Table of Contents

[Overview 2](#_Toc186406416)

[Introduction to the Project: 2](#_Toc186406417)

[Purpose and Objective of the Application: 2](#_Toc186406418)

[Business Name and Problem Statement: 3](#_Toc186406419)

[Scope of the Project 3](#_Toc186406420)

[Defined functionalities: 3](#_Toc186406421)

[Purpose of Online Presence: 3](#_Toc186406422)

[Services offered: 4](#_Toc186406423)

[Technology Stack 4](#_Toc186406424)

[Frontend Technologies: 4](#_Toc186406425)

[Backend Technologies: 4](#_Toc186406426)

[Database: 4](#_Toc186406427)

[Other Tools: 4](#_Toc186406428)

[Fulfillments of Requirements 5](#_Toc186406429)

[Single Page Application: 5](#_Toc186406430)

[Backend with CRUD operations and Business Logic: 5](#_Toc186406431)

[Multiple Actors: 5](#_Toc186406432)

[User Authentication: 5](#_Toc186406433)

[Frontend Implementation: 5](#_Toc186406434)

[CSS and Styling: 5](#_Toc186406435)

[Database Schema: 6](#_Toc186406436)

[System Architecture 6](#_Toc186406437)

[Overview of the Architecture: 6](#_Toc186406438)

[Interaction between Components: 6](#_Toc186406439)

[Main Functionalities 6](#_Toc186406440)

[User Authentication: 6](#_Toc186406441)

[Key Functional Modules 6](#_Toc186406442)

[Module 1: User Management 6](#_Toc186406443)

[Module 2: Upload and Management of CSV/Excel Files 7](#_Toc186406444)

[Module 3: Creation and Management of HTML Templates for Certificates 7](#_Toc186406445)

[Module 4: Certificates Generation, Download and Deletion 7](#_Toc186406446)

[Module 5: Meeting Scheduling 7](#_Toc186406447)

[Module 6: Sending Emails 7](#_Toc186406448)

[Error Handling and Logging: 7](#_Toc186406449)

[Database Design 8](#_Toc186406450)

[Entity-Relation Diagram: 8](#_Toc186406451)

[Database Relationships: 8](#_Toc186406452)

[RESTful API 9](#_Toc186406453)

[User Management Routes: 9](#_Toc186406454)

[File Management Routes: 10](#_Toc186406455)

[Certificate Template Management Routes 11](#_Toc186406456)

[Certificate Generation Routes: 12](#_Toc186406457)

[GitHub Repository: 13](#_Toc186406458)

[User Interface Design 13](#_Toc186406459)

# Overview

## Introduction to the Project:

CertiMeet is a dynamic MERN stack application designed to streamline the process of certificate generation and meeting management. By integrating interactive client-side views with robust server-side functionality, CertiMeet provides a seamless user experience for individuals and organizations.

## Purpose and Objective of the Application:

The main purpose of CertiMeet is to simplify administrative tasks associated with event management and certificate issuance. The objectives include:

* Providing a user-friendly interface for creating and managing certificates.
* Implementing a secure and efficient meeting scheduling system.
* Enabling multiple actors, including administrators and users, to interact with the system effectively.
* Ensuring data security and accuracy through authentication and business logic implementation.

## Business Name and Problem Statement:

**Business Name:** CertiMeet

**Problem Statement:** Many organizations struggle with the time-consuming and error-prone tasks of managing meetings and issuing certificates. Traditional methods often lead to inefficiencies, data inconsistencies, and lack of user engagement. CertiMeet addresses these challenges by offering a comprehensive platform that automates these processes while maintaining flexibility and scalability.

# Scope of the Project

## Defined functionalities:

CertiMeet offers a range of functionalities to ensure a seamless user experience:

* User Authentication and Authorization: Secure login and role-based access.
* Certificate Generation: Dynamic creation and customization of certificates.
* Meeting Management: Scheduling, organizing, and managing meetings effectively.
* Multi-Actor System: Support for administrators, users, and other relevant roles.
* Real-Time Data Updates: Ensuring synchronization and accuracy across all modules.

