PROJECT REPORT ON

"Design and implement a School Management System API using Java Spring Boot and MySQL. The API should provide CRUD operations for managing students, parents, and teachers."

Submitted By:

Emanuel Soloman

May, 2024

Design Decisions.

1. Java Spring Boot: Spring Boot is chosen as the framework for building the School Management System API due to its ease of use, rapid development capabilities, and robust ecosystem of libraries and tools. It provides features like dependency injection, auto-configuration, and embedded servers, making it ideal for developing RESTful web services.
2. MySQL Database: MySQL is selected as the database management system for storing and retrieving data related to students, parents, and teachers. MySQL is widely used, scalable, and offers good performance for relational database operations. It also integrates seamlessly with Spring Boot applications.
3. IntelliJ IDEA: IntelliJ IDEA is chosen as the Integrated Development Environment (IDE) for developing the School Management System API. IntelliJ IDEA offers a rich set of features for Java development, including code completion, refactoring tools, and seamless integration with Maven for dependency management. Its intuitive interface and powerful debugging capabilities enhance developer productivity.
4. Postman: Postman is utilized for testing the APIs during development and after deployment. It allows for easy creation of HTTP requests, testing of different endpoints, and validation of responses. Postman's collection feature enables the organization of API requests into folders, making it convenient for testing various endpoints.

Challenges Faced.

1. Data Modelling: Designing the database schema to accurately represent the relationships between students, parents, and teachers posed a challenge. Deciding on the appropriate entity relationships and ensuring data integrity required careful consideration and planning.
2. Error Handling: Implementing robust error handling mechanisms for the API endpoints was challenging. Ensuring that appropriate HTTP status codes and error messages are returned in case of failures or invalid requests required thorough testing and validation.
3. Testing: Testing the API endpoints thoroughly to verify functionality and error handling posed challenges. Identifying edge cases and ensuring that the API behaves as expected under various scenarios required comprehensive testing strategies using tools like Postman.
4. Integration with Spring Boot and MySQL: Integrating Spring Boot with MySQL and configuring the database connection settings was challenging, especially for developers new to these technologies. Understanding the configuration options and troubleshooting connection issues required careful attention to detail.
5. Deployment: Deploying the School Management System API to a production environment posed challenges, particularly in configuring the server environment and ensuring smooth deployment without disruptions to existing services. Coordinating deployment activities and addressing any compatibility issues with the hosting environment required collaboration across teams.

Improvements or Enhancements:

1. Enhanced Error Handling: Implement more detailed and informative error messages, including specific error codes and descriptions, to provide better feedback to API consumers. Additionally, implement global exception handling to centralize error handling logic and improve maintainability.
2. Validation and Data Sanitization: Enhance input validation to ensure that all incoming data is properly validated and sanitized to prevent security vulnerabilities such as SQL injection and cross-site scripting (XSS) attacks. Implement validation annotations and custom validators where necessary.
3. Pagination and Sorting: Implement pagination and sorting functionality for API endpoints that return large datasets, such as the endpoints for retrieving lists of students, parents, and teachers. This will improve performance and usability by allowing clients to retrieve data in smaller, manageable chunks.