DELHI TECHNICAL CAMPUS
GREATER NOIDA
Affiliated to GGSIPU and approved by AICTE & COA

PRACTICAL FILE

SESSION: 2023-24

Software Engineering Lab
(CIC 357)

III Year, V Sem

Submitted to:

Name: Dr Archana Sharma
Designation: Professor

Submitted by:

Name: Hitesh Bardia
Enrollment No. 07418002721

Department of Computer Science and Engineering
Delhi Technical Campus, Greater Noida

INDEX

S.NO.	PROGRAM NAME	DATE OF EXPERIMENT	DATE OF SUBMISSION	SIGN.
1	Write down the problem statement for a Real-time chat application.			
2	Do requirement analysis and develop Software Requirement Specification Sheet (SRS) for Real-time chat application.			
3	To perform the function-oriented diagram: Data Flow Diagram (DFD) and Structured Chart.			
4	Draw the entity relationship diagram for the Real-time chat application.			
5	To perform the user‘s view analysis for the Real-time chat application: Use case diagram.			
6	To draw the structural view diagram for the Real-time chat application: Class diagram, object diagram.			
7	To draw the behavioural view diagram: State-chart diagram, Activity diagram.			
8	To perform the behavioural view diagram for the Real-time chat application: Sequence diagram, Collaboration diagram.			
9	To perform the implementation view diagram: Component diagram for the Real-time chat application.			
10	Perform Estimation of effort using FP Estimation for Real-time chat application.			
11	To prepare time line chart/Gantt Chart/PERT Chart for Real-time chat application.			

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 1

AIM: Write down the problem statement for a Real-time chat application.

REAL-TIME CHAT APPLICATION

➢ Problem Statement:

The absence of a robust and user-friendly Real-time Chat Application limits efficient and effective communication, hindering swift information exchange, teamwork, and real-time decision-making processes among individuals and groups

In real-time chat applications, users often experience difficulty finding relevant conversations and messages amidst a large volume of chats. This can lead to frustration, inefficiency, and a lack of engagement. To address this issue, a system is needed that can effectively identify and present relevant content to users based on their interests, context, and preferences.

Contextualizing the Problem:

The existing landscape of real-time chat applications lacks efficiency and user-friendliness, impeding smooth information exchange, collaborative efforts, and timely decision-making processes for both individuals and groups.

Describing the Issue:

Users encounter difficulties in navigating through a vast volume of chats, leading to inefficiencies, frustration, and a lack of engagement. Relevant conversations and messages become challenging to identify within the cluttered interface, hampering the overall communication experience.

Relevance of the Problem:

This issue significantly impacts productivity, teamwork, and user satisfaction. Without an effective solution, users remain burdened by the overwhelming volume of messages, hindering their ability to locate and engage with pertinent content swiftly

Name: Hitesh Bardia **Enrolment No.** 07418002721

.

Suggested System:

System of relevance for a real-time chat application should incorporate the following features:

1. **Content-based relevance:** Analyse the content of messages to identify keywords, topics, and entities that align with user interests.

2. **Context-based relevance:** Consider the context of conversations, including participants, timestamps, and chat channels, to determine the relevance of messages.

3. **User-based relevance:** Personalize the relevance algorithm based on user preferences, such as frequently used keywords, saved messages, and followed users.

4. **Real-time relevance:** Continuously evaluate the relevance of messages as new information is added to the chat application.

5. **Adaptive relevance:** Continuously refine the relevance algorithm based on user feedback and engagement metrics.

By incorporating these features, a suggested system of relevance can significantly improve the user experience in real-time chat applications, enabling users to find and engage with the most relevant content effortlessly.

Objectives of the Research:

The primary research objectives entail:

- Designing a real-time chat application that addresses the challenge of identifying and presenting relevant content to users.
- Incorporating features such as content-based, context-based, and user-based relevance algorithms to streamline message identification.
- Implementing real-time and adaptive relevance mechanisms to continuously enhance user experience and engagement.

Name: Hitesh Bardia **Enrolment No.** 07418002721

➢ The 5 'W's:

- **Who:** Individuals and groups relying on real-time chat applications for communication and collaboration.
- **What:** The issue revolves around the inefficiency in identifying relevant content within the chat interface.
- **When:** The issue occurs whenever users engage in communication and information exchange through the application.
- **Where:** The issue affects the usability and effectiveness of chat applications across various devices and platforms.
- **Why:** It's crucial to solve this problem to enhance productivity, streamline communication, and boost user engagement within real-time chat applications.

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 2

AIM: Do requirement analysis and develop Software Requirement Specification Sheet (SRS) for Real-time chat application.

Software Requirement Specification Sheet (SRS)

Project: Real-time Chat Application

1. INTRODUCTION

1.1 PURPOSE OF THE DOCUMENT

The primary objective of this document is to establish a clear and comprehensive set of guidelines and specifications for the development of a Real-time Chat Application. It aims to provide a detailed outline of the system's requirements, functionalities, and constraints to ensure a common understanding among all stakeholders involved in the project.

1.2 SCOPE OF THE DOCUMENT

This document delineates the scope of the Real-time Chat Application, encompassing its functionalities, user classes, system interfaces, and operational constraints. It serves as a reference point for the development team, aiding in the creation of a robust and user-friendly communication platform.

1.3 ACRONYMS

- **SRS:** Software Requirement Specification
- **FRS:** Functional Requirement Specification

Name: Hitesh Bardia **Enrolment No.** 07418002721

1.4 INTENDED AUDIENCE AND READING SUGGESTIONS

The intended audience for this document includes the development team, stakeholders, project managers, and quality assurance teams involved in the creation and validation of the Real-time Chat Application.

- **Development Team:** To guide the development process.
- **Stakeholders:** To understand the project's scope and requirements.
- **Project Managers:** To ensure alignment with project objectives.
- **Quality Assurance Teams:** To validate system compliance.
- **Documentation Writers:** Involved in creating user manuals, technical documentation, and guides.
- **Marketing Staff:** Gaining an understanding of the application's features and capabilities for promotional purposes.
- **Users (End-Users):** Understanding the functionalities and capabilities of the chat application.

1.5 DOCUMENT OVERVIEW

This document is structured to provide a comprehensive understanding of the Real-time Chat Application. It is divided into sections that cover various aspects of the system, starting with an overall description, external interfaces, functional and non-functional requirements, use cases, and other essential details crucial for successful system development and deployment.

Name: Hitesh Bardia **Enrolment No.** 07418002721

2. OVERALL DESCRIPTION

2.1 PRODUCT PERSPECTIVE

The Real-time Chat Application is designed as a standalone, robust communication platform that allows users to engage in instantaneous messaging, multimedia sharing, and group discussions. It operates as an independent system, leveraging modern technologies for real-time communication while maintaining compatibility across various devices and platforms.

2.2 PRODUCT FUNCTIONS

The application serves as a multifunctional platform with the following core features:

1. **Registration and Login:** Allows users to create accounts and authenticate securely.
2. **Profile Page:** Enables users to manage credentials and avatars.
3. **User Classes:** Supports various user roles, including Students, Professors, Super Admin, and Moderators.
4. **Chat Rooms:** Provides dedicated spaces for group discussions and interactions.
5. **Online Status Feature:** Indicates users' online presence for real-time communication.
6. **Search Functionality:** Facilitates quick and efficient search within chats and messages.
7. **Emoji Pickers and Reaction System:** Allows users to express reactions using emoji during conversations.
8. **Light & Dark Mode:** Offers interface customization for user preference.

Name: Hitesh Bardia **Enrolment No.** 07418002721

2.3 USER CLASSES AND CHARACTERISTICS

The Real-time Chat Application caters to distinct user classes within the college ecosystem, each with specific roles and characteristics:

Students:

- Engage in academic discussions related to courses, projects, and extracurricular activities.
- Access dedicated chat rooms for specific classes, interest groups, or clubs.
- Collaborate on group assignments, share resources, and seek academic support.

Professors:

- Facilitate teaching, mentorship, and academic discussions.
- Create and moderate chat rooms for individual courses or study groups.
- Provide guidance, share resources, and interact with students outside of class hours.

Super Admin:

- Oversee administrative functions within the application, ensuring smooth operations.
- Manage user roles, permissions, and system configurations.
- Address technical issues and provide support as needed.

Moderators:

- Assist in maintaining decorum and relevance within specific chat rooms.
- Monitor discussions, ensuring adherence to college guidelines and academic integrity.
- Support users, resolve conflicts, and escalate issues when necessary.

The Real-time Chat Application is tailored to enhance communication and collaboration within the college community, fostering an environment conducive to academic engagement, resource sharing, and cohesive interaction among students, professors, and administrative staff.

Name: Hitesh Bardia **Enrolment No.** 07418002721

2.4 OPERATING ENVIRONMENT

The application is built using modern web technologies such as React, Tailwind, and utilizes Firebase as a Backend-as-a-Service (BaaS). The backend is implemented using Node/Express and MongoDB, with specific data models for chats, chat lines, messages, seen, reactions, and file relevance.

2.5 DESIGN AND IMPLEMENTATION CONSTRAINTS

The design focuses on a responsive and intuitive user interface to ensure ease of use across various devices and screen sizes. Additionally, the system's scalability and performance are key considerations in its design and implementation.

2.6 USER DOCUMENTATION

Comprehensive user documentation will be provided to guide users through the application's functionalities and features, ensuring an optimal user experience.

2.7 ASSUMPTIONS AND DEPENDENCIES

The successful deployment and operation of the Real-time Chat Application are dependent on stable internet connectivity and adherence to modern web standards across supported browsers and devices.

Name: Hitesh Bardia **Enrolment No.** 07418002721

3. EXTERNAL INTERFACE REQUIREMENTS

3.1 USER INTERFACES

The Real-time Chat Application is designed as a standalone, robust communication platform that allows users to engage in instantaneous messaging, multimedia sharing, and group discussions. It operates as an independent system, leveraging modern technologies for real-time communication while maintaining compatibility across various devices and platforms.

3.2 HARDWARE INTERFACES

The Real-time Chat Application does not necessitate specific hardware requirements beyond standard computing devices, ensuring accessibility across a wide range of devices, including computers, laptops, tablets, and smartphones.

3.3 SOFTWARE INTERFACES

The application integrates the following software components and interfaces:

- **Frontend Technologies:** Utilizes React and Tailwind for the web and mobile interfaces.
- **Backend Framework:** Implemented with Node.js/Express.js for server-side operations.
- **Database Management:** Relies on MongoDB for data storage, utilizing two distinct models: chatRoom and chatMessage.
- **Real-time Communication:** Employs Socket.IO for real-time data exchange and communication.

3.4 COMMUNICATION INTERFACES

The Real-time Chat Application requires stable internet connectivity to facilitate real-time communication between users. It operates over standard HTTP/HTTPS protocols for secure data transmission and Web-Socket connections for real-time messaging functionality.

Name: Hitesh Bardia **Enrolment No.** 07418002721

4. FUNCTIONAL REQUIREMENT SPECIFICATIONS (FRS)

4.1 SYSTEM FEATURES

The Real-time Chat Application encompasses the following key features:

4.1.1 Registration and Login

- **Description:** Users can create accounts and log in securely to access the application.
- **Requirements:**
  - User registration with valid credentials.
  - Secure login mechanism with authentication protocols.

4.1.2 Profile Page

- **Description:** Users can manage their credentials and avatars within their profiles.
- **Requirements:**
  - Profile editing options for credentials.
  - Avatar uploading and customization features.

4.1.3 User Classes

- **Description:** The application supports various user roles with distinct privileges.
- **Requirements:**
  - Role-based access control for Students, Professors, Super Admin, and Moderators.

4.1.4 Chat Rooms

- **Description:** Dedicated spaces for group discussions and interactions.
- **Requirements:**
  - Creation and management of chat rooms.
  - Access control and membership management within rooms.

4.1.5 Online Status Feature

- **Description:** Indicates users' online presence for real-time communication.
- **Requirements:**
  - Online/offline status indicators for users.

4.1.6 Search Functionality

- **Description:** Facilitates quick and efficient search within chats and messages.
- **Requirements:**
  - Search feature across chat rooms and messages.

4.1.7 Emoji Pickers and Reaction System

- **Description:** Allows users to express reactions using emoji during conversations.
- **Requirements:**
  - Emoji picker for message reactions.
  - Reaction system implementation within chats.

4.1.8 Light & Dark Mode

- **Description:** Offers interface customization for light and dark mode preferences.
- **Requirements:**
  - Toggle between light and dark modes for user interface.

4.2 FUNCTIONAL REQUIREMENTS

4.2.1 Front end (Storefront) Requirements

- Detailed functional requirements for the user interface, ensuring usability and functionality across platforms.

4.2.2 Back end (Administrative Tools) Requirements

- Specific backend functional requirements, including server-side operations, database interactions, and real-time communication protocols.

4.2 USE CASES

4.3.1 Front end (Storefront)

- Illustrative use cases detailing user interactions and functionalities from a frontend perspective.

4.3.2 Back end (Administrative Tools)

- Use cases outlining backend functionalities and operations for administrative and system-level tasks.

