A
PROJECT REPORT ON

Icollab
A Collaborative Tool

By

MEET ANTALA (CE-087) (22CEUOS096)
YASH GABANI (CE-104) (22CEUOS137)

B.Tech CE Semester-VI
Subject: System Design Practice

Guided by:
Prof. Neha Fotedar
Assistant Professor
Dept. of Comp. Engg.

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

CERTIFICATE

This is to certify that the practical / term work carried out in the subject of

System Design Practice and recorded in this journal is the bonafide

work of

MEET ANTALA (CE-087) (22CEUOS096)
YASH GABANI (CE-104) (22CEUOS137)

of B.Tech semester **VI** in the branch of **Computer Engineering**

during the academic year **2024-2025**.

Prof. Neha Fotedar
Assistant Professor,
Dept. of Computer Engg.,
Faculty of Technology
Dharmsinh Desai University, Nadiad

Dr. C. K. Bhensdadia,
Head,
Dept. of Computer Engg.,
Faculty of Technology
Dharmsinh Desai University, Nadiad

Table of Contents

Abstract

Icollab is a comprehensive, web-based team collaboration platform designed to empower modern teams by streamlining communication, enhancing productivity, and simplifying project coordination. Developed using the robust MERN stack (MongoDB, Express.js, React.js, Node.js), the platform offers a centralized environment where teams can manage their workflows through real-time messaging, task assignment and tracking, and live video communication.

Users can create and manage multiple workspaces and organize conversations into topic-specific channels, improving clarity and team focus. A built-in task management system enables admins to assign responsibilities, monitor progress, and ensure timely completion of goals. The platform also integrates **WebRTC** for seamless peer-to-peer video communication.

One of Icollab's standout features is its AI-powered chat summarization to generate concise summaries of lengthy discussions, saving time and improving information accessibility. Fully responsive and compatible across devices, Icollab is a future-ready solution tailored for startups, enterprises, and distributed teams seeking efficient digital collaboration.

Chapter 1
Introduction

1.1 Brief Introduction:

Icollab is a smart, interactive, and full-featured collaboration system designed for modern-day organizations and distributed teams. It offers a centralized platform to manage team communication, coordinate project activities, and handle real-time collaboration requirements. The core goal of Icollab is to bring all essential teamwork tools into one unified space. It allows teams to create project-specific workspaces that can house multiple channels dedicated to various discussions, departments, or projects. Users can engage in real-time messaging, schedule and participate in video meetings, and assign and track tasks to ensure timely delivery.

The platform emphasizes ease of use, data security, and flexibility, offering a smooth user experience with role-based access controls and interactive UI components.. One of the standout innovations of Icollab is its AI-based chat summarization tool, which uses Natural Language Processing to generate concise summaries from lengthy conversations—helping users save time and focus on key outcomes.

1.2 Scope:

Icollab is designed for small to large teams that operate across various industries, including software development, marketing, education, design, and operations. It specifically caters to the increasing demand for digital workplaces that are not just communication-driven, but also collaborative and task-oriented. The platform is ideal for professionals seeking clarity, accountability, and transparency in team communication.

The scope of Icollab includes:

- Seamless communication via real-time chat and video conferencing.

- Efficient project and task coordination through visual boards and deadlines.
- Centralized document sharing and management with permission-based access.
- Automatic chat summarization to reduce information overload.
- Scalable structure that supports multiple workspaces and hundreds of channels.
- Admin control over user access, project visibility, and collaboration settings.

By providing all these features under a single platform, Icollab reduces the need for switching between multiple tools and helps teams stay focused and organized. It can be used in academic institutions, corporate environments, remote teams, or startups aiming for structured digital collaboration.

1.3 Technology/Platform/Tools Used:

To deliver a high-performance, responsive, and scalable solution, Icollab has been developed using the following technologies and platforms:

- **Frontend:** React.js
  - A powerful JavaScript library used for building dynamic user interfaces with component-based architecture.
- **Backend:** Node.js with Express.js
  - Provides a fast and scalable server-side environment to handle APIs, database interactions, and user authentication.
- **Authentication:** Google OAuth and JSON Web Tokens (JWT)
  - Secure login system allowing users to authenticate using Google accounts, with session protection via JWTs.
- **Database:** MongoDB
  - A flexible NoSQL database that stores user data, workspace metadata, messages, tasks, and more using a document-based model.
- **Real-Time Communication:** WebRTC and Socket.IO
  - WebRTC allows for live audio/video communication directly between peers, while Socket.IO supports real-time messaging, typing indicators, and chat updates.
- **AI Integration:** OpenAI GPT APIs

    - Powers the chat summarization feature using natural language processing to turn long discussions into concise summaries.
- **Development Tools:** Visual Studio Code, Postman, GitHub
- **Deployment Platforms:** Render, Firebase Hosting, or Vercel for deploying both frontend and backend services securely.

These technologies work together to deliver a seamless user experience that is fast, secure, and intuitive for collaborative environments.

