Title:

Database Management System for Car Rental

Name: Samanvi P, SRN: PES2UG22CS493

Name: Sirivanthe CG, SRN: PES2UG22CS558

Description:

This project focuses on developing a comprehensive database management system (DBMS) tailored for car rental website. The system integrates functionalities such as customer management, vehicle inventory tracking, driver assignments, booking transactions, payment processing, and real-time reporting. It ensures secure, scalable, and efficient data management.

Requirements

User Interfaces

- Customer UI: A web interface for customers to browse, book vehicles, and manage their bookings.
- Admin UI: A dashboard for administrators to manage the system, perform CRUD operations, and generate reports.

Hardware Interfaces

- Servers: The system requires dedicated servers for hosting the database and backend services.
- Client Devices: Website used by customers and administrators to interact with the system.

Software Interfaces

- Database: A SQL-based relational database management system (RDBMS) for data storage and retrieval.

Communication Interfaces

- Network Protocols: SSL/TLS for secure data transmission between client devices and the server.
- RESTful API: To enable communication between the system and external services, such as payment gateways.

System Features

- **Customer Management**

Description:

The system allows customers to create and manage their profiles, including booking history and payment methods.

Functional Requirements:

- The system shall allow customers to register with an email and password.
- The system shall enable customers to update their personal information.
- The system shall maintain a history of bookings made by each customer.

- **Vehicle Management**

Description:

The system tracks vehicle availability, maintenance schedules, and rental history.

Functional Requirements:

- The system shall allow administrators to add, update, or remove vehicle records.
- The system shall track the status of each vehicle (available, rented, in service).
- The system shall maintain a maintenance schedule for each vehicle.

- **Booking Management**

Description:

The system facilitates the booking, modification, and cancellation of car rentals.

Functional Requirements:

- The system shall allow customers to book vehicles for rental services.
- The system shall allow customers to modify or cancel their bookings.
- The system shall update vehicle availability in real-time.

Payment Processing

Description:

The system handles payments securely, supporting multiple methods and maintaining transaction history.

Requirements:

- The system shall process payments using credit cards, debit cards, and other electronic payment methods.
- The system shall maintain a history of all transactions made by each customer.
- The system shall support refunds and partial payments.

Performance Requirements

- The system shall handle up to 10,000 concurrent users with a response time of less than 2 seconds.
- The system shall process transactions within 5 seconds, excluding network delays.

Security Requirements

- The system shall encrypt all sensitive data using industry-standard encryption algorithms.
- The system shall require multi-factor authentication for administrator access.

Usability Requirements

- The system shall provide an intuitive and user-friendly interface for customers and administrators.
- The system shall support multiple languages to accommodate a diverse user base.

Reliability Requirements

- The system shall maintain an uptime of 99.9%, ensuring high availability for users.

