Table of Contents

1.TITILE

Campus Connect

2.Short Description

Campus Connect aims to foster a vibrant community within colleges by providing a platform for students to share blogs, exchange messages, engage in video calls, and provide anonymous feedback. With ChatGPT integration, the app facilitates idea generation and streamlines communication, enhancing the overall campus experience.

3.Why This Topic

By facilitating open expression and connection among students, Campus Connect addresses the crucial need for fostering a vibrant and supportive community within college campuses. Through the platform's diverse features such as blog sharing, messaging, video calls, and anonymous feedback, students can freely express their thoughts, share experiences, and exchange insights. This not only encourages personal growth and self-expression but also promotes a sense of belonging and inclusivity among peers. Furthermore, the integration of ChatGPT adds an innovative dimension by enabling seamless idea generation and efficient communication, ultimately enhancing the overall campus experience and contributing to students' holistic development.

4.Objective And Scope

The primary objective of Campus Connect is to cultivate a dynamic digital ecosystem within college campuses, empowering students to freely express ideas, connect with peers, and provide feedback anonymously. The platform's scope encompasses a range of features, including blog sharing, messaging, video calls, anonymous feedback, and seamless ChatGPT integration. By fostering a supportive and inclusive campus culture, Campus Connect aims to enhance student engagement, well-being, and personal growth."

5.Methodology

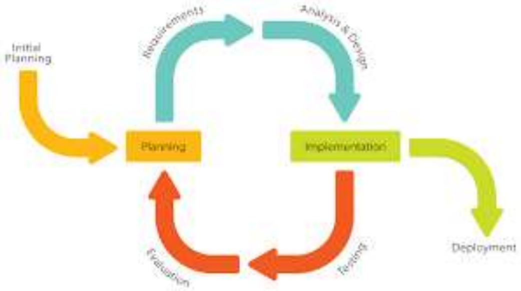
For this system, I would use an iterative approach. The advantages of this approach are:-

1. Feedback from one iteration may improve the other iteration.

2. Increments are delivered and developed

Disadvantages are:-

1. Later, increments may require modification to earlier increments.



6.Architecture

In a 3-tier architecture:
Presentation Tier (Front-end): This is the user interface layer that interacts directly with users. It handles user input, displays information, and communicates with the application server.

Application Tier (Middle-tier or Business Logic): This layer contains the application logic and business rules. It processes user requests from the presentation tier, performs data processing, and communicates with the data tier.

Data Tier (Back-end): This layer manages data storage and retrieval. It stores and retrieves data from databases or other data sources, responding to requests from the application tier.

Each tier is independent and can be developed, maintained, and scaled separately, promoting modularity, flexibility, and scalability in the application architecture

7.Requirement

Hardware Requirements:
Computer: Capable desktop or laptop for development.
Mobile Device: Android or iOS device for testing.

Software Requirements:
Flutter SDK: Cross-platform framework for app development.
Integrated Development Environment (IDE): Android Studio or Visual Studio Code.
Firebase Account: For authentication, real-time database, cloud storage, and messaging.
Firebase CLI: Command-line interface for Firebase management.
Google Cloud Platform Account: Required for ChatGPT integration.
Twilio Account: For video calling.
Git: Version control system for code management.

Platform Requirements:
Android: Compatible with Android 4.1 (API level 16) or higher.
iOS: Compatible with iOS 8.0 or higher.
Firebase: Utilized for authentication, real-time database, cloud storage, and messaging.
Google Cloud Platform: Used for ChatGPT integration.
Twilio: Integrated for video calling functionality.

8.Contribution

Facilitates community building and collaboration among students.
Enables knowledge sharing through blogs and discussions.
Provides support networks and anonymous feedback features.
Promotes inclusivity by giving voice to all students.
Enhances communication between students, faculty, and administrators.
Encourages personal growth and development.
Supports innovation through integrations like ChatGPT.
Addresses issues and improves the campus experience.
Facilitates career development and networking opportunities.
Contributes to a positive and inclusive campus culture.

9.Conclusion

Campus Connect is the heartbeat of college life, where students come together to share ideas, support each other, and grow personally. It's a place where collaboration flourishes, communication flows, and everyone feels valued. With Campus Connect, college isn't just about classes—it's about building a community that lasts a lifetime.

CHAPTER 1
INTRODUCTION TO SYSTEM

1.1 Background

The campus connect app is essential because traditional methods of communication in schools often have limitations. People may feel hesitant to share their thoughts openly due to fear of judgment, or it may be challenging to communicate quickly and effectively with others. This app addresses these issues by providing a centralized platform where students, faculty, and staff can easily connect and communicate.
One significant feature is the ability to share feedback anonymously, allowing individuals to express concerns or ideas without revealing their identity. This fosters a more open and honest dialogue, leading to better understanding and resolution of issues within the campus community.

Moreover, the app facilitates real-time interaction through messaging and video calling functionalities, enabling users to communicate seamlessly regardless of their physical location. This promotes collaboration, teamwork, and a sense of community among members of the campus

Additionally, the integration of AI technology, such as ChatGPT, streamlines communication by helping users generate content more efficiently. Whether it's drafting emails, brainstorming ideas, or writing posts, the AI assistance enhances productivity and saves time for all users.

Overall, the campus connect app aims to improve the school experience by promoting communication, collaboration, and transparency, ultimately fostering a more supportive and engaging campus environment for everyone involved.

1.2 Objectives

Facilitate open communication among students, faculty, and administration.
Provide a platform for anonymous feedback to address concerns effectively.
Foster real-time interaction through messaging and video calling features.
Enhance productivity with AI integration for content generation.
Promote collaboration and community engagement within the campus environment

1.3 Purpose, Scope and Applicability

1.3.1 Purpose

Connect everyone in the campus easily.
Let people share thoughts without worrying about being known.
Talk instantly through messages and video calls.
Write things faster with AI help.
Make campus life more collaborative and friendly.

1.3.2 Scope

The scope of the project is to provide a platform that allows everyone on campus to connect easily. This system is applicable only to V.G. Vaze College.

1.3.3 Applicability

College Socials: Improve communication and interaction among students, faculty, and campus groups.
Classrooms: Facilitate seamless communication between students and instructors, including assignment updates and discussion forums.
Clubs and Organizations: Provide a platform for club members to coordinate events, share updates, and collaborate on projects.
Campus Events: Enhance engagement and networking opportunities during campus-wide events and activities.
Anonymous Feedback: Allow students to express concerns or suggestions anonymously, promoting transparency and inclusivity.
Study Groups: Enable students to easily connect and collaborate on group projects, study sessions, and exam preparations.
Senior Guidance: Access helpful advice and study tips from experienced students through blogs.
Study Direction: Find guidance on course selection and career paths through blogs and resources.
Peer Mentorship: Connect with older students for support and advice on academic and personal matters.
Student Community: Share experiences, challenges, and success stories to inspire and support each other through blogs and forums

CHAPTER 2
SURVEY OF TECHNOLOGIES

2.1 IDE

I. Android studio

It is official integrated development environment (IDE) for developing android app built on jet brains intel J ide. It uses a gradle-based built system Android emulator core template and GitHub integration. It is a one stop place for development of Android app. It supports various programming languages such as Java Kotlin or C++ while also supporting various frameworks it received the latest updates directly from the Google keeping it up to date with the trends it also has support for a large array of third-party plugins. Superior android development project structure code completion and refactoring emulation etc are some of the many advantages and features of Android studio

II. Eclipse

It is an IDE used primarily for Java development however It supports various other language like C, C++, JavaScript to name a few for several years in version of Eclipse with an Android plugin was recommended but Google ceased support for this plugin causing developer to shift to android studio for android development.

III. NetBeans

It was originally developed by a student of Prague University it is similar to Eclipse and primarily known as Java IDE Android development is supported using plugins which are not supported to a large extend today.

IV. Visual Studio Code

Very commonly referred to as vs code It is an open source code editor made by Microsoft with the electron framework for Windows Linux and MacOs it includes vast number of features such as support for debugging syntax highlighting intelligent code completion snippets code refactoring and embedded git user can change the theme customize keyboard shortcuts and preferences and install extensions that add functionalities and notable feature is the ability to create extensions that add support for new language theme debugger etc via plugging such as the Dart plugin which vs code utilizes to develop Dart app for which the futter framework is used. Good and extensive library is also a a very prominent feature of vs code.

Why have I selected android Studio?

Currently I chose Android studio as it the official ide hence receiving continuous recent updates to the ide as well as the plugins it provides all the necessary features while being highly features rich and offers an extensive way of debugging and optimizing the application

2.2. Framework

I. Flutter

Developed by Google it is Open source mobile framework for developing application it simplifies the multi-platform development process to craft excellent native interface flutter is written in Dart language and is a method to implement hybrid app development using a single codebase it uses Google's rendering engine called Skia to develop visuals it puts a hot reload for functionalities which allows for continuous testing without having to restart applications it is used to develop fast high quality applications for Ios and Android in record time from a single code base.

II. React Native

It is an open-source framework by Facebook using react to create Android and ios app with a single code base it leverage native components and API for excellent performance reducing development time and cost and elegant interface across platform.

III. Xamarin

It is an open source platform for building modern IOS Android and windows app with .NET with a friendly environment and abstraction layer, it enables writing business logic in one language for native performance and look on each platform its standout features is using .NET and C# for cross platform apps.

IV. Ionic

It is a developer friendly framework for building hybrid and interactive mobile app ideal for web developers with a rich set of elements gestures and tools it enables high quality mobile desktop and progressive web apps from one code base integrating seamlessly with AngularJS it supports functionalities like Bluetooth and fingerprint authentication with cordova plugins for easy access to device features like camera and GPS. Utilizing emulators like reload and logging it ensures excellent performance using HTML, CSS and javascripts.

V. Apache Cordova

It uses HTML, CSS and JavaScript to build mobile app offers access to device features like the camera and GPS through pre-defined plugins. Developers can create cross platform app without rewriting for each platforms language resulting in hybrid application while not as fast as native apps they offers expanded functionalities with JS through they aren't packaged for distribution like web apps.

Why have I selected React Native?

I selected React Native for your "Campus Connect" application because it allows for cross-platform development, so that we can build both iOS and Android apps with a single codebase. It's fast, efficient, and has a strong community, which helps speed up development while providing a smooth user experience on both platforms.

2.3. Database

I. MySQL

It is the most popular open-source relational nature helps to organize the data into one or more data tables. The structured data can be inserted extracted and modified its major features are its ability to manage user allows for network access facilating testing and creation of backups. Query caching, Unicode support, multiple storage engines, SQL support are some of its prominent features.

II. PostgreSQL

PostgreSQL is an open-source database system used to store, manage, and organize data. It's known for being powerful, flexible, and reliable, supporting both simple tasks and complex applications. It works with different types of data, supports custom functions, and ensures data is safe and consistent. Many businesses and developers use it for web applications, analytics, and more.

III. SQLite

SQLite is a lightweight, open-source database system that stores data in a single file. It's easy to use, requires no setup or server, and is great for small applications or projects like mobile apps and websites. Since it's built into many devices, it's popular for storing simple data locally without needing a large database system.

IV. Firebase

Firebase is a cloud-based platform by Google that helps developers build and manage apps easily. It provides tools for storing data, managing users, and sending notifications, all without needing to set up a server. Firebase is great for mobile and web apps, offering real-time updates and simple integration, making it popular for small to medium-sized projects.

V. AWS Dynamo DB

AWS DynamoDB is a cloud-based NoSQL database service provided by Amazon. It is designed to handle large amounts of data with fast performance and automatically scales to meet demand. DynamoDB is great for applications that require low-latency and can store different types of data, like user activity, chat messages, or shopping cart details, without needing a traditional database structure. It's widely used for web, mobile, and gaming apps.

VI. MongoDB

MongoDB is a popular open-source NoSQL database that stores data in a flexible, JSON-like format. It allows you to store and manage large amounts of data without needing a fixed structure, making it great for dynamic or unstructured data. MongoDB is often used in web and mobile applications, offering fast performance, easy scaling, and support for complex queries. It's ideal for projects that require flexibility and rapid development.

Why I am using MongoDB for my application?

I’m using MongoDB for my application because it handles large amounts of flexible, unstructured data efficiently. MongoDB allows easy scaling and fast performance, which is ideal for managing dynamic content like student information, events, or messages in my app.

2.4. Other

Twilio

Twilio provides an easy-to-integrate platform for adding video calling features to your application with twilio we can enable high quality scalable video calls handles user authentication and manages call connections all through simple APIs making it ideal for enhancing communication capabilities in app like Campus Connect

---

CHAPTER 3

REQUIREMENTS AND ITS ANALYSIS

3.1 Problem Definition

The campus connect app addresses the limitations of traditional communication methods in educational institutions. It aims to overcome barriers to open expression and effective communication among students faculty and staff by providing a centralized platform the app facilitate seamless interaction through messaging and video calling features additionally, it enables anonymous feedback promoting honesty and transparency in dialogue. Integration of AI technology further enhances productivity by assisting users in generating content efficiently. Ultimately the app seeks to foster a supportive and engaging campus environment by promoting communication collaboration and transparency.

3.1.1 Sub-Systems

I. Login/Registration

a. The users would have to register for the first time.

b. Once the user are registered successfully and authentication they can access the rest of the app.

c. Once logged In the user don’t need to log in every time they wish to use the app i.e. they remains logged in unless they explicitly log out.

II. Profile Management

a. The users can view their information and also edit it.

III. Share Post/Blogs

a. Campus Connect app lets you instantly share your blogs with peers sparking instant engagement
b. With a simple interface posting on campus connect is a breeze enable seamless sharing of your ideas.

IV. Engage with blogs (Like and Comments)

a. Users can interact with blogs by liking and commenting fostering dynamic discussion.
b. Express your view with just click or a comment making feedback easy and immediate.

V. Communication(messaging)

a. Chat one-on-on or in groups for discussion and collaboration.

b. Share images videos and documents within conversation.

c. Using search functionality you can easily find past conversation and information.

VI. View/Explore Blogs

a. Explore a diverse range of blogs
b. Quickly find relevant blogs through search and filters.

VII. AI Talk

a. Explore a diverse range of blogs
b. Quickly find relevant blogs through search and filters.

VIII. Anonymous Messaging

a. Send message without revealing your identity.
b. Encourage honest communication and ensure privacy and safety.

3.2 Requirement Specification

3.2.1. Requirement Gathering

The various ways to gather requirements are:-

1. **Survey/Questionnaire**
   Questionnaires and survey are basically a set of questions used to gather stakeholder input efficiency saving time by reaching multiple parties simultaneously. However they lack flexibility for adjusting or adding questions based on responses.
2. **Interviews**
   Interviews are usually conducted one-on-one interviews offers flexibility for follow-up questions facilitating detailed information gathering. However they can be time consuming and require interviews with deep system knowledge
3. **Brainstorming**

This method explores all system perspective considering various scenarios including what if situation and innovative ideas. It aims to break from conversions to gather detailed system requirements role-play further enhances understanding by simulating different roles and scenarios.

4. **User Observations**
   User observations is one of the best ways to fully understand and discover how people and technology in the current system operate and behave. It gives us a realistic idea of how things actually works It can be categorized into active and passive. Active observation takes place when the people being observed are questioned about the action that they carry out to gain a better understanding of the process. Passive observations are better at getting feedback without any communication involved.
5. **Prototyping**
   This method allows users to experience their system firsthand leading to continuous feedback and iterative improvement. Prototypes are reverse engineered to uncover requirements and stakeholder satisfaction.

Methods I used and Why?

I gathered input from college students using survey/questionnaires. We brainstormed to tackle problems and imagine scenarios. I used google form for the survey and conducted interviews to understand what students wanted the app to look like and what features it should have.

The interview questions included:

What features do they want in the app?

Based on interview with college students the key features they wants are:

- Easy user interface
- Allow sharing of images along with blogs
- Enhanced communication tools
- Resource sharing section
- Capable of forming group of same interest peoples

How does the app helps establish a better connection among students?

The common answer what I get from students is that this app is the only platform where they can easily find and connect with each other here people share their blogs which helps students find others with the same interest. From the same platform students can connect using messaging features.

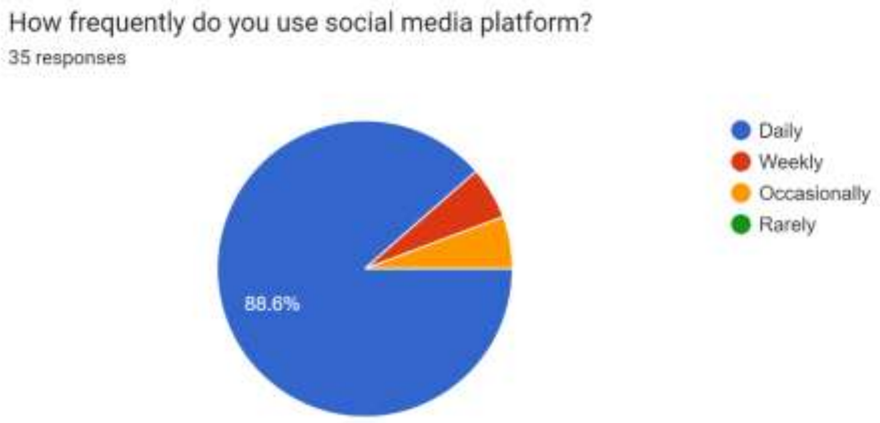
How AI can be integrated in the app?

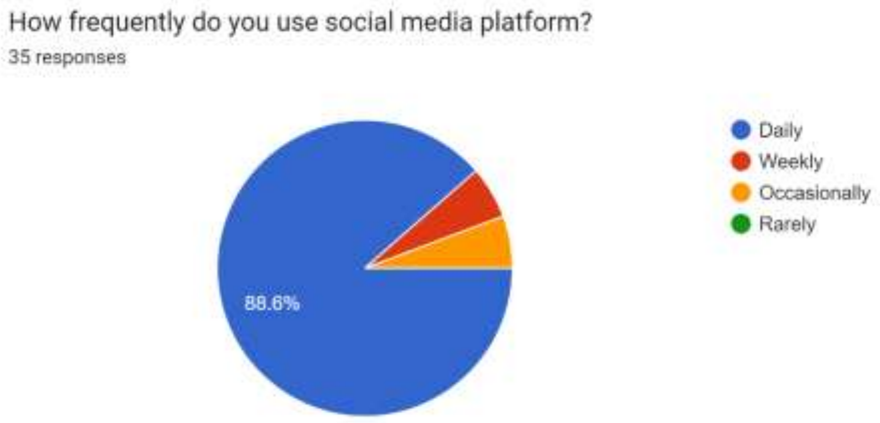
The common answer I get from students is that AI can improve user experience in several ways like

- Helps in writing blog and suggestion resources to refer
- Helps in answering the doubt comes in mind during writing or reading some blogs.

Result of Survey

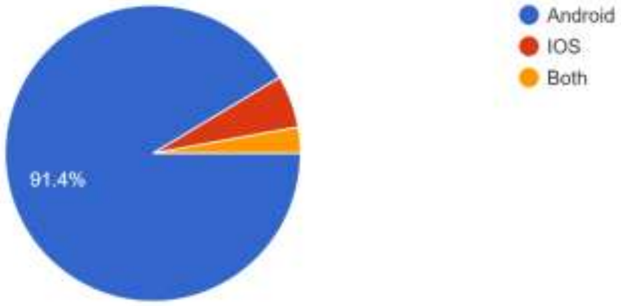
Link of Google form : https://docs.google.com/forms/d/1jkwU0OreZka3Ph2BBfbWJ3xmWjJYSnW7K3ylasZQzDw/edit





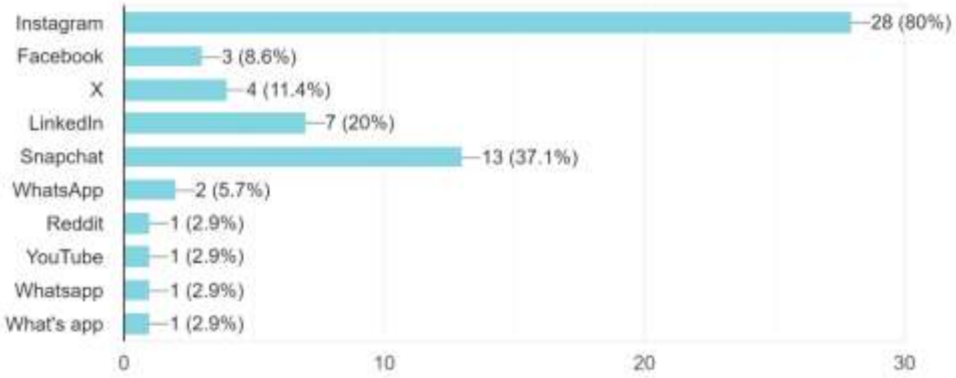
What type of operating system do you use?

35 responses



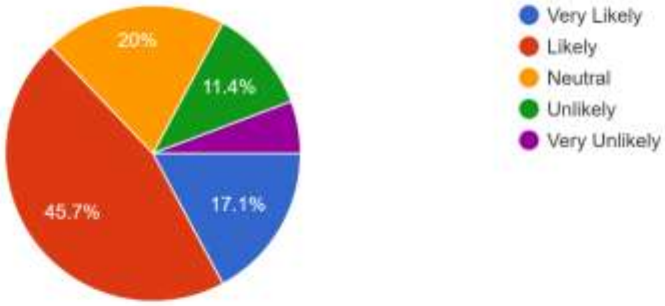
Which social media platform for you use most often?(Check all that apply)

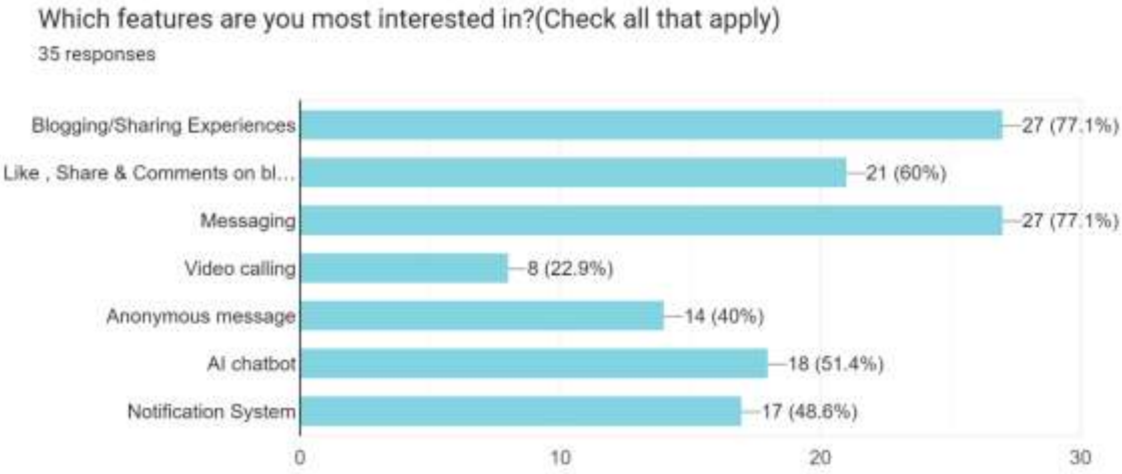
35 responses



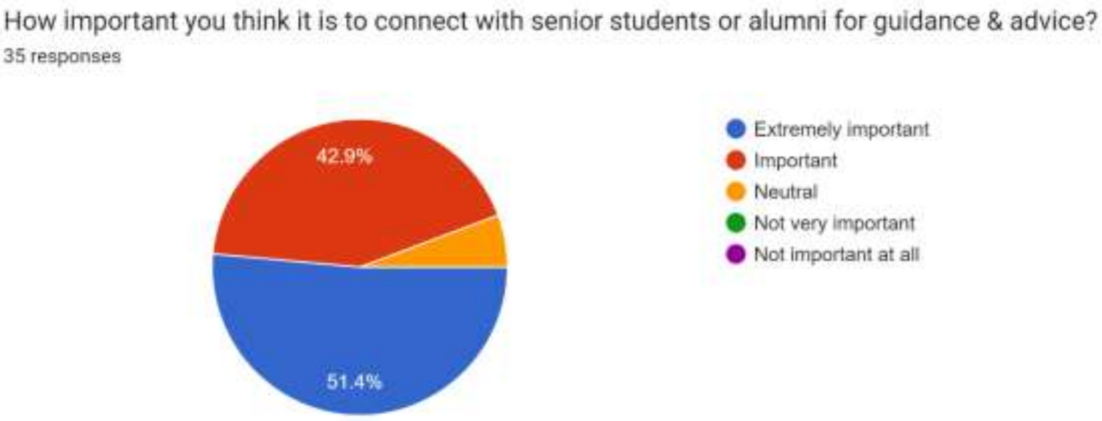
How likely are you to use a campus focused social media platform like campus connect?

35 responses



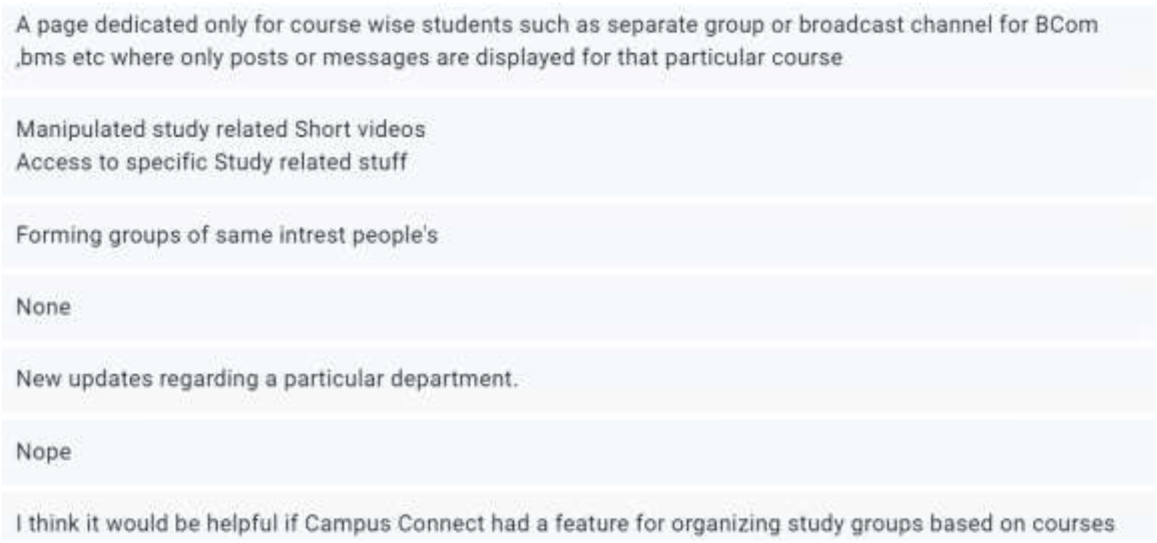

Which features are you most interested in?(Check all that apply)

35 responses


How important you think it is to connect with senior students or alumni for guidance & advice?

