Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

An Enhanced MERN-Based Ed-Tech Platform with Integrated Live Classroom Support

Ashish Kumar Saini
Department of Computer Science and Engineering
Galgotias University, Greater Noida, India

Priyanshu Raj Chauhan
Department of Computer Science and Engineering
Galgotias University, Greater Noida, India

Dr. Shikha
Galgotias University, Greater Noida,
India

***Abstract*—The emergence of digital education and the growing use of online learning paradigms have increased the need of scalable, flexible and interactive education technology (Ed-Tech) platforms. Most of the current systems have been known to be effective in providing asynchronous learning by use of recorded lectures,digital materials and self pacing courses, however they do not necessarily provide real time classroom learning. This is a restriction that limits the interaction of the learners, limits the real time interaction between the students and the instructors, and also impacts the learning process negatively. To overcome such challenges, this paper proposes a scalable MERN based Ed-Tech platform that provides live classroom features in a course-based learning system. The system proposed is built based on MERN (MongoDB, Express.js, React.js, and Node.js) technology stack that will make it modular, scaled, and maintainable. Live classes Real-time live classes are integrated right into the platform directly through the Jitsi Meet API, which avoids the use of third-party conferencing services and also allows secure and interactive classes. Besides that, the platform uses the RESTful API architecture and JWT-based authentication and role-based access control to improve the security and extensibility. Experimental and practical tests reveal that the integrated live classroom application enhances the engagement of the learners, helps in real-time communications, and ensures the consistent performance even when the users share the use. The proposed platform is an effective hybrid solution in the current digital environment of learning because it combines asynchronous learning materials with synchronous teaching.**

***Index Terms*—Ed-Tech platform, MERN stack, Live classroom, Jitsi Meet API, Hybrid learning**

I. Introduction

A. Motivation and Problem Statement

The fast development of online education has revolutionized the process of dissemination, consumption, and assessment of knowledge around the world. The recent cloud computing developments, fast internet connectivity, and sophisticated web-centered application models have facilitated the use of online learning platforms in large-scales. This transition was also hastened by the COVID-19 pandemic that forced education facilities to swiftly change their models of classroom-based instruction to fully online or hybrid education [14]. According to the recent researches, an important percentage of teachers and students now use hybrid learning strategies, which are an integration of asynchronous delivery of content and synchronous instruction strategies [2].

Although this has grown, the majority of modern educational technology (Ed-Tech) is still more asynchronous-oriented in nature. The main types of such systems offer video lectures in recorded form, downloadable materials, quizzes, and self-paced tests. Although this method is flexible and accessible, in most cases, it does not meet a number of key pedagogic needs. Lack of face-to-face interaction between instructor and students does not allow resolving doubts instantly, solving problems collectively and engaging in meaningful classroom activities. Consequently, students can lose their motivation, receive a delayed feedback, weakening the sense of academic community [4].

StudyNotion proves that the self-paced learning environment is effective in asynchronous-first platforms, which allow access to educational materials in a global area and flexibly. The real-world learning situations, however, often require synchronous learning, especially when students are introduced to complex learning material, need to be guided in solving a problem, or take part in real-time discussion. Absence of native live classroom integration compels most of the platforms to use external video conferencing tools, which lead to discontinuous workflow, security issues, and limited learning continuity [3].

B. Research Objectives

This study is driven by these constraints which suggest the following proposal: design and development of an improvement of MERN-based. Ed-Tech platform which works in harmony with synchronous. live classroom experience in a conventional course-based. learning environment. Instead of making changes to an already existing. system, the proposed platform is made in house. based on the architectural ideas of asynchronous learning. platforms, though correcting their inadequacies by means of native. live interaction support.

The primary objectives of this research are as follows:

- To design a scalable and modular Ed-Tech platform using the MERN (MongoDB, Express, React, and Node.js) stack that supports both asynchronous and synchronous learning [6].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

- To integrate real-time live classroom functionality using the Jitsi Meet API without relying on external redirection or third-party meeting workflows [13].
- To develop database schemas and backend services capable of managing live sessions, scheduling, and role-based access.
- To implement secure authentication and authorization mechanisms using JWT-based security models [11], [12].
- To evaluate system performance, usability, and learner engagement under practical usage scenarios.
- To provide a replicable architectural blueprint for hybrid learning platform development.