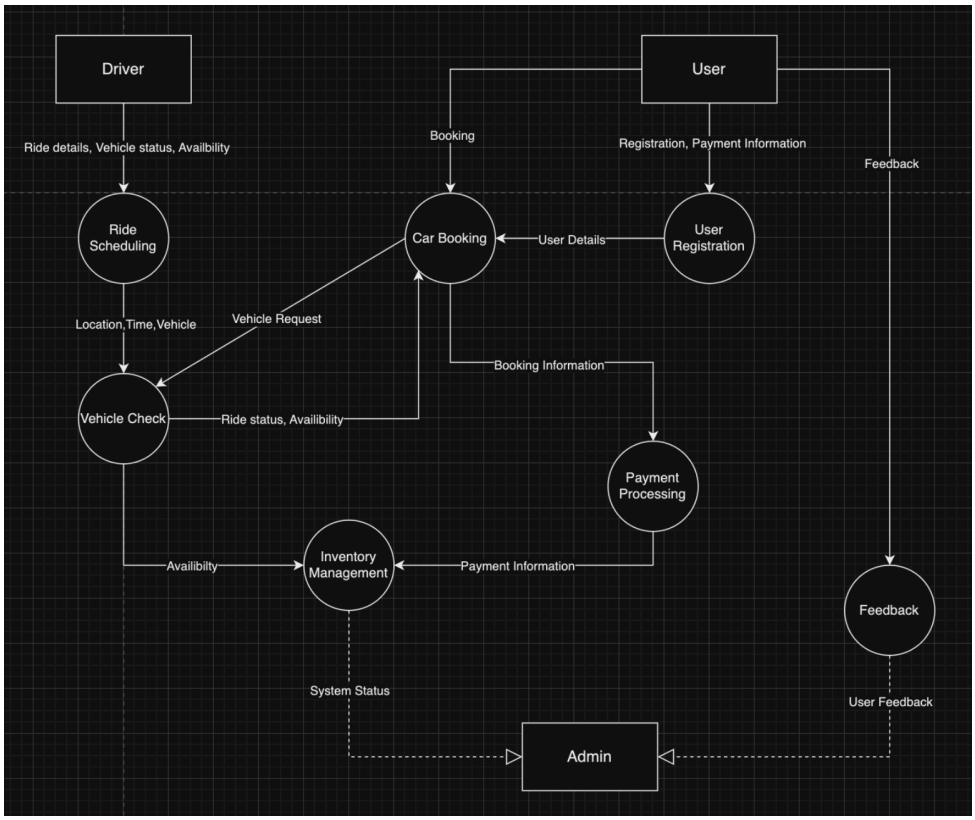
Regulatory Requirements

- The system shall comply with local and international data protection laws, such as GDPR.

Software, Tools, and Languages:

- **Frontend:** React.js
- **Backend:** Node.js
- **Database:** MySQL/PostgreSQL
- **Other Tools:** PHP

ER Diagram:



DDL COMMANDS:

```sql
CREATE TABLE `admin` (
  `id` int(11) NOT NULL,
  `UserName` varchar(100) NOT NULL,
  `Password` varchar(100) NOT NULL,
  `updationDate` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00' ON UPDATE current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```sql
CREATE TABLE `tblbooking` (
  `id` int(11) NOT NULL,
  `userEmail` varchar(100) DEFAULT NULL,
  `VehicleId` int(11) DEFAULT NULL,
  `FromDate` varchar(20) DEFAULT NULL,
  `ToDate` varchar(20) DEFAULT NULL,
  `message` varchar(255) DEFAULT NULL,
  `Status` int(11) DEFAULT NULL,
  `PostingDate` timestamp NOT NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


INSERT INTO `tblbooking` (`id`, `userEmail`, `VehicleId`, `FromDate`, `ToDate`, `message`, `Status`, `PostingDate`) VALUES
(1, 'test@gmail.com', 2, '22/06/2017', '25/06/2017', 'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
(2, 'test@gmail.com', 3, '30/06/2017', '02/07/2017', 'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
(3, 'test@gmail.com', 4, '02/07/2017', '07/07/2017', 'Lorem ipsumLorem ipsumLorem ipsumLorem ipsumLorem ipsumLorem ipsumLorem ipsumLorem ipsumLore
```

```sql
CREATE TABLE `tblbrands` (
  `id` int(11) NOT NULL,
  `BrandName` varchar(120) NOT NULL,
  `CreationDate` timestamp NULL DEFAULT current_timestamp(),
  `UpdationDate` timestamp NULL DEFAULT NULL ON UPDATE current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO `tblbrands` (`id`, `BrandName`, `CreationDate`, `UpdationDate`) VALUES
(1, 'Maruti', '2017-06-18 16:24:34', '2017-06-19 06:42:23'),
(2, 'BMW', '2017-06-18 16:24:50', NULL),
(3, 'Audi', '2017-06-18 16:25:03', NULL),
(4, 'Nissan', '2017-06-18 16:25:13', NULL),
(5, 'Toyota', '2017-06-18 16:25:24', NULL),
(6, 'Volkswagen ', '2017-06-19 06:22:13', NULL),
(7, 'KIA', '2020-10-30 20:42:03', NULL),
(8, 'Skoda', '2020-10-30 20:42:59', NULL),
(9, 'Volvo', '2020-10-30 20:43:25', NULL),
(10, 'Range Rover', '2020-10-31 11:49:39', NULL);
```

```sql
CREATE TABLE `tblcontactusinfo` (
  `id` int(11) NOT NULL,
  `Address` tinytext DEFAULT NULL,
  `EmailId` varchar(255) DEFAULT NULL,
  `ContactNo` char(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```sql
CREATE TABLE `tblcontactusquery` (
  `id` int(11) NOT NULL,
  `name` varchar(100) DEFAULT NULL,
  `EmailId` varchar(120) DEFAULT NULL,
  `ContactNumber` char(11) DEFAULT NULL,
  `Message` longtext DEFAULT NULL,
  `PostingDate` timestamp NOT NULL DEFAULT current_timestamp(),
  `status` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```sql
CREATE TABLE `tblsubscribers` (
  `id` int(11) NOT NULL,
  `SubscriberEmail` varchar(120) DEFAULT NULL,
  `PostingDate` timestamp NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```sql
CREATE TABLE `tblusers` (
  `id` int(11) NOT NULL,
  `FullName` varchar(120) DEFAULT NULL,
  `EmailId` varchar(100) DEFAULT NULL,
  `Password` varchar(100) DEFAULT NULL,
  `ContactNo` char(11) DEFAULT NULL,
  `dob` varchar(100) DEFAULT NULL,
  `Address` varchar(255) DEFAULT NULL,
  `City` varchar(100) DEFAULT NULL,
  `Country` varchar(100) DEFAULT NULL,
  `RegDate` timestamp NULL DEFAULT current_timestamp(),
  `UpdationDate` timestamp NULL DEFAULT NULL ON UPDATE current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```sql
CREATE TABLE tblpages (
    id INT AUTO_INCREMENT PRIMARY KEY,
    type VARCHAR(255) NOT NULL,
    detail TEXT,
    PageName VARCHAR(255)
);
```

```sql
CREATE TABLE `tblvehicles` (
  `id` int(11) NOT NULL,
  `VehiclesTitle` varchar(150) DEFAULT NULL,
  `VehiclesBrand` int(11) DEFAULT NULL,
  `VehiclesOverview` longtext DEFAULT NULL,
  `PricePerDay` int(11) DEFAULT NULL,
  `FuelType` varchar(100) DEFAULT NULL,
  `ModelYear` int(6) DEFAULT NULL,
  `SeatingCapacity` int(11) DEFAULT NULL,
  `Vimage1` varchar(120) DEFAULT NULL,
  `Vimage2` varchar(120) DEFAULT NULL,
  `Vimage3` varchar(120) DEFAULT NULL,
  `Vimage4` varchar(120) DEFAULT NULL,
  `Vimage5` varchar(120) DEFAULT NULL,
  `AirConditioner` int(11) DEFAULT NULL,
  `PowerDoorLocks` int(11) DEFAULT NULL,
  `AntiLockBrakingSystem` int(11) DEFAULT NULL,
  `BrakeAssist` int(11) DEFAULT NULL,
  `PowerSteering` int(11) DEFAULT NULL,
  `DriverAirbag` int(11) DEFAULT NULL,
  `PassengerAirbag` int(11) DEFAULT NULL,
  `PowerWindows` int(11) DEFAULT NULL,
  `CDPlayer` int(11) DEFAULT NULL,
  `CentralLocking` int(11) DEFAULT NULL,
  `CrashSensor` int(11) DEFAULT NULL,
  `LeatherSeats` int(11) DEFAULT NULL,
  `RegDate` timestamp NOT NULL DEFAULT current_timestamp(),
  `UpdationDate` timestamp NULL DEFAULT NULL ON UPDATE current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
ALTER TABLE `admin`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `tblbooking`
--
ALTER TABLE `tblbooking`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `tblbrands`
--
ALTER TABLE `tblbrands`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `tblcontactusinfo`
--
ALTER TABLE `tblcontactusinfo`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `tblcontactusquery`
--
ALTER TABLE `tblcontactusquery`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `tblsubscribers`
--
```

```sql
ALTER TABLE `tblvehicles`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `admin`
--
ALTER TABLE `admin`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;

--
-- AUTO_INCREMENT for table `tblbooking`
--
ALTER TABLE `tblbooking`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;

--
-- AUTO_INCREMENT for table `tblbrands`
--
ALTER TABLE `tblbrands`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;

--
-- AUTO_INCREMENT for table `tblcontactusinfo`
--
ALTER TABLE `tblcontactusinfo`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;
```