35 responses

Are there any additional features you like to see in campus connect?


A page dedicated only for course wise students such as separate group or broadcast channel for BCom ,bms etc where only posts or messages are displayed for that particular course

Manipulated study related Short videos
Access to specific Study related stuff

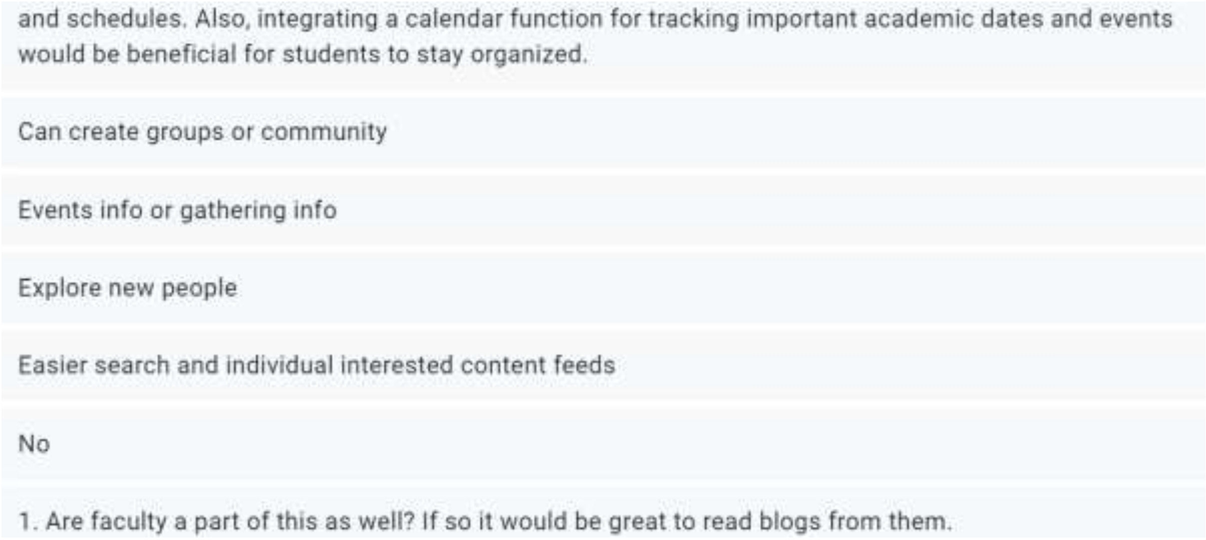
Forming groups of same intrest people's

None

New updates regarding a particular department.

Nope

I think it would be helpful if Campus Connect had a feature for organizing study groups based on courses



and schedules. Also, integrating a calendar function for tracking important academic dates and events would be beneficial for students to stay organized.

Can create groups or community

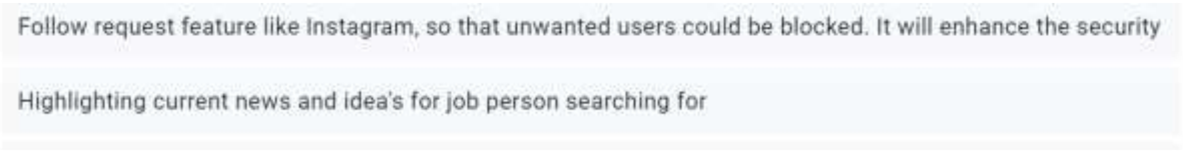
Events info or gathering info

Explore new people

Easier search and individual interested content feeds

No

1. Are faculty a part of this as well? If so it would be great to read blogs from them.



Follow request feature like Instagram, so that unwanted users could be blocked. It will enhance the security

Highlighting current news and idea's for job person searching for

REQUIREMENT ANALYSIS

All the requirements are stated in simple English statement:

1. **Students and faculty of V.G. Vaze college are user of this system**

2. **User Registration**
    a. The user if they are students should be able to register using their control id
    b. The faculty member would have to register using their email which is unique and contains @vazecollege.net
    c. The registration page should have options to separate students and faculty registration
    d. The password should be a minimum of 8 character.
    e. The students control ID and the faculty's email are used as username in the system.

3. **User Login**
    a. The user will have to enter their credential like registered email or control id depending whether there are a faculty or a student respectively.
    b. The user should have the options of "forgot password"
    c. The user need not login every time they visit the app until they explicitly log out.
    d. The user should be displayed a appropriate message of authentication fails.

4. **Posting blogs or Images**
    a. Users can create and publish blog posts or upload images.
    b. The users should be able to upload images directly from the device.
    c. Choose who can view the post (Student, faculty, everyone, specific group).
    d. Users can comment on post and like posts.
    e. Receive alert for comments and likes on their posts or images.

5. **Viewing posts and blogs**
    a. Posts are presented in a clear and readable format.
    b. Users are only able to see posts whom they follow.
    c. Posta are arranged according to their date and popularity.
    d. Users must be able to like comment on them and share them with others.
    e. Users have options to search for specific post or filter content by categories or tags.

6. **Profile**
    a. Each user have a personalized profile displaying their information
    b. Users can follow other profile to stay updated with their activities.

c. Display the count of followers and profile followed by users
d. Option to edit profile details and profile picture
e. User can view all their posts
f. Users must have options to edit and delete posts.

7. **Communication (Messaging)**
   a. User can send private messages to other users.
   b. User must be able to create and participate in group chats with multiple users.
   c. Users must have option share images videos and files within messages.
   d. Users also able to see the status of message sent delivered and read.

8. **Others**
   a. Have an AI section which suggest topics for blogs and helps with structing and editing.
   b. Have anonymous message section where users can send message without revealing identity.

3.2.2.1. Functional Requirements

1. **Login/Registration**

   The new user should register first i.e. create an account after creation they will have to enter their valid username and password and proceed further.

2. **Posting blogs/Images**

   Users can create and share blogs and images customize posts with formatting options manage visibility setting and interact with content through like and comments.

3. **Viewing blogs or posts**

   Users can browse and discover posts and blogs search filter content interact by liking and commenting and explores update from other users.

4. **Manage profile**

   Users can personalize their profile with details and a profile picture follow other users to stay updated manage their own posts.

5. **Messaging**

User can send direct messaging to individuals or participate in group chats receive notifications for new messages share media like images and files.

6. **Manage Posts**
   Users can view edit and delete their own posts ensuring they have full control over the content they share on the platform.

7. **Anonymous messaging**

8. **AI talk**

3.2.2.2. Non-Functional Requirements

1. **User Friendliness**
   The app would implement proper navigation to access various activities/features. The titles must be self -explanatory and easily understandable to the user. The UI must be catchy attractive and would use animation.

2. **Maintainability**
   The application would be well-documented the application code would contain consistent nomenclature for variables. The code would be readable and comments would be used to inform the reader about why that code was implemented.

3. **Security**
   The password stored in database would be encrypted. Options for user to control visibility of their posts and messages.
   User info and messages are also in encrypted form.

4. **Portability / Compatibility**
   The application would be compatible with older android version.

5. **Performance**

Fast response times for app for that use caching strategies optimize image size and format use lazy loading images implement native plugins or existing plugins for specific performance etc

6. **Scalability**
   Use scalable backend service and database
   Use caching
   Optimize code
   Use asynchronous programming
   Use google analytics to gain insight into user engagement.

System Requirements

1. **User Registration**
   Description: Allow users to create an account.
   Source: User input via registration form
   Output: User account created in the database
   Destination: User Database
   Action: User enter details (Name, control id/email, password...)
   Pre-condition: User navigate to the registration page and must be the part of college campus (Student or faculty).
   Post-condition: User account is created and user is redirected to login page.
   Exception: if the user is already registered then display error message and redirected to the login page or if the control id or the faculty main is incorrect then show the appropriate error message.

2. **User Login**
   Description: Allow users to login to their account.
   Source: User input via login form
   Output: User authentication status
   Destination: User session
   Action: User enter credential and press login button
   Pre-condition: User has a registered account and is on login page.
   Post-condition: User is authenticated and redirected to landing page.
   Exception: if the credential are incorrect then display an error message and prompt the user to try again.

3. **User Profile**

Description: Allow users to view and update their profile.
Source: User input via profile form
Output: User data is updated in database.
Destination: User database
Action: change the details or add new and press update button.
Pre-condition: User is on update profile page.
Post-condition: Data is updated in database.
Exception: Enter incorrect type of data or leave mandatory field then display error accordingly.

4. Upload Posts/Blogs

Description: Allow users to upload posts/blogs.
Source: User input via post form
Output: Post details is added in database.
Destination: User database
Action: Enter the post details like image/blogs visibility etc and enter post button.
Pre-condition: User is on upload post page.
Post-condition: Data is updated in the database and shown to the targeted audiance.
Exception: If user leave the mandatory field or fill incorrect data then show error accordingly.

5. Messaging

Description: Allow users within the app to send messages to each other.
Source: User (sender) within the app.
Output: Message content (text/image/attachments) sent to the targeted user.
Destination: Other user within the app.
Action: User format a message select the user to sent and send it.
Pre-condition: Both sender and recipient must have valid user accounts within the app .
Post-condition: Message is delivered to target user inbox.
Exception: If the recipient has no internet connection the message might be queued if the server is unavailable the message might not be sent.

6. Anonymous Messaging

Description: Allow users to send anonymous messages to a public forum within the app.
Source: Sender within the app.
Output: Content displayed in a public forum.
Destination: Public forum accessible by all users within the app.
Action: User compose a message and send it anonymously.
Pre-condition: User must be logged in to the app.
Post-condition: Message displayed in the public forum.
Exception: If server is unavailable message might not be send.

7. Liking and Commenting

Description: User can like or leave comments on existing posts and blogs.
Source: User within the app.
Output: Adds your like to the post/blogs increasing its popularity and your comment is displayed publicly under the post/blogs.
Destination: The specific posts or blogs you choose to interact with.
Action: You click the like button or write and submit a comment.
Pre-condition: The post/blog must be publicly accessible.
Post-condition: The number of likes is updated. Your comment appears publicly with your username.
Exception: If the server is overloaded or unavailable your action may be discarded or delayed.

8. Logging Out

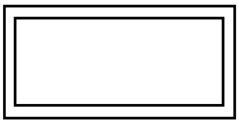
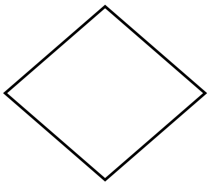
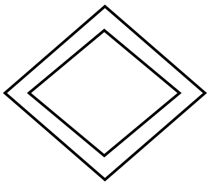
Description: User initiates a logout process.
Source: User within the app.
Output: app session ends.
Destination: Ends your current session.
Action: Press logout button.
Pre-condition: You must be currently logged in.
Post-condition: User session is terminated.
Return to app login screen.
Cached data from your session might be cleared.
Exception: Due to some technical issue the logout process might fails.

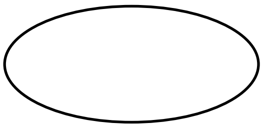
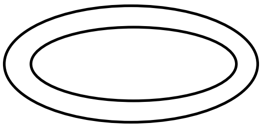
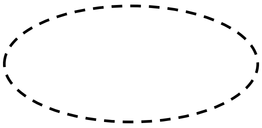
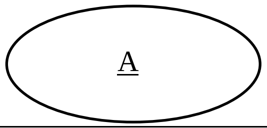
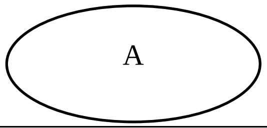
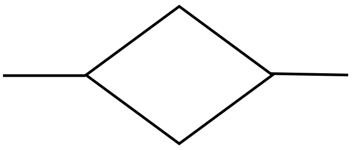
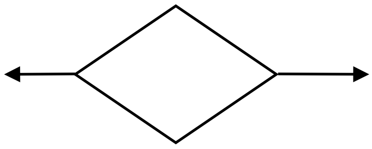
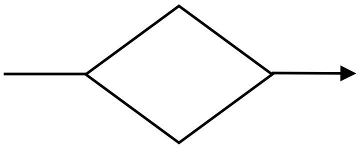
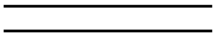
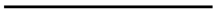
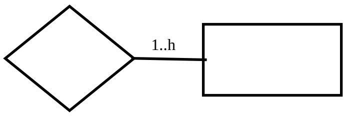
CHAPTER 4
SYSTEM DESIGN

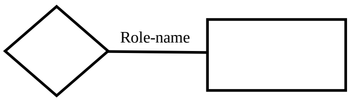
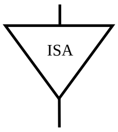
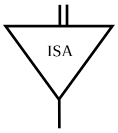
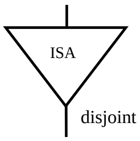
4.1. Entity Relationship Diagram

An Entity-Relationship Diagram is a visual representation used to model the structure of a database. It illustrates the relationships between different entities, which represent real-world objects, concepts, or information within the database system. ER diagrams consist of entities, attributes which are the properties of those entities, relationships which are the connections between those entities, and cardinality among those entities. ER diagrams are essential tools for database design, helping developers and stakeholders understand the logical organization of data, identifying key relationships, and ensuring the proper implementation of a database schema. By providing a clear and intuitive overview, ER diagrams play a crucial role in improving communication and ensuring the accuracy of database systems in various domains, such as business, software development, and data management. ER model becomes an abstract data model that defines a data or information structure which can be implemented typically in a relational database.

4.1.1. Diagram Notations:

Name	Symbol	Description
Rectangle		Entity set
Double Rectangle		Weak Entity set
Diamond		Relationship set
Double Diamond		Identifying relationship set for weak entity set

Eclipse		Attribute
Double Eclipse		Multivalued attribute
Dotted Eclipse		Derived attribute
Primary key		Primary key
Discriminator		Discriminating attribute of weak entity set
Many-to-many		Many-to-many relationship
One-to-one		One-to-one relationship
Many-to-one		Many-to-one relationship
Double line		Total participation of entity set in a relationship
Line		Links attribute to entity set or represents Partial participation of entity set in a relationship.
Mapping cardinality		Cardinality limits

Role indicator		Role indicator
ISA		ISA (specialization or generalization)
Total generalization		Total generalization
Disjoint generalization		Disjoint generalization

4.1.2. Entity Sets:

1. Users
2. Admin
3. Posts
4. Comments
5. Message
6. Anonymous message
7. Students
8. Group
9. Group message
10. Notifications
11. Attachment
12. Sessions
13. Report

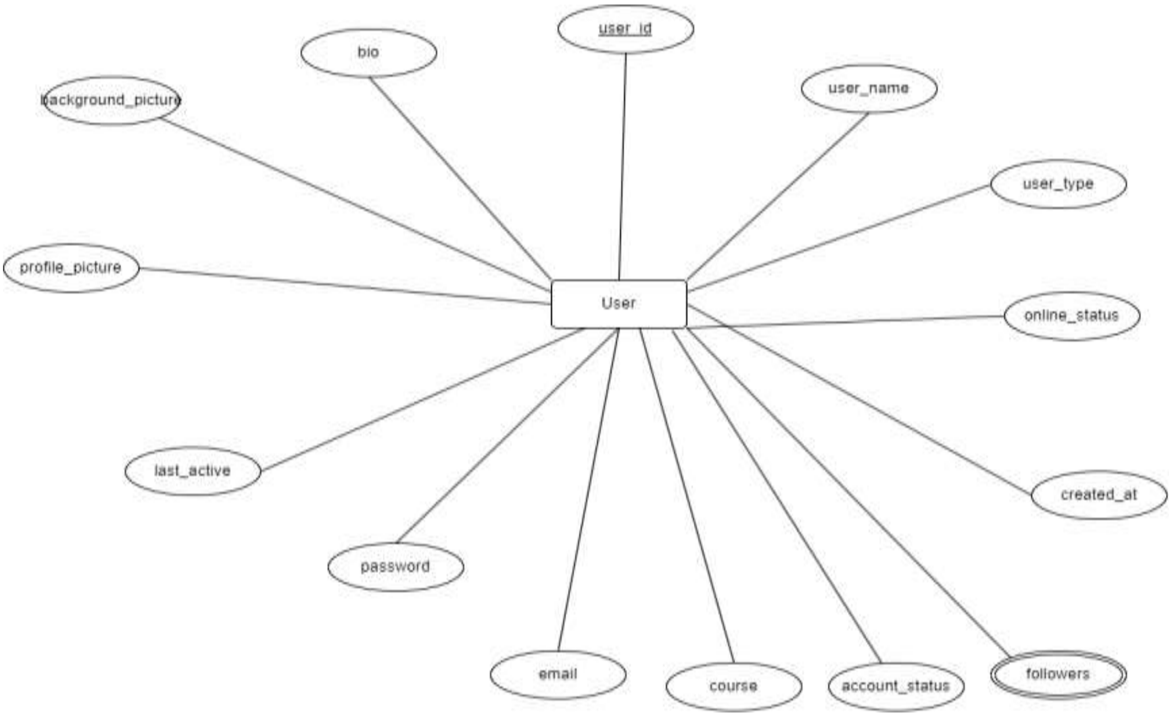
1. Users

The entity represents the user (Student and faculty) of the application.

Attributes:

- **User_id** [INT, Primary key, Simple, Single-valued, not-null]: Unique identifier for each user.
- **Email** [VARCHAR, UNIQUE, not-null, simple, single-valued]: email address of the user.

- **Password** [VARCHAR, Simple, Single-valued]: hashed password for user authentication.
- **Profile_picture[**VARCHAR, Simple, Single-valued, null]: URL of the users profile picture.
- **Background_picture** [VARCHAR, Simple, Single-valued, null]:URL of the user background picture.
- **Bio** [TEXT, Simple, Single-valued, null]: A short biography of the user.
- **Course** [Varchar, Single-valued, simple, not-null]: course name what student pursuing or faculty is teaching.
- **Created_at** [TIMESTAMP, single-valued, simple, not-null, default value- current timestamp]: timestamp when user account was created.
- **Username** [Varchar, simple, single-valued, not-null]: full name of the user.
- **User_type** [enum('faculty', 'student'), simple, single-valued, not-null]: type of user.
- **Account_status** [enum('active', 'deactivate', 'suspended'), single-valued, simple, not-null]: It represents whether the account is suspended or active or deactivated.
- **Online_status** [Boolean, single, simple Default False] It stores the online status of the user.



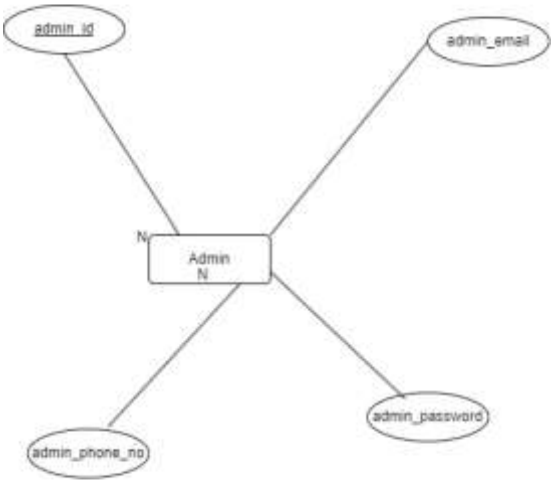
2. Admin

The entity represents the admin of the application

Attributes:

- **Admin_id** [int, simple, single-valued, unique, not-null, primary key]: unique identifier for each admin.

- **Admin_email** [varchar, simple, single-valued, unique, not-null]: email of the admin.
- **Admin_password**[varchar, single-valued, simple, not-null]: hashed password of the admins account to login.
- **Admin_phone** [varchar, single-valued, simple, not-null]: phone no. of the admin.

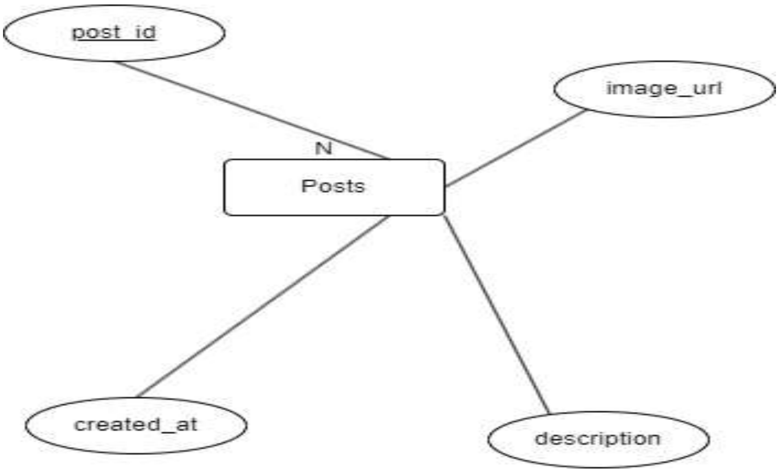


3. Posts

This entity represents the photos and blogs shared by users.

Attributes:

- **Post_id** [int, primary key, auto-increment, simple, single-valued, not-null]: unique identifier for each posts.
- **Image_url** [varchar, simple, single-valued, null]: URL of the image.
- **Description**[int, simple, single-valued, null]: description provided by the user.
- **Created_at**[timestamp, simple, single-valued, not-null default-value- current timestamp]:the timestamp when the post was created.

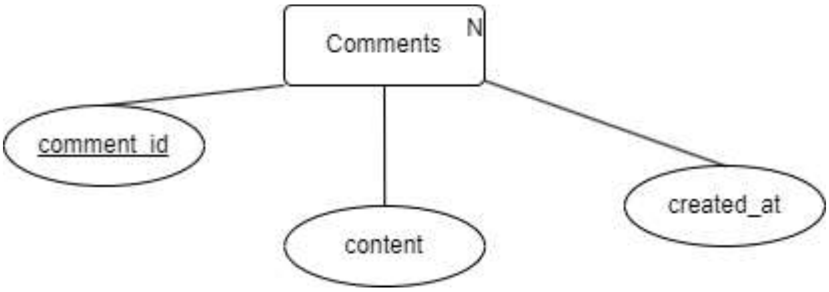


4. Comments

The entity represents the comments made on posts.

Attributes:

- **Comment_id**[int, primary key, auto-increment, simple, single-valued, not-null]: unique identifier for each comments.
- **Content**[text, simple, single-valued, not-null]: The content of the comment.
- **Created_at**[timestamp, simple, single-valued, not-null, default value- current timestamp]: the timestamp when the comment was created.

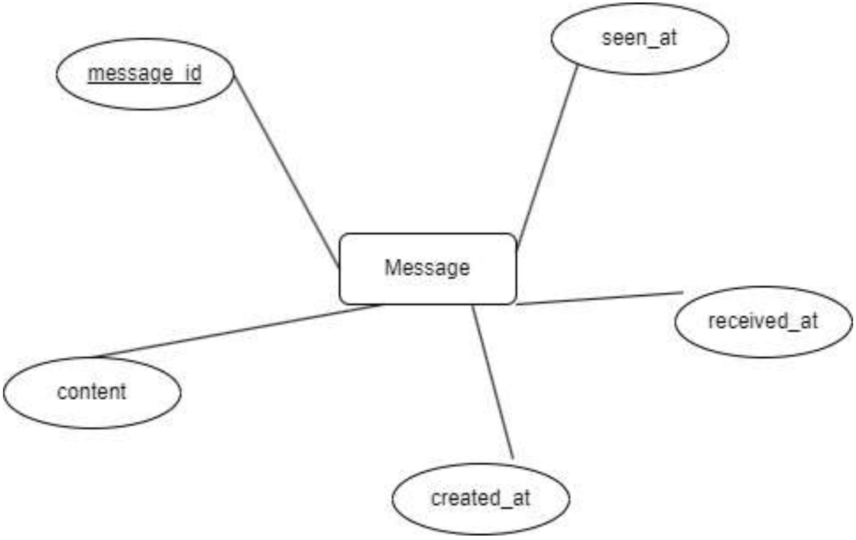


5. **Messages**

The entity represents the private messages exchanged between users.

Attributes:

- **Message-id**[Int, primary key, simple, single valued, not-null]: unique identifier for each message.
- **Content** [text, simple, single-valued, not-null]: content of the message.
- **Created_at** [timestamp, simple, single-valued, not-null, default value- current timestamp]: timestamp when the message was sent.
- **Received_at** [timestamp, simple, single-valued]
- **Seen_at[** timestamp, simple, single-valued]

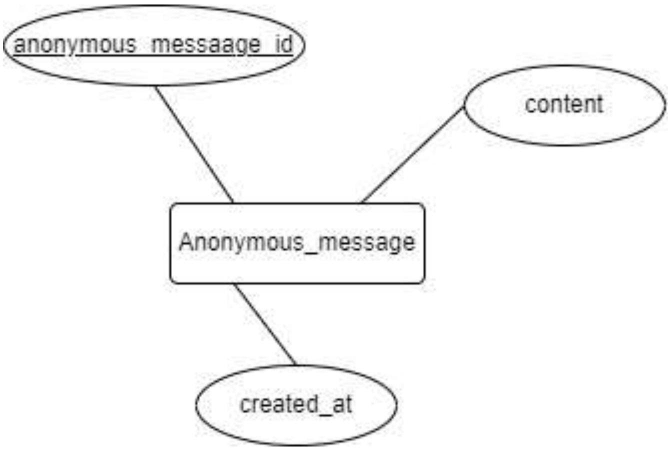


6. Anonymous Message

This entity represents the info of anonymous message.

Attributes

- **Message_id**[int, simple, single-valued, primary-key]: unique identifier for each anonymous message
- **Message_content** [text, simple, single-valued, not-null]: content of the message.
- **Created_at**[timestamp, simple, single-valued, default – current timestamp]: message creation time.