C. Contributions

The key contributions of this research are summarized as follows:

- **Hybrid Learning Architecture:** A unified MERN-based system architecture that seamlessly combines asynchronous course delivery with synchronous live classroom interaction [2].
- **Integrated Live Classroom Module:** Native embedding of Jitsi-based live sessions within the platform, eliminating dependency on external conferencing tools.
- **Secure Role-Based Design:** Implementation of JWT-based authentication and role-based access control to ensure secure instructor and student interactions.
- **Scalable System Design:** A stateless backend architecture that supports horizontal scaling and concurrent user access [15].
- **Educational Impact Analysis:** Functional evaluation demonstrating improved learner engagement, real-time interaction effectiveness, and system responsiveness.

II. Related Work

However, so fast has been the emergence of online education that there has been a proliferation of digital learning systems, learning management systems (LMS), and virtual classrooms. These systems can be loosely divided into asynchronous learning platform, synchronous communication tools and hybrid learning environments [2].

A. Asynchronous Learning Platforms

Most modern Ed-Tech systems are based on asynchronous learning platforms. Providers of Massive Open Online Courses (MOOCs) like Coursera, edX, and Udemy are facilitating mass access to recorded lecture content, reading resources, tests, and self-assessment. These platforms underline the flexibility, where the learner can move and learn at his/her own pace without time and place limitations. Although scalable and globally accessible, this model provides a low degree of interaction in real time between instructors and learners. Questions posed by students are usually resolved by the use of discussion group or delayed feedback opportunities and this might lower interaction and slow down effective explanation of the complicated topics [5].

StudyNotion is an Ed-Tech platform, which is MERN-based and is mainly dedicated to asynchronous learning. It offers organized course management, user identification and content delivery via a contemporary full stack web platform [1]. Despite the effectiveness of self-paced education on such platforms, real time classroom interaction is not one of the architectural features inherently provided by these platforms. Consequently, synchronous communication is not part of the learning process, which restrains the role of pedagogy in the context where live teaching is necessary.

B. Synchronous Communication Tools

Most of the online communication tools used in online instruction have been embraced in real time, such as Zoom, Google Meet, and Microsoft Teams. Such platforms can provide powerful video conferencing options, including screen sharing, live chat, and break out rooms. They were necessary during the COVID-19 pandemic as one of the means of conducting virtual classes. These systems are however not structured as integrated platforms of learning, but as general purpose communication systems.

The biggest weakness of such tools is that they do not integrate well with course management systems. Live sessions can be based on unrelated material to course material, enrollment, and assessment history. This division causes fragmented learning experience, more administratively overhead and possible security issues due to unauthorized access and privacy of information [4].

C. Hybrid Learning Models

Hybrid models of learning strive to embrace the merits of asynchronous and synchronous learning by incorporating recorded materials with live educational lessons. Recent scholarly research has revealed that the hybrid frameworks are much more effective in terms of engagement, retention and satisfaction levels that the asynchronous frameworks [2], [3]. Live interaction provides immediate feedback and cooperative problem-solving and more robust relations between the instructor and the student, whereas recorded materials ensure the flexibility.

In spite of these benefits, most current hybrid designs are based on the loosely integrated adoption of LMS solutions with third-party video conferencing tools. These solutions do not provide a smooth user experience, single authentication system, and uniformity of data management. What is more is that, there are limited systems that offer architectural blueprints that depict the way hybrid learning can be put in place as a native aspect of modern web application stacks.

D. Research Gap and Motivation

The literature review and analysis of the research have demonstrated the presence of an evident gap in the design of Ed-Tech systems that inherently incorporate synchronous live classroom feature into an efficient and secure full-stack structure. A majority of asynchronous platforms do not have implemented real-time interaction and most of the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

synchronous tools are independent of learning management systems. The research published on hybrid learning points out the pedagogical advantages of using real-time interaction, but lacks much information in terms of practical implementation of the system.

Inspired by these findings, the proposed study suggests a MERN-based Ed-Tech application that can easily combine the live classroom experience with the help of the Jitsi Meet API [13]. In contrast to the current methods, the given system integrates the synchronous interaction into the platform structure, which presents a single, safe, and scalable hybrid digital learning solution.

