SRS Document – car rental website

Table of Contents

1. Introduction

1.1. Purpose

The purpose of the Car Rental System is to deliver an advanced, secure, and efficient web-based platform that simplifies vehicle rental processes for both customers and administrators. Designed to automate critical functionalities, the system streamlines workflows such as car browsing, rental bookings, payment processing, and administrative oversight. It is tailored to address the operational demands of modern car rental businesses while maintaining an intuitive user interface and robust backend architecture.

For customers, the platform enables seamless car browsing with comprehensive filtering options, secure online bookings, profile management, and integrated financial calculators for purchasing vehicles. Customers can also access booking histories, manage payments, and receive notifications for key updates. For administrators, the system provides a sophisticated dashboard to manage inventory, track bookings, oversee user activities, and implement late fees for overdue returns. Additionally, the system ensures that financial transactions are processed securely through an integrated payment gateway, enhancing trust and reliability.

The overarching goal of the Car Rental System is to revolutionize the car rental experience by providing scalability, automation, and precision. By addressing both customer convenience and administrative efficiency, the platform bridges the gap between traditional operations and digital transformation in the rental industry.

1.2. Scope

The Car Rental System represents a robust, multi-faceted digital platform meticulously engineered to address the intricate requirements of both customers and administrators in the car rental industry. By integrating advanced technological capabilities, the system is tailored to enhance user experience, streamline operational processes, and support informed decision-making through its comprehensive suite of features and functionalities.

1.2.1. Customer features:

i. **Registration and Authentication**: Customers are empowered to create accounts and log in securely via encrypted credentials, ensuring robust protection of personal information while facilitating personalized access to system utilities.

ii. **Car Browsing and Filtering**: The platform offers an extensive catalog of vehicles, supported by advanced filtering capabilities that enable users to refine searches by criteria such as price range, vehicle type (SUV, Sedan, etc.), transmission preferences, mileage, and manufacturer brand. This granular level of filtering promotes user convenience and informed selection.

iii. **Car Details and Specifications**: Each vehicle entry encompasses a comprehensive repository of information, including high-resolution images, detailed descriptions, rental rates, availability statuses, technical specifications, and user reviews. This transparency enhances user trust and decision accuracy.

iv. **Car Booking**: Users can seamlessly book vehicles by specifying rental and return dates. The system autonomously computes the total cost based on rental duration and daily pricing, streamlining the booking process.

v. **Favorites Management**: Customers have the capability to curate personalized lists of favorite cars for swift future access and comparative analysis, fostering user-centric engagement.

vi. **Booking History and Management**: Users can access detailed records of their active and past bookings. This feature allows for efficient management of reservations, including updates to payment statuses and necessary modifications.

vii. **Online Payments**: Secure integration with the PayFast payment gateway enables seamless, encrypted financial transactions, ensuring compliance with stringent security standards and fostering user confidence.

viii. **Financial Calculator**: The platform incorporates an advanced financial calculator designed to assist customers in evaluating vehicle purchase options. It provides precise estimates of monthly installment amounts, interest rates, and loan durations, accompanied by an interactive pie chart that visually delineates financial allocations, including principal, interest, and total expenditures.

ix. **Notifications**: The system deploys automated email notifications to provide timely updates regarding booking confirmations, approval statuses, payment reminders, and return deadlines, ensuring customers remain informed throughout their engagement.

x. **Responsive Design**: The platform's interface is optimized for adaptability, ensuring a consistent and high-quality user experience across diverse devices, including desktops, tablets, and smartphones.

