    <Route path="/Bahriadtl" element={<Bahriadtl />} />
    <Route path="/BahriaFee" element={<BahriaFeeStructure />} />

    { /* Add more routes as needed */ }
  </Routes>
</Router>
</AuthProvider>
);
};

export default App;

```

React.js

```

import React, { createContext, useState, useEffect } from 'react';

// Create AuthContext
export const AuthContext = createContext();

// AuthProvider component
export const AuthProvider = ({ children }) => {
  const [authToken, setAuthToken] = useState(null);

  // Save token to local storage
  const saveToken = (token) => {
    setAuthToken(token);
  }

```

```

    localStorage.setItem('authToken', token); // Save in local storage for persistence
  };

  // Load token from local storage on initial render
  useEffect(() => {
    const token = localStorage.getItem('authToken');
    if (token) {
      setAuthToken(token);
    }
  }, []);

  // Clear token on logout
  const clearToken = () => {
    setAuthToken(null);
    localStorage.removeItem('authToken');
  };

  return (
    <AuthContext.Provider value={{ authToken, saveToken, clearToken }}>
      {children}
    </AuthContext.Provider>
  );
};

```

React.js/Universities.js

```

import React from 'react';
import { Button, Container, Box, Typography, Grid, TextField, InputAdornment } from '@mui/material';
import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';
import FUUAST from './FUUASTpic.png';
import InstituteofSpaceTechnology from './IST.jpeg';
import PakistanINstituteofEngineeringandAppliedSciences from './PIEAS.jpeg';
import COMSATSInstituteofInformationTechnology from './COMSATS.jpeg';
import CapitalUniversityofScienceanddechnology from './CUST.jpeg';
import InternationalIslamicUniversity from './IIUI.jpeg';
import BahriaUniversity from './Bahria.jpeg';
import RiphahInternationalUniversity from './Ripah.jpeg';
import SearchIcon from '@mui/icons-material/Search';
const Footer = () => {
  return (
    <Box
      sx={{
        width: '100%',
        backgroundColor: '#5e5bfb',
        color: 'white',
        padding: '20px 0',
        mt: 'auto',
        textAlign: 'center',

```

```

    }}
  >
  <Typography variant="h6" sx={{ mb: 1 }}>
    EduQuest
  </Typography>
  <Typography variant="body1" sx={{ mb: 1 }}>
    &copy; { new Date().getFullYear() } EduQuest. All Rights Reserved.
  </Typography>
  <Box sx={{ display: 'flex', justifyContent: 'center', gap: 2 }}>
    <Button component={Link} to="/" sx={{ color: 'white' }}>
      Home
    </Button>
    <Button component={Link} to="/about" sx={{ color: 'white' }}>
      About Us
    </Button>
    <Button component={Link} to="/contact" sx={{ color: 'white' }}>
      Contact
    </Button>
    <Button component={Link} to="/privacy" sx={{ color: 'white' }}>
      Privacy Policy
    </Button>
  </Box>
</Box>
);
};

const universities = [
  { name: 'FUUAST', image: FUUAST, id: 'university-a' },
  { name: 'Institute of Space Technology', image: InstituteofSpaceTechnology, id: 'university-b' },
  { name: 'Pakistan INstitute of Engineering and Applied Sciences', image:
PakistanINstituteofEngineeringandAppliedSciences, id: 'university-c' },
  { name: 'COMSATS Institute of Information Technology', image:
COMSATSInstituteofInformationTechnology, id: 'university-d' },
  { name: 'Capital University of Science and Technology', image: CapitalUniversityofScienceandTechnology,
id: 'university-e' },
  { name: 'International Islamic University', image: InternationalIslamicUniversity, id: 'university-f' },
  { name: 'Bahria University', image: BahriaUniversity, id: 'university-g' },
  { name: 'Riphah International University', image: RiphahInternationalUniversity, id: 'university-h' },
];

const Universities = () => {
  return (
    <
      <Container
        maxWidth={false}
        sx={{
          minHeight: '100vh',
          backgroundColor: '#cbcaf0',
          display: 'flex',

```

```

        justifyContent: 'center',
        alignItems: 'center',
        padding: '50px',
      }}
    >
    <Box
      sx={{
        width: '100%',
        minHeight: 'calc(100vh - 100px)',
        padding: '20px',
        backgroundColor: '#eeeeee',
        borderRadius: '16px',
        display: 'flex',
        flexDirection: 'column',
        alignItems: 'center',
        justifyContent: 'space-between',
      }}
    >
    <Box
      sx={{
        width: '100%',
        display: 'flex',
        justifyContent: 'space-between',
        alignItems: 'center',
        mb: 2,
      }}
    >
    <Typography variant="h4" sx={{ color: '#5e5bfb', fontWeight: 'bold' }}>
      EduQuest
    </Typography>
    <Box sx={{ display: 'flex', alignItems: 'center', gap: 1 }}>
      <Button
        variant="contained"
        sx={{
          backgroundColor: 'white',
          color: '#5c5b8f',
          fontWeight: 'bold',
          boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
          '&:hover': {
            backgroundColor: 'white',
          },
        }}
        component={Link}
        to="/"
      >
        Home
      </Button>
      <Button
        variant="contained"

```

```

sx={{
  backgroundColor: 'white',
  color: '#5c5b8f',
  fontWeight: 'bold',
  boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
  '&:hover': {
    backgroundColor: 'white',
  },
}}
component={Link}
to="/chatscren"
>
  Chat
</Button>
<Button
  variant="contained"
  sx={{
    backgroundColor: 'white',
    color: '#5c5b8f',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: 'white',
    },
  }}
  component={Link}
  to="/scholarship"
>
  Scholarship
</Button>
<Button
  variant="contained"
  sx={{
    backgroundColor: 'white',
    color: '#5c5b8f',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: 'white',
    },
  }}
  component={Link}
  to="/compare"
>
  Compare
</Button>
<TextField
  variant="outlined"
  placeholder="Search"

```

```

size="small"
sx={{ backgroundColor: 'white' }}
InputProps={{
  startAdornment: (
    <InputAdornment position="start">
      <SearchIcon />
    </InputAdornment>
  ),
}}
/>
<Button
  variant="contained"
  sx={{
    backgroundColor: 'white',
    color: '#5c5b8f',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: 'white',
    },
  }}
  component={Link}
  to="/login"
>
  Login
</Button>
<Button
  variant="contained"
  sx={{
    backgroundColor: '#4e94f7',
    color: 'ffffff',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: '#4e94f7',
    },
  }}
  component={Link}
  to="/register"
>
  Register
</Button>
</Box>
</Box>
<Box
  sx={{
    width: '100%',
    padding: '20px',
    display: 'flex',

```



```

        </Box>
        </Link>
      </Grid>
    )))
  </Grid>
</Box>
<Footer />
</Box>
</Container>
</>
);
};

export default Universities;

```

React.js

```

import React, {useState,useEffect} from 'react';
import { Button, Container, Box, Typography, TextField, InputAdornment } from '@mui/material';
import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';

import yourImage from './unidlss.png';
import imglogo from './CUSTlogo.png';
import SearchIcon from '@mui/icons-material/Search';
import PhoneIcon from '@mui/icons-material/Phone';
import EmailIcon from '@mui/icons-material/Email';

const Footer = () => {
  return (
    <Box
      sx={ {
        width: '100%',
        backgroundColor: '#5e5bfb',
        color: 'white',
        padding: '20px 0',
        mt: 'auto',
        textAlign: 'center',
      } }
    >
      <Typography variant="h6" sx={ { mb: 1 } }>
        EduQuest
      </Typography>
      <Typography variant="body1" sx={ { mb: 1 } }>
        &copy; { new Date().getFullYear() } EduQuest. All Rights Reserved.
      </Typography>
      <Box sx={ { display: 'flex', justifyContent: 'center', gap: 2 } }>

```

```

    <Button component={Link} to="/" sx={{ color: 'white' }}>
      Home
    </Button>
    <Button component={Link} to="/about" sx={{ color: 'white' }}>
      About Us
    </Button>
    <Button component={Link} to="/contact" sx={{ color: 'white' }}>
      Contact
    </Button>
    <Button component={Link} to="/privacy" sx={{ color: 'white' }}>
      Privacy Policy
    </Button>
  </Box>
</Box>
);
};

const Unidetls = () => {
  const [universityData, setUniversityData] = useState(null);
  const [aboutUsData, setAboutUsData] = useState(null);
  const [contactData, setContactData] = useState(null);

  useEffect(() => {
    // Fetch data from the APIs
    const fetchData = async () => {
      const custResponse = await fetch('http://127.0.0.1:5000/api/cust-data');
      const custData = await custResponse.json();
      setUniversityData(custData);

      const aboutResponse = await fetch('http://127.0.0.1:5000/api/about-us');
      const aboutData = await aboutResponse.json();
      console.log("ABOUT US IS:", aboutData);
      setAboutUsData(aboutData);

      const contactResponse = await fetch('http://127.0.0.1:5000/api/contact-us');
      const contactData = await contactResponse.json();
      setContactData(contactData);
    };

    fetchData();
  }, []);

  if (!universityData || !aboutUsData || !contactData) {
    return <Typography>Loading...</Typography>; // Show loading until data is fetched
  }
  return (
    <
      <Container
        maxWidth={false}

```

```

sx={ {
  minHeight: '100vh',
  backgroundColor: '#cbcaf0',
  display: 'flex',
  flexDirection: 'column',
  justifyContent: 'flex-start',
  alignItems: 'center',
  padding: '50px',
}}
>
<Box
  sx={ {
    width: '100%',
    padding: '20px',
    backgroundColor: '#eeeeee',
    borderRadius: '16px',
  }}
>
  <Box
    sx={ {
      width: '100%',
      display: 'flex',
      justifyContent: 'space-between',
      alignItems: 'center',
      mb: 2,
    }}
  >
    <Typography variant="h4" sx={ { color: '#5e5bfb', fontWeight: 'bold' } }>
      EduQuest
    </Typography>
    <Box sx={ { display: 'flex', alignItems: 'center', gap: 1 } }>
      <Button
        variant="contained"
        sx={ {
          backgroundColor: 'white',
          color: '#5c5b8f',
          fontWeight: 'bold',
          boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
          '&:hover': {
            backgroundColor: 'white',
          },
        }}
        component={Link}
        to="/"
      >
        Home
      </Button>
      <Button
        variant="contained"

```

