**Smartbridge**

**Final Project Report**

**Video Conferencing App**

ConvoConnect

**Submitted by:**

Patel Shubh Bhargavkumar

Devansh Jadon

Pratham Amit Singh

VIT BHOPAL

April 2025

**Project Report**

1. **INTRODUCTION**

**1.1 Project Overview**

"ConvoConnect: Simplifying Video Conferencing" is a comprehensive web-based solution designed to revolutionize virtual communication by providing a seamless, secure, and intuitive video conferencing experience. Developed using full-stack web development technologies, with a primary emphasis on the MERN (MongoDB, Express.js, React.js, Node.js) stack, the platform integrates modern web technologies to address the evolving needs of remote collaboration.

ConvoConnect offers a range of features, including high-quality video and audio calls, real-time messaging, screen sharing, virtual meeting rooms, and session recording capabilities. Emphasizing user experience and accessibility, the platform is built to be responsive and cross-platform, ensuring compatibility across various devices and operating systems. Robust backend integration guarantees secure data handling and privacy, while the scalable architecture allows for easy adaptability to increasing user demands.

This project demonstrates the potential of web technologies in creating innovative and practical solutions to everyday challenges. Through ConvoConnect, the development team aims to bridge the gap between technology and communication, fostering collaboration and efficiency in personal, educational, and professional environments.

* 1. **Purpose**

The purpose of the project "ConvoConnect: Simplifying Video Conferencing" is to create a versatile, user-centric platform that addresses the challenges of modern virtual communication. With the increasing reliance on remote interactions in personal, educational, and professional domains, existing video conferencing solutions often face limitations in usability, accessibility, or affordability. This project seeks to simplify the video conferencing experience by providing a seamless and intuitive platform that combines essential features like high-quality video calls, real-time messaging, screen sharing, and session recording, all within a secure and scalable environment.

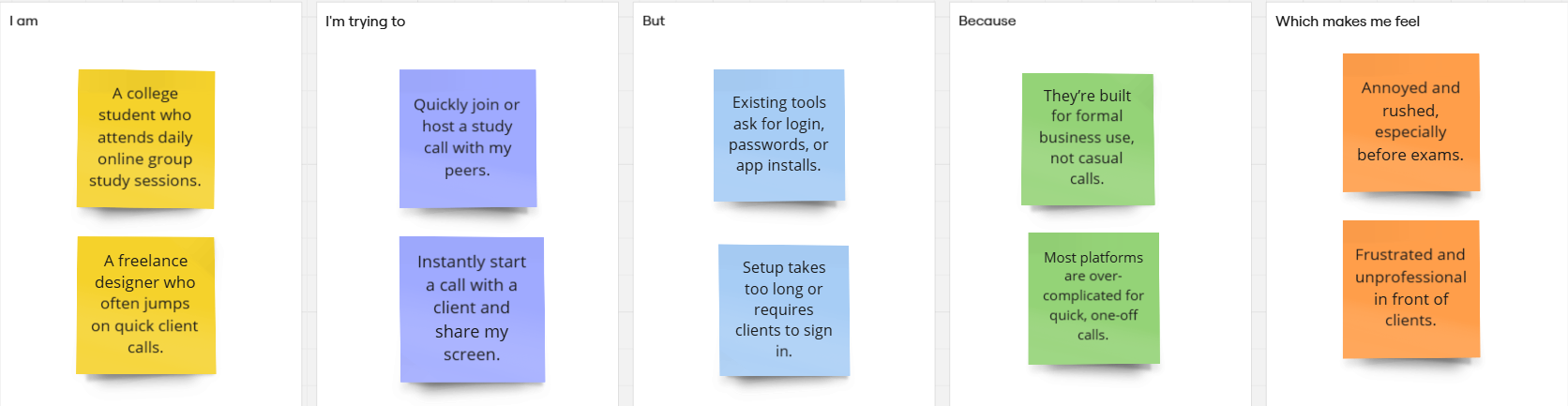
By leveraging full-stack web development technologies, particularly the MERN stack, the platform aims to enhance connectivity, foster collaboration, and bridge communication gaps. “ConvoConnect” is designed to be accessible across various devices, ensuring inclusivity and ease of use for users worldwide. Ultimately, the project aspires to redefine virtual communication by delivering a cost-effective, efficient, and reliable solution tailored to the diverse needs of individuals and organizations.

1. **IDEATION PHASE**

**2.1 Problem Statement**

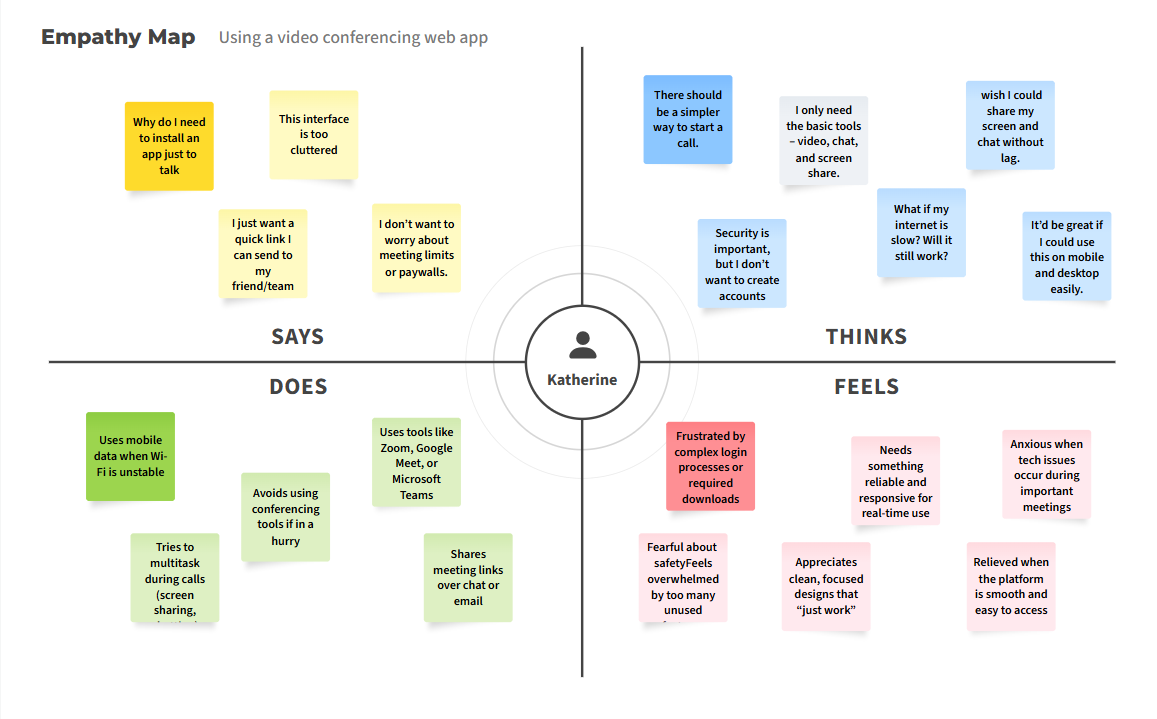
"In the current digital landscape, existing video conferencing platforms often come with steep learning curves, cluttered interfaces, unnecessary features, or limitations on free usage. Users — especially students, small teams, freelancers, and educators — seek a simpler, secure, and more intuitive solution for initiating and managing virtual meetings without the need for complex setups or installations. There is a need for a lightweight, browser-based video conferencing application that prioritizes ease of use, quick room creation, privacy, and seamless performance across devices."

