

System Design for Modern Applications

Task 1:

(marks 50)

Design the backend system for Uber's ride-hailing service.

Key Features to Consider:

1. User Authentication and Authorization

2. Geo location and Mapping

2.1 Explain the system's approach to real-time tracking of drivers and riders.

2.2 How would you design a system to efficiently calculate and update the estimated time of arrival (ETA)?

3. Ride Matching and Dispatching

3.1 Describe the algorithm for matching riders with available drivers based on factors like distance, traffic conditions, and driver ratings.

3.2 How would you handle ride requests during peak hours?

4. Real-time Updates and Notifications

5. Payment Processing

5.1 Design the payment processing system, including payment gateways, fare calculation, and handling of refunds.

6. Scalability and Fault Tolerance

7. Driver and Rider Ratings

8. Data Security and Privacy

Helping Material:

Schema less MySQL: <https://eng.uber.com/schemaless-part-two-architecture/>

Ring pop architecture: <https://eng.uber.com/ringpop-open-source-nodejs-library/>

Task 2:

(marks 50)

Design the backend System for Shopify's e-commerce platform

Key Features to Consider:

1. Product Catalog and Inventory Management

1.1 How would you design the database schema for the product catalog and manage product inventory?

1.2 Discuss strategies for handling product variations and categories.

2. User Authentication and Authorization

3. Shopping Cart and Checkout

3.1 Design the shopping cart system, including the addition/removal of items and updating quantities.

3.2 Discuss the checkout process, including payment gateways and order confirmation.

4. Order Processing and Fulfillment

4.1 Describe the system's workflow for processing orders, managing order status, and coordinating order fulfillment.

4.2 How would you handle returns and refunds?

5. Search, Analytics and, Recommendations

5.1 Implement a search functionality for products.

5.2 Discuss how you would provide product recommendations based on user behavior.

6. Scalability and High Availability

6.1 How would you ensure the system can handle a large number of concurrent users and transactions?

6.2 Discuss strategies for load balancing and ensuring high availability.

7. Security Measures

7.1 Explain the security measures in place to protect user data, especially during transactions.

7.2 How would you guard against common e-commerce security threats?

8. Customer Reviews and Ratings

8.1 Design a system for customers to leave reviews and ratings for products.











9.1 Discuss how you would prevent fake reviews and manage product ratings.

(you can include figures and tables in your document as per requirement)

Task 3:

(marks 50)

Your task is to write code in react to create a component that replicates the design and features of the leader board shared below and an API in Node JS to send data to the component.

RANK	TEAM NAME	TOTAL GAMES PLAYED	SCORE
	 The Avengers	29	+51,5678
	 Skale	29	+50,9873
	 One Million Bugs	27	+49,6677
4	 The Musketeers	29	+41,33249
5	 Bugs killer	29	+35,67841
6	 Foo Fighters	25	+29,55667
7	 The Ultimate	25	+25,99851

The above leader board have following features:

1. Displays team score and total games played
2. Team ranking based on their performance
3. Avatar/Profile pictures along with their team names
4. Achievements and badges earned by the team based on performance (E.g. badges for top three team)
5. Real-time update of score, rank, team and total games played
6. The data must come from the backend server built in Node.js connected with MongoDB database.

Create an API that retrieves data from MongoDB and send it the front end component of your application.