```

sx={{
  backgroundColor: 'white',
  color: '#5c5b8f',
  fontWeight: 'bold',
  boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
  '&:hover': {
    backgroundColor: 'white',
  },
}}
component={Link}
to="/chatscren"
>
  Chat
</Button>
<Button
  variant="contained"
  sx={{
    backgroundColor: 'white',
    color: '#5c5b8f',
    fontWeight: 'bold',
    boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
    '&:hover': {
      backgroundColor: 'white',
    },
  }}
  component={Link}
  to="/compareunis"
>
  Compare
</Button>
<TextField
  variant="outlined"
  placeholder="Search"
  size="small"
  sx={{ backgroundColor: 'white' }}
  InputProps={{
    startAdornment: (
      <InputAdornment position="start">
        <SearchIcon />
      </InputAdornment>
    ),
  }}
/>
<Button
  variant="contained"
  sx={{
    backgroundColor: 'white',
    color: '#5c5b8f',
    fontWeight: 'bold',

```

```

        boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
        '&:hover': {
            backgroundColor: 'white',
        },
    }}
    component={Link}
    to="/login"
>
    Login
</Button>
<Button
    variant="contained"
    sx={{
        backgroundColor: '#4e94f7',
        color: 'ffffff',
        fontWeight: 'bold',
        boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
        '&:hover': {
            backgroundColor: '#4e94f7',
        },
    }}
    component={Link}
    to="/register"
>
    Register
</Button>
</Box>
</Box>
<Box sx={{ display: 'flex', alignItems: 'flex-start', mb: 2,paddingTop:5 }}>
<Box
    component="img"
    sx={{
        width: '10%',
        height: '10%',
        mr: 2,

    }}
    alt="Your image description"
    src={imglogo}
/>

<Box sx={{ textAlign: 'left', mb: 2,paddingLeft:'30px' }}>
    <Typography variant="h3" sx={{ color: '#5c5b8f', fontWeight: 'bold',fontSize:70 }}>
        {aboutUsData.heading}
    </Typography>
    <Typography variant="body1" sx={{ color: '#5f5b5b', mt: 2,maxWidth:'90%',fontSize:35 }}>
        {aboutUsData.paragraph}</Typography>
    <Typography variant="body1" sx={{ color: '#blue', mt: 2,maxWidth:'90%',fontSize:35 }}>
    <a href="https://cust.edu.pk/">Visit the Page to see more</a>

```

```

    </Typography>
    <Typography variant="body1" sx={{ color: 'blue', mt: 2,maxWidth:'90%',fontSize:35 }}>
    <Link to="/fee" style={{ textDecoration: 'none', color: 'blue' }}>
    Check Fee Structure
  </Link>
  <Typography>
    <Box sx={{ textAlign: 'left', mb: 4, paddingTop: 5 }}>
    <Typography variant="h5" sx={{ color: '#5c5b8f', fontWeight: 'bold', fontSize: 35 }}>
    Contact At
  </Typography>
  <Box sx={{ display: 'flex', justifyContent: 'left', alignItems: 'center', mt: 2 }}>
    <PhoneIcon sx={{ color: '#5c5b8f', mr: 1, fontSize: 30 }} />
    <Box sx={{ display: 'flex', flexDirection: 'column' }}>
      <Typography variant="body1" sx={{ color: '#5c5b8f', fontSize: 30 }}>
        Number
      <Typography>
        <Typography variant="body2" sx={{ color: '#5c5b8f', fontSize: 20 }}>
          UAN
      </Typography>
    </Box>
    <Box sx={{ ml: 4 }}>
      <EmailIcon sx={{ color: '#5c5b8f', mr: 1, fontSize: 30 }} />
      <Typography variant="body1" sx={{ color: '#5c5b8f', fontSize: 30 }}>
        {contactData.contact.email}
      </Typography>
    </Box>
  </Box>

  <Box>
    <Box>
      <Box
        component="img"
        sx={{
          width: '40%',
          height: 'auto',
          ml: 2,
          paddingTop:30
        }}
        alt="Generic image"
        src={yourImage}
      />
    </Box>

    <Footer />
  </Box>
</Container>
</>

```

```
);
};

export default Unidetls;
```

React.js

```
import React, {useState,useEffect} from 'react';
import { Button, Container, Box, Typography, TextField, InputAdornment } from '@mui/material';
import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';

import yourImage from './unidlss.png';
import imglogo from './CUSTlogo.png';
import SearchIcon from '@mui/icons-material/Search';
import PhoneIcon from '@mui/icons-material/Phone';
import EmailIcon from '@mui/icons-material/Email';

const Footer = () => {
  return (
    <Box
      sx={{
        width: '100%',
        backgroundColor: '#5e5bfb',
        color: 'white',
        padding: '20px 0',
        mt: 'auto',
        textAlign: 'center',
      }}
    >
      <Typography variant="h6" sx={{ mb: 1 }}>
        EduQuest
      </Typography>
      <Typography variant="body1" sx={{ mb: 1 }}>
        &copy; { new Date().getFullYear() } EduQuest. All Rights Reserved.
      </Typography>
      <Box sx={{ display: 'flex', justifyContent: 'center', gap: 2 }}>
        <Button component={Link} to="/" sx={{ color: 'white' }}>
          Home
        </Button>
        <Button component={Link} to="/about" sx={{ color: 'white' }}>
          About Us
        </Button>
        <Button component={Link} to="/contact" sx={{ color: 'white' }}>
          Contact
        </Button>
        <Button component={Link} to="/privacy" sx={{ color: 'white' }}>
          Privacy Policy
        </Button>
      </Box>
    </Box>
  )
}
```

```

    </Button>
  </Box>
</Box>
);
};

const Custdtls = () => {
  const [universityData, setUniversityData] = useState(null);
  const [aboutUsData, setAboutUsData] = useState(null);
  const [contactData, setContactData] = useState(null);

  useEffect(() => {
    // Fetch data from the APIs
    const fetchData = async () => {
      const custResponse = await fetch('http://127.0.0.1:5000/api/cust-data');
      const custData = await custResponse.json();
      setUniversityData(custData);

      const aboutResponse = await fetch('http://127.0.0.1:5000/api/about-us');
      const aboutData = await aboutResponse.json();
      console.log("ABOUT US IS:", aboutData);
      setAboutUsData(aboutData);

      const contactResponse = await fetch('http://127.0.0.1:5000/api/contact-us');
      const contactData = await contactResponse.json();
      setContactData(contactData);
    };

    fetchData();
  }, []);

  if (!universityData || !aboutUsData || !contactData) {
    return <Typography>Loading...</Typography>; // Show loading until data is fetched
  }
  return (
    <
      <Container
        maxWidth={false}
        sx={{
          minHeight: '100vh',
          backgroundColor: '#cbcaf0',
          display: 'flex',
          flexDirection: 'column',
          justifyContent: 'flex-start',
          alignItems: 'center',
          padding: '50px',
        }}
      >
      <Box

```



```

sx={ {
  width: '100%',
  padding: '20px',
  backgroundColor: '#eeeeee',
  borderRadius: '16px',
}}
>
<Box
sx={ {
  width: '100%',
  display: 'flex',
  justifyContent: 'space-between',
  alignItems: 'center',
  mb: 2,
}}
>
<Typography variant="h4" sx={ { color: '#5e5bfb', fontWeight: 'bold' } }>
  EduQuest
</Typography>
<Box sx={ { display: 'flex', alignItems: 'center', gap: 1 } }>
  <Button
    variant="contained"
    sx={ {
      backgroundColor: 'white',
      color: '#5c5b8f',
      fontWeight: 'bold',
      boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
      '&:hover': {
        backgroundColor: 'white',
      },
    } }
    component={Link}
    to="/"
  >
    Home
  </Button>
  <Button
    variant="contained"
    sx={ {
      backgroundColor: 'white',
      color: '#5c5b8f',
      fontWeight: 'bold',
      boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
      '&:hover': {
        backgroundColor: 'white',
      },
    } }
    component={Link}
    to="/chatscren"
  >

```

```

    >
    Chat
  </Button>
  <Button
    variant="contained"
    sx={{
      backgroundColor: 'white',
      color: '#5c5b8f',
      fontWeight: 'bold',
      boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
      '&:hover': {
        backgroundColor: 'white',
      },
    }}
    component={Link}
    to="/compareunis"
  >
  Compare
  </Button>
  <TextField
    variant="outlined"
    placeholder="Search"
    size="small"
    sx={{ backgroundColor: 'white' }}
    InputProps={{
      startAdornment: (
        <InputAdornment position="start">
          <SearchIcon />
        </InputAdornment>
      ),
    }}
  />
  <Button
    variant="contained"
    sx={{
      backgroundColor: 'white',
      color: '#5c5b8f',
      fontWeight: 'bold',
      boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
      '&:hover': {
        backgroundColor: 'white',
      },
    }}
    component={Link}
    to="/login"
  >
  Login
  </Button>
  <Button

```

```

    variant="contained"
    sx={{
      backgroundColor: '#4e94f7',
      color: 'ffffff',
      fontWeight: 'bold',
      boxShadow: '0px 4px 6px rgba(0, 0, 0, 0.1)',
      '&:hover': {
        backgroundColor: '#4e94f7',
      },
    }}
    component={Link}
    to="/register"
  >
    Register
  </Button>
</Box>
</Box>
<Box sx={{ display: 'flex', alignItems: 'flex-start', mb: 2, paddingTop: 5 }}>
<Box
  component="img"
  sx={{
    width: '10%',
    height: '10%',
    mr: 2,

  }}
  alt="Your image description"
  src={imglogo}
/>

<Box sx={{ textAlign: 'left', mb: 2, paddingLeft: '30px' }}>
  <Typography variant="h3" sx={{ color: '#5c5b8f', fontWeight: 'bold', fontSize: 70 }}>
    {aboutUsData.heading}
  </Typography>
  <Typography variant="body1" sx={{ color: '#5f5b5b', mt: 2, maxWidth: '90%', fontSize: 35 }}>
    {aboutUsData.paragraph} </Typography>
  <Typography variant="body1" sx={{ color: '#blue', mt: 2, maxWidth: '90%', fontSize: 35 }}>
    <a href="https://cust.edu.pk/">Visit the Page to see more</a>
  </Typography>
  <Typography variant="body1" sx={{ color: '#blue', mt: 2, maxWidth: '90%', fontSize: 35 }}>
    <Link to="/fee" style={{ textDecoration: 'none', color: 'blue' }}>
    Check Fee Structure
  </Link>
  </Typography>
  <Box sx={{ textAlign: 'left', mb: 4, paddingTop: 5 }}>
    <Typography variant="h5" sx={{ color: '#5c5b8f', fontWeight: 'bold', fontSize: 35 }}>
    Contact At
  </Typography>
  <Box sx={{ display: 'flex', justifyContent: 'left', alignItems: 'center', mt: 2 }}>

```