**Customer Problem Statement**



* 1. **Empathy Map Canvas**

Empathy map for a user (Katherine) using Convoconnect



* 1. **Brainstorming**

The brainstorming phase of "ConvoConnect: Simplifying Video Conferencing" involved intensive discussions and collaborative ideation to identify the core features and challenges in creating an efficient video conferencing platform. The team began by analyzing existing solutions to pinpoint their limitations, such as complex interfaces, high costs, and lack of customization options. Emphasis was placed on understanding user needs, including ease of use, high-quality video and audio, real-time communication, and robust security.

Ideas were generated through structured brainstorming sessions, focusing on integrating features like screen sharing, session recording, and live chat within a responsive and user-friendly interface. The team also explored scalable architecture and cross-platform compatibility to ensure accessibility and adaptability. Feedback and insights from potential users further guided the conceptualization of the platform, ensuring that the final product would address real-world communication challenges effectively. This collective ideation laid the foundation for developing ConvoConnect as an innovative and practical solution for modern virtual communication.

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

ProblemStatement1:–Student(Neha,20):  
I am a college student who attends daily online group study sessions.  
I'm trying to quickly join or host a study call with my peers,  
but existing tools ask for login, passwords, or app installs,  
because they’re built for formal business use, not casual calls,  
which makes me feel annoyed and rushed, especially before exams.

ProblemStatement2:–Freelancer(Aman,28):  
I am a freelance designer who often jumps on quick client calls.  
I'm trying to instantly start a call with a client and share my screen,  
but setup takes too long or requires clients to sign in,  
because most platforms are over-complicated for quick, one-off calls,  
which makes me feel frustrated and unprofessional in front of clients.

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

**Key Ideas:  
-** One-click room creation and joining without login  
- Lightweight UI optimized for mobile and web  
- End-to-end encrypted WebRTC communication  
- Built-in chat, screen sharing, and mute toggle  
- Auto-generated room links with expiration  
- Dark mode and virtual backgrounds (future scope)  
- No installation — works directly in browser  
- Option to protect rooms with a passcode.

**Step-3: Idea Prioritization**

**High Priority:  
-** One-click room creation and sharing  
- WebRTC for video/audio/screen share  
- Responsive UI for mobile and desktop  
- Password protection for rooms

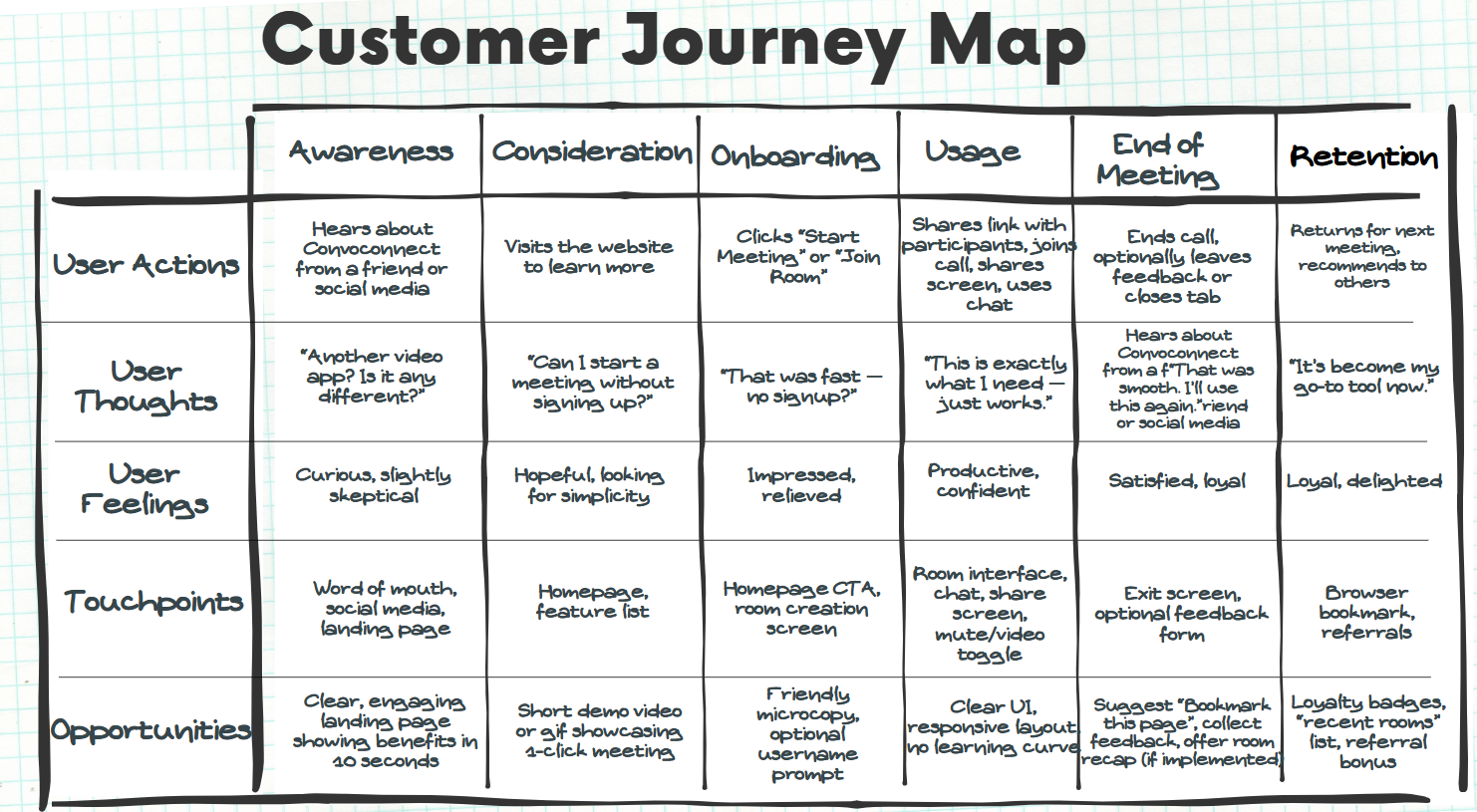
**Medium Priority:**- In-room text chat  
- Room auto-expiry feature

**Low Priority (Future Scope):**- Virtual background  
- AI meeting transcription  
- File sharing and polls

1. **REQUIREMENT ANALYSIS**

The requirement analysis phase for "Convo Connect: Simplifying Video Conferencing" focused on identifying the technical, functional, and non-functional needs essential for developing a robust and user-centric platform. Key functional requirements included high-quality video and audio calls, real-time messaging, screen sharing, session recording, and secure data handling. Non-functional requirements emphasized scalability, cross-platform compatibility, low latency, and an intuitive user interface to enhance the overall user experience. Additionally, the technological stack was chosen to meet these demands, with the MERN stack being selected for its efficiency, flexibility, and scalability. This thorough analysis provided a clear roadmap for the development process, ensuring that the platform meets the expectations of diverse users.

