SOFTWARE ENGINEERING & CONCEPTS – LAB MANUAL

REConnect Messaging System

VISHNU VELVAN - 220701325

BE CSE / 2nd Year / E section

Software Concepts & Engineering - Lab Manual

Overview of the Project

Problem:

- **Overflooded college-specific communication channels:** Existing social media platforms cater to a general audience, making it difficult for college students, faculty, and staff to connect and share information relevant to their college community.
- **Information silos:** Important college-related announcements, events, and discussions might be scattered across various platforms (e.g., email, student organization pages), making it challenging for users to stay informed.
- **Lack of engagement:** Traditional communication methods like email can be impersonal and have low engagement.

Data-driven Approach:

This project will leverage data to:

- **Connect users:** The system will utilize college affiliations (e.g., email addresses) during registration to connect users within the same college community.
- **Targeted information sharing:** Students can follow specific departments, clubs, or professors to receive relevant updates and announcements.
- **Content discovery and trends:** By analyzing user activity and interests, the system can recommend relevant content and highlight trending topics within the college community.
- **Improved communication and feedback:** The platform will facilitate real-time communication between students, faculty, and staff, fostering a more engaged college experience.

Benefits for Users:

- **Centralized communication hub:** Stay informed about college events, announcements, and discussions in one place.
- **Enhanced engagement:** Participate in discussions, share ideas, and get real-time updates.
- **Streamlined information access through Integration:** No more searching through multiple platforms for college-related information.
- **Improved sense of community:** Build a stronger connection with your college peers, faculty and even alumni.

Social Media Integration:

- Integrate with existing social media platforms to allow users to share content from the college platform to their wider social networks, potentially increasing reach and college awareness.
- Leverage existing social media logins for a more convenient user experience.
- Integrating social media platforms adds complexity and requires careful consideration of user privacy and data security.

Business Architecture Diagram

Business Need:

REConnect aims to address the shortcomings of current communication methods within colleges and universities. These shortcomings include:

- **Scattered Information Flow:** Important announcements, updates, and resources are often disseminated through various channels (email, departmental websites, student organization pages), making it difficult for users to stay informed.
- **Inefficient Communication:** Distributed information across multiple platforms is exhausting to manage and resource-intensive.

Current Process (as-is):

- **Faculty/Staff:** Create announcements, updates, or resources.
- **Distribution:** Utilize various channels like email, whatsapp, instagram or physical posters throughout the campus.
- **Students:** Check various communication channels periodically for updates.
- **Engagement:** Interaction may be limited to infrequent email replies or passive information consumption.

Business Problems:

- **Inefficiencies:** Managing multiple communication channels becomes exhausting.
- **Information Gaps:** Students may miss crucial information due to scattered communication.
- **Poor Decision Making:** Lack of data on student engagement makes it difficult to improve communication strategies.

Solution:

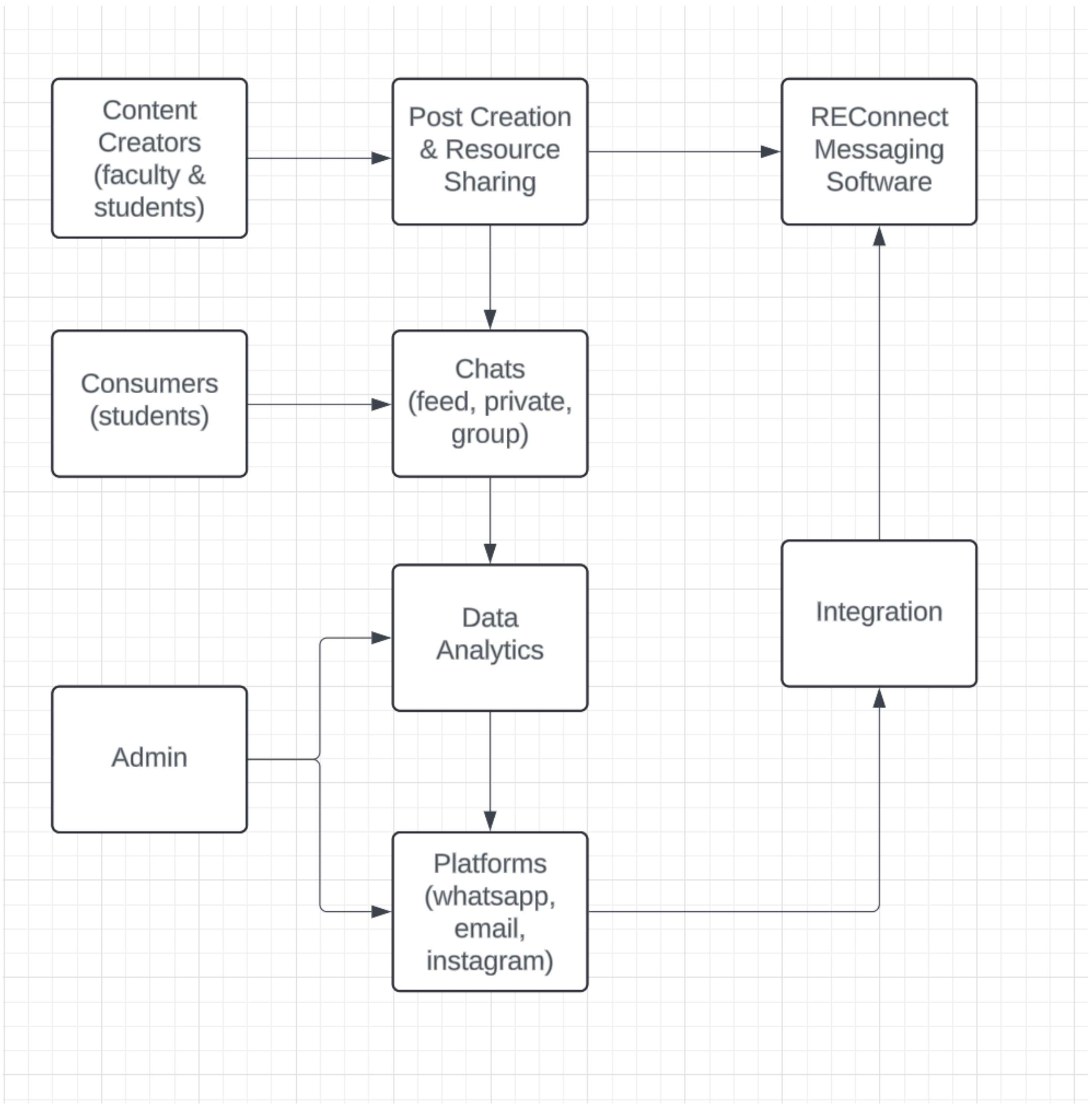
REConnect provides a centralized platform for streamlined communication, fostering engagement and improving information accessibility. This translates to:

- **Increased Efficiency:** Streamlined communication channels save time and resources.
- **Enhanced Engagement:** Real-time interaction fosters a more connected community.
- **Improved Information Access:** Centralized platform ensures easy access to relevant updates.
- **Data-Driven Insights:** User data informs decisions to improve communication and campus life.