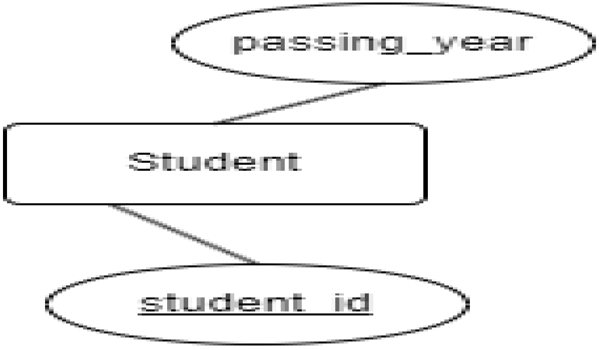


7. Student

The entity represents student info.

Attributes:

- **Student_id**[varchar, simple, single-valued, primary-key]: control id of the student provided by the college.
- **Passing_year**[timestamp, simple, single-valued, not-null]: year in which student completes his/her course.

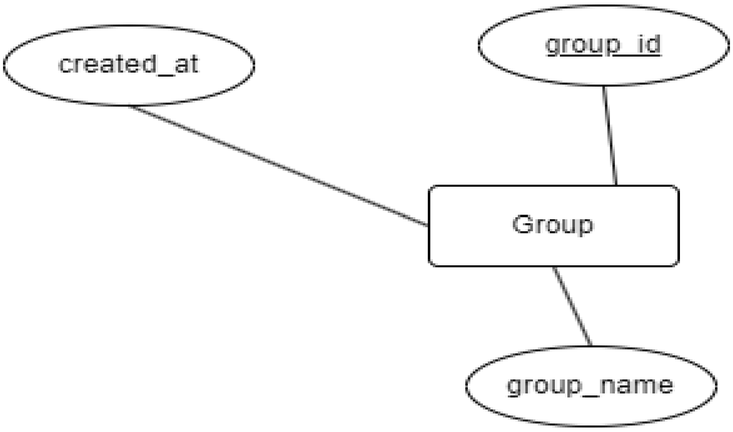


8. Group

This entity represents the general info of group.

Attributes

- **Group_id**[int, simple, single-valued, primary-key, not-null]: unique identifier for each group.
- **Group_name**[varchar, simple, single-valued, not-null]: name given to the group.
- **Created_at**[timestamp, simple, single-valued, not-null, default – current timestamp]: time of group creation.

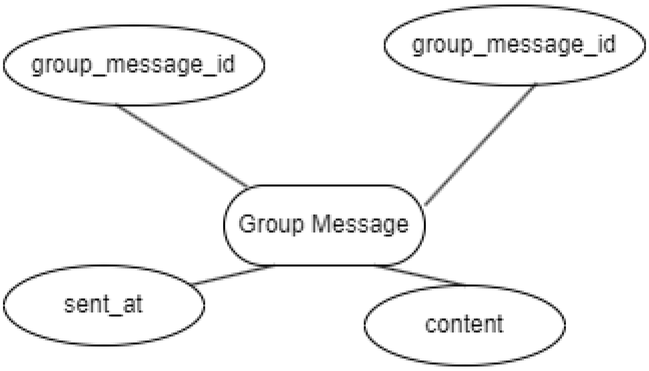


9. Group message

The entity represents the group message.

Attributes

- **Group_messsaage_id** [primary key, simple, single-valued, int]: unique identifier for every message.
- **Content** [text, simple, single-valued]
- **Sent_at**[timestamp, simple, single-valued, default – current timestamp]

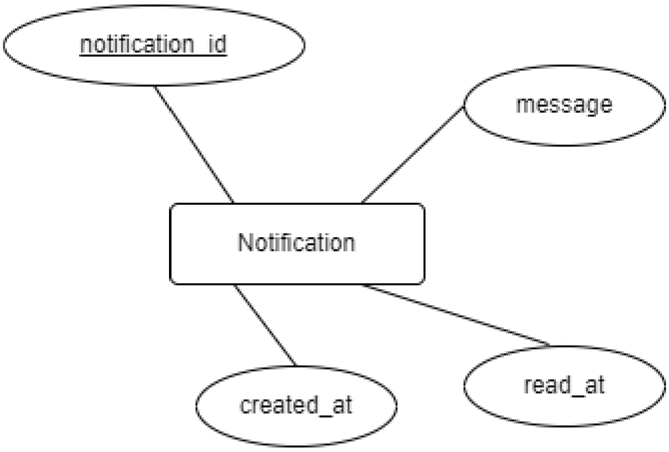


10.Notifications

This entity represents the info about notification sent to the user.

Attributes

- **Notification_id** [int, simple, single-valued, primary-key]: unique identifier for notification.
- **Message** [varchar, simple, single-valued]
- **Created_at** [timestamp, simple, single-valued, default current timestamp]: this represents the timestamp when notification is created and sent.
- **Read_at** [timestamp, simple, single-valued]: this stores when the notification is seen.

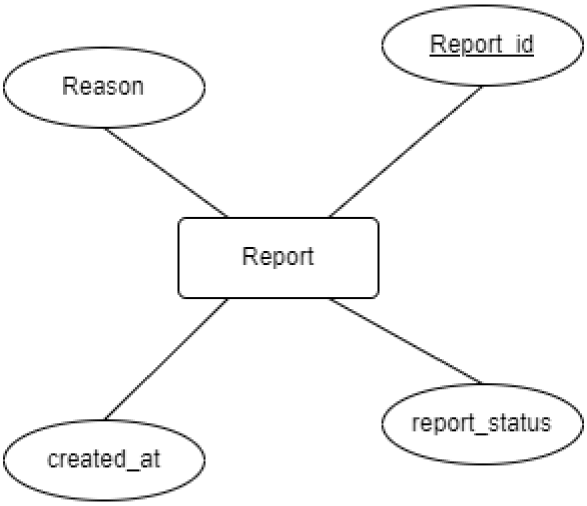


11. Report

Represents info about info

Attributes

- **Report_id** [primary-key, int, simple, single-valued]
- **Reason [**text, simple, single valued, not-null]
- **Created_at** [timestamp, simple, single-valued, default – current timestamp]
- **Report_status** [enum (‘checked’, ‘not-checked’), simple, single-valued]

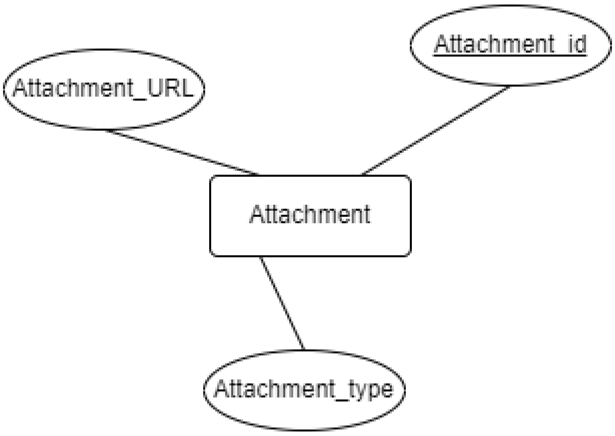


12.Attachments

The entity allows us to store the info about the atttcahments.

Attributes

- **Attachment_id** [int, simple, single-valued, primary-key]: unique identifier for the attachment file.
- **Attachment_URL** [varchar, simple, single-valued]: path where attachment file stored.
- **Attachment-type** [enum('video', 'image'), simple, single-valued]: type of attachment file



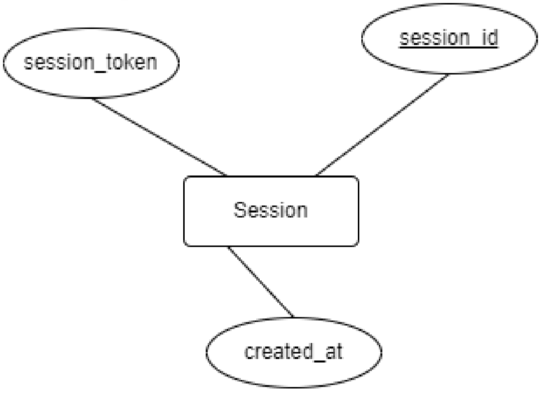
13.Session

This entity stores the session info of each user.

Attributes

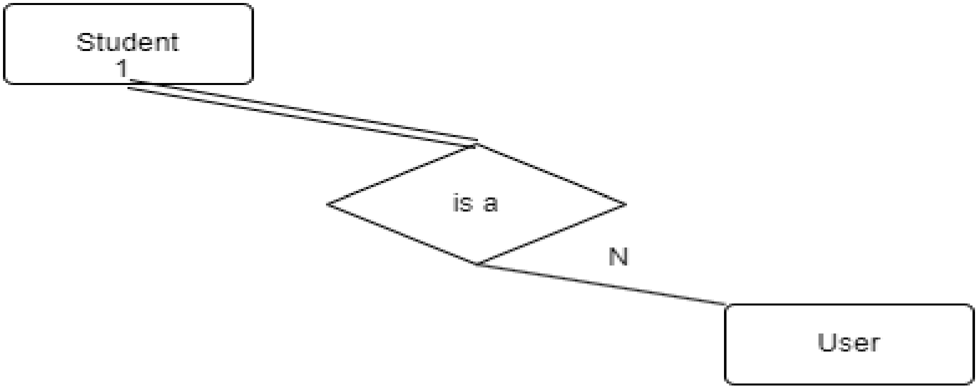
- **Session_id** [varchar, primary-key, simple, single-valued]: unique identifier.

- **Session_token** [varchar, simple, single-valued, not-null]: randomly generated token for session identification.
- **Created_at** [timestamp, simple, single-valued, default – current timestamp]: timestamp of session creation.



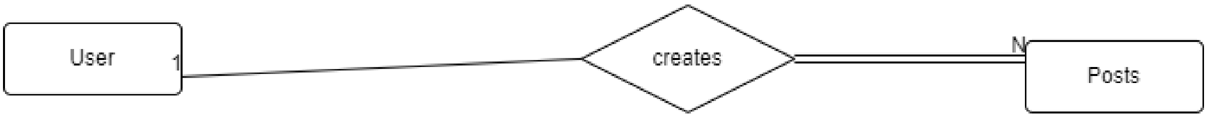
4.2.3. Relationship sets

1. Relationship between user and student

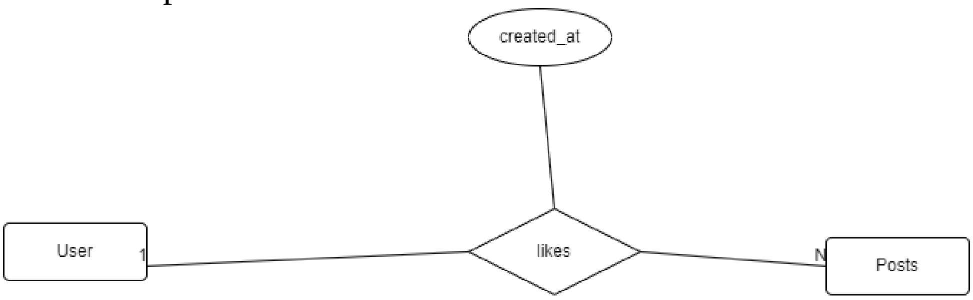


2. Relationship between user and post

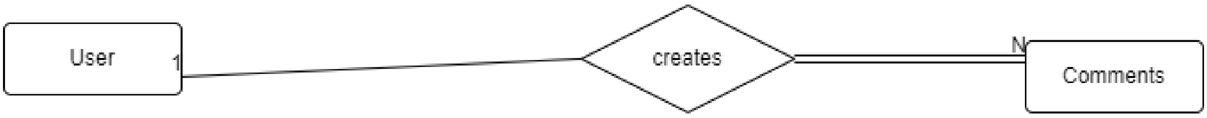
2.1. User creates posts



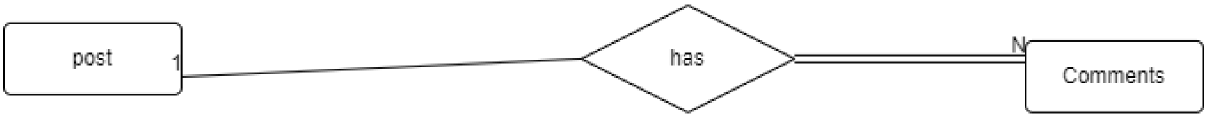
2.2. User likes post



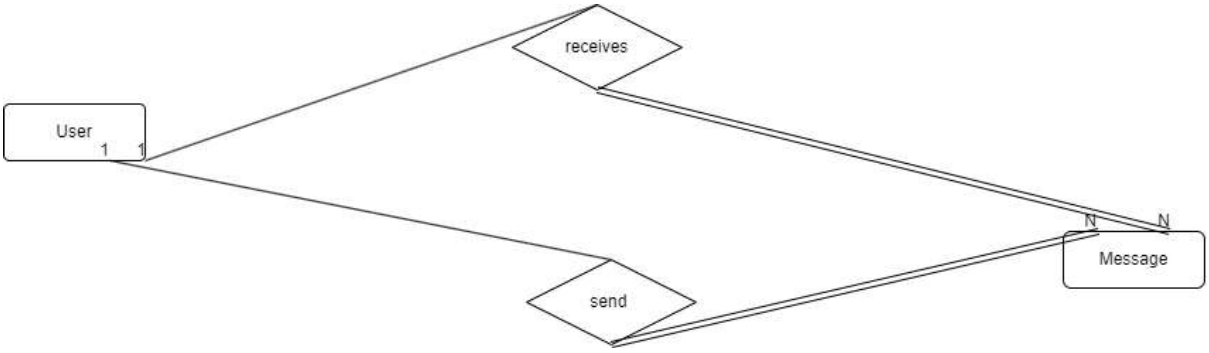
3. Relationship between user and comments



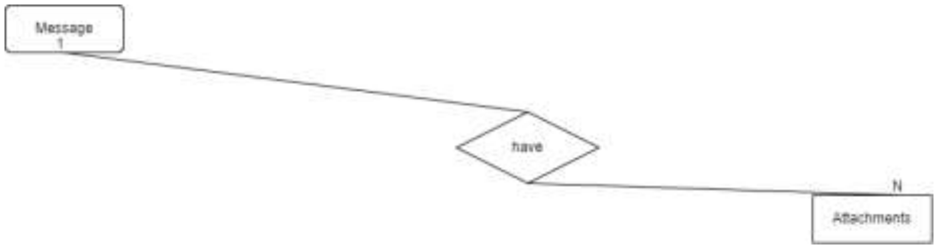
4. Relationship between post and comment



5. Relationship between user and message

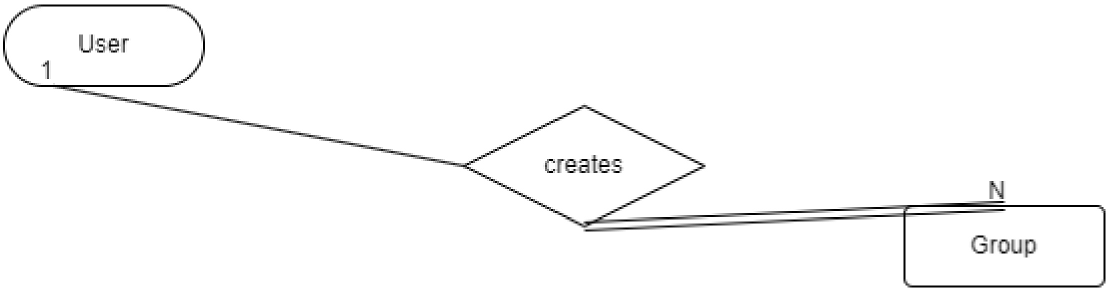


6. Relationship between message and attachment

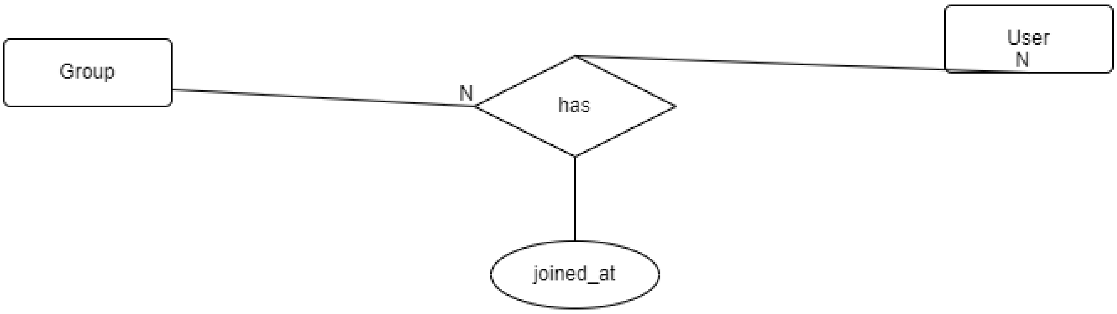


7. Relationship between user and group

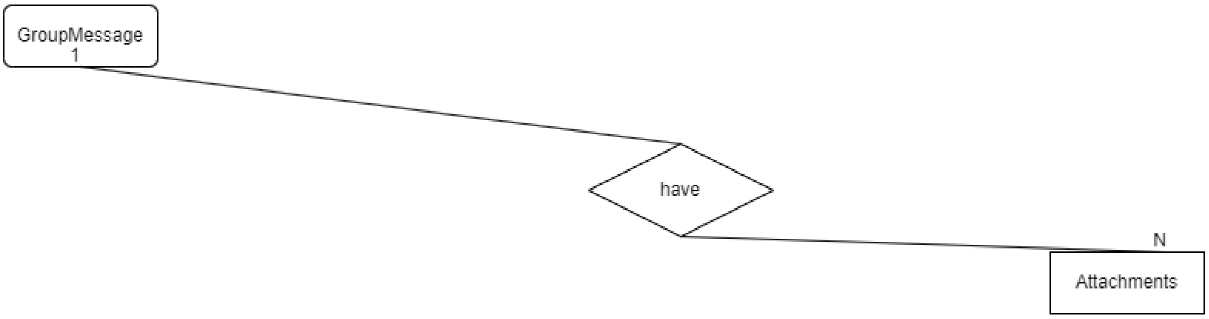
7.1. User creates group



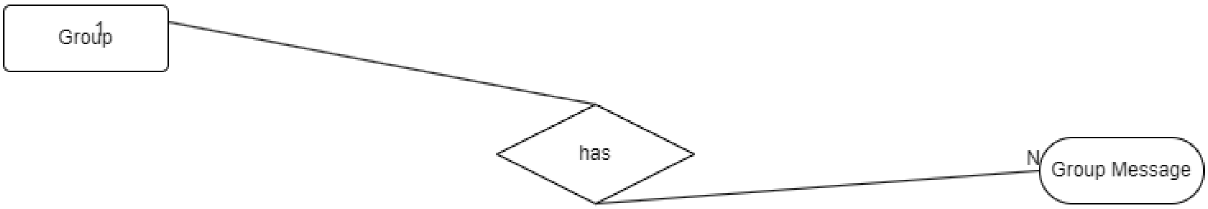
7.2. Group has user



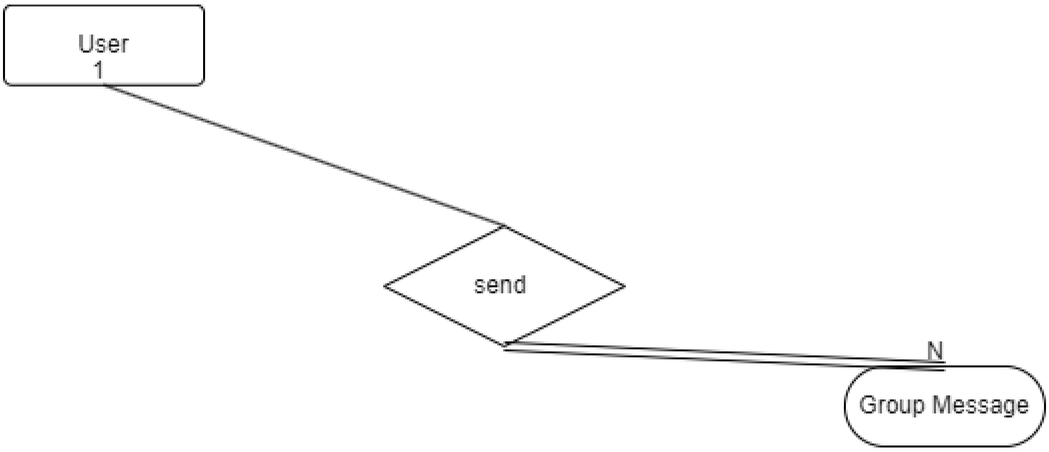
8. Relationship between Group message and attachment



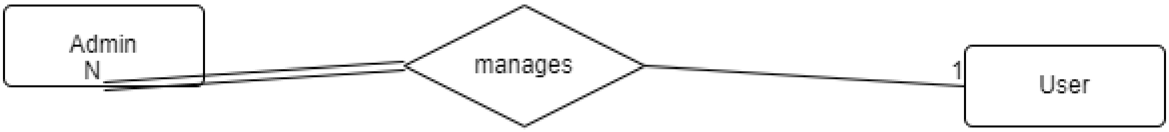
9. Relationship between group and group message



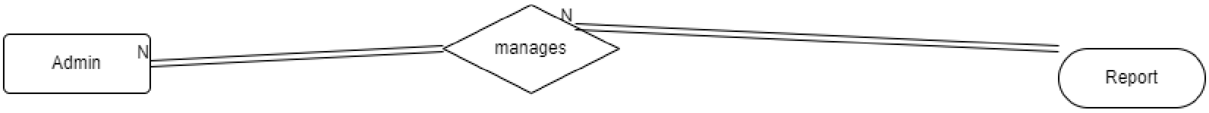
10. Relationship between user and group message



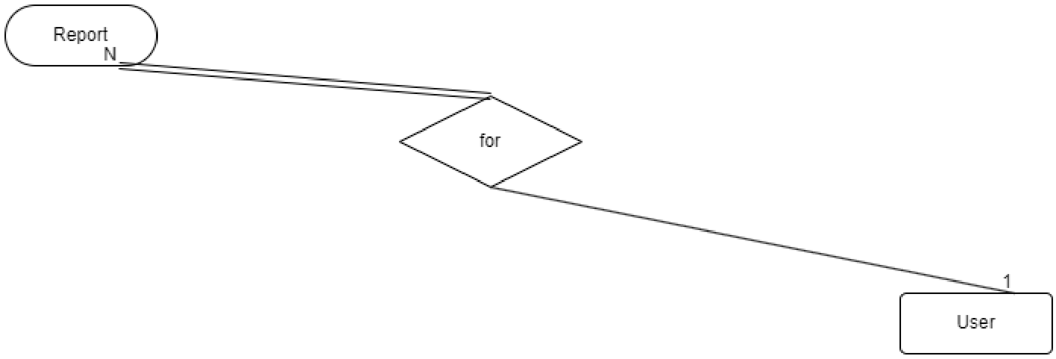
11. Relationship between Admin and user



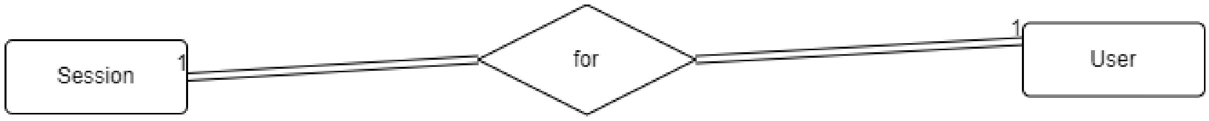
12. Relationship between Admin and report



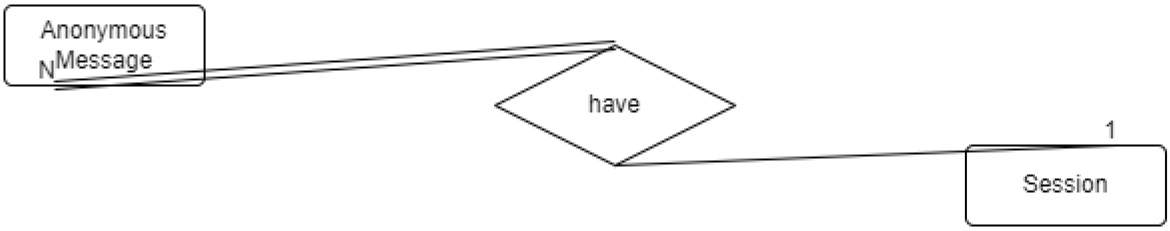
13. Relationship between Report and user



