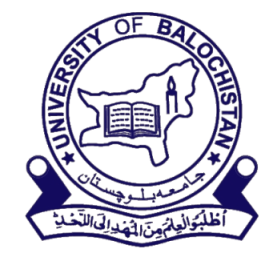
Web Based Video Conferencing App



BY

Muhammad Fahad 24

Ashar Asghar 23

Tooba Jahangir 44

**Supervisor: Sir Asfand Yar khan batazai**

A project report submitted to the

**Department of Computer Science**

In partial fulfilment of requirements for the degree of Bachelor of Science in

Computer Science at University of Balochistan, Quetta

**Session 2020 – 2024**

**Undertaking**

It is certified that this work titled “Web Based Video Conferencing App” is our own creation. The work has not been submitted for evaluation elsewhere. When content from other sources was used, it was appropriately recognized and referenced to.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

M.Fahad 24

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ashar Asghar 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tooba Jahangir 44

**Acknowledgements**

Primarily I would like to thank God for being able to accomplish this endeavour with success. Then I would like to thank our supervisor **Mr. Asfandyar khan**, whose helpful guidance has been crucial in helping me remedy this project and make it a full success; his comments and instructions have been an enormous contribution to the project's completion. Then I'd like to thank my parents and friends for their crucial remarks and assistance throughout the project's varied phases of completion.

**Dedication**

In the name of Allah, the Most Merciful and Gracious. Alhamdulillah, all glory be to Allah SWT, I accomplished my study assignment. This endeavour could not have been achieved without the aid, encouragement, guidance, and participation of my parents, employer, and co-workers. First and foremost, I want to thank my parents for their unwavering support, as well as my committed supervisor, **Sir Asfand Yar**, for all of his advice, suggestions, and assistance in completing this project effectively despite the short time frame. In a word, I'd want to thank my friends for their assistance in my quest to complete this project.

Thank you very much.

**Abstract**

The increasing demand for efficient remote communication tools has led to the development of various video conferencing solutions. This project introduces a comprehensive web-based video conferencing application specifically designed to enhance team collaboration through reliable and secure video communication. The application is built using React and Next.js, two cutting-edge web technologies that provide a solid foundation for developing dynamic, scalable, and performance-optimized web applications.

To create a user-friendly and visually appealing interface, the project incorporates Shadcn UI, which allows for the development of responsive and customizable components, ensuring a seamless user experience across different devices and screen sizes. The core functionality of real-time video conferencing is powered by the Stream API, a robust platform that enables high-quality video streams, ensuring smooth and uninterrupted communication among team members.

Security is a critical aspect of any online communication tool, and this project addresses this need through the integration of Clerk for user authentication. Clerk facilitates secure user registration, login, and session management, ensuring that only authorized users can access the video conferencing features. To further bolster security, the application includes an OTP (One-Time Password) verification system. Upon registration, users are required to verify their email addresses by entering a unique OTP sent to their inbox, thereby preventing unauthorized access and ensuring that user accounts are genuine.

This project successfully combines advanced web development practices with secure communication protocols, resulting in a video conferencing tool that is both user-centric and secure. It is particularly suited for professional environments where reliable and secure communication is paramount. By integrating these modern technologies, the application not only meets the technical challenges of real-time video communication but also provides a platform that prioritizes user experience and data security.

**Keywords**: Video Conferencing, Web Application, React, Next.js, Shadcn UI, Stream API, Clerk, OTP Verification, User Authentication, Team Collaboration.

**Table of Contents**

**Undertaking ……………………………………………………………………………………. ii**

**Acknowledgments ………………………………………………………………………………. iii**

**Dedication ………………………………………………………………………………………. iv**

**Abstract …………………………………………………………………………………………. v**

**1 INTRODUCTION ……………………………………………………………………...**

**1.1 Purpose ………………………………………………………………………………….**

**1.2 Benefit …………………………………………………………………………………...**

**1.3 Issues with Existing Software’s for Video Conferencing …………………………….**

**1.3.1 Security Concerns …………………………………………………………….**

**1.3.2 Weak Authentication Mechanism …………………………………………...**

**1.3.3 Poor User Experience ………………………………………………………...**

**1.3.4 Inconsistent Performance ……………………………………………………**

**1.3.5 High Cost ……………………………………………………………………...**

**1.3.6 Privacy Concerns ………………………………………………………………**

**1.4 Problem Associated with Software ………………………………………………………**

**1.4.1 High Resource Consumption …………………………………………………**

**1.4.2 Latency and Lag ………………………………………………………………**

**1.4.3 Performance Optimization Costs …………………………………………….**

**1.4.4 Cost Issues …………………………………………………………………….**

**1.4.5 Ongoing Maintenance …………………………………………………………**

**2 PROJECT PLAN ……………………………………………………………………….**

**2.1 Objectives ………………………………………………………………………………..**

**2.1.1 Scope ……………………………………………………………………………**

**2.1.2 Milestones ………………………………………………………………………**

**2.1.3 Time Line ……………………………………………………………………….**

**2.1.4 Resources ………………………………………………………………………..**

**2.1.5 Risk Management ………………………………………………………………**

**2.1.6 Communication Plan …………………………………………………………...**

**2.1.7 Quality Assurance ………………………………………………………………**

**2.1.8 Project Closure ………………………………………………………………….**

**2.2 Development Environment ………………………………………………………………**

**2.2.1 Tools and Technologies …………………………………………………………**

**2.2.2 Backend Development …………………………………………………………..**

**2.2.3 Deployment ………………………………………………………………………**

**2.2.4 Development Tools ………………………………………………………………**

**2.2.5 Configuration …………………………………………………………………….**

**2.2.6 Development Workflow ………………………………………………………….**

**2.2.7 Continuous Integration and Deployment (CI/CD) ……………………………..**

**3 DESIGN ……………………………………………………………………………………..**

**3.1 System Architecture ………………………………………………………………………...**

**3.2 Frontend ……………………………………………………………………………………..**

**3.3 Authentication Service ……………………………………………………………………....**

**3.4 Video Conferencing Service …………………………………………………………………**

**3.5 Deployment …………………………………………………………………………………..**

**3.6 High-Level System Design …………………………………………………………………..**

**3.7 Flow Chart ……………………………………………………………………………………**

**3.8 Design Considerations ……………………………………………………………………….**

**3.9 Deployment Strategy ………………………………………………………………………...**

**3.2 Features ……………………………………………………………………………………….**

**3.2.1 User Authentication and Registration ……………………………………………..**

**3.2.2 Real-Time Video Conferencing …………………………………………………….**

**3.2.3 User Dashboard ……………………………………………………………………..**

**3.2.4 Integration with Third-Party Services ……………………………………………..**

**3.2.5 Deployment and Continuous Integration …………………………………………..**

**3.2.6 User Experience and Accessibility ………………………………………………….**

**3.2.7 Security and Privacy …………………………………………………………………**

**3.3 Data Storage ……………………………………………………………………………………**

**3.3.1 Authentication Data Storage ………………………………………………………...**

**3.3.2 Video and Audio Data Storage ………………………………………………………**

**3.3.3 Deployment Data Storage ……………………………………………………………**

**3.3.4 User Session Management ……………………………………………………………**

**3.3.5 Static Asset Storage ……………………………………………………………………**

**Chapter No. 1**

**1- INTRODUCTION**

In today's digital age, remote collaboration has become a cornerstone of modern work environments. As teams increasingly rely on virtual communication, the demand for efficient and secure video conferencing solutions has grown exponentially. This project aims to address this need by developing a web-based video conferencing application specifically designed for team collaboration.

