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:

Document Format:
- Combine textual explanations, screenshots, and code snippets for clarity.
- Organize information in a structured manner, following a logical flow.

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

Smart Cab Allocation System for Efficient Trip Planning

1. Admin's Cab Allocation Optimization:

Objective:
Develop an algorithm to optimize cab allocation for trips, reducing overall travel distance.
Tasks:

- Develop an algorithm suggesting the best cab based on proximity to the trip start location.
- Integrate real-time location data for cabs and trip start locations.
- Test the algorithm's effectiveness in minimizing travel distance and improving overall trip efficiency.

2. Employee's Cab Search Optimization:

Objective: Enhance the user experience for employees searching for cabs by suggesting nearby cabs that are currently in use.
Tasks:

- Utilize real-time data to display cabs currently engaged in trips and nearby to the employee's location.
- Evaluate the system's effectiveness in providing quick and relevant cab suggestions for employees.

3. Real-Time Location Data Integration:

Objective: Ensure seamless integration of real-time location data for cabs and trip start locations to enhance the accuracy of suggestions.
Tasks:

- Establish a robust system for real-time tracking of cab locations.
- Integrate location data into the cab allocation algorithm to provide up-to-date suggestions.
- Address potential challenges such as data latency or inaccuracies to maintain system reliability.