Business Architecture Diagram:

The business architecture diagram will consist of the following components:

- **User Personas:** Faculty/Staff, Students, Administrators
- **Business Functions:** Information Dissemination, Content Management, User Management
- **Business Processes:** Announcement Creation, Resource Sharing, Discussion Forums, Group Chats
- **Information Systems:** College Connect Platform, Existing College Systems (e.g., student information system)
- **Data & Analytics:** User Activity Data, Content Engagement Metrics


Content Creators (faculty & students)
Post Creation & Resource Sharing
REConnect Messaging Software
Consumers (students)
Chats (feed, private, group)
Data Analytics
Integration
Admin
Platforms (whatsapp, email, instagram)

Business Architecture for REConnect Messaging System

Requirements as User Stories

- **User Story 1:** As a user, I want to see all my chats, individual chats, and group chats, so I can stay organized and keep track of my conversations. (This user story focuses on the ability to view a list of chats.)
- **User Story 2:** As a user, I want to prioritize important messages to be seen at the top so I can easily find and respond to the most important messages. (This user story focuses on the ability to prioritize messages.)
- **User Story 3:** As a user, I want buttons for jumping between the different pages (chats, contacts, settings, etc.) so I can easily navigate the application. (This user story focuses on the navigation of the application.)
- **User Story 4:** As a user, I want to post images and videos so I can share multimedia content with my contacts. (This user story focuses on the ability to send images and videos.)
- **User Story 5:** As a user, I want the UI to be easy to navigate through and all the functionalities to be easily accessible. (This user story focuses on the user interface and user experience of the application.)
- **User Story 6:** As a user, I want to have the buttons to jump to my profile and settings so I can manage my account information. (This user story focuses on the ability to access user profiles and settings.)
- **User Story 7:** As a user, I want the home page to have images and/or videos with the posts so I can see a quick preview of the content.
- **User Story 8:** As a user, I want to share my dept., section, etc. so that other users can find me and learn more about me.
- **User Story 9:** As a user, I want to find and connect with alumni to learn about their career paths and network for professional opportunities.
- **User Story 10:** As a user, it should be secure to prevent others from unauthorized access to my messages. (This user story focuses on the security of the application.)

Non-Functional Requirements

1. **Usability:**
- Friendly User-Interface
- Accessibility
2. **Maintainability:**
- Modular Design
- Documentation
- Robust
3. **Performance:**
- Scalability
- Response Time
- Availability
4. **Security:**
- Authentication and Authorization
- Data Security
- Profanity system to prevent inappropriate posts.

Poker Planning

Sure, let's do a poker planning session for these user stories and non-functional requirements. We'll use the Fibonacci sequence for estimation, with 1 being the smallest and 13 being the largest.

User Stories:

1. User Story 1: Seeing all chats - 3
2. User Story 2: Prioritizing messages - 5
3. User Story 3: Navigation buttons - 2
4. User Story 4: Sending images and videos - 8
5. User Story 5: UI/UX accessibility - 3
6. User Story 6: Accessing profiles and settings - 2
7. User Story 7: Home page media previews - 3
8. User Story 8: Sharing personal information - 3
9. User Story 9: Connecting with alumni - 5
10. User Story 10: Security measures - 8

Non-Functional Requirements:

1. Usability:
   - Friendly UI: 3
   - Accessibility: 5
2. Maintainability:
   - Modular design: 3
   - Documentation: 2
   - Robustness: 5
3. Performance:
   - Scalability: 8
   - Response Time: 5
   - Availability: 5
4. Security:
   - Authentication and Authorization: 8
   - Data Security: 8
   - Profanity filter: 3

Architecture Diagram

Components:

- **User Interface**
- **API Gateway:** TO integrate existing platforms
- **Business Logic Layer:** Handles core functionalities like announcement creation, resource management, and user interactions.
- **Data Access Layer:** Interacts with the database to store and retrieve data.
- **Database:** Stores all application data persistently (e.g., MySQL, MongoDB)
- **Notification Service:** Sends real-time customizable notifications about new announcements or messages.
- **Logging and Monitoring:** Logs application events, errors, and performance metrics for troubleshooting and analysis.

Interactions:

1. User interacts with the User Interface (Web/Mobile App).
2. User Interface sends requests to the API Gateway.
3. API Gateway authenticates/authorizes the request and routes it to the Business Logic Layer.
4. The Business Logic Layer processes the request, interacts with the Data Access Layer to access data, and generates a response.
5. API Gateway sends the response back to the User Interface.
6. Logging and Monitoring components capture relevant events throughout the process.

Error Handling:

- The API Gateway returns appropriate error messages in case of authentication failures or invalid requests.
- The Business Logic Layer handles business-specific errors and returns informative messages.
- All errors are logged for troubleshooting and analysis.

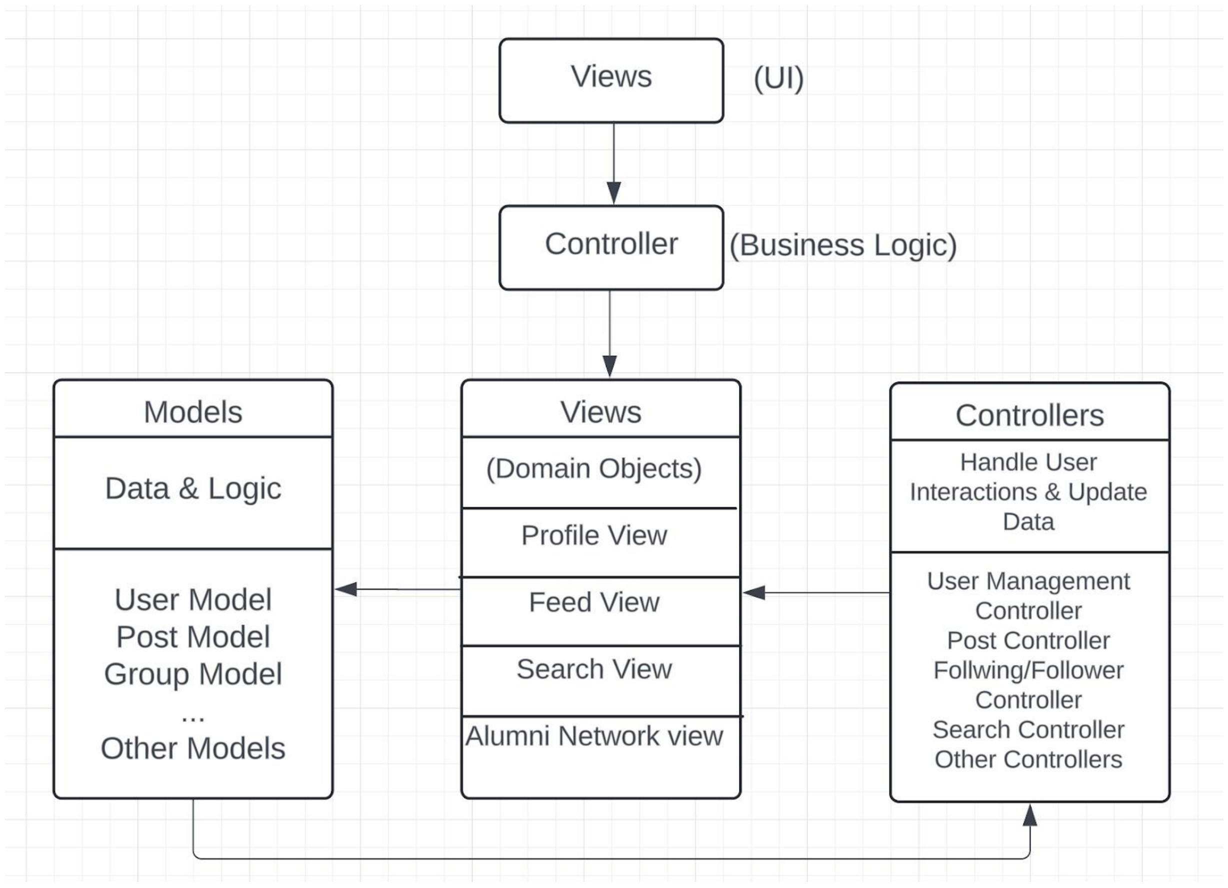
Data Storage:

- A relational database (e.g., MySQL, MongoDB) is used to store structured data like announcements, user profiles, and group information.

Architecture Pattern: MVC Architecture

- REConnect utilizes a combination of microservices & MVC architecture. The application is broken down into smaller, independent services (modules) that communicate through APIs.
- This approach offers several benefits:
  - **Scalability:** Individual services can be scaled independently based on their needs.

- **Maintainability:** Easier to develop, test, and deploy changes in smaller services.
- **Fault Tolerance:** Failure in one service doesn't necessarily impact the entire system.

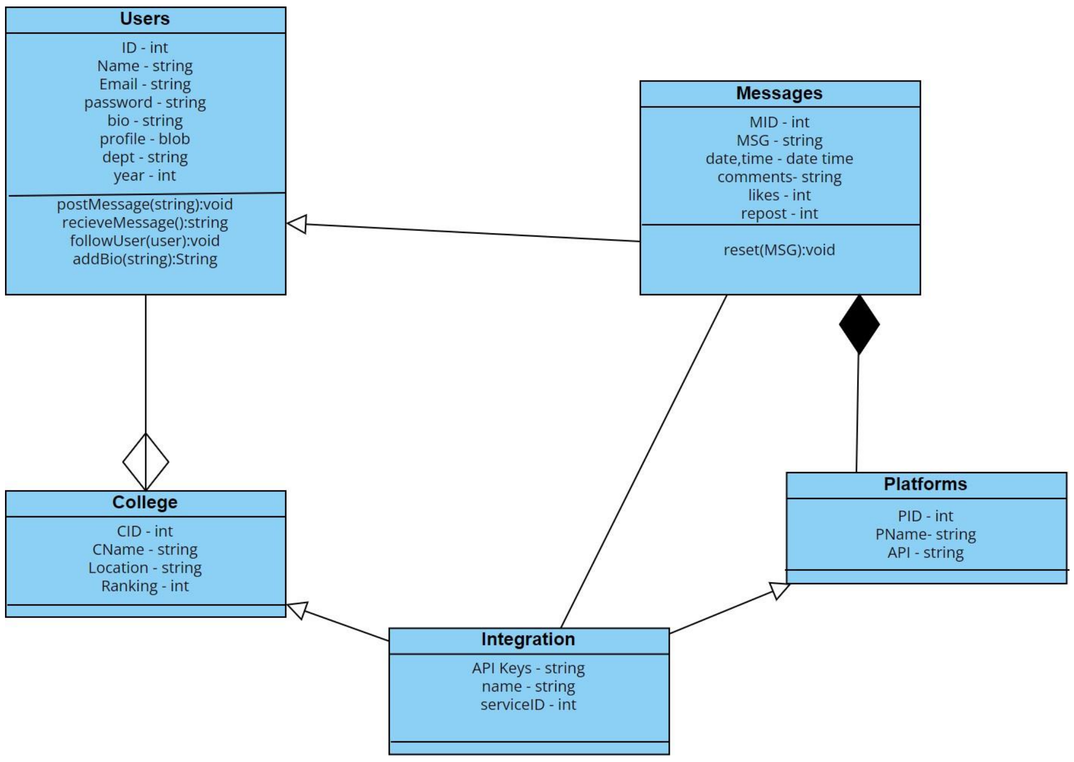


MVC Architecture

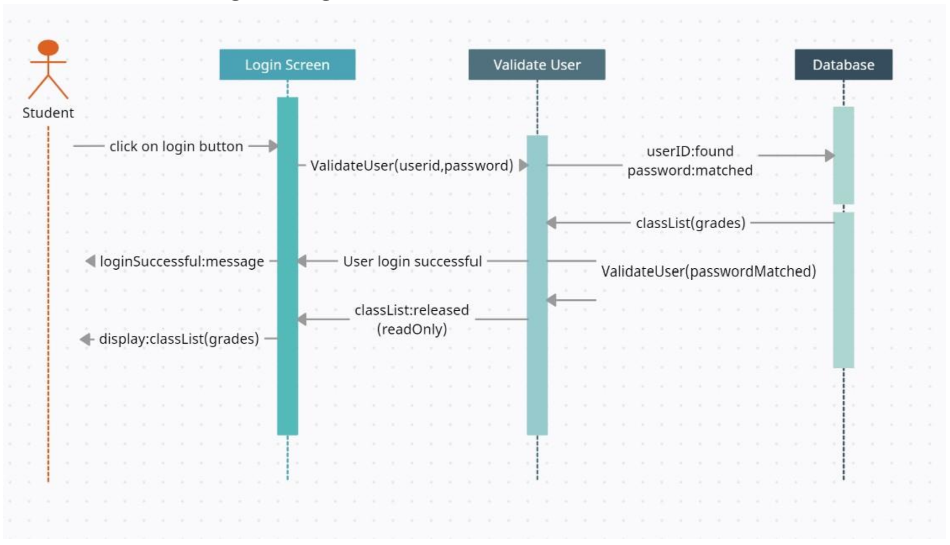
Design Principles:

- **Separation of Concerns:** Each module focuses on a specific functionality, promoting modularity and maintainability.
- Services interact through single-entry APIs, minimizing dependencies, promoting scalability, and improving security.
- **Event-driven Architecture:** Can be implemented for real-time notifications, where services communicate through events.
- **Logging and Monitoring:** Crucial for troubleshooting, performance analysis, and debugging.