Chapter 2
System Analysis

2.1 Class Diagram

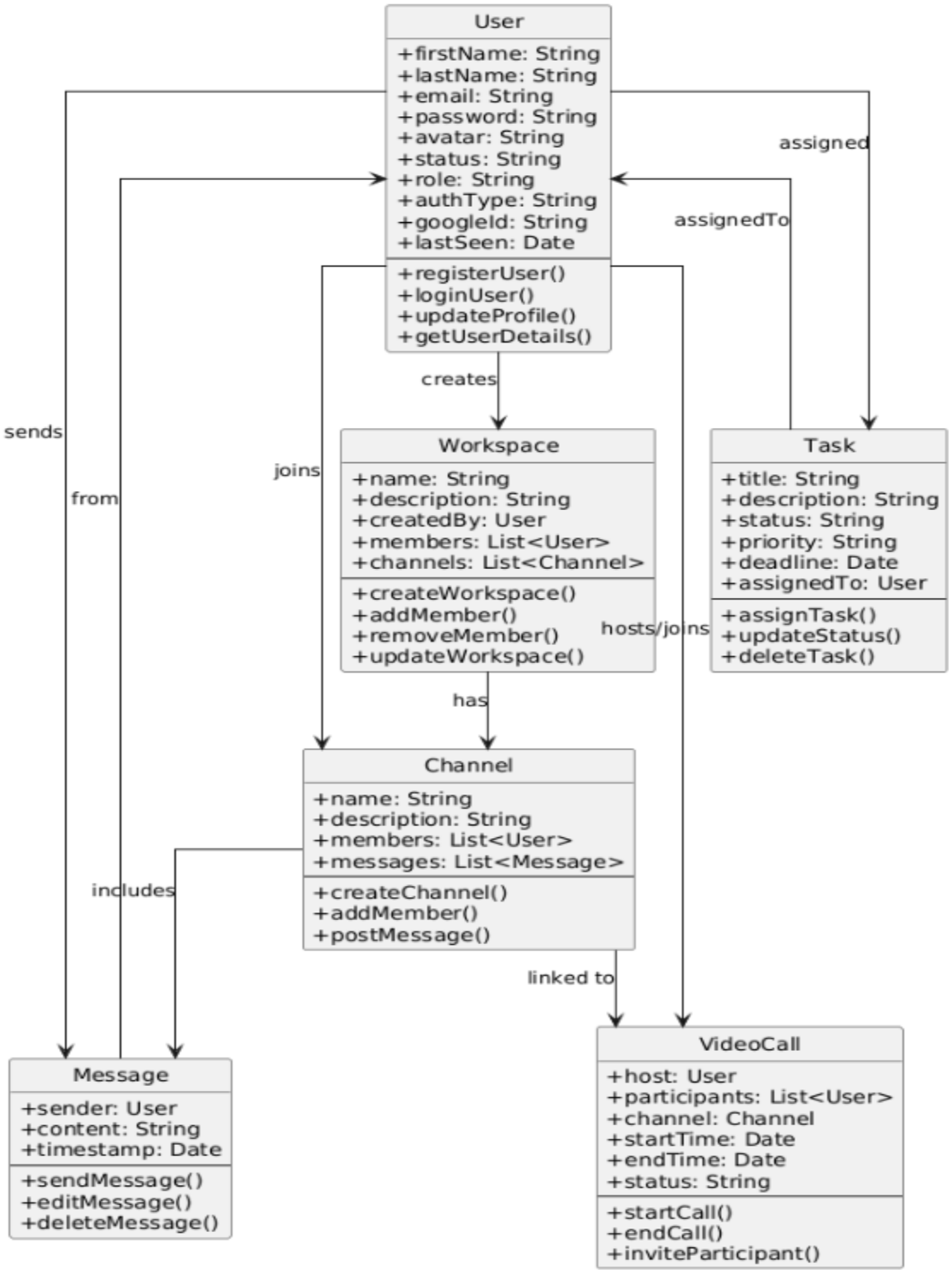

Figure 3.1 Class Diagram

2.2 Use-case diagram

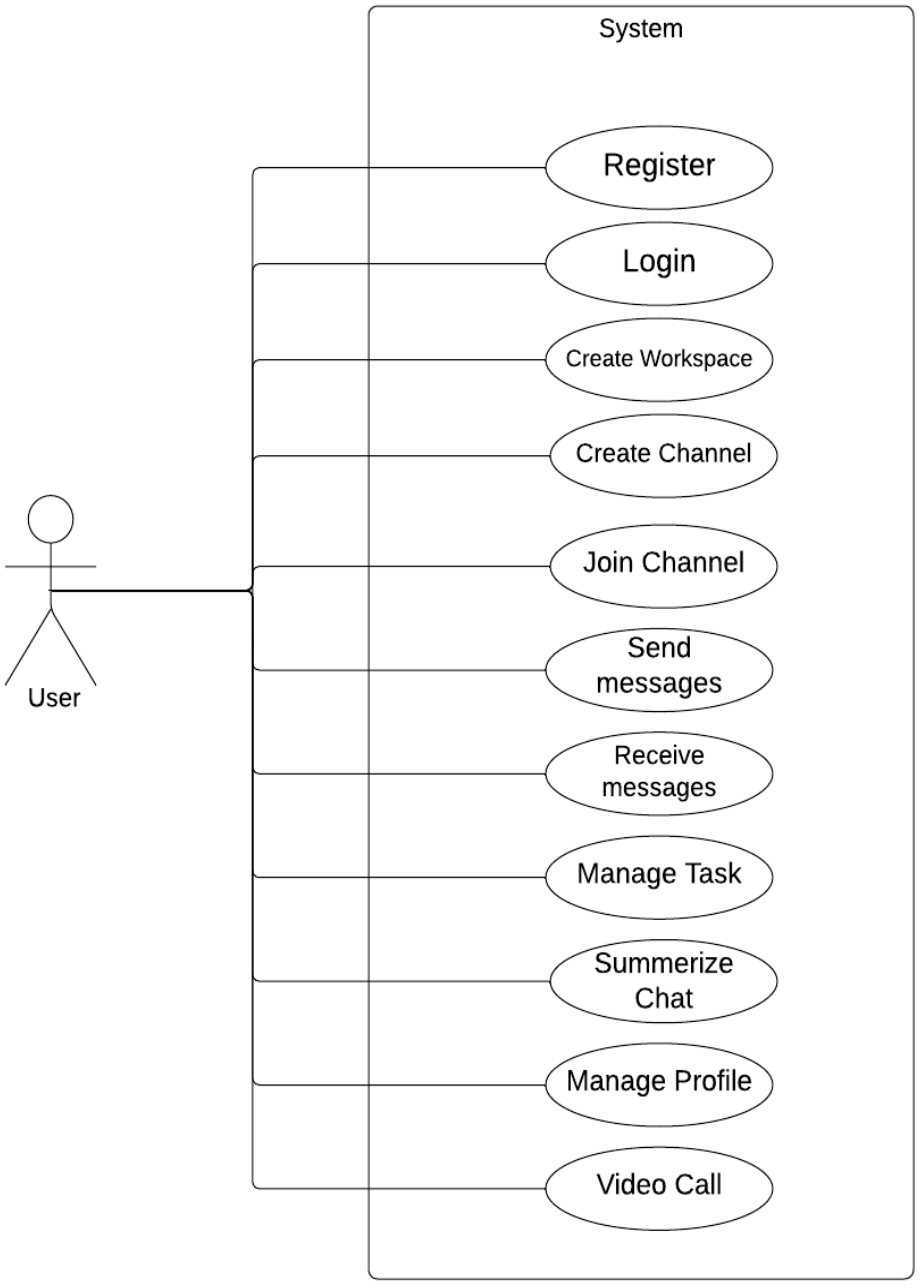

Figure 3.2 Use Case Diagram

2.3 Sequence Diagram

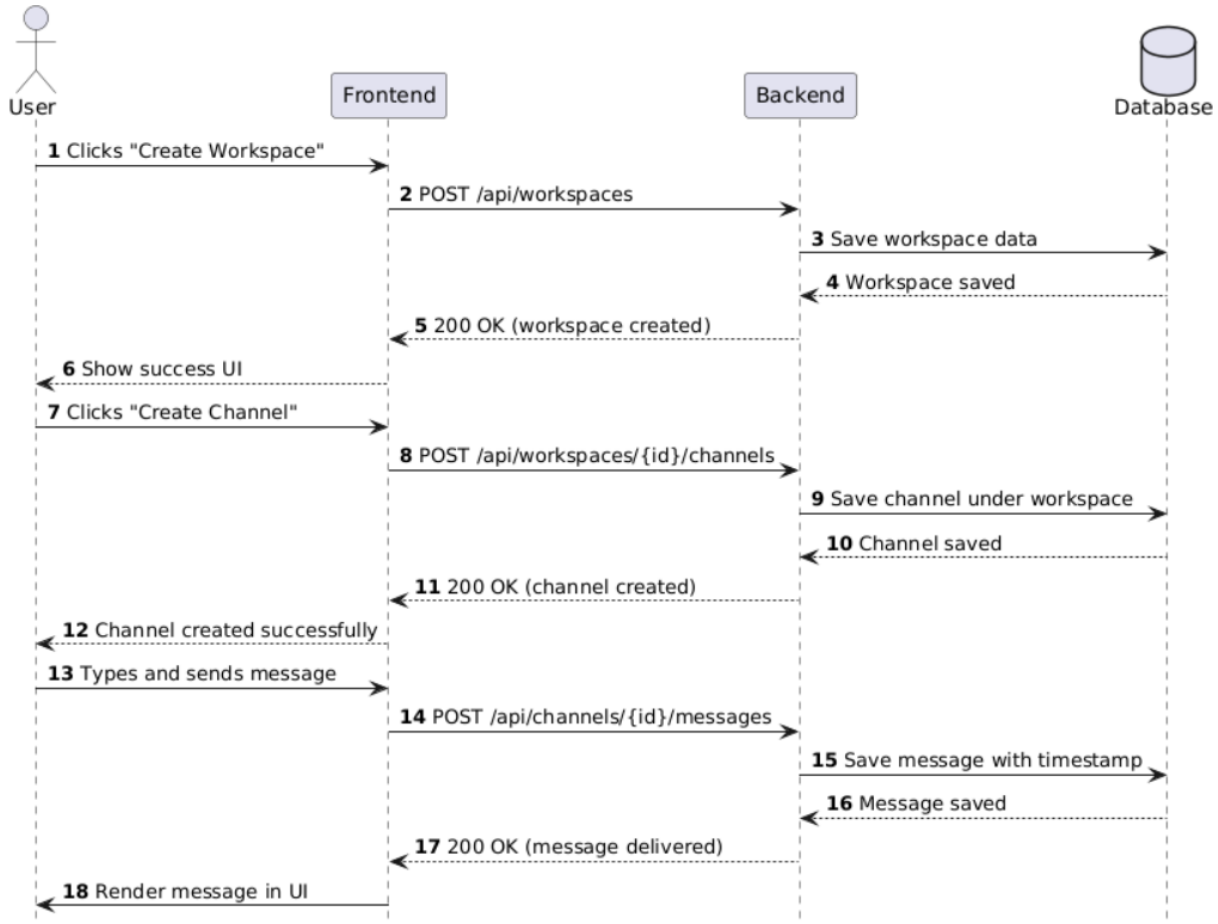

Figure 3.3 Sequence Diagram

Chapter 3
Software Requirements Specifications

3.1 Overall Description

3.1.1 Product Perspective

Icollab is a browser-based software application built using the MERN stack, consisting of MongoDB, Express.js, React.js, and Node.js. It functions as a centralized team collaboration hub offering various tools for communication, task tracking, and project management. The system is modular and supports seamless integration with third-party APIs and platforms such as OpenAI for extended functionalities like AI-driven message summarization.

The application operates independently but is capable of being embedded within broader enterprise ecosystems through integrations. It is intended for teams of all sizes, from startups to enterprise-level organizations, and is scalable to support a growing number of users, workspaces, and tasks.

3.1.2 Product Functions

Icollab provides a comprehensive suite of functionalities:

- Workspace creation and management with multi-user collaboration
- Channel creation for focused communication streams within workspaces
- Real-time messaging using WebSockets for instant communication between team members
- Task management system with options to create, assign, prioritize, and track task statuses

- Video calling and screen sharing support via WebRTC for real-time meetings
- AI-powered chat summarization that helps users quickly catch up on long conversations.

