System Architecture Design Document

Table of Contents

---

1. Introduction

The Student-Teacher Appointment Booking System aims to provide a platform for efficient scheduling and management of appointments between students and teachers within an educational institution. This document outlines the architecture designed to support the system's functionality and scalability.

---

2. Architectural Overview

The system is designed as a web-based application with separate frontend and backend components to ensure modularity and maintainability. It follows a client-server architecture model, where the frontend interacts with the backend through RESTful APIs for data exchange.

---

3. System Components

Frontend

- **Description**: The frontend is developed using React.js, designed to provide a responsive and intuitive user interface for students, teachers, and administrators.
- **Components**: Includes pages for user authentication, appointment booking, dashboard views for different roles, and forms for data input.
- **Technologies**: React, Tailwind CSS, Flowbite UI framework.

Backend

- **Description**: The backend server manages data persistence, business logic, authentication, and authorization processes.
- **Components**: Node.js with Express.js framework, MongoDB for database storage, Mongoose for ODM.
- **Technologies**: Node.js, Express.js, MongoDB, Mongoose, JWT for authentication, bcrypt for password hashing.

Database

- **Description**: MongoDB is used as the primary database to store user data, appointments, and system configurations.
- **Collections**: Users, Appointments, Roles.
- **Technologies**: MongoDB, Mongoose ODM.

---

4. Data Flow Diagram

Data Flow Diagram

Description: This diagram illustrates the flow of data through the system, depicting how users interact with the frontend, which communicates with the backend APIs to perform CRUD operations on the MongoDB database.

---

5. Deployment Diagram

Deployment Diagram

Description: The deployment diagram outlines the infrastructure setup for hosting the application components. It includes the frontend deployed as a static website and the backend deployed as a Node.js application on a cloud platform like AWS or Heroku.

---

6. Technologies Used

- **Frontend**: React.js, Tailwind CSS, Flowbite.
- **Backend**: Node.js, Express.js, MongoDB, Mongoose.
- **Deployment**: AWS (S3 for frontend hosting, EC2 for backend), Heroku.

---

7. Security Considerations

- **Authentication**: JWT tokens for user authentication and authorization.

- **Data Protection**: Passwords hashed using bcrypt, HTTPS for secure data transmission.
- **Access Control**: Role-based access control (RBAC) implemented to restrict endpoints and features based on user roles.

---

8. Scalability and Performance

- **Horizontal Scaling**: Backend services designed for horizontal scalability using load balancers.
- **Database Sharding**: MongoDB sharding for distributed data storage and improved read/write performance.
- **Caching**: Implementing Redis for caching frequently accessed data.

---

9. Monitoring and Logging

- **Logging**: Centralized logging using tools like ELK Stack (Elasticsearch, Logstash, Kibana) for debugging and performance monitoring.
- **Monitoring**: Application performance monitored using New Relic or similar tools for proactive issue detection and resolution.

---

10. Future Considerations

- **Feature Enhancements**: Integration of real-time chat functionality between students and teachers.
- **Microservices Architecture**: Transition to a microservices architecture for better scalability and maintainability.
- **Mobile Application**: Development of a companion mobile application for Android and iOS platforms.

---

11. Conclusion

This document provides a comprehensive overview of the architecture designed for the Student-Teacher Appointment Booking System. It ensures scalability, security, and performance while outlining future considerations for further enhancing the application.