14.Relationship between session and user

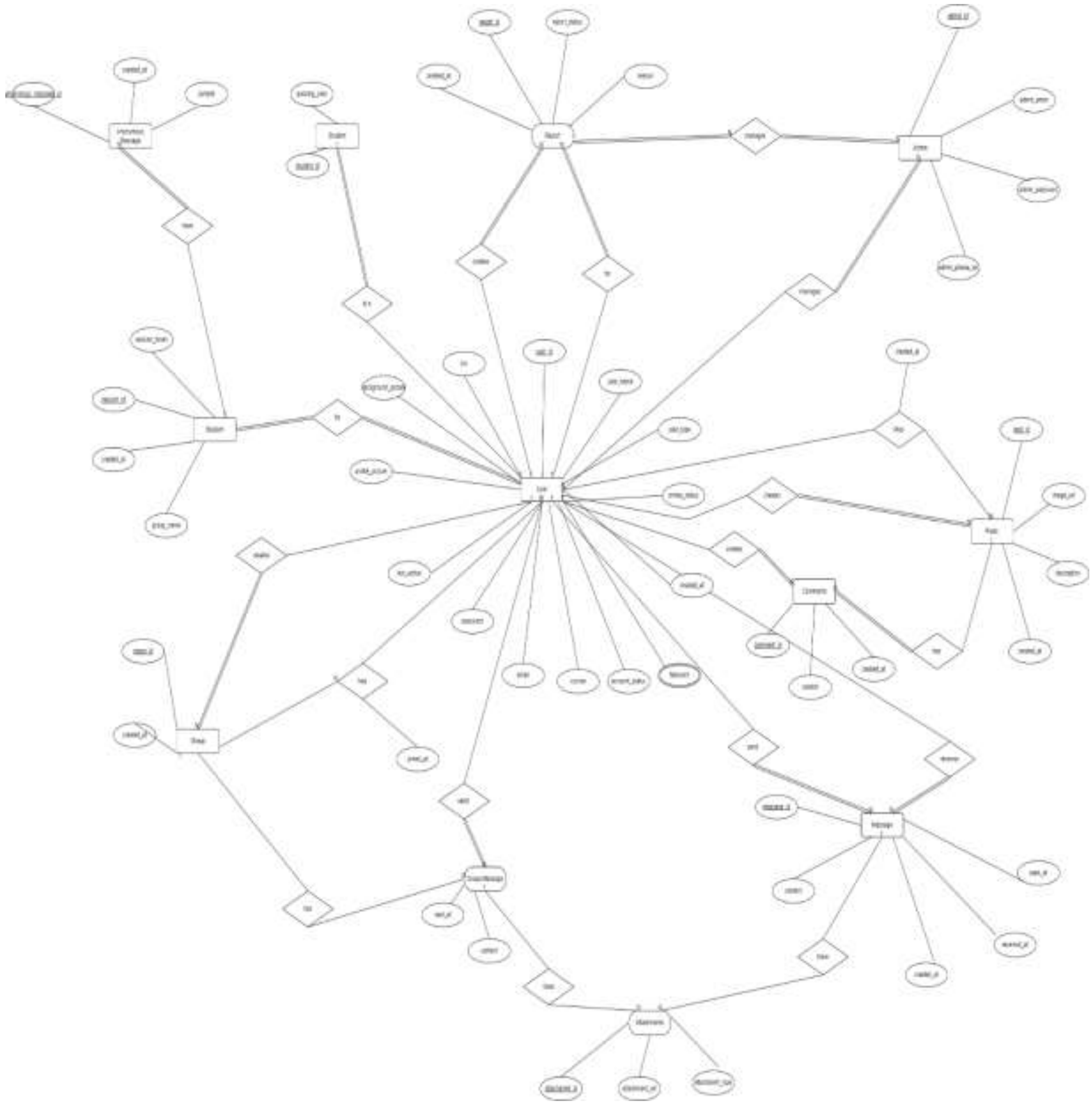


15.Relationship between Anonymous message and session



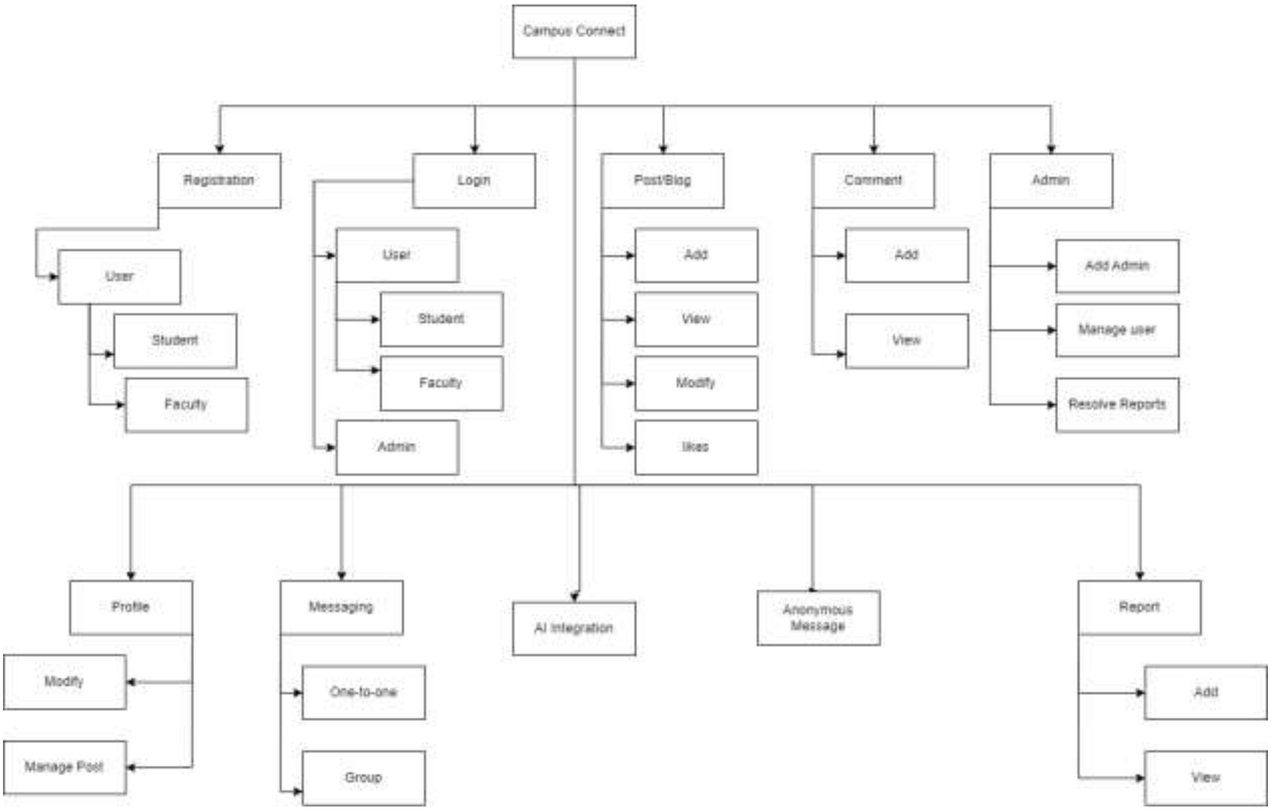
1

4.3.4. Er Diagram



4.2 Module Diagram

Module diagram is a diagram which is used for showing the allocation of classes and objects to module in the physical design of a system. Module Diagram indicates the partitioning of the system architecture. Through this diagram, it is possible to understand the general physical architecture of a system. The two essential elements of a module diagram are modules and their dependencies.



1. **Registration:** The module deals with the registration of new user in the system.
   1.1. **User:** The sub module of registration deals with the registration of new user.
   1.2. **Student:** The student registration module enables students to enrol in the campus connect application.
   1.3. **Faculty:** The faculty registration module enables faculty member to enrol in the campus connect application.

2. **Login:** The login module in campus connect allows user and admin to securely access their accounts using credentials ensuring proper authentication.
    - **2.1. User:** The module deals with the user login.
        - **2.1.1. Student:** The student login module in campus connect enables students to securely access their accounts using their credentials.
        - **2.1.2. Faculty:** The faculty login module in campus connect enables faculty to securely access their accounts using their credentials.
    - **2.2. Admin:** The admin login module in campus connect enables admin to securely access their accounts using their credentials.

---

3. **Post/Blog:** The post/blog module in campus connect allows users to share their thoughts with sub-modules for adding modifying viewing and liking posts.
    - **3.1. Add:** This sub module enables user to create and publish their thoughts or content as posts or blogs on the platform.
    - **3.2. Modify:** This sub-module allows users to edit and update their previously published posts/blogs.
    - **3.3. View:** This sub-module allows user to browse and read posts/blogs.
    - **3.4. Likes:** This sub-module allows user to express their appreciation for posts/blogs by liking them.

---

4. **Comment:** This module in campus connect allows user to add their thoughts or feedback to posts/blogs by adding and viewing comments.
    - **4.1. Add:** This sub module enables user to post their thought or response on posts/blogs shared within the platform.
    - **4.2. View:** This sub-module allows users to see and read comments posted on posts/blogs.

Report: This module includes sub-module for adding and viewing reports. User can add reports to notify administrators about inappropriate content or issue.

4.3. **Add:** This sub-module in campus connect allows user to submit reports about specific posts blogs or other content within platform.

4.4. **View:** In this sub-module user can view reports they have submitted themselves and also reports made by other users regarding their own content ensuring transparency and accountability within campus connect.

---

5. **Profile:** This module in campus connect includes sub modules for modifying personal info and managing posts.

5.1. **Modify:** This sub module allows user to update and edit their personal info.

5.2. **Manage Posts:** This sub-module enables user to oversee and administer their own posts/blogs including editing and deleting.

---

6. **Messaging:** This module includes sub-modules for one-to-one messaging and group messaging functionalities.

6.1. **One-to-one:** This sub module in campus connect allows students to engage in private conversation with another individual user on the platform.

6.2. **Group Messaging:** This sub-module in campus connect facilitates communicating among multiple users in a shared chat environment allowing users to participate in group chat.

---

7. **Anonymous Messaging:** This module in campus connect enables user to send message without revealing their identity to the recipient promoting a confidential communication channel.

---

8. **Admin:** This module includes sub-modules for resolving reports managing user accounts and adding new administrators to oversee platform operation.

8.1. **Resolve Report:** This sub module in campus connect allows administrators to review and take appropriate actions based on reports submitted by user.

8.2. **Manage User:** This sub-module enables admin to oversee and control user accounts.

8.3. **Add Admin:** This sub-module in campus connect allows admin to appoint new admins.

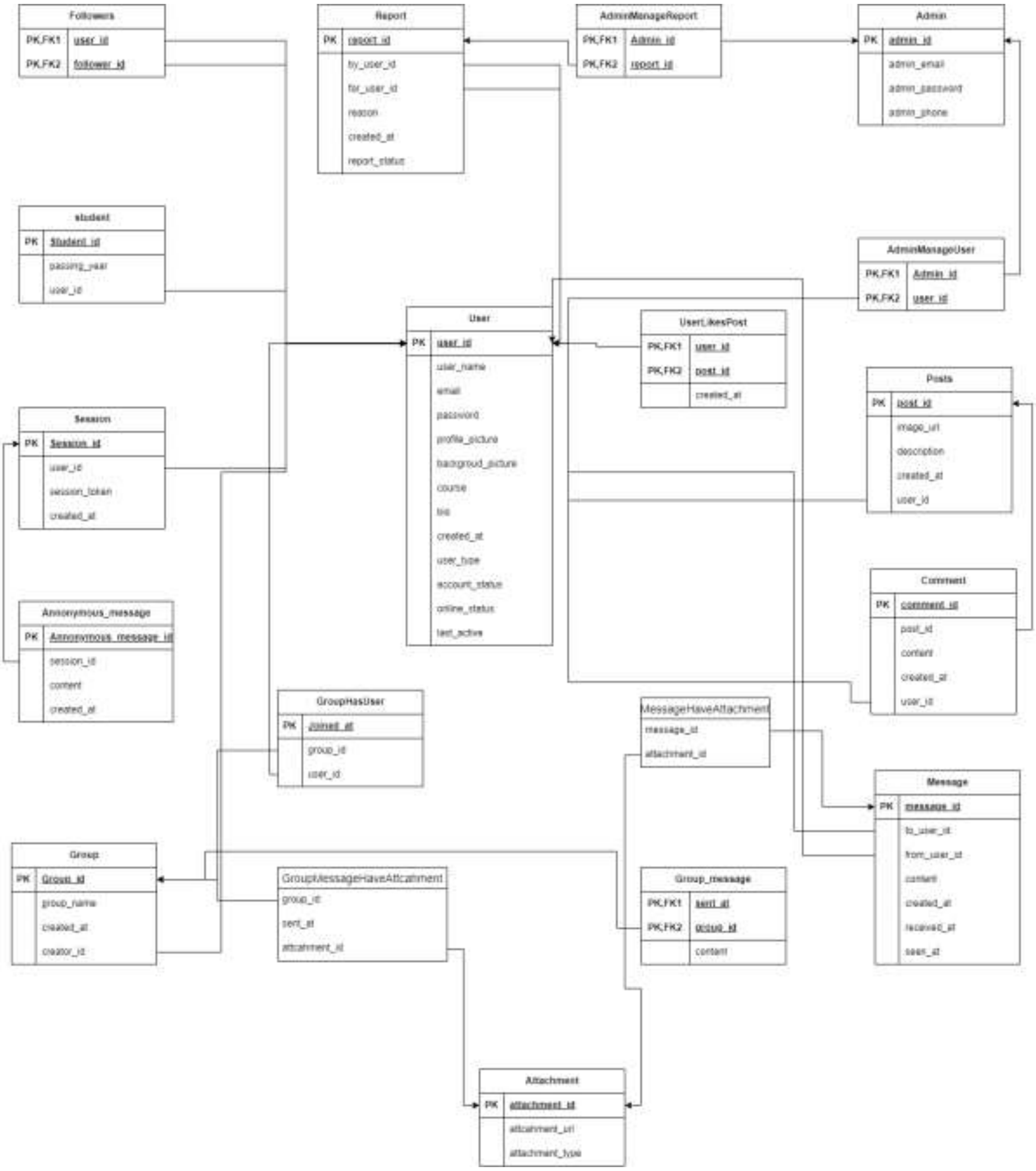
9. **AI Integration:** This module adds a chatbot powered by AI to interact with users offering assistance and answering queries to enhance user engagement and experience.

4.3. Schema Diagram

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organised and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

Diagram Notations:

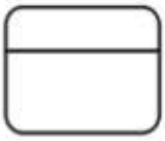
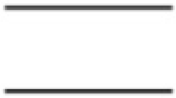
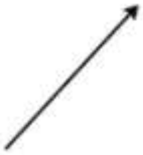
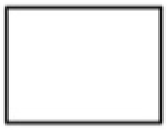
Name	Symbol	Description
Table	□	A table is a collection of related data held in table format within a database.
Relation	→	In a relational database system, a one-to-one table relationship links two tables based on a Primary Key column in the child which is also a Foreign Key referencing the Primary Key of the parent table row. Therefore, we can say that the child table share the Primary Key with the parent table.


Followers
PK,FK1 user_id
PK,FK2 follower_id
Report
PK report_id
by_user_id
for_user_id
reason
created_at
report_status
AdminManageReport
PK,FK1 Admin_id
PK,FK2 report_id
Admin
PK admin_id
admin_email
admin_password
admin_phone
student
PK Student_id
passing_year
user_id
AdminManageUser
PK,FK1 Admin_id
PK,FK2 user_id
User
PK user_id
user_name
email
password
profile_picture
backgroud_picture
course
bio
created_at
user_type
account_status
online_status
last_active
UserLikesPost
PK,FK1 user_id
PK,FK2 post_id
created_at
Posts
PK post_id
image_url
description
created_at
user_id
Session
PK Session_id
user_id
session_token
created_at
Comment
PK comment_id
post_id
content
created_at
user_id
Annonymous_message
PK Annonymous_message_id
session_id
content
created_at
GroupHasUser
PK Joined_at
group_id
user_id
MessageHaveAttachment
message_id
attachment_id
Message
PK message_id
to_user_id
from_user_id
content
created_at
received_at
seen_at
Group
PK Group_id
group_name
created_at
creator_id
GroupMessageHaveAttcahment
group_id
sent_at
attcahment_id
Group_message
PK,FK1 sent_at
PK,FK2 group_id
content
Attachment
PK attachment_id
attcahment_url
attachment_type

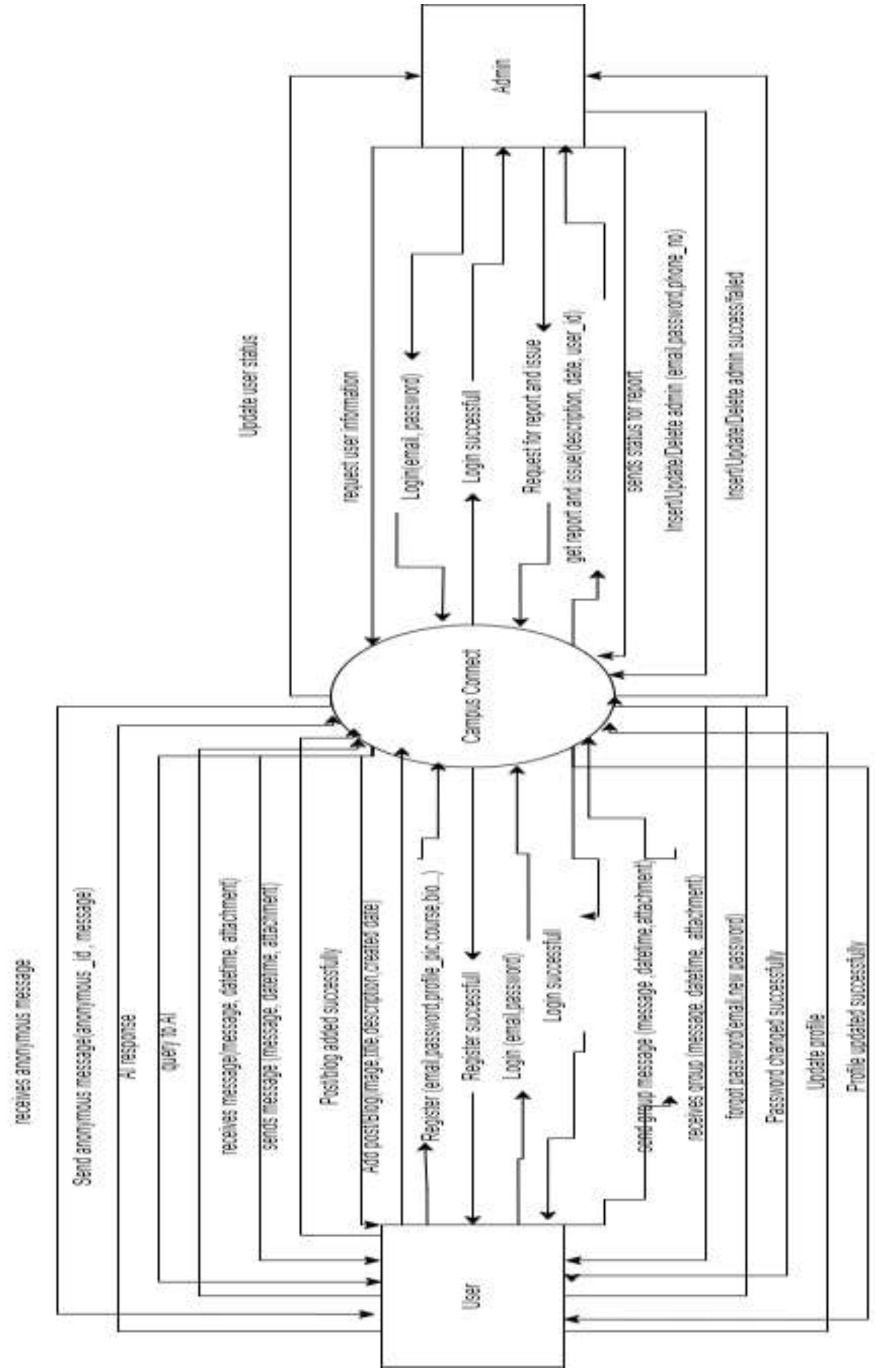
4.4. Data Flow Diagram

Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation. Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.

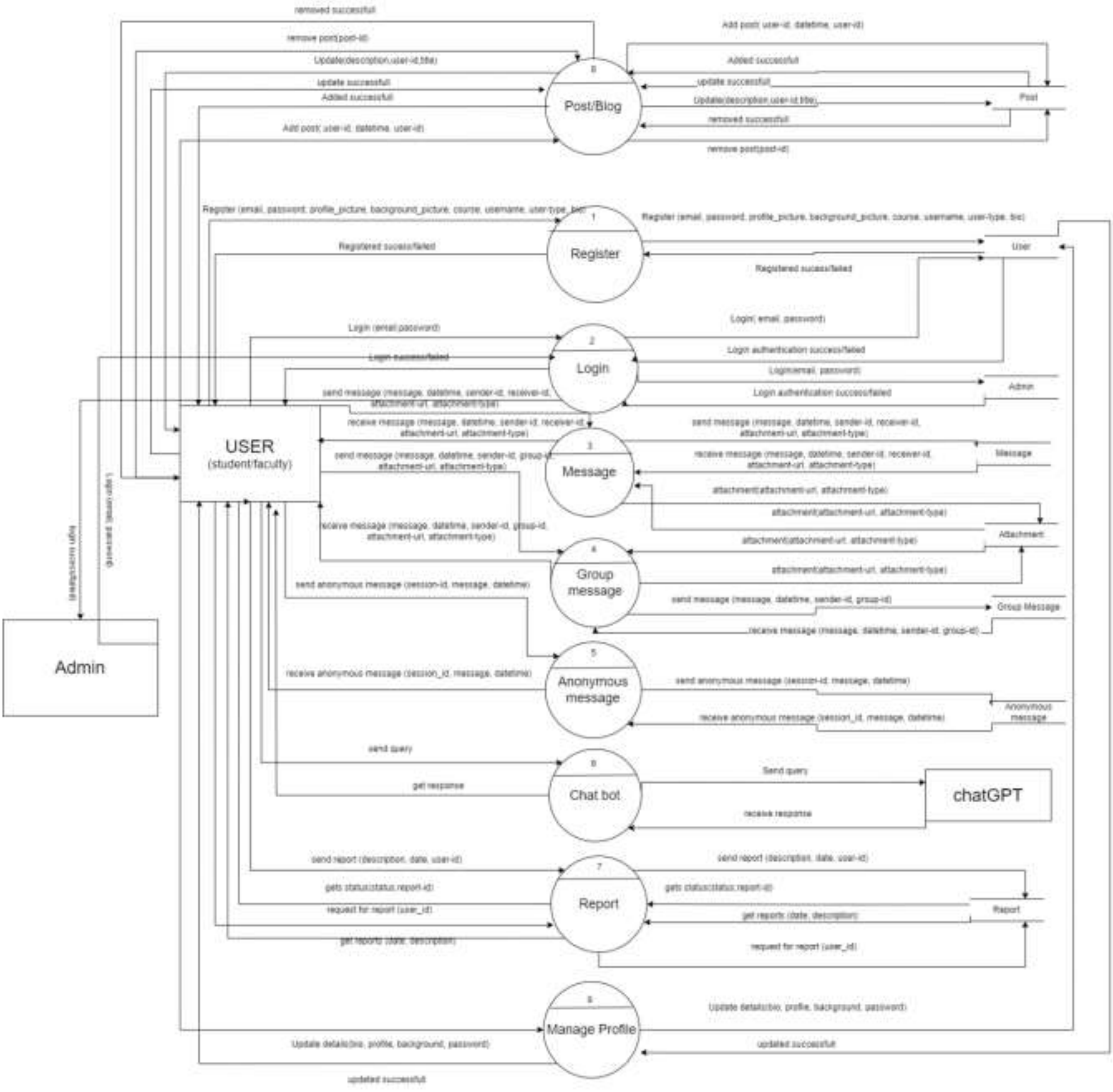
Diagram Notations:-

Name	Symbol	Description
Process		A process transforms incoming data flow into outgoing data flow.
Database		Data stores are repositories of data in the system.
Data Flow		Data flows are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it.
External Entity		External entities are objects outside the system, with which the system communicates

4.4.1. Context Level Diagram(Level 0):

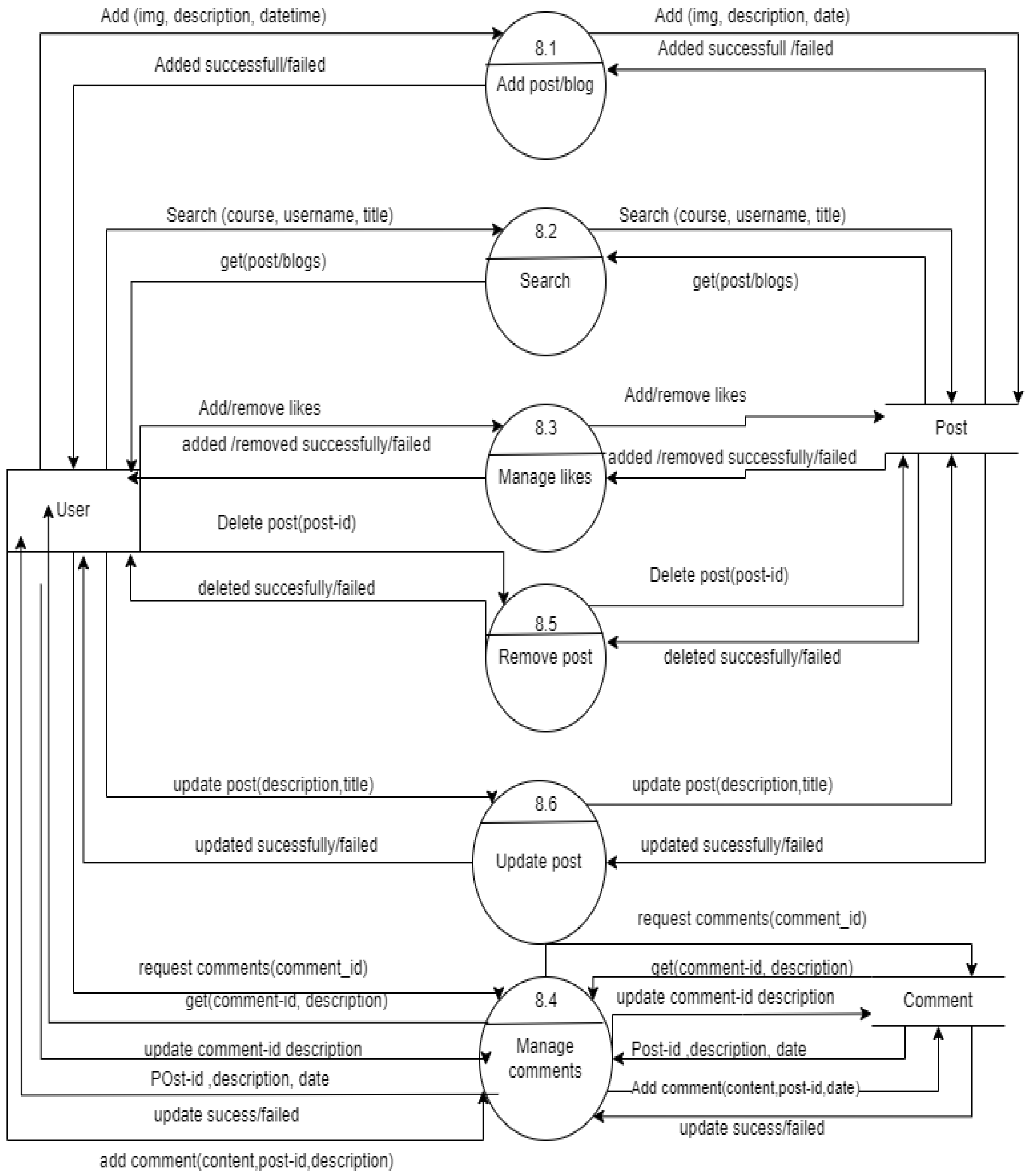

Admin
Campus Connect
User
Update user status
request user information
Login(email, password)
Login successful
Request for report and issue
get report and issue(description, date, user_id)
sends status for report
Insert/Update/Delete admin (email,password,phone_no)
Insert/Update/Delete admin successfailed
receives anonymous message
Send anonymous message(anonymous_id, message)
AI response
query to AI
receives message(message, datetime, attachment)
sends message (message, datetime, attachment)
Post/blog added successfully
Add post/blog(image,title,description,created date)
Register (email,password,profile_pic,course,bio,...)
Register successful
Login (email,password)
Login successful
send group message (message, datetime,attachment)
receives group (message, datetime, attachment)
forgot password(email,new password)
Password changed successfully
Update profile
Profile updated successfully

