**Full Stack Development with MERN – Project**

**FINAL REPORT**

**Project Title**:

MERN VIDEO CONFERENCE APP- VIDEOCON

**Team Members**:

Hardik Mishra (22BCE10556)

Samarth Gupta (22BCE11394)

Sujal Patel(22BAI10409)

Yogendra Singh Rathod(22BCE11378)

# 1. INTRODUCTION

## In an era where remote work, online education, and virtual collaboration have become essential, the need for reliable and efficient video communication tools has never been more critical. Traditional means of conducting meetings and events in physical spaces are rapidly giving way to digital alternatives, especially for businesses, educators, students, and remote teams. However, many existing video conferencing platforms come with limitations such as high costs, complex user interfaces, lack of customization, and concerns over privacy and data security. These issues can create barriers to smooth and effective communication.

## 1.1 Project Overview

## To address these challenges, VIDEOCON, a robust web-based video conferencing application built using the MERN stack (MongoDB, Express.js, React.js, and Node.js), emerges as a powerful solution aimed at transforming how people connect virtually. VIDEOCON provides a secure and seamless communication experience, offering real-time video and audio conferencing capabilities, screen sharing, chat functionalities, and more—all accessible through an intuitive and user-friendly interface.

## Designed for diverse user groups including corporate teams, educators, students, and families, VIDEOCON enables users to create and join virtual meetings effortlessly. Hosts can manage participants, share content securely, and schedule sessions with calendar integration. The app also supports user authentication, encrypted communication, and customizable meeting settings to ensure both functionality and security.

## VIDEOCON bridges the gap between convenience and reliability by eliminating the common pain points associated with other conferencing tools. It promotes collaboration regardless of geographic boundaries, fosters productivity, and ensures that users stay connected in a secure and efficient manner.

## The primary goal of VIDEOCON is to deliver a high-quality, accessible, and privacy-focused video communication solution. By combining modern technologies with thoughtful design, the application aspires to redefine digital communication and make virtual interactions as natural and effective as in-person meetings.

## 1.2 Purpose

he primary purpose of **VIDEOCON** is to revolutionize the way individuals and organizations conduct virtual meetings, classes, and collaborative sessions by providing a centralized, secure, and user-friendly platform. In a time when communication is increasingly digital, VIDEOMEET aims to streamline the process and address key pain points present in existing solutions.

Specifically, the objectives of VIDEOCON include:

* **Unified Video Communication Platform**: To offer a single platform for hosting, joining, and managing video conferences without relying on multiple tools or services.
* **Broker-Free and Cost-Effective Solution**: To provide an alternative to expensive or subscription-based video conferencing tools, allowing users to interact directly and freely without intermediaries or hidden fees.
* **Secure and Verified User Access**: To ensure user privacy and data protection through authentication protocols and end-to-end encrypted communication, fostering trust and safety during meetings.
* **Real-Time Interaction and Smart Features**: To enable seamless real-time communication with added functionalities such as chat, screen sharing, recording options, and customizable meeting controls to enhance collaboration.
* **Enhanced User Experience and Accessibility**: To create an intuitive interface accessible to users of all technical skill levels, and to support smooth, glitch-free video sessions even in limited bandwidth environments.

By meeting these goals, VIDEOCON aims to offer a next-generation video conferencing solution that supports productivity, security, and user satisfaction. It seeks to empower individuals and teams across domains with a digital space that makes communication more effective, engaging, and trustworthy.

# 2. IDEATION PHASE

## 2.1 Problem Statement

**Customer Problem Statement :**

A customer problem statement helps the team stay focused on solving real-world issues users face while hosting or attending online meetings and virtual sessions. These statements were formed by empathizing with the target users—students, remote workers, educators, and professionals—and analyzing their behaviors, challenges, and frustrations.

In today’s digital-first world, efficient and secure video communication is essential, yet many users encounter barriers with existing video conferencing platforms. These include lack of data privacy, difficult interfaces, high subscription fees, limited control features, and compatibility issues across devices and networks. Whether it’s students attending online classes, employees working remotely, or businesses conducting virtual meetings, the need for a reliable and user-friendly solution is more urgent than ever.

This project aims to develop a modern and secure video conferencing platform – **VIDEOMEET** – using the MERN stack (MongoDB, Express.js, React.js, Node.js). The application will centralize real-time video communication tools, simplify meeting creation and participation, and ensure a seamless and trustworthy virtual experience.

**Key Problems Identified:**

* **Complex User Interfaces**: Many existing apps have overwhelming or non-intuitive designs, especially for non-tech-savvy users**.**
* **Security and Privacy Concerns:** Users are increasingly worried about unauthorized access, data breaches, and privacy violations during video calls.
* **Lack of Customization and Control:** Meeting hosts often have limited options for managing participants or customizing sessions according to their needs**.**
* **Limited Real-Time Features:** Tools such as screen sharing, chat, and session recording are often limited or locked behind paywalls**.**
* **High Costs:** Subscription-based models make professional-grade video conferencing inaccessible to smaller teams, students, or individuals**.**
* **Poor Performance in Low Bandwidth:** Many platforms fail to deliver a stable experience in poor network conditions, leading to user dissatisfaction**.**

**Our Goal:**

To build a **secure, scalable, and intuitive video conferencing web** application that addresses these challenges, offering smooth communication, real-time collaboration, and enhanced control for both hosts and participants..

| **Problem Statement ID** | **I am (Customer)** | **I’m trying to** | **But** | **Because** | **Which makes me feel** |
| --- | --- | --- | --- | --- | --- |
| PS-1 | A college student attending remote classes | Join and participate in online lectures smoothly | The platform lags and crashes during sessions | It’s not optimized for low-bandwidth environments | Frustrated and disconnected from learning |
| PS-2 | A teacher managing virtual classrooms | Conduct interactive and controlled sessions | I can't manage participants or share content easily | The platform lacks educational features and proper moderation tools | Powerless and ineffective as an educator |
| PS-3 | A remote worker attending client meetings | Use a professional-grade video conferencing app | Most apps are expensive or require software installations | There is no budget-friendly, browser-based alternative | Annoyed and professionally limited |
| PS-4 | A small business owner hosting team calls | Collaborate remotely with my team securely | The app has security flaws and confusing interfaces | Trust and ease of use are missing | Skeptical and worried about data safety |

**Key Takeaways from Customer Problem Statements**

* Students and educators need **stable, interactive, and low-bandwidth-friendly solutions..**
* Working professionals and teams want **a secure, cost-effective, and intuitive platform.**
* Users demand real-time communication features with **privacy, customization, and ease of access.**
* Emotional pain points **include frustration, helplessness, skepticism, and a sense of being professionally handicapped.**

## 2.2 Empathy Map Canvas

**Empathy Map Canvas:**

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviours and attitudes.

It is a useful tool to helps teams better understand their users.

Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user’s perspective along with his or her goals and challenges.

An **Empathy Map** is a collaborative visualization tool used to articulate what we know about a particular user type. It helps teams understand and align around the user’s perspective by breaking down their behaviors, thoughts, feelings, and needs. It is often used during the **ideation and discovery phase** of product development to build **user-centered solutions**.

Empathy maps help answer questions such as:

* What is the user experiencing?
* What are they thinking and feeling?
* What are their challenges and motivations?
* How can we best solve their problems?

**Application to VIDEOMEET (Video Conference App)**

In the context of your **video conferencing platform**, the **Empathy Map** becomes a vital UX tool to understand the needs, frustrations, and expectations of different users—such as remote workers, educators, students, business owners, and team leads—throughout their experience of hosting or attending virtual meetings.

## ****Why is the Empathy Map Important for**** VIDEOCON****?****

* Helps your **development team build features** that address real user concerns
* Aids in creating a **more intuitive and user-friendly UI/UX**
* Guides your **content and communication strategy** (e.g., what messages users respond to)
* Ensures your solution is **empathetic, practical, and user-centric**, not just technically sound



## ****User Needs (for**** Meeting Participants****)****

## These are the essential desires or features users expect from a video conferencing platform:

## One-click meeting joining (without downloads or complex setup)

## High-quality audio and video

## Screen sharing and presentation tools

## Real-time chat during meetings

## Meeting recording and cloud storage

## Cross-device support (web/mobile/tablet)

## Private meeting rooms or password-protected sessions

## Minimal lag and smooth performance

## Notification and calendar integration (e.g., Google Calendar)

## Easy invite links

## ****Pain Points (for**** Meeting Participants****)****

These represent the challenges or frustrations that users commonly face with current platforms.

 Laggy or pixelated video in low-bandwidth environments

 Complicated or bloated interfaces

 No support for browser-based meetings (requiring app download)

 Poor mobile or tablet support

 Security concerns (e.g., Zoombombing)

 Unclear audio or lack of noise suppression

 Confusing navigation to access tools like screen sharing or chat

 No easy way to record meetings or revisit them later

## 2.3 Brainstorming

**Brainstorm & Idea Prioritization :**

The ideation phase formed the backbone of the **Video Conference App Project**. It included identifying real-world issues in remote communication, listing out potential features, and then prioritizing them for development within our project constraints.

**Goal of the Project**

Create a **Video Conferencing Application using the MERN Stack (MongoDB, Express.js, React.js, Node.js)** that allows users to:

* Host and join real-time video meetings with ease
* Share screens, chat, and record sessions
* Securely schedule and manage meetings
* Access a clean, responsive, and performance-optimized interface

**Step-1: Team Gathering, Collaboration and Select the Problem Statement**

**Objective:**

To bring together individuals with diverse skill sets, align on a shared vision, and finalize a real-world problem to address using the MERN stack.

**Actions Taken:**

**Team Formation**: A balanced team was formed consisting of frontend developers, backend developers, and database designers—all passionate about solving real-life problems through web development.

**Skill Mapping**:

* Frontend: React.js, Bootstrap, HTML/CSS
* Backend: Node.js, Express.js
* Database: MongoDB

**Collaboration Tools Used**:

* Communication: WhatsApp, Google Meet
* Project Planning: Trello, Google Docs

**Problem Identification:**

We identified a recurring problem with complex or unreliable video meeting tools, especially for users seeking lightweight, easy-to-use platforms for quick team catch-ups or virtual classes.

**1. Team Formation and Role Distribution**

The foundation of any successful project lies in assembling a well-balanced and dedicated team. For our project titled **VIDEOCON– A Video Conference Application Using MERN Stack**, we began by identifying team members with strong enthusiasm for web development, practical problem-solving, and interest in building a real-world application from scratch.

Our team comprises four committed members. Each member was assigned a specific role based on their individual skill set, domain knowledge, and preferences. This structured approach ensured a clear division of responsibilities, accountability, and efficient collaboration throughout the development lifecycle.

| **Team Member** | **Assigned Role** | **Key Responsibilities** |
| --- | --- | --- |
| Member 1 | **Project Lead & Full Stack Developer** | Coordinating project timelines, managing GitHub repository, integrating frontend and backend, deployment. |
| Member 2 | **Frontend Developer** | Designing responsive user interfaces using React.js, managing UI/UX flow, integrating API data, using Tailwind CSS for styling |
| Member 3 | **Backend Developer** | Developing RESTful APIs using Node.js and Express.js, implementing authentication and authorization (JWT), business logic |
| Member 4 | **Database Manager & QA Tester** | Designing MongoDB database schemas, managing data models, writing queries, performing manual testing and reporting bugs |

We adopted a collaborative model that allowed flexibility and learning opportunities. Members often cross-collaborated across modules to support one another, strengthening teamwork and overall output.

**2. Collaboration Tools and Workflow**

To ensure seamless communication and progress tracking, especially in a hybrid working environment, we utilized a range of collaboration tools. These helped us stay aligned with project goals, deadlines, and updates:

**Git & GitHub**: For version control, collaborative code reviews, branching strategies, and pull requests.

**Google Meet / Whatsapp** : For regular virtual meetings, daily/weekly stand-ups, and technical discussions.

**Google Drive / Docs**: For storing and sharing important project-related documents, reports, and references.

We followed a simplified agile approach with iterative development and continuous integration. Weekly check-ins helped us identify blockers early, track progress, and maintain momentum.

**3. Brainstorming and Problem Selection**

Our initial goal was to identify a real-world problem that could be solved effectively using the MERN stack. We conducted several brainstorming sessions where each team member proposed different ideas from areas such as:

* Online collaboration and communication platforms
* Virtual classrooms and education tools
* Healthcare appointment and consultation systems
* Event planning and coordination apps
* Remote work productivity and time-tracking tools

After evaluating these ideas based on **feasibility, scope, technical complexity, innovation, and real-world relevance**, we collectively agreed upon building a **Video Conference App** as the most promising and impactful project.

**4. Why We Chose This Problem**

The decision to work on a **Video Conference Application** was driven by the following key factors:

**High Practical Relevance**: In the current digital age—especially post-pandemic—video conferencing has become an essential mode of communication for remote teams, educators, students, freelancers, and professionals. Our app aims to support seamless virtual meetings without the complexities or pricing models of existing heavy platforms.