Name: Hitesh Bardia **Enrolment No.** 07418002721

5. NON-FUNCTIONAL REQUIREMENT

5.1 USABILITY REQUIREMENTS

The Real-time Chat Application must ensure usability in the following aspects:

- **Intuitiveness:** The user interface should be intuitive and easy to navigate.
- **Accessibility:** Accessibility standards should be met for users with disabilities.
- **Consistency:** Ensure consistent user experience across different devices and platforms.
- **Learning Curve:** Users should be able to quickly adapt to the application's functionalities.

5.2 PERFORMANCE REQUIREMENTS

The application should meet the following performance criteria:

- **Responsiveness:** Messages and chat room loading should be near-instantaneous.
- **Scalability:** The system should handle a growing user base without performance degradation.
- **Resource Utilization:** Optimize resource usage to ensure efficient operation across devices.
- **Minimum Downtime:** Ensure high availability with minimal system downtimes.

5.3 COMPATIBILITY REQUIREMENTS

The Real-time Chat Application should exhibit compatibility with various platforms and environments:

- **Cross-Platform Compatibility:** Ensure seamless operation across different operating systems and devices.

- **Browser Compatibility:** Support major web browsers and versions without functional discrepancies.

- **Device Adaptability:** Adapt the application interface to different screen sizes and resolutions.

5.4 USABILITY REQUIREMENTS

The application must adhere to robust security measures to safeguard user data and privacy:

- **Data Encryption:** Implement end-to-end encryption for secure message transmission.
- **Authentication and Authorization:** Employ secure authentication mechanisms and role-based access control.
- **Secure Storage:** Ensure encryption and secure storage of user data and credentials.
- **Protection against Threats:** Implement measures to mitigate risks from cyber threats like XSS, CSRF, etc.

5.5 RELIABILITY REQUIREMENTS

The application should exhibit reliability in terms of:

- **Stability:** Ensure consistent and stable operation without frequent crashes or errors.
- **Fault Tolerance:** Ability to recover gracefully from unexpected failures or system interruptions.
- **Data Integrity:** Maintain data integrity and consistency across the platform.

Name: Hitesh Bardia **Enrolment No.** 07418002721

6. OTHER REQUIREMENT

6.1 LEGAL AND COMPLIANCE

- **Data Protection Regulations:** Ensure compliance with relevant data protection laws and regulations.
- **Intellectual Property:** Respect intellectual property rights regarding content shared within the application.
- **Terms of Service:** Specify user guidelines and terms of service within the application.

6.2 DOCUMENTATION AND TRAINING

- **User Manuals:** Provide comprehensive user documentation explaining application features and functionalities.
- **Training Materials:** Develop training resources for new users to familiarize themselves with the application.

6.3 MAINTENANCE AND SUPPORT

- **Maintenance Schedule:** Outline regular maintenance plans and update schedules for the application.
- **Technical Support:** Offer user support channels and response times for technical issues.

6.4 PERFORMANCE METRICS AND MONITORING

- **Monitoring Tools:** Implement tools to monitor application performance and user interactions.
- **Performance Metrics:** Define key performance indicators (KPIs) to assess application performance.

Name: Hitesh Bardia **Enrolment No.** 07418002721

6.5 BACKUP AND RECOVERY

- **Data Backup:** Ensure regular backup of user data to prevent loss in case of system failures.
- **Recovery Protocols:** Establish recovery procedures to restore system functionality in case of data loss.

6.6 USER FEEDBACK AND IMPROVEMENT

- **Feedback Mechanism:** Incorporate a user feedback system to gather suggestions and improvements.
- **Iterative Development:** Plan for iterative updates and improvements based on user feedback.

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 3

AIM: To perform the function-oriented diagram: Data Flow Diagram (DFD) and Structured chart.

REQUIREMENTS:

Software Requirements:

A tool for drawing DFDs (e.g., draw.io, Microsoft Visio).
Understanding of DFD symbols and notation.
A structured programming environment for creating a Structured Chart.

System Requirements:

Processes: Order Processing, Billing, Inventory Management.
Data Stores: Customer Data, Menu Items, Order Details.
External Entities: Customer, Kitchen Staff.

Data Flow Diagram (DFD):

Level 0 DFD (Context Diagram)

The Level 0 DFD gives an overview of the system, showing interactions between the system and external entities such as users and databases.

- **Processes:**
  - Main system (Chat Application)
- **Entities:**
  - Users
  - Chat Rooms
  - Databases

Name: Hitesh Bardia **Enrolment No.** 07418002721

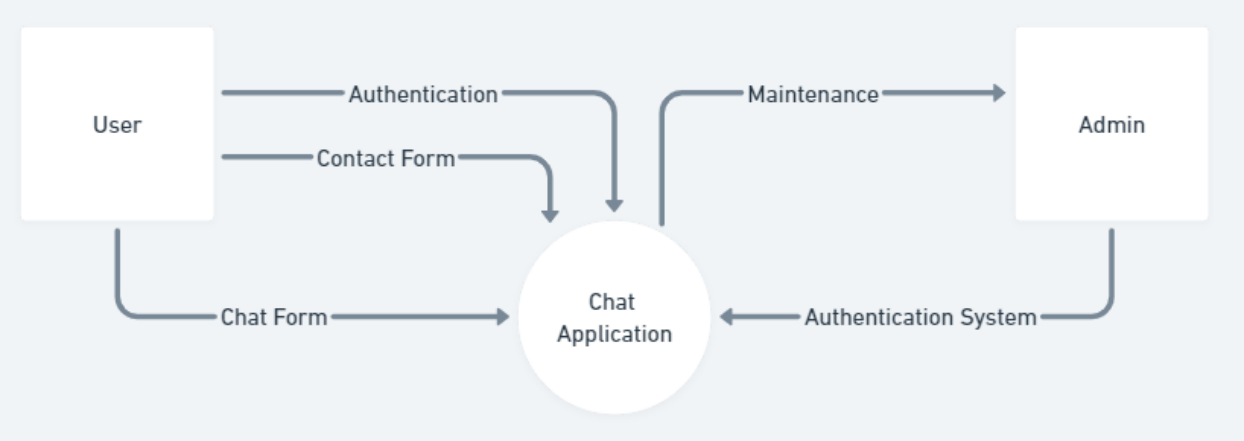


Figure: *Real-time Chat Application level 0 Data Flow Diagram*

Name: Hitesh Bardia **Enrolment No.** 07418002721

Level 1 DFD (Detailed Diagram)

The Level 1 DFD delves deeper into the system, breaking down processes and data flow within the system.

- **Processes:**
  - User Login/Logout
  - Profile Management
  - Chat Room Creation
  - Message Sending/Receiving
  - Search Functionality
  - Emoji Reactions

- **Data Stores:**
  - User Profiles
  - Chat Logs

- **Data Flows:**
  - User login details flow to authentication process
  - Messages flow from sender to receiver through the chat system

Structured Chart

A Structured Chart provides a hierarchical breakdown of system modules or functions.

- Main Module: Real-time Chat Application
  - Submodules:
    - User Authentication
    - Profile Management
    - Chat Room Management
    - Message Handling
    - Search Functionality
    - Reaction System

Each submodule can further break down into smaller functions or processes.

Name: Hitesh Bardia **Enrolment No.** 07418002721

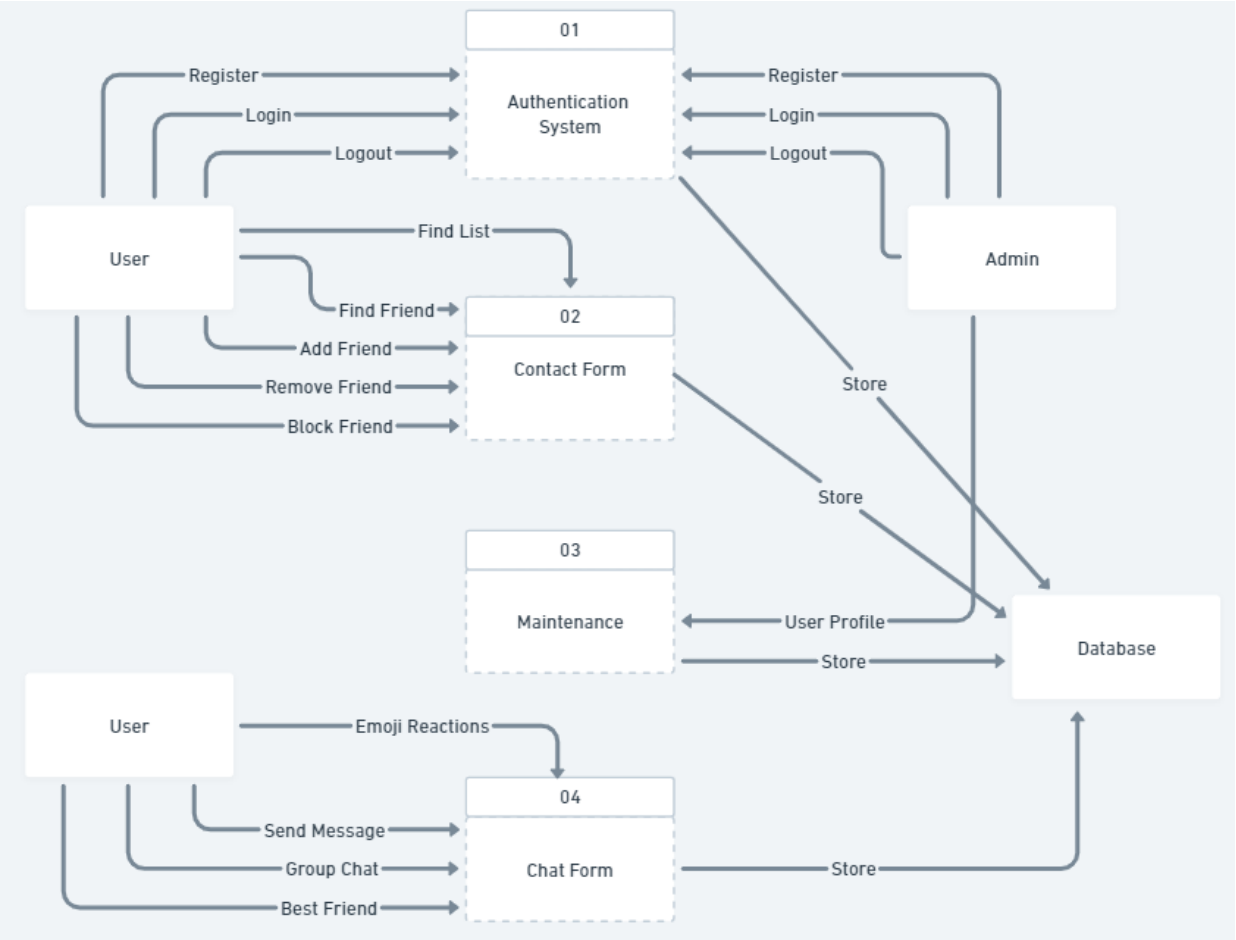


Figure: *Real-time Chat Application level 1 Data Flow Diagram*

Conclusion:

Level 0 and Level 1 Data Flow Diagram (DFD) for Real-time Chat Application have been drawn Successfully.

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 4

AIM: Draw the entity relationship diagram for the Real-time chat application.

REQUIREMENTS:

Software Requirements:

A tool for drawing ER diagrams (e.g., draw.io, Lucidchart).

Basic understanding of database concepts.

System Requirements:

Entities: Customer, Order, Item, Payment.

Relationships: Customer places Order, Order contains Item, Order requires Payment.

THEORY:

Entity Relationship Diagrams are a major data modelling tool and will help organize the data in your project into entities and define the relationships between the entities. This process has proved to enable the analyst to produce a good database structure so that the data can be stored and retrieved in a most efficient manner.