3.1.3 User Classes and Characteristics

The system is designed to cater to different user classes:

- **Admin:** Responsible for managing workspaces and assigning user roles.
- **Team Members:** Have access to the workspace and can participate in channels, communicate with peers, upload and download documents, join meetings, and manage their assigned tasks. Team members have a simplified interface focused on productivity and communication.
- **Developers (System Maintainers):** Handle the maintenance of the backend, database, and deployment. They also ensure continuous integration of APIs, real-time communication stability, and security enforcement.

3.1.4 Operating Environment

Icollab is a web application accessible through modern web browsers. It is developed to be responsive and cross-platform compatible.

- Supported Browsers: Google Chrome, Mozilla Firefox, Microsoft Edge, Apple Safari
- Supported Platforms: Windows, macOS, Linux, Android (mobile browser), iOS (mobile browser)
- Hosting: Cloud-based deployment on platforms such as Render or Vercel

3.1.5 Design and Implementation Constraints

- The application must function smoothly on both desktop and mobile resolutions
- Requires stable internet connection for all real-time and API-dependent operations
- Integration with third-party APIs (WebRTC, Firebase, OpenAI) introduces dependency and must be securely managed
- Session management must be implemented using JWTs
- Deployment must support HTTPS and SSL for secure communication

3.1.6 User Documentation

To ensure effective use of the system, the following documentation will be provided:

- User manuals with step-by-step walkthroughs
- FAQ and troubleshooting section
- In-app help tooltips and tutorials
- Admin guide for managing workspaces and monitoring activity

3.1.7 Assumptions and Dependencies

- The system assumes users have basic familiarity with collaborative tools (cha, task tracking)
- Reliable internet connectivity is necessary for all core functions

3.2 System Features

3.2.1 Workspace and Channel Creation

Users can create dedicated workspaces and organize communication into public or private channels. Channels help segment discussions based on project teams, departments, or tasks.

3.2.2 Real-time Messaging

Built using WebSocket (Socket.IO), the chat system supports live message exchange, typing indicators, message status updates, and file attachments. Messages are stored persistently and searchable by date or keyword.

3.2.3 Video Calling and Conferencing

Utilizing WebRTC, users can initiate secure, peer-to-peer video meetings within channels. The system supports group video calls, screen sharing, mute/unmute features, and room notifications.

3.2.4 Message Summarization

Integrated with OpenAI, the platform supports automatic summarization of chat history. Users can request a summary of discussions by date or channel, enabling faster understanding and decision-making.

3.2.5 Integration with Third-party Tools

The platform supports connecting with:

- **GitHub:** For linking repositories and commits with workspaces

3.3 Functional Requirements

3.3.1 User Authentication

Description: Enables users to securely log in, register, and recover passwords.

- **3.3.1.1 Login**
    - Input: Email and Password

- Output: Access to the dashboard upon successful login.

- **3.3.1.2 Registration**
  - Input: Name, Email, Password
  - Output: Account created successfully.

3.3.2 Workspace Management

Description: Users can create or join workspaces for their teams.

- **3.3.2.1 Create Workspace**
  - Input: Workspace name, description.
  - Output: Workspace successfully created and listed on the dashboard.

3.3.3 Channel Management

Description: Workspaces consist of channels where members collaborate.

- **3.3.3.1 Create Channel**
  - Input: Channel name, type (public/private).
  - Output: Channel created successfully.

3.3.4 Messaging System

Description: Real-time chat with options for file sharing and reactions.

- **3.3.4.1 Send Message**
  - Input: Message text or file upload.
  - Output: Message displayed in the chat.

3.3.5 Video Calling

Description: Supports one-on-one and group video calls using WebRTC.

- **3.3.5.1 Initiate Call**
    - Input: Select participants.
    - Output: Video call session initiated.

3.3.6 Integration with Third-Party Apps

Description: Provides integrations with tools like GitHub, Gmail

- **3.3.6.1 GitHub Integration**
    - **Input**: Repository link, API key.
    - **Output**: Commits and issues visible in the workspace.
- **3.3.6.2 Gmail Integration**
    - **Input**: Google Account authentication.
    - **Output**: Workspace events synced with Gmail.

3.3.7 Message Summarization

Description:Message summarization provides a concise overview of lengthy conversations in a channel or workspace.

- **3.3.7.1 Summarize Channel Conversations**
    - **Input**: Chat of particular Channel
    - **Output**:A brief summary of the conversations.

3.4. Other Non-Functional Requirements

3.4.1 Performance Requirements

- The platform must deliver consistently high performance, ensuring fast and responsive interactions for all users across functionalities.

- Real-time actions such as opening channels, switching workspaces, sending or receiving messages, and rendering user interfaces must occur with a maximum delay of 1 second.
- Task management operations such as task creation, status updates, and priority filtering should execute instantly with minimal latency.
- Database operations, including user data retrieval, channel information, and task assignments, should be completed within 500ms for optimal user experience.
- Chat summarization for channels with high message volumes (e.g., over 1000 messages) should return concise summaries within 3–5 seconds after the request is submitted.
- The system must be able to concurrently support a minimum of 200 users per workspace without degradation in response time.

3.4.2 Safety Requirements

- Only users with admin privileges are allowed to manage or access sensitive system configurations, user permissions, and backend database queries.
- Authentication using JWT tokens is required before users can access protected features, including viewing private messages, accessing restricted workspaces, or initiating video calls.
- All data, including messages, task records, and user credentials, must be encrypted both during transmission and at rest to ensure security.
- Session timeouts and token expiration policies must be enforced to avoid unauthorized access from idle sessions.
- In future paid versions, any failed or interrupted payment processes must be handled with a transaction rollback or secure refund/reversal mechanism to maintain financial trust and avoid inconsistencies.

3.4.3 Security Requirements

- All communication between client and server must be secured via HTTPS protocol to prevent packet sniffing or man-in-the-middle (MITM) attacks.
- Passwords should be hashed using industry-standard hashing algorithms (e.g., bcrypt) before being stored in the database.

- JWT tokens must include expiry timestamps and be refreshed regularly to ensure session security.
- Access control must be enforced throughout the system using middleware to prevent privilege escalation or unauthorized operations.
- Rate limiting and brute-force protection must be implemented to guard against login abuse or denial-of-service attempts.

3.4.4 Software Quality Attributes

- **Usability:** User interface should follow intuitive design principles and offer clear feedback during all operations (e.g., loading indicators, success/error toasts).
- **Portability:** The application should run on all operating systems supporting major browsers without requiring additional plugins.
- **Browser Compatibility:** Verified compatibility with modern browsers including Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.
- **Mobile Responsiveness:** Layouts should automatically adapt to different screen sizes and resolutions for seamless use across smartphones and tablets running Android and iOS.
- **Maintainability:** Codebase must follow modular architecture with reusable components and detailed documentation for ease of debugging and future feature enhancements.

3.4.5 Business Rules

- All proprietary content including the name "Icollab," its logo, frontend designs, and backend logic is protected under intellectual property rights and must not be replicated or redistributed without permission.
- Users are required to maintain professional behavior in public channels, and admins reserve the right to restrict or suspend access based on misuse.
- Compliance with local and international data privacy regulations such as GDPR will be maintained in user data collection and storage policies.

Chapter 4
Database Design

4.1 Schema

4.1.1 User Schema

The User schema is designed to store essential information about each individual using the Icollab platform. It includes fields for authentication, identification, workspace memberships, and session management.

```
{
 firstName: { type: String, required: true },
 lastName: { type: String, required: true },
 email: { type: String, required: true, unique: true },
 password: { type: String, required: true },
 avatar: { type: String },
 status: { type: String, default: "online" },
 workspaces: [{ type: mongoose.Schema.Types.ObjectId, ref:
'Workspace' }],
 role: { type: String, enum: ['admin', 'member'], default: 'member' },
 lastSeen: { type: Date },
 googleId: { type: String },
 authType: { type: String, enum: ['local', 'google'], default: 'local' }
}
```

Description:

- firstName, lastName, email, and password are the basic identity and login details.
- avatar stores the profile image URL.
- status tracks the online/offline state of a user.
- workspaces is an array referencing all workspaces the user is part of.
- role defines access level (admin or member).
- lastSeen records the last activity timestamp.
- googleId and authType allow Google-based login integration.