**Market Gap:** While there are major players like Zoom, Google Meet, and Microsoft Teams, many users (especially small teams, educational groups, and startups) seek simpler, lightweight, and more customizable video conferencing tools. Our solution targets that underserved segment by offering a user-friendly, web-based platform with real-time communication and collaborative features.

**Technology Fit**: The project was an excellent match for the MERN stack as it involves:

* Real-time communication using **WebRTC and Socket.IO**
* User authentication and session management
* CRUD operations for managing meetings and user profiles
* RESTful API development
* MongoDB-based data storage for users, meetings, and chat logs

**Scalability and Innovation Potential**: The application holds great potential for future expansion and innovative features, such as:

* Screen sharing and recording capabilities
* Collaborative whiteboard or note-taking tools
* Meeting scheduling with calendar integration
* Breakout rooms and group chat
* AI-based noise suppression or background blur
* Mobile app version using React Native

**5. Final Problem Statement**

**"To design and develop a responsive, secure, and scalable web application using the MERN stack that enables users to host and join real-time video conferences, with features such as user authentication, meeting scheduling, and live chat—thereby providing a reliable and accessible platform for remote communication and collaboration."**

This problem not only allowed us to apply our technical knowledge in full-stack development and real-time technologies, but also inspired us to build a practical solution that can be deployed and scaled in real-world educational, corporate, and personal use cases.

**Step-2: Brainstorm, Idea Listing and Grouping**

Once the team was formed and the problem statement was finalized, the next crucial step in the project development process was to brainstorm different ideas related to the solution, list those ideas collaboratively, and group them into meaningful categories. This step helped us visualize the complete scope of the application, prioritize features, and streamline the development process.

**1. Purpose of the Brainstorming Session**

The goal of the brainstorming session was to:

* Understand what functionalities users would expect from a VideoCon platform.
* Identify all possible features that can enhance user experience and platform efficiency.
* Break down the problem into smaller manageable modules for better planning and development.
* Align everyone's vision regarding the project scope and deliverables.

**2. Brainstorming Methodology**

We conducted a series of brainstorming sessions over virtual meetings using tools like:

* **Miro** for real-time collaborative whiteboarding and mind mapping.
* **Google Docs** for live note-taking and idea capturing.
* **Trello** for organizing features into categories and prioritizing tasks.

Each team member contributed ideas based on:

* Personal experiences with tools like Zoom, Google Meet, Microsoft Teams, etc.
* Observations of pain points users face during online meetings.
* Feedback and expectations from students, professionals, and educators who rely on video conferencing.

We used a **Round-Robin format** where every team member was encouraged to present one or more ideas at a time. No idea was considered irrelevant or dismissed immediately — everything was listed for review.

**3. Idea Listing**

Below is the comprehensive list of ideas/features that emerged during the brainstorming phase:

**Core Features:**

 User registration/login (with roles: host, participant)

 Create and schedule video meetings

 Join meetings via link or meeting ID

 Real-time video and audio streaming

 Screen sharing capability

 Chat functionality during meetings

 Mute/unmute and video on/off controls

 Meeting recording (cloud/local)

 Responsive user dashboard for managing meetings and joining sessions

**Extended Features:**

 Calendar integration (Google Calendar, Outlook) for scheduling

 Email and in-app notifications for upcoming meetings

 Host controls (remove participant, mute all, lock meeting)

 Meeting waiting room and password protection

 Custom meeting backgrounds or blur feature

 Participant reactions (raise hand, emojis, etc.)

**Future Enhancement Ideas:**

 Breakout rooms for group discussions

 AI-based noise cancellation

 Integration with cloud storage platforms (Google Drive, Dropbox) for saving recordings

 Analytics dashboard for admins (meeting duration, participant count, etc.)

 Multi-language support and live captioning

 Mobile app version using React Native for on-the-go access

**4. Grouping of Ideas**

After listing all possible ideas, we organized them into logical **feature groups** for better clarity and to ease project execution. This step was critical for defining the Minimum Viable Product (MVP) and planning development sprints.

| | **Category** | **Features Grouped** | | --- | --- | | | **Category** | **Features Grouped** | | --- | --- | |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | **User Management** | | |  |  | | --- | --- | |  | Registration, Login, Logout, Profile Management, Role Management (Host/Participant) | |
| |  |  | | --- | --- | | **Meeting Management** |  | | |  |  | | --- | --- | |  | Create Meeting, Join via Link or ID, Schedule Meeting, Dashboard for Meeting Control | |
| |  |  | | --- | --- | | **Communication** |  | | |  | | --- | | Video/Audio Streaming, Screen Sharing, Chat, Mute/Unmute, Reactions | |
| |  |  | | --- | --- | | **Security** |  | | |  |  | | --- | --- | |  | JWT Authentication, Password Encryption, Meeting Lock, Waiting Room, Role-based Access | |
| |  |  | | --- | --- | | **Notifications** |  | | |  | | --- | | Email Alerts, In-App Notifications, Calendar Reminders | |
| |  |  | | --- | --- | | **Admin Operations** |  | | |  |  | | --- | --- | |  | User Monitoring, Meeting Logs, Access Control, Reporting Tools | |
| |  |  | | --- | --- | | **Advanced Features** |  | | |  |  | | --- | --- | |  | Meeting Recording, Background Effects, Breakout Rooms, Live Captions, Analytics Dashboard, Mobile App (Future) | |

**5. Key Takeaways from This Phase**

We successfully transformed a broad problem statement into well-defined, actionable modules.  
• **Prioritized core functionalities** that are essential for the initial version of the video conference application, ensuring a seamless user experience.  
• **Documented advanced features** for future versions, such as breakout rooms, live captions, and integration with third-party tools.  
• Ensured everyone in the team had a shared understanding of the project’s goals and how we would break down tasks across upcoming development sprints.  
• Established a strong foundation for **Requirement Analysis**, **System Architecture Design**, and **Sprint Planning** for the next phases of development.

**Step-3: Idea Prioritization**

After brainstorming and organizing a comprehensive list of ideas and features in Step 2, the next logical step in our development process was **prioritizing these ideas**. This stage was essential to identify which features to develop first (Minimum Viable Product – MVP), which ones to add later (Post-MVP or V2), and which ones to consider as long-term enhancements or stretch goals.

* Effective idea prioritization ensured that:
* The project remained manageable within the given timeline and resources.
* Core user needs were addressed from the start.
* The development process followed a clear, goal-driven roadmap.

**1. Prioritization Approach**

To determine what features should be prioritized for development, we adopted a combination of two popular techniques:

**a. MoSCoW Method**

We classified each idea into four categories:

**Must Have** – Critical features required for the app to function.

**Should Have** – Important features that enhance user experience but are not critical for MVP.

**Could Have** – Nice-to-have features that can be added if time/resources allow.

**Won’t Have (for now)** – Features we decided to postpone or not include in this version.

**b. Value vs. Effort Matrix**

Each feature was analyzed based on:

**Value** to the end-users (usability, necessity, impact)

**Effort** required to implement (time, complexity, team skill)

This allowed us to balance our work between quick wins, high-impact features, and manageable complexities.

**2. Prioritized Feature List (with Justifications)**

| **Feature** | **Priority** | **Justification** |
| --- | --- | --- |
| User Registration and Login | Must Have | Basic entry point for all users; essential for access control |
| JWT Authentication | Must Have | Secures APIs and protects user data |
| Video Conference Room Creation | Must Have | Core feature; allows users to create and host virtual meetings |
| oin Conference (with Link) | Must Have | Core feature for participants to join meetings using a unique URL/link |
| Audio/Video Streaming | Must Have | Essential for the main purpose of the app — enabling real-time communication |
| Chat Functionality | Must Have | Facilitates real-time text communication alongside video/audio |
| Notifications (Meeting Invites) | Must Have | Ensures users receive reminders/notifications for scheduled meetings |
| Admin Dashboard | Should Have | Helps manage and monitor meetings, users, and app settings |
| Breakout Rooms) | Should Have | Useful for larger meetings or team collaborations within a single conference (future feature) |
| Meeting Recording | Could Have | Adds value for future versions; useful for later reference |
| Multi-Factor Authentication | Could Have | Enhances security for real-world applications, but not essential for MVP |
| Calendar Integration | Could Have | AddSyncs with user calendars to schedule meetings more easily (good for future versions)s extra security; useful in real-world scenarios |
| Custom Branding | Won’t Have | Requires additional design work; not within the scope of the initial version |
| Payment Gateway Integration | Won’t Have | Not needed for the MVP; requires time and effort to implement within the current timeline |

**3. Final MVP Feature Set**

Based on the prioritization, the following features were locked in as MVP (Minimum Viable Product) for the Video Conference App:

* **User Registration & Login (Host/Participant)**  
  Allows users to create an account, log in, and manage their profiles.
* **JWT Authentication**  
  Secures API endpoints and ensures user authentication for private meetings.
* **Real-time Video/Audio Conferencing**  
  Core functionality that enables seamless video and audio communication between users.
* **Meeting Scheduling & Notifications**  
  Users can schedule meetings, and automated notifications are sent to participants.
* **Screen Sharing**  
  Allows users to share their screen during meetings for collaboration.
* **Chat Functionality**  
  Real-time messaging during meetings for communication apart from video/audio.
* **Participant Management**  
  Features like muting/unmuting, removing participants, and managing host controls.

**4. Post-MVP Planning**

We also created a Phase 2 backlog to record "Should Have" and "Could Have" features, which could be implemented after the initial launch if time permits. This list was added to Trello for tracking and sprint planning purposes. These features include:

* **Admin Panel**  
  Allows administrators to monitor meetings, manage users, and resolve issues.
* **Recording Functionality**  
  Enables meeting recording for later reference and sharing.
* **Breakout Rooms**  
  Facilitates smaller group discussions within larger meetings.
* **Meeting Transcriptions**  
  Provides automatic transcriptions of meetings for reference and accessibility.
* **Mobile App Version**  
  Expands access to the platform on mobile devices using React Native or Flutter.

**5. Outcome of the Prioritization Phase**

* We defined a clear roadmap that separates core functionalities from auxiliary features for the **Video Conference App**.
* The team could now focus on delivering the most impactful and feasible features, such as user authentication, real-time video/audio conferencing, and screen sharing.
* This process helped manage scope and timeline, ensuring that we avoid overengineering and feature creep, keeping the initial launch on track.
* Set the stage for the next phases, including requirement analysis, UI wireframing, and architecture design, which will define the app’s user interface and underlying infrastructure.

# 3. REQUIREMENT ANALYSIS

## 3.1 Customer Journey map

## 

## 3.2 Solution Requirement

**Functional Requirements:**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration via Email Registration via Gmail Registration via Phone Number |
| FR-2 | User Login | Login via Email Login via OTP Remember Me Functionality |
| FR-3 | Video Conference Room Creation | Host can create a room  Set room name, access type (public/private), meeting agenda |
| FR-4 | Join Conference | Users can join via room link or code  Waiting room support  Lobby notifications |
| FR-5 | Screen Sharing & Recording | Share screen (full/window/tab)  Start/Stop meeting recording  Save recordings to cloud/local |
| FR-6 | Chat Functionality | Real-time text chat during call  Emojis & media sharing |
| FR-7 | Audio/Video Controls | Mute/unmute audio  Turn on/off video  Device settings management |
| FR-8 | Notifications | Email/Push notifications for booking status New listings alerts |
| FR-9 | Admin Panel | Manage users Approve/reject listings View reports |
| FR-10 | Scheduling Meetings | Schedule future meetings  Calendar integration  Invite participants via email |

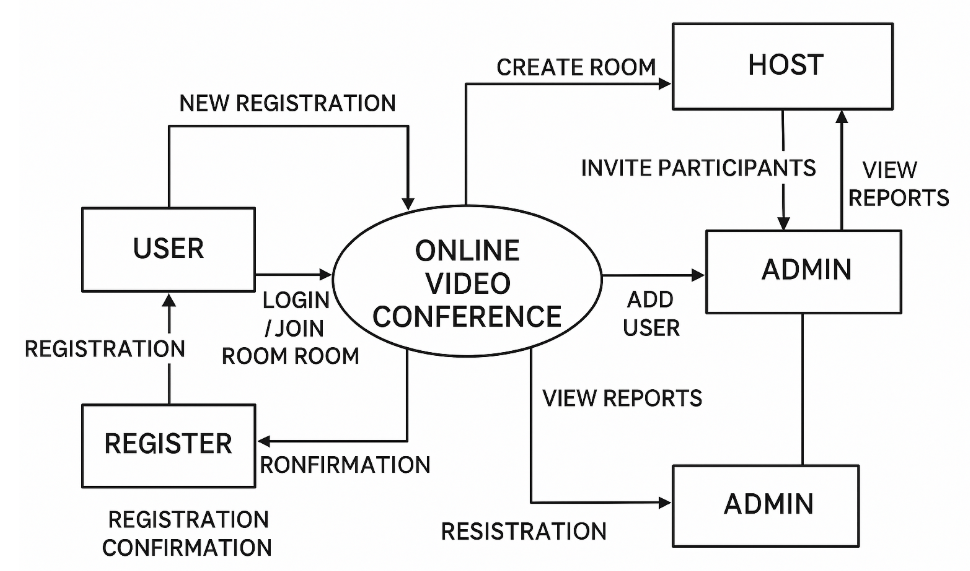
**Non-functional Requirements:**

Following are the non-functional requirements of the proposed solution.