Name: Hitesh Bardia **Enrolment No.** 07418002721

ER Diagram for Real-time Chat Application:

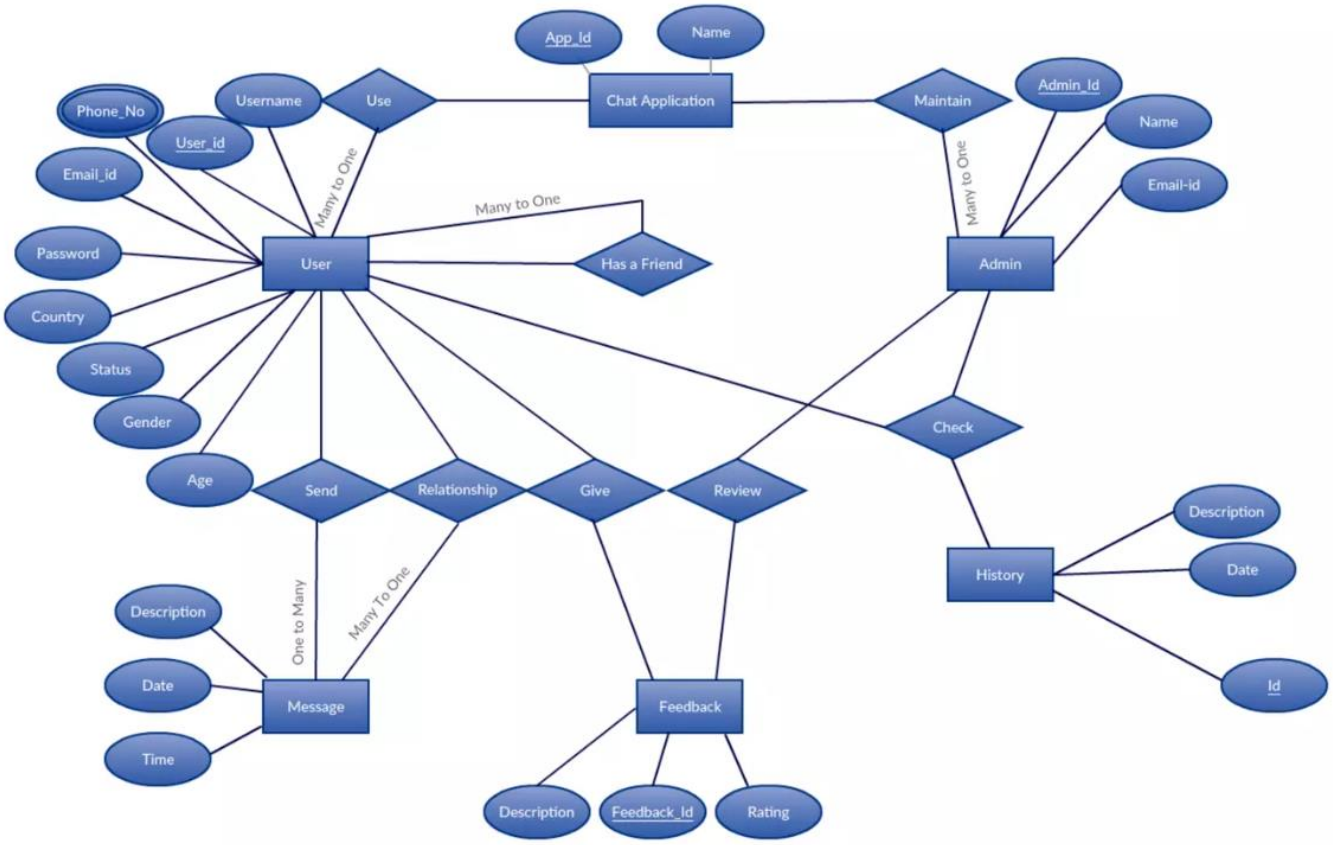


Figure: *Real-time Chat Application Entity Relationship (ER) Diagram*

Conclusion:

Structure Chart for Real-time Chat Application have been Drawn Successfully

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 5

AIM: To perform the user's view analysis for the Real-time chat application: Use case diagram.

REQUIREMENTS:

Software Requirements:

- Diagramming tool (e.g., Microsoft Visio, Lucidchart, or any UML modeling tool).
- Access to the specifications and requirements of the Fast Food Billing System.

Knowledge Requirements:

- Understanding of the fast food billing system requirements.
- Knowledge of UML (Unified Modeling Language) and Use Case Diagrams.

THEORY:

A Use Case Diagram is a visual representation of the functional requirements of a system from the user's perspective. It shows the different ways users can interact with the system and the various use cases that the system must support.

Name: Hitesh Bardia **Enrolment No.** 07418002721

Use Case Diagram:

1. Use Case Table

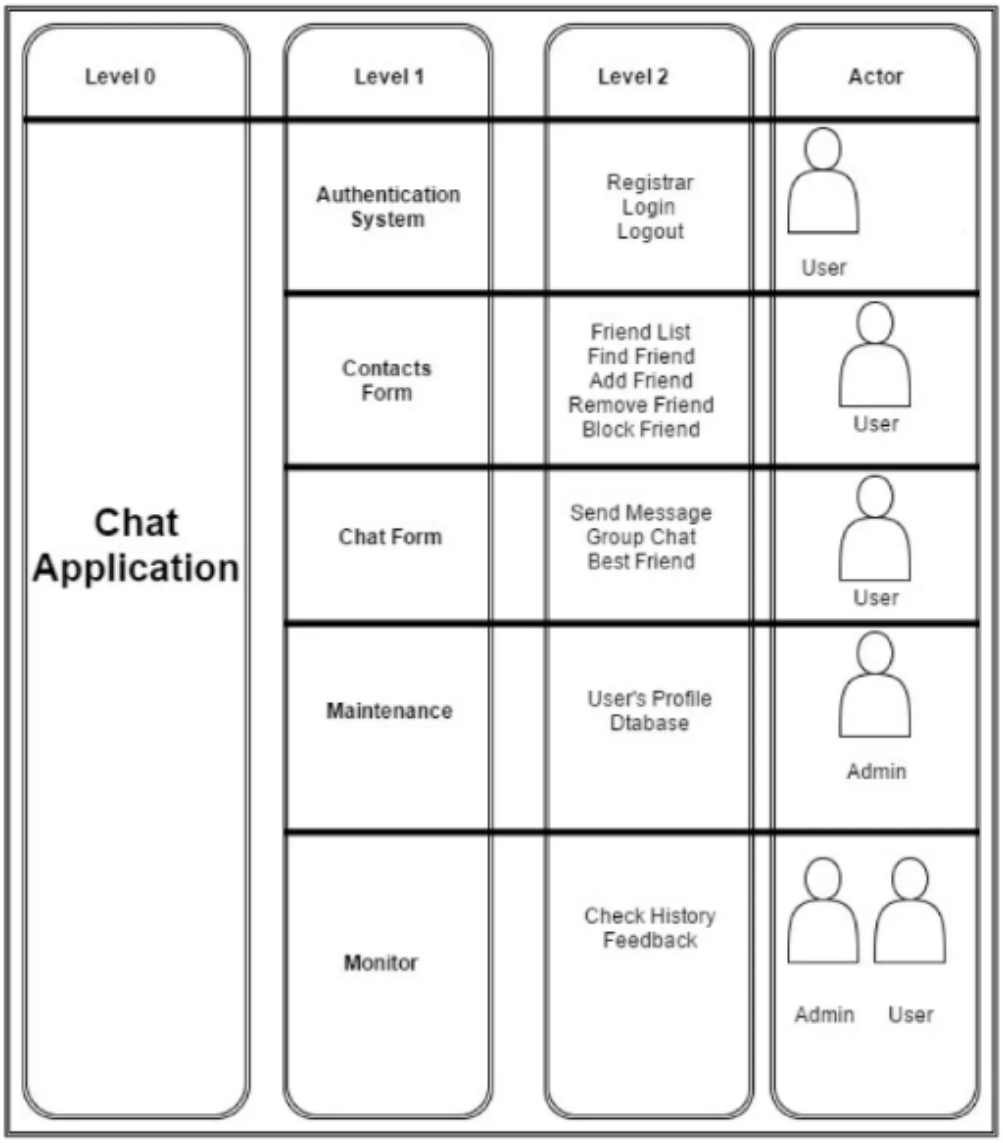
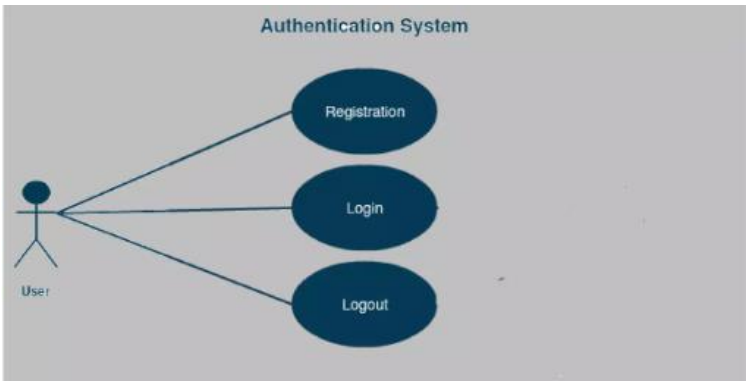


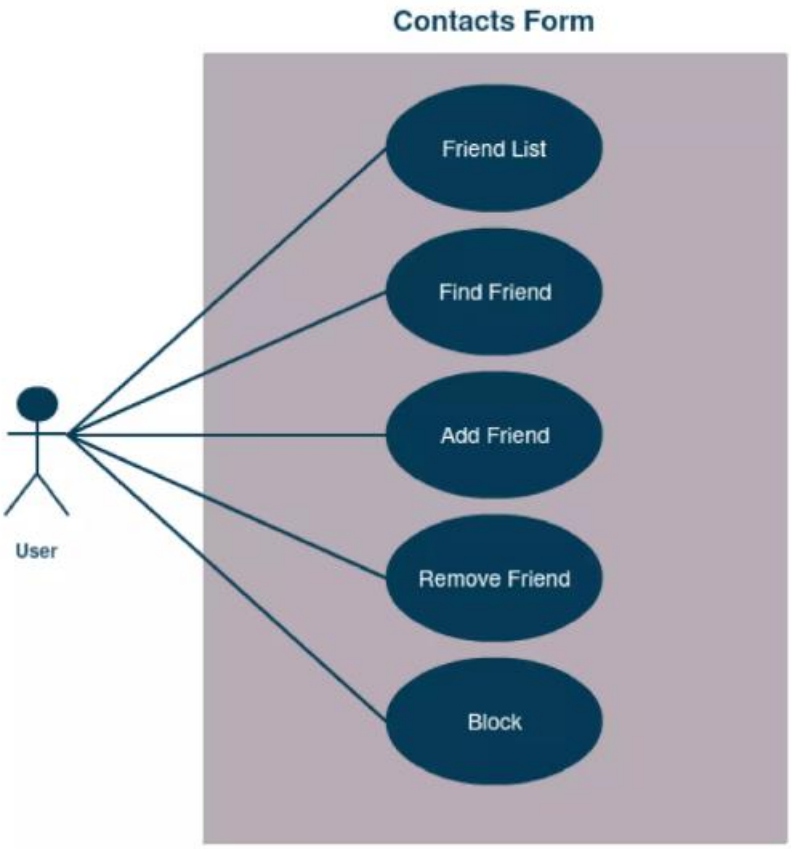
Figure: *Use Case table for Real-time Chat Application*

Name: Hitesh Bardia **Enrolment No.** 07418002721

4. Authentication System

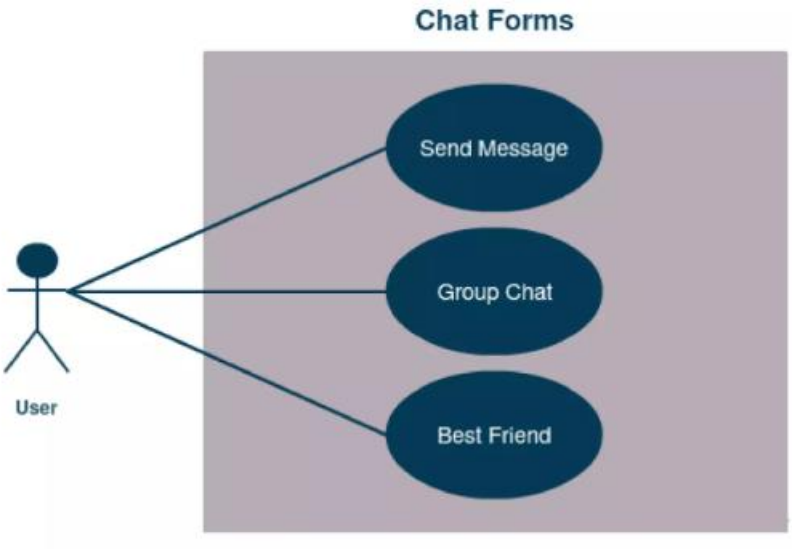


5. Contacts Form

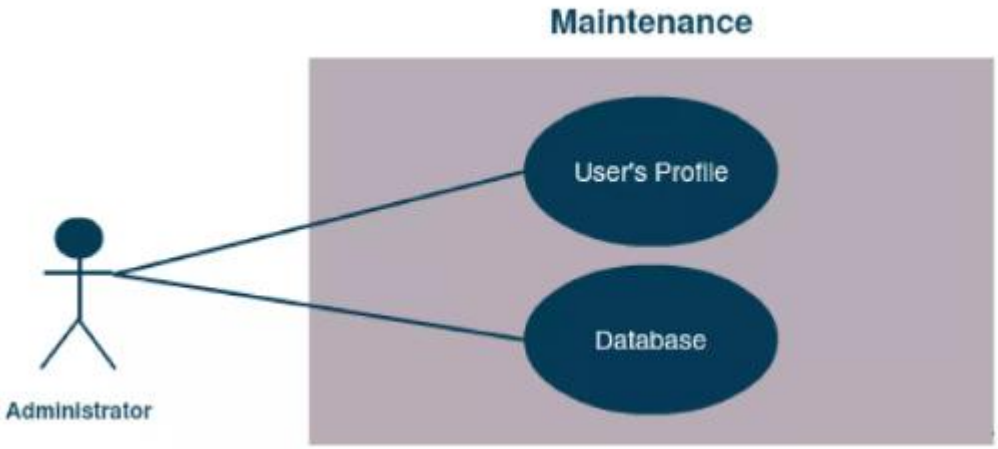


Name: Hitesh Bardia **Enrolment No.** 07418002721

2. Chat Form



3. Maintenance



Name: Hitesh Bardia **Enrolment No.** 07418002721

6. Monitor

Conclusion:

Use case diagram for Real-time Chat Application has been drawn successfully.