III. Proposed System Architecture

The Ed-Tech platform described in the paper is implemented on a scalable, secure, and modular architecture, which is built on the MERN (MongoDB, Express, React, and Node.js) technology stack. The system is built up based on the aim of providing an integration of asynchronous course based learning and synchronous live classroom interaction. Fig. 1 describes the structure of the proposed platform.

A. Architectural Overview

The framework is client server based and it has distinct separation of concerns between presentation layer, application logic layer and data layer. Such a separation makes them maintainable, scalable, and easy to enhance in future. There are two main types of users on the platform, namely: Student and Instructor, each has their respective roles and functions, as well as permissions to access particular features.

The frontend takes into consideration user interaction, course exploration and access to live classes whereas the backend deals with authentication, authorization, business logic and communication with external services. The database layer retains user information, course information, and active session data.

B. Frontend Architecture

The frontend is built with the help of React, a component-based toolkit written in JavaScript, which allows creating the responsive and dynamic user interface [9]. The application is set up in reusable units and pages, such as authentication views, dashboards, course listing and live class interfaces.

The role-based rendering is used to display various dashboards to students and instructors. Students will be able to view courses, enroll in courses of choice, and attend scheduled live classes. The instructors are given interfaces where they can develop courses, maintain the registered students and even the live classroom sessions. React hooks and context are used in managing state so that there is a consistent flow of data in the components.

C. Backend Architecture

The backend will be developed in Node.js and Express.js with a lightweight and scalable server-side environment [7], [8]. RESTful APIs are open to be used to manage the main functions like user authentication, courses handling, enrolment, and live classes scheduling.

Back end is based on a layered architecture where routing logic, controller logic and business services are separated. This design not only makes the tests more testable but also enables one to independently make changes to individual modules. Based on JWT authentication is a protection mechanism of authenticated endpoints so that only authorized users can have access to sensitive resources according to their assigned roles.

D. Database Design

MongoDB is the main data store because of the flexibility in the schema design and scalability [10]. The database has collections of users, course and live class sessions. The authentication credentials, role data, and enrolment are saved in user documents. Metadata in course documents includes course description, instructor association and students enrolled in the course.

Live class documents hold the information regarding the schedules, course identifiers, instructor, and unique session identifiers that are utilized in the integration of live classrooms. The schema design allows querying efficiently and the extensions to be made in the future like attendance tracking and recording sessions.

E. Live Classroom Integration Architecture

The proposed system brings in the integration of live classroom functionality as the fundamental contribution. Live sessions are used with the help of the Jitsi Meet API that allows arranging real-time video conferencing [13]. In the case of a live lesson created by an instructor, the backend will create a special identifier of a meeting room and connects it to the data about a course and a session.

Students who attend the course are given the right to attend the live session at the designated time. The frontend incorporates the Jitsi interface into the platform and allows a smooth transition between course materials and live instruction. The media streams are fully managed by the infrastructure of Jitsi and therefore there is a low latency and the application server has lesser computation load. Fig. 2 live classroom integration process illustrates the workflow of the live classroom integration process.

F. Security and Scalability Considerations

JWT-based authentication, secure API communication, and role-based access control are implemented as a means of security [11], [12], [16]. Environment-based configuration is used to manage sensitive credentials. The vertically-scaled system is stateless, which means that it can be scaled horizontally and could receive more users due to the load balancing approach and cloud deployment.

The architectural design makes it possible to ensure that other features like session recording, analytics, and mobile are added without causing major reorganization of the system.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