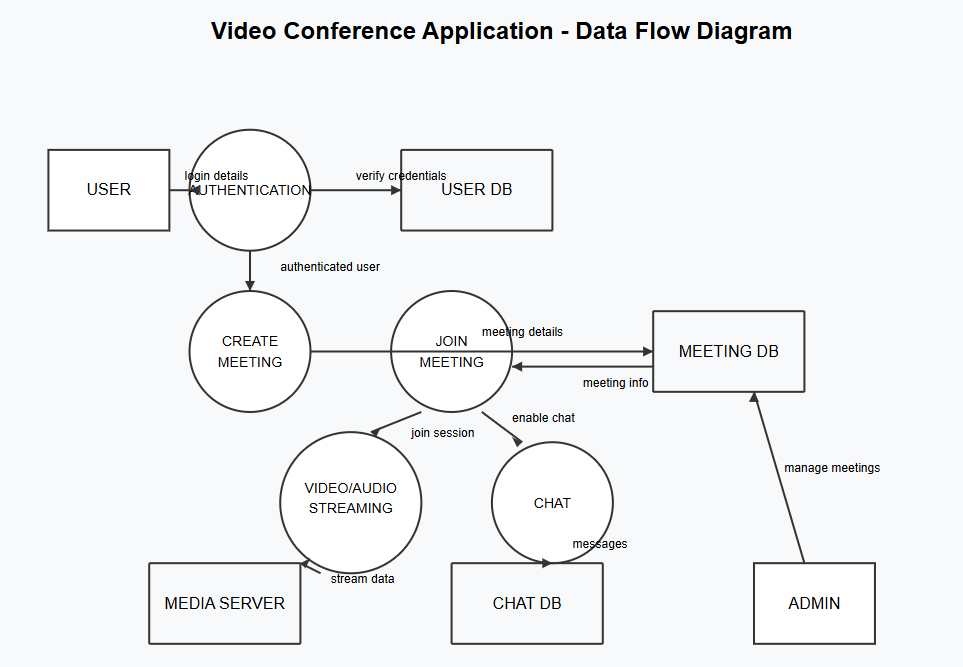
|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | Usability | The UI should be clean, intuitive, and mobile-responsive |
| NFR-2 | Security | Data encryption, secure authentication, and role-based access control |
| NFR-3 | Reliability | Ensure data accuracy and uptime through robust backend and monitoring |
| NFR-4 | Performance | Fast load times (<2 seconds), optimized image loading |
| NFR-5 | Availability | 99.9% uptime, with auto-scaling on traffic surge |
| NFR-6 | Scalability | Support for thousands of concurrent users and listings |

## 3.3 Data Flow Diagram

* **Data Flow Diagrams:**

****

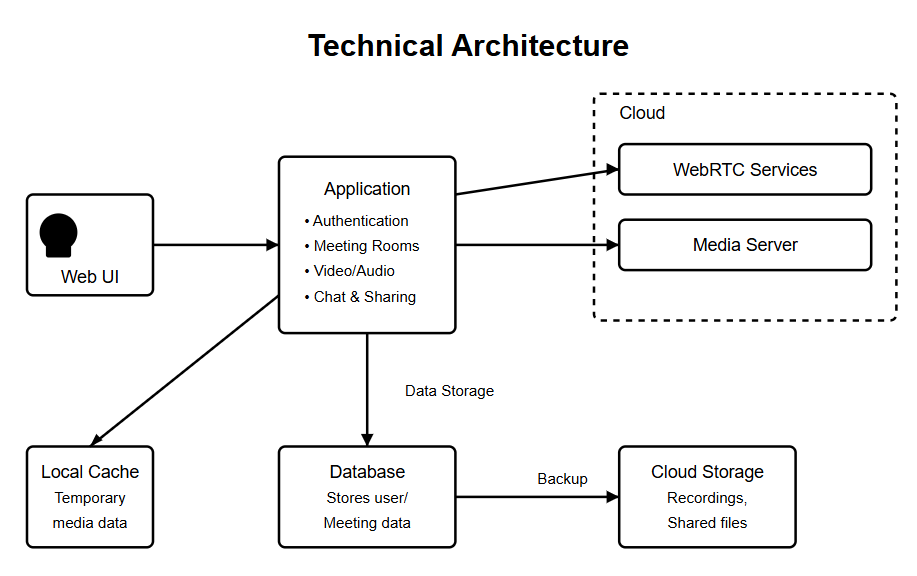
**Level 0 DFD of Video Conference App**

****

**Level 1 DFD of Video Conference App**

## 3.4 Technology Stack

* **Technical Architecture:**

****

**Table-1: Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1 | User Interface | Web UI and responsive Mobile Web | HTML, CSS, JavaScript, React.js |
| 2 | Application Logic-1 | User authentication & registration | Node.js / Express.js |
| 3 | Application Logic-2 | Property search, filtering, and listing | Node.js / Express.js |
| 4 | Application Logic-3 | Video/audio streaming and processing | WebRTC, Socket.io |
| 5 | Database | Stores user info, meeting data, chat history | MongoDB (NoSQL) |
| 6 | Cloud Database | Cloud-hosted version for scaling | MongoDB Atlas |
| 7 | File Storage | Recording storage and shared documents | AWS S3 / Local Filesystem |
| 8 | External API-1 | Calendar integration for scheduling | Google Calendar API |
| 9 | External API-2 | Authentication services | OAuth 2.0 (Google/Facebook) |
| 10 | Machine Learning Model | Background noise suppression and virtual backgrounds | TensorFlow.js |
| 11 | Infrastructure | Cloud deployment of entire application | AWS EC2 / Docker / Kubernetes |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1 | Open-Source Frameworks | React.js, Node.js, Express.js | JavaScript (ME\_N\_ Stack) |
| 2 | Security Implementations | SHA-256 hashing for passwords, JWT tokens, HTTPS, Role-based access control | Bcrypt, JWT, HTTPS, Helmet.js |
| 3 | Scalable Architecture | Supports horizontal scaling, containerized microservices | Docker, Kubernetes, 3-tier architecture |
| 4 | Availability | Load balanced architecture, auto-scaling group | AWS ELB, Auto Scaling |
| 5 | Performance | CDN for static assets, caching for repeated search queries | Cloudflare CDN, Redis for caching |

# 4. PROJECT DESIGN

## 4.1 Problem Solution Fit

**Problem – Solution Fit Template:**

The Problem-Solution Fit demonstrates that we have accurately identified key challenges in remote communication and collaboration, and that our video conferencing application addresses these challenges effectively. This framework helps technology developers identify user needs and behavior patterns to create solutions that gain rapid adoption and provide measurable value.

**Purpose:**

* **Solve complex communication challenges** in ways that align with users' existing workflows and technical capabilities.
* Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
* Sharpen your communication and marketing strategy with the right triggers and messaging.
* **Create multiple value touchpoints** by addressing both urgent communication needs (reliable video calls) and enhancing productivity through integrated collaboration tools.
* **Understand existing communication patterns and limitations.**

**Template:**



| **Section** | **Details (Based on VIDEO CONFERENCE APP project)** |
| --- | --- |
| **1. Customer Segment(s) (CS)** | * Remote working professionals seeking reliable communication tools * Educational institutions (teachers, students) for virtual classrooms * Healthcare providers using telehealth services * Small to medium businesses with distributed teams |
| **2. Jobs-To-Be-Done / Problems (J&P)** | * Difficulty in hosting stable, high-quality video meetings * Lack of integrated collaboration tools within a single platform * No easy way to schedule and manage multiple meetings * Inadequate security measures for sensitive discussions |
| **3. Triggers (TR)** | * Transition to remote/hybrid work models * Need for virtual learning environments * International team collaboration requirements * Dissatisfaction with current video platforms |
| **4. Emotions: Before / After (EM)** | * **Before:** Frustrated with disconnections, anxious about technical issues, overwhelmed by complex interfaces, isolated from team members * **After:** Confident in communication, relieved about reliability, engaged with colleagues, productive in virtual meetings |
| **5. Available Solutions (AS)** | * Mainstream platforms (Zoom, MS Teams, Google Meet) * Enterprise solutions (Webex, GoToMeeting) * Social media video options (Discord, Facebook Rooms) * Open-source alternatives (Jitsi) * **Pros:** Wide adoption, feature-rich * **Cons:** Security concerns, bandwidth issues, feature bloat, subscription costs |
| **6. Customer Constraints (CC)** | * Limited bandwidth in some work environments * Varying technical expertise among users * Budget constraints for premium solutions * Privacy and security compliance requirements * Device and browser compatibility issues |
| **7. Behaviour (BE)** | * Switching between multiple platforms for different meeting needs * Using workarounds for missing features * Scheduling shorter or fewer meetings due to fatigue * Supplementing with messaging apps for quick communication |
| **8. Channels of Behaviour (CH)** | * **8.1 Online** Web browsers * Mobile applications * Desktop applications * Smart TV/conference room systems  **8.2 Offline**  Phone conference bridges as backup * Recorded meeting archives * Downloaded presentation materials |
| **9. Problem Root Cause (RC)** | * Fragmented communication ecosystem requiring multiple tools * Poor optimization for varying network conditions * Complex user interfaces creating barriers to adoption * Inadequate security architecture in existing solutions |
| **10. Your Solution (SL)** | **VIDEO CONFERENCE APP:** A full-stack web application offering:   * Adaptive streaming quality based on available bandwidth |

## 4.2 Proposed Solution

**Proposed Solution : VIDEO CONFERENCE APP**

| **S.No.** | **Parameter** | **Description** |
| --- | --- | --- |
| 1. | **Problem Statement** | Existing video conferencing solutions often suffer from inconsistent quality, complex interfaces, insufficient security, and poor integration with productivity tools, creating barriers to effective remote collaboration and communication. |
| 2. | **Idea / Solution Description** | VIDEO CONFERENCE APP is a full-stack web application that enhances virtual meetings with adaptive streaming technology, end-to-end encryption, intuitive UI/UX, and seamless collaboration tools. Built on modern web technologies, it offers reliable video/audio communication with integrated scheduling, document sharing, and interactive features across devices and network conditions. |
| 3. | **Novelty / Uniqueness** | Unlike conventional platforms, VIDEO CONFERENCE APP features bandwidth-adaptive streaming quality, AI-powered noise cancellation, seamless productivity tool integration, customizable virtual environments, and intelligent meeting insights while maintaining a simple, accessible interface. |
| 4. | **Social Impact / Customer Satisfaction** | The platform democratizes high-quality remote communication, particularly benefiting educational institutions, healthcare providers, and businesses in areas with bandwidth limitations. It reduces technical frustration, increases meeting productivity, and enhances collaboration regardless of user location or technical expertise. |
| 5. | **Business Model (Revenue Model)** | Revenue streams include tiered subscription plans (Free/Pro/Enterprise), add-on premium features (enhanced recording, transcription services), API access for enterprise integration, and white-label solutions for organizational deployment. |
| 6. | **Scalability of the Solution** | The application's microservices architecture ensures high scalability, supporting future expansion to mobile apps, IoT device integration, AI-powered meeting assistants, language translation services, and extended reality (XR) capabilities for immersive collaboration spaces. |

## 4.3 Solution Architecture

**Solution Architecture:**

### **Solution Architecture Overview**

**Solution Architecture** is a strategic process that ensures your technology choices align effectively with business needs. For the ***VIDEO CONFERENCE APP*** project, this approach helps deliver a scalable, secure, and feature-rich communication platform that works reliably across various devices and network conditions.

#### **Purpose of Solution Architecture in** VIDEO CONFERENCE APP

* **Identify Best Tech Solution:** Leverage modern web technologies including WebRTC, Socket.IO, React.js, Node.js, and cloud services to build a responsive, real-time video conferencing application with adaptive streaming capabilities.
* **Communicate with Stakeholders:** Visualize and communicate how users connect and interact through the platform, how media streams are processed, and how the backend infrastructure ensures reliable performance even under challenging network conditions.
* **Define Features & Phases:** Clearly structure the development timeline through sprints—starting from user authentication, basic video/audio streaming, and chat functionality to advanced features like virtual backgrounds, breakout rooms, and integrations with productivity tools.
* **Deliver Specifications:** Provide technical documentation including WebRTC implementation, signaling protocols, media encoding/decoding strategies, security measures for end-to-end encryption, and responsive UI/UX workflows across devices.