1.2.2. Administrative Features:

i. **Car Inventory Management:** Administrators are equipped with sophisticated tools to manage vehicle inventories, including the addition, editing, and removal of listings, along with updates to availability, rental pricing, and technical attributes.

ii. **Booking Management:** The administrative dashboard facilitates comprehensive oversight of booking requests, enabling approval or rejection based on business policies, and providing real-time tracking of active reservations and payment statuses.

iii. **Return and Late Fee Handling:** Administrators can efficiently mark vehicles as returned, compute late fees for overdue rentals, and immediately update the vehicle's availability for subsequent bookings, ensuring optimal asset utilization.

iv. **User Management:** The system enables administrators to monitor and manage user accounts, address customer inquiries, and maintain an audit trail of user activities for quality assurance and issue resolution.

v. **Revenue Analytics:** A robust reporting module provides granular insights into revenue generation, encompassing metrics such as completed bookings, accrued late fees, and outstanding payments. This feature is instrumental for financial auditing and strategic decision-making.

vi. **Notifications Management:** Administrators have the flexibility to customize communication templates for customer notifications, ensuring clarity, professionalism, and alignment with organizational branding.

1.2.3. System Features:

i. **Secure Payment Gateway:** Integration with PayFast ensures that all monetary transactions are encrypted and adhere to PCI DSS compliance standards, safeguarding customer financial data and mitigating risk.
ii. **Interactive Financial Tools:** The system's financial calculators and visual analytics empower both customers and administrators to make data-driven decisions, enhancing financial transparency and trust.
iii. **Scalable Architecture:** The platform is architected to accommodate an expanding user base, growing vehicle inventories, and increasing transactional volumes without degradation in performance or responsiveness.
iv. **Automation:** Core processes, including booking approvals, late fee calculations, and notification dispatch, are automated to minimize administrative workloads, reduce human error, and ensure consistent service delivery.

1.3. Definitions, Acronyms and Abbreviations:

This section provides a comprehensive list of definitions, acronyms, and abbreviations used throughout the document to ensure clarity and precision in understanding the terms relevant to the Car Rental System.

i. **Admin Panel**: A backend interface designed for administrators to manage the system's operations, such as car listings, customer bookings, payments, and user accounts.
ii. **Booking**: A reservation made by a customer for a specific car over a defined rental period. Bookings may include details such as start and end dates, total cost, and payment status.
iii. **CRUD Operations**: An acronym for Create, Read, Update, and Delete, representing the core functions of persistent storage in the system. These operations are used for managing data in the database.
iv. **Customer Portal**: The front-facing interface of the system where users can register, log in, browse cars, book rentals, manage their profiles, and make payments.

v. **EJS (Embedded JavaScript)**: A templating language used to dynamically generate HTML pages for the web application by embedding JavaScript code within HTML structures.

vi. **Email Notification System**: A feature of the system that sends automated emails to users for events such as booking confirmations, payment reminders, or overdue return alerts.

vii. **Late Fee**: An additional charge applied to a booking when a car is returned past its scheduled return date. The fee is calculated based on the number of overdue days and a predefined daily rate.

viii. **Payment Gateway**: A third-party service (e.g., PayFast) integrated into the system to facilitate secure online payment processing for car rentals. It handles the transaction between the customer and the rental service provider.

ix. **Return Status**: A field in the booking system that indicates the condition of a booking. Possible values include "pending," "returned," or "late."

x. **SQL (Structured Query Language)**: A programming language used to manage and query data in the system's database. The database operations in this system are implemented using SQLite.

xi. **SQLite**: A lightweight, serverless database engine used as the primary data storage for the system. It supports structured data storage for cars, bookings, users, and payment records.

xii. **Session Management**: A mechanism used to track logged-in users during their interaction with the system. It ensures that customer data and admin privileges are securely maintained throughout the user session.

xiii. **System Constraints**: Limitations within which the system must operate, such as hardware requirements, software dependencies, or legal considerations.

xiv. **UI/UX (User Interface/User Experience)**: The design and functionality of the user-facing components of the system that ensure a seamless and visually appealing experience for users.

User Authentication: The process of verifying the identity of users before granting access to the system. It typically involves email and password verification and protects user data from unauthorized access.