4.1.2Workspace Schema

The Workspace schema stores information about a collaborative group of users, including its members, channels, messages, and notifications.

```
{
 name: { type: String, required: true },
 description: { type: String, required: true },
 createdBy: { type: mongoose.Schema.Types.ObjectId, ref: 'User',
required: true },
 members: [
  {
   userId: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
   role: { type: String, enum: ['admin', 'editor', 'viewer'], default:
'viewer' }
  }
 ],
 chat: {
  channels: [
   {
    name: String,
    description: String,
    members: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
    messages: [
     {
      sender: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
      content: String,
      timestamp: { type: Date, default: Date.now }
     }
    ]
   }
  ]
 },
 notifications: [
  {
   type: { type: String, enum: ['info', 'alert', 'task'] },
   content: String,
```

```
    priority: { type: String, enum: ['high', 'medium', 'low'], default:
'medium' },
    timestamp: { type: Date, default: Date.now }
  }
 ]
}
```

Description:

- name and description define the workspace identity.
- createdBy links the workspace to its creator.
- members contains users with assigned roles (admin, editor, viewer).
- chat.channels contains a list of communication channels and their details.
- Each channel contains messages posted by users, each storing sender, content, and timestamp.
- notifications stores system updates with type, content, priority, and time.

4.1.3 Videocall Schema

The Videocall schema stores metadata related to live video sessions within channels.

```
{
 host: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
 participants: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
 channel: { type: mongoose.Schema.Types.ObjectId, ref: 'Channel' },
 startTime: { type: Date, default: Date.now },
 endTime: { type: Date },
 status: { type: String, enum: ['active', 'ended'], default: 'active' }
}
```

Description:

- host references the user who initiated the call.
- participants holds all user IDs that joined the call.
- channel links the call to a specific communication channel.
- startTime and endTime track the duration of the video session.
- status indicates whether the call is currently ongoing or has ended.

Chapter 5
Implementation Details

5.1 Modules Created and Brief Description

5.1.1 Authentication Module

The authentication module is responsible for managing the secure login and registration of users. It utilizes industry-standard JSON Web Tokens (JWT) for session handling and authentication purposes. The module supports both local and third-party login options, including Google OAuth and email-based OTP verification, which improves convenience and security. Once a user is authenticated, the JWT token is stored on the client side to maintain an active session across different routes. Access to protected functionalities such as creating a workspace, sending messages, uploading files, and joining video calls is restricted to logged-in users only. This module also includes features like password encryption using bcrypt, token expiration handling, and secure logout operations.

5.1.2 User Management Module

This module focuses on managing user data and access levels within the platform. It provides functionalities for users to register, log in, update their profile details (such as name, avatar, and password), and manage their preferences. It also provides admin users the ability to assign or modify roles (admin, member, viewer). The module ensures that user sessions are preserved using JWT tokens stored in local storage, allowing for persistent login sessions until manually logged out. The profile page gives each user visibility into their activities, workspaces joined, and chat/channel involvement, ensuring a personalized and organized experience.

5.1.3 Workspace and Channel Management Module

The Workspace and Channel Management module provides structural organization to the collaboration process. Users can create new workspaces, define descriptions, and invite or add members. Each workspace can further have multiple public or private channels for categorizing discussions and tasks. Channels help in segmenting communication based on teams, departments, or specific projects. Admin users have access to workspace-level settings including member management, role assignments, and workspace/channel deletion. This module ensures modular and permission-based access to communication channels and is essential for organizing large collaborative environments efficiently.

5.1.4 Task Management Module

The task management module is an essential part of project planning and execution. It allows users, particularly those with admin or editor roles, to create tasks and assign them to specific team members. Each task includes attributes such as title, description, due date, priority (High, Medium, Low), and current status (To-Do, In Progress, Completed). Tasks are visualized using a Kanban-style task board for each workspace or channel, allowing users to monitor progress at a glance. Users receive real-time notifications for tasks assigned to them, and updates to task status are synchronized instantly across all participants.

5.1.5 Chat and Messaging Module

The chat and messaging module forms the real-time communication backbone of the platform. Powered by Socket.IO, it enables users within the same channel to exchange instant text messages, share documents, and react to discussions. Features include message delivery status, typing indicators, emojis, and timestamps. It supports advanced filtering options, including keyword search and date-based filtering. Messages are persistently stored in the database, and access is controlled based on channel

membership and workspace roles. This module ensures high-speed, low-latency communication, and seamless collaboration.

5.1.6 Video Chat Module

This module adds real-time video and audio calling functionality using WebRTC (Web Real-Time Communication). It allows users within a channel to initiate and join video calls either one-on-one or in groups. The video module includes support for screen sharing, mute/unmute toggles, camera switching, and room notifications. Socket.IO is used for the signaling mechanism required to establish peer-to-peer connections between users. The system automatically manages call setup, ICE candidate exchange, and session teardown. The module aims to replace the need for third-party video tools and centralize communication within the platform.

5.1.7 Chat Summarization Module

This module enhances productivity by summarizing long conversations in chat channels. Users can request summaries for specific channels and date ranges. The summarization output is displayed in a readable paragraph format, which helps users who join late or missed conversations to quickly understand what was discussed. The module reduces cognitive load and supports better decision-making by filtering out non-essential information.

5.2 Function Prototype

5.2.1 registerUser(req, res)

Parameters:

- req: An object containing the user details such as firstName, lastName, email, and password submitted through the registration form.
- res: The response object used to send the registration result back to the client.

Return Type: Promise

Description: This function is responsible for registering a new user in the system. It first validates the input data, hashes the password using the bcrypt hashing algorithm for enhanced security, and saves the user information into the database. After a successful entry, it generates a JWT token to maintain the session and returns it to the user. This token is used for all subsequent authenticated API requests.

5.2.2 loginUser(req, res)

Parameters:

- req: An object that includes the login credentials of the user, specifically email and password.
- res: The response object to send back the result of the login attempt.

Return Type: Promise

Description: This function authenticates a user by validating the provided email and password against the stored hashed credentials in the database. Upon successful validation, a JWT token is issued to the user for maintaining a secure session. If credentials are invalid, appropriate error messages are returned. The token can be used to access protected endpoints like task management, messaging, or video calls.

5.2.3 createWorkspace(req, res)

Parameters:

- req: Contains workspace-related data including the name, description, and a list of members to be added.
- res: The response object used to deliver the result of the workspace creation operation.

Return Type: Promise

Description: This function enables a user (typically with admin privileges) to create a new workspace. It stores the workspace metadata such as its name, description, and creator's ID. After creation, it links the selected members to the workspace with their assigned roles. The new workspace is then accessible to all added members and available under their workspace listings.

5.2.4 assignTask(req, res)

Parameters:

- req: Contains task information including title, description, priority, deadline, and the assignedTo user ID.
- res: The response object used to send back the task assignment status.

Return Type: Promise

Description: This function allows users with the right permissions (admin or editor) to assign tasks to members within a workspace. The task is created in the database with relevant fields and is linked to the appropriate workspace and assignee. The system also triggers a notification to the assigned user. It supports Kanban-style task categorization and is updated in real-time for all involved members.

5.2.5 sendMessage(req, res)

Parameters:

- req: Contains information such as the sender ID, channel ID, and the message content to be stored and distributed.
- res: The response object for returning the message status or any errors.

Return Type: Promise

Description: This function is part of the real-time messaging module. When a user sends a message in a channel, the message is saved in the MongoDB collection and simultaneously emitted to all other channel participants using Socket.IO. This allows users to see the message appear in real-time without refreshing the page. The message data includes the timestamp and sender metadata for reference and logging.

5.2.6 joinVideoRoom(req, res)

Parameters:

- req: Includes room ID and the user ID of the participant trying to join the video session.
- res: The response object to send back room join status.

Return Type: Promise

Description: This function facilitates the process of joining a live video meeting. It manages the peer-to-peer connection setup using WebRTC and handles the exchange of signaling data using Socket.IO. When a user joins a room, it notifies other participants and initializes audio/video streams. It also handles reconnections and edge cases like network failure or participant leave events.

5.2.7 summarizeChat(req, res)

Parameters:

- req: Contains the target channel ID and a selected date range to summarize messages.
- res: The response object used to send back the generated summary or error details.

Return Type: Promise

Description: This function extracts messages from the specified channel and date range, formats them into a conversation string.The API processes the data and returns a concise, human-readable summary capturing the key points of the discussion. This summary is then sent to the requesting user. The feature is useful for users who missed meetings or long chat sessions and need a quick update.

Chapter 6
Testing

6.1 Testing Framework and Methodology:

- **Unit Testing**: Frontend forms, navigation, and responsiveness across multiple browsers.
- **Endpoint Testing**: Postman for API endpoints.

