Plus Points in Implementation (Overall Evaluation Criteria)

1. Authentication:
- Implement robust user authentication protocols to ensure secure access.

2. Cost Estimation - Time and Space:
- Conduct a thorough analysis of time and space complexity in the system.
- Utilize efficient algorithms and data structures to optimize both time and space requirements.

3. Handling System Failure Cases:
- Implement fault-tolerant mechanisms to address system failures.
- Employ backup and recovery strategies for data integrity.
- Develop comprehensive error recovery procedures to minimize downtime.

4. Object-Oriented Programming Language (OOPS):
- Choose a robust OOPS language for structured and modular code.
- Leverage OOPS principles such as encapsulation, inheritance, and polymorphism for maintainability and extensibility.

5. Trade-offs in the System:
- Clearly define and document trade-offs made during system design.
- Evaluate and communicate the rationale behind architectural and design decisions.
- Consider trade-offs in terms of performance, scalability, and maintainability.

6. System Monitoring:
- Implement comprehensive monitoring tools to track system performance.
- Utilize real-time dashboards and logging mechanisms to promptly identify and address issues.

7. Caching:
- Integrate caching mechanisms to enhance system response times.
- Utilize caching for frequently accessed data to reduce database load.
- Implement cache eviction policies for optimal resource utilization.

8. Error and Exception Handling:
- Develop a robust error and exception handling framework.
- Provide meaningful error messages for effective debugging.
- Regularly review and update error-handling strategies based on system usage patterns.

Instructions:

1. Read and Understand the Problem Statement:
- Carefully read the problem statement provided. Understand the requirements, inputs, expected outputs, and any constraints mentioned.

2. Choose a Programming Language:
- Select a programming language you are comfortable with and that is suitable for solving the problem described in the case study.

3. Design Your Solution:

- Plan the overall structure of your solution. Consider the algorithms, data structures, and any potential optimizations needed.

4. Write the Code:
- Implement your solution in code. Follow best practices for coding standards, such as meaningful variable names, proper indentation, and comments where necessary.
- Break down the problem into smaller functions or modules to improve code readability and maintainability.

5. Test Your Code:
- Test your code thoroughly with different sets of input data, including edge cases and boundary conditions.
- Ensure that your code produces the expected outputs for all test cases.

7. Document Your Code :
- Consider adding documentation or comments to explain the logic and purpose of your code, especially for complex parts or algorithms.

8. Submit Your Solution:
- Once you're satisfied with your code and it meets all the requirements, submit your solution on GitHub and share the GitHub link.

9. Demonstration:
- Include a demonstration video showcasing key features of the ride-sharing platform.
- Alternatively, use screenshots to visually highlight the user interface and functionality.

Shuttle Management System: A Smart Campus Transit Solution.

A **Shuttle Management System** is designed to provide **efficient, cost-effective, and seamless transportation** for students across a university campus. The system allows **real-time shuttle booking, route optimization, digital fare management, and trip history tracking**, ensuring a **hassle-free commuting experience**.

I. Multi-Route Management for Efficient Campus Transit

To **optimize shuttle operations**, the system allows administrators to **create and manage multiple routes** covering various stops across the university.

Key Features for Admins

✅ **Flexible Route Creation**

- Admins can define **multiple shuttle routes**, each with a **unique set of stops**.
- Routes are optimized based on:
    - **Peak hours** (high student traffic).
    - **Class schedules** (ensuring timely arrivals).
    - **Demand analysis** (popular vs. low-traffic stops).

✅ **Dynamic Stop Management**

- Stops can be **added, removed, or modified** based on real-time demand.
- The system automatically **suggests optimal routes** based on student travel patterns.

✅ **Multi-Route Handling**

- Different shuttles operate on **different routes simultaneously**, covering all parts of the university.
- Real-time monitoring ensures **efficient vehicle allocation**.

🔹 **Example Use Case:**

- **Morning Routes:** Focus on getting students from dormitories to academic buildings.
- **Evening Routes:** Prioritize return trips from campus to residences.

🔹 **Outcome:**

- **Minimizes congestion** by offering multiple routes.
- **Ensures timely pick-ups and drop-offs** based on class schedules.
- **Optimizes vehicle usage**, reducing operational costs.

II. Student Profile Creation & Digital Access

Students need a **personalized account** to access the shuttle booking system, enabling **secure and convenient trip management**.

Key Features for Students

✅ **Email-Based Profile Registration**

- Students sign up using their **university email ID** for authentication.
- Prevents **unauthorized access** from non-university individuals.