```

    <PhoneIcon sx={{ color: '#5c5b8f', mr: 1, fontSize: 30 }} />
    <Box sx={{ display: 'flex', flexDirection: 'column' }}>
      <Typography variant="body1" sx={{ color: '#5c5b8f', fontSize: 30 }}>
        {contactData.contact.phone_uan}
      </Typography>
      <Typography variant="body2" sx={{ color: '#5c5b8f', fontSize: 20 }}>
        UAN
      </Typography>
    </Box>
    <Box sx={{ ml: 4 }}>
      <EmailIcon sx={{ color: '#5c5b8f', mr: 1, fontSize: 30 }} />
      <Typography variant="body1" sx={{ color: '#5c5b8f', fontSize: 30 }}>
        {contactData.contact.email}
      </Typography>
    </Box>
  </Box>

</Box>
<Box>
  component="img"
  sx={{
    width: '40%',
    height: 'auto',
    ml: 2,
    paddingTop: 30
  }}
  alt="Generic image"
  src={yourImage}
/>
</Box>

  <Footer />
</Box>
</Container>
</>
);
};

export default Custdtls;

```

React.js

```
import React, { useEffect, useState } from 'react';
import { fetchFeeStructure } from './apiHandler';
import { Container, Box, Typography, Table, TableBody, TableCell, TableContainer, TableHead, TableRow, Paper } from '@mui/material';

const FeeStructureDisplay = () => {
  const [feeStructure, setFeeStructure] = useState([]);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState("");

  useEffect(() => {
    const getFeeStructure = async () => {
      try {
        const data = await fetchFeeStructure();
        setFeeStructure(data);
        setLoading(false);
      } catch (err) {
        setError(err.message);
        setLoading(false);
      }
    };

    getFeeStructure();
  }, []);

  if (loading) return <Typography>Loading...</Typography>;
  if (error) return <Typography color="error">Error: {error}</Typography>;

  return (
    <Container>
      <Box my={4}>
        <Typography variant="h4" gutterBottom align="center">Fee Structure</Typography>
        <TableContainer component={Paper}>
          <Table>
            <TableHead>
              <TableRow>
                <TableCell align="center"><Typography variant="h6">S. No.</Typography></TableCell>
                <TableCell align="center"><Typography variant="h6">Programs</Typography></TableCell>
                <TableCell align="center"><Typography variant="h6">For Intake Fall-18 & Spring-19</Typography></TableCell>
                <TableCell align="center"><Typography variant="h6">For Intake Fall-19 & Spring-20</Typography></TableCell>
                <TableCell align="center"><Typography variant="h6">For Intake Fall-20 & Spring-21</Typography></TableCell>
                <TableCell align="center"><Typography variant="h6">For Intake Fall-21 & Spring-22</Typography></TableCell>
              </TableRow>
            </TableHead>
            <TableBody>
              <TableRow>
                <TableCell align="center">1</TableCell>
                <TableCell align="center">B.Tech. in Computer Science & Engineering</TableCell>
                <TableCell align="center">2018</TableCell>
                <TableCell align="center">2019</TableCell>
                <TableCell align="center">2020</TableCell>
                <TableCell align="center">2021</TableCell>
              </TableRow>
              <TableRow>
                <TableCell align="center">2</TableCell>
                <TableCell align="center">B.Tech. in Information Technology</TableCell>
                <TableCell align="center">2018</TableCell>
                <TableCell align="center">2019</TableCell>
                <TableCell align="center">2020</TableCell>
                <TableCell align="center">2021</TableCell>
              </TableRow>
              <TableRow>
                <TableCell align="center">3</TableCell>
                <TableCell align="center">B.Tech. in Software Engineering</TableCell>
                <TableCell align="center">2018</TableCell>
                <TableCell align="center">2019</TableCell>
                <TableCell align="center">2020</TableCell>
                <TableCell align="center">2021</TableCell>
              </TableRow>
              <TableRow>
                <TableCell align="center">4</TableCell>
                <TableCell align="center">B.Tech. in Cyber Security</TableCell>
                <TableCell align="center">2018</TableCell>
                <TableCell align="center">2019</TableCell>
                <TableCell align="center">2020</TableCell>
                <TableCell align="center">2021</TableCell>
              </TableRow>
              <TableRow>
                <TableCell align="center">5</TableCell>
                <TableCell align="center">B.Tech. in Data Science</TableCell>
                <TableCell align="center">2018</TableCell>
                <TableCell align="center">2019</TableCell>
                <TableCell align="center">2020</TableCell>
                <TableCell align="center">2021</TableCell>
              </TableRow>
            </TableBody>
          </Table>
        </TableContainer>
      </Box>
    </Container>
  );
};
```

```

        <TableCell align="center"><Typography variant="h6">For Intake Fall-22 & Spring-
23</Typography></TableCell>
        <TableCell align="center"><Typography variant="h6">For Intake Fall-23 & Spring-
24</Typography></TableCell>
        <TableCell align="center"><Typography variant="h6">For Intake Fall-24 & Spring-
25</Typography></TableCell>
      </TableRow>
    </TableHead>
    <TableBody>
      {feeStructure.map(({ program, fees }, index) => (
        <TableRow key={index}>
          <TableCell align="center">{index + 1}</TableCell>
          <TableCell align="center">{program}</TableCell>
          {fees.map((fee, i) => (
            <TableCell key={i} align="center">{fee}</TableCell>
          ))}
        </TableRow>
      ))}
    </TableBody>
  </Table>
</TableContainer>
</Box>
</Container>
);
};

export default FeeStructureDisplay;

```

React.j

```

import React, { useState, useEffect } from 'react';
import { Button, Container, Box, Typography, TextField, InputAdornment } from '@mui/material';
import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';
import yourImage from './Img11.png'; // Replace with FUAST-specific image

import SearchIcon from '@mui/icons-material/Search';
import PhoneIcon from '@mui/icons-material/Phone';
import EmailIcon from '@mui/icons-material/Email';

const Footer = () => {
  return (
    <Box
      sx={{
        width: '100%',
        backgroundColor: '#5e5bfb',
        color: 'white',
        padding: '20px 0',
        mt: 'auto',

```

```

      textAlign: 'center',
    }}
  >
  <Typography variant="h6" sx={{ mb: 1 }}>
    EduQuest
  </Typography>
  <Typography variant="body1" sx={{ mb: 1 }}>
    &copy; {new Date().getFullYear()} EduQuest. All Rights Reserved.
  </Typography>
  <Box sx={{ display: 'flex', justifyContent: 'center', gap: 2 }}>
    <Button component={Link} to="/" sx={{ color: 'white' }}>
      Home
    </Button>
    <Button component={Link} to="/about" sx={{ color: 'white' }}>
      About Us
    </Button>
    <Button component={Link} to="/contact" sx={{ color: 'white' }}>
      Contact
    </Button>
    <Button component={Link} to="/privacy" sx={{ color: 'white' }}>
      Privacy Policy
    </Button>
  </Box>
</Box>
);
};

const Fuastdtls = () => {
  const [universityData, setUniversityData] = useState(null);
  const [aboutUsData, setAboutUsData] = useState(null);
  const [feeStructureData, setFeeStructureData] = useState(null);
  const [contactInfo, setContactInfo] = useState(null); // State for contact info

  useEffect(() => {
    const fetchData = async () => {
      // Fetch university details
      const fuastResponse = await fetch('http://127.0.0.1:5000/api/fuust-data');
      const fuastData = await fuastResponse.json();
      setUniversityData(fuastData);

      // Fetch about us details
      const aboutResponse = await fetch('http://127.0.0.1:5000/api/fuust-about');
      const aboutData = await aboutResponse.json();
      setAboutUsData(aboutData);

      // Fetch fee structure details
      const feeResponse = await fetch('http://127.0.0.1:5000/api/fee-structure');
      const feeData = await feeResponse.json();
      setFeeStructureData(feeData);
    };
  });
};

```

```

    // Fetch contact info (phone, fax, email)
    const contactResponse = await fetch('http://127.0.0.1:5000/api/fuuast-contact');
    const contactData = await contactResponse.json();
    setContactInfo(contactData);
  };

  fetchData();
}, []);

if (!universityData || !aboutUsData || !feeStructureData || !contactInfo) {
  return <Typography>Loading...</Typography>;
}

return (
  <
    <Container
      maxWidth={false}
      sx={{
        minHeight: '100vh',
        backgroundColor: '#cbcaf0',
        display: 'flex',
        flexDirection: 'column',
        justifyContent: 'flex-start',
        alignItems: 'center',
        padding: '50px',
      }}
    >
      </* Centered Logo Container */>

      </* Main Content */>
      <Box
        sx={{
          width: '100%',
          padding: '20px',
          backgroundColor: '#eeeeee',
          borderRadius: '16px',
        }}
      >
        <Box
          sx={{
            width: '100%',
            display: 'flex',
            justifyContent: 'center',
            mb: 4, // margin below the logo
          }}
        >

```



```

</Box>
<Box
  sx={{
    width: '100%',
    display: 'flex',
    justifyContent: 'space-between',
    alignItems: 'center',
    mb: 2,
  }}
>
  <Typography variant="h4" sx={{ color: '#5e5bfb', fontWeight: 'bold' }}>
    EduQuest
  </Typography>
  <Box sx={{ display: 'flex', alignItems: 'center', gap: 1 }}>
    <Button variant="contained" component={Link} to="/Unis" sx={{ backgroundColor: '#5e5bfb', color:
'white' }}>
      Home
    </Button>
    <TextField
      variant="outlined"
      placeholder="Search"
      size="small"
      sx={{ backgroundColor: 'white' }}
      InputProps={{
        startAdornment: (
          <InputAdornment position="start">
            <SearchIcon />
          </InputAdornment>
        ),
      }}
    />
  </Box>
</Box>
<Box
  component="img"
  sx={{
    width: '50%', // Adjust width as needed
    height: '40%',
  }}
  alt="University logo"
  src={universityData.logo}
/>
<Box sx={{ display: 'flex', alignItems: 'flex-start', mb: 2, paddingTop: 5 }}>
  <Box sx={{ textAlign: 'left', mb: 2, paddingLeft: '30px' }}>
    <Typography variant="h3" sx={{ color: '#5c5b8f', fontWeight: 'bold', fontSize: 70 }}>
      {universityData.name}
    </Typography>
    <Typography variant="body1" sx={{ color: '#5f5b5b', mt: 2, maxWidth: '90%', fontSize: 35 }}>
      {aboutUsData.intro}

```