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 6

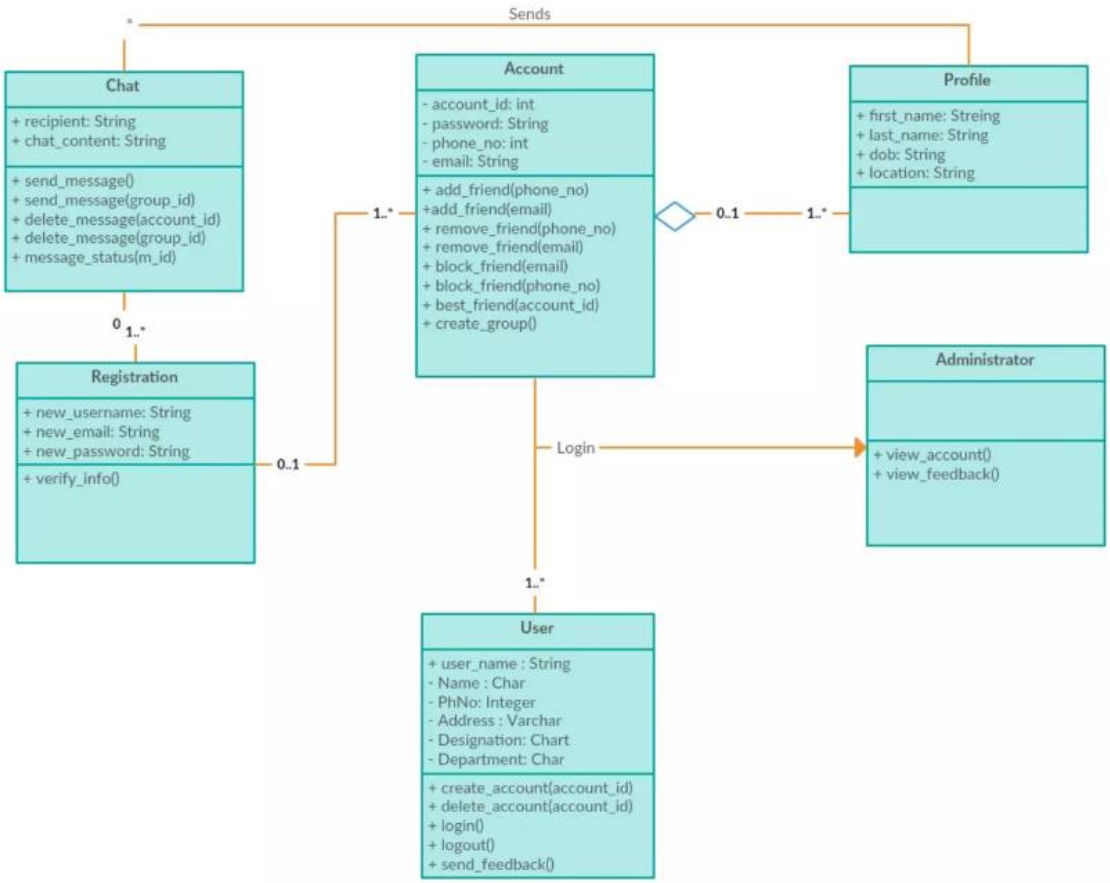
AIM: To draw the structural view diagram for the Real-time chat application: Class diagram, object diagram.

REQUIREMENTS:

Software Requirements – Microsoft Word
Hardware Requirements – Computer, keyboard, mouse, CPU.

Class Diagram:

A class diagram is a type of static structure diagram in the Unified Modelling Language (UML) that represents the structure and organization of a system or software application in terms of classes, their attributes, methods, and the relationships among them. Class diagrams are a fundamental part of object-oriented modelling and provide a visual representation of the code structure.



Name: Hitesh Bardia **Enrolment No.** 07418002721

Figure: *Class Diagram for Real-time Chat Application*

Object Diagram:

An object diagram in the Unified Modelling Language (UML) is a diagram that provides a snapshot of the instances of classes in a system at a particular point in time. It shows the objects and their relationships based on the classes defined in a class diagram. Object diagrams are particularly useful for illustrating real-world scenarios or specific system states.

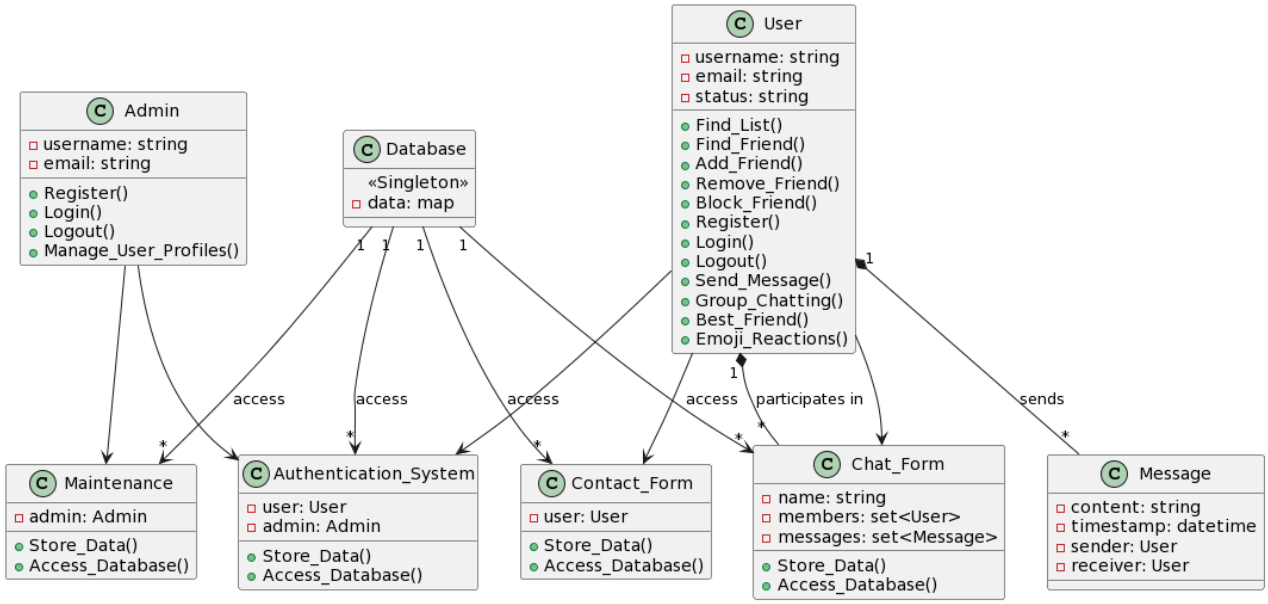


Figure: *Object Diagram for Real-time Chat Application*

Conclusion:

Class diagram and object Diagram for Real-time Chat Application has been done successfully.

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 7

AIM: To draw the behavioural view diagram: State-chart diagram, Activity diagram.

REQUIREMENTS:

Software Requirements – Microsoft Word
Hardware Requirements – Computer, keyboard, mouse, CPU.

State-Chart Diagram:

A Statechart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events. Statechart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of a Statechart diagram is to model the lifetime of an object from creation to termination.

Name: Hitesh Bardia **Enrolment No.** 07418002721

State-Chart Diagram:

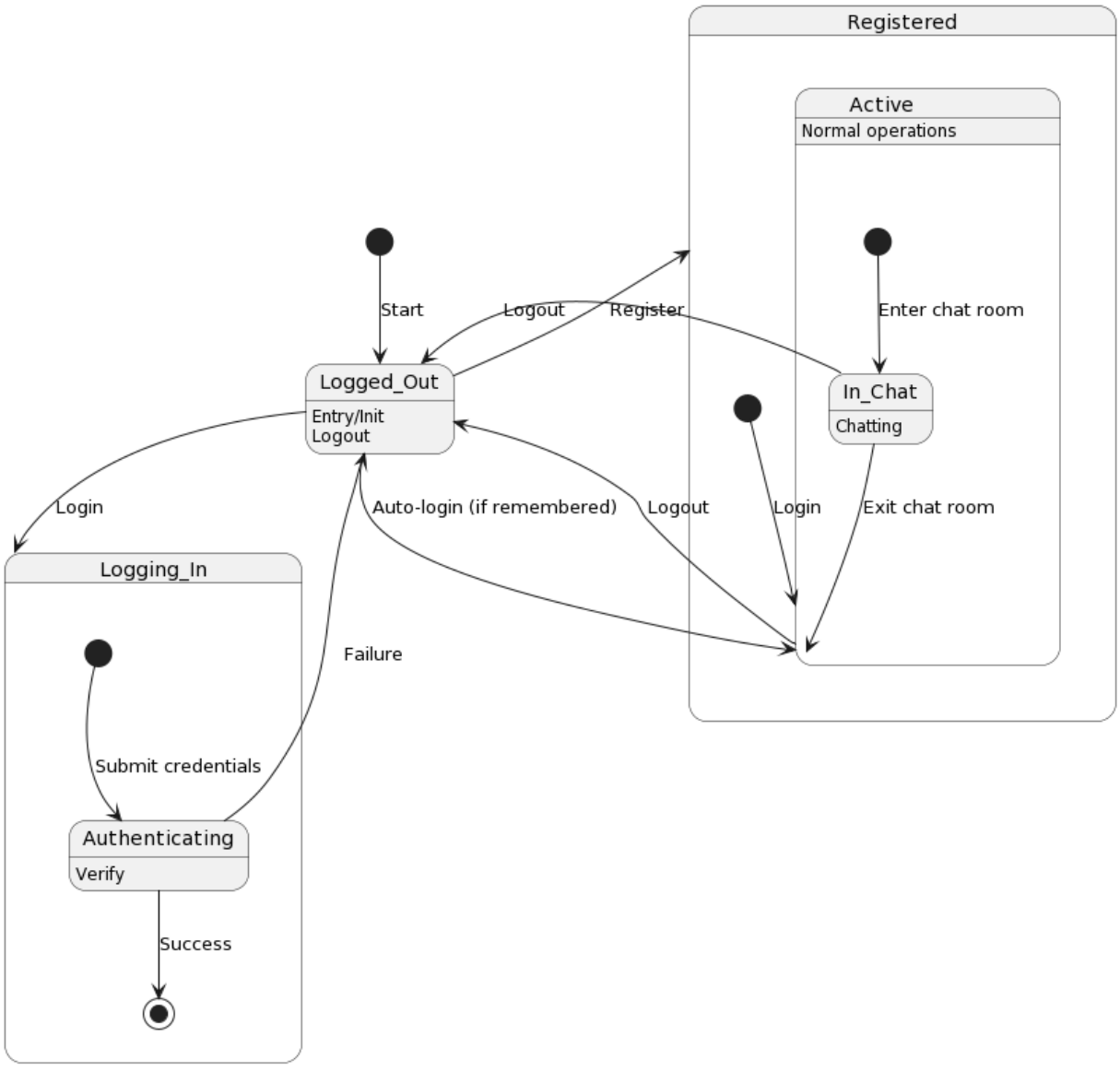


Figure: *State-Chart Diagram for Authentication System, Maintenance, Monitor*

Name: Hitesh Bardia **Enrolment No.** 07418002721

Activity Diagram:

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. Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