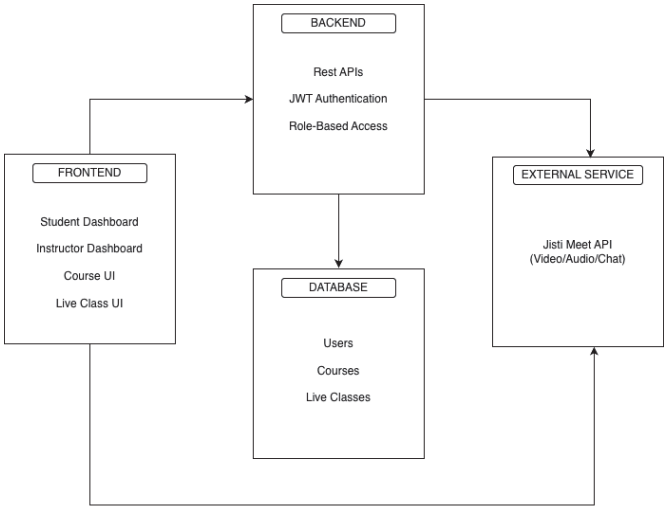

Fig. 1. Overall System Architecture of the Proposed Ed-Tech Platform

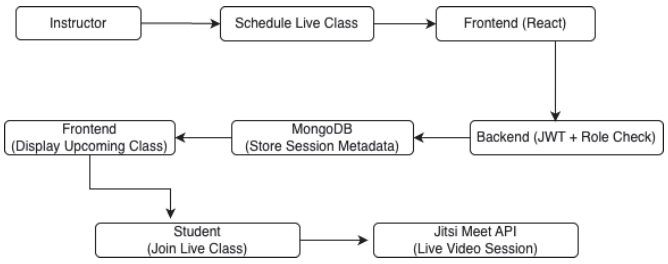

Fig. 2. Workflow of Live Classroom Session Integration

IV. Live Classroom Integration

The proposed Ed-Tech platform introduces the fundamental value in the form of the integration of synchronous live classroom functionality. Conventional asynchronous learning systems are not always associated with real-time interaction and this restricts real-time feedback, collaborative learning and instructor-student interaction. In order to overcome these shortcomings, the proposed system will be based on the natively integrated live classroom support through the Jitsi Meet API, whereby real-time communication will occur within the same learning platform [13].

A. Design Rationale

The choice of Jitsi Meet to be used as a live conferencing solution is driven by the fact that it is open-source and has a WebRTC-based architecture and is able to provide scalable real-time communication. As opposed to proprietary video conferencing tools which need to be externally redirected or have complicated authentication processes, Jitsi can simply be embedded into web applications. This feature will allow the offered platform to offer a consistent user experience with reliable and low-latency communication.

B. Live Session Creation and Scheduling

Instructors use the platform interface only to create and schedule live classroom sessions. Upon the scheduling of a session by an instructor, the backend creates a unique identification of the room and links it with the course, instructor and the time during which it will be held. The database will contain this information and will be accessible only to authorized users.

The scheduling mechanism will make it contextually connected with the specific courses so that enrolled students can find and attend future sessions with ease without having to resort to external meeting links to do so. It will enhance usability and minimize administrative overhead.

C. Secure Session Access

Role-based authorization regulates access to live classroom sessions. Only course owners can create live sessions, whereas it is only students enrolled in a particular course that are allowed to attend the live session. The security is implemented through the JWT-based authentication, such that only legitimate users can access the session [11].

The frontend also authenticates access to user backend APIs before joining a live session. When the verification is successful, the session information is relayed back to the client, and he/she is allowed to enter the live classroom setting.

D. Embedded Live Classroom Interface

Jitsi external API The frontend will incorporate Jitsi Meet interface into the application and use Jitsi external API. This design enables users to engage in live sessions without switching the platform and maintain learning continuity and enhance user experience. The embedded interface supports features like real-time video, audio, chat and screen sharing.

Jitsi structures the media streams completely, which guarantees the peer-to-peer or server-aided communication efficiency. Consequently, the application back end is lean and is not loaded with media processing assignments [15].

E. Scalability and Performance Considerations

This is because the separation of media handling, and application logic enhances scalability highly. As only session metadata is processed by the backend server and the authorization is done, it can support a high number of simultaneous users with a minimal use of resources. The platform can be used to scale in the large-scale education environment, and this architecture encourages horizontal scaling and cloud deployment.

The live classroom feature ensures a hybrid model of learning that replaces the flexibility of asynchronous learning with the effectiveness of synchronous teaching to fill a significant gap in the current Ed-Tech platforms [2].

V. Implementation Details

The suggested Ed-Tech platform is created and applied on the basis of the current full-stack web development technologies with the emphasis on modularity, scalability, and security. The system is built based on the asynchronous course management and the synchronous live classroom service in the framework of one application.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

A. Technology Stack

The platform is built upon the MERN stack that comprises MongoDB to store data, Express.js to scale the server-side routing, React to build the user interface in the front-end, and Node.js to execute things in the back-end [6]. With this technology stack, it is able to develop fast, perform strong, and maintainability easily .

The front-end application is developed with the help of React and is styled with utility-first CSS frameworks to be responsive to devices. The Express.js is used to implement the backend services, which have RESTful APIs that are used to communicate between the frontend layer and the database layer [8].