Table 6.1 (Test Cases)

Test Case ID	Description	Test Steps	Input	Expected Output	Actual Output	Status
TC01	Verify user registration and login	1. Go to registration page. 2. Enter name, email, password. 3. Click register. 4. Log in using credentials.	Name, Email, Password	User account created and login successfully	User registered and logged in successfully	Pass

TC02	Create and join workspace	1. Login as user. 2. Click "Create Workspace". 3. Enter name and invite members.	Workspace name, emails	Workspace created and accessible	Workspace created and user joined	Pass
TC03	Add and assign a task	1. Login as user. 2. Go to Tasks. 3. Click "Add Task". 4. Fill in details.	Task title, priority, description	Task appears in assignee's dashboard	Task visible in member's view	Pass
TC04	Test real-time messaging	1. Login as two users in same channel. 2. One user sends message.	Text message	Message instantly delivered to other user	Message received live	Pass

TC05	Join video call	1. Click "Join Call" in a channel. 2. Allow camera access. 3. Connect with another user.	Video room ID	Video call established	Users connected on call	Pass
TC06	Generate chat summary	1. Login. 2. Select channel and date range. 3. Click "Summarize Chat".	Channel ID, Date Range	Summary generated and displayed	AI-generated summary shown	Pass
TC07	Unauthorized access restriction	1. Try accessing workspace without login.	N/A	Redirect to login page	Access denied, redirected	Pass
TC08	Profile update	1. Login. 2. Go to profile. 3. Edit name or photo and save.	New name/profile image	Updated info saved	Profile updated successfully	Pass

TC09	Logout functionality	1. Login. 2. Click on profile menu. 3. Click logout.	N/A	Redirected to login screen	User logged out and redirected	Pass

Chapter 7
Screenshots

- **User registration and login page**

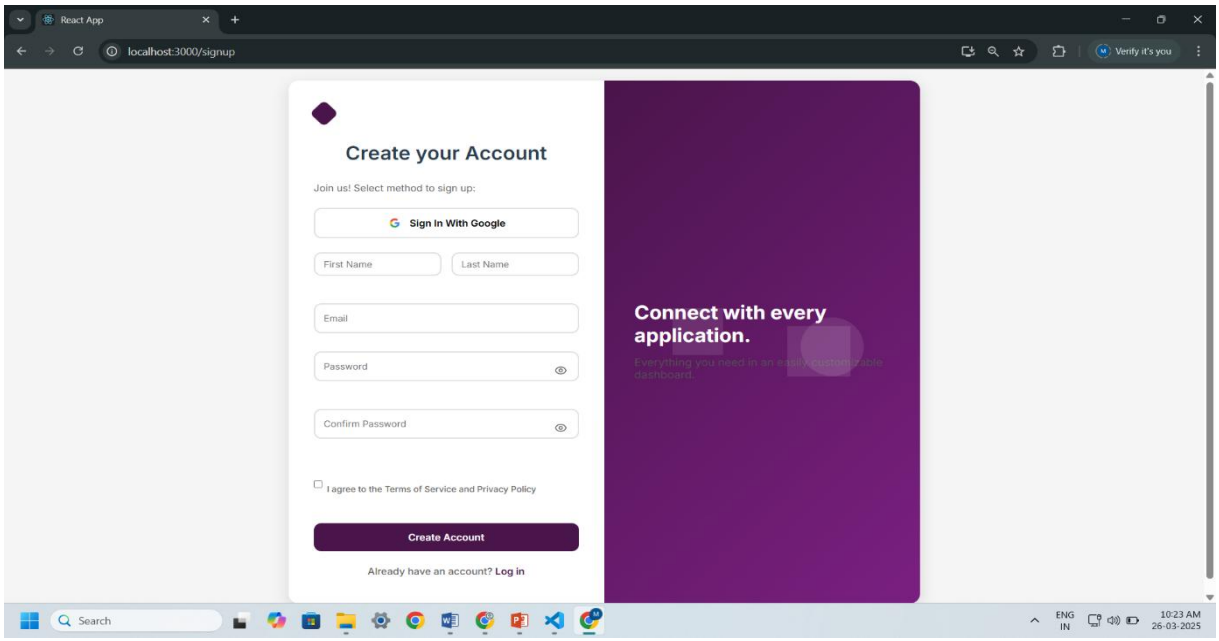

Figure 7.1 (Signup page)

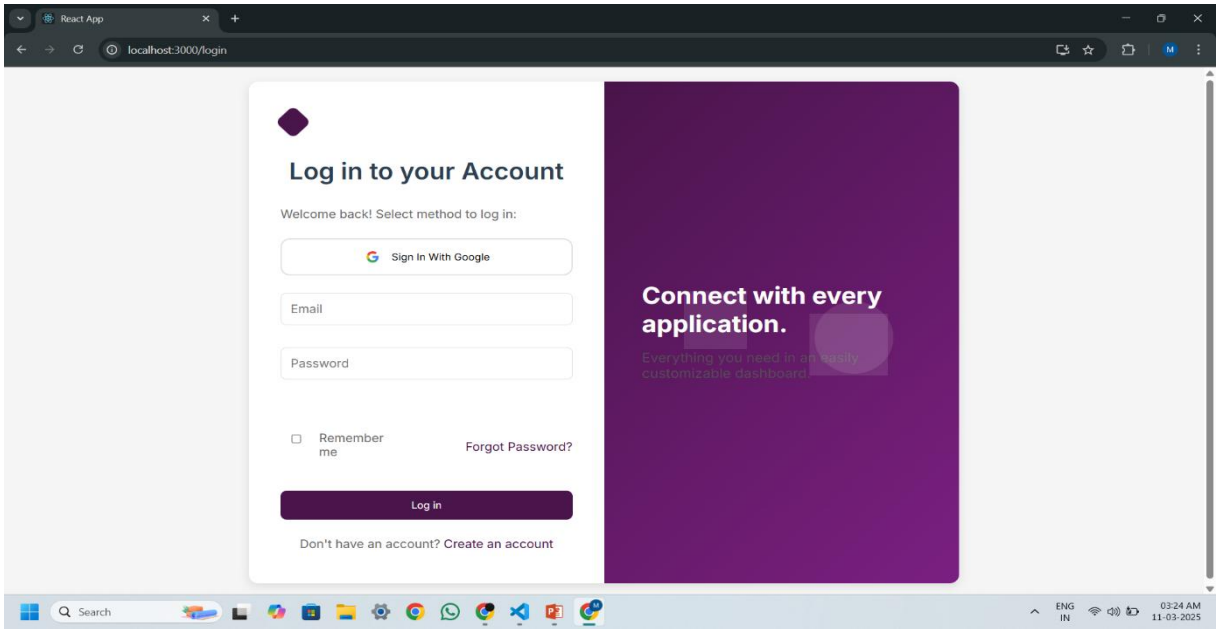

Figure 7.2 (Login page)

- **Home page**

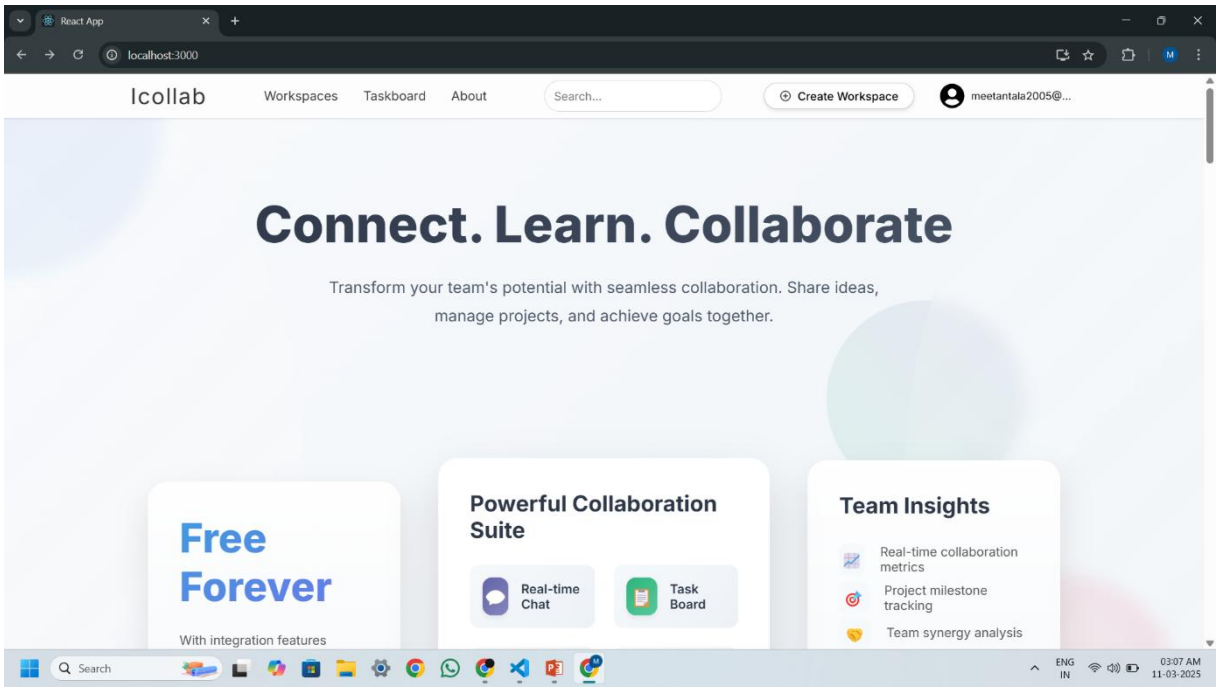

Figure 7.3 (Home page)