**3.1 Customer Journey map**



* 1. **Solution Requirement**

**Functional Requirements:**

Following are the functional requirements of the proposed solution**.**

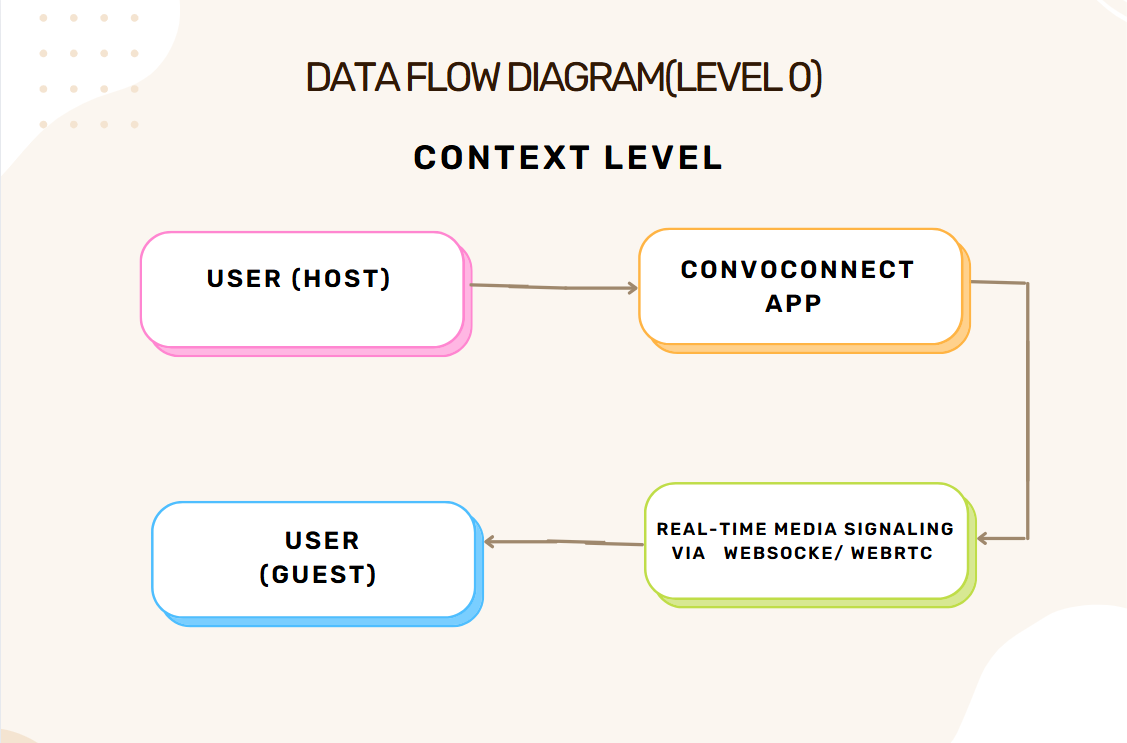
|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| **FR-1** | **Room Management** | One-click room creation  Join via room link  Auto-generated unique room ID |
| **FR-2** | **Real-Time Communication** | Peer-to-peer video and audio via WebRTC  Screen sharing  Mute/unmute mic and video toggle |
| **FR-3** | **Chat Functionality** | Text chat within the meeting room  Display user names with messages |
| FR-4 | **Room Security** | Optional passcode protection  Room auto-expiry after session ends |

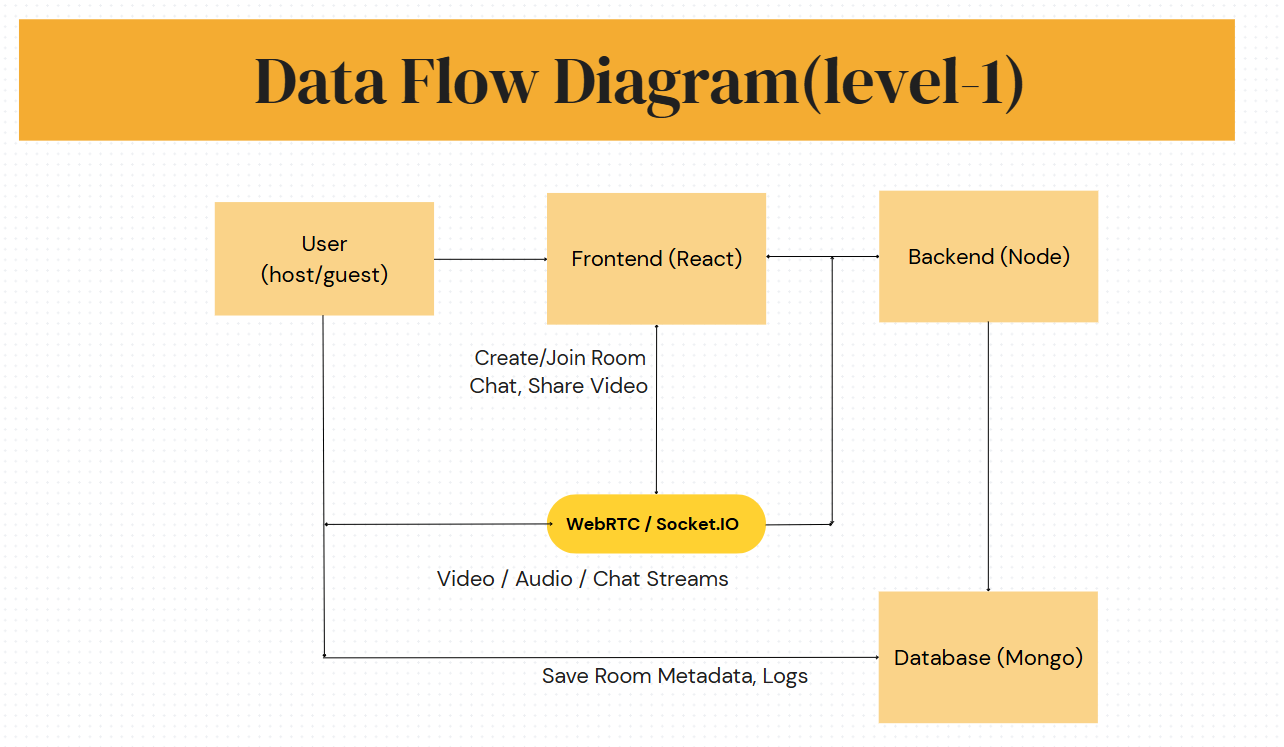
**Non-functional Requirements:**

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR**-1** | Usability | Minimal, intuitive UI requiring no learning curve |
| NFR**-2** | Security | End-to-end encrypted media streams, no unnecessary data collection |
| NFR**-3** | Reliability | Stable performance in varying network conditions |
| NFR-4 | Performance | Fast room joining and low-latency communication |
| NFR-5 | Availability | Accessible across modern browsers and mobile devices |
| NFR-6 | Scalability | Support multiple rooms and up to 10–15 users per session |