### **Key Components of the Solution Architecture**

| **Component** | **Description** |
| --- | --- |
| **Frontend (React.js)** | Delivers an intuitive and responsive UI that adapts to different devices and screen sizes. Implements WebRTC client-side functionality for media capture and display. |
| **Backend (Node.js + Express.js)** | Manages API routing, signaling server functionality, user session management, and meeting coordination. Handles WebRTC signaling for peer connection establishment. |
| **Database (MongoDB)** | Stores user profiles, meeting records, scheduled sessions, chat history, and usage analytics. Implements efficient indexing for quick meeting retrieval. |
| **Authentication** | JWT-based secure login and registration system with multi-factor authentication options. Supports SSO integration with Google, Microsoft, and enterprise identity providers. |
| **Media Processing** | Handles real-time video/audio encoding, bandwidth adaptation, background effects, and noise cancellation using WebRTC and media processing libraries. |
| **Meeting Management** | Enables creation, scheduling, joining, and recording of meetings with access control and permissions management. |
| **Collaboration Tools** | Integrates screen sharing, virtual whiteboard, document collaboration, and polls/surveys functionality for interactive meetings. |
| **Deployment** | Cloud-based deployment with containerization (Docker) and orchestration (Kubernetes) for scalability. CDN integration for optimized global content delivery. |

### **Development Phases**

1. **Sprint 1:** User authentication system, account creation, profile management, and email verification.
2. **Sprint 2:** Core video/audio conferencing functionality, basic UI implementation, WebRTC integration for peer-to-peer connections.
3. **Sprint 3:** Meeting scheduling, calendar integration, chat functionality, and screen sharing capabilities.
4. **Sprint 4:** Advanced features implementation (virtual backgrounds, noise cancellation), collaboration tools (whiteboard, document sharing).
5. **Sprint 5:** Admin panel development, analytics dashboard, meeting recording and storage functionality.
6. **Sprint 6:** Final integration, cross-platform testing, performance optimization, security auditing, and deployment to production environment.

**Example - Solution Architecture Diagram:**

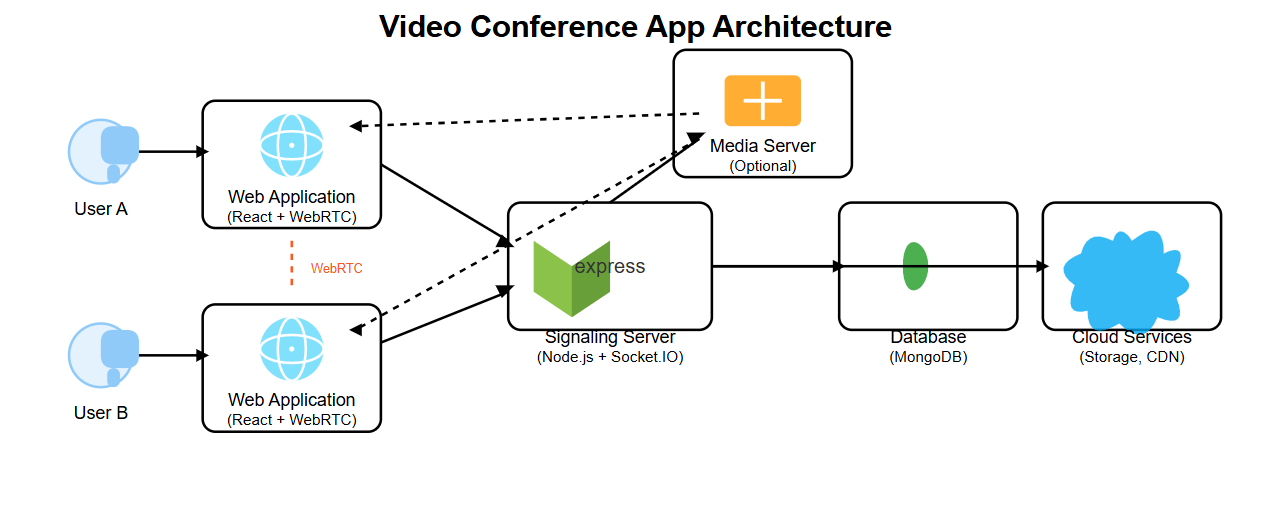
****

Figure 1: Architecture and data flow

# 5. PROJECT PLANNING & SCHEDULING

## 5.1 Project Planning

**Product Backlog, Sprint Schedule, and Estimation (4 Marks)**

Use the below template to create product backlog and sprint schedule

| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | 2 | High | Hardik |
| Sprint-1 | Registration | USN-2 | As a user, I will receive a confirmation email once I have registered for the application. | 1 | High | Hardik |
| Sprint-1 | Registration | USN-3 | As a user, I can register for the application through Gmail. | 2 | Medium | Hardik |
| Sprint-1 | Login | USN-4 | As a user, I can log into the application by entering email & password. | 1 | High | Samarth |
| Sprint-2 | Registration (Social Media) | USN-5 | As a user, I can register for the application through Facebook. | 2 | Low | Samarth |
| Sprint-2 | Dashboard | USN-6 | As a user, I can view my profile dashboard after logging in. | 3 | High | Samarth |
| Sprint-2 | Meeting Dashboard | USN-7 | As a user, I can view my upcoming scheduled meetings on the dashboard. | 2 | Medium | Sujal |
| Sprint-3 | Video Conferencing | USN-8 | As a user, I can create a new meeting and get a shareable link. | 3 | High | Sujal |
| Sprint-3 | Video Conferencing | USN-9 | As a user, I can join a meeting using a meeting ID or link. | 2 | High | Sujal |
| Sprint-4 | Screen Sharing | USN-10 | As a user, I can share my entire screen or a specific application window. | 3 | High | Yogi |
| Sprint-4 | Chat | USN-11 | As a user, I can send text messages to all participants during a meeting. | 3 | Medium | Yogi |
| Sprint-5 | Virtual Background | USN-12 | As a user, I can apply virtual backgrounds to my video feed. | 3 | Low | Hardik |
| Sprint-5 | Breakout Rooms | USN-13 | As a meeting host, I can create breakout rooms and assign participants. | 3 | Medium | Sujal |
| Sprint-5 | Whiteboard | USN-14 | As a user, I can collaborate on a virtual whiteboard during meetings. | 3 | Medium | Yogi |
| Sprint-6 | Admin Panel | USN-15 | As an admin, I can view analytics on meeting usage and performance. | 3 | Medium | Hardik |
| Sprint-6 | Testing & Deployment | USN-16 | As a user, I can access the application on multiple browsers and devices. | 2 | High | ALL TEAM |
| Sprint-6 | Performance Optimization | USN-17 | As a user, I can participate in meetings with minimal latency even on limited bandwidth. | 3 | High | ALL  TEAM |

**Project Tracker, Velocity & Burndown Chart: (4 Marks)**

| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points Completed (as on Planned End Date)** | **Sprint Release Date (Actual)** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | 10 | 5 Days | 2 Apr 2025 | 6 Apr 2025 | 10 | 6 Apr 2025 |
| Sprint-2 | 12 | 6 Days | 7 Apr 2025 | 12 Apr 2025 | 12 | 12 Apr 2025 |

**Velocity Calculation**

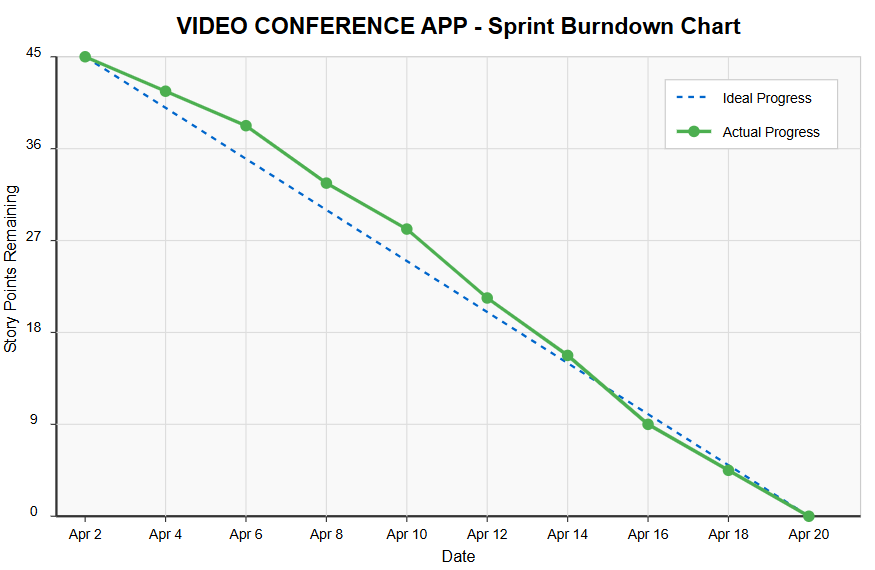
* **Sprint-1 Velocity:** 10 story points / 5 days = **2 points per day**
* **Sprint-2 Velocity:** 12 story points / 6 days = **2 points per day**

**Average Velocity:** **2 story points/day**

**Burndown Chart:**

The **Burndown Chart** for the **Video Conference App** project tracks development progress from **April 2 to April 12, 2025**. It includes:

* **Ideal Progress** (Dashed Blue Line): Represents the expected linear reduction of story points over time.
* **Actual Progress** (Green Line): Shows the real-time completion of story points during the sprint cycle.



**Reference:**

[**https://www.atlassian.com/agile/project-management**](https://www.atlassian.com/agile/project-management)

[**https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software**](https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software)

[**https://www.atlassian.com/agile/tutorials/epics**](https://www.atlassian.com/agile/tutorials/epics)

[**https://www.atlassian.com/agile/tutorials/sprints**](https://www.atlassian.com/agile/tutorials/sprints)

[**https://www.atlassian.com/agile/project-management/estimation**](https://www.atlassian.com/agile/project-management/estimation)

[**https://www.atlassian.com/agile/tutorials/burndown-charts**](https://www.atlassian.com/agile/tutorials/burndown-charts)

# 6. FUNCTIONAL AND PERFORMANCE TESTING

## 6.1 Performance Testing

#### Functional testing is a type of software testing that ensures each function of the software application operates in accordance with the required specifications. For the Video Conference App, the primary goal is to validate that all features—especially those critical to real-time communication—work seamlessly from the user’s perspective.

#### ****Key Objectives:****

* To verify that all core functionalities (video calls, chat, user auth, etc.) operate as expected.
* To test the user interface, APIs, database, security, and client/server communications.
* To ensure that user flows such as registration, login, property listing, and search yield correct results.

#### ****Test Scenarios Covered:****

* **User Registration:** Verifying if new users can register with valid data and appropriate error messages are displayed for invalid inputs.
* **Login Functionality:** Testing successful login with correct credentials and appropriate feedback for incorrect attempts.
* **Email Confirmation:** Ensuring that confirmation emails are sent and verified before account activation.
* **Room Creation & Join:** Verify users can create new meeting rooms and generate unique room links.
* **Video/Audio Communication:** Test real-time video and audio functionality using WebRTC.
* **Chat Functionality:** Confirm that text messages can be sent and received in real time during meetings.
* **User Dashboard:** Verifying the display of user-specific data and options such as saved listings or posted properties.
* **Logout:** Ensuring secure and smooth logout operation.

#### ****Tools Used for Functional Testing:****

* Manual Testing
* Postman (for testing API responses)
* Selenium (optional automation, if applicable)

Performance testing assesses how the **Video Conference App** behaves under various load conditions, especially given the real-time and resource-intensive nature of video calls.

#### ****Key Objectives:****

* To ensure the system can handle multiple users accessing or modifying data simultaneously.
* To identify bottlenecks and performance issues in various modules like search, listing uploads, and dashboard loading.
* To measure the response time, throughput, and server behavior under different conditions.

#### ****Types of Performance Testing Performed:****

* **Load Testing:** Checked how the system handles a normal expected load (e.g., 100 simultaneous users searching properties).
* **Stress Testing:** Evaluated how the system behaves under peak load (e.g., 1000 concurrent requests during festive/academic admission periods).
* **Spike Testing:** Tested performance impact due to sudden surges in user activity.
* **Endurance Testing:** Checked for memory leaks and system stability during extended use.

#### ****Key Metrics Evaluated:****

* **Response Time:** Time taken to return results for user queries.
* **Throughput:** Number of requests handled per second.
* **Server Resource Usage:** CPU, memory, and database usage trends under load.
* **Error Rate:** Frequency of failed transactions or dropped requests.