4.4.2. First Level DFD

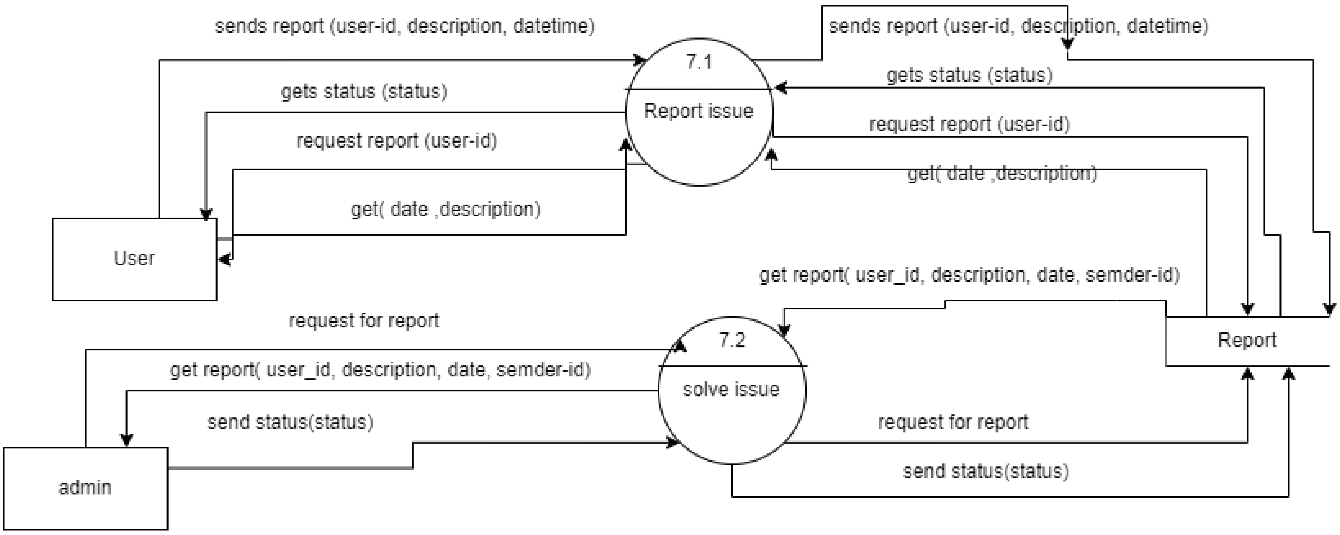

removed successfull
remove post(post-id)
Update(description,user-id,title)
update successfull
Added successfull
Add post(user-id, datetime, user-id)
8
Post/Blog
Add post(user-id, datetime, user-id)
Added successfull
update successfull
Update(description,user-id,title)
removed successfull
remove post(post-id)
Post
Register (email, password, profile_picture, background_picture, course, username, user-type, bio)
Registered sucess/failed
1
Register
Register (email, password, profile_picture, background_picture, course, username, user-type, bio)
Registered sucess/failed
User
Login (email,password)
Login success/failed
2
Login
Login(email, password)
Login authentication success/failed
Login(email, password)
Login authentication success/failed
Admin
send message (message, datetime, sender-id, receiver-id, attachment-url, attachment-type)
receive message (message, datetime, sender-id, receiver-id, attachment-url, attachment-type)
send message (message, datetime, sender-id, group-id, attachment-url, attachment-type)
USER
(student/faculty)
3
Message
send message (message, datetime, sender-id, receiver-id, attachment-url, attachment-type)
receive message (message, datetime, sender-id, receiver-id, attachment-url, attachment-type)
Message
attachment(attachment-url, attachment-type)
attachment(attachment-url, attachment-type)
attachment(attachment-url, attachment-type)
Attachment
login success/failed
Login (email, password)
receive message (message, datetime, sender-id, group-id, attachment-url, attachment-type)
4
Group message
attachment(attachment-url, attachment-type)
send message (message, datetime, sender-id, group-id)
Group Message
receive message (message, datetime, sender-id, group-id)
send anonymous message (session-id, message, datetime)
Admin
5
Anonymous message
receive anonymous message (session_id, message, datetime)
send anonymous message (session-id, message, datetime)
receive anonymous message (session_id, message, datetime)
Anonymous message
send query
6
Chat bot
get response
Send query
receive response
chatGPT
send report (description, date, user-id)
7
Report
gets status(status,report-id)
request for report (user_id)
get reports (date, description)
send report (description, date, user-id)
gets status(status,report-id)
get reports (date, description)
request for report (user_id)
Report
8
Manage Profile
Update details(bio, profile, background, password)
updated successfull
Update details(bio, profile, background, password)
updated successfull

4.4.3. Second Level DFD

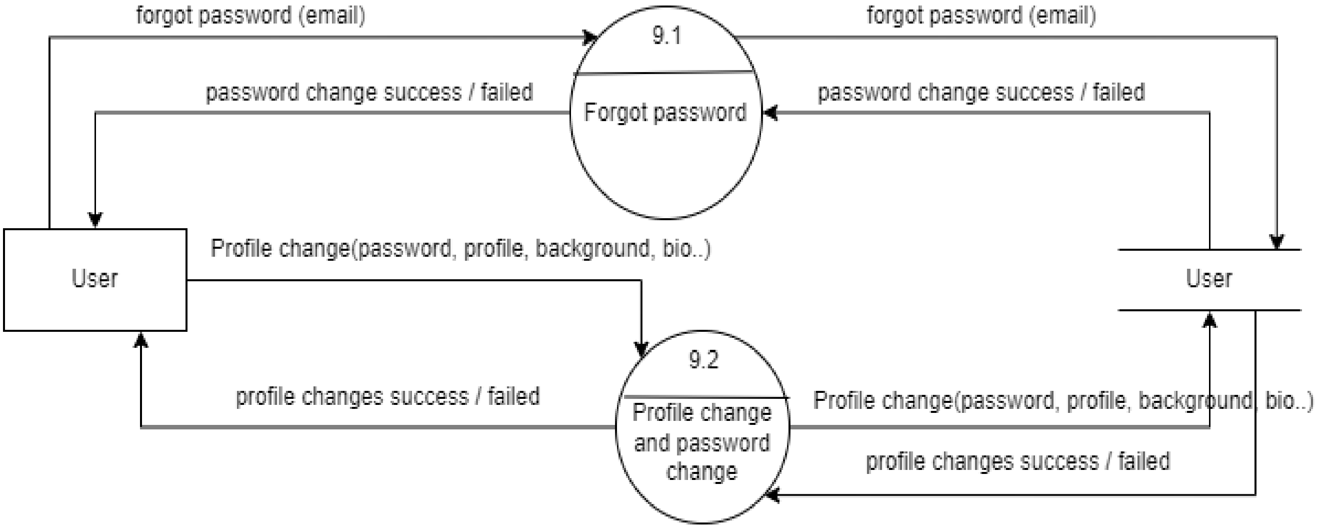
Level 2 DFD for post


Add (img, description, datetime)
Added successfull/failed
8.1
Add post/blog
Add (img, description, date)
Added successfull /failed
Search (course, username, title)
get(post/blogs)
8.2
Search
Search (course, username, title)
get(post/blogs)
Add/remove likes
added /removed successfully/failed
8.3
Manage likes
Add/remove likes
added /removed successfully/failed
Post
User
Delete post(post-id)
deleted succesfully/failed
8.5
Remove post
Delete post(post-id)
deleted succesfully/failed
update post(description,title)
updated sucessfully/failed
8.6
Update post
update post(description,title)
updated sucessfully/failed
request comments(comment_id)
request comments(comment_id)
get(comment-id, description)
get(comment-id, description)
update comment-id description
Comment
8.4
Manage comments
update comment-id description
POst-id ,description, date
Post-id ,description, date
Add comment(content,post-id,date)
update sucess/failed
update sucess/failed
add comment(content,post-id,description)

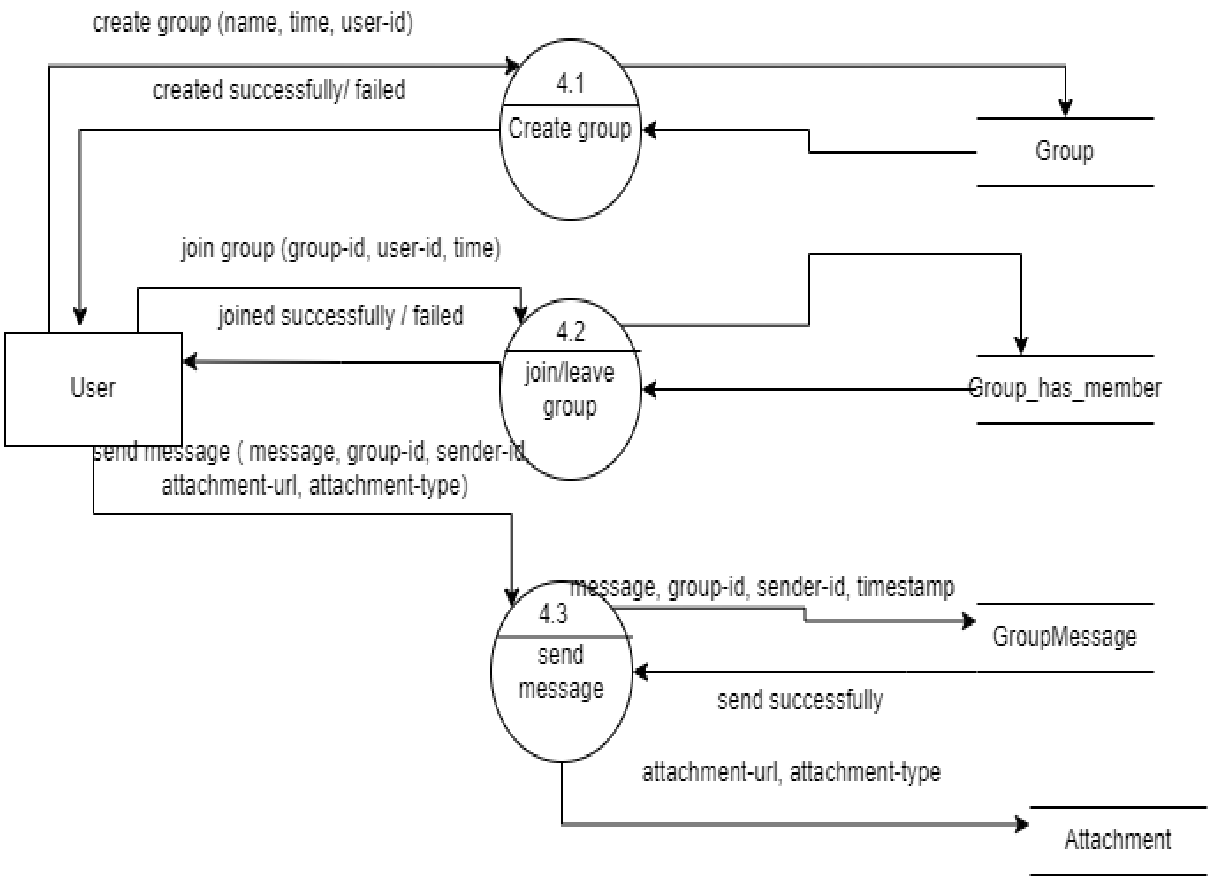
Level 2 DFD for report



Level 2 DFD for manage profile

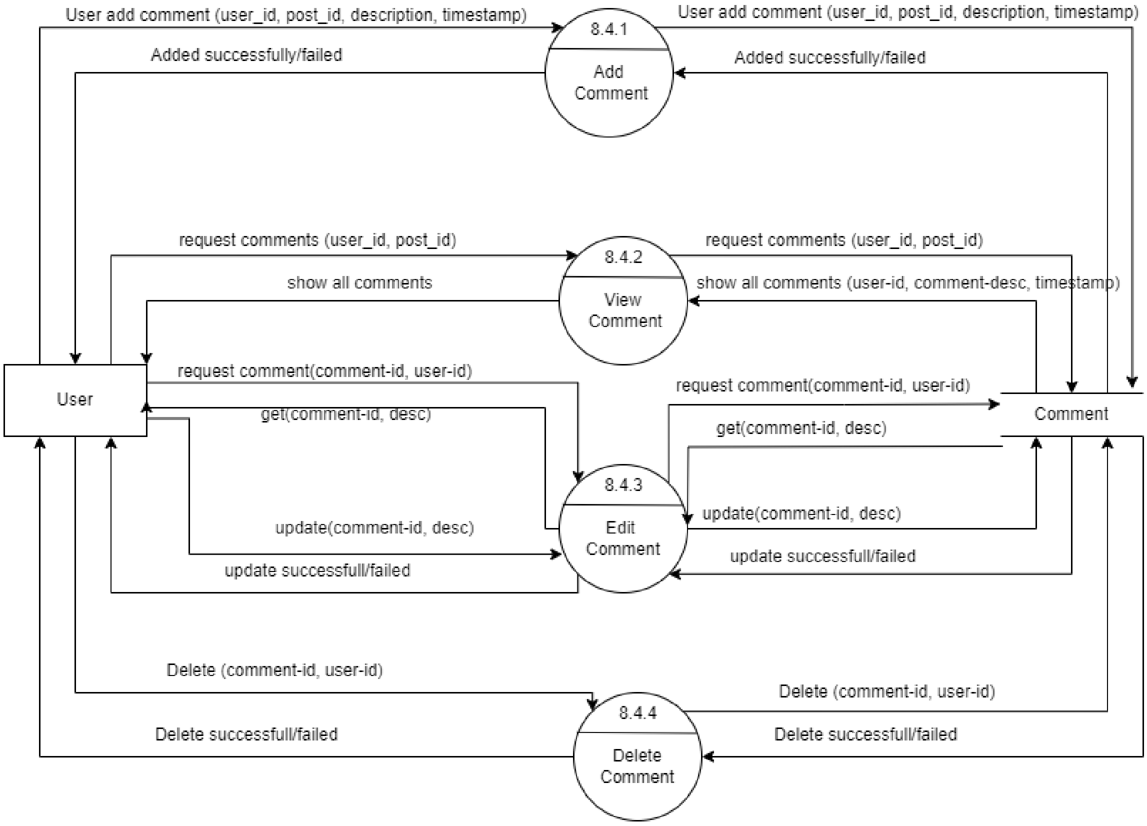

forgot password (email)
9.1
forgot password (email)
password change success / failed
Forgot password
password change success / failed
User
Profile change(password, profile, background, bio..)
User
9.2
profile changes success / failed
Profile change and password change
Profile change(password, profile, background, bio..)
profile changes success / failed

Level 2 DFD for Group Message

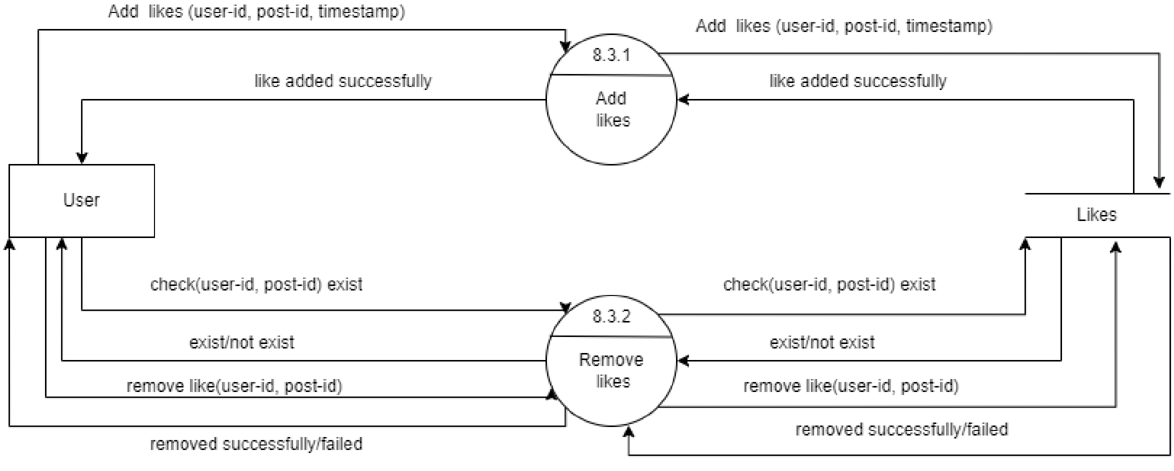

create group (name, time, user-id)
created successfully/ failed
4.1
Create group
Group
join group (group-id, user-id, time)
joined successfully / failed
4.2
join/leave group
User
Group_has_member
send message (message, group-id, sender-id attachment-url, attachment-type)
message, group-id, sender-id, timestamp
4.3
send message
GroupMessage
send successfully
attachment-url, attachment-type
Attachment

4.4.4 Third Level DFD

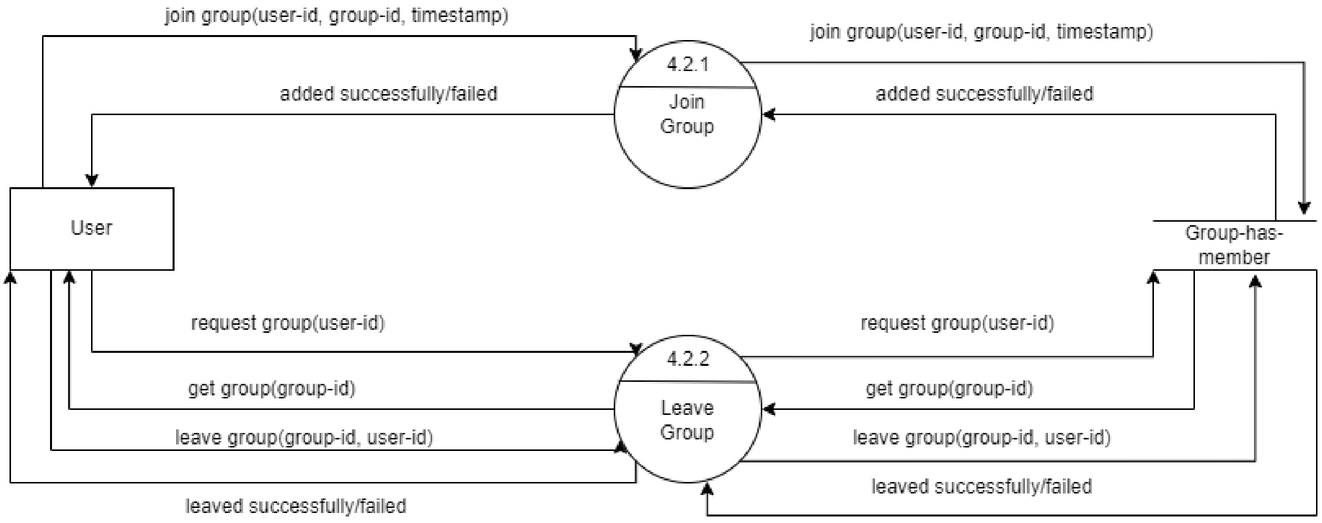
Level 3 DFD for Manage comments


User add comment (user_id, post_id, description, timestamp)
Added successfully/failed
8.4.1
Add Comment
User add comment (user_id, post_id, description, timestamp)
Added successfully/failed
request comments (user_id, post_id)
show all comments
8.4.2
View Comment
request comments (user_id, post_id)
show all comments (user-id, comment-desc, timestamp)
User
request comment(comment-id, user-id)
get(comment-id, desc)
update(comment-id, desc)
update successfull/failed
8.4.3
Edit Comment
request comment(comment-id, user-id)
get(comment-id, desc)
update(comment-id, desc)
update successfull/failed
Comment
Delete (comment-id, user-id)
Delete successfull/failed
8.4.4
Delete Comment
Delete (comment-id, user-id)
Delete successfull/failed

Level 3 DFD for Manage likes


Add likes (user-id, post-id, timestamp)
like added successfully
8.3.1
Add likes
Add likes (user-id, post-id, timestamp)
like added successfully
User
Likes
check(user-id, post-id) exist
exist/not exist
remove like(user-id, post-id)
removed successfully/failed
8.3.2
Remove likes
check(user-id, post-id) exist
exist/not exist
remove like(user-id, post-id)
removed successfully/failed

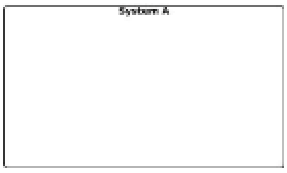
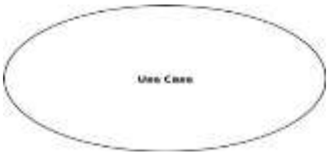
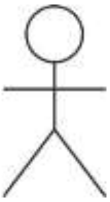
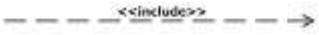
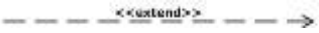
Level 3 DFD for join/leave group

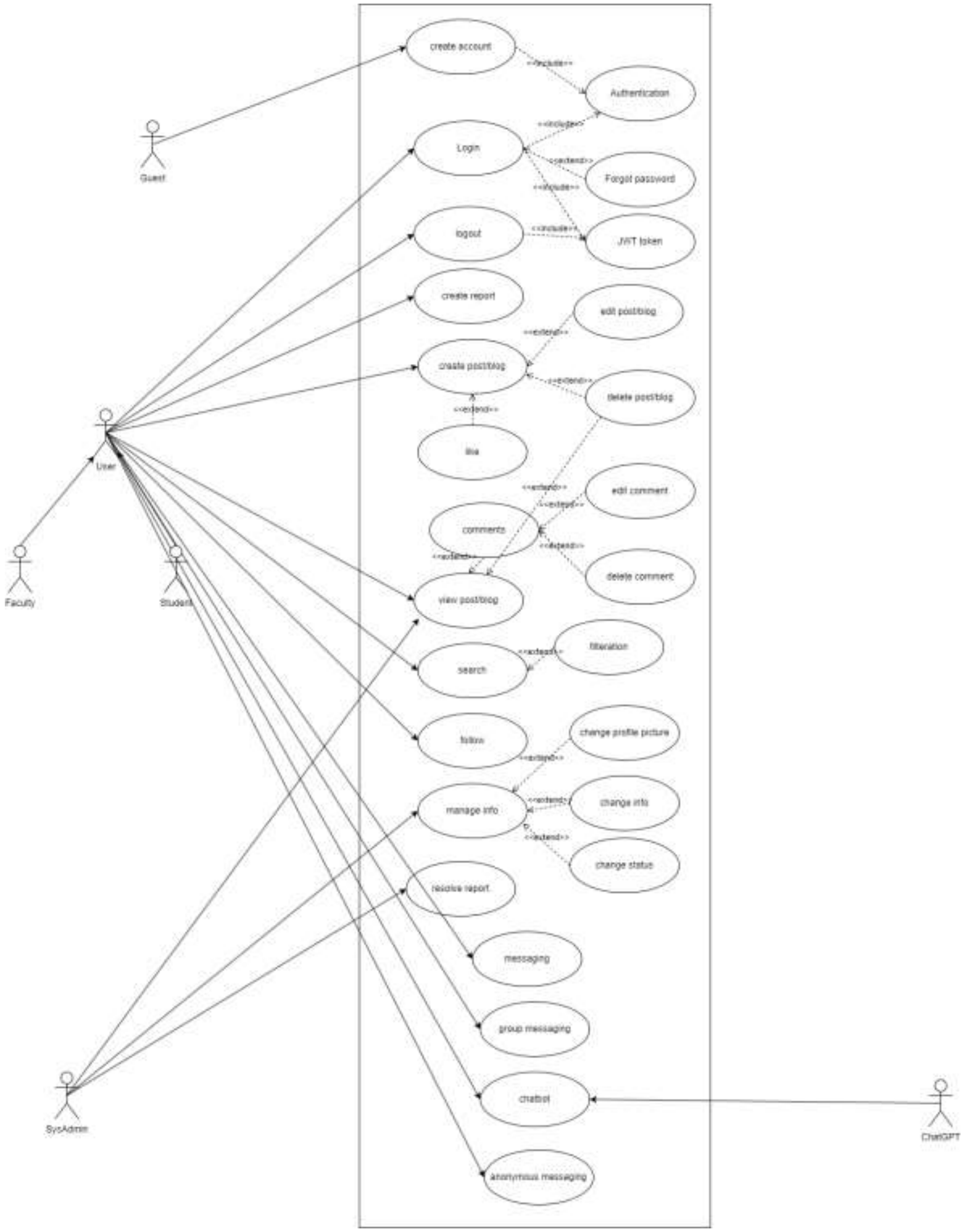

join group(user-id, group-id, timestamp)
join group(user-id, group-id, timestamp)
4.2.1
Join Group
added successfully/failed
added successfully/failed
User
Group-has-member
request group(user-id)
request group(user-id)
4.2.2
Leave Group
get group(group-id)
get group(group-id)
leave group(group-id, user-id)
leave group(group-id, user-id)
leaved successfully/failed
leaved successfully/failed

4.5. Use Case Diagram

Each use case represents a slice of the functionality the system provides. The set of use cases shows the complete functionality of the system at some level of detail. Similarly, each actor represents one kind of object for which the system can perform behaviour. The set of actors represents the complete set of objects that the system can serve. Objects accumulate behaviour from all the systems with which they interact as actors.

4.5.1. Diagram Notations: -

Name	**Symbol**	**Description**
System		The rectangular boundary is the system. Use cases fall inside it, and actors will be placed outside it.
Use Case		An oval shape represents a use case. Use cases represent the functionality of the system, as well as the end-goal of the actor. Use cases should be placed inside the system.
Actor		When an actor interacts with the system, it triggers a use case. Actors should be placed outside the system.
Association		Association between use cases.
Include Relationship		An extension indicates that one use case may include the behaviour of another use case.
Extend Relationship		An inclusion represents one use case using the functionality of another use case.