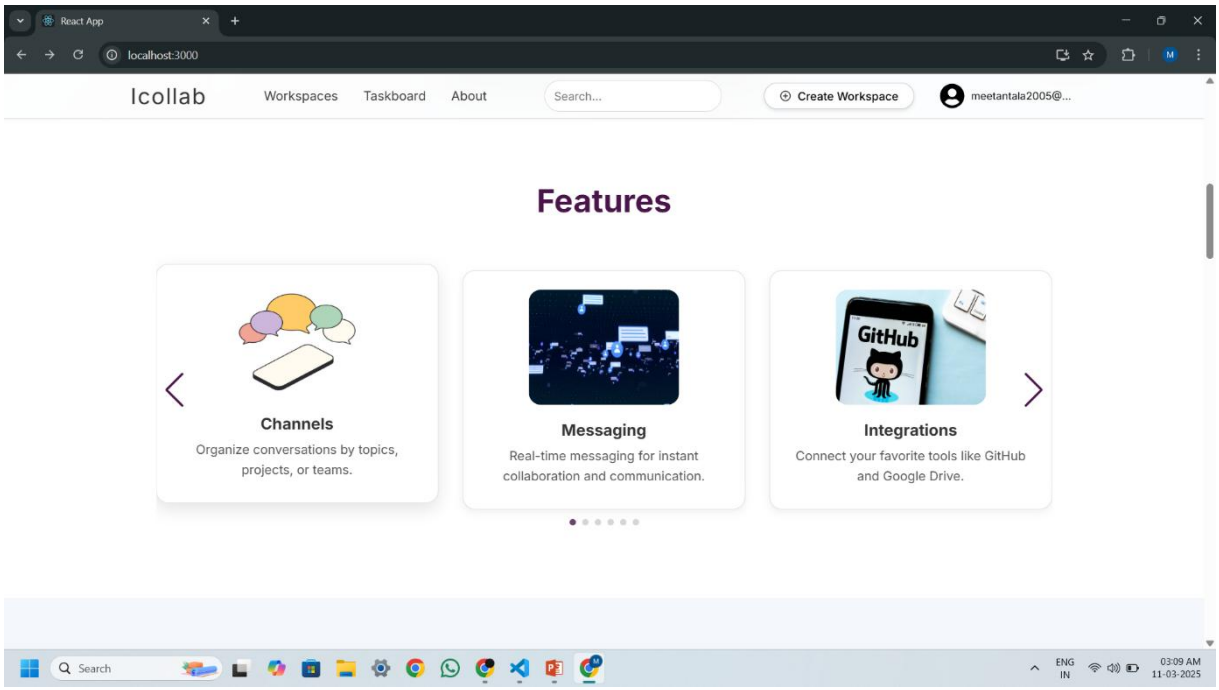

Figure 7.4 (Home page)

- **Workspace and Channel pages**

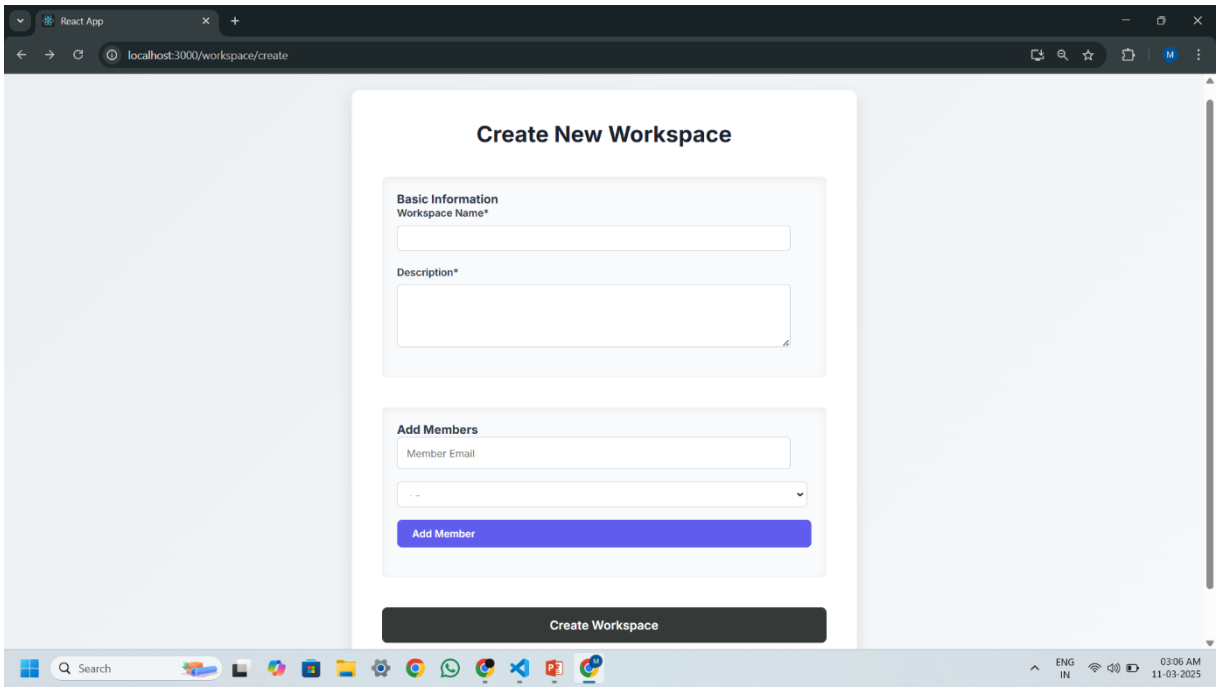

Figure 7.5 (create new Workspace Page)

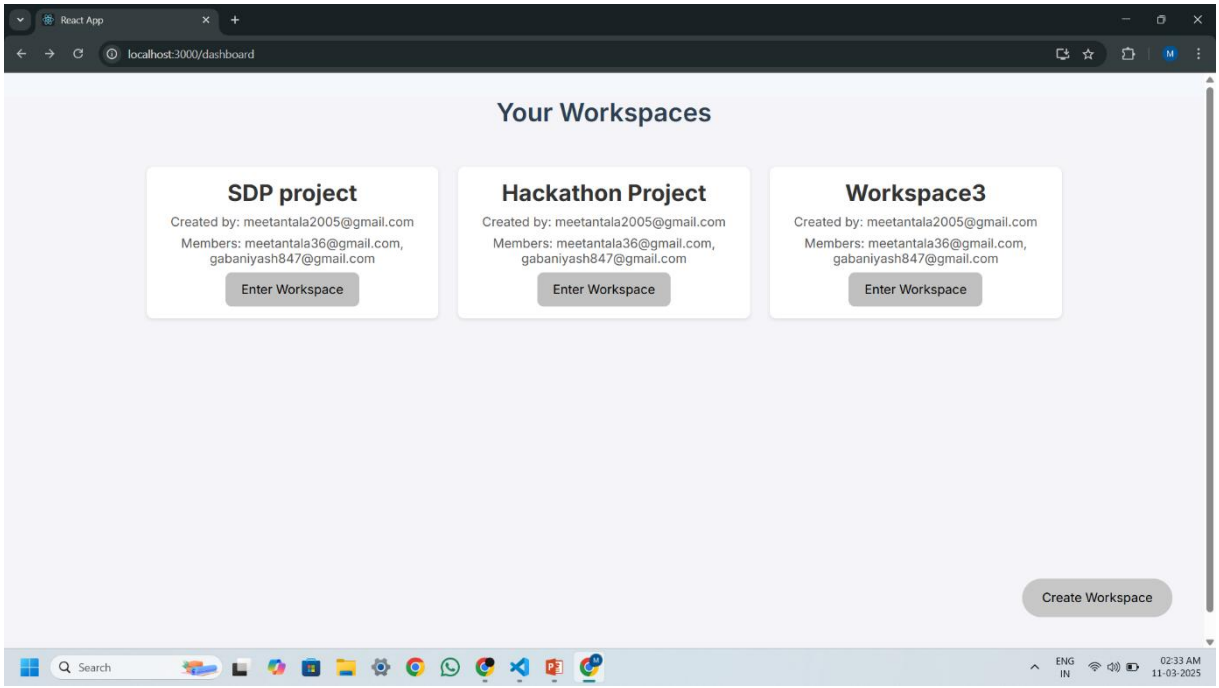

Figure 7.6 (Workspace Dashboard)

