**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 26-05-2025 |
| Team ID | LTVIP2025TMID61031 |
| Project Name | shopEZ |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

The shopEZ application will be built using a microservices architecture, with a combination of frontend and backend services. The application will utilize a scalable and secure technology stack to ensure high performance and reliability.

Components

1. Frontend

- Built using React or Angular

- Utilizes RESTful APIs to interact with backend services

- Handles user authentication, product browsing, and order management

2. Backend

- Built using Node.js or Java Spring Boot

- Exposes RESTful APIs for frontend interaction

- Handles business logic, database interactions, and payment processing

3. Database

- Relational database (e.g., MySQL) for storing product information, orders, and customer data

- NoSQL database (e.g., MongoDB) for storing product reviews, ratings, and other unstructured data

4. Payment Gateway

- Integrates with a third-party payment gateway (e.g., Stripe, PayPal) for secure payment processing

5. API Gateway

- Acts as an entry point for frontend requests, routing them to appropriate backend services

- Handles authentication, rate limiting, and caching

6. Cache Layer

- Utilizes Redis or Memcached to cache frequently accessed data, reducing database load

7. Load Balancer

- Distributes incoming traffic across multiple instances of the application

8. Security

- Implements SSL/TLS encryption for secure data transmission

- Utilizes authentication and authorization mechanisms (e.g., OAuth, JWT) to protect sensitive data

Infrastructure

1. Cloud Provider

- Hosts the application on a cloud provider (e.g., AWS, Google Cloud, Azure) for scalability and reliability

2. Containerization

- Utilizes Docker for containerization, ensuring consistent environments across development, testing, and production

3. Orchestration

- Uses Kubernetes or a similar tool for container orchestration, managing deployment, scaling, and management of containers

Key Considerations

1. Scalability: Design the application to scale horizontally, adding more instances as needed to handle increased traffic.

2. Security: Implement robust security measures to protect sensitive customer data and prevent unauthorized access.

3. Performance: Optimize the application for high performance, utilizing caching, indexing, and other techniques to minimize latency.

4. Monitoring and Logging: Implement monitoring and logging tools (e.g., Prometheus, ELK Stack) to track application performance, identify issues, and troubleshoot problems.

This technical architecture provides a solid foundation for building a scalable, secure, and high-performance e-commerce application like shopEZ.