1. Authentication System, Maintenance, Monitor

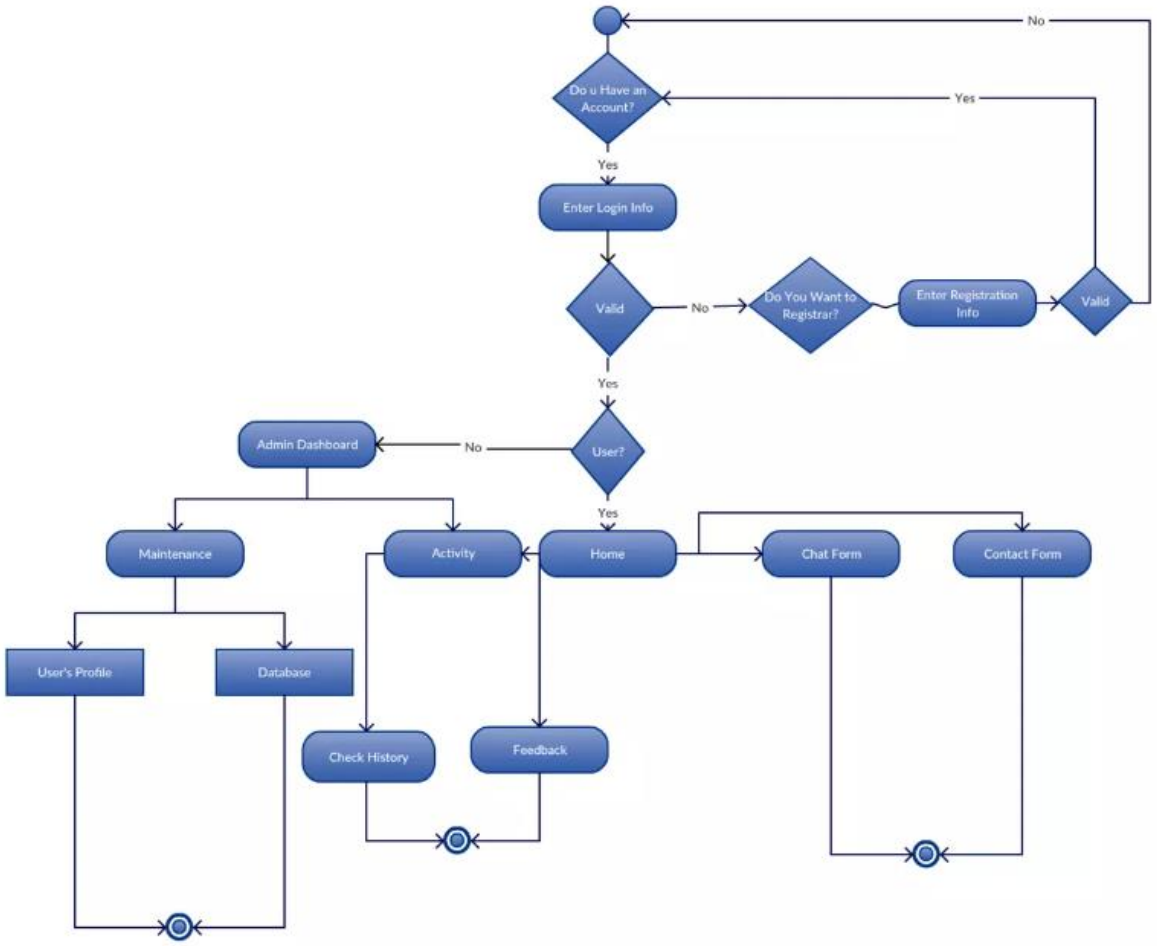


Figure: *Activity Diagram for Authentication System, Maintenance, Monitor*

Name: Hitesh Bardia **Enrolment No.** 07418002721

2. Contact Form, Chat Form

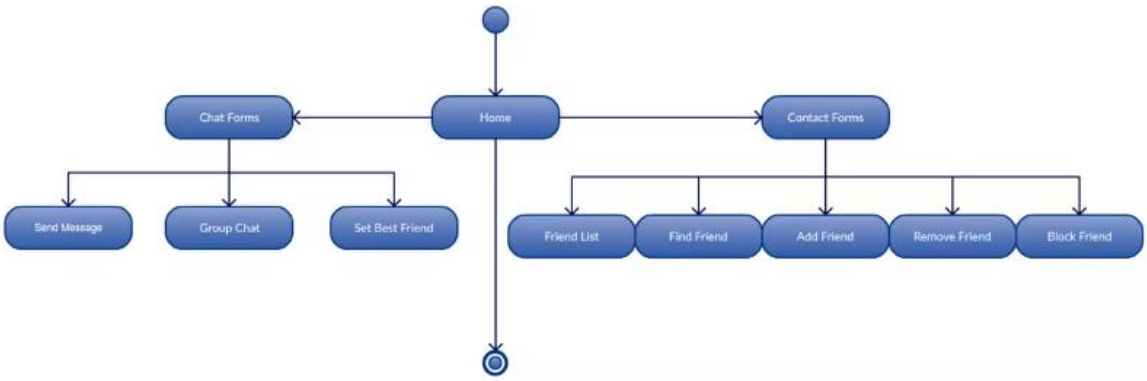


Figure: *Activity Diagram for Contact Form, Chat Form*

Conclusion:

State-Chart diagram and Activity diagram for Real-time Chat Application has been done successfully.

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 8

AIM: To perform the behavioural view diagram for the Real-time chat application: Sequence diagram, Collaboration diagram.

Sequence Diagram:

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focused and they show the order of the interaction visually by using the vertical axis of the diagram to represent time, what messages are sent and when.