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

* 1. **Data Flow Diagram**





* 1. **Technology Stack**

🖥️ Frontend (Client-Side)

|  |  |
| --- | --- |
| **Tech** | **Purpose** |
| **React.js** | Building dynamic and responsive UI components |
| **React Router** | For page routing (Home, Room, etc.) |
| **Tailwind CSS** | Utility-first styling framework for UI |
| **Axios** | To make HTTP requests to backend (e.g., create/join room) |
| **Socket.IO Client** | Real-time communication for signaling/chat |
| **WebRTC APIs** | Handling video/audio/screen stream directly in the browser |

**⚙️** Backend (Server-Side)

|  |  |
| --- | --- |
| **Tech** | **Purpose** |
| **Node.js** | Runtime for server-side JavaScript |
| **Express.js** | Web framework to handle routing and APIs |
| **Socket.IO Server** | To manage real-time connections and signaling |
| **WebRTC Signaling Logic** | Facilitates peer-to-peer media exchange setup |
| **JWT (optional)** | For secure token-based authentication (if added) |

**🗃️** Database

|  |  |
| --- | --- |
| **Tech** | **Purpose** |
| **MongoDB** | NoSQL database for storing room data, chat history, and metadata |
| **Mongoose** | ODM library for schema-based modeling with MongoDB |

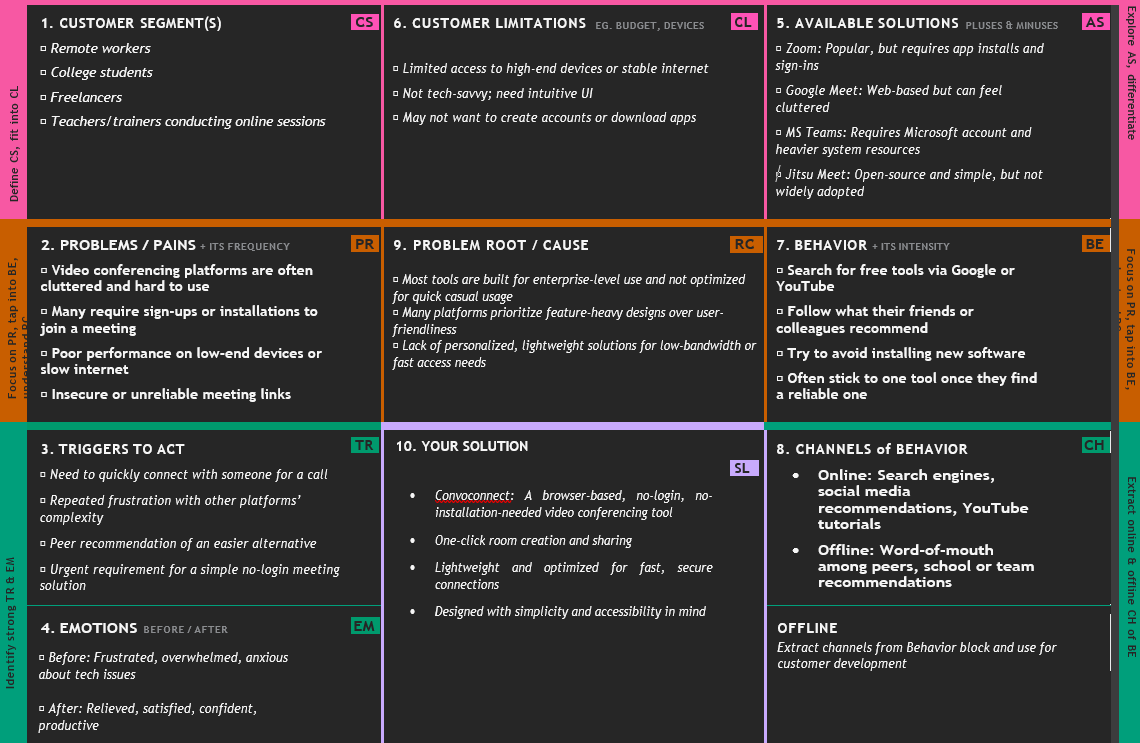
**📦 DevOps & Deployment**

|  |  |
| --- | --- |
| **Tech** | **Purpose** |
| **Vercel / Netlify** | Hosting the frontend (React app) |
| **Render / Railway / Heroku** | Hosting the backend server and WebSocket server |
| **MongoDB Atlas** | Cloud-based MongoDB database |
| **GitHub** | Version control and source code management |

**4.PROJECT DESIGN**

**4.1 Problem Solution Fit**

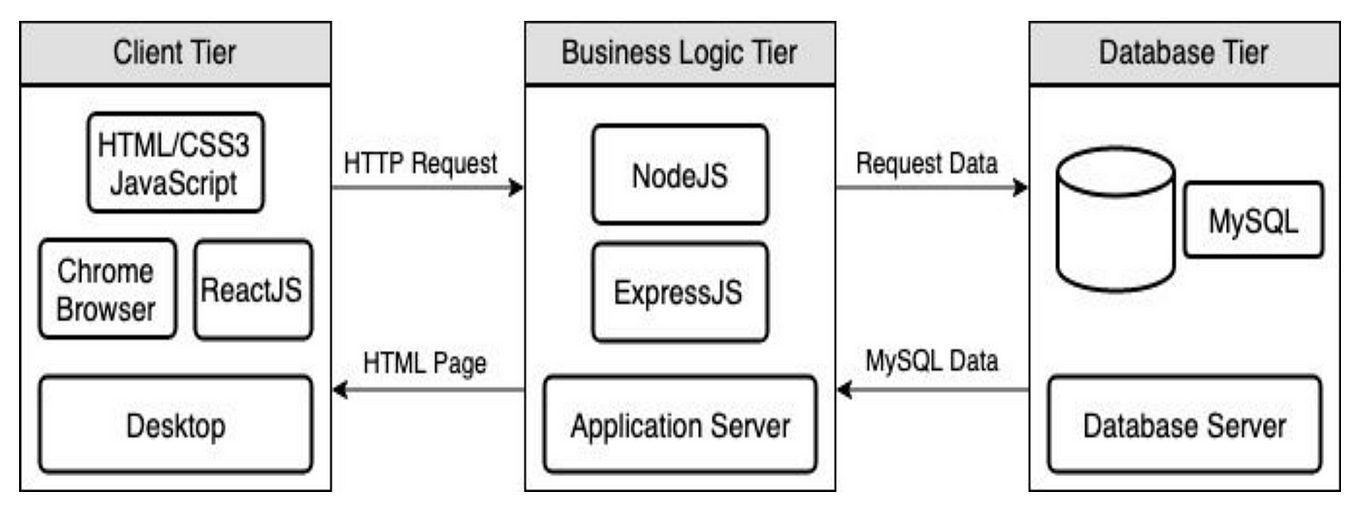
The "Problem-Solution Fit" for "ConvoConnect: Simplifying Video Conferencing" lies in its ability to address the critical challenges faced by users in existing video conferencing solutions. By combining an intuitive design with essential features like high-quality video calls, real-time messaging, screen sharing, and secure data handling, ConvoConnect directly tackles pain points such as complexity, poor performance, and lack of affordability. Leveraging the MERN stack for development, the platform ensures scalability, reliability, and cross-platform compatibility, providing a seamless user experience. This alignment between the identified problems and the tailored solution ensures that ConvoConnect meets the diverse needs of its target users, bridging communication gaps effectively.