## Purpose of Online Presence:

The online presence of CertiMeet aims to:

* Provide a centralized platform for managing certificates and meetings.
* Enhance user accessibility and interaction through a web-based interface.
* Ensure real-time availability of services, reducing manual effort and time.
* Improve organizational efficiency and user satisfaction by offering automated solutions.

## Services offered:

CertiMeet provides the following services:

* **Custom Certificate Design:** Create, modify, and distribute professional certificates.
* **Meeting Scheduling and Management:** Plan and coordinate meetings with minimal effort.
* **User Role Management:** Assign and manage roles for administrators, users, and guests.
* **Data Security and Analytics:** Secure storage and insightful analytics for better decision-making.

# Technology Stack

## Frontend Technologies:

* React.js for building interactive user interfaces.
* Tailwind CSS for styling and responsive design.
* Axios for API integration.

## Backend Technologies:

* Node.js with Express.js for server-side development.
* Mongoose for database schema management and interaction.

## Database:

MongoDB as the primary database for storing user and system data.

## Other Tools:

* Git for version control and collaboration.
* Postman for API testing.
* JWT (JSON Web Tokens) for user authentication.

# Fulfillments of Requirements

## Single Page Application:

CertiMeet is designed as a SPA, ensuring smooth navigation and a dynamic user experience without page reloads.

## Backend with CRUD operations and Business Logic:

The backend supports:

* **Create:** Adding new users, certificates, and meetings.
* **Read:** Viewing details for certificates, meetings, files and HTML templates.
* **Update:** Modifying existing records.
* **Delete:** Removing redundant or outdated records.

## Multiple Actors:

CertiMeet supports multiple user roles:

* **Administrator:** Manages the system and oversees all operations.
* **User:** Schedules meetings and generates certificates.
* **Visitor:** Visits the site, verify the certificate.

## User Authentication:

JWT-based authentication ensures secure access. Role-based authorization controls functionality access per user type.

## Frontend Implementation:

React.js components, props, and states manage the dynamic UI. Axios fetches data from the REST API.

## CSS and Styling:

Tailwind CSS ensures a modern and responsive UI, compatible with various devices.

## Database Schema:

MongoDB with Mongoose defines clear schema specifications for users, certificates, and meetings.

# System Architecture

## Overview of the Architecture:

CertiMeet follows the MERN stack architecture:

* **Frontend:** React.js for UI.
* **Backend:** Node.js with Express.js for API and business logic.
* **Database:** MongoDB for data persistence.
* **Integration:** Components interact via REST APIs.

## Interaction between Components:

* User interactions trigger frontend actions.
* API calls fetch or update data from the backend.
* Real-time updates reflect changes across components.

# Main Functionalities

## User Authentication:

* Secure login/logout.
* Role-based access control.

## Key Functional Modules

### Module 1: User Management

* Profile creation and updates.
* Role assignment.

### Module 2: Upload and Management of CSV/Excel Files

* File upload and parsing.
* Data validation and storage.

### Module 3: Creation and Management of HTML Templates for Certificates

* Dynamic design of certificates.
* Templates storage and retrieval.

### Module 4: Certificates Generation, Download and Deletion

* Generation of certificates based on uploaded files and templates.
* Download certificates in various formats.
* Delete certificates from the database.

### Module 5: Meeting Scheduling

* Create and manage meeting schedules.
* Participant management.

### Module 6: Sending Emails

* Automated email generation and dispatch.
* Tracking email status.

## Error Handling and Logging:

* Comprehensive error messages.
* Logging system to track and debug issues.

# Database Design

## Entity-Relation Diagram:

A diagram of a computer

Description automatically generated

## Database Relationships:

* Users can have multiple certificates, files, templates and meetings.
* One Certificate is based on a file and a template and belongs to a specific user.

# RESTful API

## User Management Routes:

**Register User**

* **Route:** /register
* **Method:** POST
* **Description:** Registers a new user. No avatar upload is performed here.

**Verify OTP**

* **Route:** /verify-otp
* **Method:** POST
* **Description:** Verifies the OTP sent during signup.

**Login User**

* **Route:** /login
* **Method:** POST
* **Description:** Logs in a user.