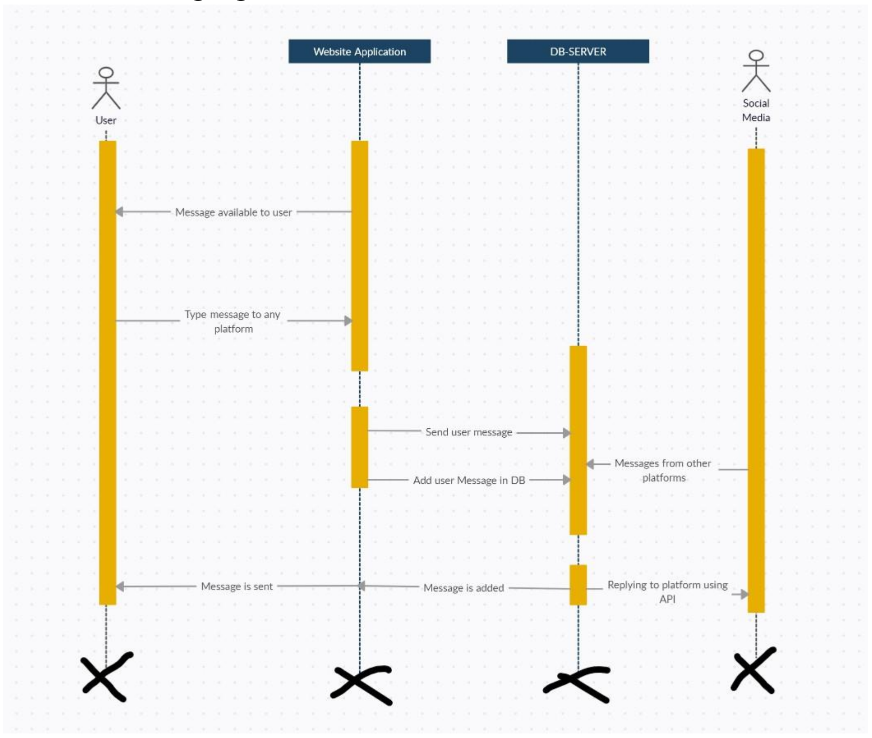
- Class diagrams



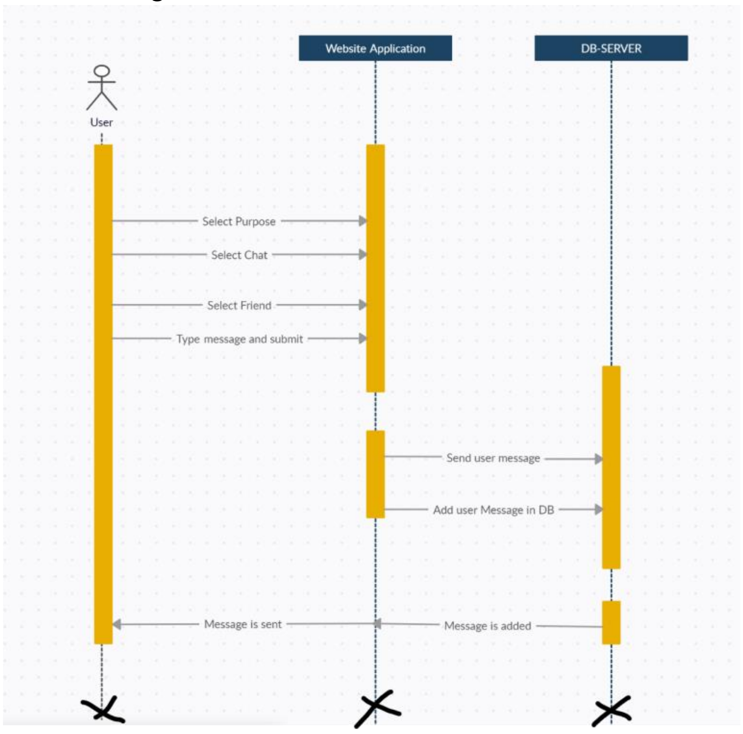
- Sequence diagram
  1. Login Page



2. Messaging



3. Integration with API



Test Strategy

User Story 1: As a student, I want to view a list of upcoming announcements to stay informed about important college events.

Test Cases:

- **Happy Path:**
  - Login as a student user.
  - Verify a list of upcoming announcements is displayed.
  - Verify announcements are sorted by chronology.
- **Error Scenarios:**
  - Login with invalid credentials (incorrect username/password). (Expected behavior: Login error message)
  - Attempt to access announcements without logging in. (Expected behavior: Access denied message)
  - Verify the system gracefully handles situations where there are no upcoming announcements. (Expected behavior: Empty announcement list message)

User Story 2: As a student, I want to join a club discussion forum to connect with other members and ask questions.

Test Cases:

- **Happy Path:**
  - Login as a student user.
  - Navigate to the specific club discussion forum.
  - View a list of existing discussion threads.
  - Open a thread and view comments from other members.
  - Post a new comment within the thread to share your question.
- **Error Scenarios:**
  - Login with invalid credentials. (Expected behavior: Login error message)
  - Attempt to access a club forum without logging in. (Expected behavior: Access denied message)
  - Verify the system handles situations where a club doesn't exist or has no forum. (Expected behavior: Error message or empty forum display)
  - Try to post a comment exceeding the character limit. (Expected behavior: Character limit error message)

User Story 3: As an administrator, I want to manage user accounts and ensure system security.

Test Cases:

- **Happy Path:**
  - Login as an administrator.

  - Access the user management console.
  - View a list of all registered users with their roles (student, faculty, admin).
  - Deactivate a user account if necessary.
- **Error Scenarios:**
  - Login with invalid administrator credentials. (Expected behavior: Login error message)
  - Attempt to access user management without administrator privileges. (Expected behavior: Access denied message)
  - Verify the system prevents unauthorized attempts to modify administrator accounts. (Expected behavior: Access denied message)

User Story 4: As a student, I want real-time custom notifications about new announcements and messages.