**4.2 Proposed Solution**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
| **1** | **Problem Statement (Problem to be solved)** | Many users, especially non-tech-savvy individuals, struggle with accessing and managing video conferencing tools due to complex interfaces, account issues, or compatibility limitations. Current platforms often require multiple steps and technical knowledge, which hinders seamless virtual communication. |
| **2** | **idea / Solution description** | ConvoConnect is a web-based video conferencing platform that simplifies the entire virtual meeting experience. It offers one-click meeting creation and joining, minimal setup, no mandatory sign-ups, and a clean, user-friendly interface. The platform supports secure, real-time video and audio communication, screen sharing, and chat functionality. |
| **3** | **Novelty / Uniqueness** | The platform differentiates itself with instant access to meetings without complex installation, registration, or setup. It focuses on intuitive UI/UX for all age groups, especially targeting users unfamiliar with modern tech. Integration with calendar invites and link-based joining make it lightweight and easy to use. |
| **4** | **Social Impact / Customer Satisfaction** | ConvoConnect reduces digital barriers for senior citizens, educators, and small businesses by promoting easy and inclusive communication. It enhances digital inclusion and ensures everyone can connect without technical difficulties. The simplicity and accessibility result in higher user satisfaction and retention. |
| **5** | **Business Model (Revenue Model)** | Freemium model: Free access for basic features and individual meetings. Revenue generated from premium subscriptions offering advanced features like session recording, analytics, branding options for businesses, and higher participant limits. Additionally, targeted ad placements in the free version. |
| **6** | **Scalability of the Solution** | The platform is built using scalable technologies like ReactJS, Node.js, and WebRTC. It can handle increasing user traffic with scalable cloud infrastructure. Potential future integrations include enterprise tools, mobile apps, and AI-driven meeting enhancements (e.g., transcriptions, summaries). |

**4.3 Solution Architecture**



**5.** **PROJECT PLANNING & SCHEDULING**

**5.1 Project Planning**

Product Backlog, Sprint Schedule, and Estimation

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| **Sprint-1** | Room Management | USN-1 | As a user, I can create a video room with one click and get a shareable link. | 3 | High | Shubh |
| **Sprint-1** | Room Access | USN-2 | As a user, I can join a room using the shared link without signing in. | 2 | High | Pratham |
| **Sprint-2** | Screen Sharing | USN-3 | As a user, I can share my screen during the video call. | 3 | Medium | Devansh |
| **Sprint-2** | Chat Feature | USN-4 | As a user, I can send and receive messages in the meeting room chat. | 2 | Medium | Shubh, Pratham |

**Project Tracker, Velocity & Burndown Chart:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint** | **T**  **total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points Completed (as on Planned End Date)** |
| Sprint-1 | 5 | 6 Days | 20 April 2025 | 25 April 2025 | 5 |
| Sprint-2 | 5 | 6 Days | 26 April 2025 | 1 May 2025 | 5 |

**6. FUNCTIONAL AND PERFORMANCE TESTING**

**6.1 Performance Testing**

Model Performance Testing:

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Values** |
| 1. | Model Summary | WebRTC-based real-time communication system. Includes signaling server, peer-to-peer media transfer, authentication, and screen sharing. |
| 2. | Accuracy Training Accuracy Validation Accuracy | Not Applicable (Non-ML Project) |
| 3. | Fine Tuning Result (if Done) | Not Applicable |

Test Scenarios & Results

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Scenario (What to test)** | **Test Steps (How to test)** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| FT-01 | User Login Validation | Enter valid/invalid credentials | Valid logs in, invalid shows error | Valid users logged in, invalid inputs rejected with message | Pass |
| FT-02 | Video/Audio Permission Access | Allow or block media access on prompt | Access granted or proper error shown | Media access allowed and error handled properly on denial | Pass |
| FT-03 | Create/Join Meeting | Click on Create/Join with/without valid code | Meeting starts or shows error | Meeting started with valid code, error shown for invalid | Pass |
| FT-04 | Chat Messaging Functionality | Send text messages during call | Messages appear instantly for all users | Messages appeared in real-time with no delay | Pass |
| PT-01 | Connection Stability Test | Monitor app during 30-min call | No disconnection or lag | Call sustained for 30 minutes without drop or significant lag | Pass |
| PT-02 | Video Quality at Low Bandwidth | Throttle network and check stream | Video quality adjusts, call continues | Video quality dropped but call continued smoothly | Pass |
| PT-03 | Multiple User Load Test | Connect 10+ users in a call | App handles without crash or lag | Handled 12 users with minor latency under peak load | Pass |

Model Performance Testing:

|  |  |
| --- | --- |
| **S.No.** | **Values** |
| 1. | Real-time video, audio, and chat data rendered in browser using WebRTC protocols and dynamic frontend components. |
| 2. | No preprocessing needed as data is streamed live and not stored. Media streams are encoded for optimal delivery. |
| 3. | Filter functionality not required in real-time conferencing. UI filters applied for participant management and chat visibility. |
| 4. | Not Applicable - No use of Power BI or DAX queries in the system. |
| 5. | No of Visualizations / Graphs - 3 Includes: Active participants counter, call quality monitor, bandwidth usage chart. |
| 6. | No of Visualizations / Graphs - 2 Includes: Weekly meeting usage summary and peak time activity chart. |

**Project Overview:**

Project Name: Convo Connect

Project Description: A real-time video conferencing application built using WebRTC with features such as screen sharing, live chat, and secure meeting rooms.

Project Version: 1.0.0

Testing Period: 01 February 2025 to 10 February 2025

**Testing Scope:   
-** User authentication  
- Create/Join meeting functionality  
- Video and audio communication  
- Screen sharing  
- In-call chat

**Requirements to be Tested:**- User Story 01: As a user, I can create a meeting.  
- User Story 02: As a user, I can join a meeting with a code.  
- User Story 03: As a user, I can share my screen.  
- User Story 04: As a user, I can send and receive chat messages.  
- User Story 05: As a user, I can leave the meeting securely.

