**Assignment 02**

**Noman Ejaz**

**S23BSEEN1E0246**

**Software Design and Architecture**

**Submitted to: Sir Talal Sb**

**Submitted from: Noman Ejaz**

**Question 01: Architectural Style Recommendation**

1. **Scalability**: Microservices architecture enables horizontal scaling for individual services, such as product catalog or order processing, ensuring the platform can handle high traffic during peak times.
2. **Modularity**: Each module, such as user authentication, shopping cart, and payment processing, can be developed, deployed, and maintained independently, which aligns with the e-commerce platform's functional requirements.
3. **Resilience**: If one service fails (e.g., payment gateway), other parts of the platform can continue functioning, improving overall reliability and user experience.
4. **Technology Flexibility**: Development teams can use the most appropriate technology stack for each microservice, optimizing performance and development efficiency.
5. **Future Scalability and Adaptability**: As business needs evolve, new features can be added as separate microservices without disrupting existing services, ensuring the platform remains adaptable to future requirements.

**Alignment with Project Goals:**

* **Customer Satisfaction**: Ensures high availability and responsiveness, critical for an e-commerce platform.
* **Business Growth**: Supports the addition of new services or scaling existing ones to accommodate growth.
* **Maintenance and Operations**: Independent modules simplify updates and troubleshooting, reducing downtime.

**Question 02: Coupling Analysis**

**Coupling between Module A (User Authentication) and Module B (Product Catalog)**:  
**No direct Coupling**: No direct coupling refers to the modules in which there is no connection in between. So modules are unrelated with each other and have no interconnections. Such modules are called as the independent modules.

These modules operate independently, with no direct interaction. The user authentication module handles user credentials, while the product catalog manages product information.

**Coupling between Module C (Shopping Cart) and Module B (Product Catalog)**:  
**Data coupling :** Data coupling exists when data are passed from one modules to another module via argument list. So simple data are passed and a one to one correspondence of items exists. As such data coupling is simpler and does not make much errors.

The shopping cart retrieves product information (e.g., prices, availability) from the product catalog. They exchange structured data through APIs, ensuring a well-defined dependency.

**Coupling between Module D (Order Processing) and Module A (User Authentication)**:  
**Control Coupling:** When one module passes parameters to control the activity of another module. it is said that there is control coupling between the two.

The order processing module communicates with the user authentication module to verify user credentials or retrieve user-specific details (e.g., address), introducing a reliance on authentication logic.

**Coupling between Module D (Order Processing) and Module E (Payment Gateway)**:  
**Message Coupling**  
The order processing module sends order details to the payment gateway, which processes payments. This interaction, typically via secure APIs, represents message-based coupling, ensuring loose integration.

**Summary:**

The microservices architecture supports the modularity required for scalability and maintainability in Case Study 1. In Case Study 2, understanding coupling helps identify dependencies and design robust, loosely coupled modules for an efficient and adaptable e-commerce system.