```

</Typography>
<Typography variant="body1" sx={{ color: 'blue', mt: 2, maxWidth: '90%', fontSize: 35 }}>
  <a href="https://fuuast.edu.pk/">Visit the Page to see more</a>
</Typography>
<Typography variant="body1" sx={{ color: 'blue', mt: 2, maxWidth: '90%', fontSize: 35 }}>
  <Link to="/Fuuastfee" style={{ textDecoration: 'none', color: 'blue' }}>
    Check Fee Structure
  </Link>
</Typography>
<Box sx={{ textAlign: 'left', mb: 4, paddingTop: 5 }}>
  <Typography variant="h5" sx={{ color: '#5c5b8f', fontWeight: 'bold', fontSize: 35 }}>
    Contact Us
  </Typography>
  <Box sx={{ display: 'flex', justifyContent: 'left', alignItems: 'center', mt: 2 }}>
    <PhoneIcon sx={{ color: '#5c5b8f', mr: 1, fontSize: 30 }} />
    <Typography variant="body1" sx={{ color: '#5c5b8f', fontSize: 30 }}>
      {contactInfo.phone}
    </Typography>
  </Box>
  <Box sx={{ display: 'flex', justifyContent: 'left', alignItems: 'center', mt: 2 }}>
    <PhoneIcon sx={{ color: '#5c5b8f', mr: 1, fontSize: 30 }} />
    <Typography variant="body1" sx={{ color: '#5c5b8f', fontSize: 30 }}>
      {contactInfo.fax}
    </Typography>
  </Box>
</Box>
</Box>
</Box>
<Box
  component="img"
  sx={{
    width: '40%',
    height: 'auto',
    ml: 2,
    paddingTop: 30,
  }}
  alt="Generic image"
  src={yourImage}
/>
</Box>
<Footer />
</Box>
</Container>
</>
);
};

export default Fuastdtls;

```

React.js

```
import React, { useState, useEffect } from 'react';

import { Container, Box, Typography, Table, TableBody, TableCell, TableContainer, TableHead, TableRow, Paper } from '@mui/material';

const FUUASTFeeStructure = () => {
  const [feeData, setFeeData] = useState(null);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState(null);

  // Fetch fee structure data from the backend
  useEffect(() => {
    const fetchFeeStructure = async () => {
      try {
        const response = await fetch('http://127.0.0.1:5000/api/fee-structure-fdrl-urdu');
        const data = await response.json();
        setFeeData(data);
      } catch (err) {
        setError('Failed to fetch fee structure data');
      } finally {
        setLoading(false);
      }
    };

    fetchFeeStructure();
  }, []);

  if (loading) {
    return <Typography>Loading Fee Structure...</Typography>;
  }

  if (error) {
    return <Typography color="error">{error}</Typography>;
  }

  return (
    <Container maxWidth="lg" sx={{ paddingTop: '50px', paddingBottom: '50px' }}>
      <Typography variant="h4" sx={{ color: '#5e5bfb', fontWeight: 'bold', mb: 3 }}>
        Fee Structure for Bachelors and Other Programs (Faculty of Law)
      </Typography>

      {feeData && feeData.length > 0 ? (
        feeData.map((faculty, index) => (
          <Box key={index} sx={{ mb: 4 }}>
            <Typography variant="h5" sx={{ fontWeight: 'bold', color: '#5c5b8f', mb: 2 }}>
              {faculty.faculty}
            </Typography>

```

```

<TableContainer component={Paper}>
  <Table sx={{ minWidth: 650 }} aria-label="fee structure table">
    <TableHead>
      <TableRow>
        <TableCell><strong>Department</strong></TableCell>
        <TableCell><strong>Admission Fee</strong></TableCell>
        <TableCell><strong>Semester Fee</strong></TableCell>
        <TableCell><strong>Other Costs</strong></TableCell>
      </TableRow>
    </TableHead>
    <TableBody>
      {faculty.programs.map((program, programIndex) => (
        <TableRow key={programIndex}>
          <TableCell>{program.department}</TableCell>
          <TableCell>{program.admission_fee}</TableCell>
          <TableCell>{program.semester_fee}</TableCell>
          <TableCell>{program.others}</TableCell>
        </TableRow>
      ))}
    </TableBody>
  </Table>
</TableContainer>
</Box>
))
): (
  <Typography>No fee structure available at the moment.</Typography>
)
</Container>
);
};

```

```
export default FUUASTFeeStructure;
```

React.js

```

import React, { useState, useEffect } from 'react';
import { Button, Container, Box, Typography, TextField, InputAdornment } from '@mui/material';
import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';
import yourImage from './Img11.png'; // Replace with a relevant image if needed

import SearchIcon from '@mui/icons-material/Search';
import PhoneIcon from '@mui/icons-material/Phone';
import EmailIcon from '@mui/icons-material/Email';
const Footer = () => {
  return (

```

```

<Box
  sx={ {
    width: '100%',
    backgroundColor: '#5e5bfb',
    color: 'white',
    padding: '20px 0',
    mt: 'auto',
    textAlign: 'center',
  } }
>
  <Typography variant="h6" sx={ { mb: 1 } }>
    EduQuest
  </Typography>
  <Typography variant="body1" sx={ { mb: 1 } }>
    &copy; { new Date().getFullYear() } EduQuest. All Rights Reserved.
  </Typography>
  <Box sx={ { display: 'flex', justifyContent: 'center', gap: 2 } }>
    <Button component={Link} to="/" sx={ { color: 'white' } }>
      Home
    </Button>
    <Button component={Link} to="/about" sx={ { color: 'white' } }>
      About Us
    </Button>
    <Button component={Link} to="/contact" sx={ { color: 'white' } }>
      Contact
    </Button>
    <Button component={Link} to="/privacy" sx={ { color: 'white' } }>
      Privacy Policy
    </Button>
  </Box>
</Box>
);
};

const Bahriadtlis = () => {
  const [universityData, setUniversityData] = useState(null);
  const [aboutUsData, setAboutUsData] = useState(null);
  const [contactInfo, setContactInfo] = useState(null); // State for contact info

  useEffect(() => {
    const fetchData = async () => {
      // Fetch university details
      const universityResponse = await fetch('http://127.0.0.1:5000/api/bahria-data');
      const universityData = await universityResponse.json();
      setUniversityData(universityData);

      // Fetch about us details
      const aboutResponse = await fetch('http://127.0.0.1:5000/api/bahria-about');
      const aboutData = await aboutResponse.json();
      setAboutUsData(aboutData);
    };
    fetchData();
  });
};

```

```

    // Fetch contact info (phone, fax, email)
    const contactResponse = await fetch('http://127.0.0.1:5000/api/bahria-contact');
    const contactData = await contactResponse.json();
    setContactInfo(contactData);
  };

  fetchData();
}, []);

if (!universityData || !aboutUsData || !contactInfo) {
  return <Typography>Loading...</Typography>;
}

return (
  <Container
    maxWidth={false}
    sx={{
      minHeight: '100vh',
      backgroundColor: '#cbcaf0',
      display: 'flex',
      flexDirection: 'column',
      justifyContent: 'flex-start',
      alignItems: 'center',
      padding: '50px',
    }}
  >
    { /* Main Content */ }
    <Box
      sx={{
        width: '100%',
        padding: '20px',
        backgroundColor: '#eeeeee',
        borderRadius: '16px',
      }}
    >
      <Box
        sx={{
          width: '100%',
          display: 'flex',
          justifyContent: 'space-between',
          alignItems: 'center',
          mb: 2,
        }}
      >
        <Typography variant="h4" sx={{ color: '#5e5bfb', fontWeight: 'bold' }}>
          EduQuest
        </Typography>

```

```

<Box sx={{ display: 'flex', alignItems: 'center', gap: 1 }}>
  <Button
    variant="contained"
    component={Link}
    to="/home" // Modify to appropriate route for the homepage
    sx={{ backgroundColor: '#5e5bfb', color: 'white' }}
  >
    Home
  </Button>
  <TextField
    variant="outlined"
    placeholder="Search"
    size="small"
    sx={{ backgroundColor: 'white' }}
    InputProps={{
      startAdornment: (
        <InputAdornment position="start">
          <SearchIcon />
        </InputAdornment>
      ),
    }}
  />
</Box>

</Box>
<Box
  sx={{
    width: '100%',
    display: 'flex',
    justifyContent: 'center',
    mb: 4,
  }}
>
  <img
    src={universityData.logo}
    alt="University Logo"
    style={{
      width: '50%', // Adjust width as needed
      height: 'auto',
    }}
  />
</Box>

<Box sx={{ display: 'flex', flexDirection: 'column', mb: 2, paddingLeft: 4 }}>
  <Typography variant="h5" sx={{ color: '#5c5b8f', fontWeight: 'bold', fontSize: 35 }}>
    About Us
  </Typography>
  <Typography variant="body1" sx={{ color: '#5f5b5b', mt: 2, fontSize: 18 }}>
    <strong>Vision:</strong> {aboutUsData.vision.join(', ')}
  </Typography>

```