**Logout User (Protected)**

* **Route:** /logout
* **Method:** POST
* **Middleware:** verifyJWT
* **Description:** Logs out a user. Requires a valid JWT token.

**Change Password (Protected)**

* **Route:** /change-password
* **Method:** PUT
* **Middleware:** verifyJWT
* **Description:** Allows a user to change their password.

**Change Full Name (Protected)**

* **Route:** /change-full-name
* **Method:** PUT
* **Middleware:** verifyJWT
* **Description:** Updates the user's full name.

**Change Username (Protected)**

* **Route:** /change-username
* **Method:** PUT
* **Middleware:** verifyJWT
* **Description:** Updates the user's username.

**Change Email (Protected)**

* **Route:** /change-email
* **Method:** PUT
* **Middleware:** verifyJWT
* **Description:** Updates the user's email address.

**Upload Avatar (Protected)**

* **Route:** /upload-avatar
* **Method:** POST
* **Middleware:** verifyJWT, upload.single("avatar")
* **Description:** Uploads an avatar for the user.

## File Management Routes:

**Upload File (Protected)**

* **Route:** /upload
* **Method:** POST
* **Middleware:** verifyJWT, upload.single("file")
* **Description:** Allows users to upload a file.

**Get Specific File (Protected)**

* **Route:** /getFile/:fileId
* **Method:** GET
* **Middleware:** verifyJWT
* **Description:** Fetches a specific file using its ID.

**Get All Files (Protected)**

* **Route:** /getAllFiles
* **Method:** GET
* **Middleware:** verifyJWT
* **Description:** Fetches all files uploaded by the logged-in user.

**Delete Specific File (Protected)**

* **Route:** /deleteFile/:fileId
* **Method:** DELETE
* **Middleware:** verifyJWT
* **Description:** Deletes a specific file using its ID.

**Delete All Files (Protected)**

* **Route:** /deleteAllFiles
* **Method:** DELETE
* **Middleware:** verifyJWT
* **Description:** Deletes all files uploaded by the logged-in user.

## Certificate Template Management Routes

**Create Template (Protected)**

* **Route:** /create-template
* **Method:** POST
* **Middleware:** verifyJWT
* **Description:** Creates a new certificate template.

**Get Template by ID (Protected)**

* **Route:** /get-template/:id
* **Method:** GET
* **Middleware:** verifyJWT
* **Description:** Fetches a single certificate template by its ID.

**Get All Templates (Protected)**

* **Route:** /get-templates
* **Method:** GET
* **Middleware:** verifyJWT
* **Description:** Fetches all templates for the logged-in user.

**Delete Template by ID (Protected)**

* **Route:** /delete-template/:id
* **Method:** DELETE
* **Middleware:** verifyJWT
* **Description:** Deletes a specific template using its ID.

**Delete All Templates (Protected)**

* **Route:** /delete-all-templates
* **Method:** DELETE
* **Middleware:** verifyJWT
* **Description:** Deletes all templates for the logged-in user.

## Certificate Generation Routes:

**Generate Certificates (Protected)**

* **Route:** /generate
* **Method:** GET
* **Middleware:** verifyJWT
* **Description:** Generates certificates for participants.

**Download Certificates as ZIP (Protected)**

* **Route:** /download/:generatedCertificateId
* **Method:** GET
* **Middleware:** verifyJWT
* **Description:** Downloads all certificates for a specific generation ID as a ZIP file.

**Send Certificates via Email (Protected)**

* **Route:** /send-certificates/:generatedCertificateId
* **Method:** POST
* **Middleware:** verifyJWT
* **Description:** Sends certificates to participants via email.

**Delete Generated Certificates (Protected)**

* **Route:** /delete/:generatedCertificateId
* **Method:** DELETE
* **Middleware:** verifyJWT
* **Description:** Deletes generated certificates for a specific generation ID.

# GitHub Repository:

* Access the source code and documentation on [GitHub](https://github.com/Danyal-Rana/CertiMeet).

# User Interface Design

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a login form

Description automatically generated

A screenshot of a login form

Description automatically generated