**Railway Reservation System**

**EXPERIMENT 7**

**Prepare Software Design Document (SDD) of the Railway Reservation System**

**Software Design Document (SDD)** – A formal document that describes the architecture, design choices, and implementation details of a software system. It serves as a blueprint for developers and stakeholders.

**Software Design Document (SDD) for Railway Reservation System**

**1. Introduction**

**1.1 Purpose**

The Railway Reservation System (RRS) is a web-based application designed to facilitate ticket booking, reservation management, and schedule tracking for railway passengers. This document outlines the system’s architecture, design, and implementation details.

**1.2 Scope**

The system enables passengers to search for trains, book tickets, cancel reservations, and view train schedules. It supports multiple users, including passengers, administrators, and railway staff.

**1.3 Definitions, Acronyms, and Abbreviations**

* **SDD** – Software Design Document: A document that describes the architecture, design choices, and implementation details of a system.
* **RRS** – Railway Reservation System
* **UI** – User Interface: The front-end platform that interacts with users.
* **DBMS** – Database Management System: Software for data storage and retrieval.
* **API** – Application Programming Interface: A set of protocols for building software applications.
* **MVC** – Model-View-Controller: A design pattern for separating business logic from user interfaces.
* **JWT** – JSON Web Token: Used for authentication.
* **SSL/TLS** – Secure Sockets Layer / Transport Layer Security: Ensures secure communication over networks.
* **RESTful API** – Representational State Transfer API: A web service architecture for communication between client and server.

**2. System Overview**

The Railway Reservation System is designed to handle train schedules, user reservations, ticket bookings, and payment processing efficiently. It ensures secure transactions and user data protection.

**3. System Architecture**

**3.1 Architectural Design**

The system follows a three-tier architecture:

* **Presentation Layer** – Web-based UI for user interaction
* **Business Logic Layer** – Implements system functionalities
* **Data Layer** – Stores and manages data in a relational database

**3.2 Technologies Used**

* **Frontend:** React.js, HTML, CSS, JavaScript
* **Backend:** Node.js, Express.js
* **Database:** MySQL
* **Security:** JWT authentication, role-based access control
* **Payment Integration:** PayPal, Stripe

**4. Module Design**

**4.1 User Management**

* User registration, login, and profile management
* Role-based access for passengers and administrators
* Password encryption using bcrypt

**4.2 Train Management**

* Train schedule management by railway staff
* Route and seat availability management
* Train status updates (delays, cancellations)

**4.3 Reservation and Ticketing**

* Ticket booking and cancellation
* Dynamic seat allocation
* E-ticket generation and fare calculation
* Ticket history tracking

**4.4 Payment Processing**

* Integration with third-party payment gateways
* Secure transaction handling
* Refund processing for canceled reservations

**4.5 Reporting and Logs**

* Generation of reservation reports
* System logs for monitoring activities
* Admin dashboard for statistical reports

**5. Data Design**

**5.1 Database Schema**

Key tables include:

* **Users** (UserID, Name, Email, Role, PasswordHash, ContactNumber)
* **Trains** (TrainID, Name, Route, DepartureTime, ArrivalTime, Capacity)
* **Reservations** (ReservationID, UserID, TrainID, SeatNumber, Status, BookingDate)
* **Payments** (PaymentID, UserID, Amount, TransactionID, Status, PaymentDate)
* **Logs** (LogID, UserID, Action, Timestamp)

**6. Interface Design**

**6.1 User Interface**

* **Passenger Interface**: Search trains, book tickets, view bookings, cancel tickets
* **Admin Interface**: Manage trains, approve refunds, view logs, manage users

**6.2 API Design**

RESTful APIs for:

* **User authentication** (register, login, logout)
* **Train search** (available trains based on route and date)
* **Reservation handling** (book, cancel, view reservations)
* **Payment processing** (initiate payment, verify status)

**7. Security Considerations**

* **Authentication:** JWT-based authentication
* **Authorization:** Role-based access control
* **Encryption:** Secure data transmission using SSL/TLS
* **Data Protection:** Hashing passwords and encrypting sensitive data
* **Audit Logs:** Tracking user activities to prevent fraud

**8. Performance Considerations**

* Database indexing for fast query execution
* Load balancing for high-traffic management
* Caching frequently accessed data to optimize response time