```

<Typography variant="body1" sx={{ color: '#5f5b5b', mt: 2, fontSize: 18 }}>
  <strong>Mission:</strong> {aboutUsData.mission.join(', ')}
</Typography>
<Typography variant="body1" sx={{ color: '#5f5b5b', mt: 2, fontSize: 18 }}>
  <strong>Core Values:</strong> {aboutUsData.core_values.join(', ')}
</Typography>
</Box>
<Typography variant="body1" sx={{ color: 'blue', mt: 2, maxWidth: '90%', fontSize: 35 }}>
  <a href="https://www.bahria.edu.pk/">Visit the Page to see more</a>
</Typography>
<Typography variant="body1" sx={{ color: 'blue', mt: 2, maxWidth: '90%', fontSize: 35 }}>
  <Link to="/BahriaFee" style={{ textDecoration: 'none', color: 'blue' }}>
    Check Fee Structure
  </Link>
</Typography>
<Box sx={{ textAlign: 'left', mb: 4, paddingTop: 5 }}>
  <Typography variant="h5" sx={{ color: '#5c5b8f', fontWeight: 'bold', fontSize: 35 }}>
    Contact Us
  </Typography>
  <table border="1">
<thead>
  <tr>
    <th>Institution</th>
    <th>Address</th>
    <th>Phone</th>
    <th>Fax</th>
    <th>Website</th>
    <th>Google Map</th>
    <th>Office Directory</th>
  </tr>
</thead>
<tbody>
  {contactInfo.map((info, index) => (
    <tr key={index}>
      <td>{info.institution}</td>
      <td>{info.address}</td>
      <td>{info.phone ? info.phone : 'None'}</td>
      <td>{info.fax ? info.fax : 'None'}</td>
      <td>
        <a href={info.website} target="_blank" rel="noopener noreferrer">
          {info.website}
        </a>
      </td>
      <td>
        <a href={info.google_map} target="_blank" rel="noopener noreferrer">
          Google Map
        </a>
      </td>
      <td>

```



```

      {info.office_directory ? (
        <a href={info.office_directory} target="_blank" rel="noopener noreferrer">
          Office Directory
        </a>
      ) : (
        'None'
      )}
    </td>
  </tr>
)}
</tbody>
</table>
</Box>
<Footer />
</Box>
</Container>
);
};

export default Bahriadtls;

```

React.js/BahriaFeeStructure.js

```

import React, { useState, useEffect } from "react";
import {
  Container,
  Box,
  Typography,
  Table,
  TableBody,
  TableCell,
  TableContainer,
  TableHead,
  TableRow,
  Paper
} from "@mui/material";

const BahriaFeeStructure = () => {
  const [feeData, setFeeData] = useState(null);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState(null);

  useEffect(() => {
    const fetchFeeStructure = async () => {
      try {
        const response = await fetch("http://127.0.0.1:5000/api/fee-structure-bahria");
        const data = await response.json();
        setFeeData(data);
      }
    }
  });

```

```

    } catch (err) {
      setError("Failed to fetch fee structure data");
    } finally {
      setLoading(false);
    }
  };

  fetchFeeStructure();
}, []);

if (loading) {
  return (
    <Container>
      <Typography variant="h6" align="center" my={4}>
        Loading...
      </Typography>
    </Container>
  );
}

if (error) {
  return (
    <Container>
      <Typography variant="h6" align="center" color="error" my={4}>
        {error}
      </Typography>
    </Container>
  );
}

return (
  <Container>
    <Box my={4}>
      <Typography variant="h4" align="center" gutterBottom>
        Bahria Fee Structure
      </Typography>
      <TableContainer component={Paper}>
        <Table>
          <TableHead>
            <TableRow>
              <TableCell><b>Description</b></TableCell>
              {Object.keys(feeData[0].fees).map((program, index) => (
                <TableCell key={index}><b>{program}</b></TableCell>
              ))}
            </TableRow>
          </TableHead>
          <TableBody>
            {feeData.map((item, index) => (
              <TableRow key={index}>

```

```

        <TableCell>{item.description}</TableCell>
        { Object.values(item.fees).map((fee, i) => (
            <TableCell key={i}>{fee}</TableCell>
        ))}
    </TableRow>
    ))}
</TableBody>
</Table>
</TableContainer>
</Box>
</Container>
);
};

export default BahriaFeeStructure;

```

Python/Cust.py

```

from flask import Flask, jsonify
from flask_cors import CORS
import requests
from lxml import html
from bs4 import BeautifulSoup

app = Flask(__name__)
CORS(app) # Enable CORS for all routes

# Helper function to fetch and parse HTML with error handling
def fetch_html(url):
    response = requests.get(url)
    if response.status_code != 200:
        return None, {"error": "Failed to fetch the webpage"}
    return html.fromstring(response.content), None

# Helper function to extract text from XPath
def extract_text(tree, xpath, join=True, default="Data not found"):
    elements = tree.xpath(xpath)
    return " ".join(elements).strip() if join and elements else (elements[0].strip() if elements else default)

# University homepage data
@app.route('/api/cust-data', methods=['GET'])
def get_university_data():
    url = "https://cust.edu.pk/"
    tree, error = fetch_html(url)
    if error:
        return jsonify(error), 500

```

```

data = {
    "name": extract_text(tree, '/html/head/title/text()'),
    "logo": extract_text(tree, '//*[@id="masthead"]/div/div/div/div/div[1]/a[1]/img/@src', default="Logo not found")
}
return jsonify(data)

# About Us page data
@app.route('/api/about-us', methods=['GET'])
def get_about_us():
    url = "https://cust.edu.pk/about/"
    tree, error = fetch_html(url)
    if error:
        return jsonify(error), 500

    about_data = {
        "heading": extract_text(tree, '//*[@id="post-418662"]/div/div/div[3]/div/div/div/h2/text()'),
        "paragraph": extract_text(tree, '//*[@id="post-418662"]/div/div/div[3]/div/div/div/div/div/p/text()'),
        "vision": extract_text(tree, '//*[@id="post-418662"]/div/div/div[6]/div[2]/div/div/div/div/p/text()'),
        "mission": extract_text(tree, '//*[@id="post-418662"]/div/div/div[7]/div[2]/div/div/div/div/p/text()'),
        "diversity_inclusion": extract_text(tree, '//*[@id="post-418662"]/div/div/div[8]/div[2]/div/div/div/div/p/text()')
    }
    return jsonify(about_data)

# Contact Us page data
@app.route('/api/contact-us', methods=['GET'])
def get_contact_us():
    url = "https://cust.edu.pk/contact/"
    tree, error = fetch_html(url)
    if error:
        return jsonify(error), 500

    # Extract the required contact details using the updated XPath
    contact_details = { }

    # Extract phone number (UAN)
    phone_uan = extract_text(tree, '//*[@class="contact-list-info"]/li[1]/div[@class="text"]/a/text()')
    contact_details['phone_uan'] = phone_uan.strip() if phone_uan else "Not available"

    # Extract email address
    email = extract_text(tree, '//*[@class="contact-list-info"]/li[2]/div[@class="text"]/a/text()')
    contact_details['email'] = email.strip() if email else "Not available"

    # Extract phone number (general)
    phone_general = extract_text(tree, '//*[@class="contact-list-info"]/li[3]/div[@class="text"]/a/text()')
    contact_details['phone_general'] = phone_general.strip() if phone_general else "Not available"

    # Extract address

```

```

address = extract_text(tree, '//*[@class="contact-list-info address"]//li//div[@class="text"]/text()')
contact_details['address'] = ' '.join(address.split()).strip() if address else "Not available"

return jsonify({"contact": contact_details})

# Fee Structure page data
@app.route('/api/fee-structure', methods=['GET'])
def get_fee_structure():
    url = "https://cust.edu.pk/fee-structure/"

    tree, error = fetch_html(url)
    if error:
        return jsonify(error), 500

    fee_structure_data = []
    rows = tree.xpath('//table[contains(@class, "sems_table")]/tbody/tr')

    for row in rows:
        cols = row.xpath('td')
        if len(cols) > 1:
            program = cols[1].text_content().strip()
            fees = [col.text_content().strip() for col in cols[2:]]
            fee_structure_data.append({"program": program, "fees": fees})
    return jsonify({"fee_structure": fee_structure_data})

# FUUAST About Us page data
@app.route('/api/fuuast-about', methods=['GET'])
def get_fuuast_about_us():
    url = "https://fuuast.edu.pk/history/"

    response = requests.get(url)
    if response.status_code != 200:
        return jsonify({"error": "Failed to fetch the webpage"}), 500

    soup = BeautifulSoup(response.content, 'html.parser')
    main_content = soup.find("main", {"id": "content"})
    about_info = {
        "intro": " ".join(p.get_text(strip=True) for p in main_content.find_all("p") if p.get_text(strip=True)),
        "highlights": [li.get_text(strip=True) for li in main_content.find_all("li")]
    }
    return jsonify(about_info)

# FUUAST homepage data
@app.route('/api/fuuast-data', methods=['GET'])
def get_fuuast_data():
    url = "https://fuuast.edu.pk/"
    tree, error = fetch_html(url)
    if error:
        return jsonify(error), 500

    data = {
        "name": "Federal Urdu University of Arts, Science, and Technology",

```

```

        "logo": extract_text(tree, '//*[@id="header"]/div[2]/div/div/a/img/@src', default="Logo not found")
    }
    return jsonify(data)

# FUUAST Contact page data
@app.route('/api/fuuast-contact', methods=['GET'])
def get_fuuast_contact():
    url = "https://fuuast.edu.pk/registrar/"
    response = requests.get(url)
    if response.status_code != 200:
        return jsonify({"error": "Failed to fetch the webpage"}), 500

    soup = BeautifulSoup(response.content, 'html.parser')
    main_content = soup.find("main", {"id": "content"})

    contact_info = {
        "head_info": main_content.find("p", align="justify").get_text(strip=True) if main_content.find("p",
align="justify") else "Head information not found",
        "phone": main_content.find_all("p")[1].get_text(separator="\n",
strip=True).split("\n")[0].split(":")[1].strip() if main_content.find_all("p") else "Phone not found",
        "fax": main_content.find_all("p")[1].get_text(separator="\n", strip=True).split("\n")[1].split(":")[1].strip()
if main_content.find_all("p") else "Fax not found",
        "email": main_content.find_all("p")[1].get_text(separator="\n",
strip=True).split("\n")[2].split(":")[1].strip() if main_content.find_all("p") else "Email not found",
        "functions": [li.get_text(strip=True) for li in main_content.find("ul").find_all("li") if
main_content.find("ul")],
        "download_link": main_content.find("a", text="Download (Right to Information Act, 2017)
Format")["href"] if main_content.find("a", text="Download (Right to Information Act, 2017) Format") else
"Download link not found"
    }
    return jsonify(contact_info)

# FUUAST Fee Structure data
@app.route('/api/fee-structure-fdrl-urdu', methods=['GET'])
def scrape_fee_structure_fuuast():
    url = 'https://fuuast.edu.pk/fee-structure-bachelors-programmes-morning/'
    tree, error = fetch_html(url)
    if error:
        return jsonify(error), 500

    fee_data = []
    faculties = tree.xpath('//p[starts-with(text(), "Faculty of")])

    for faculty in faculties:
        faculty_name = faculty.text_content().strip()
        rows = faculty.xpath('following-sibling::div/table/tr')

```

