## System Design for Car Rental System

1. User Interface Design:

- Homepage: Displays featured vehicles, seek bar, login/sign in options.

- Vehicle Listings Page: Shows available automobiles with filters (vicinity, date, type).

- Booking Page: Lets customers select cars, condo dates, insurance, and make bills.

- User Dashboard: Displays booking records, agreements, and profile settings.

- Admin Dashboard: Manages fleet, bookings, bills, and generates reports.

2. Database Structure:

- Users Table: Stores consumer info (ID, call, e-mail, password, position).

- Cars Table: Contains vehicle facts (ID, make, model, year, rate, availability).

- Bookings Table: Records reserving information (ID, user ID, vehicle ID, dates, insurance, repute).

- Payments Table: Stores fee statistics (ID, reserving ID, amount, repute).

- Agreements Table: Holds virtual condo agreements (ID, reserving ID, agreement text).

3. Backend Architecture:

- C# Handles server-aspect good judgment, routing, and APIs.

- MySQL Database: Manages person, automobile, reserving, payment, and settlement facts.

- Authentication Middleware: Ensures secure person login and get entry to control.

- Payment Gateway Integration: Facilitates secure online bills for bookings.

- Email Notifications: Sends automatic emails for reserving confirmations and updates.

4. Frontend Structure:

- C#: Builds person interfaces and additives for a continuing experience.

- State Management (Redux or Context API): Manages utility nation, authentication, and booking information.

- Responsive Design: Ensures usability across gadgets (computing device, cellular, tablet).

Security Measures:

SSL Encryption: Secures information at some point of transmission.

User Authentication: Uses JWT or OAuth for steady login.

Data Validation: Ensures enter information is secure to prevent vulnerabilities.

Role-Based Access Control (RBAC): Manages person permissions based on roles.

Scalability and Performance:

Cloud Hosting: Offers scalability, reliability, and overall performance (AWS or Google Cloud).

Load Balancing: Distributes site visitors for advanced performance.

Caching: Reduces database queries for faster responses.

Monitoring and Logging: Monitors overall performance and logs for troubleshooting.

Third-Party Integrations:

Payment Gateway: Integrates steady charge processing.

Google Maps API: Provides region offerings for pickup/drop-off.

Email Service Provider: Sends transactional emails and notifications.

Backup and Recovery:

Regular Backups: Ensures information safety with computerized backups.

Disaster Recovery Plan: Establishes protocols for data recuperation.

Testing Strategy:

Unit, Integration, and End-to-End Testing: Ensures gadget reliability and functionality.

User Acceptance Testing (UAT): Validates machine usability with real users.

Documentation and Training:

System Documentation: Provides certain publications on device usage and deployment.

Training Materials: Prepares schooling classes for apartment corporation team of workers.

This gadget layout encompasses the essential additives, technology, security features, scalability strategies, integrations, checking out techniques, and documentation/schooling factors for the car condominium gadget.