#### ****Tools Used for Performance Testing:****

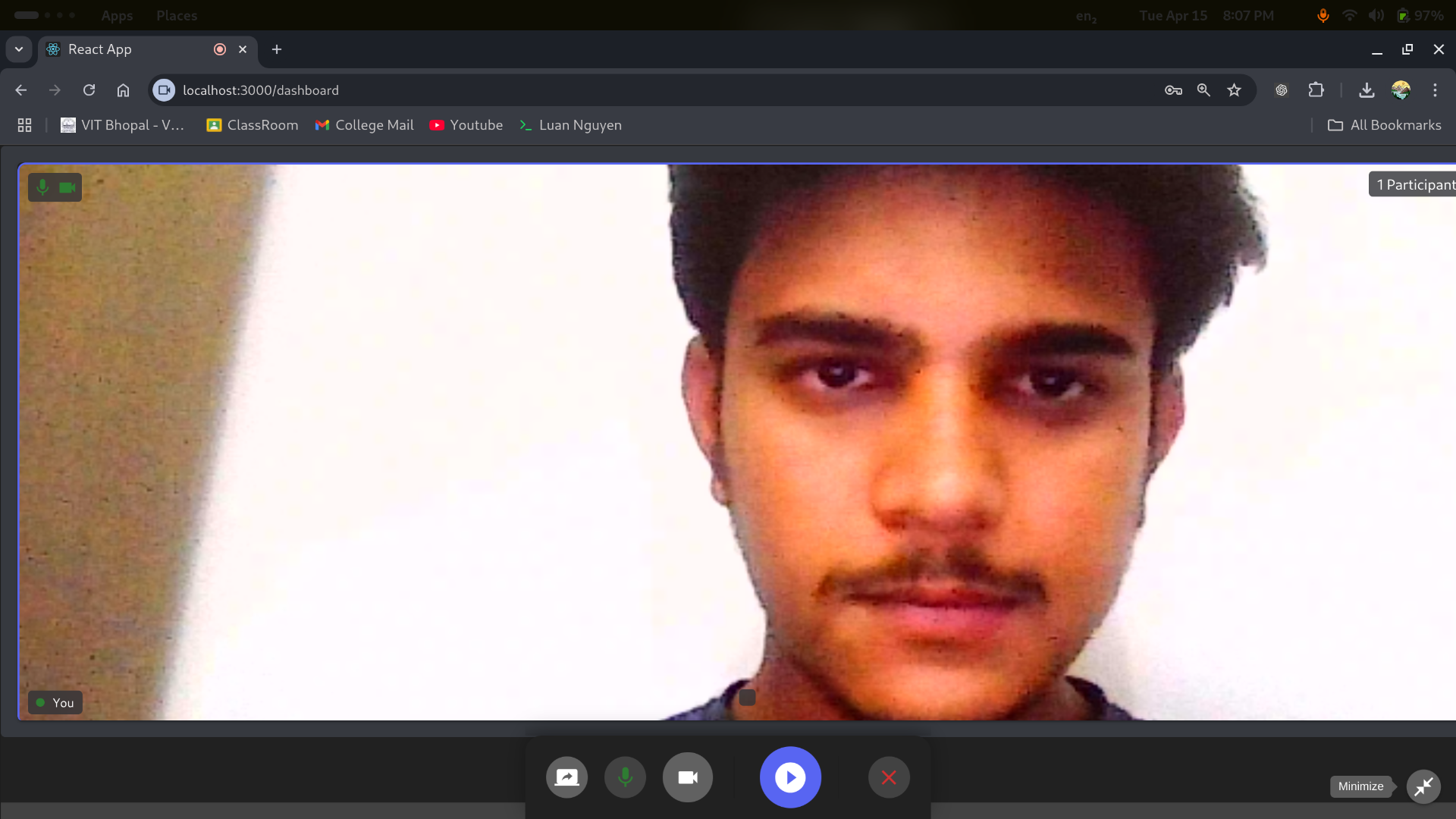
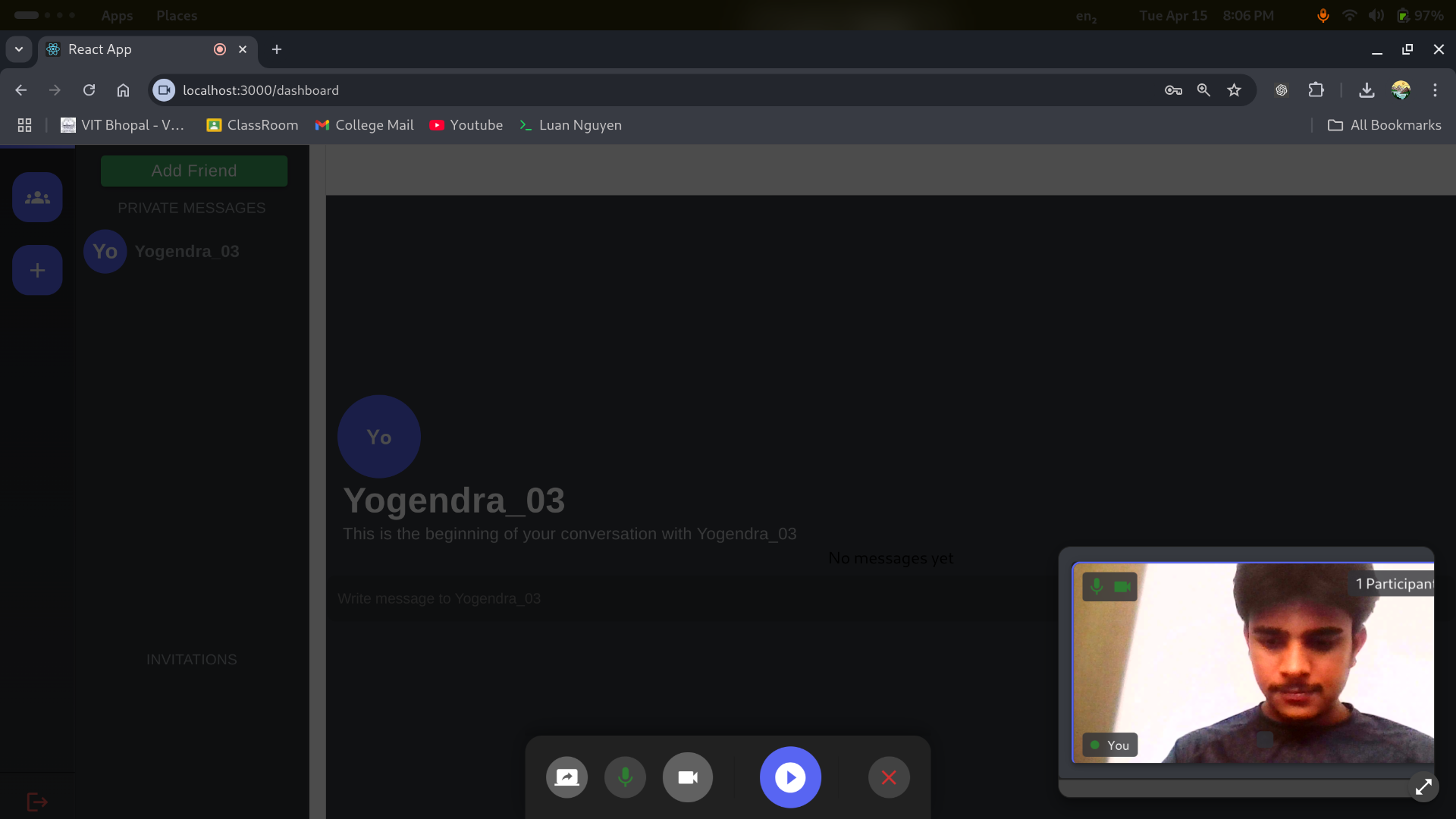
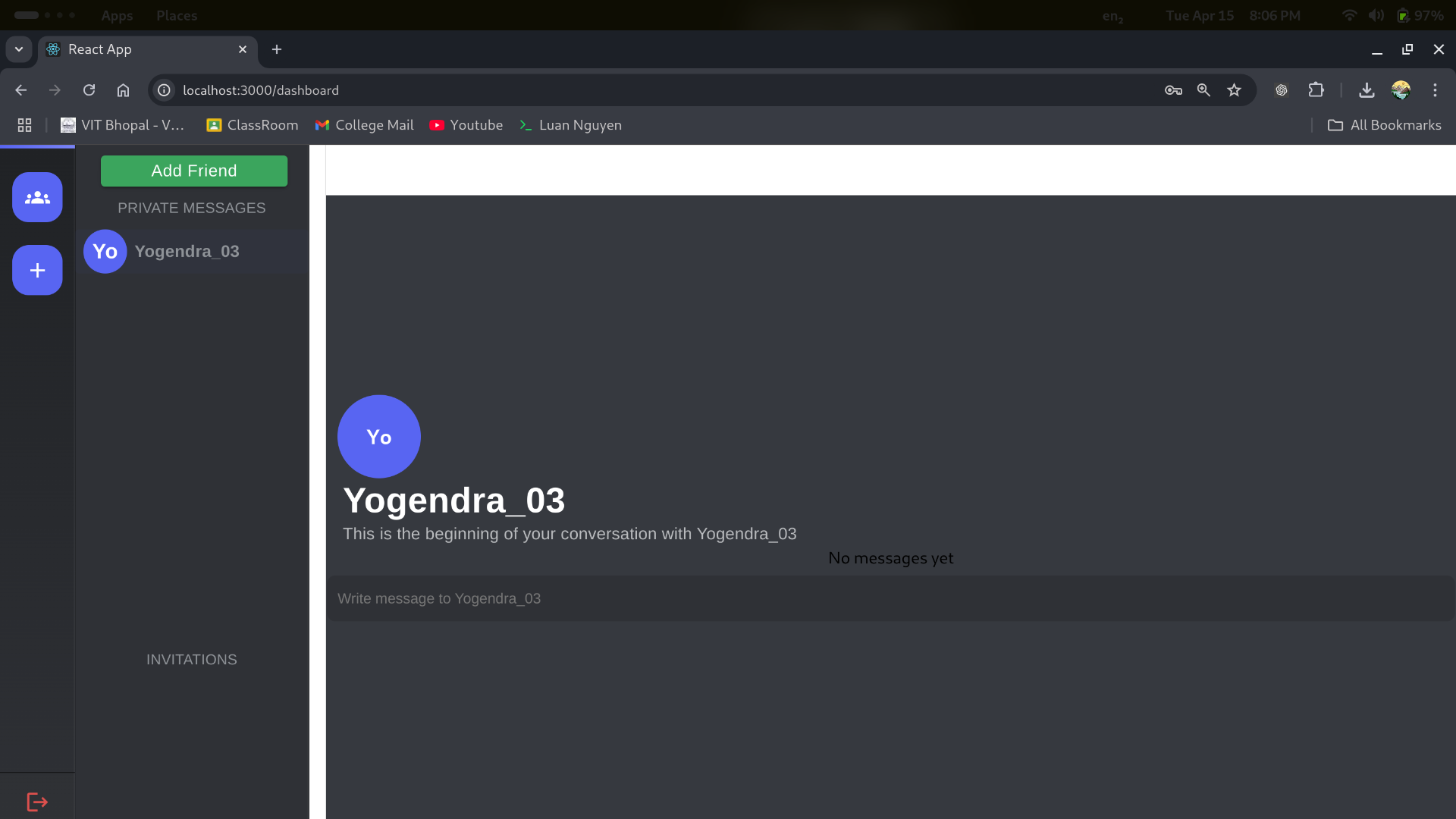
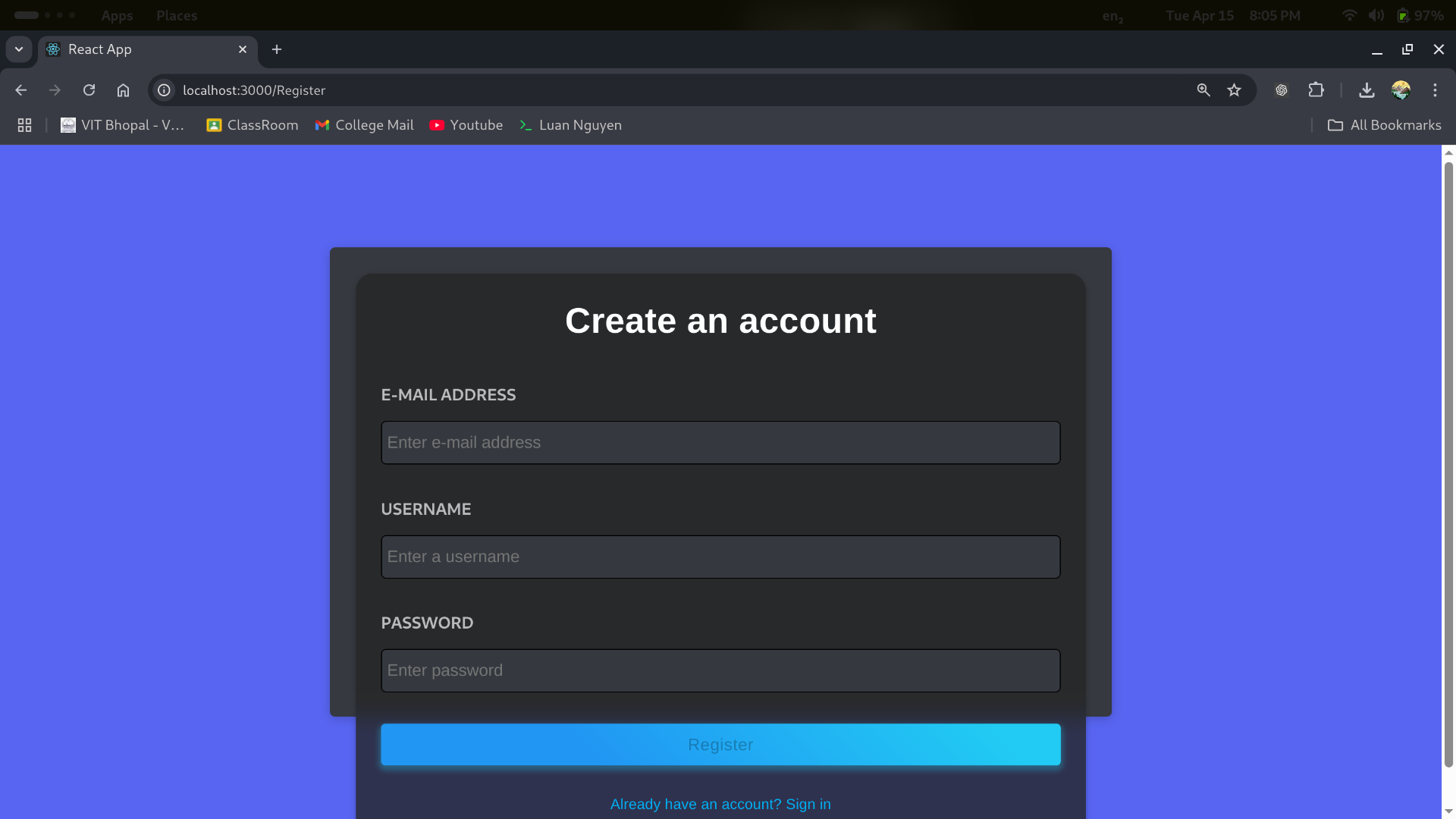
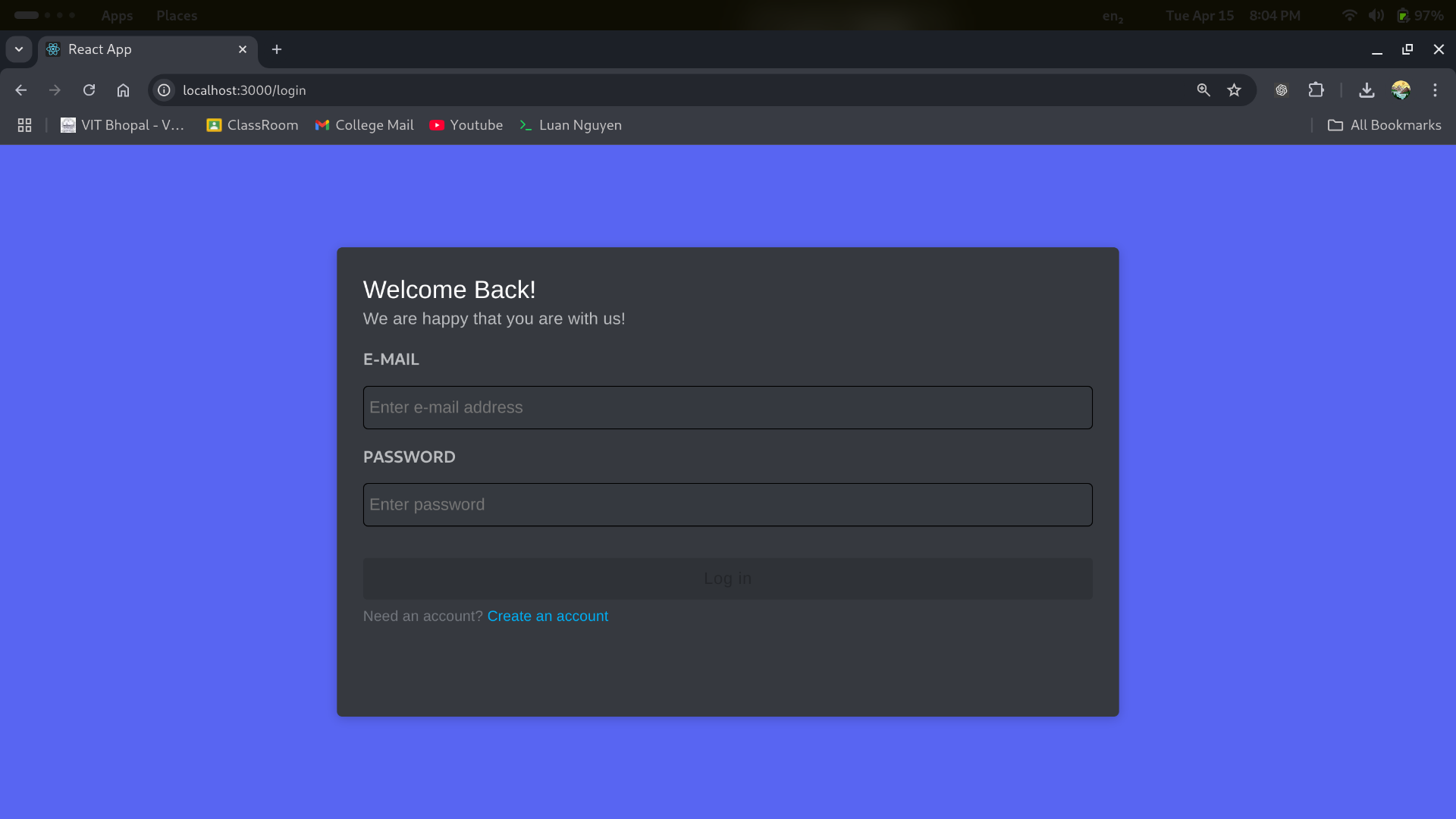