```

faculty_fees = []
for row in rows:
    columns = row.xpath('./td/text()')
    if len(columns) == 5:
        faculty_fees.append({
            "department": columns[1].strip(),
            "others": columns[2].strip(),
            "admission_fee": columns[3].strip(),
            "semester_fee": columns[4].strip()
        })
    fee_data.append({"faculty": faculty_name, "programs": faculty_fees})

return jsonify(fee_data)

# Fetch and parse HTML content
def fetch_html(url):
    try:
        # Bypass SSL errors using `verify=False`
        response = requests.get(url, verify=False)
        response.raise_for_status()
        tree = html.fromstring(response.content)
        return tree, None
    except requests.exceptions.RequestException as e:
        return None, str(e)

@app.route('/api/fee-structure-bahria', methods=['GET'])
def scrape_fee_structure_bahria():
    url = 'https://www.bahria.edu.pk/index.php/under-graduate-programs-fee/'
    tree, error = fetch_html(url)
    if error:
        return jsonify({"error": error}), 500

    fee_data = []

    # Locate table rows
    rows = tree.xpath('//tbody/tr')

    # Extract program names from the second row
    program_names = rows[1].xpath('./td[position()>1]/strong/text()')

    # Loop through the rest of the rows to extract fee details
    for row in rows[3:]:
        description = row.xpath('./td[1]/text()')
        if not description:
            continue
        description = description[0].strip()

        fees = row.xpath('./td[position()>1]/text()')

```

```

        program_fees = {program_names[i]: fees[i].strip() if i < len(fees) else None for i in
range(len(program_names))}

    fee_data.append({
        "description": description,
        "fees": program_fees
    })

    return jsonify(fee_data)

def fetch_htmldata(url):
    try:
        response = requests.get(url, verify=False)
        response.raise_for_status()
        tree = html.fromstring(response.content)
        return tree, None
    except requests.exceptions.RequestException as e:
        return None, str(e)

def extract_text(tree, xpath, default=None):
    try:
        result = tree.xpath(xpath)
        return result[0] if result else default
    except Exception:
        return default

@app.route('/api/bahria-data', methods=['GET'])
def get_bahria_data():
    url = "https://www.bahria.edu.pk/"
    tree, error = fetch_htmldata(url)
    if error:
        return jsonify({"error": error}), 500
    # Corrected XPath to extract the logo
    logo_xpath = '//*[@id="tc"]/div/div[1]/a/img/@src'
    logo = extract_text(tree, logo_xpath, default="Logo not found")
    data = {
        "name": "Bahria University",
        "logo": logo
    }
    return jsonify(data)

@app.route('/api/bahria-about', methods=['GET'])
def get_bahria_about():
    url = "https://www.bahria.edu.pk/index.php/vision-mission/"
    response = requests.get(url, verify=False) # Bypassing SSL verification

    if response.status_code != 200:
        return jsonify({"error": "Failed to fetch the webpage"}), 500

```



```

soup = BeautifulSoup(response.content, 'html.parser')

# Find the "About Us" section by targeting the description div
about_section = soup.find("div", class_="description")

if not about_section:
    return jsonify({"error": "About Us information not found"}), 500

# Extract Vision, Mission, and Core Values
vision = about_section.find("h2", string="Vision")
mission = about_section.find("h2", string="Mission")
core_values = about_section.find("h2", string="Core Values")

# Helper function to extract list items under each heading
def extract_list_items(heading):
    list_items = []
    ul_tag = heading.find_next('ul') # Find the next <ul> tag after the heading
    if ul_tag:
        list_items = [li.get_text(strip=True) for li in ul_tag.find_all('li')]
    return list_items

# Extract values for Vision, Mission, and Core Values
about_info = {
    "vision": extract_list_items(vision) if vision else "Vision not found",
    "mission": extract_list_items(mission) if mission else "Mission not found",
    "core_values": extract_list_items(core_values) if core_values else "Core Values not found",
}

return jsonify(about_info)

@app.route('/api/bahria-contact', methods=['GET'])
def get_bahria_contact():
    url = "https://www.bahria.edu.pk/index.php/contact-us/"
    response = requests.get(url, verify=False)

    if response.status_code != 200:
        return jsonify({"error": "Failed to fetch the webpage"}), 500

    soup = BeautifulSoup(response.content, 'html.parser')

    # Find the table containing contact information
    contact_table = soup.find("div", class_="description").find("table")

    contact_info = []

    if contact_table:
        rows = contact_table.find_all("tr")

```

```

# Loop through each row in the table and extract contact details
for row in rows:
    columns = row.find_all("td")
    if len(columns) == 6: # Ensure that we have the expected number of columns
        contact_details = {
            "institution": columns[0].get_text(strip=True),
            "address": columns[1].get_text(strip=True),
            "phone": columns[2].get_text(strip=True).split(":")[1].strip() if "Phone" in columns[2].get_text()
else None,
            "fax": columns[2].get_text(strip=True).split(":")[1].strip() if "Fax" in columns[2].get_text() else
None,
            "google_map": columns[3].find("a")["href"] if columns[3].find("a") else None,
            "office_directory": columns[4].find("a")["href"] if columns[4].find("a") else None,
            "website": columns[5].find("a")["href"] if columns[5].find("a") else None
        }
        contact_info.append(contact_details)

    return jsonify(contact_info)