4.5.2 Description

1. Create Account

Description: The user needs to create account first i.e. add details such as email, password, name, profile-picture, background-picture, bio, course, user-type.
Summary: The user create account for application.
Actor: Guest
Pre-condition: User should don't have an account and also the student or faculty of college.
Post-condition: User can login with registered mail and password.
Exception: An error message is showing if any request input is empty or data type or pattern is not match with defined one.

2. **Login**
Description: User/ Admin will be able to login into the system using login credentials.
Summary: Input the details and access the system.
Actor: User, System Admin
Pre-condition: User should have registered for account admin should have an account.
Post-condition: A JWT token is created which is used to access all the application feature and home page of post is displayed.
Exception: An error message is displayed if user/sysadmin credentials are wrong.

3. **Logout**
Description: user/sysadmin will be able to logout from the system.
Summary: The user/sysadmin logout from the system.
Actor: user /sysadmin
Pre-condition: User must be logged In.
Post-condition: User is logged out and redirected to login screen.
Exception: server out of reach.

4. **Create Report**
Description: user can report about inappropriate content or issue.
Summary: User report about inappropriate content or issue.
Actor: User.
Pre-condition: User must be logged In and user must have some issue.
Post-condition: The users issue solved by the admin and got the status.
Exception: If any invalid input is entered or error message is shown.

5. **Create Post/Blog**
Description: The user can add posts/Blogs info such as image description created-on title.
Summary: User can add post/blog.
Actor: User
Pre-condition: User must be logged In.
Post-condition: post will be uploaded successfully and other users is able to see, react and comment on it.
Exception: Show error if the description field is leaved empty or any invalid input is entered.

6. **View Post/Blogs**

   Description: user can able to see view post/blogs.
   Summary: user can view post/blogs.
   Actor: User
   Pre-condition: User must be logged In and system must have some posts.
   Post-condition: User is able to react and comment on the post.
   Exception: Shows message no post if nothing is available.

7. **Search**

   Description: The user can simply type the title of the post in the search box and result of post with that title is displayed.
   Summary: User search the post and post displayed.
   Actor: User.
   Pre-condition: User must be logged In and have some posts with searched title.
   Post-condition: Displaying post having searched title.
   Exception: Showing error of invalid input or no post having searched title.

8. **Follow/Connection**

   Description: the user can able to follow the other user so the post and blogs posted by that user is shown at the home page
   Summary: User can follow other users
   Actor: User
   Pre-condition: User need to logged In and system have other users.
   Post-condition: User is able to see followed user post at home screen.
   Exception: server error.

9. **Messaging**

   Description: The user will able to communicate to each other by sending and receiving message.
   Summary: The users can communicate to each other.
   Actor: User
   Pre-condition: User must be logged In and selected the user to send message.
   Post-condition: Message is sent to the other user.
   Exception: If any invalid input is entered error message is shown.

10. **Group Messaging**

    Description: The user will able to communicate to multiple users in one time by sending group message.
    Summary: Communicate with a group.
    Actor: User
    Pre-condition: User must be logged In and selected the group to send message.
    Post-condition: Message is sent to the other user of group.
    Exception: If any invalid input is entered error message is shown.

11. **Chatbot**

Description: The user can send the query to the chatbot and in return gets the response.
Summary: User sends query and gets response.
Actor: User, chat GPT.
Pre-condition: the input field should not be empty and user must be logged In.
Post-condition: Response is displayed based on query.
Exception: If any invalid input is entered error message is shown.

12.Anonymous Messaging

Description: Users can send message anonymously and view others anonymous message.
Summary: Anonymous communication.
Actor: User
Pre-condition: User must be logged in so that they have an anonymous id and selected the anonymous forum.
Post-condition: Message is sent.
Exception: If any invalid input is entered error message is shown.

13.Manage info

Description: User/sysadmin can change the info and status of account.
Summary: Used to modify the info.
Actor: User, sysadmin.
Pre-condition: User have an active account and logged in it and for admin received report about that account and logged in as admin.
Post-condition: Info changed.
Exception: Invalid change.

14.Resolve Report

Description: The admin resolves the issue made by the user and send the status whether it is solved or not.
Summary: Admin resolves report and send status.
Actor: sysadmin
Pre-condition: Admin must be logged In and have some issues.
Post-condition: The issue is solved of the user which is reported.
Exception: Server out of reach.

4.5.3. Use Case Scenario

Registration

1. The user request for registration.
2. User select the type of user.
3. The user fills the details and submit the form
4. If detailed are correct then a confirmation mail is sent to the respected mail address.
5. If any detail is incorrect error message is shown.

Login

1. If the user is already logged in the user is redirected to the home page.
2. The user input the credentials.
3. If the credentials are valid then the user is redirected to the home page.
4. If credentials are incorrect the error message is displayed.

Messaging

1. When the user sends the message to the other user the message is stored in database and is sent to the other user.
2. If the other user is online the user will receive the message and reply to the other user who has sent the message.

Group Messaging

1. Group member of that group sends the message to the group the message is stored in database and is sent to that group.
2. As user comes online they are able to see the message with sent user name and also able to reply.

Anonymous Messaging

1. When the user sends the message in the public forum the message is stored in database and is sent to the public forum.
2. As user comes online they are able to see the message with sent users anonymous id.

Search

1. The user inputs the search data that the user wants to search topic, date, username.
2. If there is no post/blogs that match the search criteria the system display no result message.
3. If there is post that match the search criteria the system displayed the list of search result.

Report Issue

1. The user request for the report issue page.
2. The user gets the report issue form.
3. The user fills the issue forums and submits.
4. If the invalid data is input the system will response with error message.
5. If the data is valid the data is stored in database and in return gets the response and status of the issue.
6. If the user wants to view the report status the user can request for viewing of the details.

Resolve issue

1. Admin request for reporting page.
2. System provides the all issue with details.
3. Admin view the issue and resolve the issue.
4. If the issue is resolved the admin sends the resolved the admin sends the resolved status.
5. If the issue is not resolvable the admin sends the not resolvable status.

Chatbot

1. User request for chatbot.
2. System provides the chatbot.
3. User inputs the query.
4. If the system has response related to the query the system response with the answer.
5. If the system don't have the response to the query the system displayed the message of not answering query.

Forgot Password

1. The user visits the login page.
2. Click the forgot password.
3. Clicks the forgot password.
4. Gets the forgot password page.
5. Fills the details required to recover password.
6. Gets the OTP on mail.
7. User enters and verify the OTP.
8. If the OTP is valid then user can input the new password.
9. If the OTP is invalid then the user wants to be able to change the password.

Profile Update and Password Update

1. The user visits the profile page.
2. Fills the field which need to be updated.
3. If the data entered is valid then it is stored successfully.
4. If the data entered is invalid an error is displayed.

Add posts/blogs

1. The user requests for add post.
2. The system provides the page.
3. The user fills the details of posts.
4. The data is stored in database.
5. If the data is stored successfully then user get message posts added successfully and other users are able to see it.
6. If the data is not stored successfully due to invalid datatype user gets the error message.

Update Posts/Blogs

1. User request for update post page.
2. The system provides the page.
3. The user update the description, title of the post.
4. The data is updated in database.
5. If it updated successfully then user gets a message post updated successfully and other users are able to see the updated post.
6. If due to some reason data is not updated in database then user gets an error.

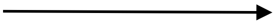
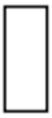
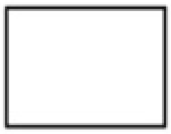
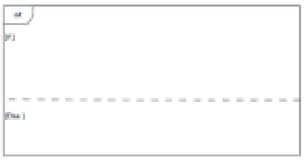
View Reports

1. The user request for viewing report.
2. The system request for the details of the report on that particular user.
3. If there is any report it will be displayed.
4. If no report found then it will show no report found

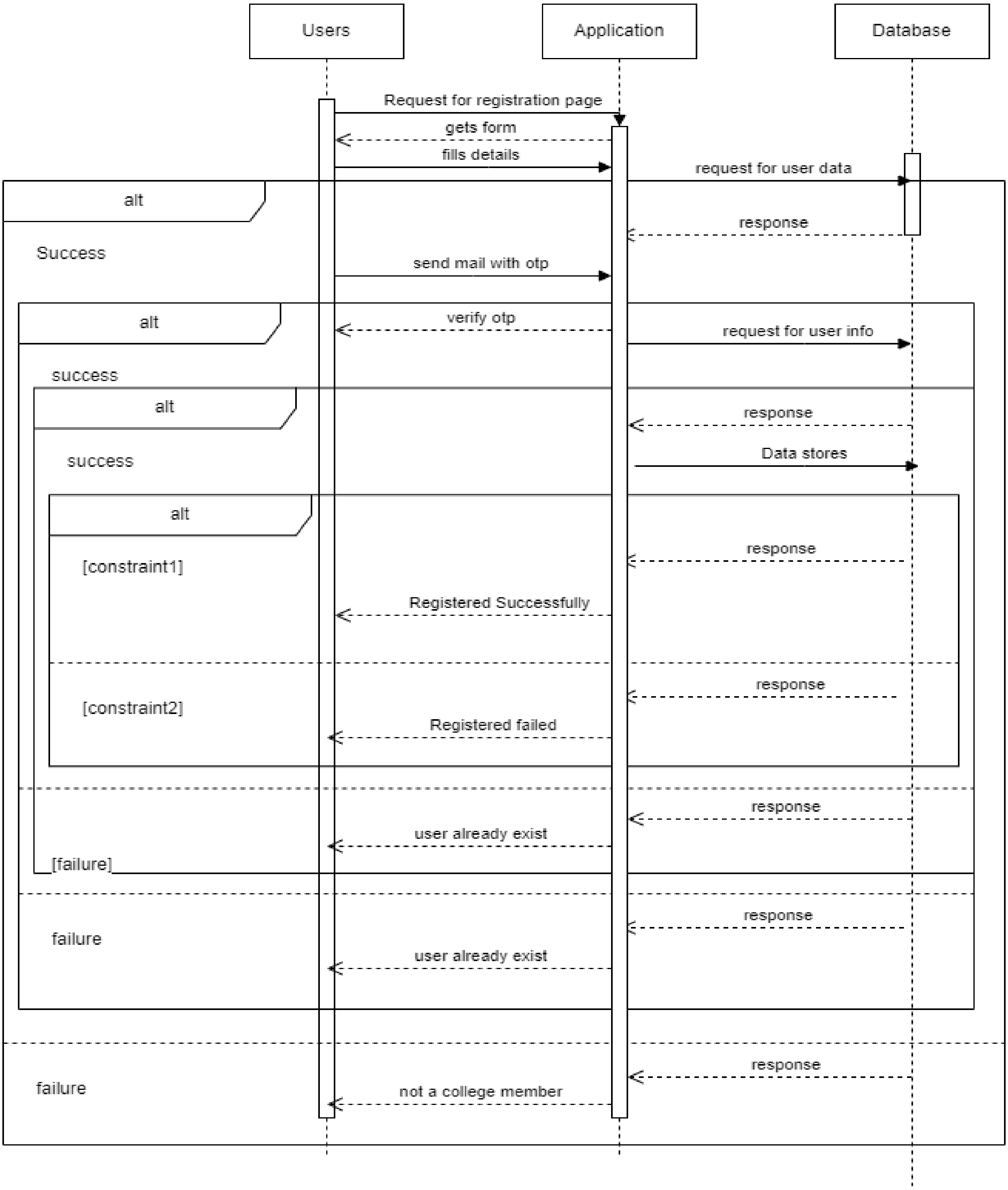
4.6. Sequence Diagram

A sequence diagram in a Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams typically are associated with use case realizations in the Logical View of the system under development.

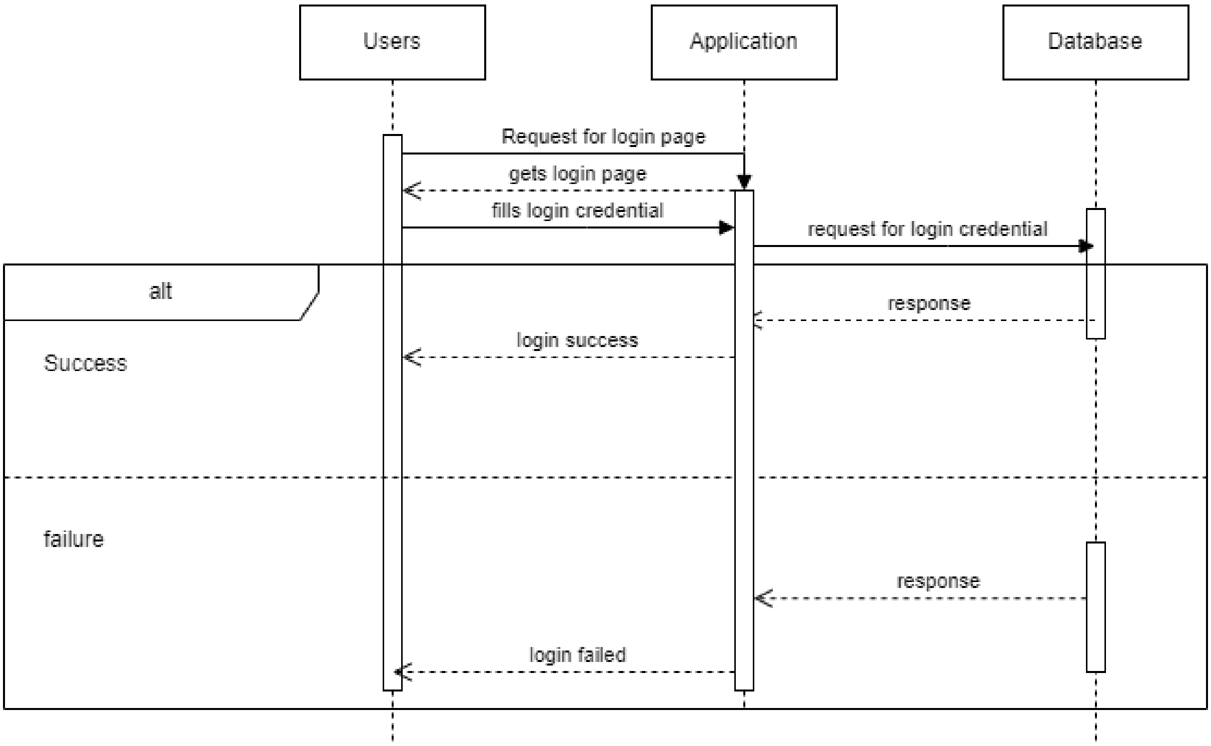
Diagram Notations:

Name	**Symbol**	**Description**
Synchronous Message		An instantaneous communication between objects that conveys information, with the expectation that an action will be initiated as a result.
Activation Box		Represents an active object during an interaction between two objects. The length of the bar represents the duration of an object's activeness.
Object		An object that is created, performs actions, and/or is destroyed during the lifeline
Alternative Fragment		Two or more message sequences exist, and a choice must be made between the two of them.