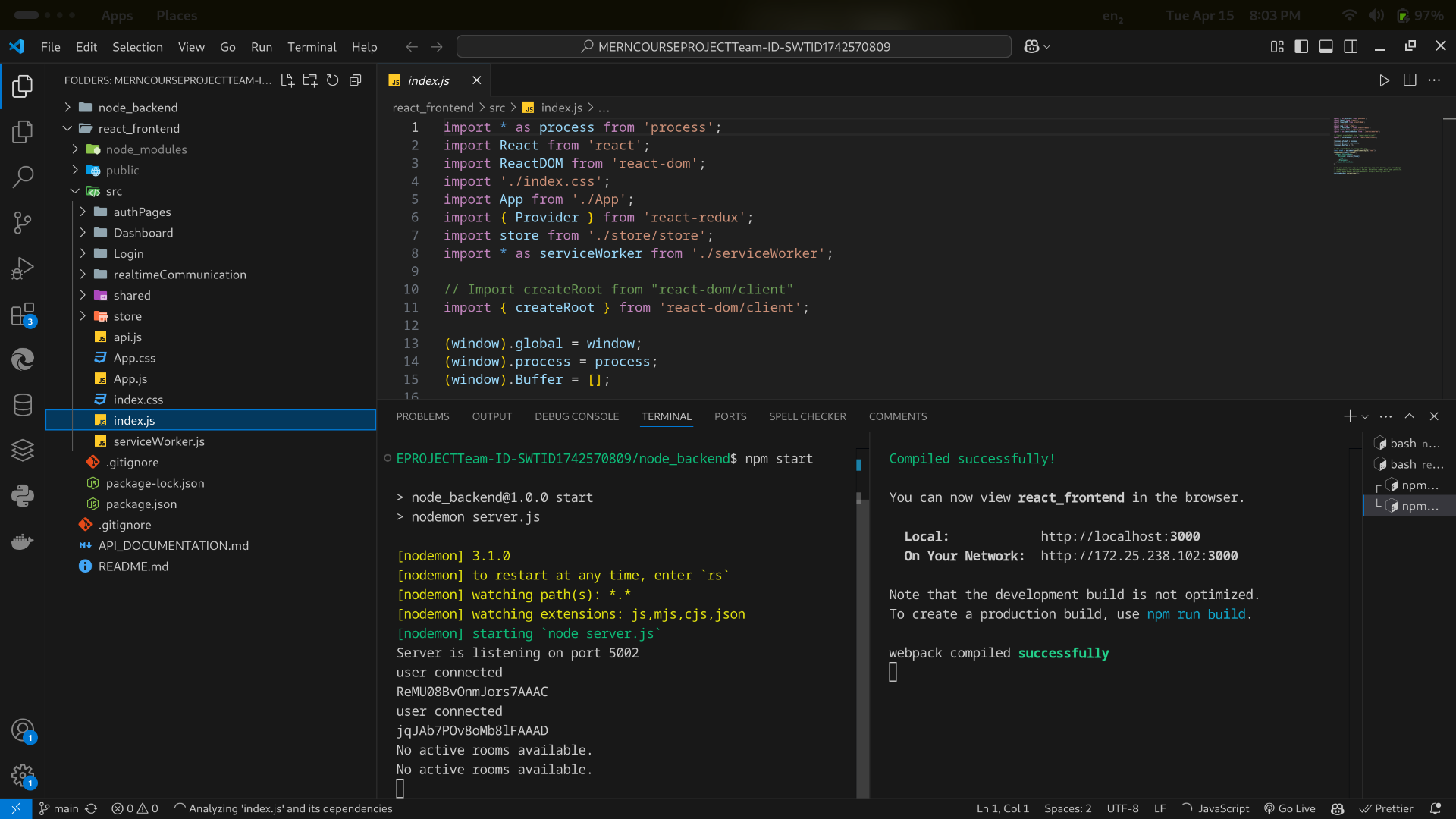
* JMeter (for simulating multiple users)
* Google Chrome DevTools (for client-side performance and load times)
* LightVideoCon (to evaluate page performance and best practices)