Test Cases:

- **Happy Path:**
  - Login as a student user and enable push notifications in the app settings.
  - Verify the student receives a notification when a new announcement relevant to their interests is created (they can specify their interests by following certain accounts).
  - Verify the student receives a notification when they receive a new message within a discussion forum.
- **Error Scenarios:**
  - Login with invalid credentials. (Expected behavior: Login error message)
  - Verify notifications are not sent to users who haven't opted-in.
  - Verify the system gracefully handles situations where push notification delivery fails due to internet connectivity issues on the user's device. (Expected behavior: Notification delivery attempt with potential retry mechanism)

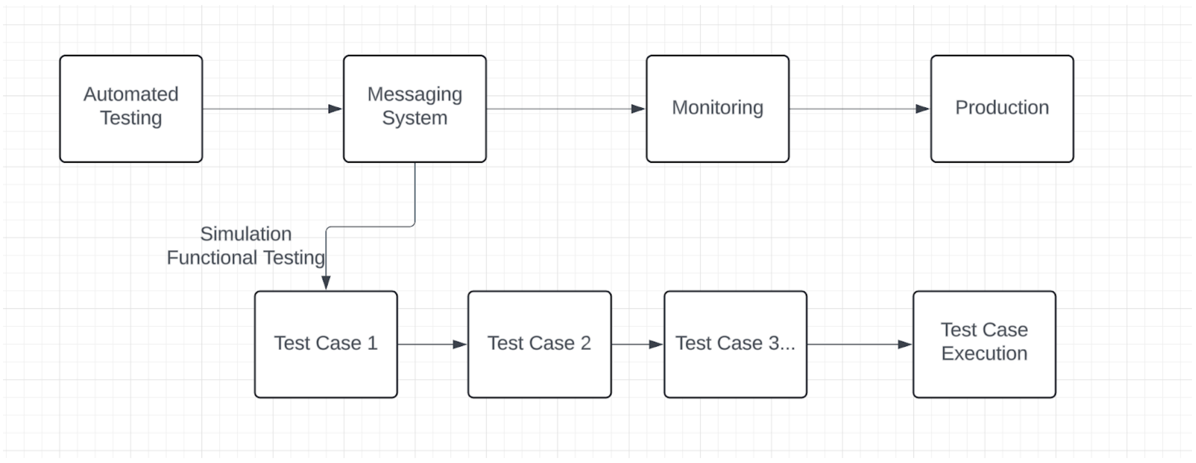
User Story 5: As a user, I want the platform to integrate and display messages from other social media platforms with college presence.

- **Happy Path:**

1. The user logs in to REConnect.
2. REConnect securely connects to the user's various messaging platforms (email, WhatsApp, Instagram, etc.) through authorized integrations.
3. Once connected, REConnect retrieves all unread and recent messages from the various platforms.
4. REConnect presents a unified view of all messages within its interface.
5. Users can filter messages by platform, sender, or keywords for easier management and organization.

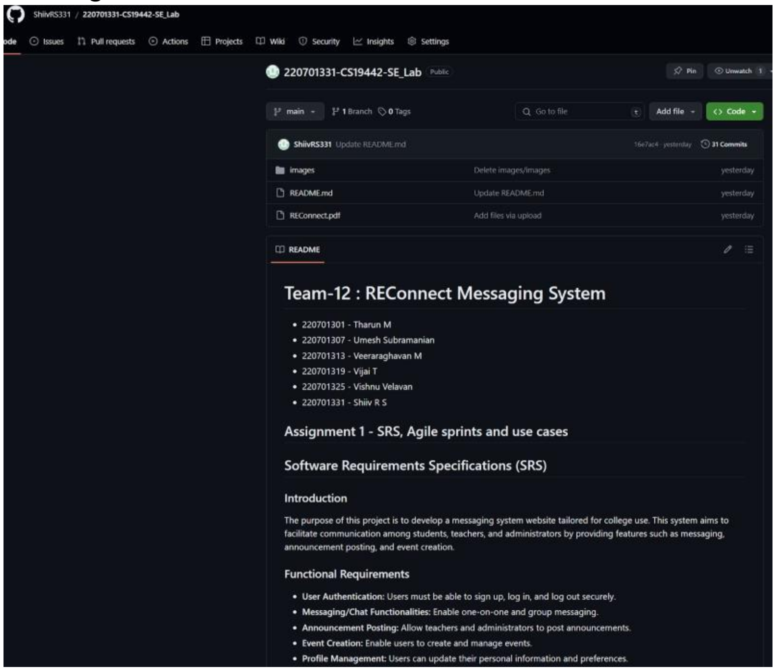
Error Scenarios:

1. **Invalid Login Credentials**
2. **Connection Failure:** In case of temporary network issues or server outages during the connection, the user should receive a user-friendly message explaining the issue.
3. **API Permissions:** If the user's account on a specific platform lacks sufficient permissions to access messages, REConnect should inform the user about this limitation
4. **Security Concerns:** REConnect should prioritize user data security.



Testing Architecture Diagram

5. A view of the github repository showcasing the Project structure, their naming conventions



220701331-CS19442-SE_Lab

ShiivRS331 Update README.md 16e7ac4 · yesterday 31 Commits

images	Delete images/images	yesterday
README.md	Update README.md	yesterday
REConnect.pdf	Add files via upload	yesterday

Team-12 : REConnect Messaging System

- 220701301 - Tharun M
- 220701307 - Umesh Subramanian
- 220701313 - Veeraraghavan M
- 220701319 - Vijai T
- 220701325 - Vishnu Velavan
- 220701331 - Shiiv R S

Assignment 1 - SRS, Agile sprints and use cases

Software Requirements Specifications (SRS)

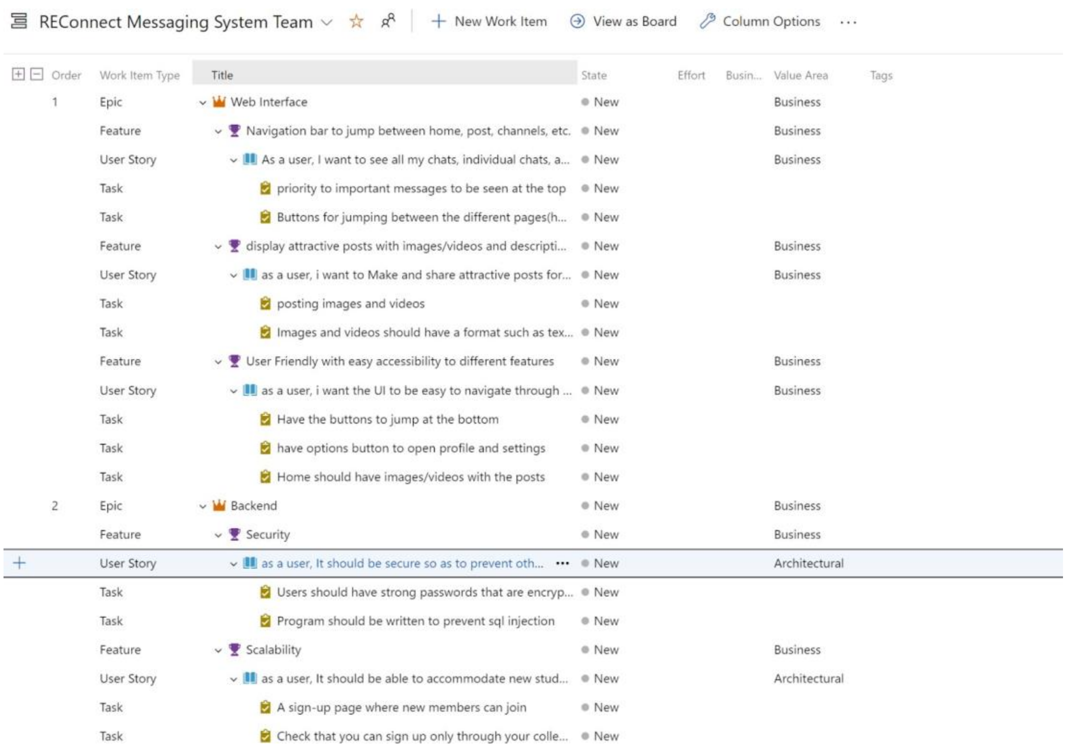
Introduction

The purpose of this project is to develop a messaging system website tailored for college use. This system aims to facilitate communication among students, teachers, and administrators by providing features such as messaging, announcement posting, and event creation.