B. Authentication and Authorization

Authentication of users is done with the help of the JSON Web Tokens (JWT). When registration or logging in is successful, a JWT that has been signed is issued by the server with the information on user identifier and role. This token is safely locked in the client machine and verified in the further API requests to get the access to the protected resources.

There is role-based access control at the backend to distinguish between the student and instructor credentials. The instructors have the right to design courses and schedule live classes whilst students are allowed to take courses and attend scheduled live classes. The cryptographic keys are industry-standard to ensure that the passwords are securely hashed to avoid unauthorized access [16].

C. API Design and Backend Services

The backend is based on the pattern of API design of REST. Separate endpoints are used in authentication, course management, enrolment processing and scheduling of live classes. The isolation between routes, controllers, and services guarantees a well-organized code and improves its maintainability.

The session of classes is linked to a course and an instructor in every live class. Each time an instructor arranges a meeting, the backend system creates a unique identifier of the live classroom and inserts the metadata of the meeting in the database. The use of authorization checks will make sure that the session details are only accessible to the registered students.

D. Database Schema Implementation

MongoDB is used as the main data storage because it has the quality of schema flexibility and scalability [10]. Authentication credentials, role details and enrollment data are stored in user documents. Course metadata, instructor associations and student lists enrolled in a course are stored in course documents.

The details in live class session documents include the session title, the time it will be held, instructor reference, course involved and meeting identifying numbers. Such a schema is efficient to query and allow the further improvement of the system in terms of attendance countering, session analytics, and recording.

E. Live Classroom Integration

The live classroom feature is realised based on the Jitsi Meet API that enables video conferencing in real-time by using an embeddable interface. The frontend dynamically loads the Jitsi client and connects to the specified meeting room depending on the session identifier that the backend sends.

The media streaming is managed by the infrastructure of Jitsi and ensures both low latency and decreased server side controls. The application server has the sole role of session authorization and metadata management, which enhances scalability and system performance.

F. Deployment Considerations

The backend services are stateless, which means that they can be scaled horizontally by using containerization and load balancing. Sensitive credentials and deployment-specific parameters are configured by environment-based configuration. The modular architecture enables the system to be deployed in cloud platforms with minimum configuration modifications.

VI. Experimental Evaluation

Ed-Tech proposed platform was tested in terms of functional testing and practical usage scenarios to test its effectiveness, performance, and usability. Given that the main input of the given work is system integration and architectural design, the assessment is aimed at the verification of the basic functionalities and monitoring the system behavior in the context of real-life conditions.

A. Evaluation Environment

The application was developed in a controlled development environment of a web-based frontend and a Node.js backend, which was linked to a MongoDB database [7], [10]. The standard web browsers were used to access the frontend and the backend services were implemented in a local server setup. Jitsi meet API was used in classroom live functionality, which allows real time communication via browser based clients.

This configuration is much closer to real world deployment scenarios and enables efficient testing of system behavior even without the need of special hardware or infrastructure [15].

B. Functional Validation

Functional testing was undertaken to ensure that major features of the platform were correct. The workflows of user authentication were tested through the registration and the logging-in of users of various roles such as students and instructors. Role based control access was confirmed because only the instructors were allowable to design courses and come up with live sessions in the classroom whereas the students were limited to taking up courses and attending classes [11], [12].

The functionality of live classroom was tested by scheduling the sessions of particular courses and letting enrolled students participate at the given time. The system was able to create distinct identifiers of a session, to author users accordingly and to integrate a live classroom interface within the application.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

These processes verified the appropriateness of the session scheduling, authorization, and integration processes.

C. Performance Observations

Performance testing was aimed at monitoring system responsiveness when interacting with users in a typical way. The response time of API was consistent with authentication, accessing courses, and revisiting live session. Because of the fact that media streaming is supported by the Jitsi infrastructure, the application backend had only a light load in the form of computational overhead in the live classes [13], [15].

The stateless backend architecture enabled the platform to be used by many users at the same time without any performance being observed to be slowed down. These are observations that the proposed architecture is capable of supporting moderate-large scale deployment when the right deployment settings are in place [15].