1.4. References:

This section provides a comprehensive list of references used in the development of the Car Rental System. These references serve as foundational resources for understanding the technologies, tools, and APIs integrated into the system.

i. PayFast Developers (n.d.) *PayFast API Documentation*. Available at: https://developers.payfast.co.za (Accessed: 20 December 2024).

ii. SQLite (n.d.) *SQLite Official Documentation*. Available at: https://sqlite.org/docs.html (Accessed: 20 December 2024).

iii. Bootstrap (n.d.) *Bootstrap CSS Framework Documentation*. Available at: https://getbootstrap.com (Accessed: 20 December 2024).

iv. EJS (n.d.) *EJS Templating Documentation*. Available at: https://ejs.co/ (Accessed: 20 December 2024).

v. Mozilla Developer Network (MDN) (n.d.) *Web Forms – Working with User Data*. Available at: https://developer.mozilla.org/en-US/docs/Learn/Forms (Accessed: 20 December 2024).

vi. Node.js (n.d.) *Node.js Official Documentation*. Available at: https://nodejs.org/en/docs/ (Accessed: 20 December 2024).

vii. Express (n.d.) *Express Framework Documentation*. Available at: https://expressjs.com (Accessed: 20 December 2024).

viii. Chart.js (n.d.) *Chart.js Documentation*. Available at: https://www.chartjs.org/docs/latest/ (Accessed: 20 December 2024).

ix. Stack Overflow (n.d.) *Programming Questions and Solutions*. Available at: https://stackoverflow.com/ (Accessed: 20 December 2024).

x. W3Schools (n.d.) *HTML, CSS, and JavaScript Tutorials*. Available at: https://www.w3schools.com/ (Accessed: 20 December 2024).

xi. Gmail Developers (n.d.) *Using Gmail SMTP for Nodemailer*. Available at: https://developers.google.com/gmail/api (Accessed: 20 December 2024).

xii. Khan Academy (n.d.) *JavaScript Basics*. Available at: https://www.khanacademy.org/computing/computer-programming (Accessed: 20 December 2024).

1.5. Overview:

The Car Rental System is a modular, web-based application built to facilitate seamless car rental operations. The system is divided into two major components: a Customer Portal for end-users and an Admin Panel for administrators, ensuring that the platform serves all stakeholders effectively. The platform integrates features such as user management, car catalog browsing, booking, payment processing, and administrative operations into a single cohesive system.

The system's architecture relies on:

i. **Frontend User Interface**: Built with EJS and styled with Bootstrap to provide a dynamic, responsive, and user-friendly experience for customers and administrators.

ii. **Backend Framework**: Developed using Node.js and Express, ensuring scalability, reliability, and easy integration of additional functionalities.

iii. **Database Layer**: Powered by SQLite, offering lightweight yet robust data storage for users, cars, bookings, payments, and administrative records.

Key system components include:

i. **Customer Portal**:

- Enables users to register, log in, browse cars, book rentals, and make payments.
- Includes features like managing profiles, viewing booking history, and saving cars to a favorites list.

ii. **Admin Panel**:

- A comprehensive dashboard for administrators to manage cars, bookings, and users.
- Allows admins to mark bookings as returned, apply late fees, and make cars available for new rentals.

iii. **Payment Integration**:

- A secure payment gateway (PayFast) facilitates online payments.

- Payment statuses are tracked, and notifications are sent to users for pending or completed payments.

The system's modular design ensures scalability for future enhancements, such as integrating additional payment options, implementing mobile support, or adding advanced search features. By clearly defining customer and administrative workflows, the system optimizes the car rental process, making it accessible, efficient, and easy to manage for all stakeholders.

2. System Overview

2.1. System context:

The Car Rental System operates as a web-based platform designed to manage the end-to-end car rental process for both customers and administrators. It serves as the primary interface for users to interact with the car rental service, enabling seamless bookings, payments, and management of car listings. The system integrates multiple components to handle data processing, user interaction, and administrative operations efficiently.