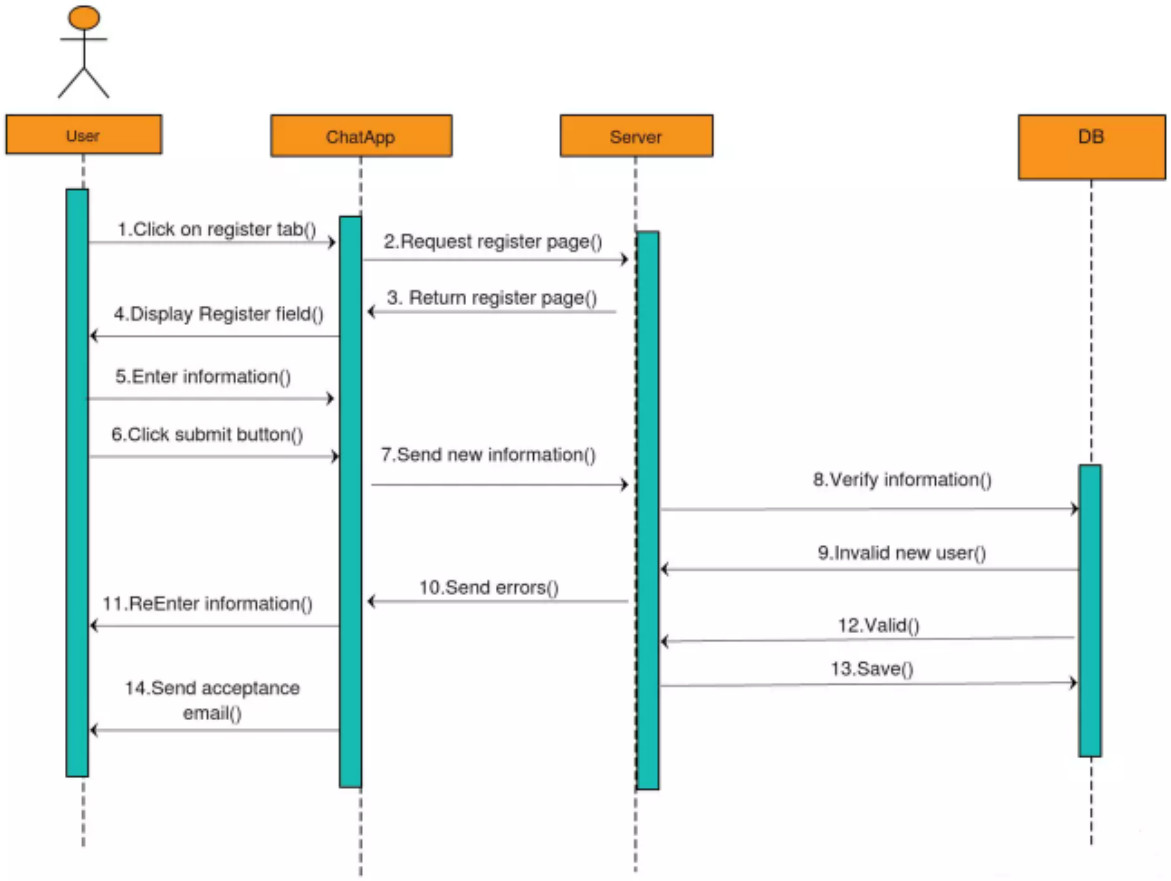


Figure: *Sequence Diagram for Registration Functionality*

Name: Hitesh Bardia **Enrolment No.** 07418002721

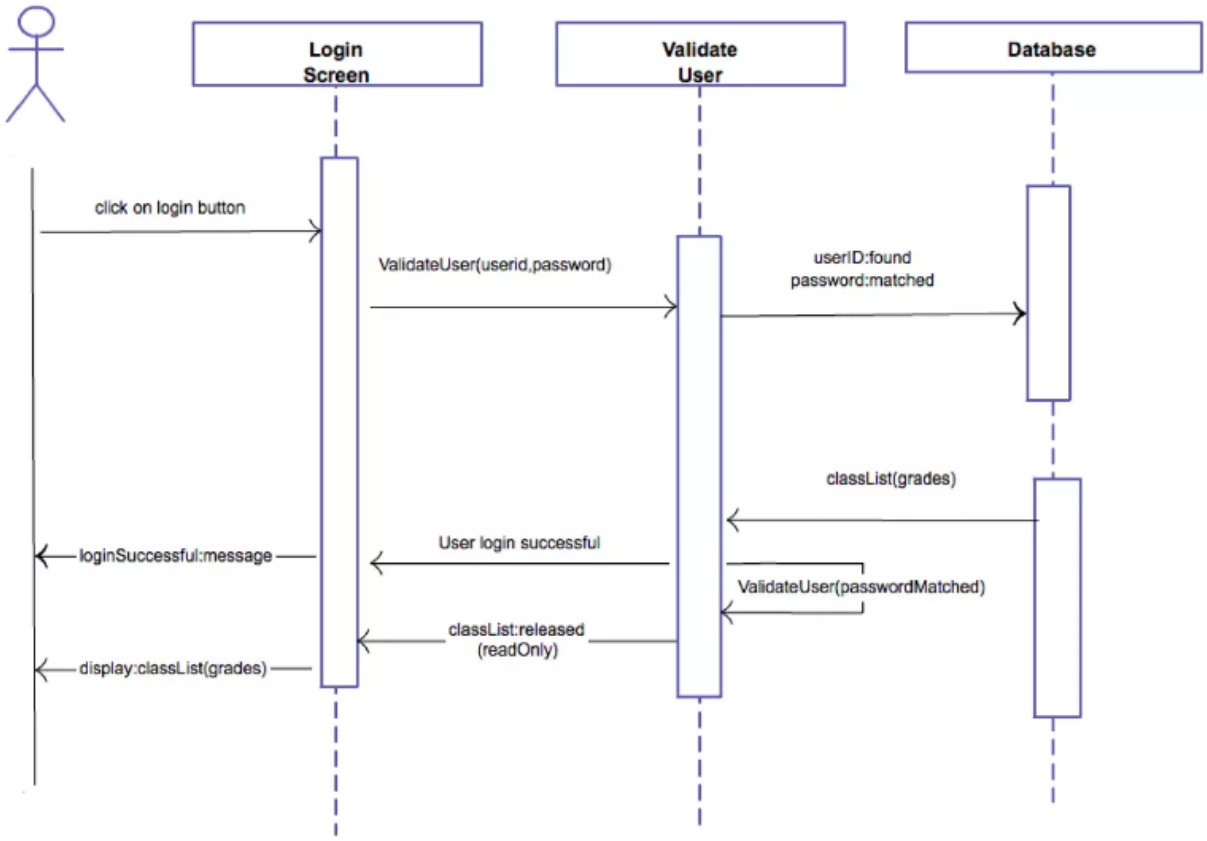


Figure: *Sequence Diagram for Login Functionality*

Name: Hitesh Bardia **Enrolment No.** 07418002721

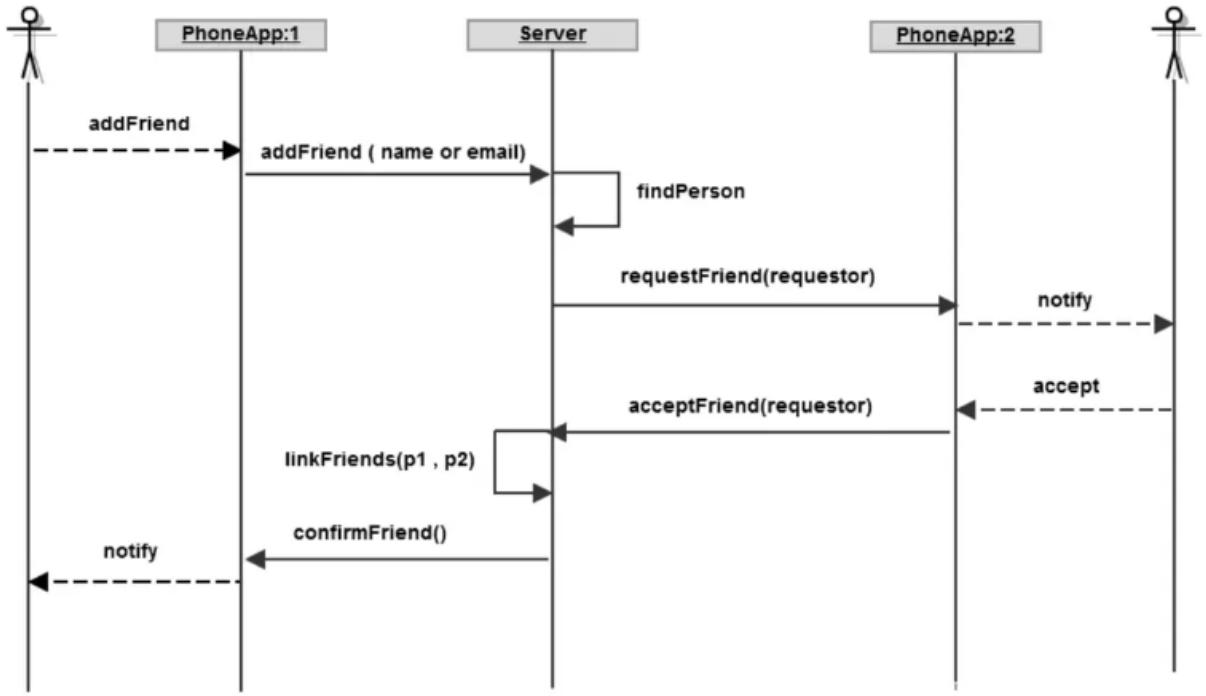


Figure: *Sequence Diagram for Add Friend Functionality*

Name: Hitesh Bardia **Enrolment No.** 07418002721

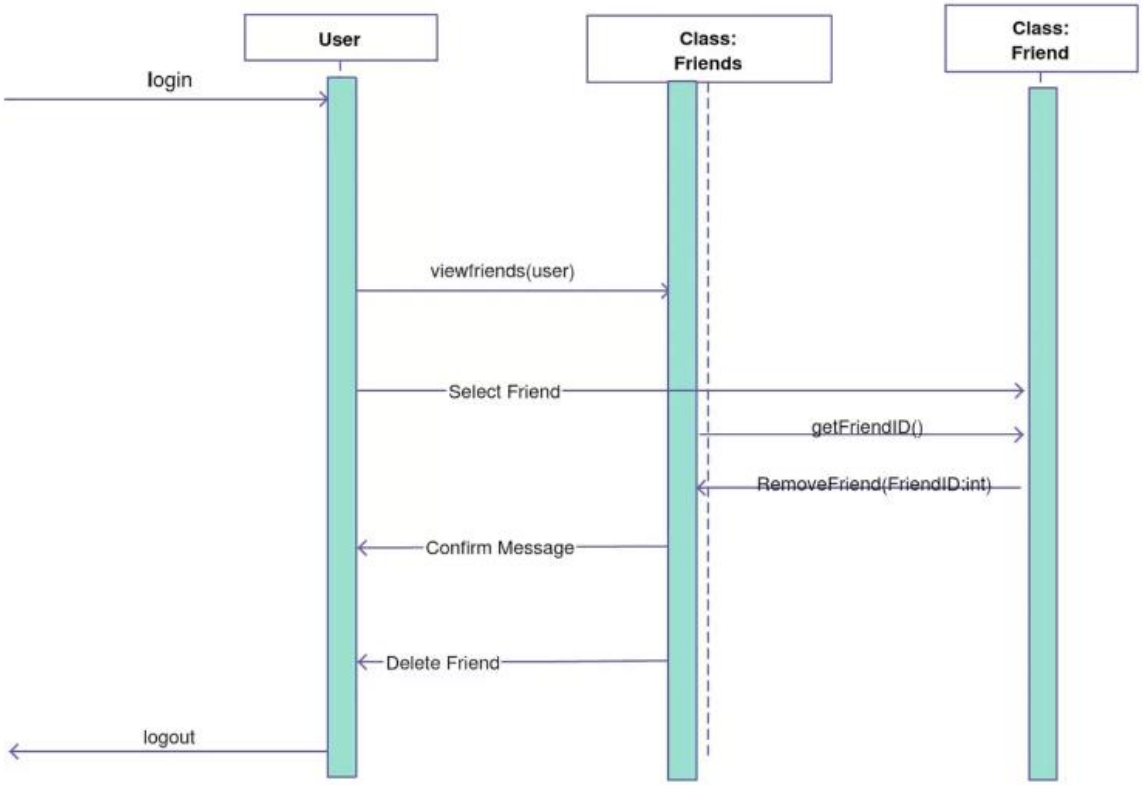


Figure: *Sequence Diagram for Remove Friend Functionality*

Name: Hitesh Bardia **Enrolment No.** 07418002721

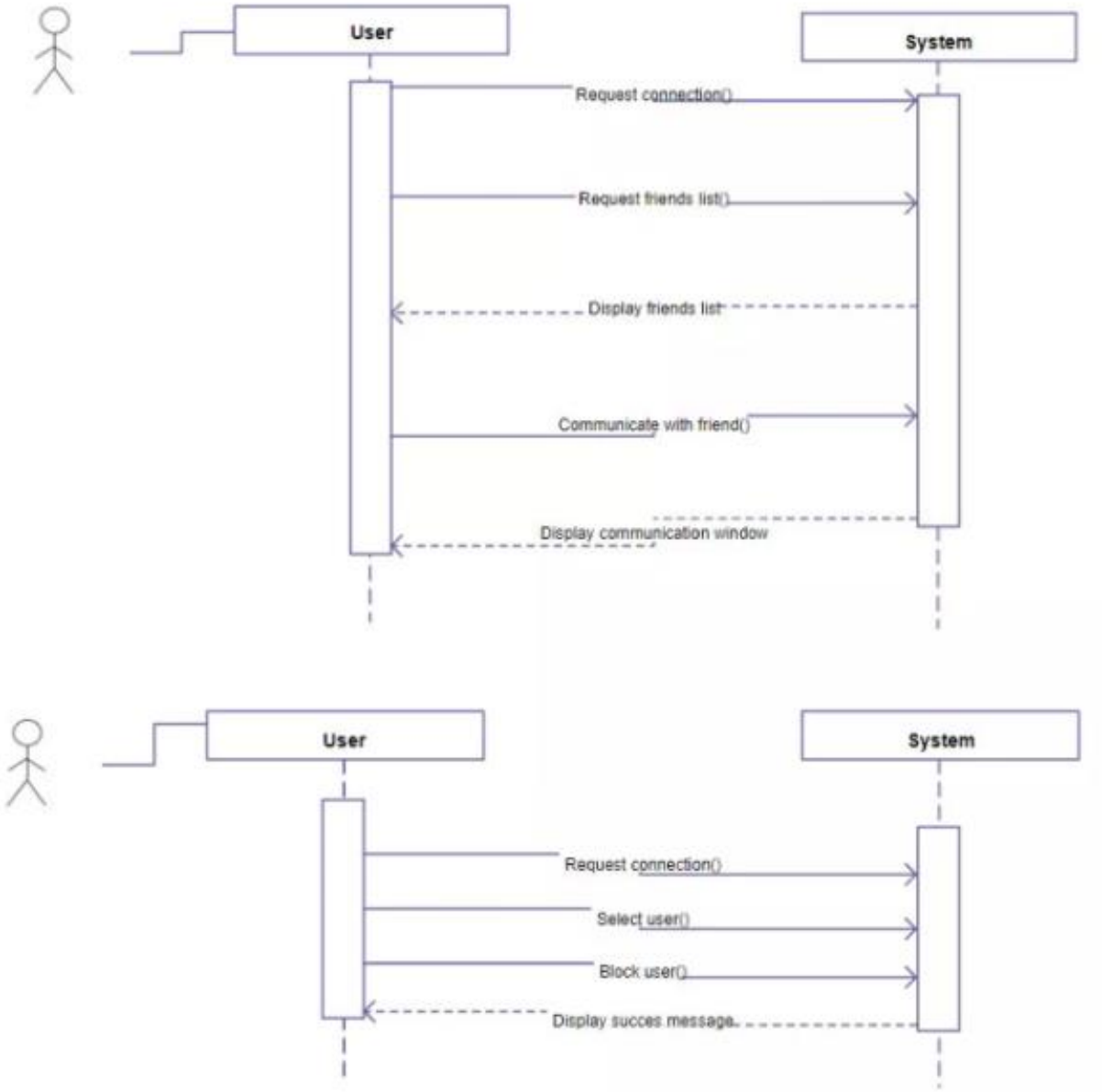


Figure: *Sequence Diagram for Block Friend Functionality*

Name: Hitesh Bardia **Enrolment No.** 07418002721

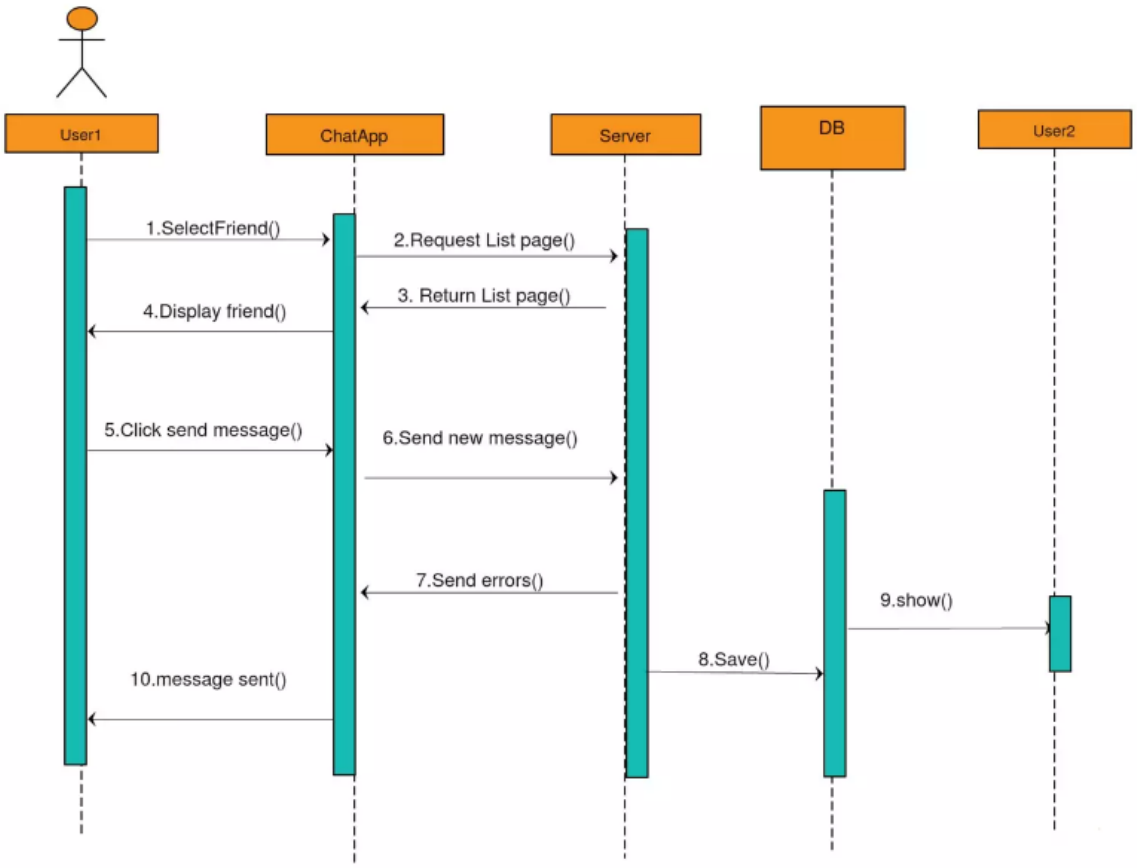


Figure: *Sequence Diagram for Send Message Functionality*

Name: Hitesh Bardia **Enrolment No.** 07418002721

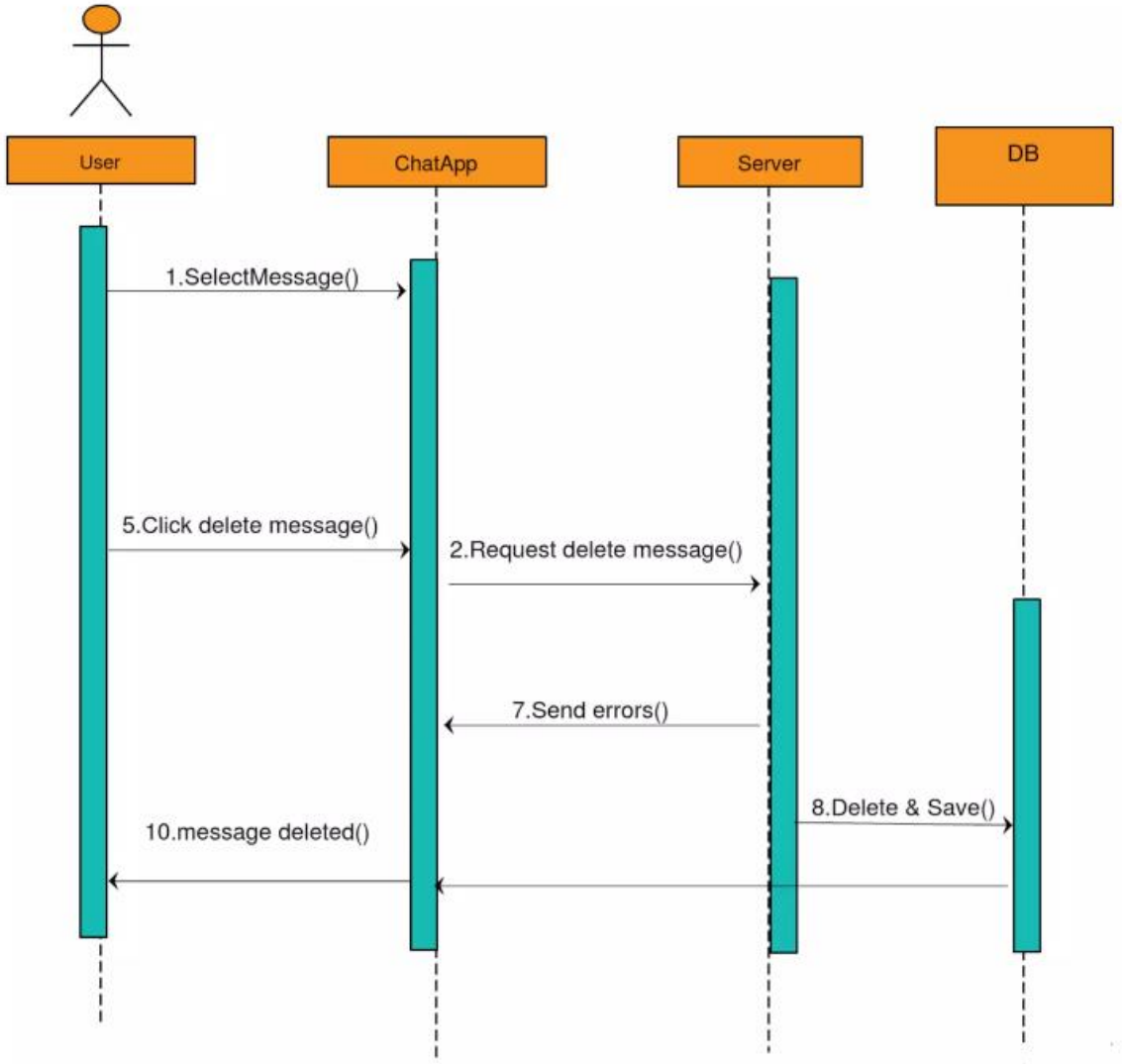


Figure: *Sequence Diagram for Delete Message Functionality*

Name: Hitesh Bardia **Enrolment No.** 07418002721

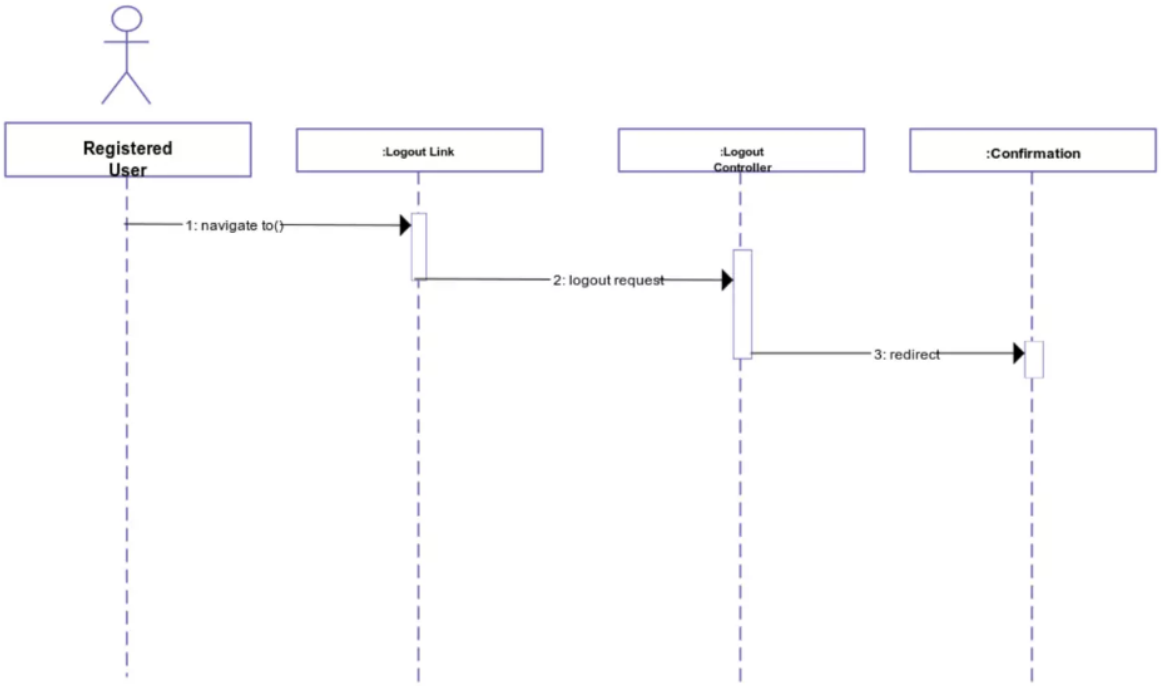


Figure: *Sequence Diagram for Logout Functionality*

Conclusion:

Sequence diagram for Real-time Chat Application has been done successfully.

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 9

AIM: To perform the implementation view diagram: Component diagram for the Real-time chat application.

REQUIREMENTS:

Software Requirements – Microsoft Word
Hardware Requirements – Computer, keyboard, mouse, CPU.

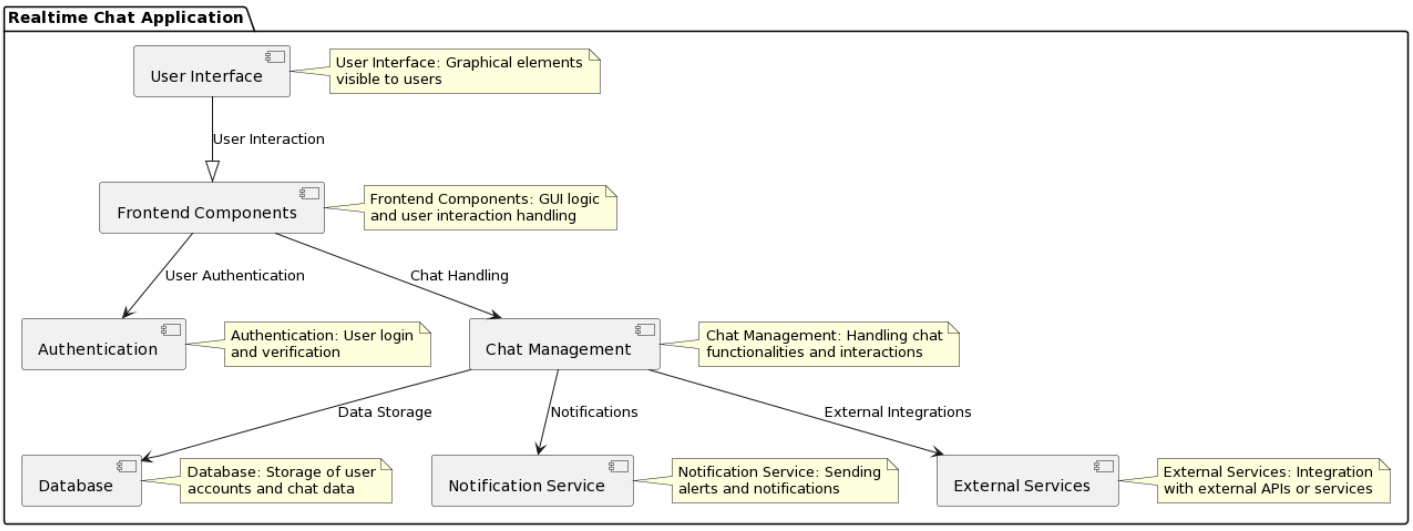
Components:

- **User Interface:** Handles graphical interface elements visible to users, facilitating user interaction with the application.
- **Frontend Components:** Encompasses GUI logic and user interaction handling, translating user actions into backend requests.
- **Authentication:** Manages user login and verification processes, ensuring secure access to the application.
- **Chat Management:** Handles chat functionalities, facilitating message sending, receiving, and other chat-related features.
- **Database:** Stores user accounts, chat logs, and other application-related data securely.
- **Notification Service:** Responsible for sending alerts and notifications to users for various events within the application.
- **External Services:** Integrates with external APIs or services, enabling additional functionalities or data exchange beyond the application's core features.

Name: Hitesh Bardia **Enrolment No.** 07418002721

Component Diagram:

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node. It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system. A component is a single unit of the system, which is replaceable and executable. The implementation details of a component are hidden, and it necessitates an interface to execute a function. It is like a black box whose behaviour is explained by the provided and required interfaces.



Conclusion:

Component diagram for Real-time Chat Application has been done successfully.

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 10

AIM: Perform Estimation of effort using FP Estimation for Real-time chat application.

REQUIREMENTS:

Software Requirements – Microsoft Word
Hardware Requirements – Computer, keyboard, mouse, CPU.

THEORY:

Function Point (FP) estimation is a method used for estimating the size of software projects based on functionalities or functional components. For a Real-time Chat Application, we can estimate the effort using the Function Point Analysis approach.

Here are the basic steps for FP estimation:

1. Identify Functional Components:

- External Inputs (EI): User logins, message sending, profile edits.
- External Outputs (EO): Displaying chat messages, notifications.
- External Inquiries (EQ): Search functionality, user profile view.
- Internal Logical Files (ILF): User accounts, chat logs, message database.
- External Interface Files (EIF): Integration with external services, APIs.

2. Assign Complexity Weights:

Assign complexity weights based on the complexity of each functional component. For example:

- Low Complexity: Simple user login.
- Medium Complexity: Chat message sending/receiving.
- High Complexity: Complex search functionalities across multiple parameters.

Name: Hitesh Bardia **Enrolment No.** 07418002721

3. Calculate Function Points:

- For each functional component, use complexity weights and count the occurrences of each.
- Calculate Unadjusted Function Points (UFP) using a weighted sum of the components.
- Apply complexity factors (such as Technical Complexity, Environmental Complexity, etc.) to calculate Adjusted Function Points (AFP).

4. Effort Estimation:

- Use historical data or industry norms to convert Adjusted Function Points (AFP) into person-months or person-hours based on your team's productivity.

Name: Hitesh Bardia **Enrolment No.** 07418002721