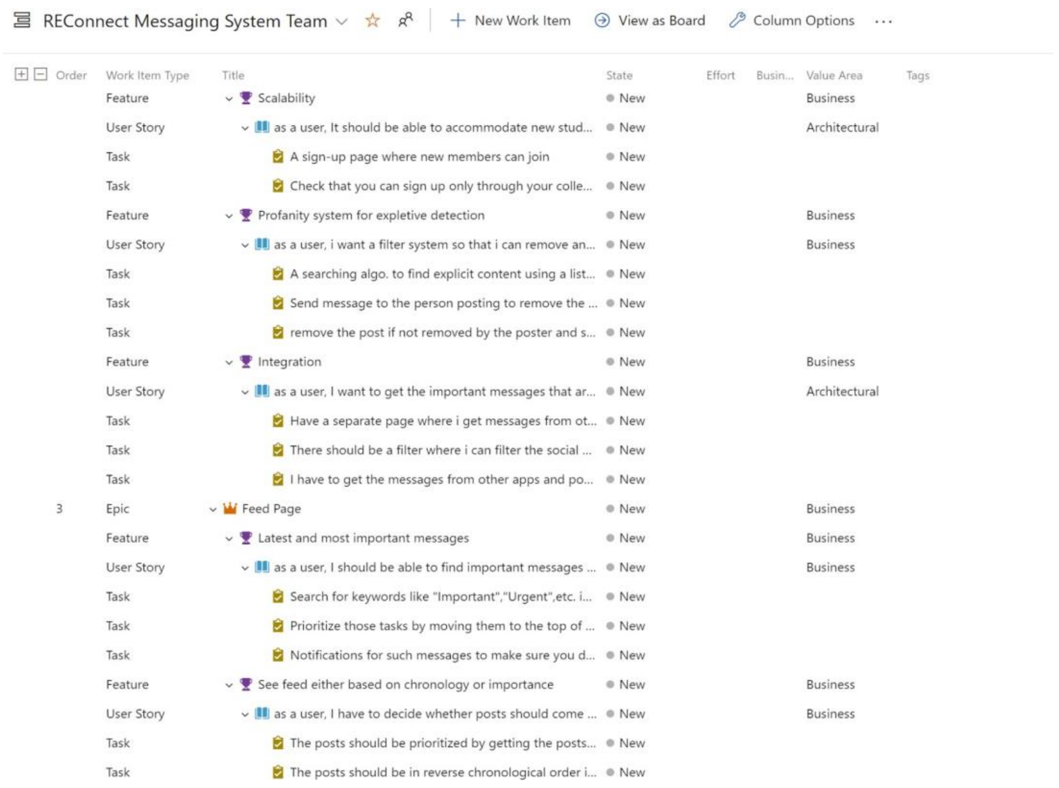
Functional Requirements

- **User Authentication:** Users must be able to sign up, log in, and log out securely.
- **Messaging/Chat Functionalities:** Enable one-on-one and group messaging.
- **Announcement Posting:** Allow teachers and administrators to post announcements.
- **Event Creation:** Enable users to create and manage events.
- **Profile Management:** Users can update their personal information and preferences.

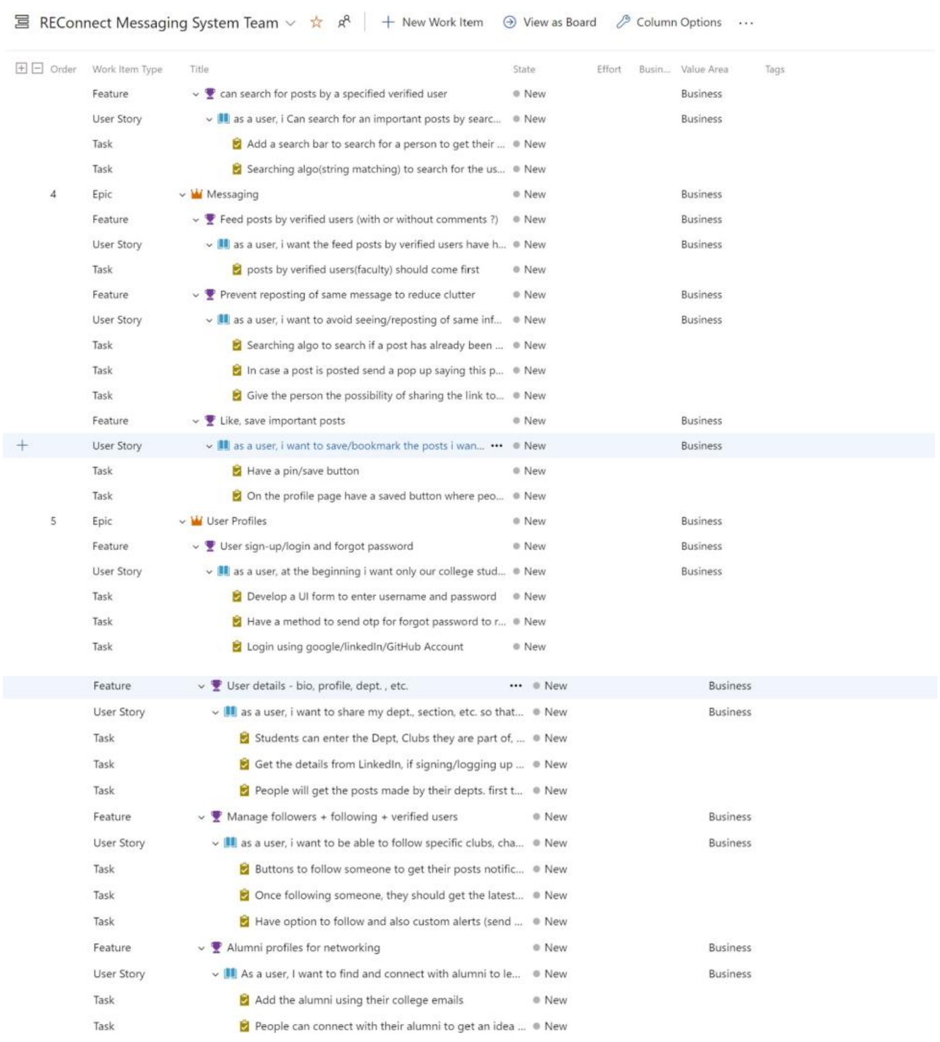
A view of their DevOps Architecture for their respective project and the associated tools used in Azure