if __name__ == '__main__':
    app.run(debug=True)

```

Chapter 5

5. Software Testing

5.1 Testing Methodology

The process flow manager is tested for functional errors upon implementation. We will do Unit and Integration Testing, which is the testing of the functional requirements implemented in our A system without taking into account the code, as well as Black Box Testing, which involves Passing randomly picked variables and mapping them against the expected output in a typical Flow. No tools are used; all test cases are completed by hand.

5.2 Test cases

5.2.1. User Registration

<i>Date:</i> 10-Nov-2024	
<i>System:</i> EduQuest	
<i>Objective:</i> Test the user's Registration Functionality	<i>Test ID:</i> 1
<i>Version:</i> 1	<i>Test Type:</i> Unit Testing
<i>Input:</i> Name, Email, Password	
Expected Result: The user will be registered successfully & a success message will be returned	
Actual Result: User Registered Successfully	

Table 1 User Registration

5.2.2. User Login

Date: 10-Nov-2024	
System: EduQuest	
Objective: Test the user's sign-in Functionality	Test ID: 2
Version: 1	Test Type: Unit Testing
Input: Email and Password	
Expected Result: The user will sign in successfully & a success message will be returned.	
Actual Result: The user successfully login to the system.	

Table 2 User Login

5.2.3. Search University

Date: 10-Nov-2024	
System: EduQuest	
Objective: Testing the Search University Functionality	Test ID: 3
Version: 1	Test Type: Unit Testing
Input: The user clicks the Search Button	
Expected Result: Relevant university results are displayed based on the search query.	
Actual Result: The search returns results as observed.	

Table 3 Search University

5.2.4. Compare Universities

Date: 10-Nov-2024	
System: EduQuest	
Objective: Testing the Compare University Functionality	Test ID: 4
Version: 1	Test Type: Functional Testing
Input: University A and University B selected for comparison.	
Expected Result: The system displays a comparison table showing the fee structure, departments, and offered programs for both universities.	
Actual Result: The comparison results are observed.	

Table 4 Compare Universities

5.2.5. User Chat

Date: 10-Nov-2024	
System: EduQuest	
Objective: Testing the User Chat Functionality	Test ID: 5
Version: 1	Test Type: Functional Testing
Input: User A sends a message to User B in an active chat session.	
Expected Result: User B receives the message in real time without delay.	
Actual Result: The messages are sent and received successfully.	

Table 5 User Chat

5.2.6. Admin Sign-in

Date: 10-Nov-2024	
System: EduQuest	
Objective: Testing the Admin Sign-in Functionality	Test ID: 6
Version: 1	Test Type: Functional Testing
Input: Admin's Email and Password.	
Expected Result: The admin will sign in successfully & a success message will be returned	
Actual Result: The admin successfully sign-in to the system.	

Table 6 Admin Sign-in

5.2.7. Approve Registration

Date: 10-Nov-2024	
System: EduQuest	
Objective: Testing the admin's approved registration Functionality	Test ID: 7
Version: 1	Test Type: Functional Testing
Input: Admin click pending registrations button	
Expected Result: Approved registrations should update user status in the database	
Actual Result: Registration requests are displayed correctly, and admin actions (approve/reject) update the database.	

Table 7 Approve Registration

5.2.8. Delete Chat Messages

Date: 10-Nov-2024	
System: EduQuest	
Objective: Testing the admin's delete chat messages Functionality	Test ID: 8
Version: 1	Test Type: Functional Testing
Input: Admin click the manage chat button	
Expected Result: Admin deletes specific chat messages.	
Actual Result: Messages are successfully deleted from the UI and database.	

Table 8 Delete Chat

Chapter 6

6. Software Deployment

This chapter on software deployment aims to put forth the appropriate procedures that are to facilitate the implementation of the developed software solution into the intended environment. This comprehensive chapter furnishes a detailed account of the multifaceted steps integral to rendering the software solution irrational. These include the configuration of hardware and software components, installation procedures, integration of the solution with existing structures where necessary, and carrying out the required testing and verification processes. The ultimate goal is to confirm that the software solution is deployable and usable by the target group, in other words, the software can be adequately delivered.

6.1 Deployment Platform