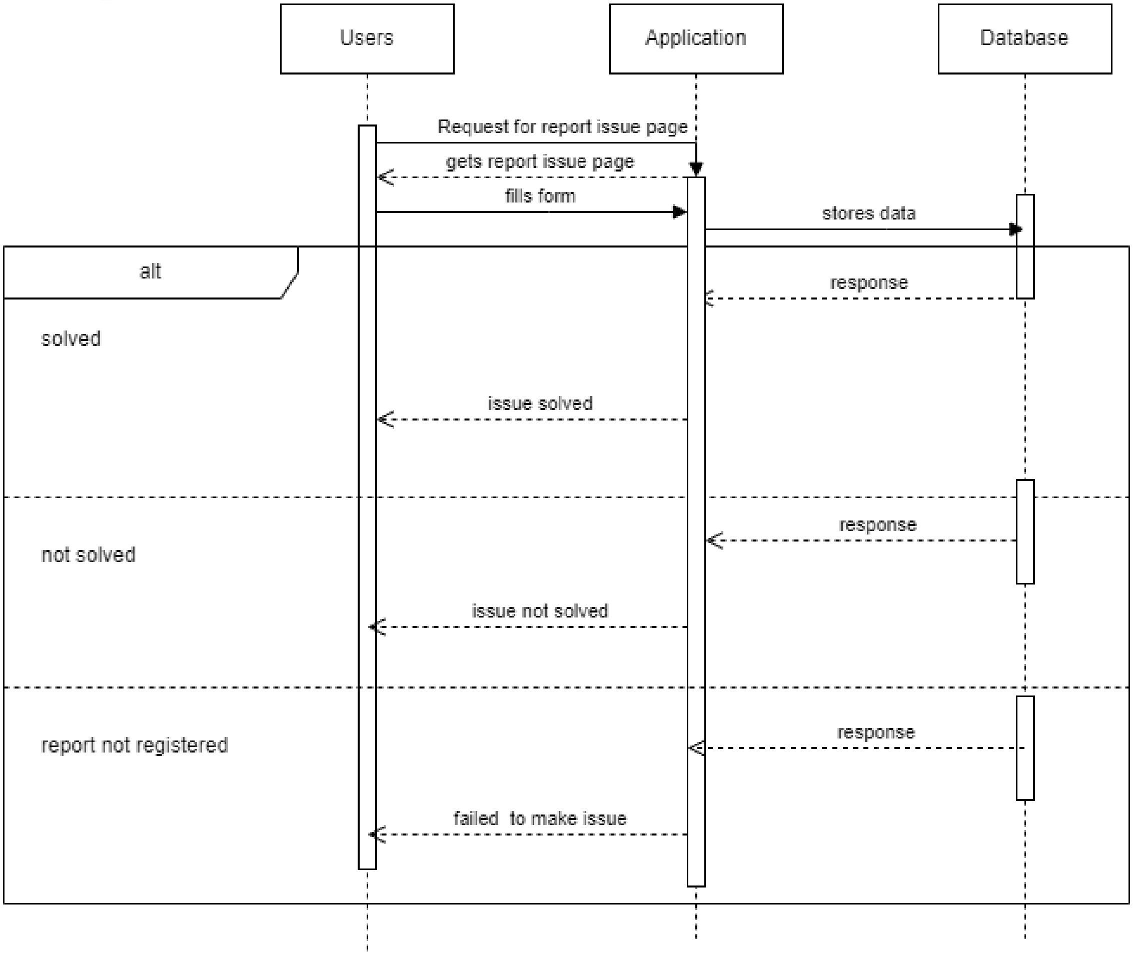
1. Registration



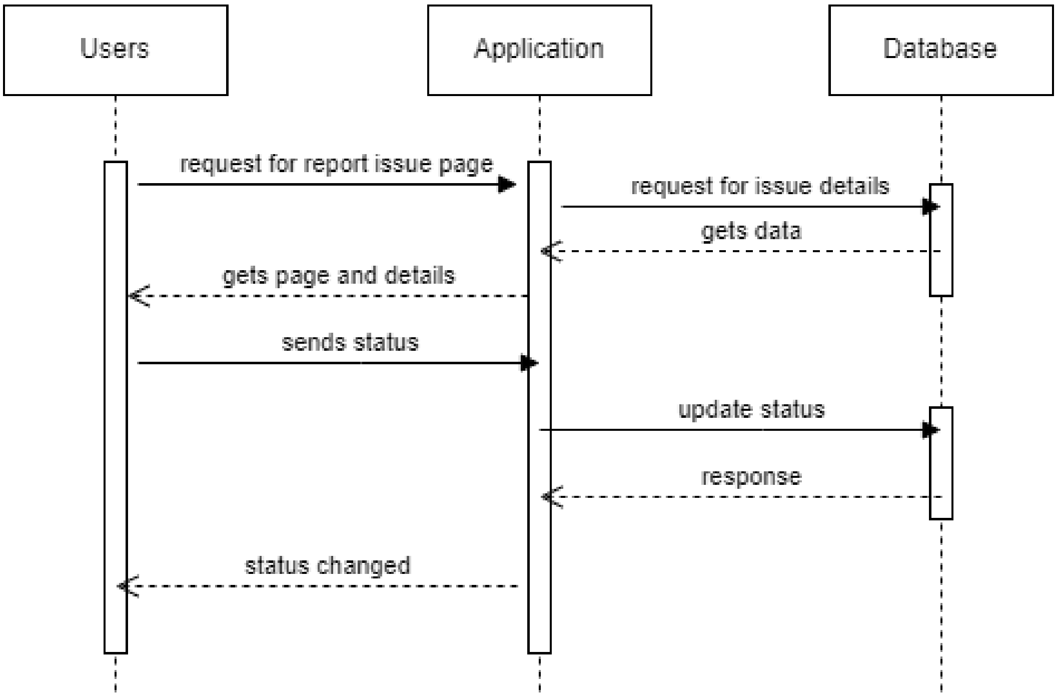
2. Login



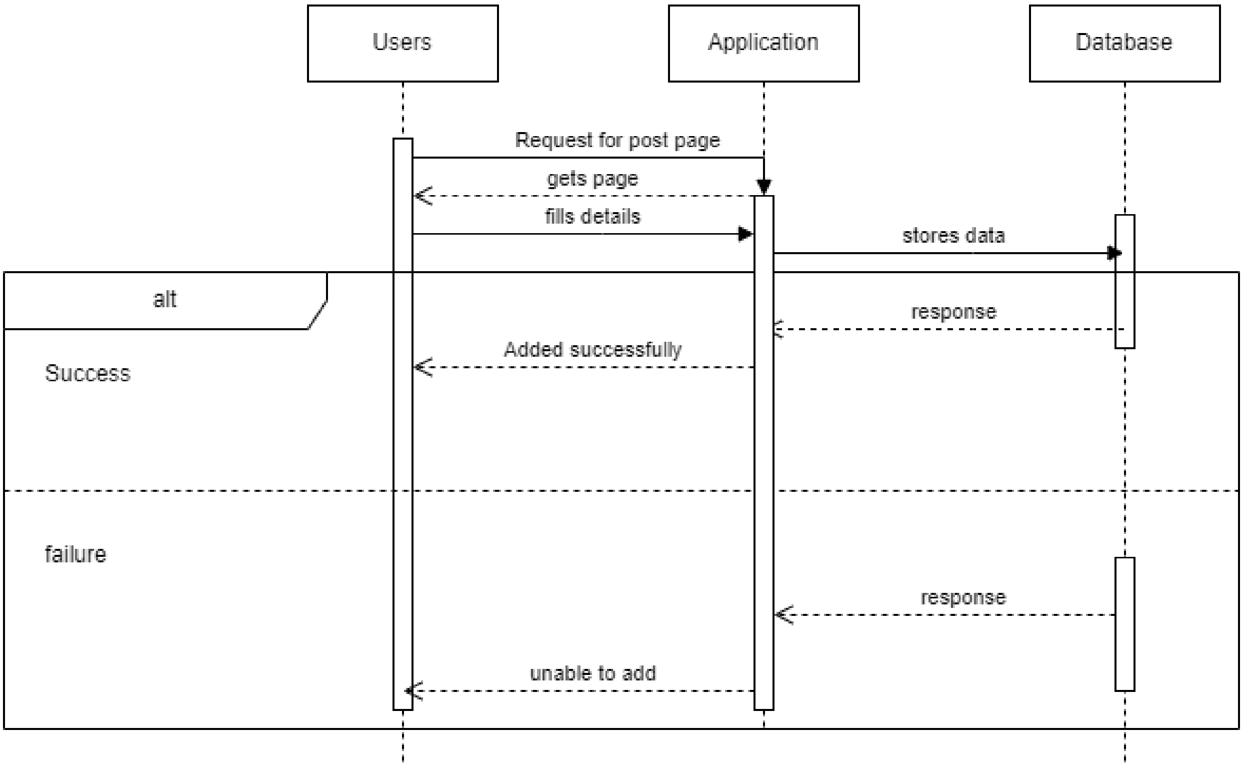
3. Report Issue



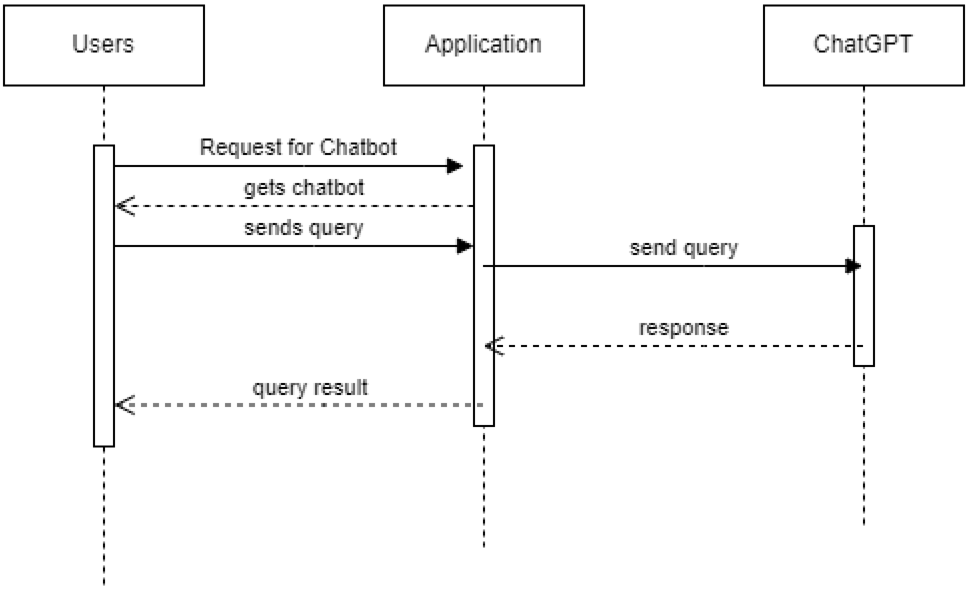
4. Resolve Issue



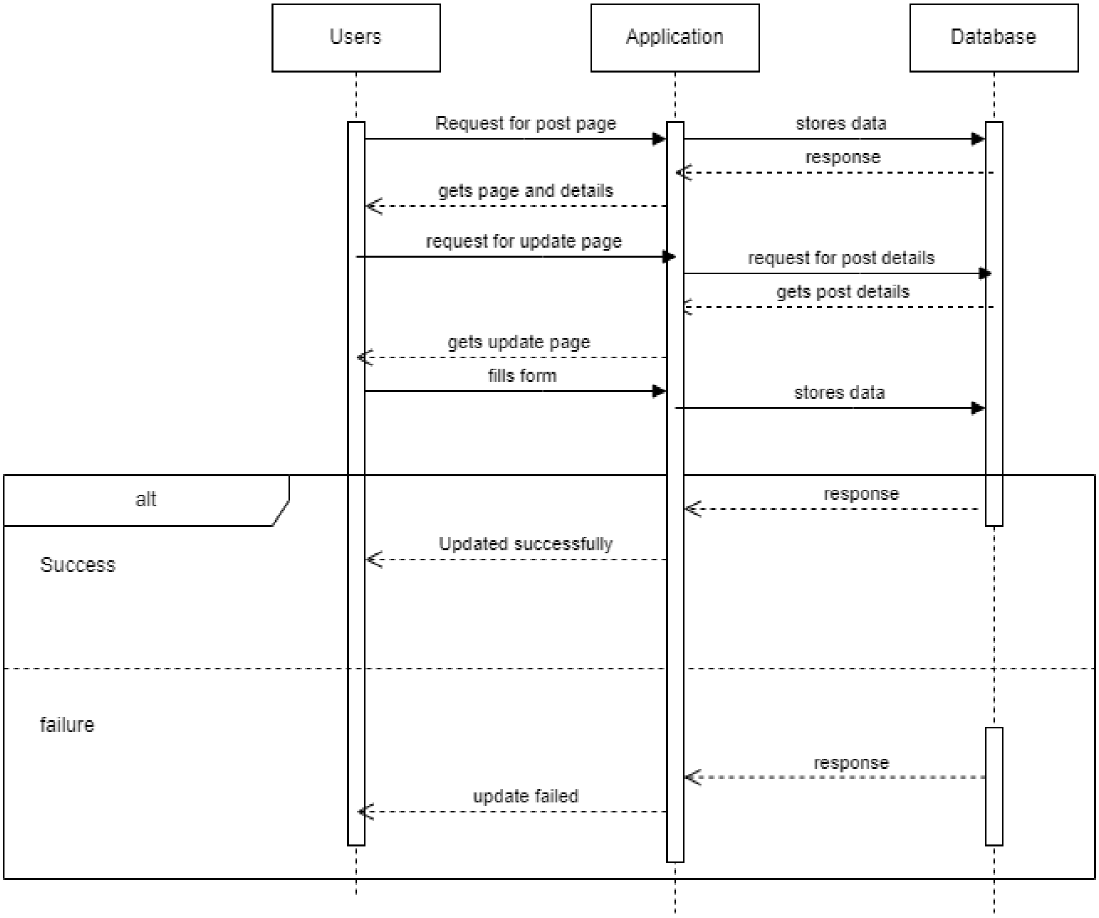
5. Add Post



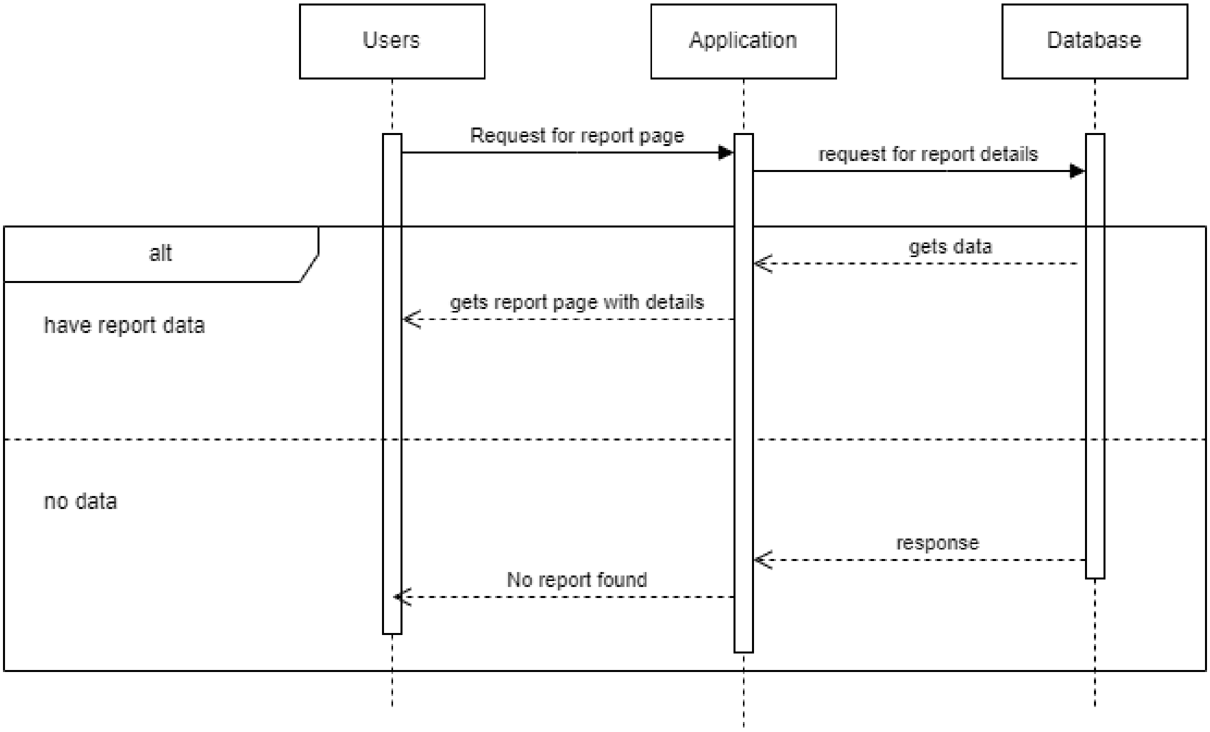
6. Chatbot



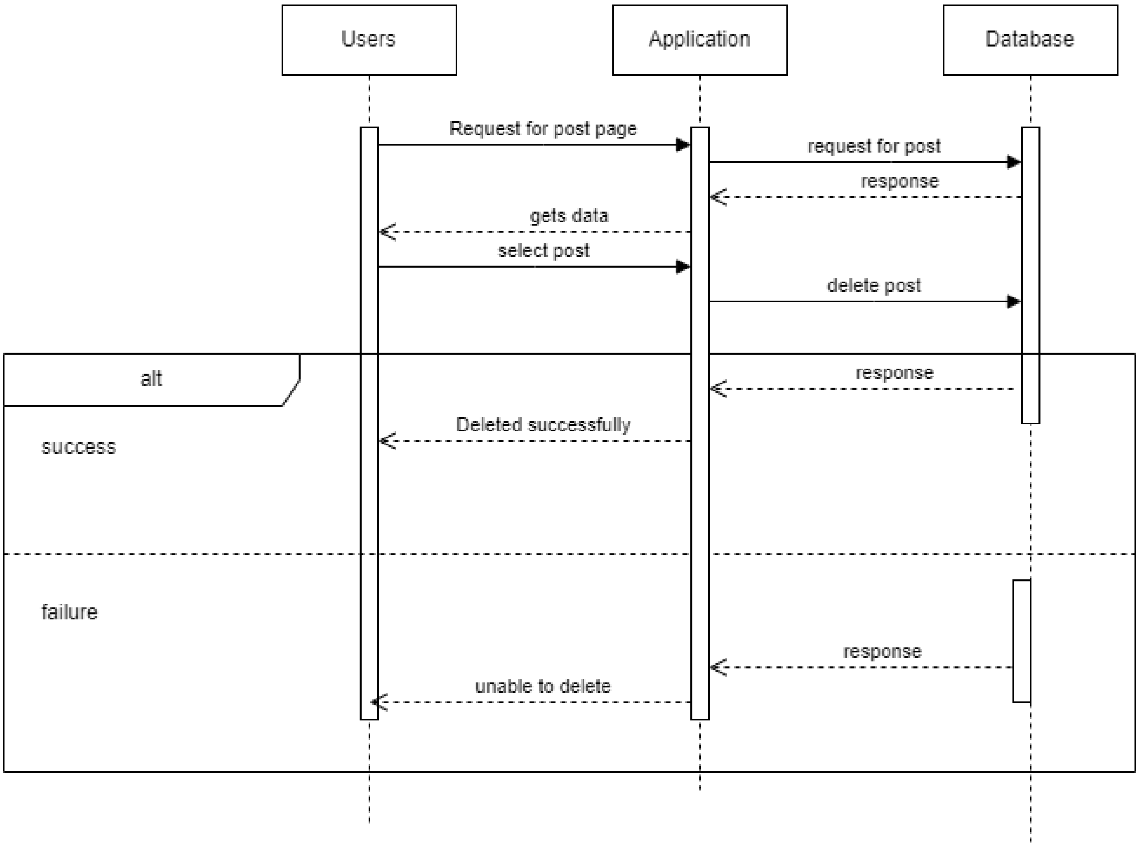
7. Update post



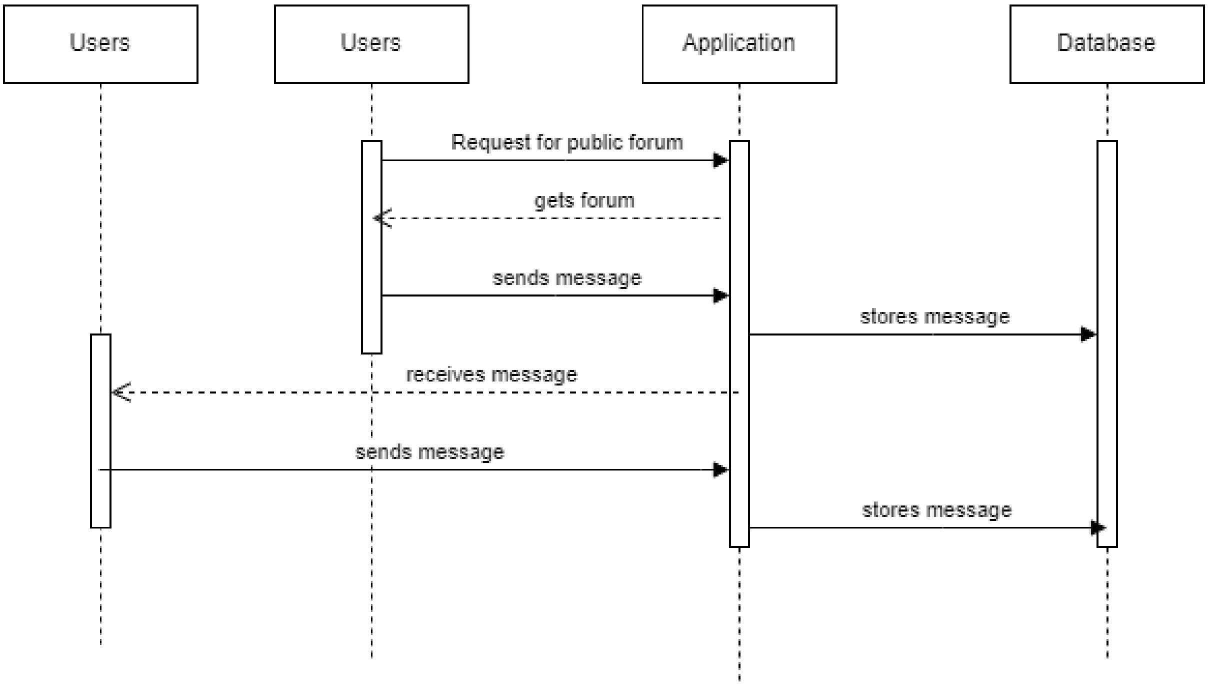
8. View Report



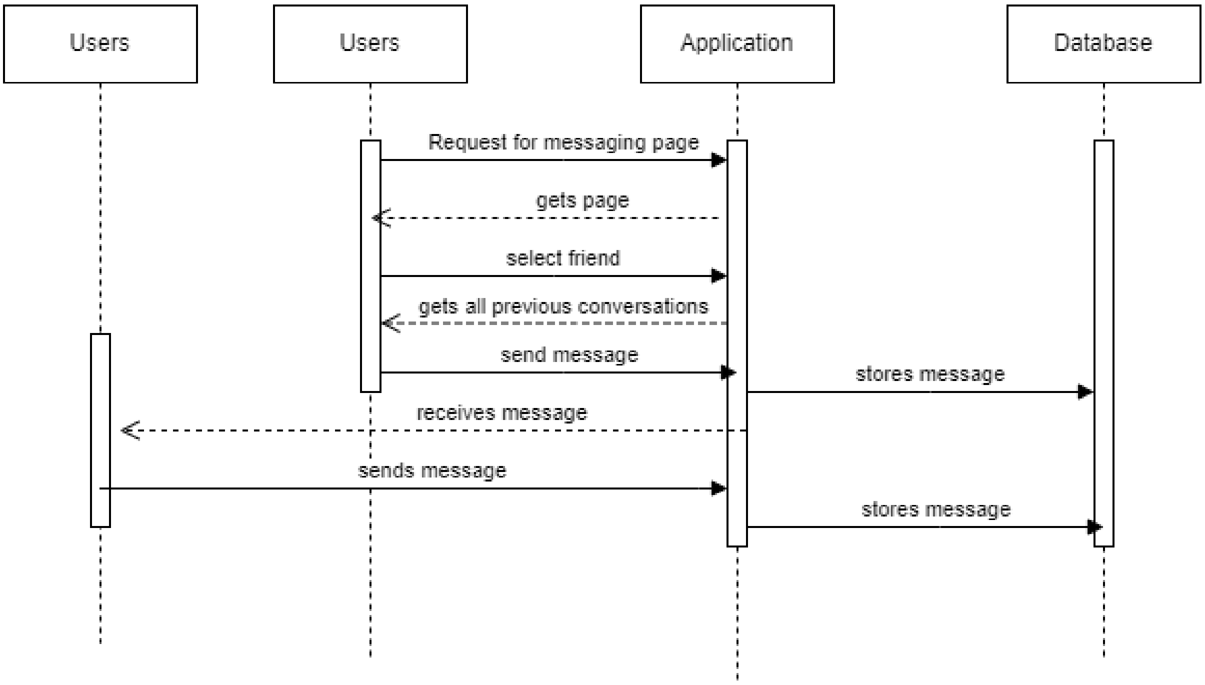
9. Delete post



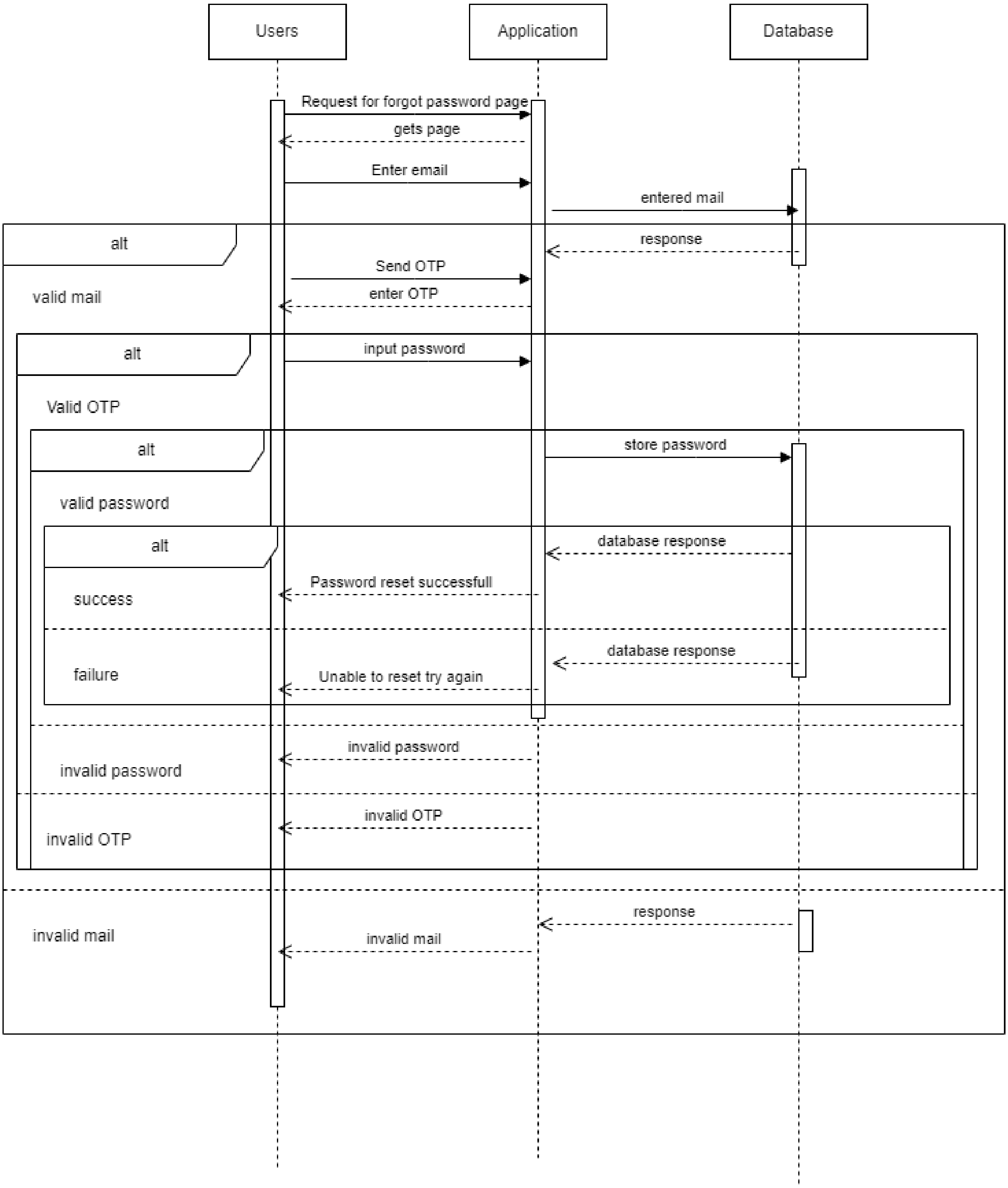
10. Anonymous messaging /Group chat



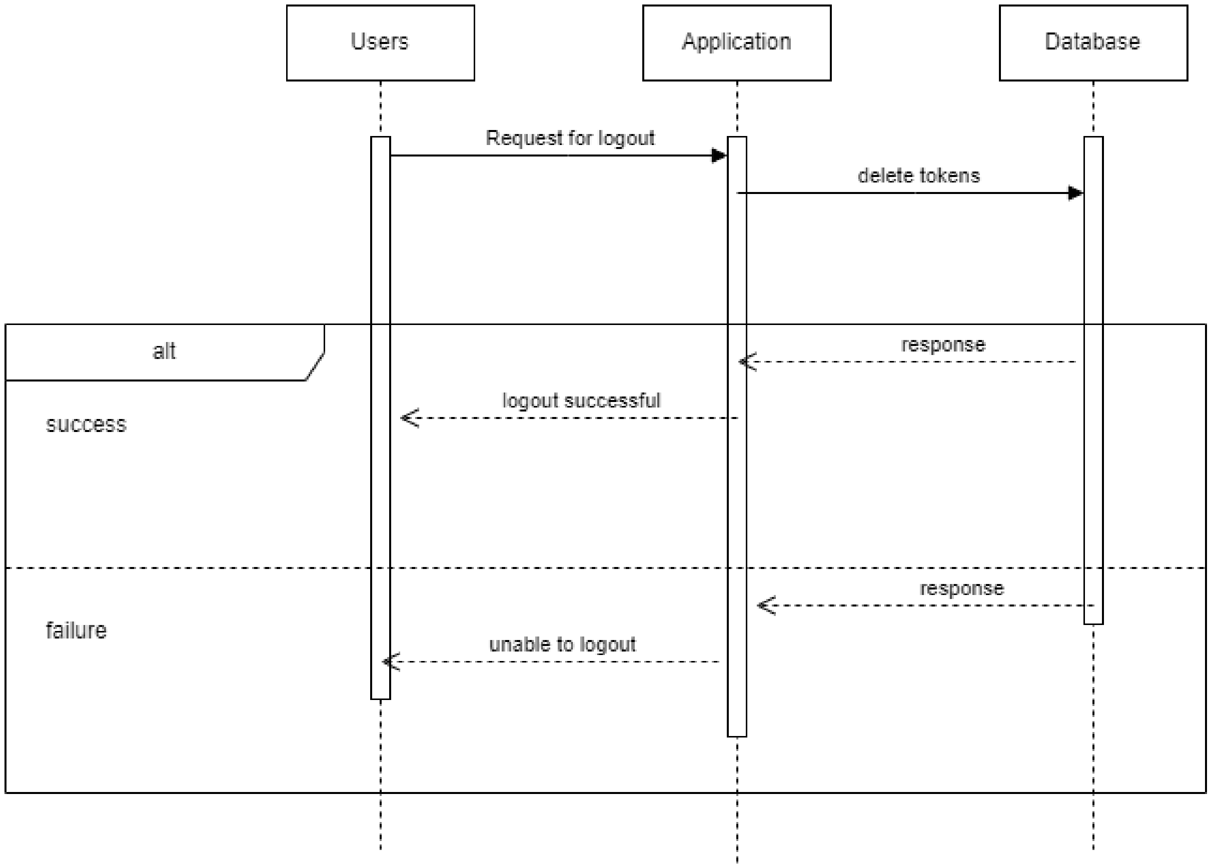
11. Messaging



12. Forgot Password


Users
Application
Database
Request for forgot password page
gets page
Enter email
entered mail
alt
response
valid mail
Send OTP
enter OTP
alt
input password
Valid OTP
alt
store password
valid password
alt
database response
Password reset successfull
success
failure
database response
Unable to reset try again
invalid password
invalid password
invalid OTP
invalid OTP
response
invalid mail
invalid mail

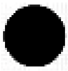
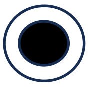
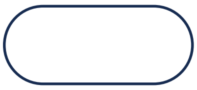
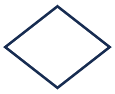
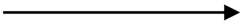
13. Logout


Users
Application
Database
Request for logout
delete tokens
alt
response
success
logout successful
response
failure
unable to logout

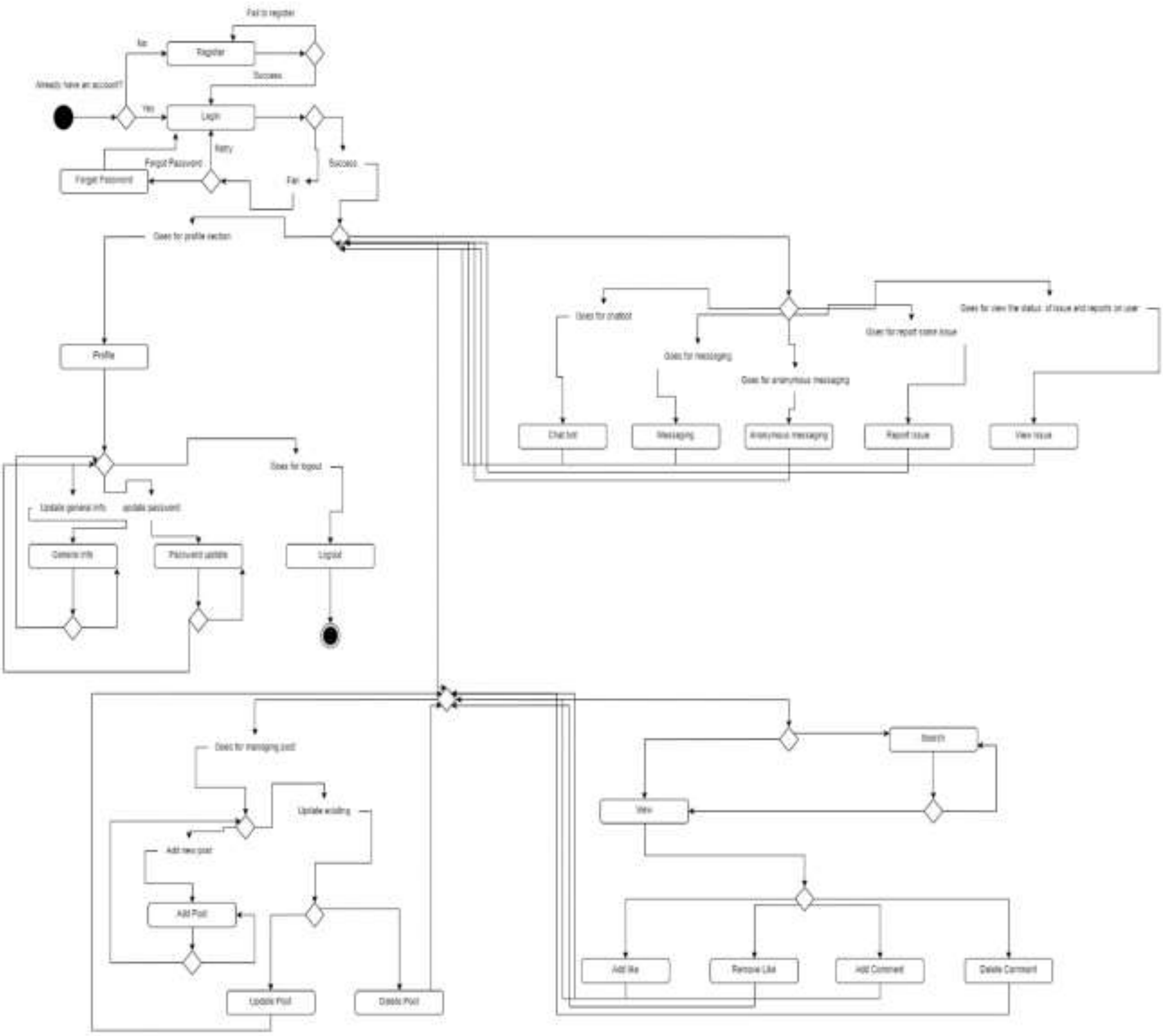
4.7. Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc

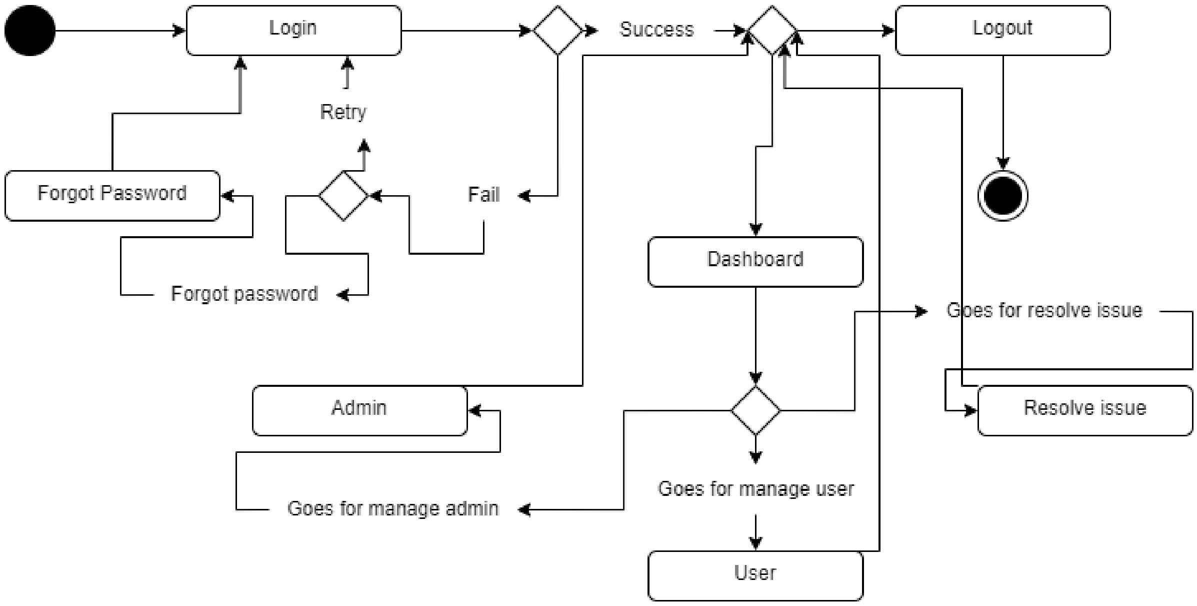
Diagram Notations:

Name	Symbol	Description
Initial State		This shows the starting point or first activity of the flow.
Final State		The end of the Activity diagram, also called as a final activity.
Action		It represents the activity to be performed.
Decision		A logic where a decision is to be made is depicted by a diamond.
Transition		A transition link represents control flow between nodes.