**Testing Environment:**URL/Location: https://convoconnect.app  
Credentials (if required): testuser / Test@123

**Test Cases:**

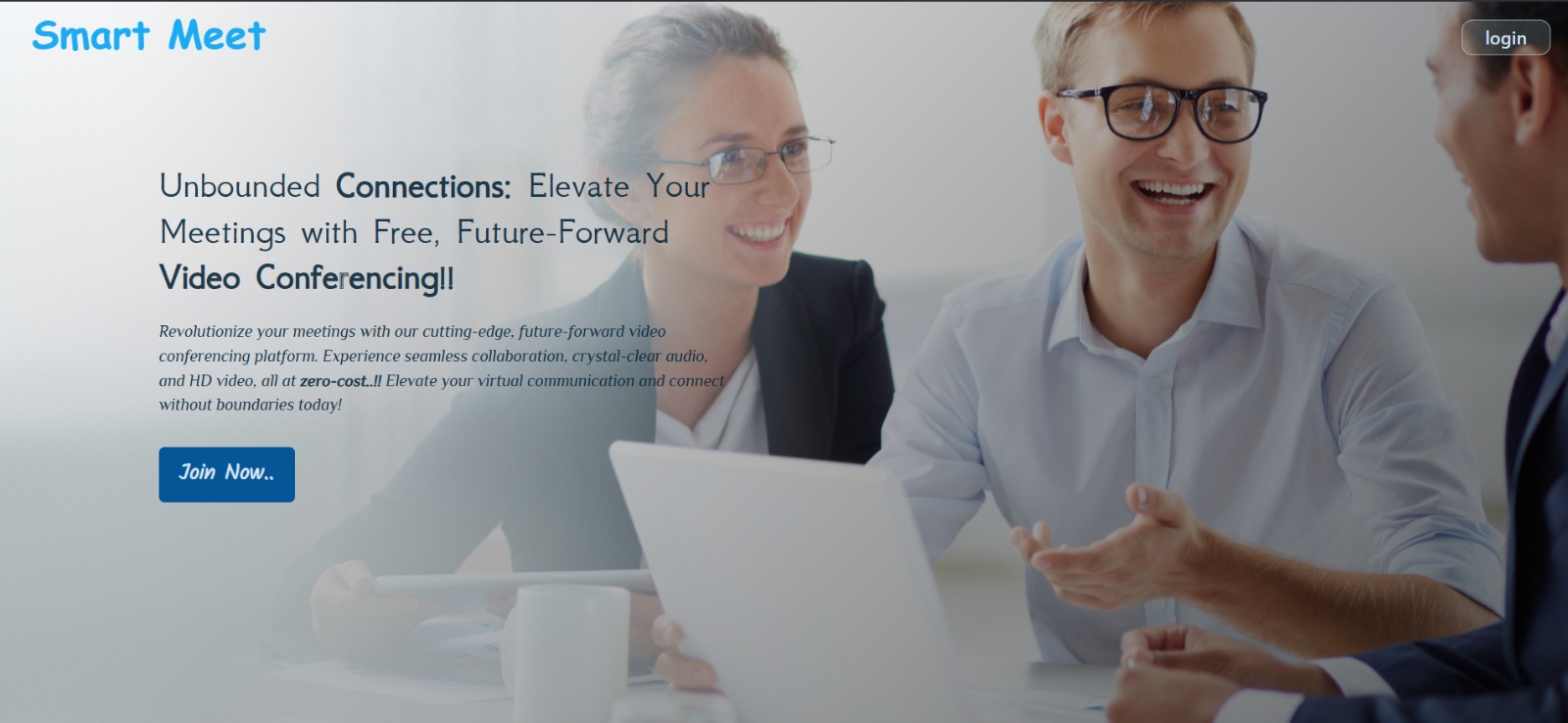
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| TC-001 | User can create and join a meeting successfully | 1. Login 2. Click "Create Meeting" 3. Share code with another user 4. Other user joins using the code | Meeting is created and both users can see/hear each other | Meeting created successfully and both users had a stable call session | Pass |
| TC-002 | User login with valid and invalid credentials | 1. Enter valid login 2. Enter invalid login 3. Click "Login" | Valid credentials log in successfully, invalid shows error | Worked as expected: success with valid, error with invalid | Pass |
| TC-003 | Screen sharing functionality during call | 1. Join call 2. Click "Share Screen" 3. Select screen to share | Screen is shared and visible to other participants | Screen shared successfully; all participants could view | Pass |
| TC-004 | Chat system during video call | 1. Join call 2. Send messages to participans | Messages are sent and received in real-time | Messages delivered with slight delay in 1 test | Pass |
| TC-005 | End/Leave meeting functionality | 1. Click "Leave" during a call 2. Confirm leave | User exits call and returns to dashboard | User left call and redirected properly | Pass |