2.1.1. The system's context involves three primary stakeholders:

i. **Customers**:

- Individuals who register on the platform to browse available cars, make bookings, process payments, and manage their profiles.
- They interact with the system through the Customer Portal, which includes functionalities such as car search, booking history, and favorites management.

ii. **Administrators**:

- Staff responsible for managing car listings, booking approvals, user accounts, and tracking payments.
- They access the system via the Admin Panel, which provides tools for booking management, applying late fees, and marking cars as returned.

iii. **Payment Gateway**:

- An external service (PayFast) integrated into the system to securely process online payments. It communicates with the backend to validate transactions and update payment statuses.

2.1.2. The system architecture consists of:

- **Frontend**: A responsive and interactive user interface developed using EJS templates and styled with Bootstrap for both the Customer Portal and Admin Panel.
- **Backend**: A server-side application built with Node.js and Express, which processes business logic, handles database operations, and communicates with external services like PayFast and email providers.
- **Database**: An SQLite database that stores structured data for users, cars, bookings, payments, and administrative records.

The system ensures secure session handling for users and administrators, maintaining data integrity and providing a reliable platform for car rental operations.

2.2. Major features:

The Car Rental System offers a range of features designed to meet the needs of both customers and administrators. Below are the system's major functionalities:

i. **User Registration and Authentication**:

- Allows customers to register using their personal details and log in securely.
- Includes functionality for password recovery and profile updates.

ii. **Dynamic Car Listings**:

- Displays a catalog of available cars with detailed specifications, including brand, model, transmission type, fuel type, price per day, and more.
- Supports search and filter options for users to find cars that match their preferences.

iii. **Booking Management**:

- Enables customers to book cars for specific rental periods, calculate costs based on rental days, and track booking statuses.
- Allows administrators to approve or decline bookings and mark cars as returned.

iv. **Payment Integration**:

- Provides a secure payment gateway (PayFast) for processing booking payments.
- Supports payment tracking and updates booking statuses automatically after successful payments.

v. **Email Notifications**:

- Sends automated email alerts to customers for booking confirmations, payment reminders, and overdue return notices.
- Keeps users informed of any status changes related to their bookings.

vi. **Favorites Management**:

- Allows customers to save cars to a favorites list for quick access and future bookings.

vii. **Admin Panel**:

- A comprehensive dashboard for administrators to manage car listings, monitor bookings, apply late fees for overdue returns, and track returned statuses.
- Includes tools to update car availability and process user queries.

viii. **Late Fee Calculation**:

- Automatically calculates late fees for cars returned after the scheduled date.
- Late fees are based on the number of days overdue and a predefined rate.

ix. **Profile Management**:

- Customers can update their personal information, including email addresses and passwords.

- Administrators can view and edit user information for account management purposes.

x. **System Scalability and Security**:

- Designed to handle increased user traffic and a growing number of bookings without performance degradation.
- Implements secure session management and database access controls to protect user data.

3. Functional requirements

The functional requirements define the specific behaviors and functionalities of the **Car Rental System**. These requirements outline the expected interactions between the users (both customers and administrators) and the system. Each requirement is critical to ensuring the system's usability, operational efficiency, and overall effectiveness.

3.1. User Registration and Login:

The system must provide a secure and intuitive mechanism for users to create accounts and access the platform. This includes:

- **Registration**:
  - Users must be able to register by providing their first name, last name, email address, password, phone number, physical address, and age.
  - The system must validate all input fields, ensuring they meet predefined criteria (e.g., password strength, unique email).
  - Passwords must be securely hashed using modern encryption techniques (e.g., bcrypt) to ensure data protection.
- **Login**:

- Registered users must be able to log in using their email and password.
- The system must implement session-based authentication to maintain user login states while preventing unauthorized access.

- **Forgotten Password Recovery**:
  - Users must have the ability to recover their passwords by entering their registered email address.
  - The system must send a secure password reset link to the user's email.

- **Access Restrictions**:
  - Non-authenticated users must be restricted from accessing protected pages, such as booking history or profile management.
  - Administrators must have separate login credentials to access the admin panel.