The application leverages a combination of modern web technologies to provide a seamless and user-friendly experience. At its core, the application is built using React and Next.js, which together offer a robust and scalable framework for developing dynamic web applications. To enhance the user interface, the project integrates Shadcn UI, ensuring that the platform is not only functional but also visually appealing and easy to navigate.

Real-time communication is facilitated by the Stream API, a powerful tool that supports high-quality video conferencing with low latency. The integration of Clerk for authentication ensures that user data is handled securely, with features such as email verification through OTP (One-Time Password) providing an additional layer of security. This verification process helps to prevent unauthorized access and ensures that only legitimate users can participate in video calls.

The result is a comprehensive video conferencing platform that balances performance, security, and usability, making it an ideal solution for teams looking to enhance their remote collaboration efforts. Whether for small teams or large organizations, this application is designed to meet the growing needs of remote work by providing a reliable and secure environment for virtual meetings.

* 1. **Purpose**

The purpose of this project is to develop a secure, reliable, and user-friendly web-based video conferencing application tailored to the needs of remote teams. With the shift towards remote and hybrid work environments, effective communication tools have become essential for maintaining productivity and collaboration. This application aims to address the challenges associated with remote communication by providing a platform that enables seamless video calls, real-time collaboration, and secure user authentication.

**The key objectives of the project are:**

**1. Enhance Team Collaboration:** By offering a platform for real-time video communication, the application helps teams stay connected, share ideas, and work together efficiently, regardless of their physical location.

**2. Ensure Security and Privacy:** The integration of robust authentication mechanisms, including email verification through OTP, ensures that only authorized users can access the platform, safeguarding sensitive information and maintaining user privacy.

**3. Deliver a Seamless User Experience:** Leveraging modern web technologies like React, Next.js, and Shadcn UI, the application is designed to be intuitive, responsive, and visually appealing, making it easy for users to navigate and utilize its features effectively.

**4. Support Scalability and Performance:** The use of scalable technologies and the Stream API for video conferencing ensures that the application can handle varying loads while maintaining high performance and video quality.

* 1. **Benefit**

The development of this web-based video conferencing application offers several significant benefits, particularly for teams and organizations that rely on remote communication. These benefits include:

**1. Improved Collaboration:** The application fosters effective communication among team members, regardless of their geographic locations. By providing a platform for real-time video calls, it enables teams to collaborate more efficiently, share ideas instantly, and make decisions faster.

**2. Enhanced Security:** Security is a top priority, with features like OTP-based email verification ensuring that only authorized users can access the platform. This minimizes the risk of unauthorized access, protecting sensitive information and maintaining the integrity of team communications.

**3. User-Friendly Interface:** The integration of Shadcn UI ensures that the application is not only functional but also easy to use. With a clean and intuitive interface, users can quickly navigate the platform, participate in meetings, and manage their video conferencing sessions without any technical hurdles.

**4. Scalability and Performance:** Built on a robust stack of technologies, including React, Next.js, and the Stream API, the application is designed to scale with the needs of any team or organization. Whether it’s a small start-up or a large enterprise, the platform can handle varying levels of usage while maintaining high performance and video quality.

**5. Cost-Effective Solution:** By leveraging modern web technologies and deploying the application on platforms like Vercel, the project offers a cost-effective solution for organizations looking to implement or enhance their video conferencing capabilities without the need for expensive infrastructure or software licenses.

By delivering these benefits, the video conferencing application not only meets the immediate needs of remote teams but also positions itself as a vital tool for future-ready organizations.

* 1. **Issues with Existing Software for Video Conferencing**

While video conferencing has become an essential tool for remote communication, many existing solutions face a number of challenges that can impact user experience and overall effectiveness. These issues include:

**1.3.1 Security Concerns:** Data Breaches and Unauthorized Access: Many video conferencing platforms have been plagued by security vulnerabilities, leading to data breaches and unauthorized access to meetings. This raises concerns about the privacy and confidentiality of sensitive information shared during video calls.

**1.3.2 Weak Authentication Mechanisms:** Some platforms rely on basic or outdated authentication methods, making it easier for unauthorized users to gain access to private meetings. The lack of robust authentication can compromise the security of the entire communication system.

**1.3.3 Poor User Experience:** Complex Interfaces: Many existing platforms have cluttered and unintuitive interfaces, making it difficult for users to navigate and access essential features. This can be particularly challenging for non-technical users, leading to frustration and decreased productivity.

**1.3.4 Inconsistent Performance:** Users often experience issues with video quality, audio lag, and connection instability, especially when the platform is under heavy load. These performance issues can disrupt meetings and hinder effective communication.

**1.3.5 High Costs:** Scaling existing platforms to accommodate more users often comes with high costs, either through expensive licensing models or the need for additional infrastructure. This can be prohibitive for smaller teams or organizations with limited budgets.

**1.3.6 Privacy Concerns:** Some video conferencing services have been criticized for their data collection practices, where user data is collected, stored, and even shared with third parties without proper consent. This raises significant privacy concerns, especially in sensitive or regulated industries.

By addressing these issues, your project aims to provide a more secure, user-friendly, scalable, and customizable video conferencing solution that better meets the needs of modern teams and organizations. Let me know if you'd like to expand on any of these point!

**1.4 Problem Associated with Software**

Software development and deployment often come with significant challenges related to performance and cost. These factors can impact both the user experience and the overall viability of a software solution, particularly in resource-intensive applications like video conferencing. Below are the key issues related to performance and cost:

**1.4.1High Resource Consumption.** Video conferencing software typically requires high bandwidth to maintain video and audio quality. Users with limited or unstable internet connections may experience lag, dropped calls, or reduced video quality, significantly impacting the effectiveness of the communication.

**1.4.2 Latency and Lag:** In applications like video conferencing, even slight delays can disrupt the flow of conversation. High latency, caused by inefficient code, overloaded servers, or poor network conditions, can result in out-of-sync audio and video, leading to a frustrating user experience.

**1.4.3 Performance Optimization Costs:**

**Complexity of Optimization:** Achieving optimal performance often requires complex optimizations, such as code refactoring, server tuning, or the use of advanced technologies like content delivery networks (CDNs). These optimizations can be time-consuming and costly, requiring specialized skills and resources.

**Maintenance of Performance Levels:** Maintaining high performance over time requires continuous monitoring, regular updates, and addressing emerging issues like security patches, which can add to the overall operational costs.

**1.4.4 Cost Issues:**

Developing high-performance software often requires significant upfront investment in skilled developers, advanced tools, and technologies. The need to ensure high performance from the outset can drive up development costs, particularly when dealing with complex applications like video conferencing.

**1.4.5 Ongoing Maintenance:** Software requires continuous maintenance to ensure it remains secure, compatible with new technologies, and optimized for performance. Ongoing maintenance can be expensive, particularly if the software is resource-intensive or if it requires frequent updates to address performance issues.

**Chapter No. 2**

**2- Project Plan**

The project involves developing a secure, scalable, and user-friendly web-based video conferencing application tailored to the needs of remote teams. The application will include features such as real-time video communication, secure user authentication, and seamless integration with existing tools and workflows.

**2.1 Objectives**

Develop a robust video conferencing platform that supports real-time communication.

Ensure high levels of security and privacy for users through OTP-based email verification and secure data handling.

Create a responsive, intuitive, and aesthetically pleasing user interface using Shadcn UI.

Leverage modern web technologies (React, Next.js, Stream API) to ensure scalability and high performance.

Deploy the application on reliable cloud infrastructure (e.g., Vercel) with continuous integration and deployment (CI/CD) practices.