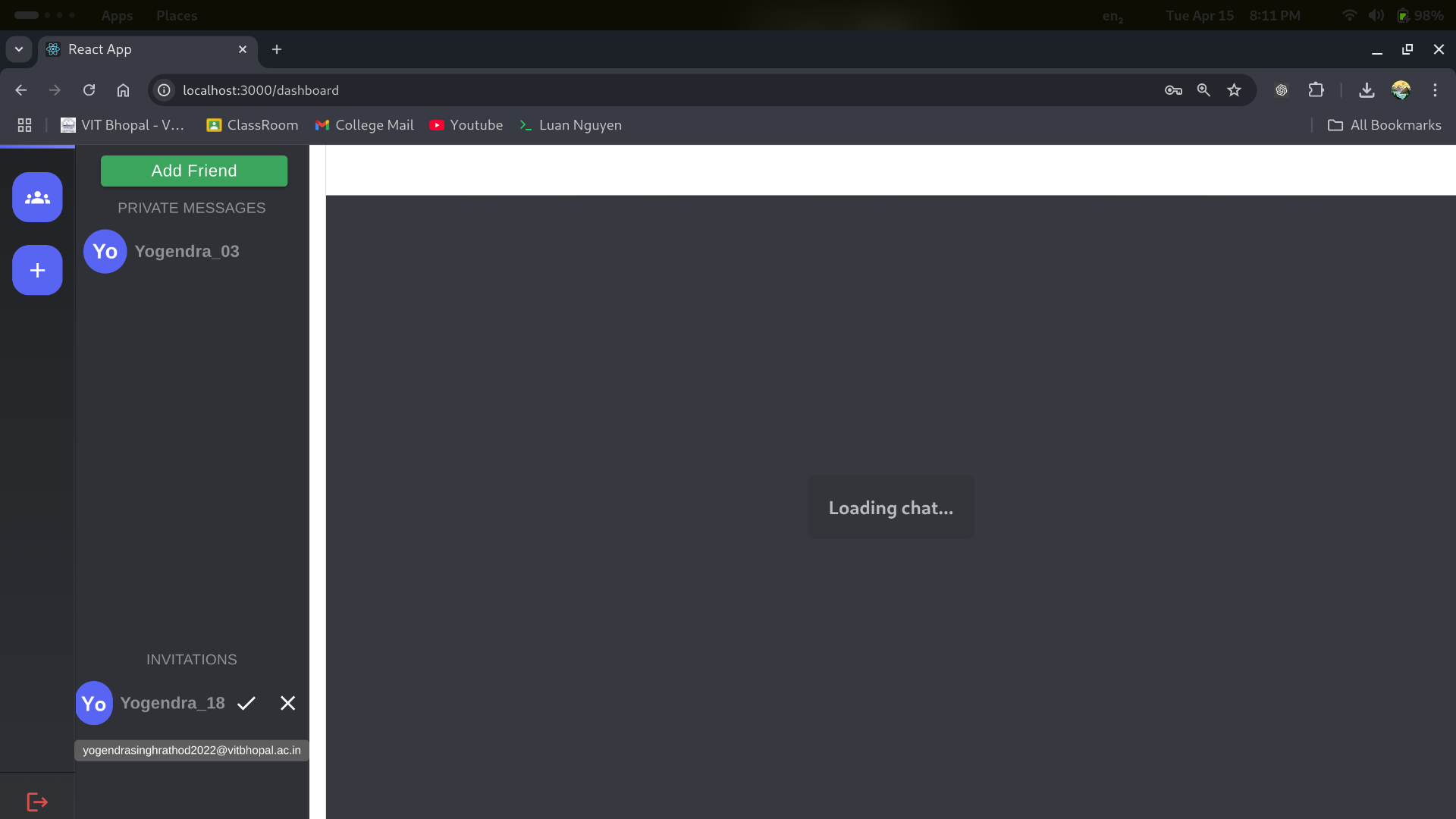
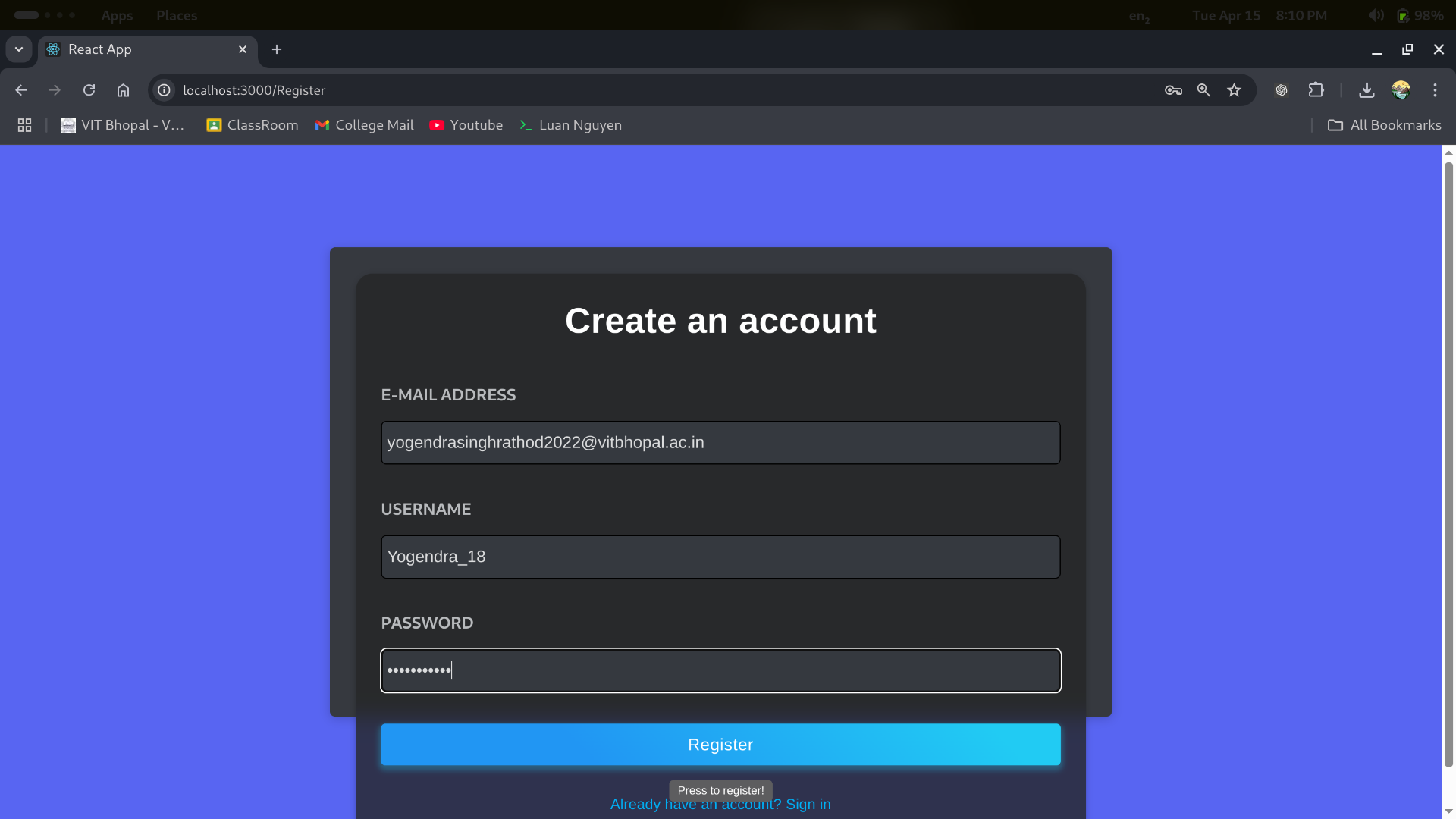
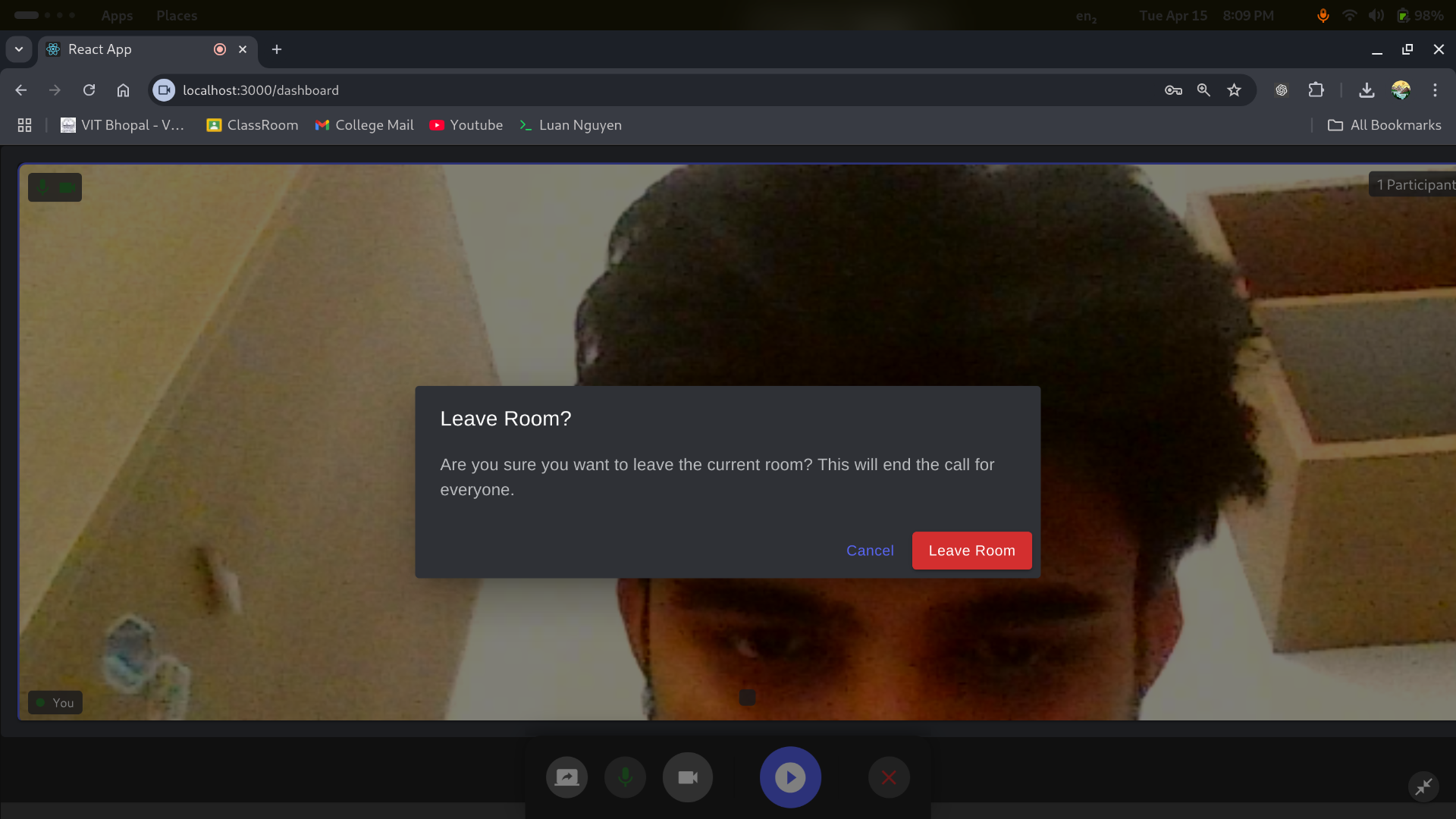
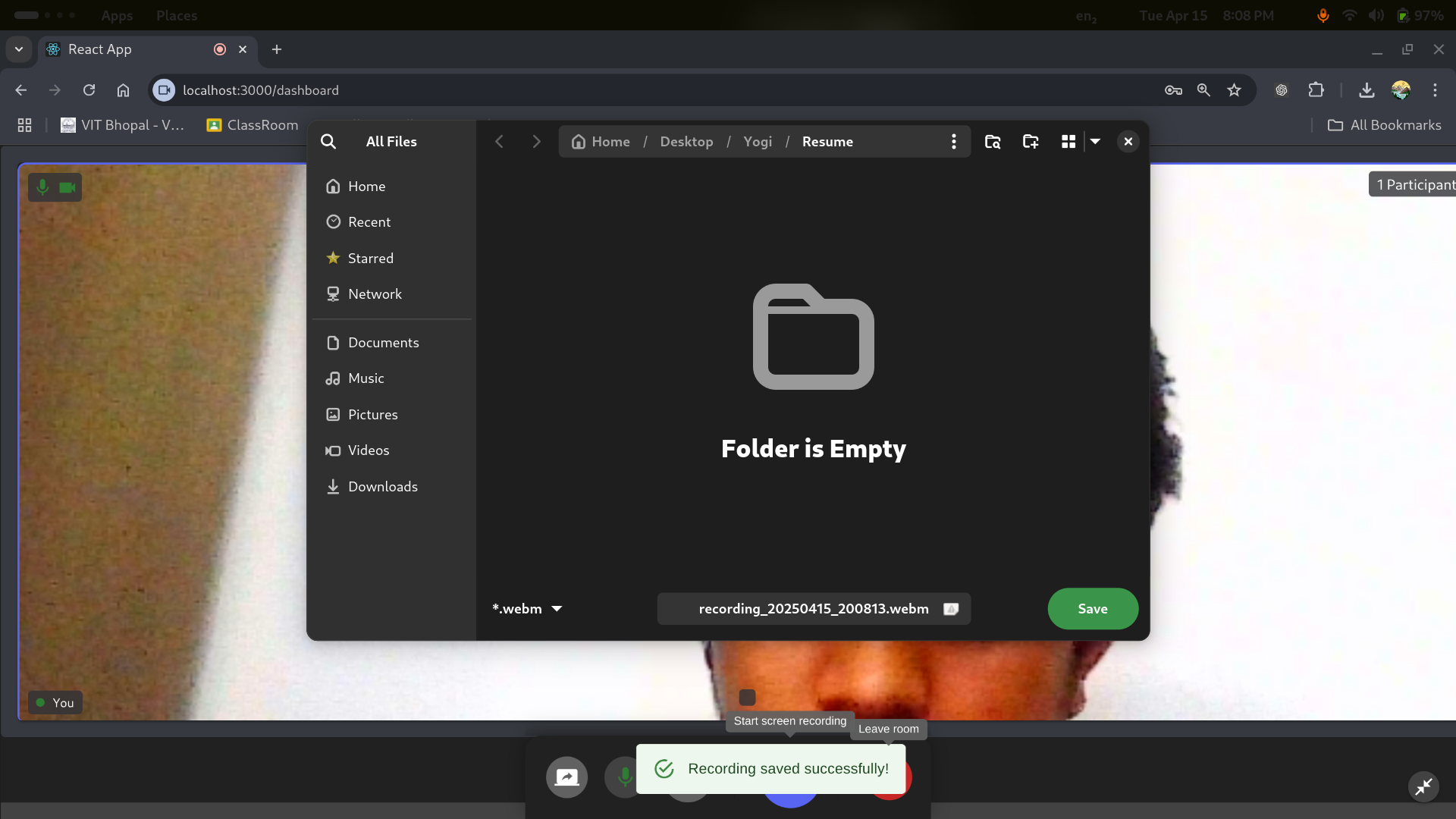
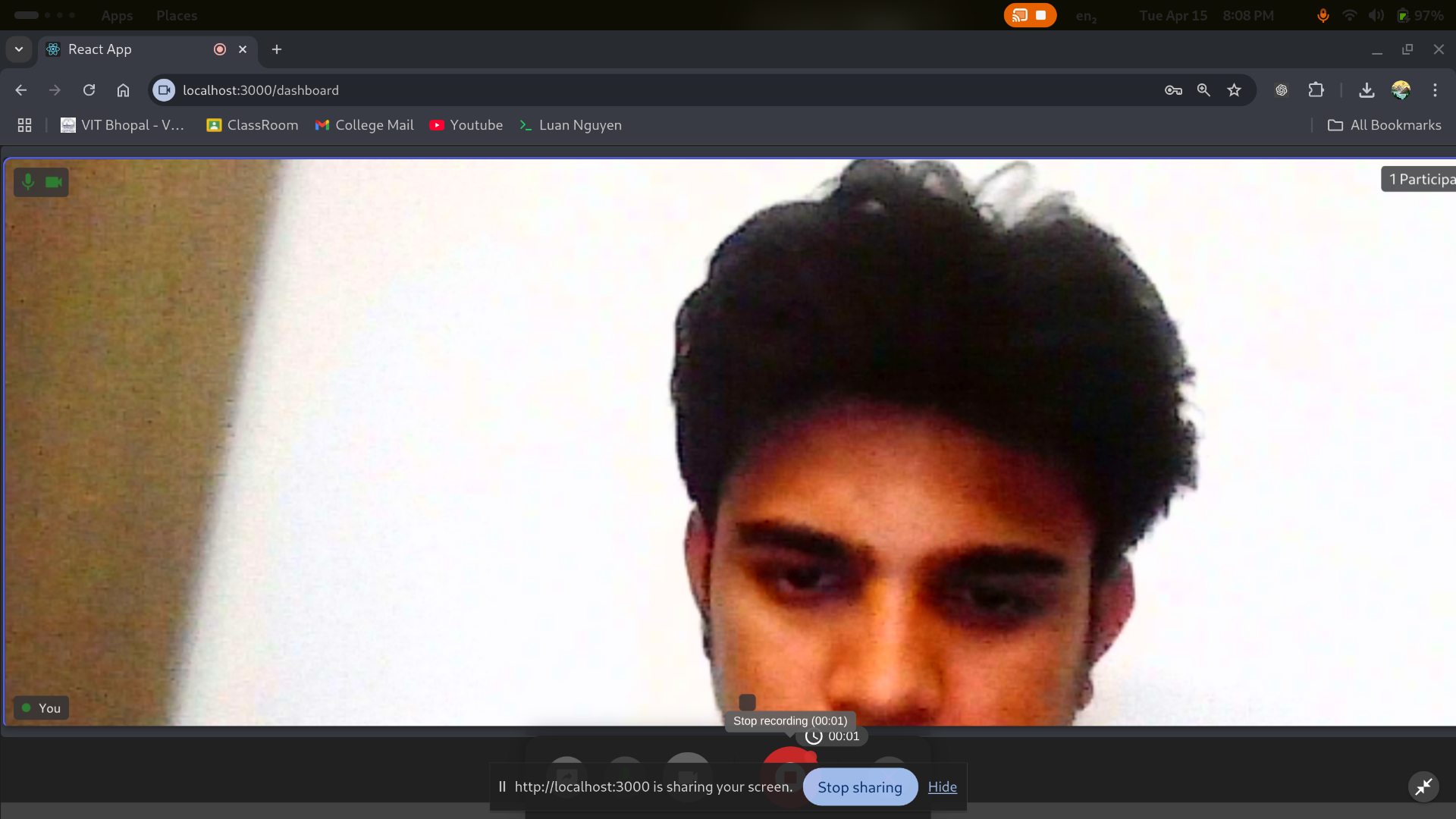
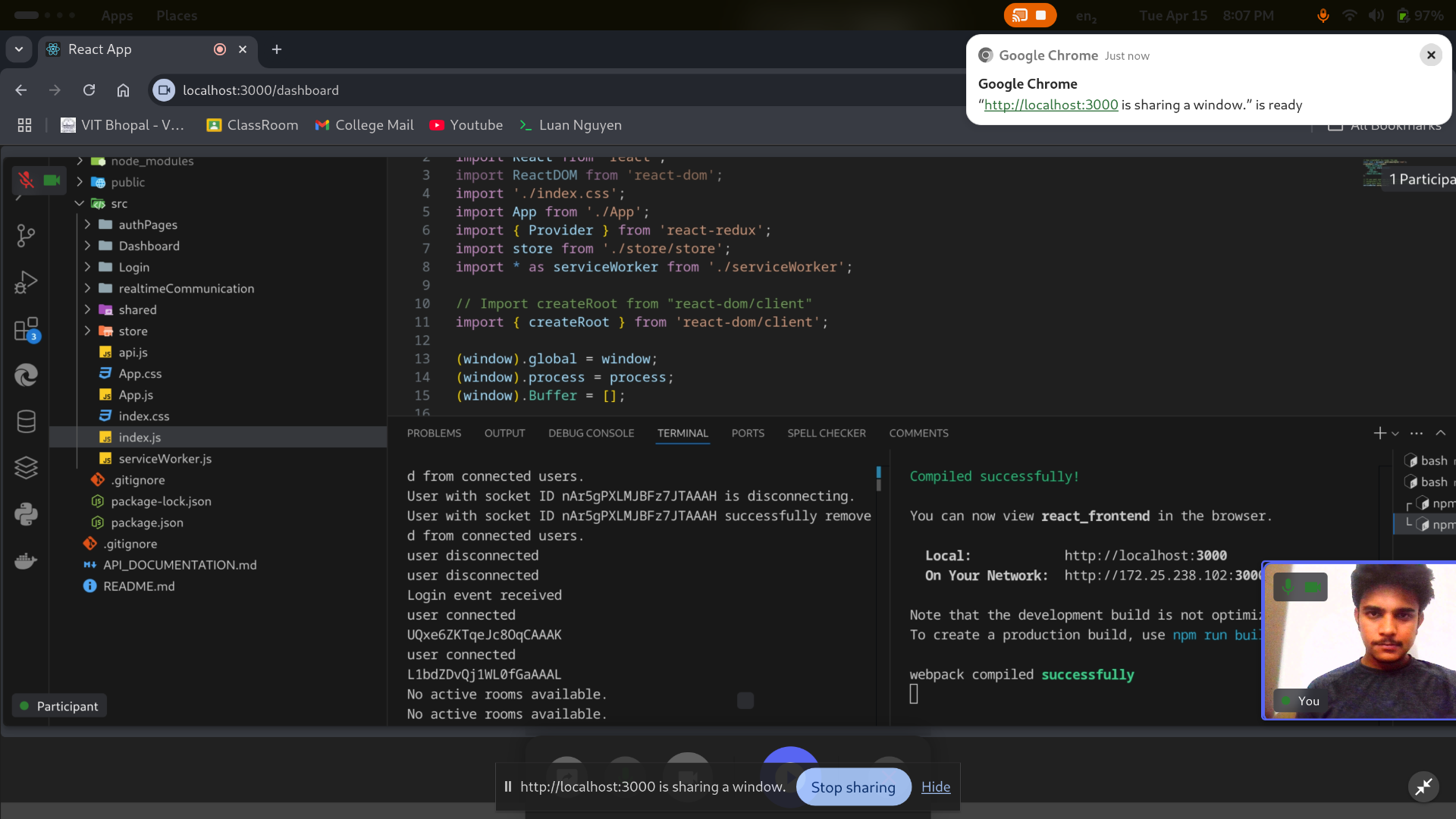
### ****Summary:****