3.2. Car Management:

Administrators must be able to manage the inventory of available cars, ensuring the car catalog is always up to date. The car management functionality includes:

- **Adding Cars**:
  - Admins must be able to add new cars to the system by providing details such as brand, model, type (e.g., Sedan, SUV), transmission, fuel type, seating capacity, mileage, year, price per day, and car description.
  - The system must allow uploading multiple images for each car.

- **Editing Cars**:
  - Admins must be able to edit car details, including updating specifications or changing the availability status.
  - The system must validate all input fields to ensure consistency in the data.

- **Deleting Cars**:

    - Admins must be able to delete cars that are no longer available for rental.
    - The system must check for active bookings before allowing deletion.
- **Viewing Cars**:
    - The car catalog must be displayed to admins with sortable and filterable options for quick management.

3.3. Booking Management:

The system must allow customers to book cars and enable administrators to manage bookings efficiently. Key functionalities include:

- **Car Booking**:
    - Customers must be able to book cars by selecting a rental start and return date.
    - The system must calculate the total cost based on the rental duration and the car's price per day.
    - Overlapping bookings for the same car must be prevented.
- **Booking Approval and Status Updates**:
    - Admins must have the ability to approve or decline pending bookings.
    - The system must update the car's availability status to "unavailable" when a booking is approved.
- **Late Fee Calculation**:
    - If a car is returned after the scheduled return date, the system must calculate a late fee based on the number of days late and the daily rate.
    - Admins must be able to mark bookings as "returned" or "late" and apply additional charges if necessary.
- **Booking History**:

- Customers must have access to a comprehensive list of their previous and current bookings, including details like car information, rental dates, total cost, and booking status.

3.4. Payment Processing:

The system must provide secure and efficient payment processing for customer bookings. Key requirements include:

- **Payment Gateway Integration**:
  - Payments must be processed using the PayFast gateway, ensuring secure transactions.
  - Customers must be redirected to the PayFast payment page with pre-filled booking details (e.g., booking ID, amount).
- **Payment Validation**:
  - The system must validate payments through the PayFast ITN (Instant Transaction Notification) to confirm successful transactions.
- **Payment Status Updates**:
  - Once payment is confirmed, the system must update the booking status to "paid."
  - Customers must receive email notifications confirming successful payments.

3.5. Favorites Management:

The system must allow customers to manage a personalized list of favorite cars for future reference. Key functionalities include:

- **Adding Favorites**:
  - Customers must be able to add cars to their favorites list from the car catalog.
- **Viewing Favorites**:

- The system must display a dedicated favorites page showing all cars marked as favorites by the customer.

- **Removing Favorites**:
  - Customers must be able to remove cars from their favorites list.

3.6. Admin Panel:

The admin panel is the central dashboard for managing the system. It must include the following features:

i. **Booking Management**:
   - Admins must have access to a comprehensive list of all bookings, with filters for status (e.g., pending, accepted, returned, late).
   - They must be able to approve or decline bookings and apply late fees if applicable.

ii. **User Management**:
   - Admins must be able to view, edit, or delete customer accounts.
   - The admin panel must display detailed user profiles, including contact information and booking history.

iii. **Car Inventory Management**:
   - Admins must have tools to manage car listings, including adding, editing, and deleting cars.

iv. **Analytics and Reports**:
   - The admin panel should include visual data on bookings, payments, and car availability for operational insights.

3.7. Notifications and Email Services

The system must implement automated notifications to keep users informed about their bookings and payments. Key requirements include:

i. **Booking Notifications**:
   - Customers must receive emails for booking confirmations, approvals, and declines.

ii. **Payment Notifications**:
   - Customers must be notified upon successful payment, including a detailed receipt.

iii. **Late Fee Notifications**:
   - If a booking incurs a late fee, the system must send an email specifying the charges and instructions for payment.

iv. **System Alerts**:
   - Admins must receive notifications about pending bookings and other important system events.

4. Non-Functional Requirements

The Car Rental System must adhere to a set of non-functional requirements to ensure the application performs reliably, securely, and efficiently. These requirements establish the system's operational qualities, focusing on its performance, security, usability, and availability.