**2.1.1 Scope**

Core Features: User authentication via Clerk, video conferencing via Stream API, real-time messaging, and responsive UI.

**Security:** OTP-based email verification, encrypted data transmission, and secure user session management.

**User Interface:** Customizable and accessible design using Shadcn UI with responsive layouts for various devices.

**Deployment:** Cloud-based deployment with Vercel, including automatic scaling, load balancing, and monitoring.

**Integration:** Seamless integration with third-party tools and APIs as required by the project.

**Documentation:** Comprehensive documentation including system design, user guides, and API references.

**2.1.2 Milestones**

|  |  |  |  |
| --- | --- | --- | --- |
| Milestone | Description | Duration | Deliverables |
| M1: Project Initiation | Define project scope, objectives, and requirements. | 1 week | Project charter, requirements document. |
| M2: System Design | Create a high-level system design, including architecture diagrams and technology stack decisions. | 2 weeks | System design document, architecture diagrams. |
| M3: UI/UX Design | Develop the user interface design, including wireframes and prototypes using Shadcn UI. | 2 weeks | UI/UX design mockups, interactive prototypes. |
| M4: Backend Development | Develop the backend, including user authentication with Clerk and video conferencing integration with Stream API. | 4 weeks | Backend codebase, API documentation. |
| M5: Frontend Development | Implement the frontend using React and Next.js, integrating with the backend and ensuring responsive design. | 4 weeks | Frontend codebase, integration tests. |
| M6: Security Implementation | Implement security features, including OTP verification, data encryption, and secure session management. | 2 weeks | Security protocols, testing reports. |
| M7: Testing and QA | Conduct thorough testing, including unit, integration, and performance tests. Address any bugs or issues. | 3 weeks | Test plans, bug reports, QA approval. |
| M8: Deployment | Deploy the application on Vercel, set up CI/CD pipelines, and monitor performance. | 2 weeks | Deployed application, CI/CD pipeline setup. |
| M9: Documentation | Prepare user manuals, API documentation, and system design documentation. | 2 weeks | Comprehensive documentation package. |
| M10: Final Review and Launch | Conduct a final review, gather feedback, and officially launch the application. | 1 week | Final project report, launch plan. |

**2.1.3 Timeline**

The total duration of the project is estimated to be approximately 21 weeks, considering potential delays and buffer time.

**2.1.4 Resources**

**Team:** Project Manager, Frontend Developer, Backend Developer, UI/UX Designer, Security Specialist, QA Engineer.

**Tools and Technologies:** React, Next.js, Shadcn UI, Stream API, Clerk, Vercel, GitHub, CI/CD tools (e.g., GitHub Actions), testing frameworks (e.g., Jest, Cypress).

**Budget:** Costs associated with cloud infrastructure, third-party services (Clerk, Stream API), development tools, and team resources.

**2.1.5 Risk Management**

**Security Risks:** Mitigated by implementing strong encryption, OTP verification, and regular security audits.

**Performance Risks:** Addressed by optimizing code, using efficient algorithms, and employing scalable cloud infrastructure.

**Scope Creep:** Managed by strict adherence to the project scope, regular project reviews, and stakeholder communication.

**Timeline Delays:** Buffer time included in the timeline, regular progress tracking, and proactive issue resolution.

**2.1.6 Communication Plan**

**Stakeholder Meetings:** Bi-weekly meetings with stakeholders to review progress, address concerns, and gather feedback.

**Team Meetings:** Weekly team meetings to discuss progress, assign tasks, and resolve any issues.

**Status Reports:** Weekly status reports to be shared with all stakeholders, detailing progress, upcoming tasks, and any risks or issues.

**2.1.7 Quality Assurance**

**Code Reviews:** Regular peer code reviews to ensure code quality and adherence to best practices.

**Testing:** Comprehensive testing strategy including unit, integration, performance, and security testing.

**User Feedback:** Collect user feedback during beta testing phases to identify and address usability issues.

**2.1.8 Project Closure**

**Final Deliverables:** Complete project documentation, final codebase, and a deployed application.

**Post-Launch Support:** Initial support period to address any post-launch issues or user feedback.

**Project Review:** Conduct a project retrospective to evaluate successes, lessons learned, and areas for improvement.

**2.2 Development Environment**

The development environment for your video conferencing application is structured to ensure seamless integration of various tools and technologies, enabling efficient development, testing, and deployment processes.

**2.2.1 Tools and Technologies**

**React:** A popular JavaScript library for building user interfaces. React is used to create interactive and reusable UI components.

**Next.js:** A React framework that provides server-side rendering, static site generation, and optimized performance. It's used to build the frontend architecture of the application.

**Shadcn UI:** A design system used to create a cohesive and modern user interface, ensuring a consistent look and feel across the application.

**2.2.2 Backend Development:**

**Next.js API Routes:** Server-side functionality is handled within Next.js using API routes, which provide backend capabilities without the need for a separate server framework.

**Stream API:** A third-party API used for implementing real-time video conferencing features. Stream API handles video streaming, user management, and live interactions.

**Authentication:**

**Clerk:** A secure and customizable authentication service that handles user sign-ups, logins, and OTP-based email verification. Clerk simplifies user management and session handling.

**2.2.3 Deployment:**

**Vercel:** A cloud platform optimized for Next.js applications, Vercel handles the deployment of the application. It provides automatic scaling, serverless functions, and continuous deployment from the GitHub repository.

**GitHub:** Used for version control and collaboration. The codebase is managed in GitHub repositories, and changes are tracked through branches and pull requests.

**2.2.4 Development Tools:**

**Visual Studio Code (VSCode):** The primary code editor used for development. VSCode is equipped with extensions for React, Next.js, ESLint, Prettier, and Git integration to streamline the development process.

**2.2.5 Configuration**

**Node.js Version:** The project uses the latest LTS version of Node.js to ensure compatibility with all dependencies and tools.

**npm or Yarn:** Either npm or Yarn is used to manage the project's dependencies, ensuring consistent installation and versioning across different environments.

**Environment Variables:** Sensitive data such as API keys, Clerk credentials, and Stream API keys are managed through environment variables. These are defined in a `. env.local` file for local development and securely managed on Vercel for production.

**2.2.6 Development Workflow**

**Main Branch:** The `main` branch is used for stable, production-ready code.

**Development Branch:** The `dev` branch serves as the integration branch where features and fixes are merged before being pushed to `main`.

**Feature Branches:** Individual features are developed in their own branches (e.g., `feature/video-calling`), allowing for isolated changes and easier code reviews.

**GitHub Pull Requests:** Changes are submitted via pull requests, where they are reviewed by team members for quality, security, and adherence to coding standards.

**Testing:**

**Unit Testing:** React Testing Library is used for unit testing components, ensuring they behave as expected in isolation.

**Integration Testing:** Tests are conducted to ensure that different parts of the application work together correctly, particularly the interaction between the frontend, Clerk, and Stream API.

**2.2.7 Continuous Integration and Deployment (CI/CD):**

**Vercel Integration:** The application is deployed automatically to Vercel whenever changes are pushed to the `main` branch. Vercel handles the build process and deployment, ensuring that the latest version of the app is always available.

**Chapter No. 3**

**3- Design**

The project design of the video conferencing application aims to provide a robust, scalable, and user-friendly solution for real-time video communication. The design integrates various technologies and services to ensure seamless functionality, security, and performance. This document outlines the high-level architecture, components, workflows, and design considerations for the application.

**3.1 System Architecture**

The system architecture is designed to handle user interactions, authentication, video streaming, and deployment effectively. The architecture consists of the following key components:

**3.2 Frontend (User Interface)**

- **Technologies**: Next.js, React, Shadcn UI

**- Role**: The frontend serves as the user interface, allowing users to interact with the application. It is responsible for rendering the UI components, managing user input, and communicating with the backend.

**- Backend**

**- Technologies:** Next.js API Routes

**- Role:** The backend handles the application logic, including user authentication, session management, and API integrations. It interacts with third-party services to authenticate users and manage video conferencing sessions.

**3.3 Authentication Service**

**- Service:** Clerk

**- Role:** Clerk manages user authentication, including registration, login, and OTP-based email verification. It ensures secure access to the application.

**3.4 Video Conferencing Service**

**- Service:** Stream API

**- Role:** The Stream API handles real-time video and audio communication between users. It manages the establishment of video calls, streaming, and connection management.

**3.5 Deployment**

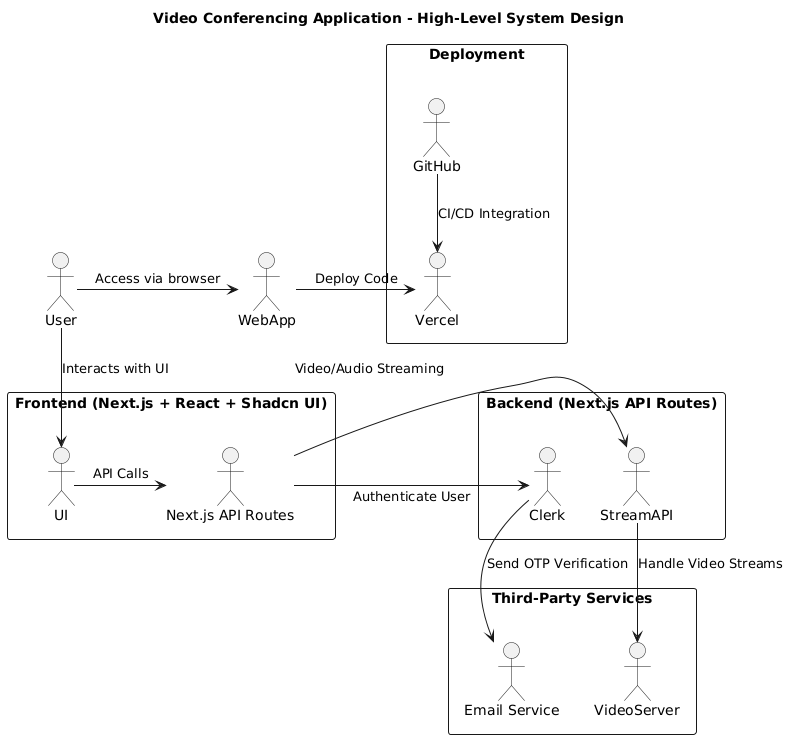
**- Services:** Vercel, GitHub

**- Role:** Vercel is used for deploying the application, with GitHub managing version control. The CI/CD pipeline ensures that the application is continuously integrated and deployed whenever updates are made.

**3.6 High-Level System Design**

The high-level system design provides a visual representation of the overall architecture of the video conferencing application. It illustrates how the different components interact with each other and with third-party services.

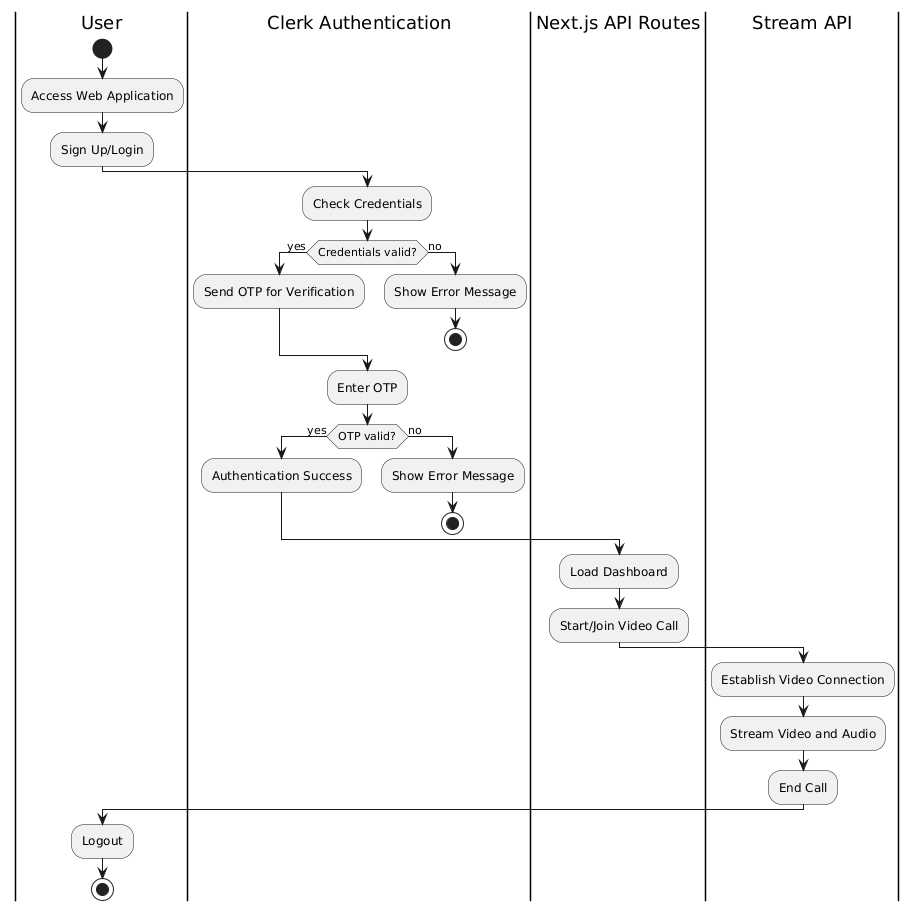
**Diagram:**



**3.7 Flowchart**

The flowchart provides a step-by-step visual representation of the user journey within the application, from accessing the app to initiating a video call.

**Diagram:**



**3.8 Design Considerations**

The following design considerations were taken into account during the development of the application:

**1. Scalability:** The system is designed to handle multiple concurrent users, with the ability to scale as needed based on user demand.

**2. Security:** User authentication is managed by Clerk, ensuring that sensitive data is securely handled. OTP verification adds an extra layer of security.

**3. Performance:** The use of Next.js ensures efficient server-side rendering and fast load times. The Stream API provides low-latency video and audio streaming, ensuring a smooth user experience.

**4. User Experience:** The UI is built with Shadcn UI, ensuring a modern, responsive, and intuitive interface. The application is designed to be easy to use, with clear navigation and feedback.

**3.9 Deployment Strategy**

The application is deployed using Vercel, with continuous integration and deployment managed through GitHub. Vercel ensures that the application is always up-to-date with the latest code changes. The CI/CD pipeline automates testing, building, and deployment processes, reducing the risk of errors and ensuring quick rollouts of new features.

**3.2 Features**

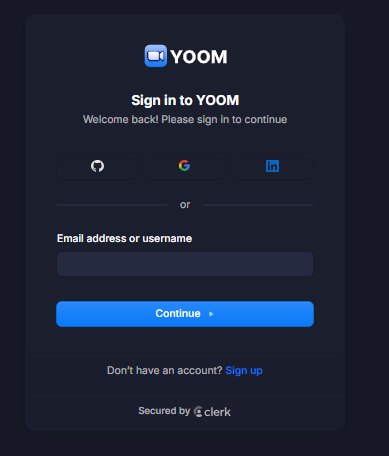
The video conferencing application is designed to provide a comprehensive and user-friendly platform for real-time communication. The following are the key features of the application:

**3.2.1 User Authentication and Registration**

Secure Registration/Login: Users can securely register and log in using their email and password. The application leverages Clerk for managing user authentication, ensuring that user data is handled securely.

OTP Verification: Upon registration, users receive a One-Time Password (OTP) via email for verification. This adds an extra layer of security, ensuring that only legitimate users gain access to the platform.

**Diagram:**



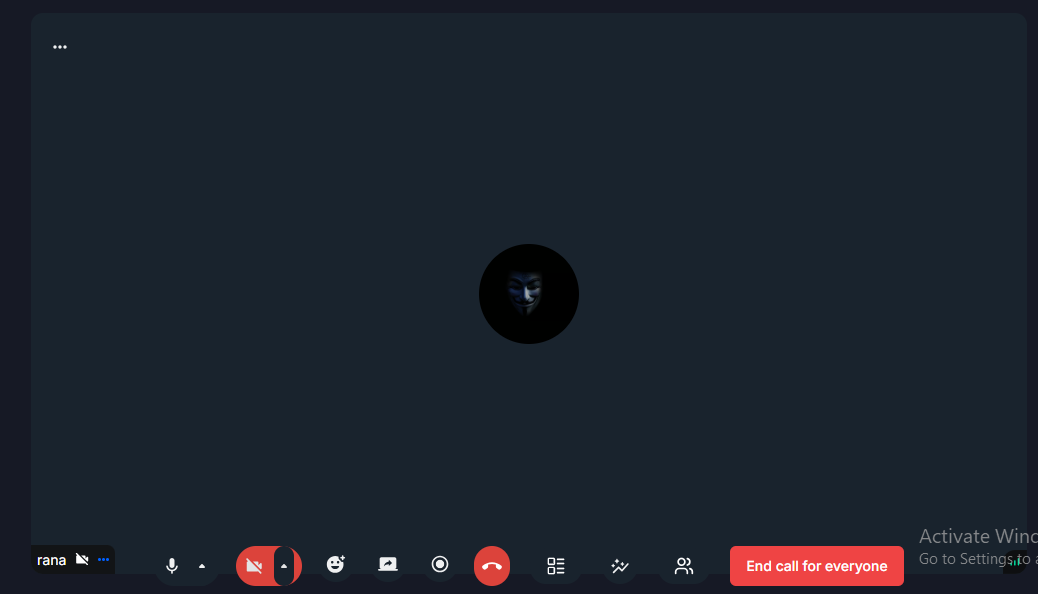
**3.2.2 Real-Time Video Conferencing**

High-Quality Video and Audio Streaming: The application utilizes the Stream API to provide high-quality, low-latency video and audio streaming, ensuring clear communication during meetings.

Multi-Participant Video Calls: The platform supports video calls with multiple participants, making it ideal for team meetings, virtual classrooms, and other collaborative activities.

**Screen Sharing:** Users can share their screens during a video call, facilitating presentations, demonstrations, and collaborative work.

**Diagram:**

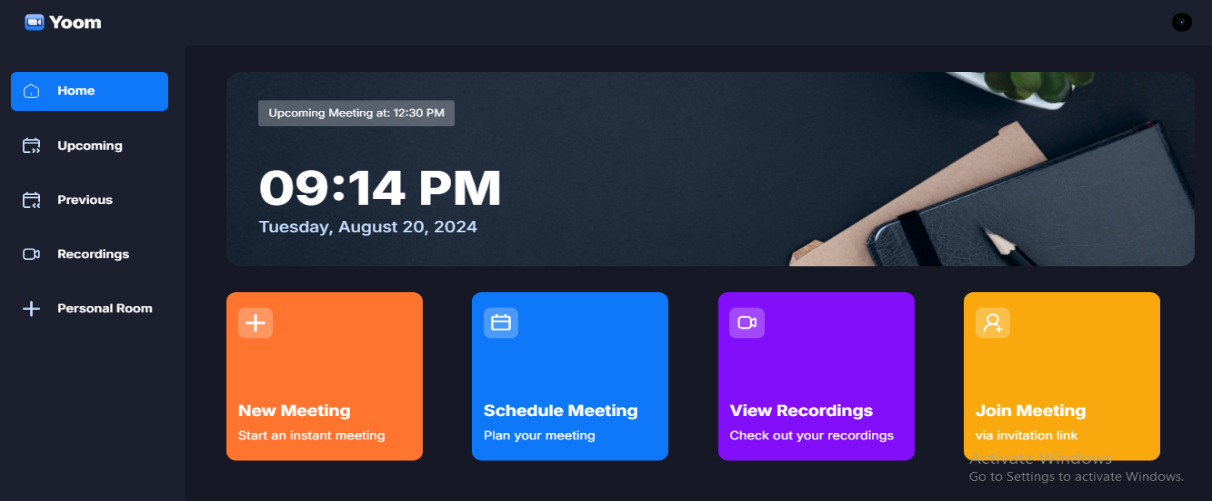
****

**3.2.3 User Dashboard**

Intuitive Interface: The user dashboard is designed with Shadcn UI, providing a modern, clean, and responsive interface. Users can easily navigate through the features and access their scheduled meetings.

Meeting Management: Users can schedule, start, or join meetings directly from the dashboard. The dashboard also displays upcoming meetings and past meeting history.

**Diagram:**

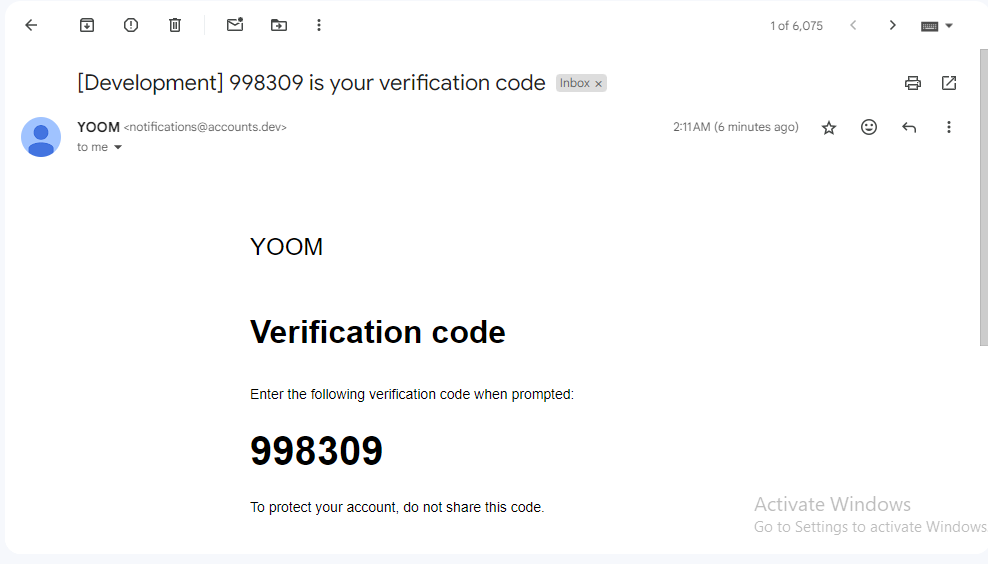


**3.2.4 Integration with Third-Party Services**

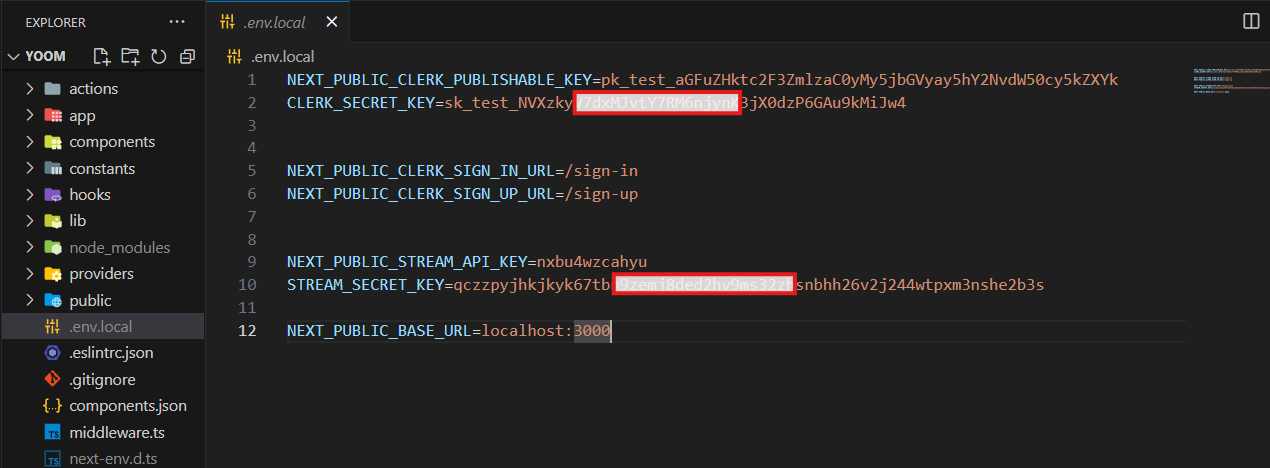
**Email Notifications:** The application integrates with an email service to send notifications, such as OTPs for verification and reminders for upcoming meetings.

**API Integrations:** The backend is designed to integrate seamlessly with third-party APIs, including the Stream API for video conferencing and Clerk for authentication. This ensures that the application is both robust and flexible.

**Diagram Email Notifications:**



**Diagram Api Integrations:**

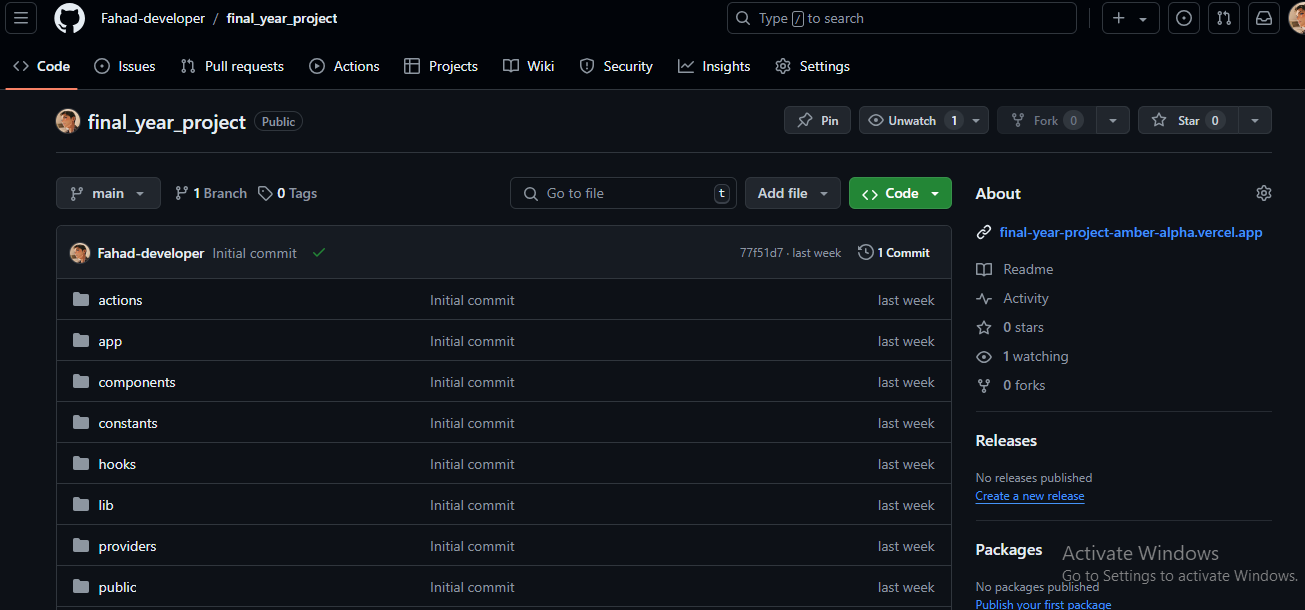


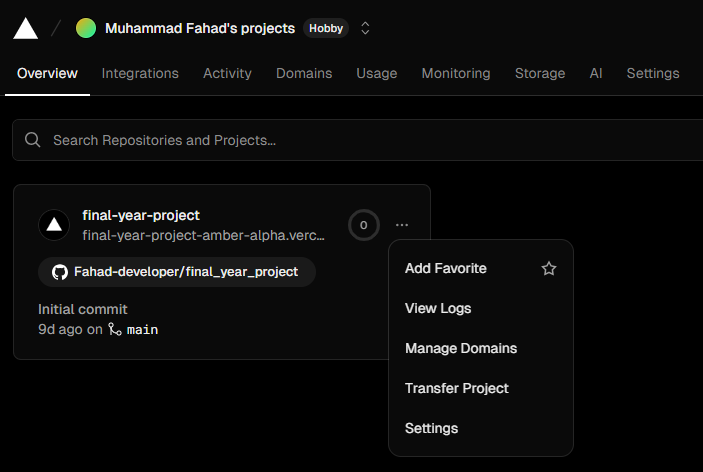
**3.2.5 Deployment and Continuous Integration**

Seamless Deployment: The application is deployed using Vercel, a platform known for its efficient deployment and scalability. This ensures that the application is always available and can handle varying user loads.

Continuous Integration/Continuous Deployment (CI/CD): The application is integrated with GitHub for version control. The CI/CD pipeline automates the process of testing, building, and deploying the application, ensuring that updates are rolled out quickly and without errors.

**Diagram Continuous:**



**Diagram** **Deployment**: 

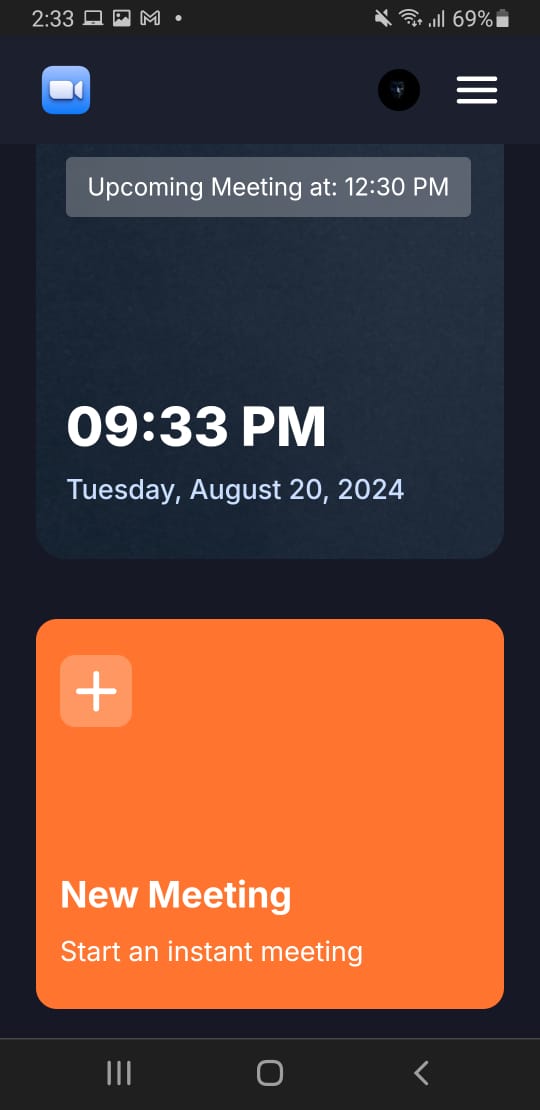
**3.2.6 User Experience and Accessibility**

Responsive Design: The application is designed to be fully responsive, ensuring that users have a seamless experience whether they are accessing the platform from a desktop, tablet, or mobile device.

Cross-Browser Compatibility: The application is compatible with all modern web browsers, providing a consistent experience across different platforms.

User Feedback and Support: The application includes features for users to provide feedback and access support, ensuring continuous improvement and user satisfaction.

**Diagram:**



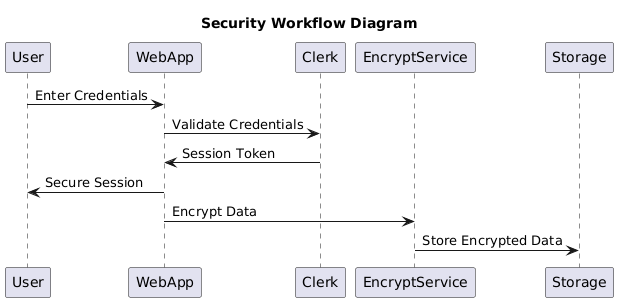
**3.2.7 Security and Privacy**

**Data Encryption:** All sensitive data, including user credentials and video streams, are encrypted to ensure privacy and security.

**Session Management**: The application manages user sessions securely, with features like automatic logout after a period of inactivity to prevent unauthorized access.

**Compliance:** The application is designed with compliance in mind, adhering to relevant data protection regulations to ensure the privacy of user data.

**Diagram:**

****

**3.3 Data Storage**

In this video conferencing application, the architecture is designed to minimize the need for direct data storage by leveraging external services. Since the application does not use any external database, the data storage responsibilities are distributed across various third-party services that handle authentication, user data, and video streaming.

**3.3.1 Authentication Data Storage**

**Service Used:** Clerk

Data Stored: Clerk manages and stores user authentication data, including user credentials (email and password), session tokens, and OTPs for verification.

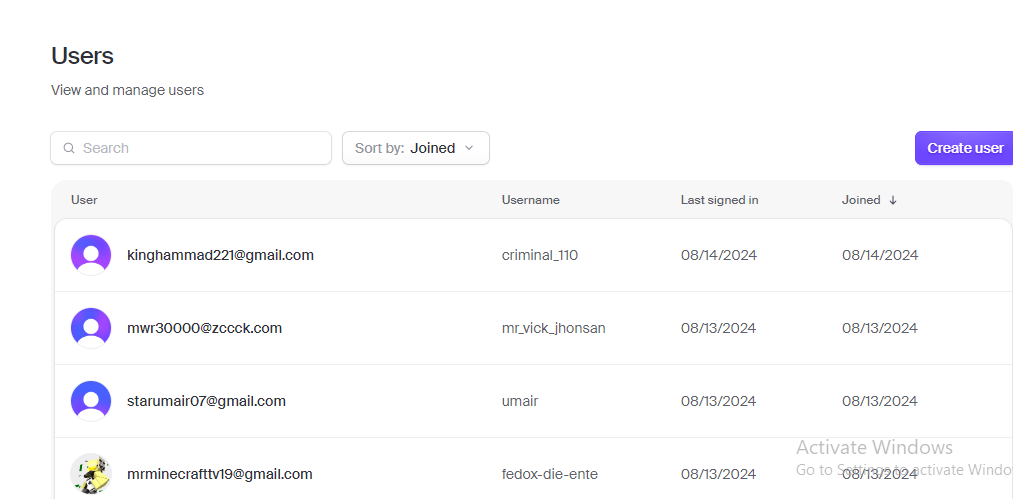
**Storage Details:** Clerk securely stores user credentials and manages session data in its own secure, encrypted storage. This data is not stored directly within the application, reducing the risk associated with handling sensitive user information.

**Security Considerations:**

Clerk uses encryption to protect sensitive user information during both storage and transmission.

Compliance with GDPR and other data protection regulations is ensured by Clerk, providing peace of mind regarding user data privacy.

**Figure:**



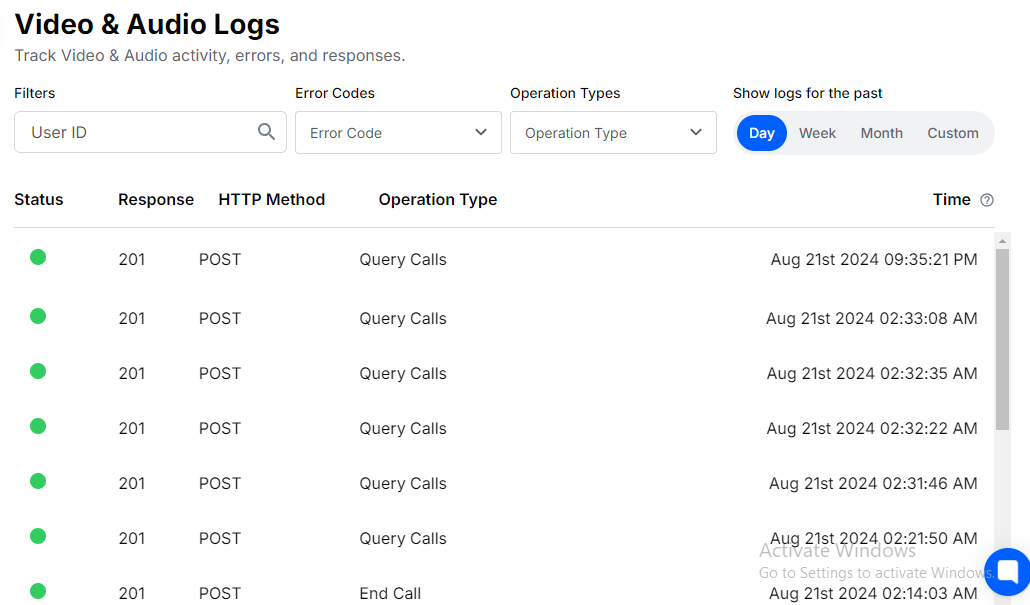
**3.3.2 Video and Audio Data Storage**

**Service Used**: Stream API

**Data Stored:** The Stream API handles real-time video and audio data during live conferencing sessions.

**Storage Details:** Video and audio streams are typically not stored permanently unless the application is configured to record sessions. In this case, the Stream API handles temporary data storage to manage the streaming process and ensure smooth delivery. If recording is enabled, the storage and access to the recorded sessions would be managed by Stream or another designated service.

**Figure:**



**Security Considerations:**

The Stream API ensures that video and audio data are transmitted securely between users, with appropriate encryption protocols in place to prevent unauthorized access.

Temporary storage of streaming data is handled within the Stream infrastructure, which complies with industry standards for data security and privacy.