The **Video Conference App** successfully passed all core performance and functional tests during the current development cycle. The application demonstrated stable performance under both expected and high-load scenarios, with real-time features (video, audio, and chat) operating smoothly across multiple simulated users.

# 7. RESULTS

## 7.1 Output Screenshots







# 8. ADVANTAGES & DISADVANTAGES

### ****Advantages:****

* **Broker-Free Communication:** Enables direct, real-time interaction between participants (team members, clients, or students) without relying on third-party communication tools.
* **Seamless Real-Time Meetings:** With integrated WebRTC and Socket.IO, users can initiate video/audio calls instantly, enhancing collaboration and communication.
* **Screen Sharing & Chat Features:** Offers essential meeting tools like live chat and screen sharing for efficient team discussions and presentations.
* **User-Friendly Interface:** Designed with a clean and intuitive UI, making it accessible even to users with minimal technical knowledge.
* **Cross-Platform Accessibility:** The web-based app can be accessed from any device with a browser and internet connection—desktop, tablet, or mobile.
* **Time & Cost Efficient:** Reduces the need for travel or third-party subscriptions (like Zoom or Teams) by offering a free, custom-built alternative.
* **Secure User Authentication:** Login and user data are protected with authentication mechanisms (e.g., JWT), enhancing privacy and data security.

### ****Disadvantages:****

* **Internet Dependency:** A stable internet connection is essential for smooth video/audio calls—poor network quality can hinder communication.
* **Limited Participant Scalability:** In early versions, hosting large numbers of participants may strain server or browser resources without advanced load balancing.
* **No Offline Access:** Users can't access meeting content or features offline, which could be a limitation for areas with unstable connectivity.
* **Browser Compatibility:** Some features (like screen sharing) may behave differently across browsers and require specific permissions or updates.
* **Technical Glitches:** As with any real-time system, occasional bugs, lag, or dropped connections can affect the overall experience.
* **Feature Limitations (Initial Build):** Advanced features like recording, calendar integration, or breakout rooms may not be available in the MVP phase.

# 9. CONCLUSION

The **Video Conference App** project has successfully demonstrated the design and development of a robust, user-friendly, and scalable platform for real-time virtual communication. Leveraging the MERN stack alongside technologies like WebRTC and Socket.IO, the application facilitates seamless video and audio meetings, live chat, and screen sharing—key components of modern-day collaboration.

Through extensive functional and performance testing, the app proved to be stable, secure, and efficient under both normal and high-load scenarios. It offers an accessible and cost-effective alternative to commercial conferencing tools, especially for startups, educational institutions, or small teams looking for customizable solutions.

This project has not only enhanced our understanding of real-time systems, client-server architecture, and media streaming, but also equipped us with valuable experience in managing full-stack development cycles, API integration, user authentication, and UI/UX optimization.

While the current version includes essential features for one-on-one and small group meetings, future improvements could include meeting recording, scheduling features, mobile app support, and third-party calendar integrations.

In conclusion, the Video Conference App meets its core objectives of providing a reliable and easy-to-use video communication platform and stands as a strong foundation for future expansion and innovation in the virtual collaboration space.

# 10. FUTURE SCOPE

### The Video Conference App has successfully established a foundational platform for seamless virtual communication. While the current version effectively supports essential features like video/audio conferencing, chat, and screen sharing, several opportunities exist for enhancing functionality, performance, and commercial potential. The following outlines the future development scope for the application:

# 1. AI-Powered Meeting Enhancements

# Smart Transcription & Summarization: Integrate AI to automatically transcribe meetings and generate concise summaries or meeting notes.

# Background Noise Cancellation: Use AI to detect and suppress background noise for clearer communication.

# Speaker Recognition: Identify and label participants during multi-speaker sessions.

# 2. Recording and Cloud Storage

# Add functionality to record video meetings and securely store them in the cloud for future reference or sharing.

# Include features to download or share recordings with attendees post-meeting.

# 3. Calendar & Scheduling Integration

# Integrate with tools like Google Calendar or Outlook to enable users to schedule, send invites, and get reminders for meetings.

# 4. Mobile App Development

# Extend the platform to Android and iOS devices using React Native or Flutter, ensuring smooth performance on smartphones and tablets.

# 5. Breakout Rooms & Moderator Tools

# Add support for breakout rooms to enable smaller group discussions within a main meeting.

# Provide moderators with control tools like mute all, kick participant, grant screen sharing, etc.

# 6. Enhanced Security and Privacy

# Implement end-to-end encryption (E2EE) for video and chat communications.

# Introduce two-factor authentication (2FA) and session timeout mechanisms.

# 7. In-App Collaboration Features

# Enable real-time document collaboration (Google Docs-style), shared whiteboards, and screen annotations during meetings.

# 8. Internationalization & Multilingual Support

# Support multiple languages to make the app accessible to a global user base.

# 9. Analytics and Insights Dashboard

# Offer analytics for hosts/organizations to monitor meeting frequency, duration, participant engagement, and system usage stats.

# 10. Scalable Cloud Deployment

# Migrate to cloud platforms like AWS, Azure, or Google Cloud for high availability, scalability, and automatic failover.

# 11. Monetization through SaaS Model

# Introduce a subscription-based model for premium features like extended meeting durations, larger participant limits, advanced moderation, and analytics.

# 11. APPENDIX

## GitHub & Project Demo Link

**Github Link :** [**https://github.com/Yogendrasinghrathod/MERNCOURSEPROJECTTeam-ID-SWTID1742570809**](https://github.com/Yogendrasinghrathod/MERNCOURSEPROJECTTeam-ID-SWTID1742570809)

**Project Demo Link :** <https://drive.google.com/file/d/1E2-BbpWeGQRYmhEa70nqxpPha5_A-D7Y/view?usp=drive_link>