Features

1. Customer Management
    - Customers can register, log in, and manage their accounts.
    - View and manage booking history.
2. Vehicle Management
    - Administrators can add, update, and remove vehicles.
    - Real-time tracking of vehicle availability and maintenance schedules.
3. Booking System
    - Customers can search and book available vehicles.
    - Modify or cancel bookings.
    - Real-time updates on vehicle availability.
4. Reporting and Analytics
    - Generate reports on vehicle usage and bookings.
    - Export reports in multiple formats (PDF, CSV).
5. Administrative Dashboard
    - Centralized platform for managing customers, vehicles, and bookings.
    - Role-based access for secure admin operations.
6. Real-Time Updates

- Dynamic updates to vehicle availability after bookings.
- Notifications for maintenance schedules.

7. User-Friendly Interface

- Responsive design for compatibility with mobile, tablet, and desktop.
- Intuitive and easy-to-navigate interface.

Triggers and Procedures:

```
DELIMITER $$

CREATE TRIGGER before_vehicle_price_update
BEFORE UPDATE ON tblvehicles
FOR EACH ROW
BEGIN
    IF OLD.PricePerDay != NEW.PricePerDay THEN
        INSERT INTO VehiclePriceAudit (VehicleId, OldPrice, NewPrice)
        VALUES (OLD.id, OLD.PricePerDay, NEW.PricePerDay);
    END IF;
END $$

DELIMITER ;
```

```sql
--- Trigger for tblbooking
--Automatically update the Status of a booking when a new record is inserted.
---
DELIMITER $$

CREATE TRIGGER confirm_all_bookings
AFTER INSERT ON tblbooking
FOR EACH ROW
BEGIN
    -- Automatically set Status to '1' for the new booking
    UPDATE tblbooking
    SET Status = 1
    WHERE id = NEW.id;
END $$

DELIMITER ;
```

```sql
----------PROCEDURES----------

----------Procedure to Add a New Vehicle: Insert a new vehicle and automatically log the operation.----------
DELIMITER $$

CREATE PROCEDURE AddNewVehicle (
    IN VehicleTitle VARCHAR(255),
    IN Brand INT,
    IN Overview TEXT,
    IN Price DECIMAL(10, 2),
    IN Fuel VARCHAR(50),
    IN ModelYear INT,
    IN Seats INT
)
BEGIN
    INSERT INTO tblvehicles (
        VehiclesTitle, VehiclesBrand, VehiclesOverview, PricePerDay, FuelType, ModelYear, SeatingCapacity
    ) VALUES (
        VehicleTitle, Brand, Overview, Price, Fuel, ModelYear, Seats
    );

    -- Log the addition
    INSERT INTO VehiclePriceAudit (VehicleId, OldPrice, NewPrice)
    VALUES (LAST_INSERT_ID(), NULL, Price);
END $$

DELIMITER ;
```

```
---------- Procedure to Fetch All Bookings for a User: Retrieve booking details for a given user email.----------

DELIMITER $$

CREATE PROCEDURE GetBookingsByUser (
    IN UserEmail VARCHAR(100)
)
BEGIN
    SELECT tblbooking.id, tblvehicles.VehiclesTitle, tblbooking.FromDate, tblbooking.ToDate, tblbooking.Status
    FROM tblbooking
    INNER JOIN tblvehicles ON tblbooking.VehicleId = tblvehicles.id
    WHERE tblbooking.userEmail = UserEmail;
END $$

DELIMITER ;
```

GitHub Link:

https://github.com/Samanvi23/Car_Rental_DBMS