Calculating Function Points (FP) involves a detailed analysis of the functionalities, their complexities, and assigning weights to them. Since we don't have detailed information on each functional component and its complexity in your Real-time Chat Application, I'll provide a basic hypothetical estimation:

Complexity Levels:

Simple (Low Complexity):

- User Authentication
- Contact List Management
- Basic Messaging
- User Profile Customization

Average (Medium Complexity):

- Group Chat Management
- Emoji and Reaction System
- Profile Customization

Complex (High Complexity):

- Real-time Messaging
- Notifications
- Search Functionality
- Security and Authentication
- Data Storage and Management
- Integration with External Services

Name: Hitesh Bardia **Enrolment No.** 07418002721

Simple (Low Complexity):

1. **Inputs:**

    - User Authentication: 4 inputs (username, password, etc.)
    - Contact List Management: 2 inputs (add contact, remove contact)
    - Basic Messaging: 1 input (message text)
    - User Profile Customization: 3 inputs (avatar upload, status update, etc.)

    Total Inputs: 10

2. **Outputs:**

    - Authentication Success/Failure: 1 output
    - Contact List Display: 1 output
    - Display of Sent/Received Messages: 1 output
    - Updated User Profile Details: 1 output

    Total Outputs: 4

3. **Enquiries:**

    - Authentication Status: 1 enquiry
    - Contact List Updates: 1 enquiry
    - Message Status: 1 enquiry

    Total Enquiries: 3

4. **Interfaces:**

    - User Authentication Interface: 1 interface
    - Contact List Management Interface: 1 interface
    - Messaging Interface: 1 interface
    - Profile Customization Interface: 1 interface

    Total Interfaces: 4

5. **User Files:**

    - Profile Images: 1 file per user
    - Customized User Profile Data: 1 file per user

    Total User Files: 2

Name: Hitesh Bardia **Enrolment No.** 07418002721

Average (Medium Complexity):

1. **Inputs:**

    - Group Chat Management: 3 inputs (create group, add member, remove member)
    - Emoji and Reaction System: 2 inputs (select emoji, react to message)
    - Profile Customization: 2 inputs (extended profile customization)

    Total Inputs: 7

2. **Outputs:**

    - Group Chat Displays: 1 output
    - Reaction Displays on Messages: 1 output
    - Extended Profile Customization Updates: 1 output

    Total Outputs: 3

3. **Enquiries:**

    - Group Chat Status: 1 enquiry
    - Reaction Feedback: 1 enquiry
    - Additional Profile Customization Details: 1 enquiry

    Total Enquiries: 3

4. **Interfaces:**

    - Group Chat Management Interface: 1 interface
    - Emoji/Reaction Selection Interface: 1 interface
    - Advanced Profile Customization Interface: 1 interface

    Total Interfaces: 3

5. **User Files:**

    - Additional Files for Group-Related Data: Varies with group count
    - Emoji/Reaction Preferences: 1 file per user
    - Extended Profile Customization Data: 1 file per user

    Total User Files: Varies

Name: Hitesh Bardia **Enrolment No.** 07418002721

Complex (High Complexity):

1. **Inputs:**

   - Real-time Messaging: Continuous inputs per message
   - Notifications: Preference settings
   - Search Queries: Dynamic user input
   - Security and Authentication: Varies based on security features
   - Data Storage and Management: Continuous inputs for data handling
   - Integration with External Services: Inputs specific to service integration

   Total Inputs: Dynamic and variable

2. **Outputs:**

   - Real-time Message Updates: Continuous outputs
   - Notifications: Varied outputs
   - Search Results: Dynamic outputs
   - Security-Related Alerts: Varied outputs
   - Integrated External Service Data: Varied outputs

   Total Outputs: Dynamic and variable

3. **Enquiries:**

   - Real-time Message Status: Continuous enquiries
   - Notification Delivery Status: Varied enquiries
   - Search Result Accuracy: Varies with search complexity
   - Security Checks and Logs: Continuous enquiries
   - Interaction with External Services: Enquiries specific to integration

   Total Enquiries: Dynamic and variable

4. **Interfaces:**

   - Real-time Messaging Interface: Continuous interface
   - Notification Interface: Varied interfaces
   - Search Functionality Interface: Dynamic interface
   - Advanced Security Settings Interface: Varied interfaces
   - Interfaces for Integrated External Services: Specific interfaces

   Total Interfaces: Dynamic and variable

Name: Hitesh Bardia **Enrolment No.** 07418002721

5. **User Files:**

- Real-time Message Logs: Continuous files
- Notification Preferences and Logs: Varied files
- Search History and Preferences: Varied files
- Security-Related Logs and Data: Continuous files
- Files Related to External Service Integration: Specific files

Total User Files: Dynamic and variable

The complexity level significantly impacts the variability and quantity of inputs, outputs, enquiries, interfaces, and user files in a real-time chat application. The higher the complexity, the more dynamic and variable these elements become.