The platform that will be used for deployment is AWS (Amazon Web Service) since it is a cloud computing platform that provides a range of services.

6.1.1. AWS (Amazon Web Service)

Amazon Web Services is a comprehensive web service provided by Amazon. AWS is a leading cloud computing platform that provides a range of services. Such services include computing power, storage, databases, machine learning, and more. Businesses and developers are allowed to easily build and scale their applications with this. It allows for flexibility, scalability, and efficiency in terms of cost. Hence, it has been widely used in hosting websites, running applications, managing databases, and storing huge data.

6.2 Deployment Process Description

6.2.1. Creating an account on AWS

The screenshot shows the AWS Sign In interface. The 'Sign In' section on the left includes a 'User type (not sure?)' dropdown with 'Root user' selected, an 'Email address' field containing 'gul.mem@gmail.com', and a 'Next' button. Below this is a link for 'New to AWS? Sign up'. The right side features a promotional banner for 'AWS re:Invent' highlighting 'Amazon Nova'.

Figure 34 Creating an account for AWS

6.2.2. Dashboard

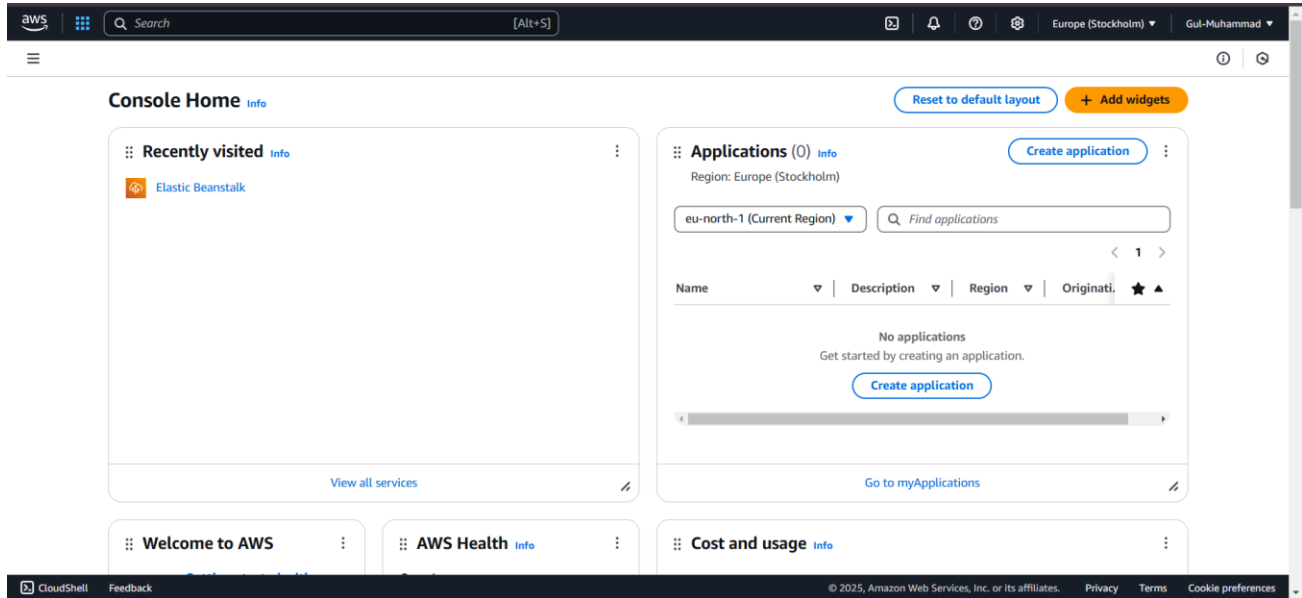


Figure 35 AWS Dashboard

6.2.3. Service (Elastic Beanstalk)

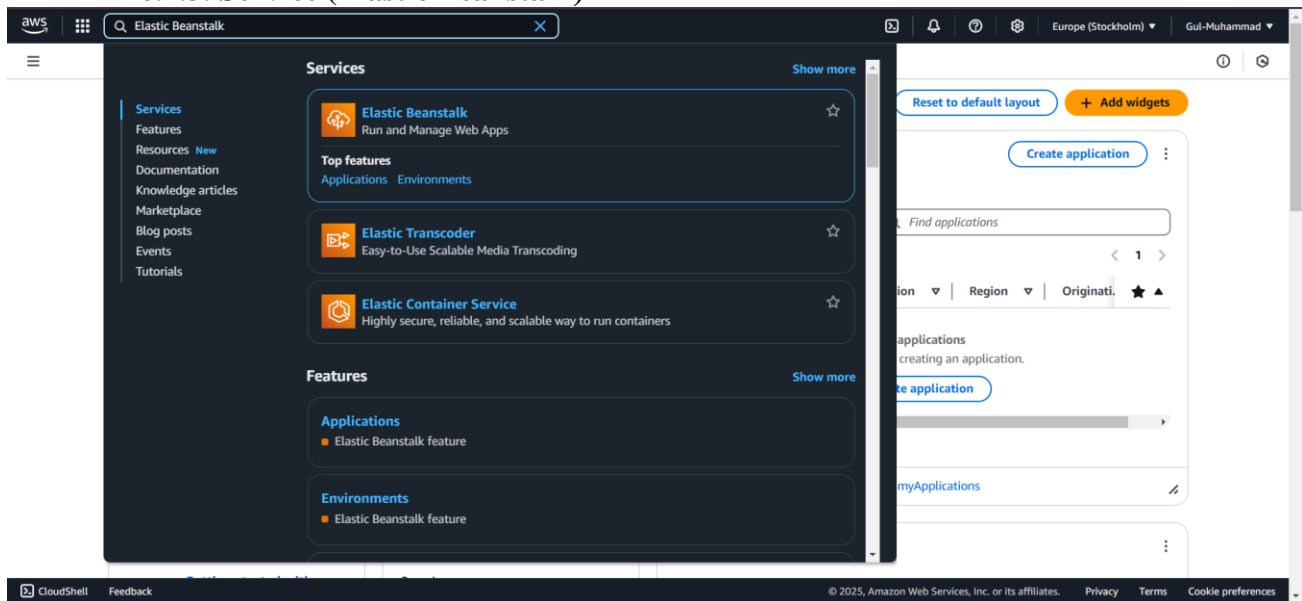


Figure 36 Elastic Beanstalk (Service)

6.2.4. Creating Configure Environment

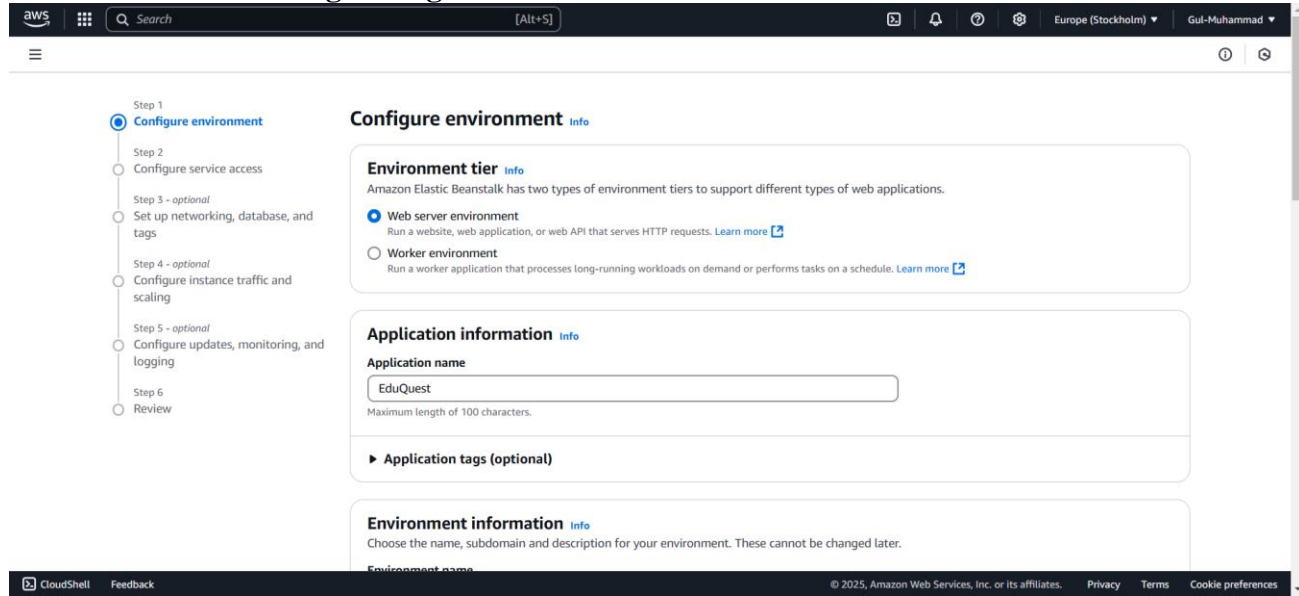


Figure 37 Creating Configure Environment

6.2.5. Creating a .rar file for uploading

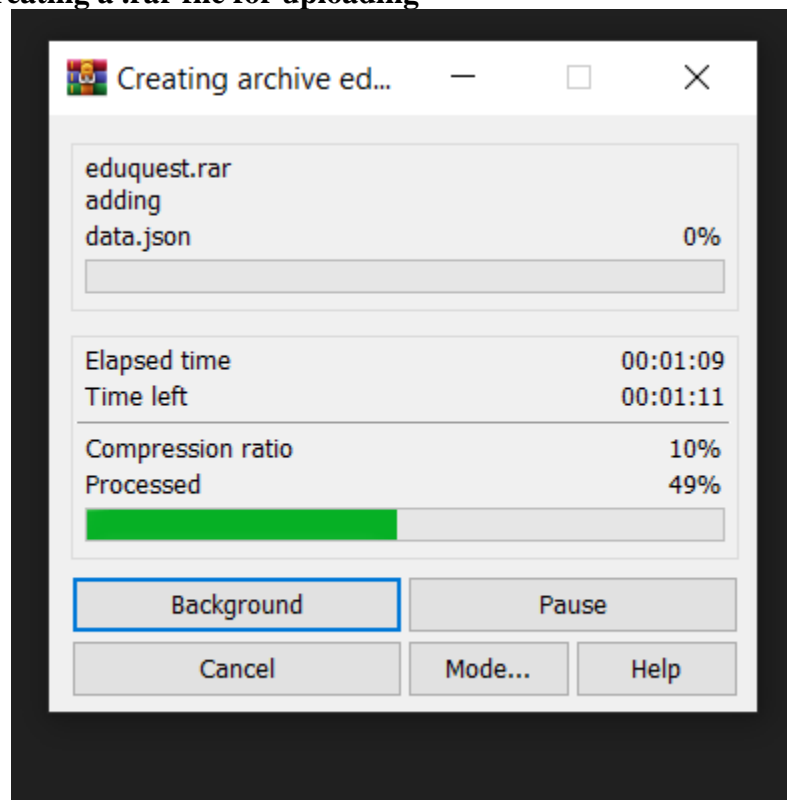


Figure 38 Creating the .rar file of the Project

6.2.6. Uploading the Application Code on the AWS

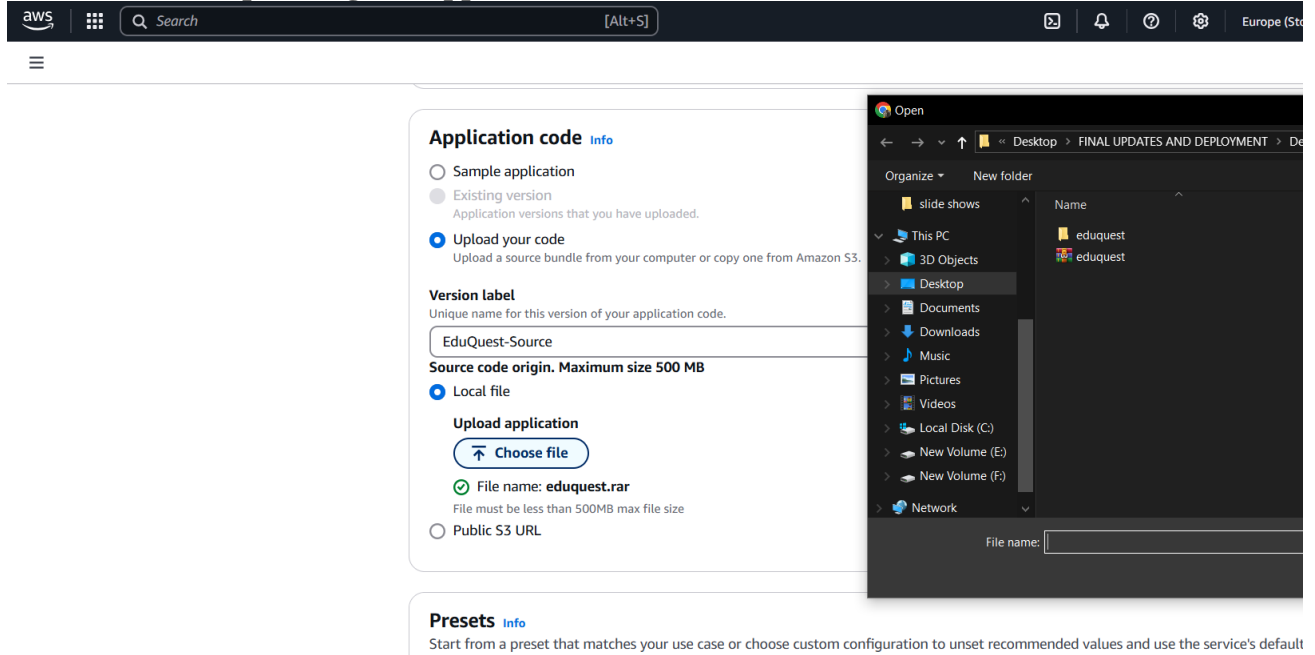


Figure 39 Uploading the Application Code

6.2.7. Creating a Cloud MongoDB Database (cluster)

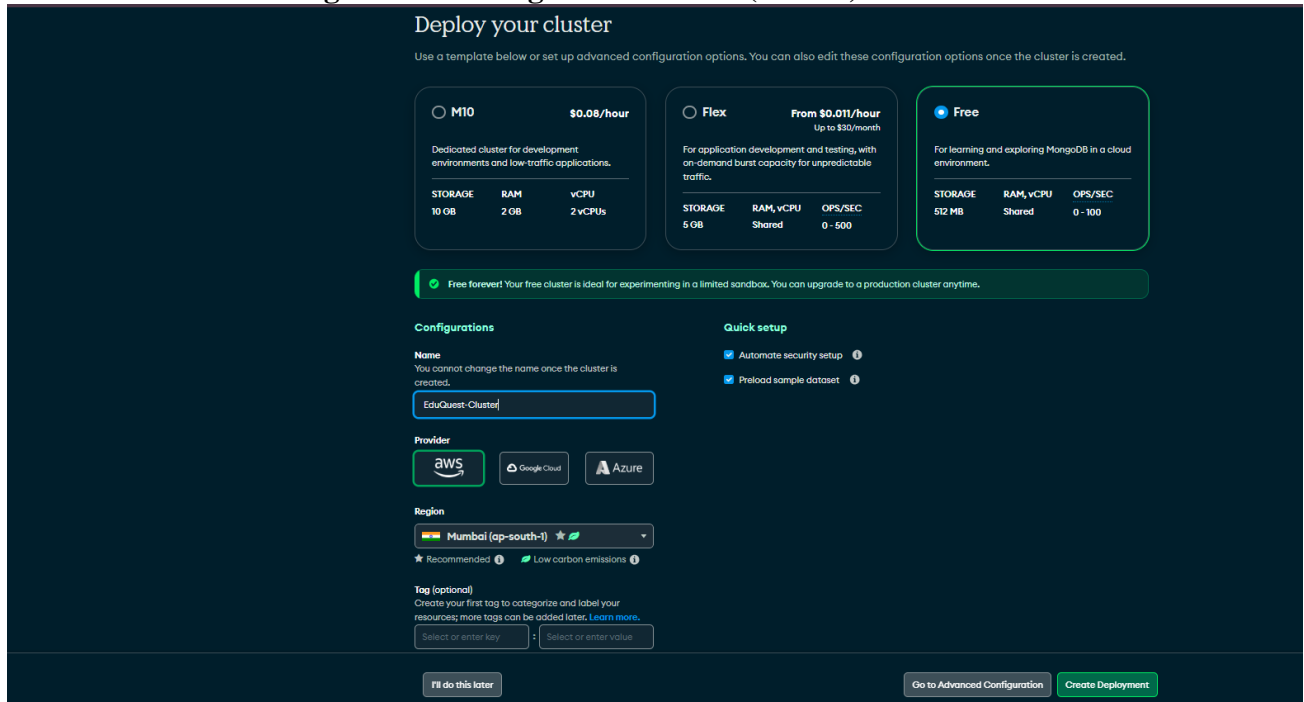


Figure 40 Creating a Cloud MongoDB Database (cluster)

6.2.8. Creating a Database user

Atlas Gul Muham... Access Manager Billing

Project 0 Data Services Charts

Overview

DATABASE

Clusters

SERVICES

Atlas Search

Stream Processing

Triggers

Migration

Data Federation

SECURITY

Quickstart

Backup

Database Access

Network Access

Advanced

New On Atlas 3

1 How would you like to authenticate your connection?

Your first user will have permission to read and write any data in your project.

Username and Password **Certificate**

Create a database user using a username and password. Users will be given the *read and write to any database privilege* by default. You can update these permissions and/or create additional users later. Ensure these credentials are different to your MongoDB Cloud username and password.

Username

GulMuhammad-EduQuest

Password

GulMuhammad-EduQuest

Create User

6.2.9. Database cluster connected with web server

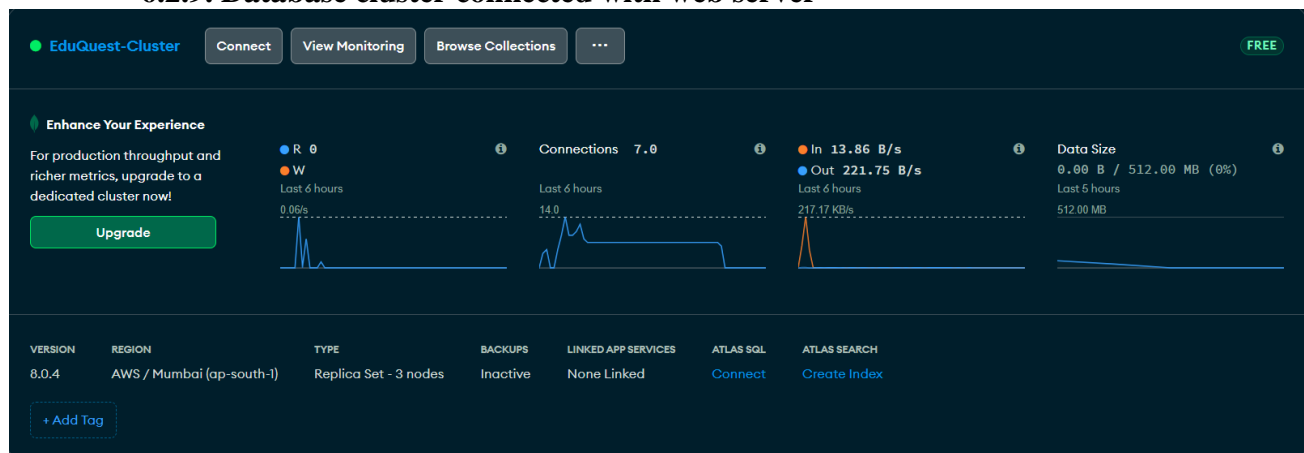


Figure 41 Database cluster connected with web server

References

- [1] D. Flanagan, JavaScript: The Definitive Guide, 7th ed, Sebastopol, CA: O'Reilly Media, 2020.
- [2] R. Dahl, Node.js Design Patterns, 2nd ed, Birmingham, UK: Packt Publishing, 2014.
- [3] I. Sommerville, Software Engineering, Boston: MA: Addison-Wesley, 2015.
- [4] I. Sommerville, Software Engineering, 10th ed., Boston: MA: Addison-Wesley, 2015.
- [5] M. B. W. P. F. E. James Rumbaugh, Object-Oriented Modeling and Design, Englewood Cliffs, NJ: Prentice Hall, 1991.
- [6] J. R. I. J. Grady Booch, The Unified Modeling Language User Guide, Boston: MA: Addison-Wesley, 2005.
- [7] K. E. W. a. J. Beatty, Software Engineering, 10th edition ed., Pearson: Joy Beatty, 2015.

Plagiarism Report

Report Approval Certificate

The project report, “EduQuest Islamabad and Rawalpindi Uni Guide,” has been approved based on the following evaluation guidelines.

Project Evaluation Guidelines

Artifacts Guidelines

Analysis and Design artifacts are syntactically correct (use-case model, SSDs, domain model, class diagram, SDs, ERDs, Flow charts, Activity Diagram, DFDs)

Consistency and traceability have been maintained among different artifacts

General Guidelines

Formatting (font style, indentation) is according to the FYP template and consistent throughout the document

Captions are added to all the figures and tables. Figure captions must be placed below each figure and table captions must be provided above the table Each figure or table is followed by some text describing what it represents

Name & Signature
(Examiner 1)

Name & Signature
(Examiner 2)

Name & Signature
(Examiner3)

Name &Signaturee
(Supervisor)