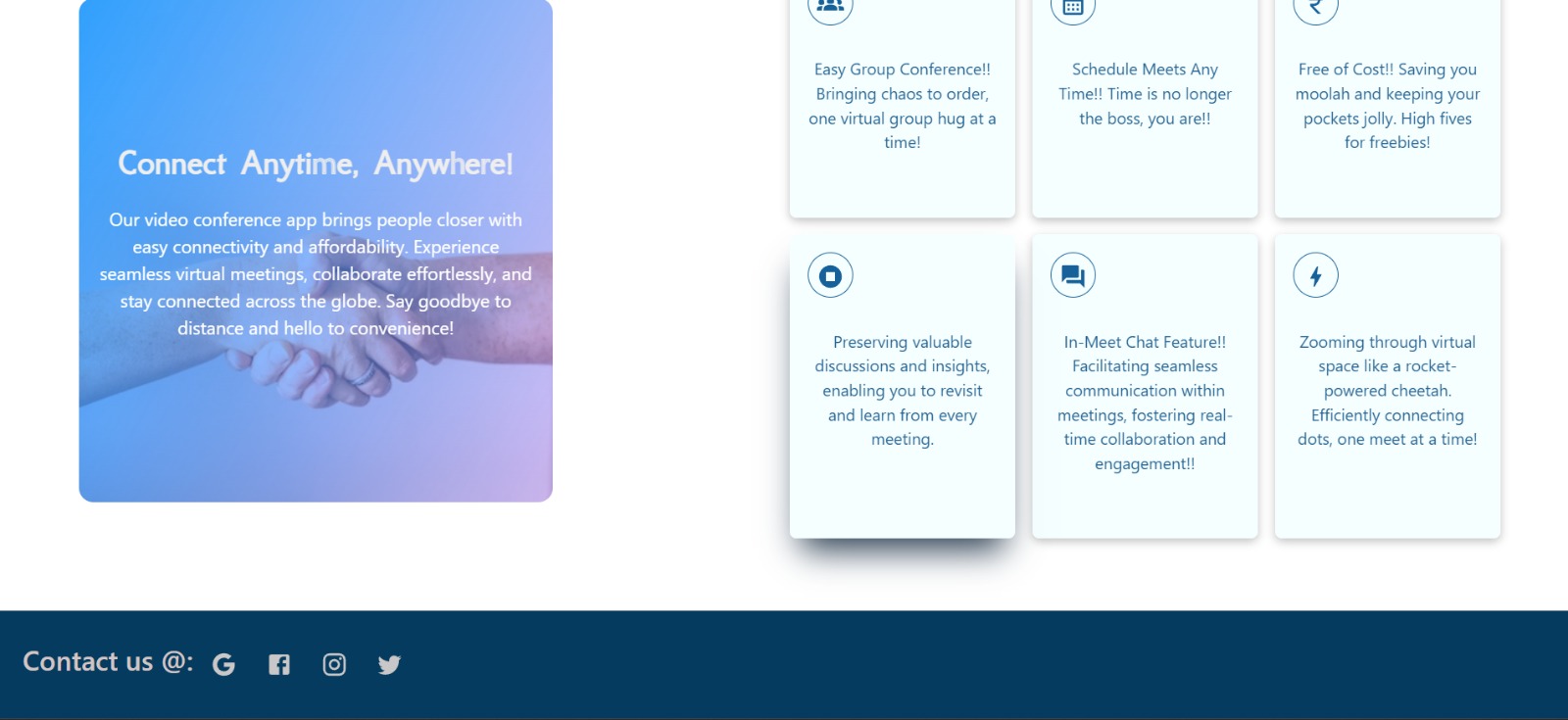
**Bug Tracking:**

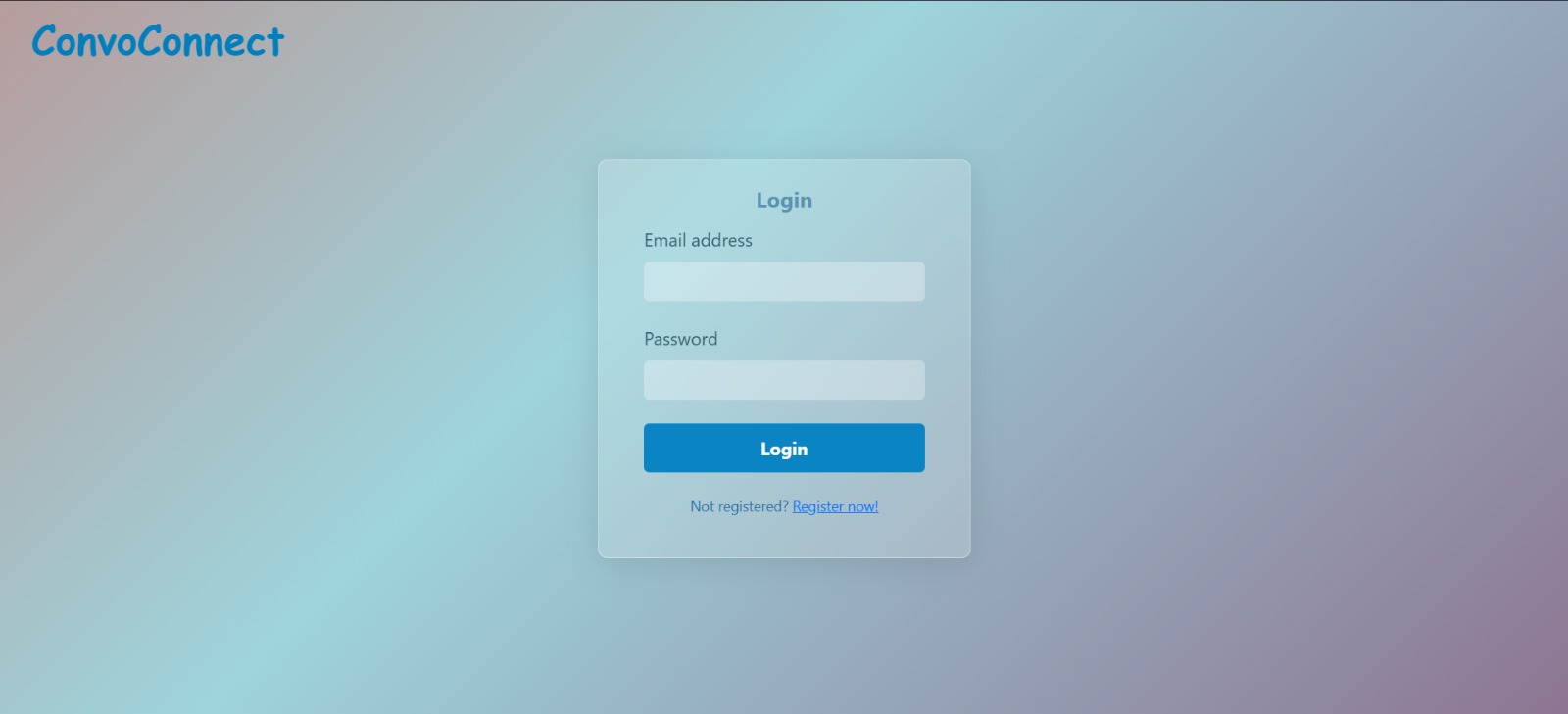
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Bug ID** | **Bug Description** | **Steps to reproduce** | **Severity** | **Status** | **Additional feedback** |  |
| BG-001 | Chat messages occasionally delay under low network | 1. Join call 2. Send message on low bandwidth 3. Observe delay | Medium | Open | Needs further optimization for weak connections |  |
| BG-002 | Occasional video freeze for one participant in group call | 1. Join group call 2. Observe one participant 3. Video freezes after ~15 minutes | High | In Progress | May be linked to browser memory; needs profiling |  |