The UFP is calculated by summing up the scores for each type of function (inputs, outputs, inquiries, interfaces, and user files) across the three complexity levels.
Let's assume these UFP values for each complexity level:

Functional Aspect	Simple (Low Complexity)	Average (Medium Complexity)	Complex (High Complexity)
Inputs	10	7	Dynamic and variable
Outputs	4	3	Dynamic and variable
Enquiries	3	3	Dynamic and variable
Interfaces	4	3	Dynamic and variable
User Files	2	Varies	Dynamic and variable

Name: Hitesh Bardia **Enrolment No.** 07418002721

Calculations:

Using the Simple (Low Complexity) weighting factors for each aspect. Typically, Simple (Low Complexity) has a weighting factor of 3 for Inputs, Outputs, Enquiries, Interfaces, and User Files.

1. Unadjusted Function Point (UFP):

The UFP is calculated by summing up the scores for each type of function (inputs, outputs, inquiries, interfaces, and user files)

Functional Aspect	Count	Weighting Factor Simple (Low Complexity)	Subtotal
Inputs	10	3	30
Outputs	4	3	12
Enquiries	3	3	9
Interfaces	4	3	12
User Files	2	3	6
Total UFP			69

UFP=Total Subtotal of all Functional Aspects = 30+12+9+12+6 = 69

2. Complexity Adjustment Factor (CAF):

CAF is determined based on various factors like database size, transaction rate, complexity, etc. For the sake of an example, let's assume a CAF of 1.2.

3. Functional Point Analysis (FPA):

FPA(Simple) = UFP x CAF
FPA = 69 x 1.2
FPA = 82.8

Therefore:

- Unadjusted Function Point (UFP) = 69
- Complexity Adjustment Factor (CAF) = 1.2
- Functional Point Analysis (FPA) = 82.8

Name: Hitesh Bardia **Enrolment No.** 07418002721

EXPERIMENT 11

AIM: To prepare time line chart/Gantt Chart/PERT Chart for Real-time chat application.

REQUIREMENTS:

Software Requirements – Microsoft Word, PlantUML
Hardware Requirements – Computer, keyboard, mouse, CPU.

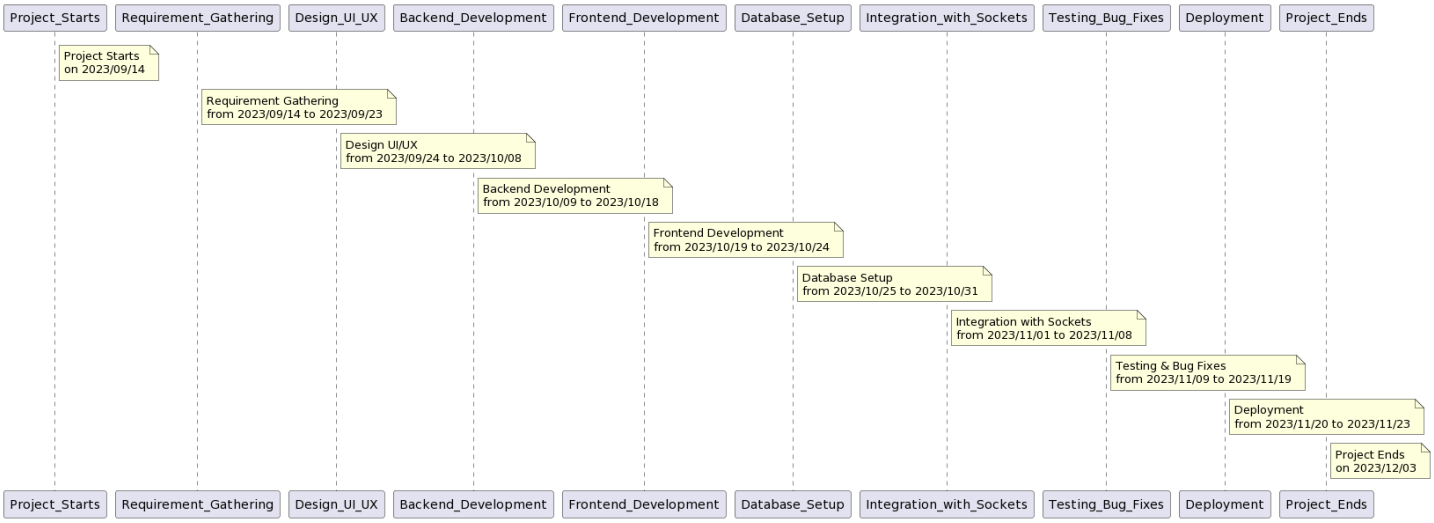
Steps to Create Time-Line Chart for Real-time Chat App:

1. **Define Project Timeline Parameters:**
   - Determine the project's start and end dates.
   - Identify the duration of each task involved in the project.
2. **Task Identification:**
   - List and define the individual tasks constituting the project.
3. **Utilize PlantUML's Syntax:**
   - Employ PlantUML's Gantt chart syntax for visualizing project timelines.
   - Use the appropriate syntax elements to represent tasks, their durations, and start and end dates.
4. **Task Dependencies:**
   - Identify task dependencies, if any, within the project workflow.
   - Apply PlantUML's syntax to illustrate task dependencies effectively.
5. **PlantUML Code Creation:**
   - Develop a PlantUML code snippet reflecting the project's timeline.
   - Include task durations, start and end dates, and accurately depict task interdependencies.
6. **Test and Validation:**
   - Execute the PlantUML code to generate and verify the accuracy of the timeline visualization.
   - Review the representation for completeness, correctness, and coherence.

Name: Hitesh Bardia **Enrolment No.** 07418002721

7. **Refinement and Adjustment:**
    - Fine-tune the timeline chart, ensuring clear visualization and accurate portrayal of task sequences and dependencies.
    - Make necessary adjustments to enhance clarity and precision.
8. **Incorporate into Documentation:**
    - Integrate the finalized timeline chart into project documentation, presentations, or related materials.
    - Embed the chart in relevant sections to convey the project's timeline effectively.

Time Line Chart:



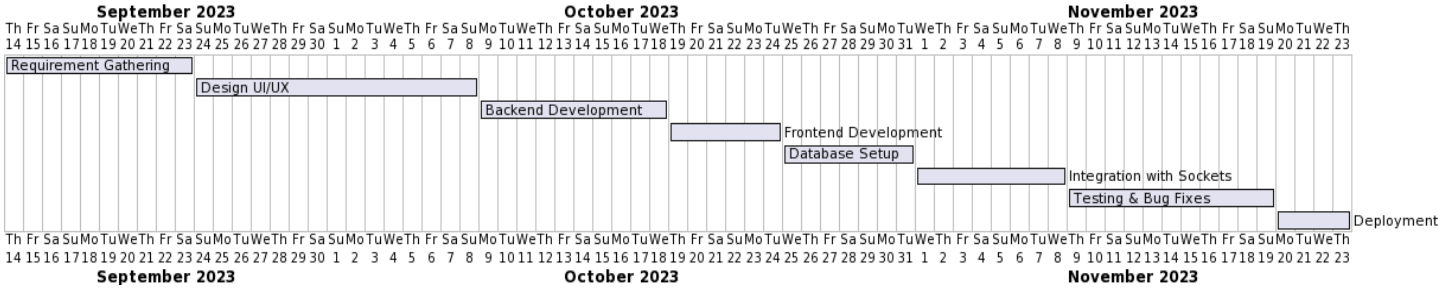
Name: Hitesh Bardia **Enrolment No.** 07418002721

Steps to Create Gantt Chart for Real-time Chat App:

1. **Task Identification:**
    - List all project tasks such as design, development, testing, deployment, etc.
    - Break down tasks into granular activities for each phase.
2. **Task Dependencies:**
    - Define dependencies between tasks. Some tasks may need to be completed before others can start.
3. **Estimate Duration:**
    - Estimate the time required for each task or activity. Use historical data or expert judgment.
4. **Sequence Tasks:**
    - Arrange tasks in chronological order considering dependencies.
5. **Use Gantt Chart Software:**
    - Use specialized software like Microsoft Project, Asana, or even online Gantt chart generators like PlantUML.
    - Enter tasks, durations, dependencies, and start dates to create the Gantt chart.
6. **Visual Representation:**
    - The Gantt chart visually represents tasks over time, showing start and end dates, durations, and dependencies.
7. **Review and Update:**
    - Regularly review and update the Gantt chart as the project progresses.

Name: Hitesh Bardia **Enrolment No.** 07418002721

Gantt Chart:

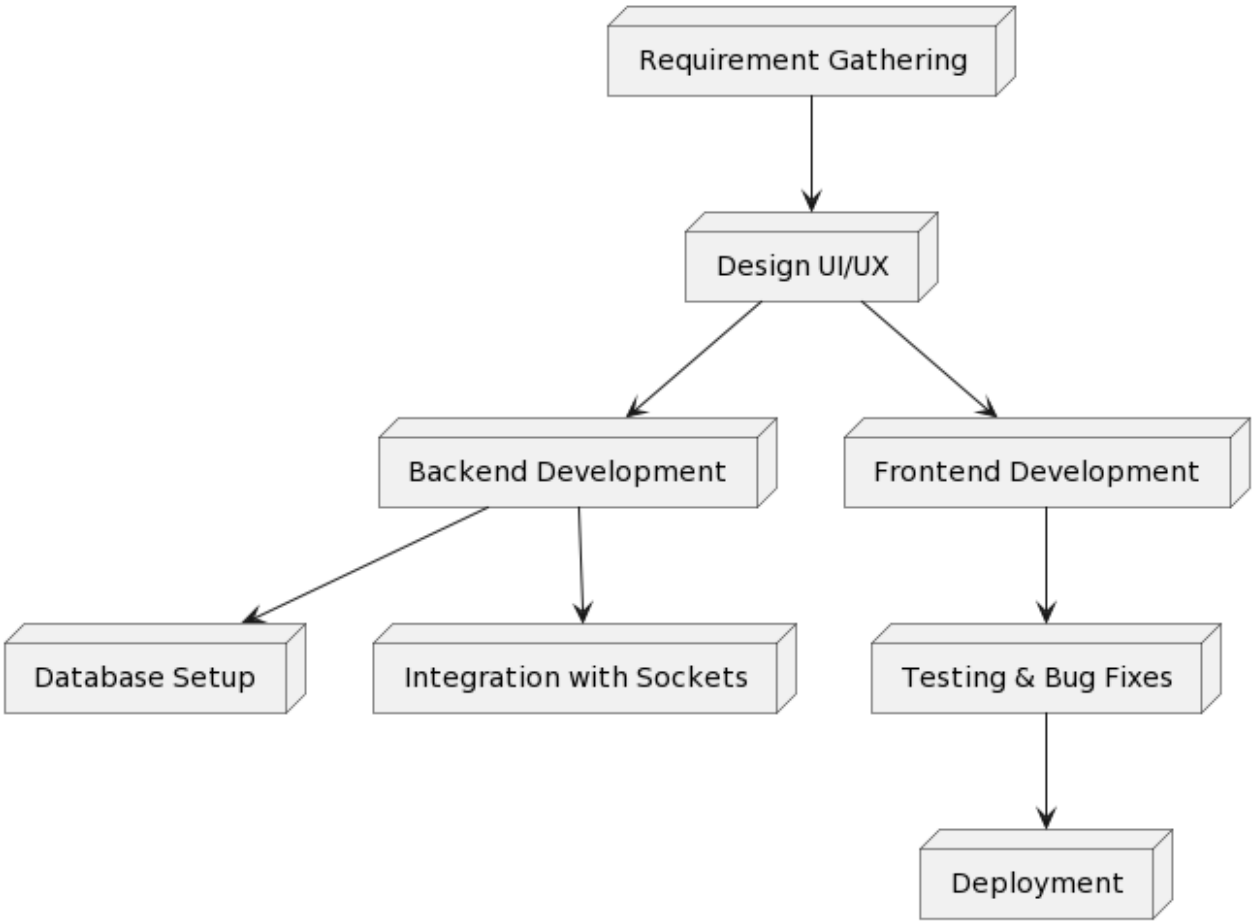


Name: Hitesh Bardia **Enrolment No.** 07418002721

Steps to Create a PERT Chart for the Real-time Chat App

1. **Task Identification and Sequencing:**
    - List and define the individual tasks essential for the Real-time Chat App project.
    - Determine the sequence and dependencies among these tasks.
2. **Estimate Task Durations:**
    - Assess and estimate the duration required for each task based on project requirements and resources available.
3. **Identify Critical Path:**
    - Determine the critical path, i.e., the sequence of tasks with the longest duration, crucial for the project's timeline.
4. **Utilize PlantUML's Syntax:**
    - Use PlantUML's notation to represent tasks and their interdependencies effectively.
5. **Create Nodes and Connectors:**
    - Employ PlantUML's nodes and arrows to illustrate tasks as nodes and connect them using arrows to display task flow and dependencies.
6. **Task Relationships:**
    - Establish relationships between tasks, specifying precedence and dependencies clearly.
7. **Test and Validation:**
    - Execute the PlantUML code to generate the PERT chart.
    - Validate the chart to ensure accurate representation of task sequences and critical paths.

Name: Hitesh Bardia **Enrolment No.** 07418002721

8. **Refinement and Adjustment:**
    - Fine-tune the PERT chart for clarity and precision, ensuring the depiction of task sequences and dependencies is accurate.
9. **Incorporate into Documentation:**
    - Integrate the finalized PERT chart into project documentation or presentations, ensuring it appropriately represents the project's task flow and critical path.

PERT Chart: