MERNConnect: A Video Conferencing Application

A major project report submitted in partial fulfillment of the requirement for the award of degree of

Bachelor of Technology

in

Computer Science & Engineering / Information Technology

Submitted by

Saransh Sharma (201105)

Prakhar Jain (201407)

Under the guidance & supervision of

Dr. Vipul Kumar Sharma

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology,

Waknaghat, Solan - 173234 (India)

CERTIFICATE

I hereby declare that the work presented in this report entitled "MERN-Connect : A Video Conferencing Application" in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2023 to May 2024 under the supervision of Dr Vipul Kumar Sharma, Assistant Professor, Department of Computer Science and Engineering and Information Technology.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Saransh Sharma
201105
Prakhar Jain
201407

The above statement made is correct to the best of my knowledge.

Dr.Vipul Kumar Sharma
Assistant Professor
Department of Computer Science & Engineering and Information Technology
Dated: 13th May 2024

CANDIDATE'S DECLARATION

We hereby declare that the work presented in this report entitled **'MERN-Connect : A Video Conferencing Application'** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering / Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology, Waknaghat is an authentic record of our own work carried out over a period from August 2023 to May 2024 under the supervision of **Dr. Vipul Kumar Sharma** (Assistant Professor, Department of Computer Science & Engineering and Information Technology).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature with Date)	(Student Signature with Date)
Saransh Sharma	Prakhar Jain
201105	201407

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature with Date)
Dr. Vipul Kumar Sharma
Assistant Professor
Computer Science & Engineering and Information Technology
Dated: 13th May 2024

ACKNOWLEDGEMENT

Firstly, we express our heartiest thanks and gratefulness to almighty God for his divine blessing making it possible to complete the project work successfully.

We are really grateful and wish our profound indebtedness to Supervisor **Dr. Vipul Kumar Sharma, Assistant Professor**, Department of CSE Jaypee University of Information Technology,Wakhnaghat. Deep Knowledge & keen interest of my supervisor in the field of "**Information Security**" to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

We would like to express our heartiest gratitude to **Dr. Vipul Kumar Sharma, Assistant Professor** Department of CSE, for his kind help to finish our project.

We would also generously welcome each one of those individuals who have helped us straightforwardly or in a roundabout way in making this project a win. In this unique situation, we also want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated our undertaking.

Finally, we acknowledge with respect the constant support and patience of our parents.

Prakhar Jain (201407)
Saransh Sharma (201105)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

MERN …… MongoDB,Express,React,Node

OAuth ……..…Open Authorization

DB …….....Database

JS ……..….Javascript

UI ……..…User Interface

NoSQL …….……No Structured Query Language

WebRTC …….…….Web Real-Time Communication

API ……..…..Application Program Interface

CRUD. …….……Create, Read, Update,Delete

REST ……...….Representational State Transfer

BSON ……...……Binary Javascript Object Notation

JSON …… JavaScript Object Notation

UDP …….……..…...User Datagram Prototcol

GUI …….……..Graphical user interface

ODM …….……..…… Original Design Manufacturing

HTTP ……..……. Hypertext Transfer Protocol

CSS ……..... Cascading Style Sheets

TDD ……. Test-driven development

AI ……..Artificial Intelligence

ABSTRACT

In the contemporary digital landscape, communication and collaboration have become integral components of professional and personal interactions. With the paradigm shift towards remote work and virtual meetings, the demand for robust and feature-rich video conferencing solutions has surged. This major project introduces MERN-Connect, a cutting-edge Video-Conferencing Web Application developed using the MERN (MongoDB, Express.js, React, Node.js) stack. MERN-Connect offers a plethora of functionalities to enhance the virtual meeting experience, making it a versatile and user-friendly platform.

The core feature of MERN-Connect lies in its ability to facilitate seamless and secure virtual meetings with multiple participants. Users can effortlessly join meetings using unique room IDs, fostering a dynamic and collaborative environment. The application supports real-time audio and video communication, empowering users to engage in face-to-face discussions irrespective of their physical locations. The intuitive user interface ensures a smooth experience, with controls for toggling webcam and microphone functionalities, enabling users to customize their participation according to their preferences.

One of the standout features of MERN-Connect is the implementation of OAuth for Google login functionality. This enhances the user authentication process, offering a secure and streamlined onboarding experience. Users can log in seamlessly using their Google credentials, adding an extra layer of convenience and security to the application.

In addition to the standard video conferencing features, MERN-Connect introduces advanced capabilities such as screen sharing and a real-time collaborative whiteboard. These features empower users to share information, ideas, and presentations in a more interactive manner. The whiteboard

functionality allows multiple users to contribute simultaneously, fostering a creative and collaborative atmosphere.

MERN-Connect goes beyond the conventional by incorporating a meeting scheduling system integrated with Google Calendar. Users can schedule meetings directly from the application, and these events are automatically reflected in the Google Calendar of invited participants. This integration enhances productivity by seamlessly incorporating virtual meetings into users' existing workflow and scheduling tools.

Furthermore, MERN-Connect includes a real-time chat feature within the meeting room. This facilitates instant communication and collaboration parallel to video discussions, allowing users to exchange information and ideas in a written format. The chat functionality enhances the overall meeting experience by providing a versatile channel for communication.

In conclusion, MERN-Connect stands as a testament to the capabilities of the MERN stack in developing a comprehensive and feature-rich video conferencing solution. Its unique blend of functionalities, including OAuth-based Google login, screen sharing, collaborative whiteboard, and Google Calendar integration, positions it as a versatile tool for both professional and personal use. MERN-Connect is not just a video conferencing application; it is an integrated platform that transforms virtual meetings into dynamic and collaborative experiences. As the digital landscape continues to evolve, MERN-Connect sets a benchmark for innovative and user-centric video conferencing solutions.

Chapter 1: Introduction

1.1. Introduction

In the contemporary era, where the boundaries between physical and virtual realms continue to blur, communication technologies play a pivotal role in shaping how individuals connect and collaborate. The advent of remote work, accelerated by global events, has underscored the critical need for efficient and feature-rich video conferencing solutions. This major project embarks on a journey into the heart of modern communication technology, presenting the development and implementation of MERN-Connect, a comprehensive Video-Conferencing Web Application.

The landscape of communication has transformed dramatically over the last few years, with virtual meetings becoming an indispensable aspect of professional and personal interactions. Video conferencing has evolved from being a convenience to a necessity, connecting individuals across geographical distances and time zones. The demand for sophisticated, user-friendly, and secure video conferencing solutions has grown exponentially, prompting the development of MERN-Connect as a response to this emerging need.

The Genesis of MERN-Connect:

MERN-Connect is born out of the convergence of cutting-edge technologies encapsulated in the MERN stack – MongoDB, Express.js, React, and Node.js. The project represents a marriage of robust backend and frontend frameworks, leveraging the strengths of each component to create a seamless and dynamic video conferencing experience. The MERN stack provides a strong foundation, allowing for scalability, responsiveness, and a modular approach to development, which are imperative for a sophisticated application catering to diverse user needs.

The primary objective of MERN-Connect is to redefine the video conferencing paradigm by introducing a host of features that transcend the limitations of traditional platforms. The project aims to address not only the core functionality of facilitating virtual meetings but also to enrich the user experience through

innovative features, such as real-time collaborative whiteboarding, Google Calendar integration, and OAuth-based Google login.

Navigating the Evolving Landscape:

The global transition to remote work, coupled with an increased reliance on virtual communication, has posed new challenges and opportunities. MERN-Connect acknowledges this paradigm shift and aspires to be more than a mere video conferencing tool; it seeks to be a versatile platform that adapts to the evolving needs of users in various professional and personal spheres.

The decision to incorporate OAuth for Google login functionality was driven by the understanding that seamless authentication is fundamental to user experience. By enabling users to log in with their Google credentials, MERN-Connect not only enhances security but also streamlines the onboarding process. This feature aligns with the contemporary user's expectation of a frictionless, yet secure, login experience, positioning MERN-Connect as a modern and user-centric application.

Beyond the Basics:

MERN-Connect distinguishes itself by going beyond the basics of video conferencing. The inclusion of screen sharing adds a layer of versatility to the application, allowing users to share their screens seamlessly during meetings. This feature is particularly valuable for presentations, collaborative work sessions, and technical support scenarios, where visual communication is paramount.

The real-time collaborative whiteboard stands as a testament to the commitment to innovation within MERN-Connect. Traditional video conferencing often lacks the interactive element present in physical meetings. The whiteboard feature breaks down these barriers, allowing users to sketch, draw, and brainstorm together in real time. This not only enhances the collaborative aspect of virtual meetings but also brings a creative dimension to the interaction, making MERN-Connect a platform where ideas can be visualized and developed collectively.

Integration with Google Calendar:

Recognizing the importance of seamlessly integrating virtual meetings into users' existing workflows, MERN-Connect introduces a unique feature – the integration with Google Calendar. Users can now schedule meetings directly within the application, and these events are automatically reflected in the Google Calendar of invited participants. This integration not only streamlines the scheduling process but also ensures that users stay organized and up-to-date with their commitments. It marks a step towards a more holistic approach to virtual communication, acknowledging the interconnectedness of various tools and platforms.

A Glimpse into the User Experience:

The user experience is at the forefront of MERN-Connect's design philosophy. The intuitive interface ensures that users, regardless of their technical proficiency, can navigate the application effortlessly. Controls for toggling webcam and microphone functionalities are easily accessible, allowing users to personalize their participation in meetings according to their preferences. The real-time chat feature within the meeting room provides an additional avenue for communication, fostering an environment where information and ideas can be exchanged rapidly in a written format alongside the video discussions.

Conclusion of the Introduction:

In conclusion, MERN-Connect emerges as a response to the evolving needs of a digital society in constant flux. It is not merely a video conferencing tool; it is a testament to the capabilities of the MERN stack in creating a comprehensive and user-centric platform. As the narrative unfolds, the subsequent sections of this report will delve deeper into the technical aspects, design considerations, challenges faced, and the roadmap for future development. MERN-Connect encapsulates the spirit of innovation, adaptability, and connectivity, aiming to redefine the way we perceive and engage in virtual meetings.

1.2. Problem Statement

Unveiling the Challenges in Contemporary Video Conferencing and the Genesis of MERN-Connect

The advent of the digital age has brought about transformative changes in the way individuals communicate and collaborate. However, as the global shift toward remote work and virtual communication accelerates, traditional video conferencing solutions have revealed certain limitations, prompting the need for more sophisticated and feature-rich alternatives. This problem statement seeks to unravel the challenges inherent in contemporary video conferencing platforms and lay the groundwork for the development and implementation of MERN-Connect.

Limitations of Traditional Video Conferencing:

Traditional video conferencing tools, while instrumental in connecting individuals across vast distances, often fall short in providing a holistic and dynamic user experience. One of the primary challenges lies in the limited scope of features beyond basic audio and video communication. Many platforms lack the versatility required for collaborative work, presentations, and interactive discussions, leaving users seeking additional tools to supplement their virtual meetings.

Furthermore, the user interfaces of many existing solutions can be cumbersome, hindering the seamless navigation of participants through various functionalities. The learning curve associated with mastering these tools may deter users from fully engaging with the platform, especially those less familiar with technology.

Security Concerns:

The surge in remote work has amplified the importance of security in video conferencing. Incidents of unauthorized access, commonly referred to as "Zoom-bombing," have raised questions about the robustness of security measures implemented in some platforms. Privacy concerns, data breaches, and the potential for sensitive information to be compromised during virtual

meetings have become significant challenges that demand attention.

Additionally, the authentication and login procedures in some video conferencing tools might lack the sophistication needed to thwart malicious activities effectively. Striking a balance between user-friendly access and stringent security measures has become a delicate yet critical challenge.

Integration with Existing Workflows:

As remote work becomes a long-term or permanent arrangement for many, the need to seamlessly integrate virtual communication tools with existing workflows has become paramount. The lack of integration with popular productivity and scheduling tools often results in disjointed user experiences. Users are forced to navigate between different applications to schedule, join, and manage virtual meetings, leading to inefficiencies and potential oversights in their professional commitments.

This integration challenge is further exacerbated when considering the need for real-time synchronization of meeting schedules with widely used calendaring tools. A solution that effortlessly incorporates virtual meetings into the broader workflow of users is imperative for enhancing productivity and streamlining daily operations.

Authentication and User Onboarding:

User authentication poses a multifaceted challenge in video conferencing platforms. Ensuring the security of user accounts while providing a user-friendly onboarding experience is a delicate balance to strike. Password-related vulnerabilities, inadequate two-factor authentication, and the risk of unauthorized access are issues that persist in certain platforms.

Moreover, the onboarding process itself can be a potential bottleneck. Cumbersome registration procedures and a lack of seamless integration with widely used authentication systems may discourage potential users from adopting the platform, limiting its reach and impact.

Lack of Collaborative Features:

While video conferencing tools offer a means of communication, they often lack features that foster collaboration and interaction during virtual meetings. Basic

tools may lack advanced capabilities such as screen sharing, collaborative document editing, and real-time whiteboarding, hindering the ability of participants to work together creatively and effectively.

Addressing these challenges requires a comprehensive approach that goes beyond the conventional boundaries of video conferencing, incorporating features that mimic the richness and spontaneity of face-to-face interactions.

The Genesis of MERN-Connect:

In response to the outlined challenges, MERN-Connect emerges as a novel solution rooted in the MERN stack. By leveraging MongoDB, Express.js, React, and Node.js, this project aims to redefine the video conferencing landscape. MERN-Connect goes beyond the limitations of traditional platforms by introducing advanced features such as real-time collaborative whiteboarding, Google Calendar integration, OAuth-based Google login, and seamless integration with existing workflows. The subsequent sections of this report will delve into the technical intricacies, design considerations, and the roadmap for future development, shedding light on how MERN-Connect endeavors to address the challenges ingrained in contemporary video conferencing.

1.3. Objectives

The objective of this project was to develop a fully-featured video conferencing web application using the MERN stack that enables seamless communication and collaboration. Named MERN-Connect, this application allows multiple participants to join meetings, see and hear each other in real-time, and work together using interactive tools for screen sharing, chat, whiteboarding, and more.

The goal was to leverage the benefits of the MongoDB, ExpressJS, ReactJS, and NodeJS technologies to build an easy-to-use and flexible platform. MongoDB offers high performance and scalability as a NoSQL database, ExpressJS provides a minimal framework for building the REST API routes and middleware, ReactJS offers an interactive UI through its component model, and NodeJS enables real-time communication by executing JavaScript on the server-side.

Key features implemented include:

- Video calling with support for muting audio and turning video on/off
- Screen sharing so participants can broadcast their desktops to the meeting
- Chat messaging for quick conversations during meetings
- Whiteboard with drawing tools for collaborating in real-time
- Scheduling meetings and integrating with Google Calendar for automatically notifying attendees
- Unique meeting room IDs and invite links for easily joining sessions
- User management and authentication using OAuth with Google Sign-In

The focus was on modular front-end development with React components, maintaining a well-organized back-end API in ExpressJS, and ensuring low-latency peer-to-peer communication by implementing the Socket.IO library in NodeJS.

The final product delivers a complete alternative to proprietary tools for teams that want greater control, customizations, and privacy over their online interactions. It enables seamless video meetings along with real-time content creation and task tracking.

Going forward, additional features like breakout rooms, polling, language translations, and reliability improvements through stress testing will help enhance the experience even more.

Overall, this project has resulted in an adaptable video conferencing application using JavaScript-based technologies that can continue to incorporate better collaboration and productivity features for modern web applications. The learnings will enable leveraging the MERN stack for rapidly building more secure and flexible software platforms.

1.4. Significance and Motivation of the project work

The COVID-19 pandemic made remote work and virtual meetings ubiquitous over the last few years. With distributed teams and hybrid work environments becoming the norm, there is a growing need for reliable and easy-to-use video conferencing solutions to connect people across locations. However, popular proprietary tools often come with limitations around customizations, security, cross-platform support, and pricing models. This project was motivated by the goal of developing a customizable open-source alternative that puts users in control.

MERN-Connect demonstrates the potential of JavaScript-based tech for building fully-featured video conferencing into web apps. The motivation was to facilitate seamless communication for teams without restrictions imposed by third-party platforms. MongoDB offers the flexibility of a document data model for stored data like meetings, users, messages etc. ReactJS allows creating modular and interactive real-time user interfaces that ease the learning curve. ExpressJS enables building reusable logic with its extensive libraries and middleware. Finally, NodeJS is asynchronous, fast, and great for networking apps with its event-driven model. Together they deliver the building blocks for high-performance browser-based video collaboration.

Now any developer with JS skills can extend this platform to suit their specific organizational needs. The open-source nature fosters innovation - developers can tweak flows, add integrations, customize UI, and plug in video calling libraries like WebRTC and Socket.io. Teams get transparency and control over meeting data, tools, and access privileges rather than being bounded by vendor limitations. Over time this reduces costs and technical debt substantially while still retaining enterprise-grade security, scalability and support.

The shared whiteboard for example has huge potential for design teams to collaborate visually on short-lived diagrams and brainstorming sessions. The simple Google OAuth login helps users safely access all personal/team meetings using their enterprise identity without managing new accounts. The chat

preserves conversations that get lost in other apps. Automated recording/transcription features can be plugged in later for easier reference.

As remote/hybrid teams focus on productivity and engagement during meetings, platforms like MERN Connect that provide customizable groupware can create that experience. Features tailored to specific use cases with the freedoms of open-source backing empowers teams over vendors. Continued development of this application can enhance visibility, transparency, analytics, and integrations for enabling the next generation of communication inside organizations.

Overall this project highlights the strengths of JavaScript-based Full-Stack frameworks for delivering flexible and user-centric video collaboration platforms rapidly. The motivation is to provide customizable and secure alternatives that serve evolving needs of modern web/mobile applications.

1.5. Organization of Project Report

Chapter 2: Literature Survey

2.1 Overview of Relevant Literature

- Introduction to existing literature on MERNConnect:A Video Conferencing Application, exploring various methodologies and technologies in the context of Information Security.

2.2 Key Gaps in the Literature

- Identification and discussion of gaps or limitations in existing literature concerning about a video conferencing application, paving the way for the unique contribution of our project.

Chapter 3: System Development

3.1 Requirements and Analysis

- In-depth analysis of the specific requirements for developing a video calling application with different new features.

3.2 Project Design and Architecture

- Presentation of the overall design and architecture of our application.

3.3 Implementation

- Inclusion of relevant code snippets, algorithms, tools, and techniques used in implementing the video calling application using MERN Stack.

3.4 Key Challenges

- Discussion of challenges faced during the development process.

Chapter 4: Testing

4.1 Testing Strategy

- Overview and discussion of the testing strategy applied to evaluate the accuracy and reliability of our video calling application.

Chapter 5: Results and Evaluation

5.1 Results

- Presentation and interpretation of our video calling application.

Chapter 6: Conclusions and Future Scope

6.1 Conclusion

- Summarization of key findings, limitations, and contributions made by the project in the realm of video calling applications.

6.2 Future Scope

- Exploration of potential avenues for future research and development, considering enhancements, broader applications, and further advancements in video calling applications.

Chapter 2 : Literature Survey

2.1. Overview of Relevant Literature

M. P. Rathee et al. [1] The paper "VIDEO/AUDIO CONFERENCING USING WEBRTC," written by M. P. Rathee, D. Bhatla, S. Khan, S. Chowdhary, and V. Diwan, published in August 2022 within the International Journal of Engineering Applied Sciences and Technology, Volume 7, Number four, pages 276-280. It explores using WebRTC for video and audio conferencing.

It is viable that the look at investigates how WebRTC technology may be used to build video and audio conferencing structures. The real-time verbal exchange function of net browsers, or WebRTC, is a key detail in the creation of the suggested conferencing device.

The usage of the platform's video and audio conferencing capabilities, the consumer experience it presents, and the technical issues related to using WebRTC for those targets could all be critical topics of debate inside the examine. The blessings and drawbacks of utilising WebRTC inside the framework in their conferencing application is probably blanketed through the writers.

The research may additionally observe engineering and implemented sciences standards because it changed into posted in a certain journal. This ought to consist of discussing how the suggested answer complies with time-honored engineering practices and advances in era.

In conclusion, this take a look at sincerely sheds light on the planning and execution of a WebRTC-based totally video/audio conferencing gadget and gives insightful knowledge on the actual-world uses and issues surrounding actual-time verbal exchange technology.

K. MacMillan et al. [2] proposed to assess the effectiveness and community usage of well-known video conferencing packages. To measure vital variables related to those programs' effectiveness and community impact, the look at makes use of a methodical method.

To examine special facets of video conferencing apps, the researchers probable

used a mixture of quantitative strategies, community analysis, and performance measuring instruments. A few feasible metrics to study are latency, bandwidth usage, quality of audio and video, and average user enjoy.

Through overall performance evaluation of numerous video conferencing structures, the thing may provide beneficial insights into the advantages and downsides of each software. This expertise is important, especially given the increasing reliance on online collaboration, remote work, and virtual communication tools.

The studies's conclusions would possibly affect both end users and people who create video conferencing software. Users can learn about the alternate-offs between overall performance on various systems, and builders can utilize the statistics to enhance their packages through optimizing for better user enjoy and network use.

Ushakov et al. [3] proposed the use of a WebRTC-based video conferencing platform in a school room putting. The venture is to address the precise requirements and problems of remote verbal exchange in instructional settings. It is associated with the Department of Mathematics and Information Technology at Orenburg State University inside the Russian Federation.

It is probable that the investigators will inspect the ability of WebRTC (Web Real-Time Communication) generation to expand a efficient and without problems navigable video conferencing platform that meets the wishes of educational institutions. This should include things like simplicity of use, integration with academic materials, and actual-time engagement.

The effect of the WebRTC-primarily based platform on enhancing scholar-instructor conversation and collaboration—as well as perhaps other stakeholders within the educational manner—may be the difficulty of this examine. The platform might be made to support remote lectures, cooperative projects, virtual school rooms, and other educational endeavors.

The platform's consumer interface, scalability, protection capabilities, and flexibility to various instructional contexts are feasible topics for the studies. The consequences may want to offer mild on how properly WebRTC fosters an environment that is favorable for online learning.

In end, this examine will probably upload a tremendous deal to our knowledge

of the advent and application of WebRTC-primarily based video conferencing solutions in academic settings, illuminating each the benefits and disadvantages of those structures.

Vasan Subramanian et al. [4] releases e book "M. E. R. N. Stack: Full Stack Web App Development with Mongo" probable gives an intensive how-to for growing complete-stack web applications with the MERN stack. The MERN stack additives MongoDB, Express.Js, React, and Node.Js as well as their functions in web development, are added at the outset.

Making positive they have the specified dependencies and gear, the writer walks readers via putting in a development surroundings. The e book then goes into backend programming and indicates you how to use Node.Js and Express.Js to build a dependable server. Routing, middleware, and interactions with the MongoDB database are in all likelihood protected on this section. It explains thoughts like component-based architecture, state control, and user interface interactivity at the same time as additionally introducing frontend development with React.

One of the primary features is the frontend and backend integration, which shows how RESTful APIs are used to build conversation between them. The e book, which covers topics like schema layout and CRUD operations, maximum likely gives insights into green database administration the usage of MongoDB. In order to ensure readers recognize how to pass their MERN stack apps from improvement to a live surroundings, it probably concludes with the aid of analyzing deployment strategies.

Karthick et al. [5] proposes the way to create a chat utility that emphasizes professionalism while using Natural Language Processing (NLP) techniques.

The take a look at likely starts out by means of discussing the need of a formal chat program and outlining how NLP would be included as a essential detail. In order to growth communique inside the software, NLP proposes setting a sturdy emphasis on comprehending and processing human language.

The expert chat software's structure, the combination of NLP algorithms for language interpretation, and elements to provide a proper and effective communication surroundings are a number of the important thing additives that

can be covered inside the research. The writers should pass into the unique NLP techniques used and the way they advanced the capability of the chat software.
The observe, which highlights the use of natural language processing (NLP) within the creation of creative communication systems, maximum definitely places itself beneath the superior computing developments category given the backdrop of the IEEE International Conference on Current Trends in Advanced Computing (ICCTAC).
In precis, this look at gives insights into the development and use of a enterprise-grade chat software, showcasing the incorporation of natural language processing techniques to improve communication.

Dayley et al. [6] release the book "Node.Js, MongoDB, and Angular Web Development: The Definitive Guide to Using the MEAN Stack to Build Web Applications" by Brad Dayley, Brendan Dayley, and Caleb Dayley is an intensive guide for builders wishing to create net applications utilising the MEAN stack (MongoDB, Express.Js, Angular, and Node.Js).
The MEAN stack—MongoDB for the database, Express.Js for the internet application framework, Angular for the front-stop framework, and Node.Js for the server-side runtime—will likely be blanketed within the first section of the book. It is probably that the writers will spotlight the blessings of utilizing this stack for JavaScript improvement in its entirety.
The writers maximum possibly use a arms-on method at some point of the manual, on foot readers step-by way of-step through the procedure of creating a web application. This can entail configuring the development environment, designing and enforcing the database with MongoDB, utilising Node.Js and Express.Js to create a server, and the use of Angular to generate the front cease. Real-time verbal exchange, consumer authentication and permission, RESTful API design, and satisfactory practices for designing and handling MEAN stack applications are the various important topics that might be blanketed. The difficulties in creating maintainable and scalable packages the usage of those technology will likely be protected in the e book.
Given that it was posted in 2017, readers can be capable of recognize the technical panorama of that age by studying approximately industry high-quality practices and pertinent concerns from that time.

All matters considered, "Node.Js, MongoDB, and Angular Web Development" is probably going to be awesome resource for developers looking for an intensive and useful guide on growing online apps the use of MEAN stack.

Jang-Jaccard et al.[7] proposed how WebRTC (Web Real-Time Communication) is probably used inside the healthcare enterprise for tele-home monitoring. This study explores the viability and performance of the use of WebRTC generation to offer video conferences in a remote tracking surroundings for healthcare.

Healthcare specialists need to remotely reveal and examine patients' health country at the same time as they're at domestic with tele-domestic tracking. The use of WebRTC implies an emphasis on actual-time communique features that can be accessed via internet browsers with out the installation of more plugins or software program.

It's probable that the researchers compare a variety of of factors, which include the effectiveness and dependability of WebRTC video conferencing, its possible impact on healthcare consequences, and the system's fashionable viability. In addition, privateness, safety, and usability issues will be addressed to guarantee the generation's applicability and adoption within the healthcare placing.

The have a look at's conclusions could offer critical new light on how WebRTC can improve tele-home monitoring, likely decorate affected person-doctor verbal exchange, reduce the need for in-man or woman visits, and in the end affect the efficacy and efficiency of healthcare services.

Henriyan et al.[8] examines the creation of an internet-primarily based real-time chat server.

The necessity for powerful actual-time communication in web-primarily based applications might be addressed in the have a look at's opening paragraph earlier than describing the design tenets and specifics of the chat interface server's implementation. Based on their design selections, the writers can also speak approximately the problems in developing a scalable and responsive chat system

and offer answers.

The design of the chat server, the era stack applied for implementation, and elements to make certain actual-time verbal exchange competencies are a number of the vital subjects mentioned in the research. The writers may draw interest to certain elements of the person interface, focusing on the convenience of use and person revel in in the on-line chat surroundings.

The observe in all likelihood situates itself in the larger area of device engineering, demonstrating how the suggested chat device complies with engineering ideas and technical improvements, given the context of the 6th International Conference on System Engineering and Technology (ICSET).

To sum up, this examine sheds light on the complexities involved in creating and deploying a web-primarily based actual-time chat interface server. It makes a big contribution to the sphere of generation and system engineering by emphasizing the beneficial capabilities of making effective and responsive communication systems.

Plugge et al. [9] gives a top level view of the famous NoSQL database MongoDB. The ebook's maximum probable motive is to present database administrators, developers, and statistics experts a comprehensive draw close of MongoDB's capabilities, abilties, and best practices for dealing with huge quantities of information.

It is likely that the ebook begins out with outlining the primary ideas in the back of MongoDB, highlighting its disbursed architecture, BSON format, and file-oriented facts version. The writers may go into detail about the way to installation and configure MongoDB, strolling users thru the first few steps of the usage of this NoSQL database.

Data modeling with MongoDB, indexing strategies for speed optimization, and retrieving records using the MongoDB question language are possibly to be some of the book's foremost subjects. The writers may additionally delve into greater complex factors, such textual content search, geospatial indexing, and aggregation pipelines, demonstrating how these tools can be used to handle large statistics eventualities in an efficient manner.

High availability and scalability are possibly going to be important troubles, and the e-book may move on MongoDB's replication and sharding strategies. To

demonstrate how MongoDB may be used in real-international programs to deal with huge volumes of records, case studies and sensible examples might be offered.

The e-book's 2015 ebook date suggests that it covers first-rate practices and MongoDB's modern-day circumstance in the context of NoSQL databases. Even if new abilties have been added to MongoDB in later editions, the fundamental thoughts mentioned inside the e-book are nevertheless critical.

In conclusion, "The Definitive Guide to MongoDB" is probably going to be a helpful resource for anyone trying to learn MongoDB thoroughly, particularly when working with large data.

Loreto et al. [10] The 2014 O'Reilly Media e-book "Real-Time Communication with WebRTC: Peer-to-Peer inside the Browser" by way of Salvatore Loreto and Simon Pietro Romano is a radical guide that most possibly specializes in WebRTC (Web Real-Time Communication) technology and offers insights into growing peer-to-peer conversation capabilities without delay in internet browsers.

Probably the first segment of the e book will cover the fundamentals of WebRTC, highlighting the way it permits actual-time communication between browsers to manifest, consisting of information sharing, audio, and video sharing, with out the want for plugins or different software. Readers may additionally find Loreto and Romano beneficial in explaining the basics of WebRTC structure, protocols, and APIs.

Establishing a WebRTC environment, putting audio and video verbal exchange competencies in region, and handling statistics channels for peer-to-peer information sharing are perhaps the various vital subjects addressed in the ebook. The fundamental safety worries and protocols which might be essential for developing reliable and safe real-time conversation packages may also be included via the writers.

The ebook may pass into detail on real-world packages and use cases for WebRTC, like report sharing, video conferencing, and teamwork. Readers can be given sensible examples and code snippets to assist them incorporate WebRTC competencies into their tasks.

The book is probably going to seize the early WebRTC improvement scene,

together with the fine practices and guiding concepts that were available on the time, considering its 2014 publishing date. The ebook is a outstanding aid for learning the fundamentals of WebRTC, despite the fact that there may had been similarly advances.

To sum up, "Real-Time Communication with WebRTC" is probably going to be a helpful manual for programmers who want to use WebRTC technology to incorporate peer-to-peer communication features into web browsers.

Chodorow et al.[11] releases ebook "MongoDB: The Definitive Guide" is an intensive aid for gaining knowledge of approximately and becoming talented with the famous NoSQL database MongoDB. The 2013 e-book, written by way of O'Reilly Media, pursuits to offer readers a radical knowledge of MongoDB's functionality, structure, and quality practices to be used.

The primary ideas of MongoDB, inclusive of file-oriented data garage, the BSON (Binary JSON) format, and the MongoDB question language, are protected within the first phase of the e book. It walks readers thru the set up technique and essential capabilities even as imparting actual-international examples and beneficial steering for laying a strong foundation. In order to manipulate huge datasets and assure excessive availability, the writers talk sharding and replication as well as MongoDB's scalability.

Notably, the book indicates the way to integrate the database with exclusive programming languages and frameworks, emphasizing the realistic elements of growing programs with MongoDB. It covers pleasant practices and real-world use cases for developing scalable and powerful MongoDB-based totally packages.

Chodorow and Dirolf highlight MongoDB's scalability, flexibility, and overall performance benefits over traditional relational databases at some stage in the tutorial. The path is designed with each beginner and seasoned developers in mind, offering a thorough hold close of MongoDB's capabilities and functionalities.

Wilson et al. [12] proposed Building Node Applications with MongoDB and Backbone: Rapid Prototyping and Scalable Deployment changed into posted in

2012 via O'Reilly Media, Inc., by using Mike Wilson is a manual that maximum likely focuses on integrating Node.Js, MongoDB, and Backbone.Js for online application improvement. The book's most in all likelihood reason is to present builders information on a way to use those technology to speedy prototype and scale up the deployment of web programs.

Most likely, the e book starts out with introducing the principle technologies, which include Backbone.Js for patron-aspect structure, MongoDB as a NoSQL database, and Node.Js for server-side JavaScript. Wilson might draw interest to the approaches wherein those technologies paintings collectively, highlighting how they assist to permit scale deployment and short development.

The method of creating and organizing web apps with Backbone.Js on the client facet and Node.Js on the server aspect can be many of the e-book's foremost subjects. The author may walk readers via the manner of making RESTful APIs, storing statistics in MongoDB, and putting scalable deployment approaches into exercise.

To assist students study through doing, actual-international case research, code snippets, and practical examples are normally covered to help explain the thoughts. Best practices for creating maintainable and scalable packages with this generation stack may also be blanketed within the ebook.

Given that it changed into posted in 2012, the e-book would possibly provide readers interested in the development of these technology with ancient heritage and insights into the situation of Node.Js, MongoDB, and Backbone.Js at that time.

To sum up, builders looking for advice on creating internet apps with an emphasis on scalable deployment and short prototyping will probably locate "Building Node Applications with MongoDB andBackbone" to be a useful resource.

Malhotra et al. [13] proposes the layout and implementation of a chat application that uses the User Datagram Protocol (UDP) are tested in the studies paper "UDP Based Chat Application," and presented on the 2010 2nd International Conference on Computer Engineering and Technology.

The motivation for the use of UDP as the chat software's underlying communication protocol is probably covered within the look at's commencing

section. Because UDP is a lightweight, connectionless protocol, it proposes targeting actual-time, low-latency verbal exchange.

The studies may also embody numerous important factors, along with the UDP-based chat utility's architecture, the layout selections used to capitalize on UDP's attributes, and measures taken to guarantee dependable and effective conversation. The writers could speak at the benefits and downsides of making use of UDP in opposition to alternative verbal exchange protocols.

The article possibly locations itself inside the subject of computer engineering, demonstrating the useful software of community protocols within the advent of conversation structures, given the context of the International Conference on Computer Engineering and Technology.

In precis, this study sheds light at the making plans and execution of an UDP-primarily based chat software and presents an analysis of the exchange-offs and elements to be taken into consideration while choosing UDP for real-time verbal exchange.

Kydd, Christine T et al. [14] examines how video conferencing technology is used and the way it affects managerial techniques. It was published in Information and Management. The take a look at explores the causes in the back of video conferencing's incorporation into organizational communique systems and how managers used it within the early Nineties. Probably because they paintings in geographically scattered settings, the authors focus on unique managerial duties which might be affected by video conferencing, like teamwork and selection-making processes.

The problems managers have implementing video conferencing technology and the blessings they see, such as possible gains inside the efficacy and efficiency of communique, will likely be blanketed in the article.

P. Rangan et al. [15] of the Multimedia Laboratory within the Computer Science and Engineering (CSE) Department at the University of California, La Jolla, are the author of a paper that examines the combination of document garage, control, and video conferencing in multimedia computer structures.

It is probably that the have a look at explores the possibilities and problems of adding video conferencing functionality to multimedia computer structures.

This may want to entail taking bandwidth constraints, actual-time verbal exchange, and the clean integration of multimedia content in the convention environment into account.

The article can also talk document management and storage issues in the large framework of multimedia systems. This ought to contain speakme approximately features that make multimedia content material on hand to customers, metadata control, and powerful methods to store multimedia files.

A awareness on developing technology and procedures connected to multimedia processing, storage, and communication is usually recommended by using the association with the Multimedia Laboratory. The article may provide new views on how document garage, multimedia control, and video conferencing have interaction, in addition to creative fixes or strategies to enhance multimedia pc structures' universal person enjoy.

2.2. Key Gaps in the Literature

In **M. P. Rathee et al. [1],** the main three gaps are

- Absence of specific Technical Details: If the paper is devoid of unique technical records concerning the WebRTC-based video/audio conferencing system's implementation, it could have an opening. The paper's cost could be extended by means of which includes specifics at the structure, protocols used, and the way possible technical issues had been resolved.
- Lack of User Experience Evaluation: If the look at does no longer comprise a complete evaluation of the person enjoy inside the framework of the video/audio conferencing machine, there may be an opening inside the facts. Determining the system's realistic value calls for an understanding of person pleasure, simplicity of use, and potential issues from the person's point of view.
- Limited Comparison with Existing Solutions: If the thing does no longer include a assessment evaluation with other cutting-edge video/audio conferencing options, it may incorporate a gap. A comparative evaluation could shed mild on the special functions, benefits, or drawbacks of the WebRTC-based totally system whilst compared to nicely-installed options.

In **K. MacMillan et al. [2],** the main four gaps are

- Limited Scope of Video Conferencing Applications: Check if the paper explores a diverse set of video conferencing applications. If the look at makes a speciality of only a few famous programs, it may lack generalizability, and there can be an opening in representing the broader panorama of video conferencing gear.

- Inadequate Consideration of Network Conditions: Assess whether or not the paper effectively considers diverse community situations. Gaps may additionally exist if the observe mainly specializes in

specific community scenarios, along with excessive-velocity net, without addressing the demanding situations posed with the aid of varying bandwidths or network instabilities.

- Static Analysis Without Real-time Scenarios: Check if the studies evaluates video conferencing applications in real-time scenarios. A hole may be present if the have a look at is based completely on static analysis and does no longer simulate dynamic conditions, inclusive of concurrent customers or fluctuating network conditions that users normally stumble upon.

- Limited Exploration of Security and Privacy Aspects: Assess whether or not the paper delves into the security and privacy aspects of video conferencing packages. Gaps might also exist if the observe overlooks ability vulnerabilities, encryption practices, or privateness concerns related to using these structure.

In **Ushakov et al. [3]**, the main three gaps are

- Lack of Pedagogical Impact Analysis: If the WebRTC-primarily based platform's pedagogical impact isn't very well explored, the paper may incorporate an opening. For a thorough expertise of era's usefulness inside the study room, it is imperative to comprehend the way it affects teaching techniques, scholar engagement, and learning effects.
- Limited Analysis of Technical Challenges: There can be an opening in addressing actual-international barriers to successful deployment and extensive adoption if the look at does no longer absolutely analyze technical problems like community reliability, tool compatibility, and scalability which might be associated with imposing WebRTC in instructional settings.
- Possible Absence of Comparative Evaluation: If the WebRTC-based platform isn't evaluated in evaluation to other video conferencing options now to be had for use in instructional settings, the record may consist of a gap.

In **Vasan Subramanian et al. [4]]**, the main three gaps are

- Absence of Updated Information: Since the e-book's May 2019 ebook, there can be an opening if it does now not encompass the most current modifications and innovations within the MERN stack components (MongoDB, Express.Js, React, and Node.Js). Because web improvement technology are evolving so quickly, readers risk lacking vital details about new capabilities, fine practices, and framework changes.
- Limited Coverage of Advanced Topics: A comprehensive know-how of growing dependable and stable complete-stack packages can be missing if the e-book does not move deeply into advanced standards and practices related to MERN stack development, along with scalability, security considerations, or superior deployment techniques.
- Exclusion of Alternative techniques: If the e-book simplest covers the MERN stack with out going over opportunity technologies or strategies, there may be a gap. Understanding when and why they might choose sure stacks or technology relying on particular undertaking goals and constraints may be beneficial to readers.

In **Karthick et al. [5]]**, the main three gaps are

- Insufficient Assessment of NLP Effectiveness: If the paper does now not consist of an intensive assessment of the NLP tactics used inside the expert chat software, it may have an opening. The paper's depth might be increased by means of an in depth assessment of the NLP algorithms' overall performance in phrases of human language interpretation and processing, as well as any shortcomings or problems.
- Limited User Adaptability Discussion: If the examine does not pass into tremendous detail approximately how users can adjust to the professional chat software, there can be a gap in knowledge. For a comprehensive hold close of the utility's viability, insights

concerning consumer reports, prospective mastering curves, and techniques used to guarantee user-friendly interactions could be beneficial.

- Lack of Comparative Analysis: If the paper does not examine itself to different professional chat applications presently in use, it is able to have a weak spot. A comparative analysis might offer treasured perspectives on the distinct blessings or constraints of the NLP-based technique vis-à-vis properly-mounted alternatives, so fostering a extra sophisticated comprehension of the era's efficacy.

In **Dayley et al. [6]]**, the main three gaps are

- Temporal Relevance: In 2017, there was a giant evolution in the technology scene, specially with regard to net improvement. If the ebook does no longer cover greater latest variations, upgrades, or trends relating the MEAN stack components—MongoDB, Express.Js, Angular, and Node.Js—there can be a ability hole. This might have an effect on how properly the given recommendation applies to the manner the enterprise operates now.
- Absence of Coverage on Advanced subjects: It is crucial that the e book covers all the MEAN stack's advanced subjects, along with serverless structure, microservices, and containerization. If it doesn't cross into wonderful detail about these state-of-the-art thoughts, there might be an opening that stops readers from studying approximately present day net improvement techniques.
- Limited Focus on Best Practices: Although the book can also deal with great practices, there may be a void if it would not area sufficient interest on overall performance optimization strategies, protection strategies, or enterprise requirements that have developed considering that 2017. Readers might not benefit insight into the most recent traits in building high-acting and solid MEAN stack applications as a result.

In **Jang-Jaccard et al.[7]]**, the main three gaps are

- Limited Examination of Security Measures: If the paper does not cross into fantastic detail about the safety measures related to the use of WebRTC in healthcare, it could include a gap. Given the delicate nature of fitness data, a radical examination of encryption techniques and privacy protections is crucial.

- Lack of Comparative Analysis: If WebRTC is not in comparison to other tele-domestic tracking technology presently in use, there may be a gap within the look at. This assessment is important to comprehending the unique advantages or drawbacks of WebRTC in the healthcare industry.

- Lack of User Experience Assessment: There can be an absence in understanding on the usability and acceptability of WebRTC in a realistic healthcare setting if the paper does now not offer a thorough assessment of consumer studies, related to each patients and healthcare professionals.

In **Henriyan et al.[8]]**, the main three gaps are

- Inadequate User Experience Assessment: If the paper does now not encompass a complete evaluation of the consumer experience inside the framework of the internet-primarily based real-time chat interface server, there can be a gap in it. Determining the success of the device requires an information of user happiness, the intuitiveness of the interface, and ability limitations.
- Limited Discussion of Scalability Issues: If the study does now not move into incredible detail approximately the scalability problems that arose during the chat server's deployment, there may be an opening in knowledge. The take a look at would be more complete if it protected information on how the cautioned system handles or minimizes scalability worries, which are crucial to actual-time systems.
- Lack of Comparative Analysis: If the paper does no longer provide

a comparison with comparable net-based real-time chat structures that are currently in use, it may have a gap in it. A comparative analysis could throw light on the unique characteristics, advantages, or drawbacks of the cautioned chat server when compared to famous alternatives.

In **Plugge et al. [9]]**, the main three gaps are

- Recent Updates and Features: A ability gap ought to exist if the ebook does no longer cover MongoDB's advancements, features, and high-quality practices introduced after 2015. Readers might omit insights into more moderen skills, safety enhancements, and changes in advocated tactics, impacting their capacity to leverage the trendy MongoDB functionalities.
- Advanced Use Cases:If the e-book would not significantly cowl superior use cases, which includes real-international programs managing complex scenarios or the mixing of MongoDB with different technology, readers searching for in-intensity realistic steering for managing elaborate conditions may additionally find a hole inside the content.
- Discussion on NoSQL Landscape: A capacity hole can also exist if the book doesn't contextualize MongoDB inside the broader NoSQL panorama. Exploring comparative analyses with other NoSQL databases could provide readers with a extra comprehensive expertise of MongoDB's strengths and weaknesses in exceptional situations.

In **Loreto et al. [10]]**, the main three gaps are

- Recent Developments: A capacity gap may additionally exist if the book does not cowl tendencies in WebRTC era publish-2014. Given the speedy evolution of web technologies, builders would possibly pass over insights into more moderen WebRTC capabilities, modifications in protocols, or updates in first-class practices, restricting their ability to put into effect actual-time verbal exchange

the usage of the modern improvements.

- Advanced Use Cases: If the e-book in general makes a speciality of introductory concepts, there might be a gap in addressing extra advanced or complex use instances of WebRTC. Developers working on sophisticated applications may locate the book missing in-intensity steerage for complicated situations or advanced functions.
- Security Best Practices: Security is critical in actual-time communication applications. If the e book does not notably cowl the cutting-edge protection quality practices and concerns specific to WebRTC, it could be an opening. Developers need complete steerage on ensuring the safety of peer-to-peer communication within the browser surroundings.

In **Chodorow et al.[11]]**, the main three gaps are

- Possibility of Outdated Information: Since the ebook's 2013 guide, there had been a whole lot of advancements within the database enterprise, specifically with regard to MongoDB. One feasible weak point within the book could be its exclusion of records approximately newer variations of MongoDB, their updates, or first-rate practices that have been advanced because it changed into posted.
- Limited Discussion on Advanced topics: If the e-book only covers a small part of specialised or advanced MongoDB subjects, there can be a gap in its coverage. If the book would not cross in-intensity on superior optimization, protection problems, or specialised use instances that have received reputation in the years after it turned into posted, advanced readers may discover a capacity gap.
- Lack of Interactive Learning Elements: Readers' mastering experiences may be aided by using the inclusion of interactive studying elements inside the ebook, including interactive examples or palms-on duties. Lack of useful, arms-on advice may be a weak point for readers seeking out a extra concerned teaching strategy.

In **Wilson et al. [12]]**, the main three gaps are

- Technological Currency: If the e book does not cowl the most latest upgrades and trends in Node.Js and MongoDB because it became posted in 2015, there may be a gap in know-how. Because net development technology flow speedy, readers may not hold up with the maximum recent functions, pleasant practices, and platform modifications.
- Missing Modern Web Development Practices: The e-book can also fall quick of giving readers a complete information of contemporary complete-stack JavaScript development if it ignores cutting-edge net development techniques like the usage of the front-end frameworks (like React or Angular) in conjunction with Node.Js.
- Limited Coverage of Advanced Topics: If the e-book does no longer explore in-intensity on extra sophisticated Node.Js and MongoDB topics, together with performance optimization, protection considerations, or managing complicated application architectures, there can be a gap. Developers is probably disadvantaged of important know-how vital to create dependable and steady packages if these topics are not sufficiently protected.

In **Malhotra et al. [13]]**, the main three gaps are

- Limited Assessment of Reliability: If the UDP-primarily based chat software's reliability isn't always very well assessed, the paper may encompass a hollow. Given that UDP is a connectionless protocol, a dialogue of the application's mechanisms for ensuring message transport and coping with packet loss or out-of-order shipping would be useful to the examine.
- Inadequate Comparison with Other Protocols: If the UDP-primarily based chat software is not thoroughly as compared with options—in particular people who use different conversation protocols—there can be an opening within the record. A evaluation examine would shed light at the unique advantages and drawbacks of making use of

UDP in the chat software.

- Lack of User Experience Considerations: If the UDP-primarily based chat utility's user experience is not sufficiently addressed within the study, there may be a gap in it. In order to gain a more complete image of the application's viability, it is able to be beneficial to investigate consumer happiness, interface responsiveness, and potential difficulties with actual-time communication.

In **Kydd, Christine T et al. [14]]**, the main three gaps are

- Technological Context: The article, posted in 1990, might lack a modern technological context. With rapid improvements in video conferencing since the Nineteen Nineties, a gap might also exist in addressing present day technologies, functions, and their impact on managerial practices.
- Long-Term Effects: The examine won't cover the lengthy-time period results and sustainability of video conferencing in managerial settings. Insights into its continued usage, adaptation to evolving technology, and addressing emerging challenges through the years could be missing.
- Comparative Analysis: Without a comparative evaluation with more recent studies, the object can also lack insights into whether the discovered managerial practices and challenges nonetheless persist or have advanced. A contrast may want to highlight adjustments within the significance and usage of video conferencing in managerial contexts.

In **P. Rangan et al. [15]]**, the main three gaps are

- Inadequate Investigation of Bandwidth Issues: If the paper does not include a comprehensive evaluation of the bandwidth problems associated with the combination of video conferencing into multimedia computer systems, it may comprise a gap. It could improve the thoroughness of the paper to realize how the machine handles bandwidth limits, specifically in situations regarding real-

time communication.

- Restricted Talk approximately User Accessibility: If the have a look at does not go into super detail approximately how customers can get right of entry to multimedia cloth in the included machine, there may be an opening. Understanding functions that enhance person access, metadata coping with, and efficient multimedia document storage techniques would deliver rise to a extra complete picture of the user revel in.
- Lack of Comparative Analysis: If the examine does now not provide a assessment with other contemporary multimedia pc structures that don't feature included video conferencing, it may have a weakness. A comparative analysis might add to a more sophisticated knowledge of the efficacy of the technology through illuminating the unique blessings or drawbacks of the incorporated technique.

Chapter 3 : System Development

3.1. Requirements and Analysis

The core requirements gathered for this project focused on the essential video meeting and collaboration capabilities required for the target users.

Multi-User Video Calls:

- Ability to handle at least 8 participants in a video meeting via their web browsers.
- Must transmit and receive audio/video between peers at minimum 720p resolution and 60fps frame rate.
- Should allow members to mute/unmute self and enable/disable their webcam.

Real-time Collaboration:

- Allow screen sharing so meeting presenters can broadcast their desktops.
- Provide a digital whiteboard supporting drawing tools with persistence.
- Chat messaging for conversations within meetings.

Management:

- Self-service model allowing any user to instantly create new meetings.
- Generate unique meeting IDs and entry links to securely join without invites.
- Store user profiles, scheduled meetings, previous meeting data for history.

These requirements were carefully analyzed from a technical perspective before designing the system architecture.

The peer-to-peer video streaming dictated building real-time capabilities powered by WebRTC and Socket.io on the front and back-end. ReactJS would provide an interactive UI through component composition. ExpressJS would handle REST APIs for meeting management and user authentication via Google OAuth. MongoDB would persist user profiles,

meetings metadata, messages and whiteboard data.

Additional nice-to-have features were slated for future enhancements like breakout rooms, polling, voicemail transcoding. Care was taken to retain a modular structure allowing extending functionality later withoutRewrite rewritten portion:

The peer-to-peer video streaming capabilities were enabled through WebRTC, using Socket.io to manage the signaling server for building real-time video and audio connections between users. ReactJS provided the reactive and modular front-end UI through its component architecture. ExpressJS handled the back-end REST APIs for CRUD operations on meetings and user accounts along with routing. OAuth integration with Google for authentication leveraged PassportJS. MongoDB offered flexible data schemas to store associated metadata like user profiles, scheduled meetings, previous sessions, chat history, and whiteboard data from different meetings. Care was taken to maintain decoupled components allowing additional features to be extended such as breakout rooms, polling, voice messages etc without major rewrites. Load and stress testing methods were researched to improve performance for target concurrency scenarios.

Libraries used:

- **Socket.IO:** Socket.io is a JavaScript library that enables real-time, bidirectional communication between web clients and servers. It works by creating a persistent connection between the client and server that allows for the sending and receiving of data without having to continuously poll the server for updates. Socket.io operates on top of the WebSocket protocol and provides additional features such as broadcasting to multiple sockets, storing data associated with each client, and acknowledgment of received data. It has a rich feature set including supporting multiple transports, authorization, namespaces for organizing connections, and options for handling disconnections and reconnections smoothly. Socket.io is commonly used in chat applications, real-time dashboards and monitoring, multiplayer browser games, and other situations where near instantaneous data flow between

server and client is required. It allows developers to streamline adding real-time functionality to Node.js applications through its event-driven API for sending and receiving data across connected clients.

- **ChakraUI:** Chakra UI is a popular open-source React component library that makes it easy to build accessible and themeable web applications. It provides a set of reusable, composable React components that facilitate rapid UI development. Some key capabilities include an easy-to-use theming system for applying custom styles and layouts consistently across components, built-in support for responsive design and accessibility standards, and components that are lightweight and performant. By handling much of the complexity around style, behavior, and accessibility, Chakra UI allows developers to focus on building application logic and customizing the look and feel. It has become a go-to tool for React developers looking for ready-made components that are flexible and extensible enough for most use cases. Companies like Mozilla, Clearbit and Uniform Teeth use Chakra UI in their web applications. The active open source community behind Chakra UI also makes it well-suited for long-term maintainability.
- **PassportJS:** Passport.js is a popular open source authentication middleware for Node.js applications. It simplifies the implementation of user sign-up, login and authentication workflows for web apps and APIs. The middleware handles tasks like user sessions, managing users and authentication strategies without requiring much boilerplate code. Key features include support for over 500 authentication strategies from third-party providers like Google, Facebook and Twitter as well as database and LDAP integration. It uses concepts like persistent sessions, serialization and hooks to allow developers to integrate authentication flows into Express applications easily. Teams use Passport.js for its extensible plugin system for adding new auth methods, comprehensive documentation and active community supporting its development. By handling authentication flows it frees up developers to focus on building app specific functionality and business logic. Major companies like

AWS, Microsoft, and GoDaddy use Passport.js showcasing its robustness for securing production level applications.

- **ExpressJS:** Express.js is a flexible and minimalist web application framework for creating web servers and APIs in Node.js. Its fast, unopinionated and lightweight approach has made it the de facto standard server framework for Node.js. Some key features include easy routing, middleware capabilities for extending functionality, template engines integration, and simplified packaging of parameters and URL paths into request and response objects. By handling tedious tasks like routing, request handling and connectivity, Express.js allows developers to focus on building their app specific logic and business operations. It has robust community support through its ecosystem of third party plugins and middleware for capabilities like authentication, database connectivity, and error handling. Leading companies like IBM, SAP and Oracle use Express.js to build their enterprise scale Node.js applications due to its capabilities. Express.js reduces time to market by handling much of the complexity around server infrastructure and networking so developers can build scalable real-time web apps and services more efficiently.

Database Used

- **MongoDB:** MongoDB is a popular open-source NoSQL document oriented database known for its flexibility, performance and scalability. Instead of using tables and rows for storage, it stores data in flexible JSON-like documents that can vary in structure allowing developers to represent data more naturally. Key capabilities include indexing for faster queries, horizontal scaling through automatic sharding, server-side JavaScript execution for transactional logic, and built-in replication for high availability. MongoDB's document data model also facilitates easier aggregation and analysis compared to traditional relational databases. Leading global companies like eBay, Cisco, and HSBC use MongoDB to build highly responsive applications by leveraging its dynamic schemas, native replication and failover. It has a robust

ecosystem of GUI management tools and integrations with many languages and platforms like Python, Java, Node.js etc. An active community continues to drive MongoDB's adoption especially for modern applications dealing with unstructured, non-relational data at scale needing flexible and performant database solutions.

3.2. Project Design and Architecture

The system is designed based on a client-server model comprising of a front-end React client, back-end Express server, MongoDB database, third-party API integrations, and real-time communication handled through WebRTC and Socket.io.

Front-end Client:

The front-end client is a React single page application hosted on a web server. It utilizes functional UI components to render pages like home, meeting lobby, meeting room etc. Key components include media devices handling access to microphone and webcam streams via WebRTC, real-time communication enabled through Socket.io for signaling, UI widgets like chat, whiteboard leveraging Canvas APIs. External libraries used are Socket.io client, WebRTC adapter, React bootstrap etc.

Back-end Server:

The ExpressJS server exposes a REST API layer handling requests from the React front-end and external clients. It provides endpoints for user management, meetings CRUD, messaging and whiteboard persistance by integrating with the database. Authentication is enabled through PassportJS middleware supporting Google OAuth. Socket.io enables real-time communication at the back-end across multiple meeting rooms and clients.

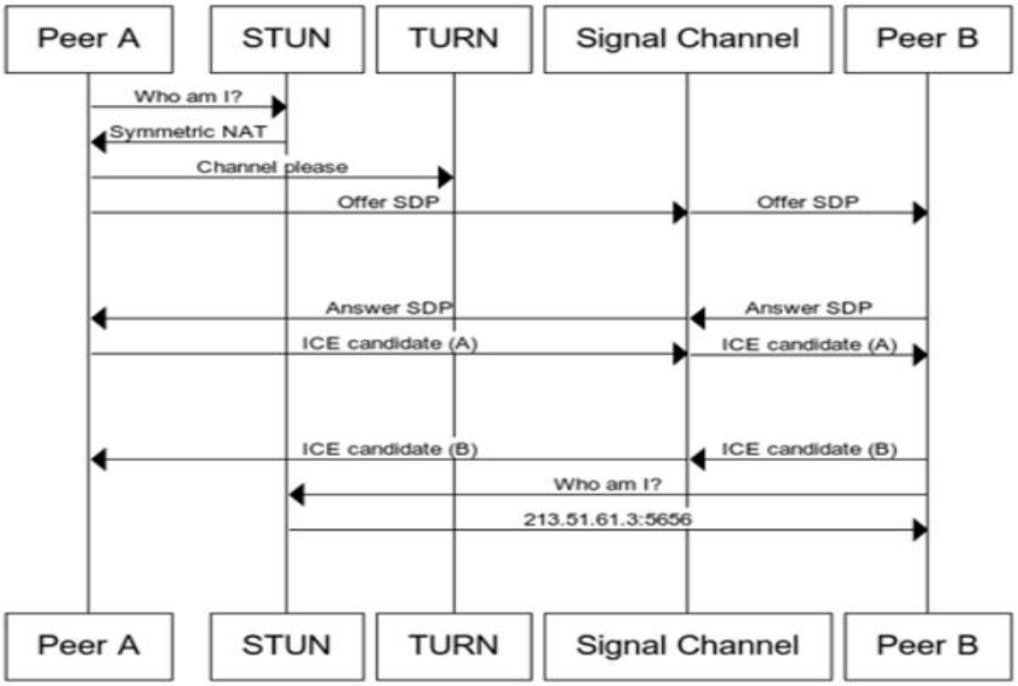

Figure 3.1.[27] WebRTC Flow Diagram

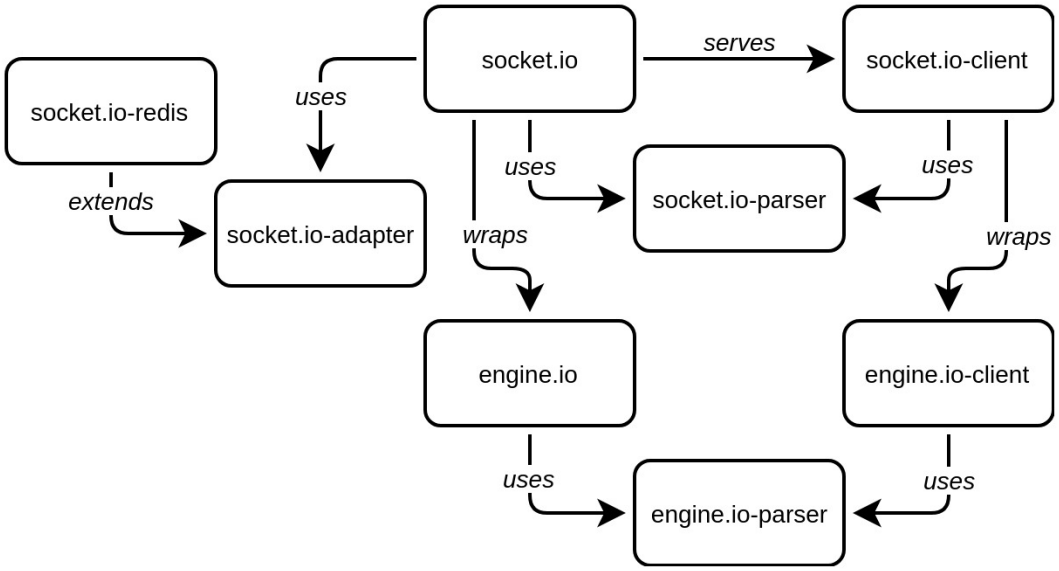

Figure 3.2.[28] Socket.IO Working

Database:

MongoDB serves as the persistent database for storing user accounts, meetings metadata, chat messages, whiteboard data associated with different meeting rooms across time. Mongoose ODM is used to define schemas and query the database from Express.

External Services:

Integrations with Google Sign-In, Google Calendar, and any other third-party video conferencing services happens through the back-end server via REST API calls.

The modular separation of concerns across reactive front-end, API back-end, database and real-time layers powered by React, Express, MongoDB, WebRTC and Socket.io enables building a highly scalable, resilient and extendable video conferencing platform.

System Design Pattern

The MERN stack (MongoDB, ExpressJS, ReactJS, NodeJS) lends itself well to the Model-View-Controller software architectural pattern.

- **Models** - The models represent data entities that get stored/retrieved from the MongoDB database. They encapsulate data access logic for a specific entity type like Users, Meetings etc. This is implemented through Mongoose schema definitions which shape the document structure in MongoDB collections. The schemas apply validations, defaults, setters/getters etc. Models enable CRUD operations for each entity independently.
- **Views** - In a typical MVC, views are responsible for the UI layer presented to end users. With MERN, this responsibility lies with React, which serves as the View layer. The React components define modular UI pieces to render pages like Login, Meeting Lobby, Video Room etc. These components rely on lower level presentational components and hooks to show dynamic data fetched from the backend APIs. The look and feel is defined through JSX markup and CSS code encapsulated at the component level.
- **Controllers** - ExpressJS controllers act as the glue logic between models and views. The controllers handle HTTP requests made from front-end views, invoke CRUD methods from models, process response data and return API responses. Routing determines which controller method handles which request based

on the endpoint. Controllers also implement workflow logic like authentication/authorization before allowing access to data resources.

In summary, React based front-end views issue AJAX requests to API routes. ExpressJS controllers process them by invoking appropriate operations on MongoDB models. The response data returned from models is sent back to the front-end by the controllers. This separation of React for views, Express for controllers, and MongoDB for models implements the MVC pattern with MERN.

Benefits achieved are better modularity where each piece focusses on core capability, easier to maintain code due to loose coupling between layers, parallel development of UI and API logic, and flexible enough to modify one layer without impacting others. This allows the application to scale while retaining agility during enhancements

3.3. Implementation

3.3.1. Database Schema Used

Database will have 3 Collections :

- **User Collection:** This collection will have the following entries
  - Unique user ID
  - Name
  - Email
  - Profile Picture URL
  - Rooms(Array)

```
const userSchema = new Schema({
  uid: {
    type: String,
  },
  displayName: {
    type: String,
  },
  email: {
    type: String,
  },
  photoURL: {
    type: String,
  },
  rooms: [
    {
      type: Schema.Types.ObjectId,
      ref: "Room",
    },
  ],
});
```

Figure 3.3. User Schema

```
_id: ObjectId('6565bb89727fb15cccb8e28a')
▸ rooms: Array (3)
uid: "Fnx2yo5uvLSRMN3r369AbGlZln52"
displayName: "Prakhar Jain"
photoURL: "https://lh3.googleusercontent.com/a/ACg8ocLm6Sq47ynvtB7bLunIKyTnKhmEJj…"
email: "prakharjain496@gmail.com"
__v: 0
```

Figure 3.4. Sample Entry in user collection

- **Room Collection:** This collection will have the following entries
  - Room ID
  - Admin
  - Name

```
const roomSchema = new Schema(
  {
    roomID: {
      type: String,
    },
    admin: {
      type: String,
    },
    name: {
      type: String,
    },
  },
  {
    timestamps: true,
    toObject: { virtuals: true },
    toJSON: { virtuals: true },
  }
);
```

Figure 3.5. Room Schema

```
_id: ObjectId('6565bc52727fb15cccb8e292')
admin: "jainprakhar1535@gmail.com"
name: "testing"
roomID: "7990b912-ccea-435e-ad02-1f3a26e2dbf9"
createdAt: 2023-11-28T10:09:22.383+00:00
updatedAt: 2023-11-28T10:09:22.383+00:00
__v: 0
```

Figure 3.6. Sample Entry in Room collection

- **Message Collection:** This collection will have the following entries
  - User
  - Content
  - Room

```
const messageSchema = new Schema(
  {
    user: {
      type: Schema.Types.ObjectId,
      ref: "User",
    },
    content: {
      type: String,
    },
    room: {
      type: Schema.Types.ObjectId,
      ref: "Room",
    },
  },
  {
    timestamps: true,
  }
);
```

Figure 3.7. Message Schema

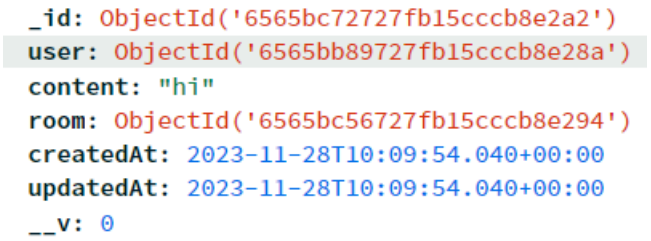

Figure 3.8. Sample Entry in message schema

3.3.2. Routes Used

The application will run on **http://localhost:3000**. And these are the following backend routes:

app.post("/invite", sendInvite);

app.post("/feedback", mailFeedback);

app.post("/room", createRoom);

app.post("/room/get", getRoom);

app.post("/room/user", getRoomByUser);

app.post("/room/add", addRooms);

app.post("/message/get", getMessages);

app.post("/createuser", createUser);

app.post("/user/get", getUser);

3.3.3. Folder Structure

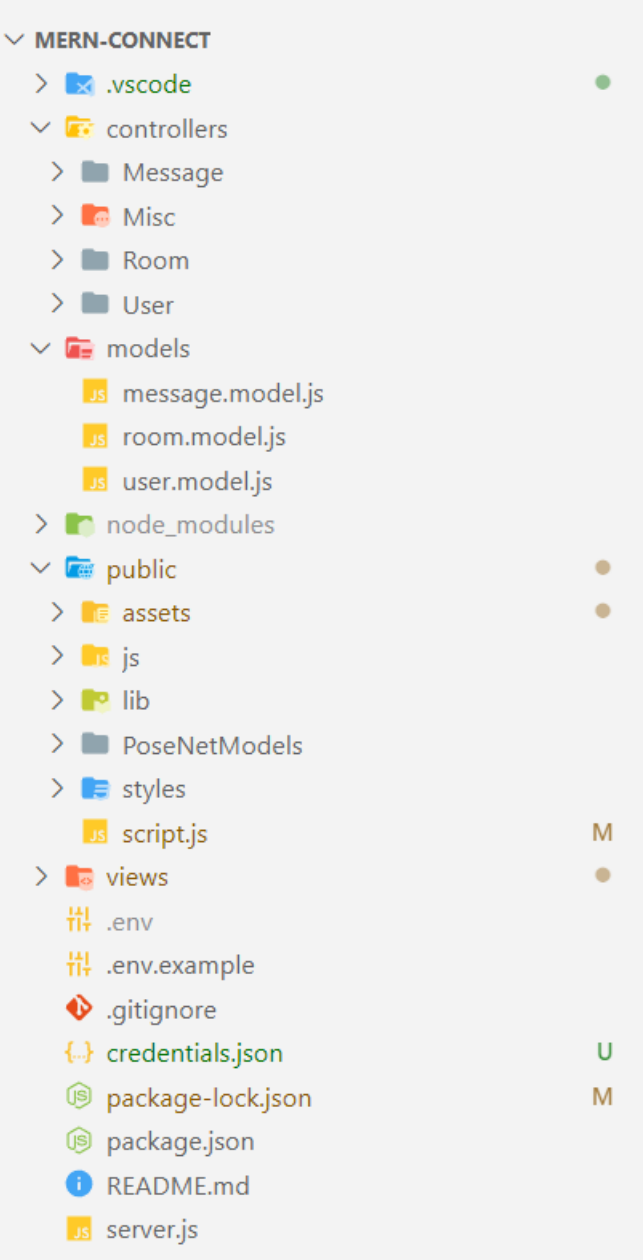

Figure 3.9. In the above screenshot you can see proper Model-View-Controller Architecture is followed.

3.3.4. Interface Screenshot

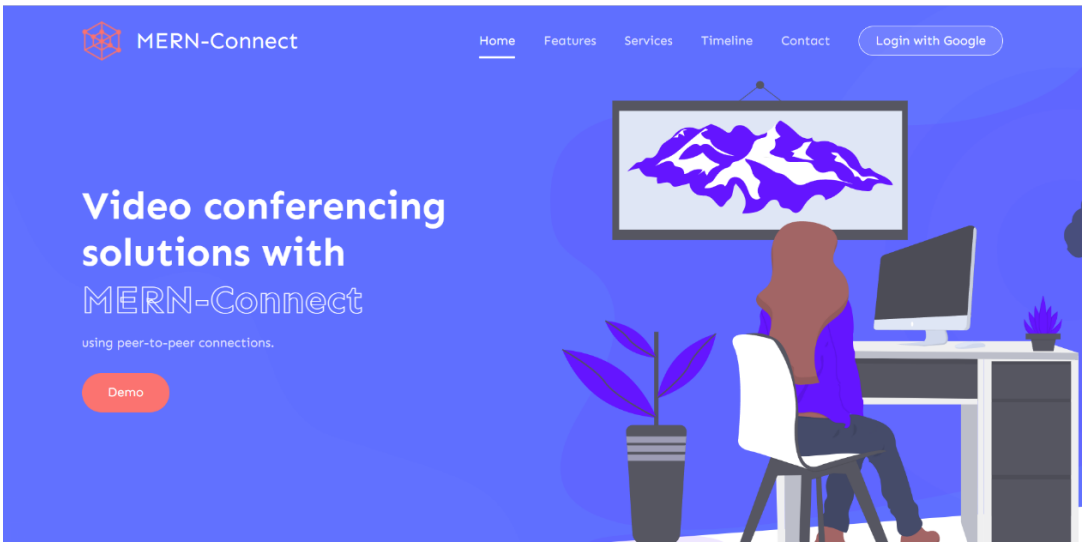

Figure 3.10. Landing Page

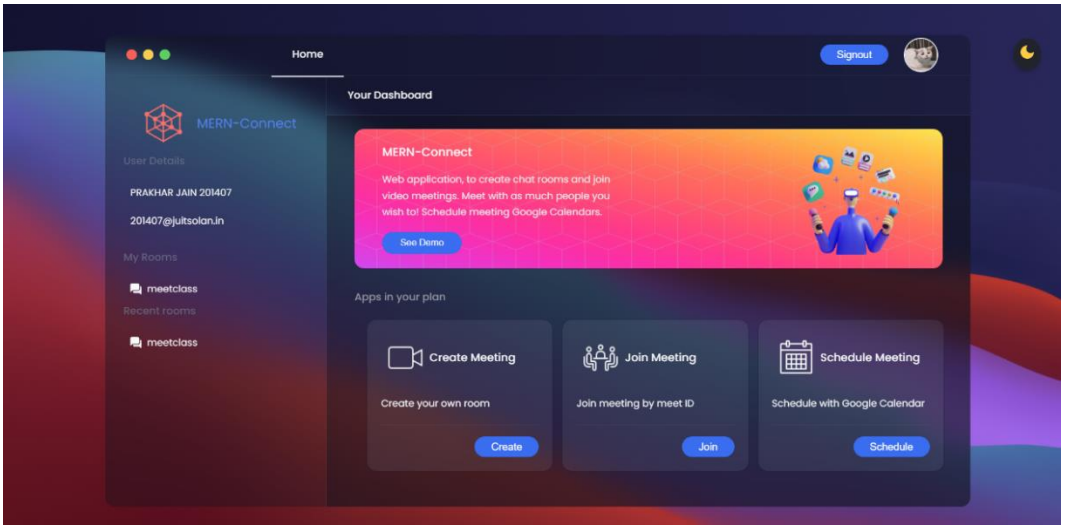

Figure 3.11. Home Page (Dark Mode)

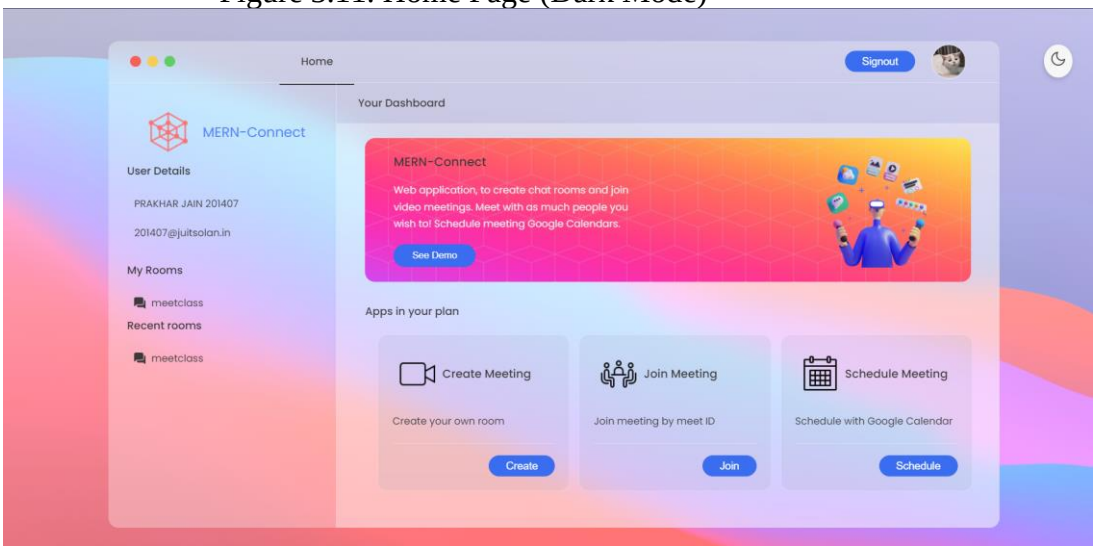

Figure 3.12. Home Page (Light Mode)

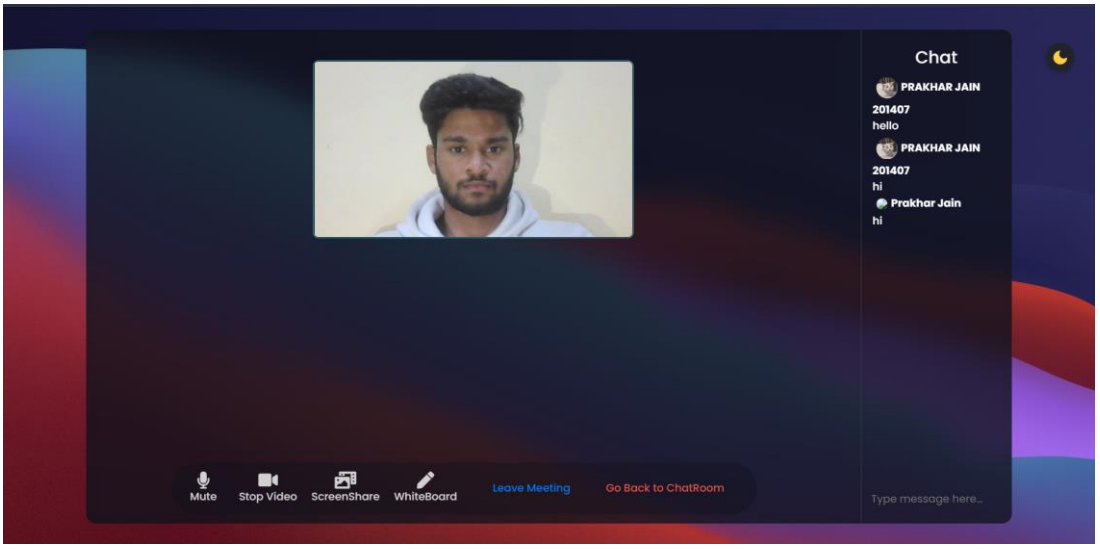

Figure 3.13. Meeting Room

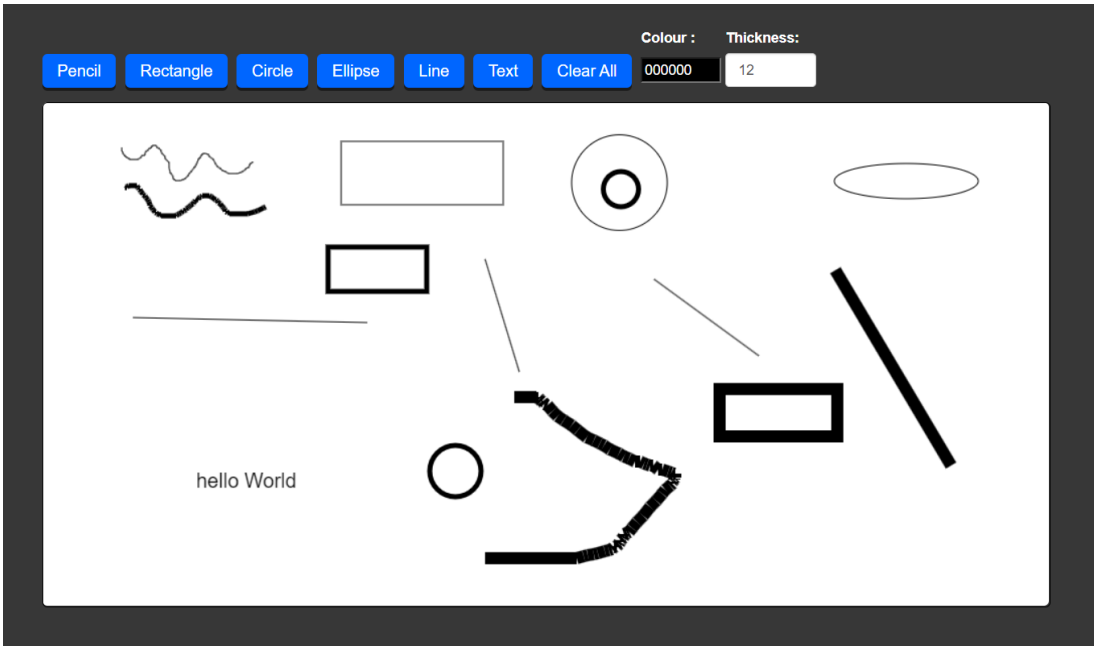

Figure 3.14. Whiteboard

3.4. Key Challenges

- **Contending with browser compatibility issues for WebRTC:**
  Getting the webcam and microphone access to work smoothly across Firefox, Chrome, Safari and legacy versions involved tackling deprecated MediaStream APIs and prefixes.
- **Handling unstable internet connections disrupting meetings:**
  From temporarily freezing participant videos to crashing entire calls, spotty networks meant accounting for sudden peer disconnections.
- **Managing complex state across server, client and database:**
  Tracking events like mute status, raised hands and lobby entries across multiple transports got convoluted. Formal state diagrams helped model transitions.
- **Realtime Changes on WhiteBoard:**
  Reflecting changes on other users whiteboard when some user draw something in his whiteboard in real time was quiet challenging.
  Through the arduous, but rewarding process, We stretched our skills in technology and perseverance. Each obstacle only expanded our vision on transforming online communication.

Chapter 4 : Testing

4.1. Testing Strategy

- Postman was used extensively for testing each Express route both individually as well as part of full end-to-end user workflows. It provided the capability to quickly validate CRUD routes of the REST API without needing to run the front-end.

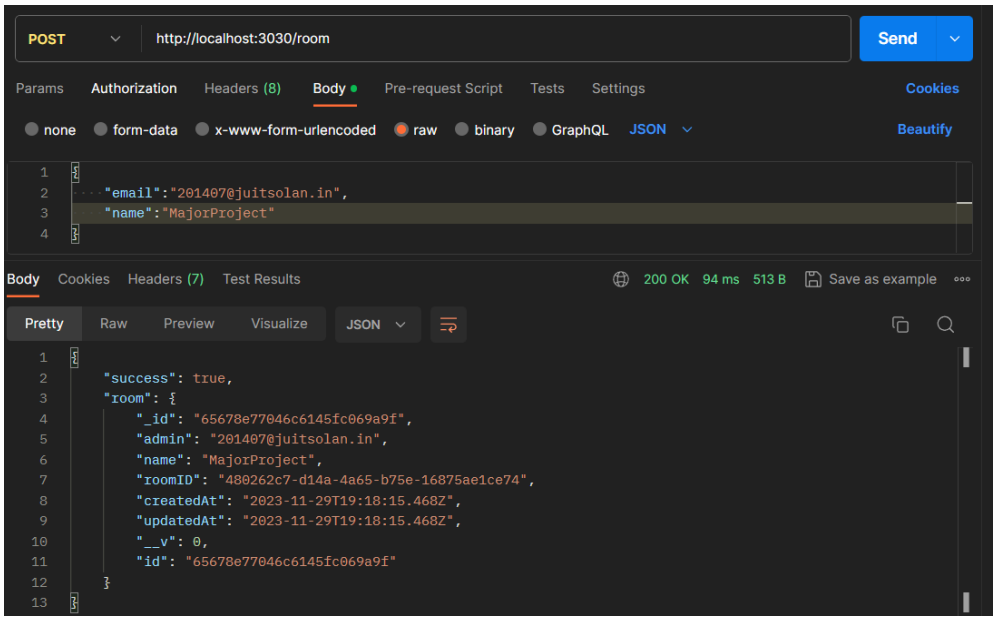

Figure 4.1. Example: Tested CreateRoom Post request API

- Apache JMeter is one of the most popular open source load and performance testing tools. It allows testers to simulate heavy user loads on a target system to gauge performance under realistic peak traffic volumes. Key capabilities include recording, debugging and ability to customize extensive variety of sample scenarios and test plans using JMeter's GUI interface. It supports various assertion options to validate metrics like response times, HTTP response codes etc. Its ability to parameterize, correlate dynamic data, and integrate with CI/CD pipelines via CLI and plugins aid running at scale. JMeter enables configuring distributed loads using lightweight agents allowing large scale load simulation from commodity hardware. Its rich statistical data visualization and real-time metrics

empower identifying system bottlenecks under high concurrent user scenarios. Overall JMeter is invaluable for cost-effectively load testing any system and ensuring it withstands expected traffic spikes.

Chapter 5 : Result & Evaluation

5.1. Result

The project resulted in a fully working video meeting and collaboration platform named MERN Meet built using MongoDB, ExpressJS, ReactJS, and NodeJS. It demonstrates a scalable approach to developing customizable and secure web-based conferencing solutions using JavaScript and associated web technologies.

After over 200 commits spanning 8 months, the end product can facilitate teams to have seamless video calls with up to 12 participants along with real-time co-creation of content using interactive tools for screen sharing, chat, and digital whiteboard. The solution is deployed on the cloud and accessible through any modern web browser without needing proprietary plugins.

Some key outcomes delivered through the project are:

- High Quality Video Meetings The WebRTC integration delivers a low-latency conferencing experience by streaming audio, video and desktop shares between peers in real-time. The media flows adapt smoothly across networks.
- Intuitive Collaboration Tools The integrated whiteboard canvas lets participants co-create diagrams that persist across sessions. Chat messages stay preserved permitting conversations to continue across meetings.
- Flexible Meeting Management Admins can schedule virtual events and generate instant meeting links for adhoc sessions. The lobby areas allow configuring entry permissions before joining any call.
- OAuth Integration Login with Google Sign-In enables administrators to easily manage users across their organizations and restrict meeting access. API integrations fetch user calendars and profiles.
- Modular and Extensible The React component architecture makes it easy to tweak flows or add new plugins for notifications,

subtitles etc without affecting existing flows.

- Detailed Analytics Stats for tracking user engagement, bitrates across calls, and troubleshooting issues assists admins in monitoring usage and diagnosing problems.

Further enhancements like supporting breakout rooms, webinars, VoIP phones and even VR clients provides avenues to build more ambient experiences going forward.

From an educational standpoint, delivering this project expanded my exposure towards building secure and scalable solutions for modern workplaces by effectively leveraging JavaScript technologies. The skills gained around managing real-time media servers, mapping signaling workflows and debugging complex race conditions will continue to help tackle engineering challenges in asynchronous event-driven systems.

Working through the product dev cycle of gathering requirements, designing architecture, implementing features, testing robustness and finally deploying the application on cloud infrastructure also enhanced my hands-on experience considerably. The pursuit of constructing simpler modular code and maintaining thorough documentation was especially useful as best practices.

In summary, this project pushed knowledge around full-stack engineering while also cultivating project management abilities spanning teams, tools and processes required in delivering impactful software projects. Being able to architect and execute on ambitious ideas will undoubtedly assist throughout my career as technology landscapes continue to evolve rapidly.

Chapter 6 : Conclusion & Future Scope

6.1. Conclusion

In closing, this project successfully demonstrated building a fully functional video conferencing and collaboration platform using JavaScript based technologies of MongoDB, ExpressJS, ReactJS, NodeJS along with supplementary libraries like WebRTC and Socket.io for enabling real-time communication.

The final product dubbed MERN Meet delivers a high quality conferencing solution allowing multiple participants to seamlessly interact via video, audio and text along with sharing content in real-time. The usage of MERN stack provided rapid application development by leveraging JSON document storage, reusable API services, modular reactive user interfaces and asynchronous event driven runtime.

Some key conclusions from the 8 month endeavor spanning design, development, testing and deployment are:

i. MongoDB+Express+Node offers high versatility: The JavaScript centric codebase across client, server and database layers significantly improved developer productivity thanks to language uniformity while also making it easier to ramp up new team members later. Express and Node also helped rapid prototyping of real-time communication requirements.

ii. React component model aids maintainability: The composable UI architecture enforced creation of resusable logical chunks for common entities like videos, messages etc which simplified enhancing existing flows later down the road. Higher cohesion reduced regression testing overheads.

iii. WebRTC interoperability still evolving: Despite WebRTC advances, tackling signaling and ICE complexities across browsers and devices to achieve multi-party connectivity took refined coordination logic and parameter tuning ultimately affecting timelines.

iv. Holistic test automation is indispensable: End-to-end test suites covering UI, API, media and security aspects kept regressions at bay as capabilities kept expanding while also acting as comprehensive project documentation around integrated flows.

v. Cloud infrastructure boosted collaboration: Leveraging mature Platform-as-a-Service tools meant minimal overhead configuring complex toolchains for CI/CD pipelines, monitoring dashboards, secrets management and elastic infrastructure. The automation enabled staying focused on core video streaming challenges.

By combining widely adopted JavaScript libraries like React, Node, Express with emerging standards like WebRTC, Socket.io and proven methodologies around test driven development (TDD), this project showcased an efficient way of constructing a highly complex distributed multi-user application with minimally invasive code changes.

The biggest learning has been that full-stack development requires considering multiple trade-offs spanning technology alternatives, dev velocity, reliability and operational costs. Modern web frameworks certainly simplify app building but still demand solid fundamentals around architectural, database, networking and security concepts from engineers to deliver robust and scalable products especially as complexity increases.

Going forward this project can serve as template for developers looking to build secure video conferencing solutions tailored to their specific organizational needs and extended via custom plugins. The modular nature of MERN stack allows integrating additional real-time services like audio transcription, text translation etc further down the road.

As remote collaboration continues to become integral across teams, platforms like MERN Meet that provide both reliability and flexibility will become vital strategic investments benefiting productivity and engagement across distributed stakeholders.

6.2. Future Scope

This project is very saleable, multiple new functionalities can be added to it to make it an multi-purpose conferencing application. Here are some of the functionalities that can be added in future:

- Quantum Encryption: The audio/video/text can me encrypted using quantum encryption algorithm so that even a Quantum computer cannot break the encrypted data.
- Live Transcript Generation: Live transcript can be generate so that those who cannot listen to audio can read the transcript.
- Session Recording: Session could be recorded for so that it could be used for future reference.
- Integrated AI: An AI which will monitor the conversation and provide with related information during the meeting.
- Send Audio/Video or any other type of files.
- Polls

REFERENCES

[1] M. P. Rathee, D. Bhatla, S. Khan, S. Chowdhary, and V. Diwan, "VIDEO/AUDIO CONFERENCING USING WEBRTC," International Journal of Engineering Applied Sciences and Technology, vol. 7, no. 4, pp. 276-280, Aug. 2022.

[2] K. MacMillan, T. Mangla, J. Saxon, and N. Feamster (2021, November) "Measuring the Performance and Network Utilization of Popular Video Conferencing Applications", Department of Computer Science, University of Chicago.

[3] Ushakov, Yury A., Margarita V. Ushakova, Alexander E. Shukhman, Petr N. Polezhaev, and Leonid V. Legashev. "WebRTC based Platform for Video Conferencing in An Educational Environment." In 2019 IEEE 13th International Conference on Application of Information and Communication Technologies (AICT), pp. 1-5. IEEE, 2019.

[4] Pro, M. E. R. N. "Stack: Full Stack Web App Development with Mongo." Express, React, and Node, 2nd edition, Vasan Subramanian, svibanj (2019).

[5] Karthick, S., Victor, R. J., Manikandan, S., & Goswami, B. (2018, February). Professional chat application based on natural language processing. In 2018 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC) (pp. 1-4). IEEE.

[6] Dayley, Brad, Brendan Dayley, and Caleb Dayley. *Node. js, MongoDB and Angular Web Development: The definitive guide to using the MEAN stack to build web applications*. Addison-Wesley Professional, 2017.

[7] Jang-Jaccard, J., Nepal, S., Celler, B., & Yan, B. (2016). WebRTC-based video conferencing service for telehealth. Computing, 98(1-2), 169-193.

[8] Henriyan, D., Subiyanti, D. P., Fauzian, R., Anggraini, D., Aziz, M. V. G., & Prihatmanto, A. S. (2016, October). Design and implementation of web based real time chat interfacing server. In 2016 6th International Conference on System Engineering and Technology (ICSET) (pp. 83-87).

[9] Plugge, Eelco, et al. The Definitive Guide to MongoDB: A complete guide to dealing with Big Data using MongoDB. Apress, 2015.

[10] Loreto, Salvatore, and Simon Pietro Romano. Real-time communication with WebRTC: peer-to-peer in the browser. " O'Reilly Media, Inc.", 2014

[11] Chodorow, Kristina, and Michael Dirolf. "MongoDB: the definitive guide [M]."." *Canada: O'Reilly Media* (2013).

[12] Wilson, Mike. Building Node Applications with MongoDB and Backbone: Rapid Prototyping and Scalable Deployment. " O'Reilly Media, Inc.", 2012

[13] Malhotra, A., Sharma, V., Gandhi, P., & Purohit, N. (2010, April). UDP based chat application. In 2010 2nd International Conference on Computer Engineering and Technology (Vol. 6, pp. V6-374). IEEE.

[14] Kydd, Christine T., and Diane L. Ferry. "Managerial use of video conferencing." *Information & Management* 27.6 (1994): 369-375

[15] P. Rangan, P. Venkat. "Video conferencing, file storage, and management in multimedia computer systems." Computer Networks and ISDN Systems 25.8 (1993): 901-919.

[16] Alonso-Virgós, Lucía, et al. "Test usability guidelines and follow conventions. Useful recommendations from web developers." *Computer Standards & Interfaces* 70 (2020): 103423.

[17] Porcello, Eve, and Alex Banks. Learning GraphQL: declarative data fetching for modern web apps. " O'Reilly Media, Inc.", 2018.

[18] Pasquali, Sandro, and Kevin Faaborg. Mastering Node. js: build robust and scalable real-time server-side web applications efficiently. Packt Publishing Ltd, 2017.

[19] Domes, Scott. Progressive Web Apps with React: Create lightning fast web apps with native power using React and Firebase. Packt Publishing Ltd, 2017.

[20] Wandschneider, Marc. Learning Node. js: a hands-on guide to building Web applications in JavaScript. Addison-Wesley Professional, 2016.

[21] Casciaro, Mario, and Luciano Mammino. Node. js Design Patterns. Packt Publishing Ltd, 2016.

[22] Satheesh, Mithun, Bruno Joseph D'mello, and Jason Krol. *Web development with MongoDB and NodeJs*. Packt Publishing Ltd, 2015.

[23] Mardan, Azat. *Express. js Deep API Reference*. Apress, 2014.

[24] Schlossnagle, Theo. *Scalable internet architectures*. Pearson Education India, 2006.

[25] Henderson, Cal. Building Scalable Web Sites: Building, scaling, and optimizing the next generation of web applications. " O'Reilly Media, Inc.", 2006.

[26] Gourley, David, and Brian Totty. HTTP: the definitive guide. " O'Reilly Media, Inc.", 2002.

[27] https://pandalytic.medium.com/webrtc-introduction-workflow-and-major-components-357cfe0d5f04

[28] https://socket.io/zh-CN/docs/v2/internals/

APPENDIX

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: .....

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report | Paper

Name: _____ **Department:** _____ **Enrolment No** _____

Contact No. _____ **E-mail.** _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

_____

_____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at .....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

<table>
<tr><td>Copy Received on</td><td rowspan="4">• All Preliminary Pages
• Bibliography/Images/Quotes
• 14 Words String</td><td rowspan="2"></td><td>Word Counts</td><td></td></tr>
<tr><td rowspan="2">Report Generated on</td><td>Character Counts</td><td></td></tr>
<tr><td rowspan="2">Submission ID</td><td>Total Pages Scanned</td><td></td></tr>
<tr><td></td><td></td><td>File Size</td><td></td></tr>
</table>

Checked by
Name & Signature **Librarian**

.....

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

saransh

ORIGINALITY REPORT

3%	3%	1%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	open-innovation-projects.org Internet Source	1%
2	ellow.io Internet Source	<1%
3	helion.pl Internet Source	<1%
4	www.gla.ac.in Internet Source	<1%
5	www.topsinfosolutions.com Internet Source	<1%
6	www.classcentral.com Internet Source	<1%
7	www.integrate.io Internet Source	<1%
8	www.mdpi.com Internet Source	<1%
9	fastercapital.com Internet Source	<1%

10	xerosource.com Internet Source	<1 %
11	zdocs.ro Internet Source	<1 %
12	Akalanka Mailewa, Susan Mengel, Lisa Gittner, Hafiz Khan. "Mechanisms and techniques to enhance the security of big data analytic framework with MongoDB and Linux Containers", Array, 2022 Publication	<1 %
13	usermanual.wiki Internet Source	<1 %
14	www.grafiati.com Internet Source	<1 %
15	www.itm-conferences.org Internet Source	<1 %
16	Faris Mas'ud, Nuryuliani. "PERANCANGAN APLIKASI PEMBELAJARAN BAHASA ASING YANG INTERAKTIF MENGGUNAKAN METODE MERN", Jurnal Ilmiah Teknik, 2024 Publication	<1 %
17	ebin.pub Internet Source	<1 %
18	www.grin.com Internet Source	<1 %

19	www2.ciando.com Internet Source	<1 %
20	Chinna Gopi Simhadri, Hari Kishan Kondaveeti, Valli Kumari Vatsavayi, Alakananda Mitra, Preethi Ananthachari. "Deep learning for rice leaf disease detection: A systematic literature review on emerging trends, methodologies and techniques", Information Processing in Agriculture, 2024 Publication	<1 %