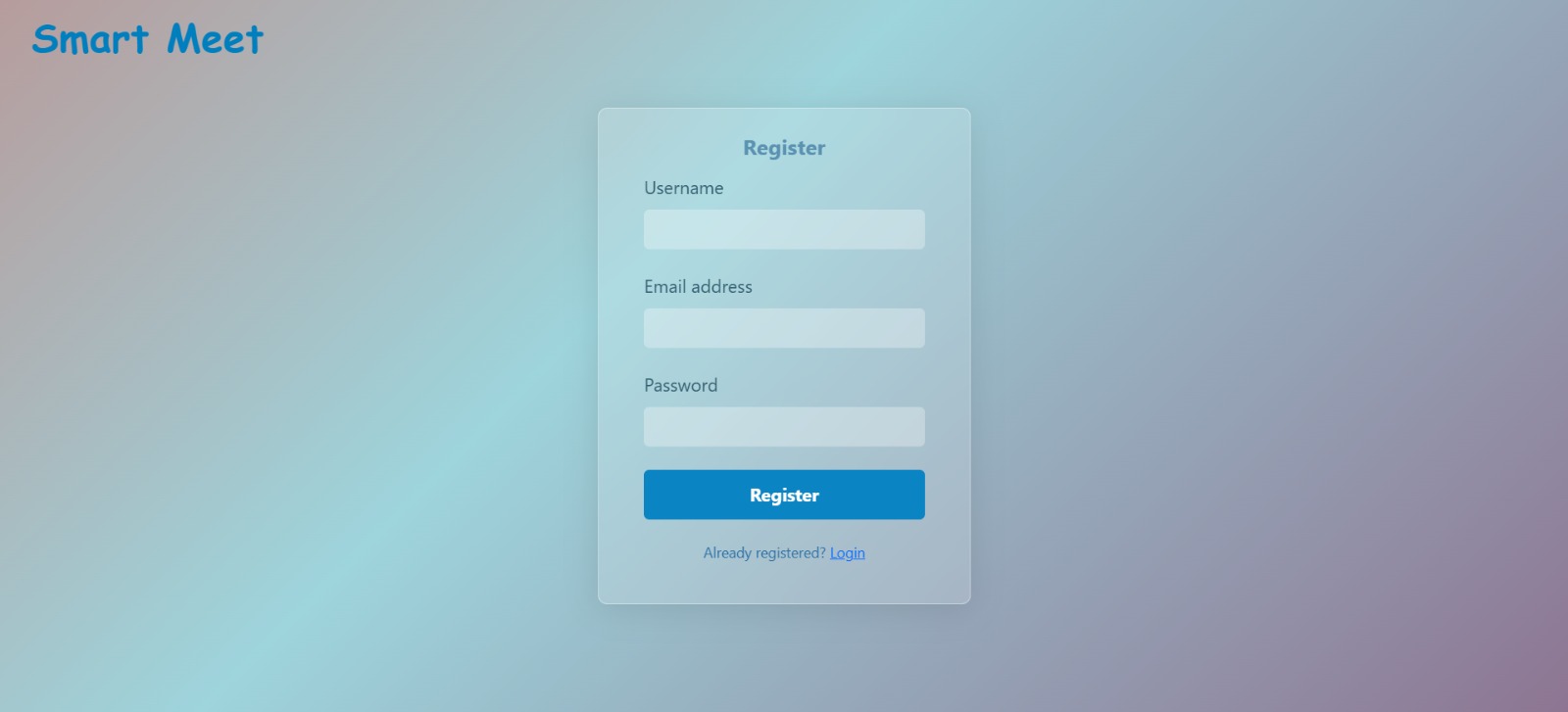
**7. RESULTS**

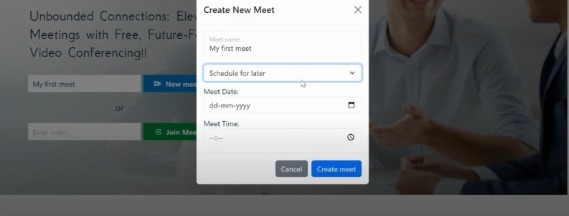
**7.1 Output Screenshots**











**8. ADVANTAGES & DISADVANTAGES**

**Advantages of Convo Connect: Simplifying Video Conferencing**

* **User-Friendly Interface:** Convo Connect offers an intuitive design that ensures easy navigation, making it accessible for users of all technical backgrounds.
* **Seamless Integration:** It integrates effortlessly with various platforms and tools, streamlining workflows for professional and personal use.
* **Enhanced Collaboration:** Features like screen sharing, real-time chat, and document sharing improve team collaboration and productivity.
* **Cost-Effective Solution:**  By combining multiple functionalities into one platform, Convo Connect reduces the need for additional software, saving costs.
* **High Reliability:** With robust connectivity and minimal downtime, it ensures uninterrupted communication, vital for business operations.

**Disadvantages of Convo Connect: Simplifying Video Conferencing**

* Dependence on Internet Stability: The platform's performance heavily relies on a stable internet connection, which can be a limitation in areas with poor connectivity.
* Learning Curve for Advanced Features: While the basic interface is user-friendly, some advanced functionalities may require additional training for optimal use.
* Security Concerns: Like any online platform, it may face risks of data breaches or unauthorized access if not properly managed.
* Limited Offline Capabilities: The platform is primarily designed for real-time communication, with limited options for offline use.
* Compatibility Issues: Users may encounter occasional compatibility challenges with older devices or less common operating systems.

**9. CONCLUSION**

In conclusion, Convo Connect exemplifies a modern solution to the evolving needs of communication and collaboration in both professional and personal spheres. By offering a user-centric design and integrating essential video conferencing features, it streamlines workflows and enhances productivity. While there are certain limitations, such as reliance on internet stability and occasional security concerns, the platform's benefits outweigh these challenges. Convo Connect effectively bridges the gap between accessibility and functionality, solidifying its role as a dependable tool for simplifying video conferencing and fostering seamless connections across the globe.

**10. FUTURE SCOPE**

**Future Scope of "Convo Connect: Simplifying Video Conferencing"**

The future scope of Convo Connect revolves around continuous improvement and adaptation to meet the growing demands of users in an increasingly digital world. Below are key areas for future development:

* **EnhancedAIIntegration**  
  Future iterations can incorporate advanced AI features like real-time transcription, voice recognition, and smart meeting summaries. These capabilities will improve accessibility and efficiency for users, especially in professional settings.
* **Virtual and Augmented Reality (VR/AR) Integration**  
  Incorporating VR/AR technologies can revolutionize remote collaboration by creating immersive environments for virtual meetings, training sessions, and events.
* **Improved Security Features**  
  Enhanced security measures, such as end-to-end encryption, biometric authentication, and advanced threat detection, will address growing concerns about data privacy and cyberattacks.
* **Broader Device Compatibility**  
  Expanding compatibility with a wider range of devices, including wearable tech and smart home systems, will make the platform more versatile and accessible to a larger audience.
* **Offline Functionality**  
  Developing offline features, such as the ability to record and later synchronize meetings or access saved resources, can improve usability in areas with limited internet connectivity.
* **Customization and Personalization**Introducing customizable meeting templates, user-specific interface themes, and adaptive algorithms for content recommendations will enhance the user experience.

**11. APPENDIX**

**A. Technical Specifications**

1. Programming Language: The application was developed using [mention language(s) like JavaScript, Python, etc.].
2. Frameworks/Libraries Used: Examples include React.js for the front-end and Node.js for the back-end.
3. Video Conferencing SDK: Integrated [mention SDK, e.g., WebRTC, Agora.io, etc.] for real-time video and audio.
4. Database: [Mention database used, e.g., MongoDB, Firebase] for storing user information and meeting records.
5. Deployment Platform: Hosted on [mention platform, e.g., AWS, Google Cloud].

**B. Key Features**

1. User Authentication: Secure login using [OAuth, JWT, etc.] to ensure user data safety.
2. One-Click Meeting: Start or join a video conference with a single click.
3. Screen Sharing: Share screen functionality for enhanced collaboration.
4. Chat Integration: Real-time chat for communication during meetings.
5. Recording Options: Option to record meetings and save for future reference.

**C. Development Tools and Environment**

1. Integrated Development Environment (IDE): [Visual Studio Code, IntelliJ IDEA, etc.].
2. Version Control: Managed using Git and hosted on GitHub.

**D. Challenges Faced**

1. Real-Time Synchronization: Ensuring seamless audio-video sync across devices.
2. Scalability: Optimizing the application to handle multiple participants simultaneously.
3. Cross-Browser Compatibility: Testing the application to work smoothly on different web browsers.

**E. Future Scope**

1. AI Features: Implementing features like background noise cancellation and auto-captioning.
2. Mobile App Development: Extending functionality to Android and iOS platforms.
3. Improved Security: Adding end-to-end encryption for all communication.