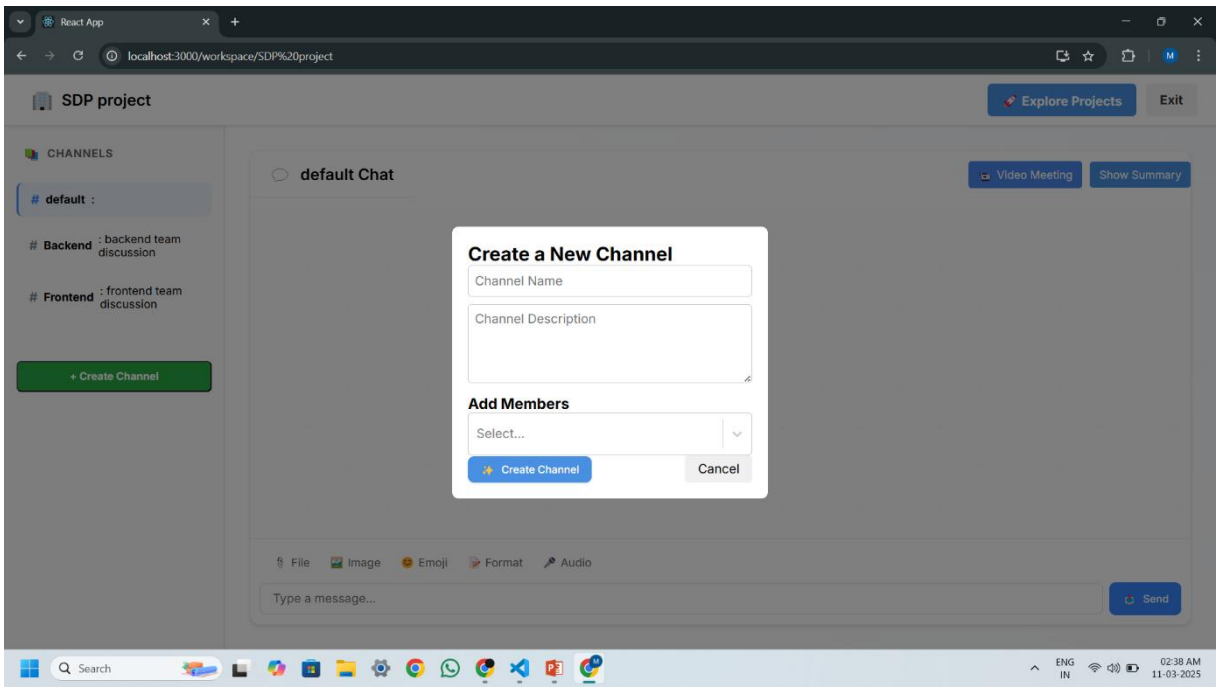

Figure 7.7 (Create New Channel page)

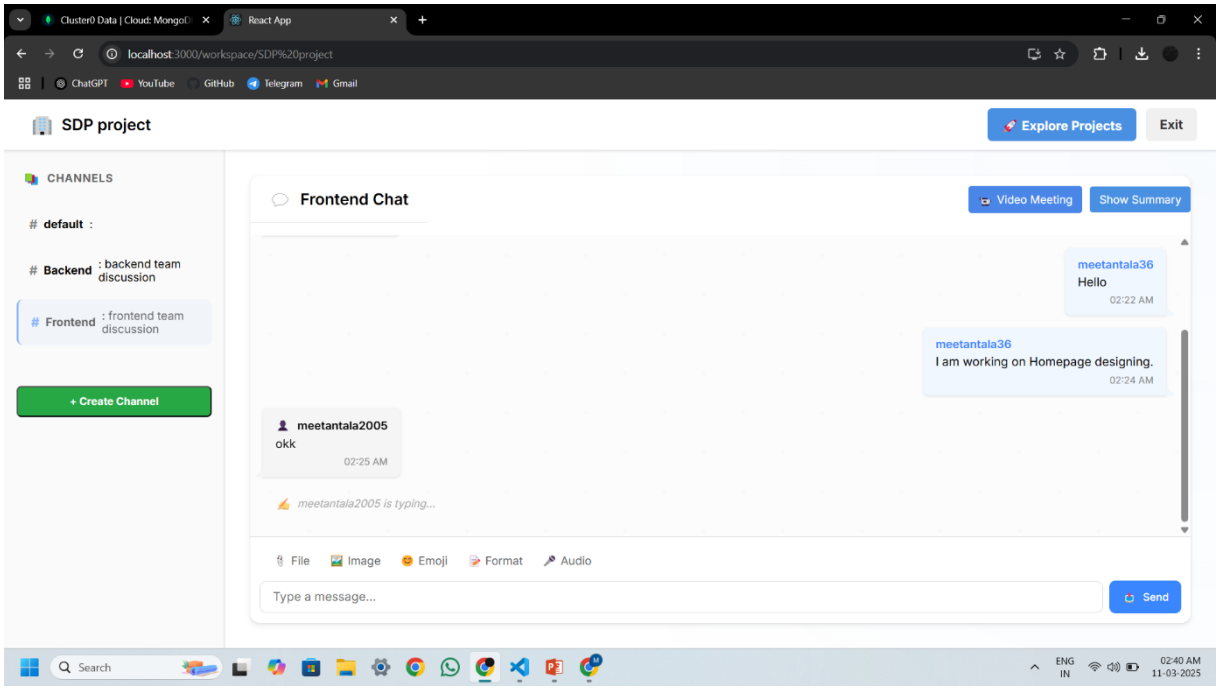

Figure 7.8 (Channel Chating Interface)

- **Task mangement pages**

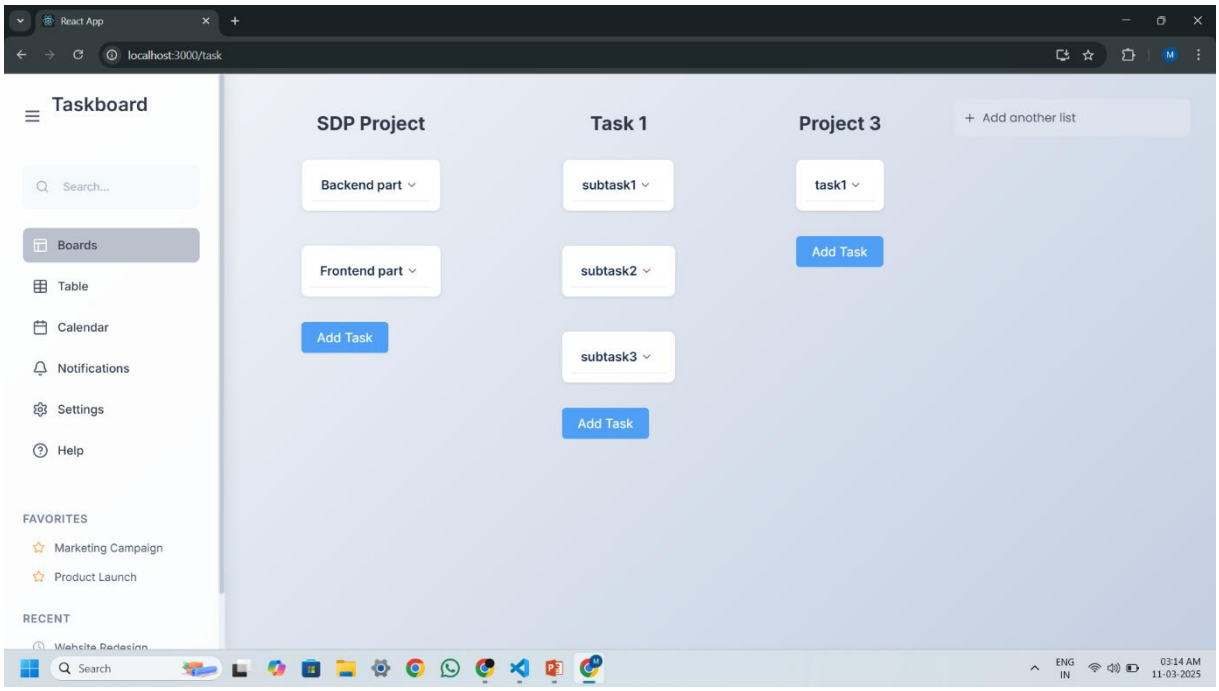

Figure 7.9 (Task Management page)