4.1. Performance Requirements:

The system must exhibit optimal performance under various workloads to ensure a smooth user experience. The performance requirements are as follows:

i. **Response Time**:

- The system must respond to user actions within 2 seconds for all standard operations, such as loading car details, managing bookings, or updating user profiles.
- Payment gateway redirection and processing should not exceed 5 seconds, excluding external delays.

ii. **Scalability**:

- The system must support up to 500 concurrent users without degradation in performance.
- The database must handle at least 10,000 records for cars, users, and bookings with efficient query execution.

iii. **Data Processing**:

- The system should execute complex queries (e.g., filtering available cars or calculating late fees) within 1 second under normal conditions.

iv. **Stress Handling**:

- The application must gracefully handle sudden spikes in traffic, such as during promotional events, by ensuring at least 80% system uptime during peak usage.

4.2. Security Requirements:

Given the sensitive nature of user data and financial transactions, the system must implement stringent security measures to prevent unauthorized access and data breaches:

i. **Authentication and Authorization**:

- User access must be restricted based on roles (e.g., customers vs. administrators).
- Passwords must be encrypted using industry-standard algorithms such as **bcrypt** with a salt.

ii. **Data Protection**:

- All user data, including personal information and payment details, must be encrypted both in transit (using **TLS/SSL**) and at rest.
- Sensitive fields such as passwords and payment tokens must not be stored in plaintext.

iii. **Session Management**:

- User sessions must include a timeout period of 30 minutes of inactivity.
- Secure cookies with the HttpOnly and Secure flags must be implemented to prevent session hijacking.

iv. **Payment Security**:

- Payment transactions must adhere to PCI DSS (Payment Card Industry Data Security Standards) guidelines.
- ITN (Instant Transaction Notifications) from PayFast must be validated to ensure transaction integrity.

v. **Vulnerability Management**:

- Regular vulnerability assessments must be conducted to identify and mitigate security risks.
- The system must protect against common web vulnerabilities, such as SQL injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF).

4.3. Usability Requirements:

The system must provide an intuitive and accessible user experience for both customers and administrators, ensuring high usability standards:

i. **Intuitive User Interface**:

- The customer portal and admin panel must follow a responsive design to ensure compatibility with desktop, tablet, and mobile devices.

- Navigation must be clear, with all features accessible within 3 clicks from the homepage or dashboard.

ii. **Error Handling and Feedback**:

- Users must receive clear error messages for invalid inputs, failed bookings, or payment issues, with actionable instructions to resolve the problem.
- Success messages and visual cues (e.g., checkmarks) must be provided to confirm completed actions.

iii. **Accessibility**:

- The system must comply with WCAG 2.1 Level AA accessibility guidelines to support users with disabilities.
- Keyboard navigation, screen reader compatibility, and high-contrast themes must be implemented.

iv. **Consistency and Aesthetics**:

- The system must maintain a consistent color scheme, typography, and layout across all pages.
- Buttons, forms, and tables must adhere to design standards for alignment and spacing.

4.4. Availability and Reliability:

The system must be dependable and available for use with minimal downtime, ensuring users can access its features at all times:

i. **Uptime**:

- The system must achieve a 99.5% uptime annually, allowing for only minimal scheduled maintenance.
- Downtime should be scheduled outside peak usage hours and communicated to users in advance.

ii. **Fault Tolerance**:

- In the event of a hardware or software failure, the system must recover within 5 minutes to minimize disruption.

iii. **Data Backup and Recovery**:

- Daily backups of all critical data, including user accounts, bookings, and transaction records, must be stored securely.
- In the event of data loss, the system must support recovery from the most recent backup within **1 hour**.

iv. **Concurrency**:

- The system must handle 100 active sessions simultaneously with no more than a 5% increase in response time.

v. **Monitoring and Alerts**:

- Real-time monitoring tools must be integrated to track system health, including CPU usage, memory utilization, and network activity.
- Alerts must be sent to administrators when performance thresholds or downtime limits are exceeded.