✅ **Digital Wallet Integration**

- Each student is assigned a **virtual wallet** where trip credits (points/money) are stored.
- **Admins can allocate** funds or students can **recharge their accounts** via online payment methods.

🔹 **Example Use Case:**

- A student logs in and books a ride using their **university email** and available wallet balance.

🔹 **Outcome:**

- **Simplifies booking process** by personalizing user experience.
- **Prevents fraudulent use** by enforcing university-only access.

III. Admin-Assigned Points for Digital Fare Management

To ensure a **cashless and transparent fare system**, admins can **assign travel points** to students, allowing **easy fare deductions for each trip**.

Key Features

✅ **Admin-Assigned Travel Points**

- Each student receives a **predefined number of points (credits)** based on their eligibility.
- Admins can:
    - Allocate **monthly/semester-based credits**.
    - Add bonus points for **frequent users or special occasions**.
    - Deduct points for **violations (e.g., last-minute cancellations, no-shows)**.

✅ **Auto-Fare Deduction**

- Upon booking confirmation, **points are deducted automatically** from the student's wallet.
- Dynamic pricing ensures:
    - **Lower fare for off-peak hours**.
    - **Higher fare during peak travel times**.

✅ **Recharge & Payment Options**

- If students **exhaust their assigned points**, they can **recharge their wallet** via:
    - Online payment methods (UPI, credit/debit card, digital wallets).
    - In-person payment kiosks at campus locations.

🔹 **Example Use Case:**

- A student is allocated **500 points per month** and can book rides until their balance runs out.

🔹 **Outcome:**

- **Eliminates the need for cash transactions**.
- **Encourages students to plan travel efficiently** to avoid running out of points.
- **Gives admins control over fare allocation and student usage monitoring**.

<u>IV. Smart Shuttle Booking with Best Route Suggestions</u>

Students can book a shuttle **between two stops**, with AI-powered suggestions for **nearby stops and the most efficient route**.

Key Features

✅ **Intelligent Stop Recommendations**

- The system suggests **the closest shuttle stops** based on the student's:
    - **Current location (GPS-based auto-detection)**.
    - **Preferred departure time**.
    - **Historical booking data**.

✅ **Route Optimization & Best Path Selection**

- The system **analyzes available routes** and recommends the:
    - **Fastest option (minimal travel time)**.
    - **Least crowded route (based on real-time occupancy tracking)**.
    - **Most cost-effective choice (cheapest fare based on distance)**.

🔹 **Example Use Case:**

- A student at **Library Stop** wants to go to the **Sports Complex**. The system suggests:
    - **Route A:** Direct route, 15 min, 4 points.
    - **Route B:** Requires changing buses, 10 min, 3 points.

🔹 **Outcome:**

- **Saves student time** by suggesting **optimal routes**.
- **Improves shuttle occupancy efficiency** with **smart allocations**.

V. Bus Transfer for Best Route Selection

Students can **change buses at specific stops** to reach their destination via the shortest/quickest route.

Key Features

✅ **Seamless Route Transfers**

- The system automatically:
    - Detects if a **route change is needed**.
    - Suggests the **best transfer stop**.
    - Shows available **connecting shuttles & wait times**.

✅ **One-Ticket System for Multi-Leg Journeys**

- If a student **switches buses**, their **fare is automatically adjusted**, preventing multiple deductions.

🔹 **Example Use Case:**

- A student traveling from **Dormitory Stop → Science Building**:
    - **Direct Route:** 25 minutes.
    - **Transfer Route:** Change at **Main Gate Stop**, saves 10 minutes.

🔹 **Outcome:**

- **Reduces travel time** by offering **smart transfers**.
- **Optimizes shuttle occupancy** by **balancing passenger distribution**.

VI. Trip History & Booking Records

Students can track their **past trips and fare deductions**, helping them **plan future rides effectively**.

Key Features

✅ **Comprehensive Ride History**

- Students can **view and manage past bookings** with details such as:
    - **Date & time of the ride**.
    - **Start & end stops**.
    - **Points deducted for the trip**.

✅ **Frequent Route Suggestions**

- The system **remembers frequently booked routes** and suggests them for quick access.

✅ **Expense Tracking & Wallet Statements**

- Students can download a **detailed expense report**, showing:
    - **Total points used per week/month**.
    - **Upcoming top-ups or allocated travel credits**.

🔹 **Example Use Case:**

- A student wants to check if they have enough points left for the week and reviews their last **5 trips** to manage travel better.

🔹 **Outcome:**

- **Increases transparency** in ride usage and fare deductions.
- **Encourages students to plan their commute** based on travel history.