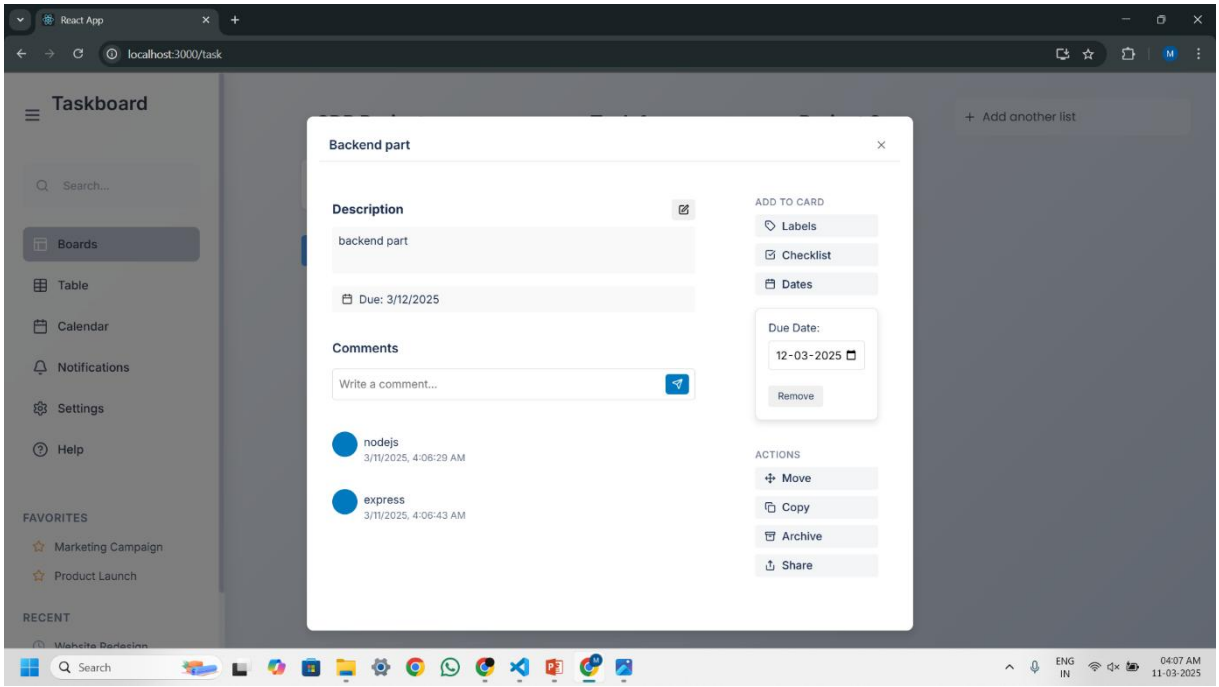

Figure 7.10 (Task management features)

- **Video call page**

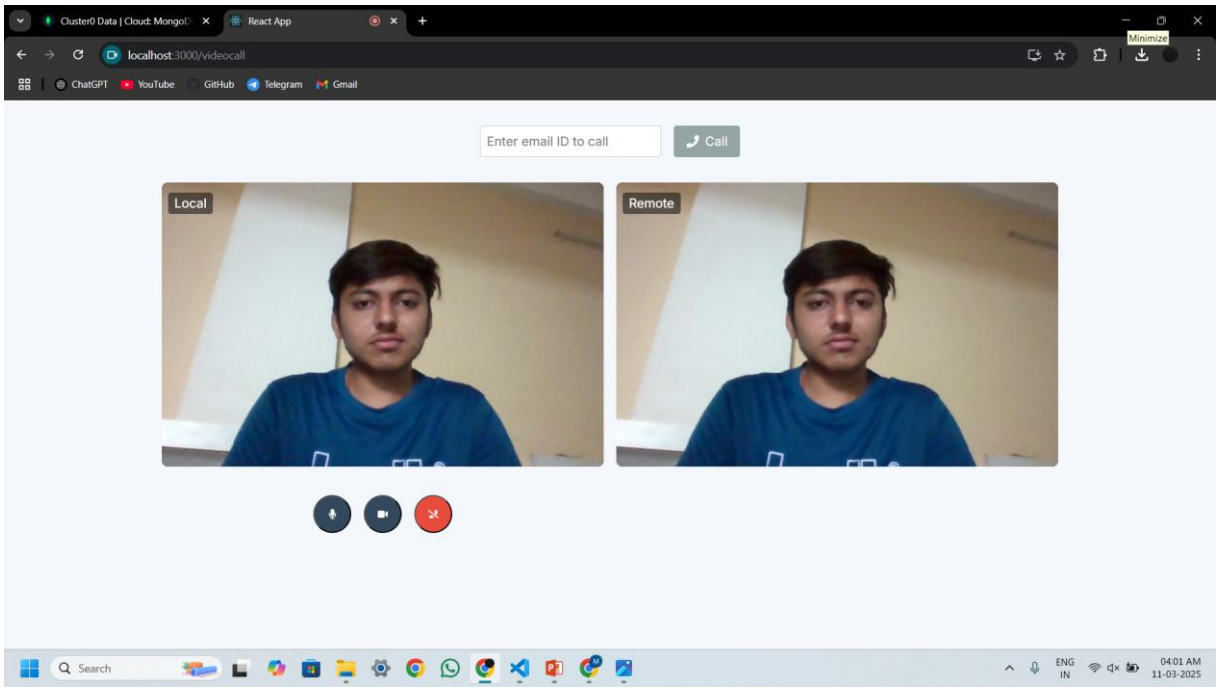

Figure 7.11 (Video call Interface)

Chapter 8
Conclusion

- The Icollab platform delivers a robust and scalable solution for modern teams seeking an integrated workspace for communication, task management, and collaboration.
- Role-based access control ensures secure, personalized access to core functionalities like messaging, task assignment and workspace customization.
- The system incorporates secure JWT-based authentication and also provides seamless login options through Google OAuth enhancing accessibility and security.
- Real-time messaging built on WebSockets, alongside live video conferencing powered by WebRTC, allows teams to collaborate effectively without needing third-party communication tools.
- The task management module streamlines workflow by allowing team leads to assign tasks, set deadlines, and track progress using visual task boards categorized by status and priority.
- AI-powered chat summarization helps users quickly catch up on channel conversations, improving team productivity and reducing information overload.
- All system features including chat, task management, and video meetings have been thoroughly tested across different browsers and devices to ensure reliability and performance.
- Icollab ultimately serves as a unified platform where teams can plan, discuss, assign, and deliver—making it highly valuable for startups, remote teams, and enterprise collaboration workflows.

Chapter 9
Limitations and future extension

9.1 Limitations

- **Web-Only Platform:** Currently, Icollab is designed exclusively for web browsers. It does not yet offer native mobile applications, which limits accessibility and convenience for users who rely heavily on mobile devices for collaboration and communication.

- **Manual AI Summarization:** The chat summarization feature, although powered by AI, requires manual user input to generate summaries for specific channels and date ranges. This makes it less efficient for ongoing or automatic summarization across workspaces.

- **Lack of Offline Functionality:** Icollab is entirely cloud-based and dependent on internet connectivity. Users are unable to access previously loaded data or perform essential operations such as viewing messages or tasks without a live connection.

- **Limited Role Management:** The system currently supports only two roles—Admin and Member. This lacks flexibility for organizations with hierarchical team structures. There is no provision for custom roles like Project Manager, Viewer, or Moderator with tailored permissions.

- **Basic Notification System:** Notifications are limited to in-app alerts and do not yet support advanced options like scheduling, user

preferences, or cross-platform delivery (e.g., email or push notifications).

9.2 Future Extensions

- **Mobile Application Development:** One of the top priorities is to develop fully functional mobile applications for Android and iOS platforms. This will ensure users can access all features—chat, tasks, video calls, and file uploads—on the go.

- **Advanced Role Management:** The platform will introduce customizable user roles with granular permissions. Roles such as Moderator, Project Lead, Viewer, and Contributor will be configurable based on team needs.

- **Automated AI Features:** AI functionalities will be expanded to include auto-summarization of conversations, intelligent task suggestions, deadline prediction, and sentiment analysis to improve team productivity.

- **Enhanced Workspace Customization:** Future updates will allow users to personalize their workspace themes, add branding elements (e.g., logos and custom colors), and configure notification preferences for different events.

- **Integrated Analytics Dashboard:** A powerful admin analytics dashboard will be introduced to provide insights into team activity, task completion rates, active/inactive users, most used channels, and communication trends.

- **Calendar and Reminder System:** A built-in calendar will help users schedule meetings, deadlines, and task milestones. Users will receive timely reminders via notifications or email to stay updated on critical events.

These planned enhancements are aimed at making Icollab a highly customizable, intelligent, and fully integrated team collaboration suite that evolves with the growing needs of its users.

Chapter 10
Bibliography

MERN Stack Documentation

- MongoDB: MongoDB Official Documentation
- Express.js: Express.js Official Documentation
- React.js: React.js Official Documentation
- Node.js: Node.js Official Documentation