5. System Design and Architecture

This section outlines the design and structure of the Car Rental System, detailing the system's overall architecture and database schema. The architecture describes the interaction between the components of the system, while the database schema defines how data is organized and stored.

5.1. System Architecture

The Car Rental System follows a three-tier architecture model to ensure scalability, modularity, and ease of maintenance. The system is divided into three main layers:

5.1.1. Presentation Layer (Frontend):

- **Purpose**: This layer serves as the user interface for customers and administrators.
- **Technologies**: Developed using EJS (Embedded JavaScript) for server-side rendering, with Bootstrap for responsive design and styling.
- **Features**:
  - Dynamic rendering of pages like car catalogs, booking history, and admin dashboards.
  - Input validation on the client-side for forms (e.g., registration, booking, and payment).

5.1.2. Application Layer (Backend):

- **Purpose**: Handles all business logic, user authentication, and communication between the frontend and the database.
- **Technologies**: Built using **Node.js** and **Express.js**.
- **Responsibilities**:
  - Managing user sessions and access control.
  - Handling CRUD operations for cars, bookings, and user data.
  - Processing payments through the PayFast API.
  - Sending email notifications using Nodemailer.

5.1.3. Data Layer (Database):

- **Purpose**: Stores and retrieves data related to users, cars, bookings, payments, and system logs.
- **Technologies**: **SQLite** is used as the database engine for lightweight and efficient storage.
- **Responsibilities**:
  - Storing structured data in normalized tables.
  - Supporting transactional integrity for operations like bookings and payments.

Interaction Flow

The interaction between the layers is as follows:

1. Users interact with the system via the Presentation Layer, performing actions such as browsing cars or booking rentals.
2. Requests are sent to the Application Layer, which processes the business logic and interacts with the Data Layer to retrieve or modify data.
3. Responses are sent back to the Presentation Layer, where the results are rendered dynamically for the user.

Recommendation for Diagrams:

To provide a clear visualization, the following diagrams are recommended:

1. **System Architecture Diagram**: To illustrate the interaction between the frontend, backend, database, and external APIs (e.g., PayFast, SMTP for emails).
2. **Component Diagram**: To detail the components within each layer, such as user authentication, payment processing, and booking management.

5.2. Database Schema

The database schema defines the organization and relationships between the tables used in the system. The tables are normalized to reduce redundancy and ensure efficient data management.

Key Tables:

Users Table

Purpose: Stores user details.

Column	Data Type	Description
`id`	Primary Key, Auto-increment	Unique identifier for each user.
`name`	Text	First name of the user.
`surname`	Text	Last name of the user.
`email`	Text, Unique	User's email address.
`password`	Text, Encrypted	Encrypted user password for secure authentication.
`phone`	Text	Contact number of the user.
`address`	Text	Residential address of the user.
`role`	Text	Role of the user (e.g., 'customer', 'admin').
`created_at`	Timestamp	Date and time the account was created.

Cars Table

Purpose: Stores details about the cars available for rental.

Column	Data Type	Description
`id`	Primary Key, Auto-increment	Unique identifier for each car.
`brand`	Text	Brand of the car (e.g., Mercedes, BMW).
`model`	Text	Model name of the car.
`type`	Text	Type of the car (e.g., 'SUV', 'Sedan').
`transmission`	Text	Transmission type (e.g., 'Manual', 'Automatic').
`fuel_type`	Text	Type of fuel used (e.g., 'Petrol', 'Diesel').
`price_per_day`	Real	Rental price per day for the car.
`availability`	Text	Availability status of the car ('available', 'unavailable').

Bookings Table

Purpose: Records all car rental bookings.

Column	Data Type	Description
`id`	Primary Key, Auto-increment	Unique identifier for each booking.
`car_id`	Foreign Key	References the `id` column in the Cars table.
`customer_id`	Foreign Key	References the `id` column in the Users table.
`rental_date`	Date	Date when the rental begins.
`return_date`	Date	Date when the rental ends.
`total_cost`	Real	Total cost of the rental calculated based on duration.
`status`	Text	Booking status (e.g., 'pending', 'accepted', 'returned').
`late_fee`	Real, Nullable	Late fee incurred if the car is returned late.