Order	Work Item Type	Title	State	Effort	Busin...	Value Area	Tags
1	Epic	Web Interface	New			Business	
	Feature	Navigation bar to jump between home, post, channels, etc.	New			Business	
	User Story	As a user, I want to see all my chats, individual chats, a...	New			Business	
	Task	priority to important messages to be seen at the top	New				
	Task	Buttons for jumping between the different pages(h...	New				
	Feature	display attractive posts with images/videos and descripti...	New			Business	
	User Story	as a user, i want to Make and share attractive posts for...	New			Business	
	Task	posting images and videos	New				
	Task	Images and videos should have a format such as tex...	New				
	Feature	User Friendly with easy accessibility to different features	New			Business	
	User Story	as a user, i want the UI to be easy to navigate through ...	New			Business	
	Task	Have the buttons to jump at the bottom	New				
	Task	have options button to open profile and settings	New				
	Task	Home should have images/videos with the posts	New				
2	Epic	Backend	New			Business	
	Feature	Security	New			Business	
	User Story	as a user, It should be secure so as to prevent oth...	New			Architectural	
	Task	Users should have strong passwords that are encryp...	New				
	Task	Program should be written to prevent sql injection	New				
	Feature	Scalability	New			Business	
	User Story	as a user, It should be able to accommodate new stud...	New			Architectural	
	Task	A sign-up page where new members can join	New				
	Task	Check that you can sign up only through your colle...	New				

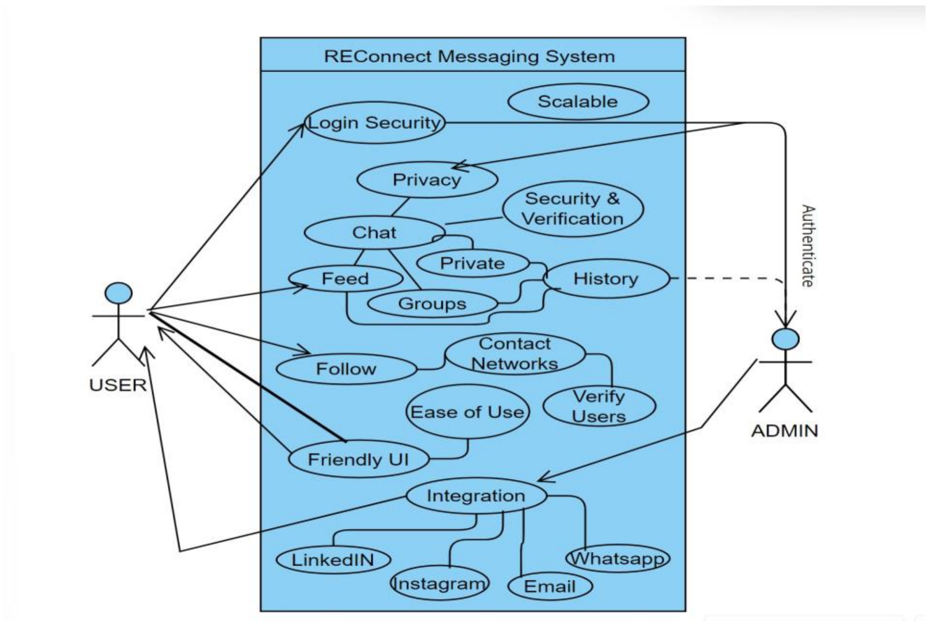


Order	Work Item Type	Title	State	Effort	Busin...	Value Area	Tags
	Feature	Scalability	New			Business	
	User Story	as a user, It should be able to accommodate new stud...	New			Architectural	
	Task	A sign-up page where new members can join	New				
	Task	Check that you can sign up only through your colle...	New				
	Feature	Profanity system for expletive detection	New			Business	
	User Story	as a user, i want a filter system so that i can remove an...	New			Business	
	Task	A searching algo. to find explicit content using a list...	New				
	Task	Send message to the person posting to remove the ...	New				
	Task	remove the post if not removed by the poster and s...	New				
	Feature	Integration	New			Business	
	User Story	as a user, I want to get the important messages that ar...	New			Architectural	
	Task	Have a separate page where i get messages from ot...	New				
	Task	There should be a filter where i can filter the social ...	New				
	Task	I have to get the messages from other apps and po...	New				
3	Epic	Feed Page	New			Business	
	Feature	Latest and most important messages	New			Business	
	User Story	as a user, I should be able to find important messages ...	New			Business	
	Task	Search for keywords like "Important","Urgent",etc. i...	New				
	Task	Prioritize those tasks by moving them to the top of ...	New				
	Task	Notifications for such messages to make sure you d...	New				
	Feature	See feed either based on chronology or importance	New			Business	
	User Story	as a user, I have to decide whether posts should come ...	New			Business	
	Task	The posts should be prioritized by getting the posts...	New				
	Task	The posts should be in reverse chronological order i...	New				



REConnect Messaging System Team | New Work Item | View as Board | Column Options

Order	Work Item Type	Title	State	Effort	Busin...	Value Area	Tags
	Feature	can search for posts by a specified verified user	New			Business	
	User Story	as a user, i Can search for an important posts by searc...	New			Business	
	Task	Add a search bar to search for a person to get their ...	New				
	Task	Searching algo(string matching) to search for the us...	New				
4	Epic	Messaging	New			Business	
	Feature	Feed posts by verified users (with or without comments ?)	New			Business	
	User Story	as a user, i want the feed posts by verified users have h...	New			Business	
	Task	posts by verified users(faculty) should come first	New				
	Feature	Prevent reposting of same message to reduce clutter	New			Business	
	User Story	as a user, i want to avoid seeing/reposting of same inf...	New			Business	
	Task	Searching algo to search if a post has already been ...	New				
	Task	In case a post is posted send a pop up saying this p...	New				
	Task	Give the person the possibility of sharing the link to...	New				
	Feature	Like, save important posts	New			Business	
	User Story	as a user, i want to save/bookmark the posts i wan...	New			Business	
	Task	Have a pin/save button	New				
	Task	On the profile page have a saved button where peo...	New				
5	Epic	User Profiles	New			Business	
	Feature	User sign-up/login and forgot password	New			Business	
	User Story	as a user, at the beginning i want only our college stud...	New			Business	
	Task	Develop a UI form to enter username and password	New				
	Task	Have a method to send otp for forgot password to r...	New				
	Task	Login using google/linkedIn/GitHub Account	New				
	Feature	User details - bio, profile, dept. , etc.	New			Business	
	User Story	as a user, i want to share my dept., section, etc. so that...	New			Business	
	Task	Students can enter the Dept, Clubs they are part of, ...	New				
	Task	Get the details from LinkedIn, if signing/logging up ...	New				
	Task	People will get the posts made by their depts. first t...	New				
	Feature	Manage followers + following + verified users	New			Business	
	User Story	as a user, i want to be able to follow specific clubs, cha...	New			Business	
	Task	Buttons to follow someone to get their posts notific...	New				
	Task	Once following someone, they should get the latest...	New				
	Task	Have option to follow and also custom alerts (send ...	New				
	Feature	Alumni profiles for networking	New			Business	
	User Story	As a user, I want to find and connect with alumni to le...	New			Business	
	Task	Add the alumni using their college emails	New				
	Task	People can connect with their alumni to get an idea ...	New				