**3.3.3 Deployment Data Storage**

**Services Used:** Vercel, GitHub

**Data Stored:** Vercel and GitHub manage the application’s codebase, deployment configurations, and version control data.

**Storage Details:**

**GitHub:** Stores the entire codebase, including source code, configuration files, and documentation. It also manages version history, allowing for easy rollback and collaboration.

**Vercel:** Stores deployment configurations, environment variables, and the static assets of the application. This ensures that the application can be quickly deployed and scaled as needed.

**Security Considerations:**

Both GitHub and Vercel offer robust security features, including access controls, encryption, and audit logs to monitor and protect the stored data.

Environment variables containing sensitive information (e.g., API keys) are securely managed by Vercel, ensuring they are not exposed in the codebase.

**3.3.4 User Session Management**

**Data Managed:** Session tokens and temporary session data.

**Storage Details:**

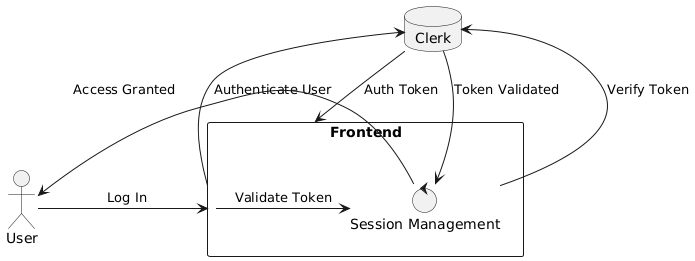
Session data is managed using Clerk’s secure token system, which eliminates the need for storing session data within the application itself. This temporary data is crucial for maintaining user sessions and ensuring a seamless user experience.

**Security Considerations:**

Session tokens are securely stored in the user's browser cookies or local storage, depending on the application's configuration, and are transmitted securely between the frontend and backend.

Clerk handles session expiration and renewal, ensuring that stale session data does not pose a security risk.

**Diagram:**

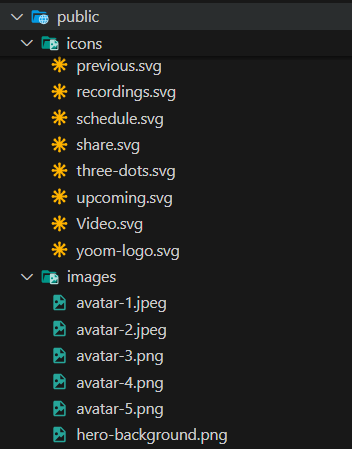


**3.3.5 Static Asset Storage**

**Data Stored:** Static assets such as images, CSS, JavaScript files, and other media.

**Storage Details:** Static assets are stored and served by Vercel’s CDN, ensuring fast load times and reliable delivery across various regions.

**Figure:**

****

**3.4 Communication**

Effective communication is at the core of this video conferencing application. The system is designed to facilitate seamless and real-time communication between users, ensuring that they can interact with each other through video, audio, and text. The communication architecture leverages various technologies and protocols to deliver a high-quality user experience while maintaining security and reliability.

**3.4.1 Real-Time Video and Audio Communication**

**Technology Used:** Stream API

**Communication Method:** The application uses WebRTC (Web Real-Time Communication) for peer-to-peer communication, facilitated by the Stream API. WebRTC is a widely adopted standard that enables browsers and mobile applications to communicate directly with each other without the need for an intermediary server.

**Key Features:**

**Low Latency:** WebRTC ensures low-latency communication, which is critical for maintaining natural conversation flow in video calls.

**High-Quality Streaming:** The Stream API optimizes video and audio quality based on the available network bandwidth, ensuring that users have a smooth experience even in fluctuating network conditions.

**Peer-to-Peer Connectivity:** WebRTC establishes direct connections between users, minimizing the need for centralized servers and reducing potential bottlenecks.

**Security Considerations:**

**Encryption:** All video and audio streams are encrypted using SRTP (Secure Real-Time Protocol), ensuring that communication is secure and protected from eavesdropping.

**ICE and STUN/TURN Servers:** WebRTC uses ICE (Interactive Connectivity Establishment) to find the best path for communication, even when users are behind NATs (Network Address Translators) or firewalls. STUN (Session Traversal Utilities for NAT) and TURN (Traversal Using Relays around NAT) servers are used to establish connections in more challenging network environments.

**3.4.2 Signalling and Session Management**

**Technology Used:** WebSockets

**Communication Method:** WebSockets provide a full-duplex communication channel over a single TCP connection, allowing the server to push real-time updates to clients without requiring them to constantly poll for changes.

**Key Features:**

**Session Initiation:** WebSockets are used to manage the signalling process for WebRTC, including the exchange of offer/answer messages, candidate information, and other metadata required to establish a connection.

**Real-Time Updates:** WebSockets enable real-time updates, such as user presence, session status, and chat messages, ensuring that all participants are kept in sync during a call.

**Security Considerations:**

**Secure WebSockets (wss):** WebSocket communication is secured using TLS (Transport Layer Security), which encrypts data and ensures the integrity and confidentiality of the communication channel.

**Authentication:** WebSocket connections are established only after successful user authentication, preventing unauthorized access to the communication channels.

**3.4.3 Chat and Messaging**

**Technology Used:** WebSockets, Stream API (for advanced messaging)

**Communication Method:** The chat feature is integrated into the video conferencing interface, allowing users to send text messages during a video call. This is especially useful for sharing links, documents, or making notes without interrupting the ongoing conversation.

**Key Features:**

**In-Call Chat:** Users can send and receive messages in real time while participating in a video call.

**Persistent Messaging:** If required, messages can be stored temporarily within the session, ensuring that late joiners can view the chat history.

**Security Considerations:**

Encryption: Chat messages are encrypted during transmission to prevent interception and unauthorized access.

**Moderation and Compliance:** Depending on the use case, messages can be moderated or logged to comply with organizational policies or legal requirements.

**3.4.4 Notifications and Alerts**

**Technology Used:** Email, Web Push Notifications

**Communication Method:** The application uses email and web push notifications to keep users informed about important events, such as upcoming meetings, session expirations, and new messages.

**Key Features:**

**Meeting Reminders:** Users receive notifications about scheduled meetings, ensuring they join on time.

**Session Expiry Alerts:** Users are notified if their session is about to expire or if they have been inactive for a prolonged period.

**New Message Notifications:** Users are alerted to new chat messages or other important updates, even when the application is minimized or running in the background.

**Security Considerations:**

Opt-In/Opt-Out: Users have control over the notifications they receive and can opt-in or out of specific types of alerts to manage their communication preferences.

Secure Channels: Notifications, especially those containing sensitive information, are sent through secure channels, with email notifications using TLS and web push notifications secured via HTTPS.

**3.4.5 Backend Communication**

**Technology Used:** REST APIs, WebSockets

**Communication Method:** The backend communicates with various third-party services and the frontend using REST APIs and WebSockets. REST APIs are used for standard operations like authentication, while WebSockets handle real-time communication.

**Key Features:**

Inter-Service Communication: The backend uses REST APIs to communicate with services like Clerk (for authentication) and Stream API (for video conferencing), ensuring a modular and decoupled architecture.

**Real-Time Data Sync:** WebSockets are employed to synchronize real-time data, such as user status and session details, between the backend and the frontend.

**Security Considerations:**

**API Security:** All API calls are authenticated and encrypted, ensuring that data is transmitted securely between services.

**Rate Limiting and Throttling:** The backend implements rate limiting and throttling mechanisms to prevent abuse of the communication channels and to protect against DDoS attacks.

**Chapter No. 4**