Favorites Table

Purpose: Tracks customers' favorite cars.

Column	Data Type	Description
`id`	Primary Key, Auto-increment	Unique identifier for each favorite record.
`user_id`	Foreign Key	References the `id` column in the Users table.
`car_id`	Foreign Key	References the `id` column in the Cars table.

6. External Interface Requirements

This section outlines the interfaces the system interacts with, including user interfaces, API endpoints, and external payment gateway integration.

6.1. User Interfaces

The **Car Rental System** provides an intuitive and user-friendly interface for customers and administrators. The interfaces include:

1. **Customer Portal**:
    - **Homepage**: Displays the catalog of cars with options to filter by brand, type, availability, and price range.
    - **Profile Page**: Allows users to view and update their personal details and manage their passwords.
    - **Favorites Page**: Lists cars marked as favorites by the user.
    - **Booking Management**: Displays booking history, statuses, and payment options.
    - **Payment Page**: A secure page for users to review and proceed with payment.
2. **Admin Panel**:
    - **Dashboard**: Summarizes key metrics such as total bookings, payments, and car availability.
    - **Car Management**: Allows admins to add, update, or delete car listings.
    - **Booking Management**: Enables administrators to approve, decline, or mark bookings as returned, along with the ability to apply late fees.
    - **User Management**: Allows admins to view and edit user details.

6.2. API Endpoints

The system provides RESTful API endpoints for various operations. Below is a list of key endpoints:

Endpoint	**Method**	**Purpose**
/auth/register	POST	Registers a new user.

Endpoint	Method	Purpose
/auth/login	POST	Authenticates a user and initiates a session.
/cars	GET	Retrieves a list of available cars.
/book/:carId	POST	Creates a new booking for the specified car.
/user/bookings	GET	Retrieves the booking history for a logged-in user.
/admin/bookings	GET	Fetches all bookings for admin review.
/admin/bookings/approve/:id	POST	Approves a specific booking.
/admin/bookings/return/:id	POST	Marks a booking as returned and calculates late fees if needed.
/payment/initiate	POST	Initiates a payment request via the PayFast gateway.
/payment/success	GET	Handles the payment success callback.
/payment/cancel	GET	Handles the payment cancellation callback.

These endpoints are built using **Express.js** and follow standard REST principles.

7. System constraints

7.1. Hardware Constraints

1. The system requires a server with the following minimum specifications:
    - **Processor**: Dual-Core 2.0 GHz or higher.
    - **RAM**: 4 GB (minimum), 8 GB (recommended).
    - **Storage**: 50 GB for application data and backups.

- **Network**: Reliable internet connection with at least 10 Mbps bandwidth.

2. For scalability, the system should support cloud deployment on platforms such as AWS, Azure, or Google Cloud.

7.2. Software Constraints

1. **Development Environment**:
   - Node.js version 14 or higher.
   - SQLite version 3.0 or higher.
   - Compatible with modern web browsers such as Chrome, Firefox, Edge, and Safari.
2. **Deployment Environment**:
   - Operating System: Linux (preferred), Windows Server (supported).
   - Web Server: Nginx or Apache for reverse proxying.
3. **Dependencies**:
   - Third-party libraries like **Nodemailer**, **PayFast API**, and **Bootstrap** must be regularly updated to avoid security vulnerabilities.

8. Assumptions and Dependencies

1. The users will have access to a reliable internet connection for seamless operation of the platform.
2. The payment gateway (PayFast) will be available and functional without significant downtime.
3. Users will provide valid and accurate personal details during registration and bookings.
4. Administrators will periodically monitor and update car availability to ensure accurate data.

5. The system depends on the proper functioning of third-party libraries and frameworks such as Node.js, Express.js, SQLite, and PayFast.