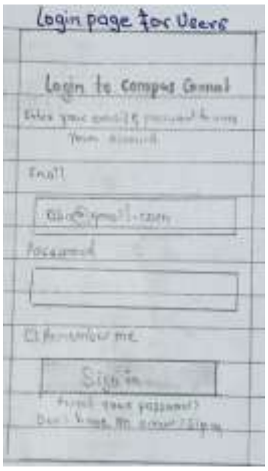
Activity diagram of user

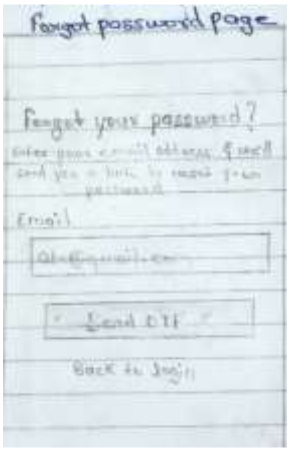


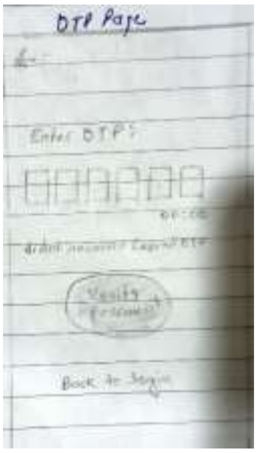
Activity diagram of admin

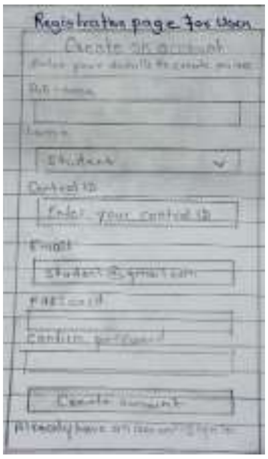


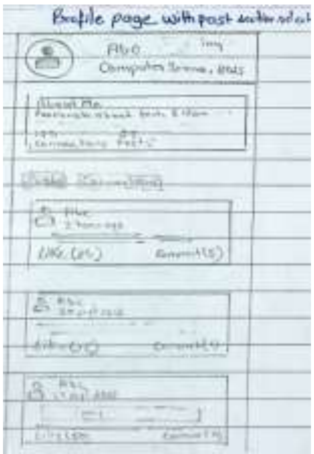
4.8 User Interface Design

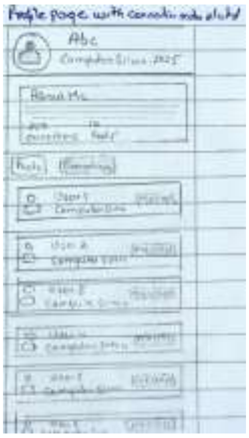


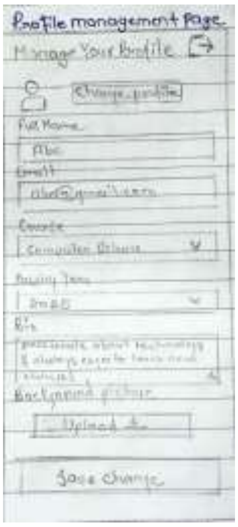


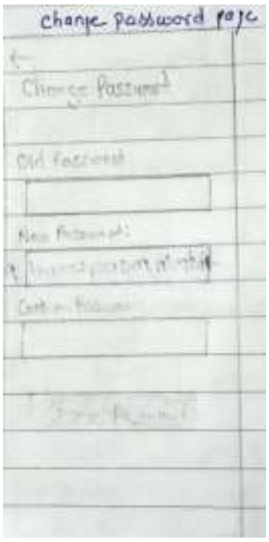


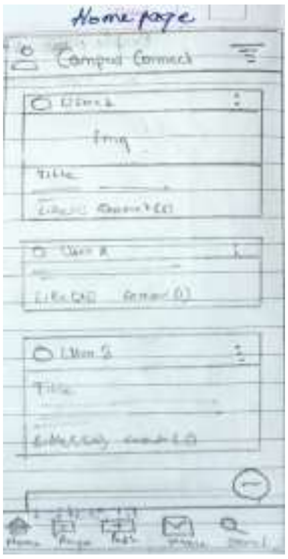


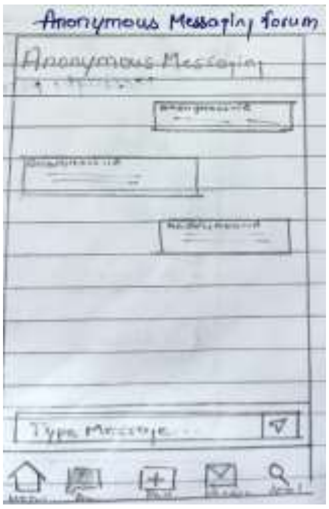


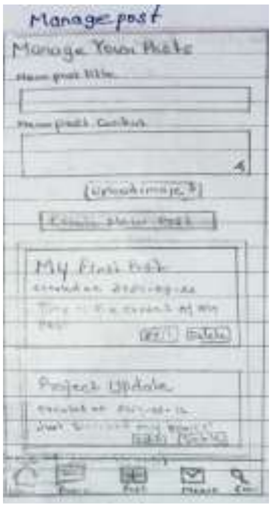


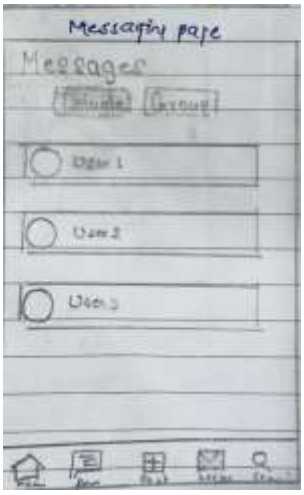


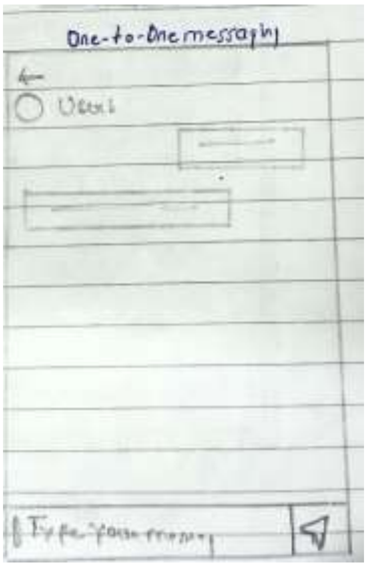


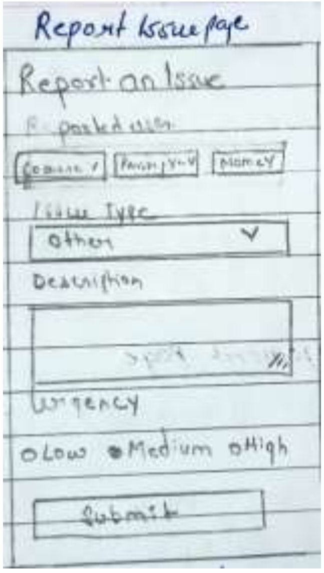


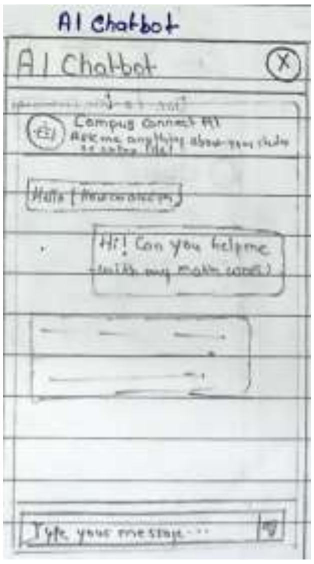

Anonymous Messaging forum
Anonymous Messaging
Type Message...

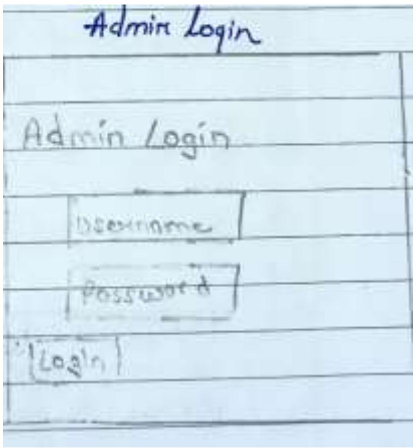

Manage post
Manage Your Posts
New post title
New post Content
Upload image
Create New Post
My First Post
Edit
Delete
Project Update
Post
Message

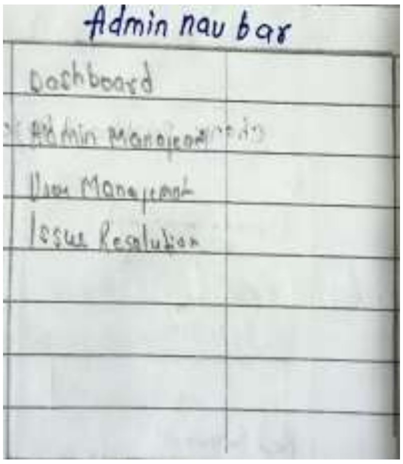

Messaging page
Messages
Single
Group
User 1
User 2
User 3

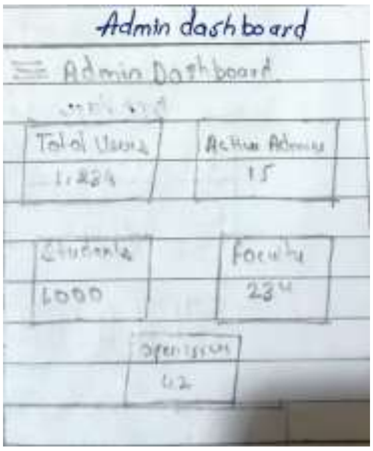

One-to-One messaging
User1
Type your message

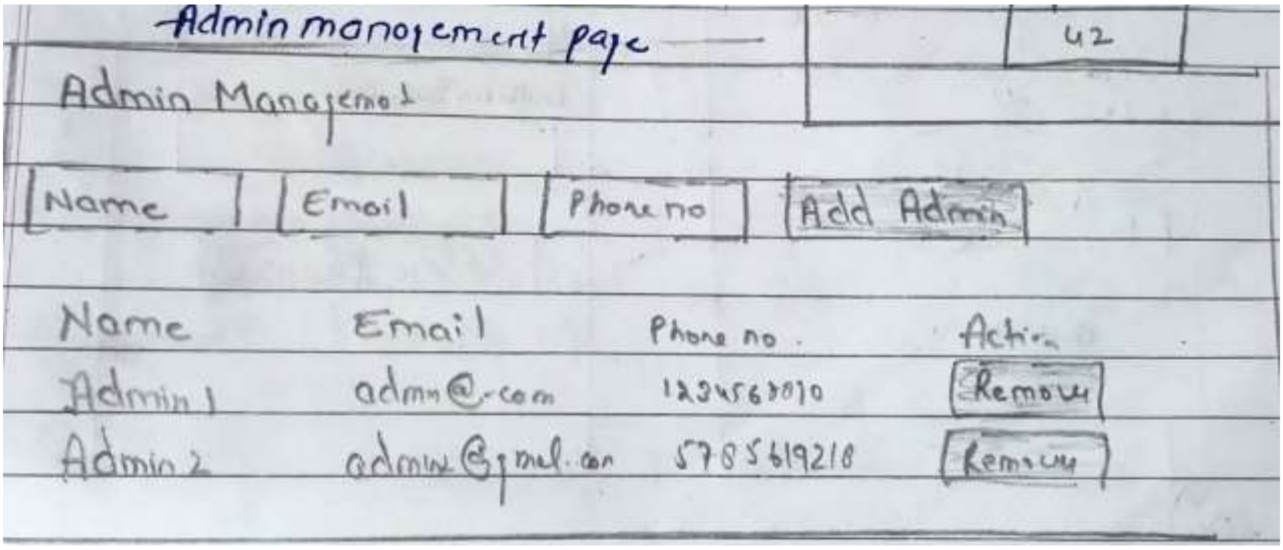

Report Issue page
Report an Issue
Issue Type
Other
Description
Urgency
○Low ●Medium ○High
Submit


AI Chatbot
AI Chatbot
Campus Connect AI
Ask me anything about your clubs
Hi! Can you help me with my math code?
Type your message...


Admin Login
Admin Login
Username
Password
Login


Admin nav bar
Dashboard
Admin Management
User Management
Issue Resolution


Admin Dashboard
Admin Dashboard
Total Users
1,234
Active Admins
15
Students
1000
Faculty
234
Open issues
42



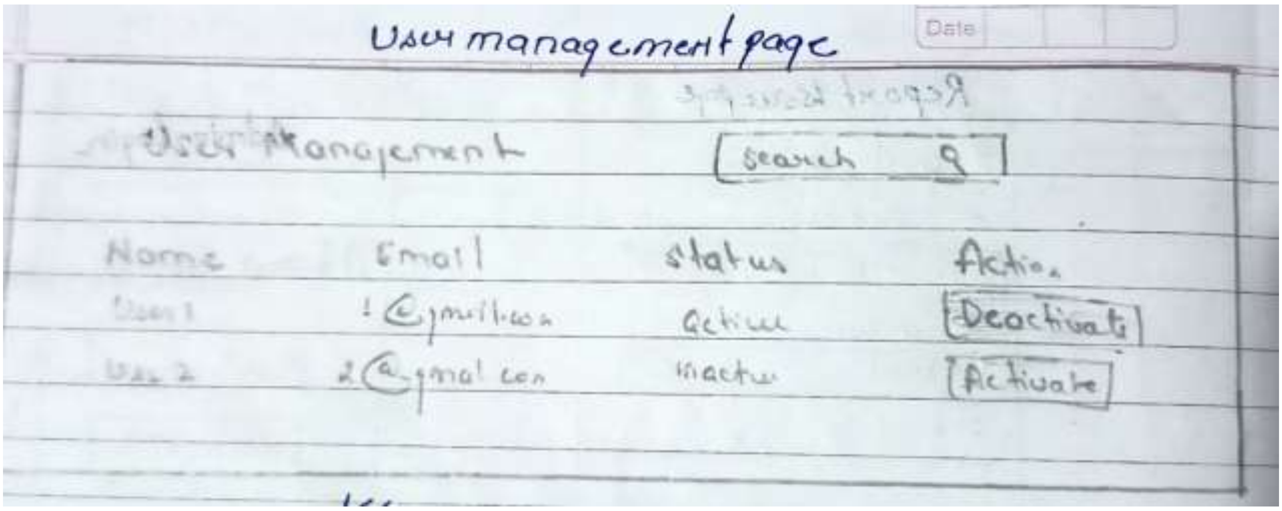
Admin manogement page

42

Admin Managemnt

[Name] [Email] [Phone no] [Add Admin]

Name	Email	Phone no.	Action
Admin 1	admn@-com	1234567890	Remove
Admin 2	admnw@gmail.con	5785619218	Remove

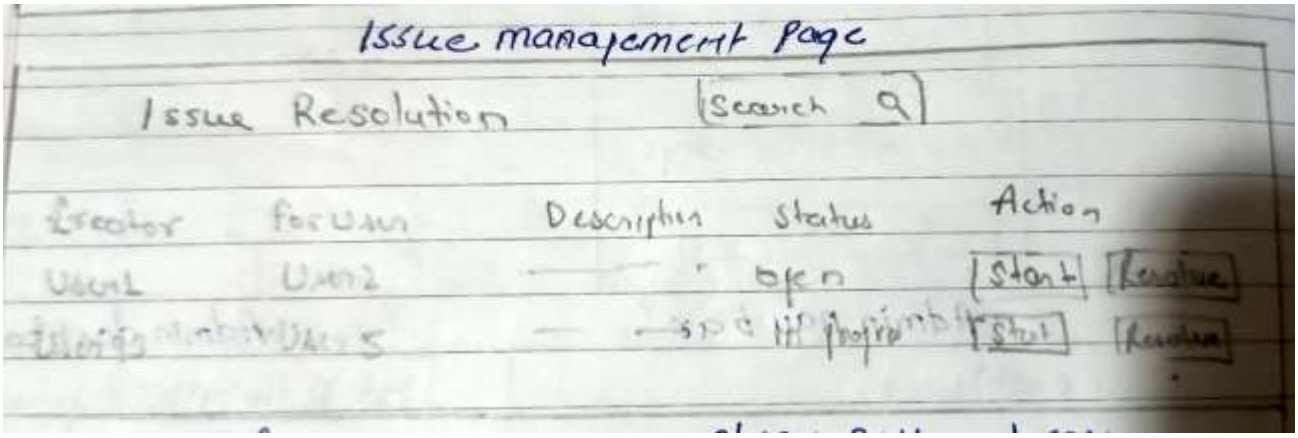


User management page

Date

User Management [search 🔍]

Name	Email	status	Action
User 1	1@gmail.com	active	Deactivate
User 2	2@gmal con	inactive	Activate



Issue management Page

Issue Resolution [Search 🔍]

Creator	For User	Description	Status	Action
User1	User2	——	open	Start Resolve
[illegible]	User 5	——	in progress	Start Resolve

4.9. Test Case Design

Test case no.	Test case	Expected outcome	Actual outcome	Remarks
1	Registration I am a:	Please select your role!	-	-
2	Registration I am a: Student Control ID:	Please enter a valid control id	-	-
3	Registration I am a: Student Control ID:123	Please a provide a valid Control id.	-	-
4	Registration I am a: Student Control ID: 4803099 Email: college.nitinsingh.com	Enter a valid email	-	-
5	Registration I am a: Student Control ID: 4803099 Email: college.nitinsingh@gmail.com Email: college.nitinsingh@gmail.com Password:	Password must be greater than 8 characters.	-	-
6	Registration I am a: Student Control ID: 4803099 Email: college.nitinsingh@gmail.com Email: college.nitinsingh@gmail.com Password: Nitin@123 Confirm Password: Nitin@123	Registration successful	-	-
7	Registration I am a: Faculty Email: faculty@gmail.com	Enter a valid mail		
8	Registration I am a: Faculty Email: faculty@vazecollege.net Password: Faculty@123 Confirm Password: Nitin@123	Password do not match	-	-
9	Registration I am a: Faculty Email: faculty@vazecollege.net Password: Faculty@123 ConfirmPassword:Faculty@123	registration successfull	-	-

10	Login Email: nitinsinghgmail.com	Enter a valid mail	-	-
11	Login Email: college.nitinsingh@gmail.com Password: Abc	Password must be greater than 8 character	-	-
12	Login Email: college.nitinsingh@gmail.com Password: NotaUser123	Incorrect email or password	-	-
13	Login Email: college.nitinsingh@gmail.com Password: Nitin@123	Login successfull	-	-
14	Forgot Password Email: college.nitinsingh.com	Invalid email	-	-
15	Forgot Password Email: college.nitin@gmail.com	User doesn’t exist	-	-
16	Forgot Password Email: college.nitinsingh@gmail.com	Your OTP sent to the email	-	-
17	Forgot Password OTP: 12345	Please fill all input	-	-
18	Forgot Password OTP: 123456	Invalid OTP	-	-
19	Forgot Password OTP: 228001	After 5 min reset token expired	-	-
20	Forgot Password OTP: 228023	OTP verified	-	-
21	Forgot Password Password: Raghav@911 Confirm Password: Dev51@121	Password do not match	-	-
22	Forgot Password Password: Raghav@911 Confirm Password: Raghav@911	Password changed successfully	-	-
23	Manage Profile Name: Nitin Singh Email: college.nitinsingh@gmail.com Course:	Please select your course	-	-
24	Manage Profile Name: Nitin Singh Email: college.nitinsingh@gmail.com Course: Computer Science Passing Year:	Please choose your passing year!	-	-
25	Manage Profile Name: Nitin Singh Email: college.nitinsingh@gmail.com	Profile information has been	-	-

	Course: Computer Science Passing Year: 2025 Bio: Passionate about technology Profile-pic: https://images.pexels.com/photos/1704488/pexels-photo-1704488.jpeg?auto=compress&cs=tinysrgb&w=400 Background-pic: https://images.pexels.com/photos/28549352/pexels-photo-28549352/free-photo-of-close-up-of-dslr-camera-lens-on-black-background.jpeg?auto=compress&cs=tinysrgb&w=400	changes successfully!		
26	Manage Profile Name: Nitin Singh Email: college.nitinsingh@gmail.com Course: Computer Science Passing Year: 2025 Bio: Profile-pic: Background-pic:	Profile information has been changes successfully!	-	-
27	Manage Profile Name: Faculty Email: faculty@vazecollege.net Course: Computer Science Bio: Profile-pic: Background-pic:	Profile information has been changes successfully!	-	-
28	Password change Old Password: Raghav@911 New Password: Raghav@911	New password must be different from old password!	-	--
29	Password change Old Password: Raghav@911 New Password: Nitin@123 Confirm Password: Nitin@999	Password do not match!	-	-
30	Password change Old Password: Raghav@999 New Password: Nitin@123 Confirm Password: Nitin@123	Incorrect old password	-	-
31	Password change Old Password: Raghav@911 New Password: Nitin@999 Confirm Password: Nitin@999	Password changed successfully!	-	-
32	Search Technology	Get data	-	-

33	Search Search a post that does not exist	Display "No results found" message	-	-
34	Report Issue Course:	Select Course	-	-
35	Report Issue Course: Computer Science Passing Year:	Select Passing Year	-	-
36	Report Issue Course: Computer Science Passing Year: 2025 Name:	Select name	-	-
37	Report Issue Course: Computer Science Passing Year: 2025 Name: Nitin Singh Issue type:	Select issue type!	-	-
38	Report Issue Course: Computer Science Passing Year: 2025 Name: Nitin Singh Issue type: False info Description:	Please provide some description	-	-
39	Report Issue Course: Computer Science Passing Year: 2025 Name: Nitin Singh Issue type: False info Description: Info provided in blogs are false. Urgency: Medium	Issue has been reported successfully!	-	-
40	Add Post Title:	Please enter a title!	-	-
41	Add Post Title: Hey	Title must be of atleast 5 character	-	-
42	Add Post Title: Information Technology Description:	Please enter the description of the post!	-	-
43	Add Post Title: Information Technology Description: Hlo	Description must be of atleast 10 characters!	-	-
44	Post Deletion Delete an existing post	The post should be removed	-	-

		and "Post deleted successfully" message is shown.		
45	Add Post Title: Information Technology Description: Utilizing computers software networks and digital system to manage process and share data.	Post added successfully!	-	-
46	Add Post Title: Information Technology Description: Utilizing computers software networks and digital system to manage process and share data. Img: https://images.pexels.com/photos/2047905/pexels-photo-2047905.jpeg?auto=compress&cs=tinysrgb&w=400	Post added successfully!	-	-
47	Anonymous Messaging Send an anonymous message	Message is posted without revealing the identity.	-	-
48	Group Creation Group Name:	Group name is required	-	-
49	**Group Messaging** Send a message to a group	Message is posted in the group and visible to all members.	-	-
50	File Upload Upload a file that exceeds the size limit	File size exceeds the limit	-	-
51	Chatbot Interaction Ask the chatbot an invalid query	I'm sorry, I don't understand	-	-
52	Register Email: college.nitinsingh@gmail.com Register with an email that’s already in the system.	Email already registered, please login.	-	-
53	Register Registration form without filling out any fileds.	All fields are required.	-	-

54	Resend OTP User requests to resend OTP for password recovery	OTP has been resent to your email.	-	-
55	Image Upload Upload an invalid file format (e.g., .exe) as a profile picture.	Invalid file format. Please upload an image.	-	-
56	Edit Blog Try to update a blog post with invalid content (e.g., HTML tags).	Invalid content format	-	-
57	Add Post Try to post a large number of blogs consecutively in a short period (limit for a day is 5)	Post limit exceeded. Try again tomorrow.	-	-
58	Delete Post Delete a blog post that has user comments.	Post and all associated comments are deleted	-	-
59	Messaging Attempt to send a blank message in a chat	Message cannot be empty	-	-
60	Messaging User receives a message while logged out of the app.	Message is delivered, and notification is shown once logged in.	-	-
61	Chatbot Send a query that exceeds the character limit.	Your query is too long	-	-
62	Search and Filter Search user who does not exist in the system	No user found	-	-
63	Filter Apply a date filter has no matching post	No data found	-	-