Project Title	Video Conferencing Web App
Technologies	MERN
Domain	Social Media
Project Level	Hard

Table

Table of Contents

1. Problem Statement:

Design a Video Conferencing Web based application.

1.1. Overview of Video Conferencing Web Application?

In the contemporary world, video conferencing is a crucial component. However, due to its intricacy, the majority of developers have trouble implementing it.

“The visual communication between two or more nodes linked to the internet is known as video conferencing. It enables the transmission of text, full-motion video, still photos, and high-definition audio between several nodes. “

People now have the option to work from home thanks to the pandemic. However, effective video conferencing and packet handling are needed for this.

1.1.1 Tools requirement for completing this project

a. React JS
b. Node JS
c. GraphQL
d. WebRTC

WebRTC: - An application can have real-time communication capabilities thanks to the open-source WebRTC technology. It allows for the transmission of data including audio, video, and other types between nodes.

It thus permits the incorporation of speech and video functions into applications by developers.

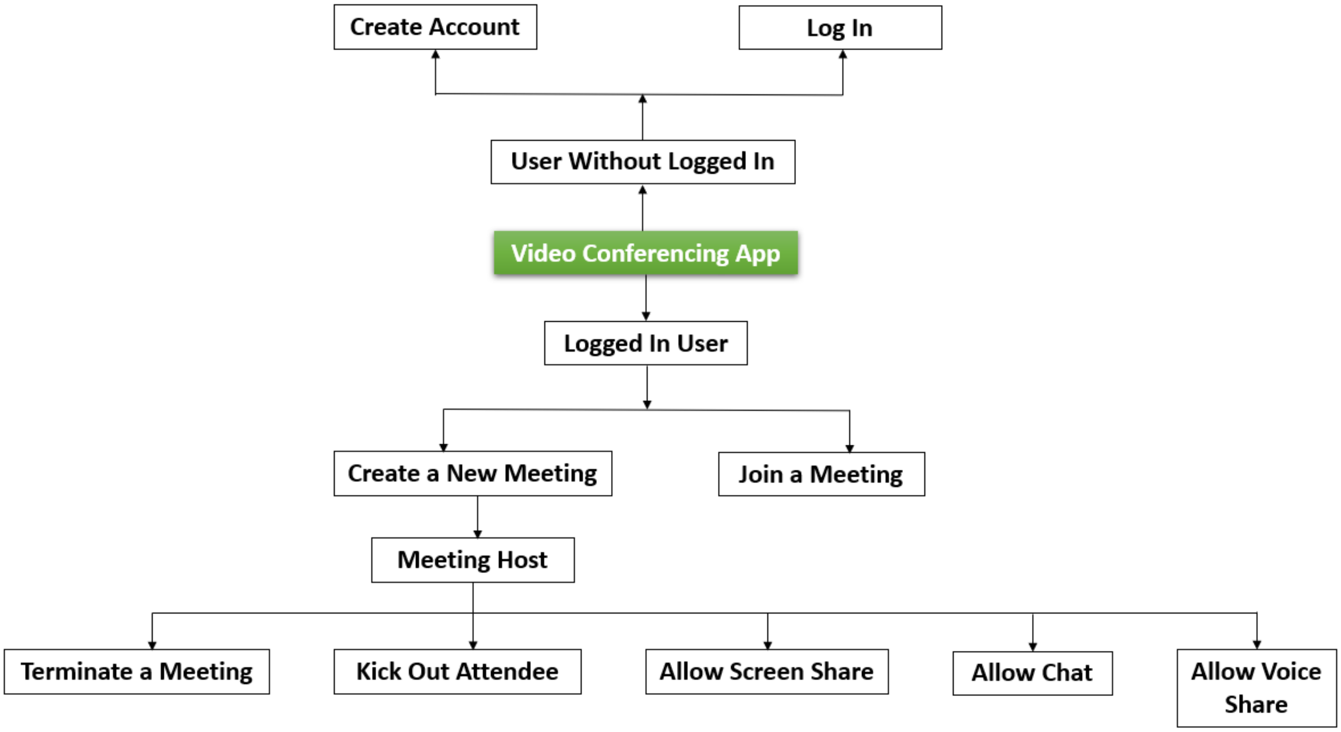
Note: **Google Meet** and **Zoom** is the best example to have a look.

1.1.2 Expected features of the Video Conferencing Application

Here is a detailed list of the expected features which should be available in your application-

a. User should have a **login** and **signup option** so that the new user can create an account and the existing user can login through that, this means you need to have **proper authentication-based login system** for the user.
b. No one can participate in the meeting without log in to the account.
c. After log in to the system a user can join a meeting by its **meeting link** or **meeting id**, which the meeting host have to generate randomly so that neither of any link is same as of other.
d. A user should be able to **schedule** a meeting or can also start it **instantly**.
e. A chat option should be available once a user joined in into the meeting to which the Host can disable at any time.
f. A user or host can share his screen, user can only share the screen once the host has allowed him.
g. A list of all the joined user should be displayed to host.
h. There should be an **active talk indicator**. Whenever a user speaks, he should be easily identified on the top of list to the host.
i. Host can **kick out** any participant from the meeting.

Low Level user flow



1.2. Project Objective

• **Video Conferencing Platform:** A simple and easy to use platform for everyone.

• **User-friendliness:** The project should be very easy to use enabling even a novice person to use it.

1.3. Scope of The Project

1. It is an in-demand technology used by millions.

2. Project should have a neat and clean user interface which is easy to use.

3. Users can join, share, or collab with each other here.

1.4. Functional and Non-Functional Requirements: -

1.4.1. Functional Requirements

1. **User Registration:** The program must allow users to sign up using their email, username, and password. Users must be able to self-register or, if they already have an account, log in immediately after opening the program.

2. **Creating New Meeting:** The user should be able to create a new meeting by self, and can schedule it accordingly.

3. **Voice Sharing:** All the users can share their voice with each other efficiently.

5. **Screen Sharing:** All the users should be able to share their screen only when the host will allow him to do so.

6. **Remove User:** The host can remove any attendee by self.

1.4.2. Non-Functional Requirements

1. **Privacy:** Only logged in user can join any meeting.

2. **Robustness:** In case the number of users increases, it should be able to handle all of them.

3. **Performance:** The application must be lightweight and must loads quickly.

1.5. Use Case Table

Authentication System	Register, Login, Logout	User and Host
Panel	Display all joined attendee	Host
User Dashboard	Share Screen and Voice Option	User and Host
Create Meeting	Creating a new meeting	Any Logged in User

Table 1. Use Case

2. Project Evaluation Metrics:

2.1. Code:

- You are supposed to write code in a modular fashion
- Safe: It can be used without causing harm.
- Testable: It can be tested at the code level.
- Maintainable: It can be maintained, even as your codebase grows.
- Portable: It works the same in every environment (operating system).
- You have to maintain your code on GitHub.
- You have to keep your GitHub repo public so that anyone can check your code.
- Proper readme file you have to maintain for any project development.
- You should include the basic workflow and execution of the entire project in the readme file on GitHub.
- Follow the coding standards.

2.2. Database:

MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas.

2.3. API Details or User Interface:

You have to expose your complete solution as an API or try to create a user interface for your model testing. Anything will be fine for us.

2.4. Deployment:

Implementation of reverse proxy, load balancing, and security group is mandatory for deployed applications.

2.5. Solutions Design:

You have to submit complete solution design strategies in High-level Document (HLD), Low-level Document (LLD), and Wireframe documents**.**

2.6. System Architecture:

You have to submit a system architecture design in your wireframe document and architecture document.

2.7. Optimization of solutions:

Try to optimize your solution on code level, architecture level, and mention all of these things in your final submission.

Mention your test cases for your project.

3. Submission requirements:

3.1. High-level Document:

You have to create a high-level document design for your project. You can reference the HLD form below the link.

Sample link: HLD Document Link

3.2. Low-level document:

You have to create a Low-level document design for your project; you can refer to the LLD from the link below.

Sample link: LLD Document Link

3.3. Architecture:

You have to create an Architecture document design for your project; you can refer to the Architecture from the link below.

Sample link: Architecture sample link

3.4. Wireframe:

You have to create a Wireframe document design for your project; refer to the Wireframe from the link below.

Demo link: Wireframe Document Link

3.5. Project code:

You have to submit your code to the GitHub repo in your dashboard when the final submission of your project.

Demo link: Project code sample link

3.6. Detail project report:

You have to create a detailed project report and submit that document as per the given sample.

Demo link: DPR sample link

3.7. Project demo video:

You have to record a project demo video for at least 5 Minutes and submit that link as per the given demo.

3.8. The project LinkedIn a post:

You have to post your project details on LinkedIn and submit that post link in your dashboard in your respective field.