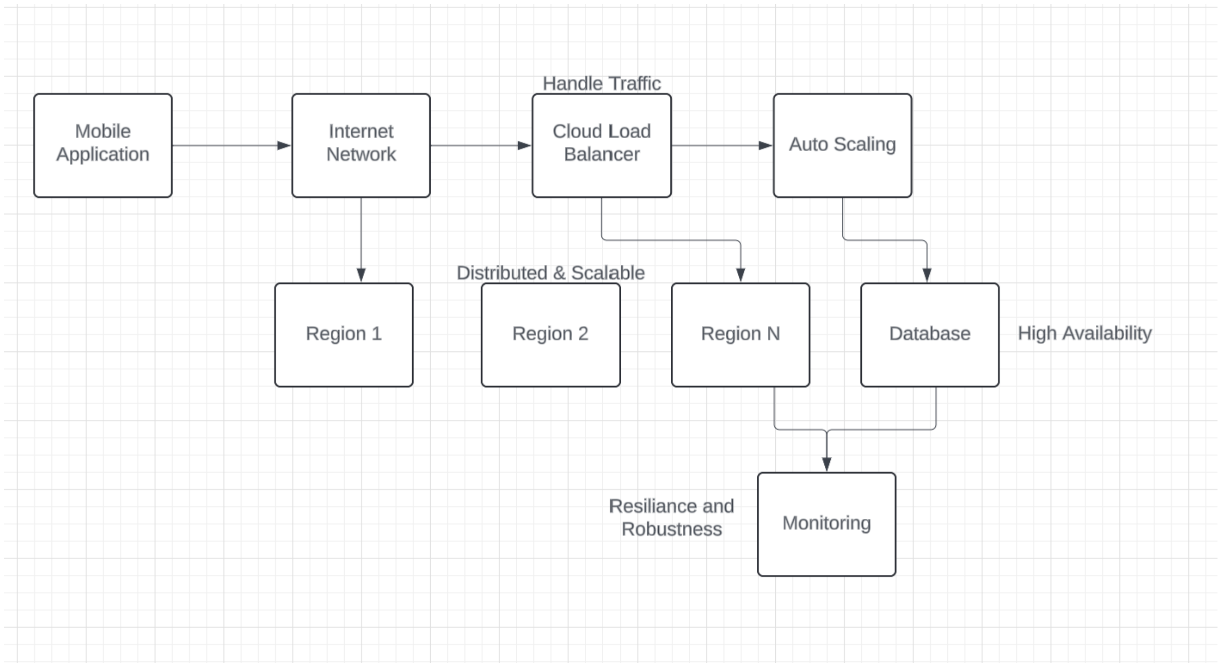


Components:

Source Code Repository (Azure Repos): Stores your project's source code in a secure and version-controlled environment.
CI/CD Pipeline (Azure Pipelines): Automates building, testing, and deploying your application throughout the development process.
Infrastructure as Code (IaaC): Defines and manages your infrastructure resources (e.g., virtual machines, databases) in code for consistent and repeatable deployments. Tools like Azure Resource Manager (ARM) templates can be used.
Artifact Repository (Azure Artifacts): Stores and manages project artifacts (e.g., compiled code, NuGet packages) used during the build and deployment process.
Testing Tools: Integrate various testing tools into your pipeline depending on your project needs. Some examples:
Unit Testing Framework (e.g., JUnit, Jest): Unit tests focus on individual units of code.
API Testing Tool (e.g., Postman, SoapUI): Tests functionality of your application's APIs.
UI Testing Tool (e.g., Selenium, Cypress): Automates interactions with your application's user interface.
Security Testing Tool: Scans your code for potential vulnerabilities.
Azure Monitor: Collects and analyzes application performance data to identify issues proactively.
Alerting and Notification System: Sends notifications to developers and operations teams in case of issues identified during the pipeline execution.
Deployment Environment (e.g., Azure App Service, Azure Kubernetes Service (AKS)): Hosts your application in a scalable and secure cloud environment.
Data Flow:
Procedure:
Developers commit code changes to the Source Code Repository.
The CI/CD Pipeline automatically triggers upon code commits.
The pipeline performs the following actions (depending on your configuration):
Builds the application code using appropriate build tools.
Runs Unit Tests to ensure individual code units function correctly.
(Optional) Runs API and UI Tests to verify application functionality.
(Optional) Scans code for vulnerabilities using security testing tools.
Packages the application and any dependencies into artifacts.
Deploys the application to the chosen deployment environment (e.g., Azure App Service, AKS).
Azure Monitor collects and analyzes application performance data.
(Optional) Alerting and Notification System sends alerts to developers and operations teams based on defined thresholds or error conditions.
Benefits:
Advantages:
Faster Delivery: Automation streamlines software delivery, allowing for quicker deployments.

Improved Quality: Automated testing helps identify and fix bugs early in the development lifecycle.
Increased Reliability: Infrastructure as Code and automated deployments ensure consistent and reliable infrastructure provisioning.
Reduced Costs: Automation minimizes manual errors and rework, leading to potential cost savings.
Scalability: Cloud-based deployments allow for easy scaling of resources based on application demands.

Deployment Architecture of the application



- **User Devices (Web/Mobile App):**Users interact with the application through their devices.
- **Internet:** Public network connection for users to access the application.
- **Cloud Load Balancer:** Distributes incoming traffic across multiple instances of the messaging system in different regions for optimal performance and scalability.
- **Auto Scaling:** These groups manage instances of your messaging system across different Availability Zones (AZs) within a cloud region. They automatically scale the number of instances up or down based on traffic demands.
- **Messaging System:** This represents the core application logic deployed across multiple instances within ASGs for redundancy and scalability.
- **Database:** A highly available database service deployed across multiple AZs within regions ensures data persistence and accessibility.
- **Monitoring:** Continuously monitors the health and performance of the messaging system and database, providing alerts and logs for troubleshooting and scaling decisions.

Benefits of this Architecture:

- High Availability
- Scalability
- Resilience
- Performance
- Observability