D. User Interaction and Engagement Analysis

The user interaction qualitative analysis showed better continuity of learning because of the smooth integration of live classrooms on the platform. There were also no external meeting links and users could easily switch between asynchronous course content and synchronous live sessions.

The live classroom interface that was embedded minimized the workflow fragmentation and improved instructor-student communication. These findings indicate that the hybrid learning environment facilitated by the suggested platform has the potential to enhance engagement of the learners and make the educational process more interactive than asynchronous-based models.

VII. Conclusion and Future Work

The current paper has introduced an improved version of MERN-based educational technology platform that includes a live classroom operation into a classic asynchronous learning system. The proposed platform was developed and built because the current Ed-Tech systems are designed to guide self-paced learning and thus did not accommodate a hybrid one. The system allows instructor-student interaction in a single application system by integrating live classroom real-time interaction (Jitsi Meet API) [13].

The suggested structure will allow uniting scalability, modularity, and security, utilizing the latest web technologies, role-based access control, and authentication based on JWT. Embedding live classes into the platform removes the need to rely on external conferencing systems, decreases the discontinuity of the workflow, and enhances the continuity of the learning process. Functional testing and performance observation show that the platform can successfully support the fundamental educational processes and the system behavior remains stable when using it concurrently [15].

The findings show that the incorporation of synchronous interaction in an asynchronous learning environment can greatly make the learning process more interactive and effective in the overall delivery of online learning. The architectural proposal in this article gives a realistic and repeatable proposal on how to build hybrid digital learning through full-stack web technology.

The further development of the platform will involve adding new features, including recording of live sessions, automated attendance, and analytics-based insights into the behavior of the learners. Another potential new direction is the addition of artificial intelligence methods to individual learning suggestions and feedback evaluation. In addition, the development of the platform into a large-scale cloud setting with extensive user studies will give more information about how the system performs and how it affects education [2].

Acknowledgment

The authors would be pleased to acknowledge the invaluable advice, constant encouragement, and positive feedback that **Dr. Shikha**, the School of Computing Science & Engineering, Galgotias University gave to them during the ongoing research work.

References

[1] M. A. Alvi, A. Misha, A. Srivastava, and K. K. Singh, "StudyNotion: MERN Based Ed-Tech Platform," in *Proc. 2nd Int. Conf. Computational and Characterization Techniques in Engineering & Sciences (IC3TES)*, IEEE, 2024.
[2] B. Means and J. Neisler, "Teaching and Learning in the Time of COVID," *Online Learning*, vol. 25, no. 1, 2021.
[3] F. Martin and M. Parker, "Use of Synchronous Virtual Classrooms," *Computers & Education*, vol. 127, pp. 68–79, 2019.
[4] M. Bond, "Facilitating Student Engagement Through Educational Technology," *Int. J. Educational Technology in Higher Education*, vol. 17, 2020.
[5] K. F. Hew and W. S. Cheung, "Students' and Instructors' Use of MOOCs," *Educational Research Review*, vol. 12, pp. 45–58, 2014.
[6] MongoDB Inc., "The MERN Stack Explained," 2023. [Online]. Available: https://www.mongodb.com/mern-stack
[7] OpenJS Foundation, "Node.js Architecture Overview," 2023. [Online]. Available: https://nodejs.org/en/about
[8] Express.js Foundation, "Express Web Framework," 2023. [Online]. Available: https://expressjs.com
[9] Meta Platforms Inc., "React Documentation," 2023. [Online]. Available: https://react.dev
[10] MongoDB Inc., "MongoDB Atlas Documentation," 2023. [Online]. Available: https://www.mongodb.com/docs
[11] M. Jones, "JSON Web Token (JWT)," RFC 7519, Internet Engineering Task Force (IETF), 2015.
[12] R. Sandhu, "Role-Based Access Control," *IEEE Computer*, vol. 29, no. 2, 2018.
[13] Jitsi, "Jitsi Meet Developer Guide," 2023. [Online]. Available: https://jitsi.github.io/handbook
[14] W. Bao, "COVID-19 and Online Teaching," *Human Behavior and Emerging Technologies*, vol. 2, pp. 113–115, 2020.
[15] Q. Zhang, "Scalable Web Application Architectures," *IEEE Internet Computing*, vol. 21, no. 3, 2017.
[16] E. Fernandez, "Security Patterns for Web Applications," *IEEE Security & Privacy*, vol. 17, no. 2, 2019.