**COMSATS UNIVERSITY ISLAMABD (ABBOTTABAD)**

**CAMPUS**

**NAME MUHAMMAD BILAL**

**RE NO FA20-BSE-071**

Lab task

**Task 1**

**1 - Scalability Issues in a Web Application**

Applications struggle to handle increasing traffic, leading to performance bottlenecks and downtime.

**Challenges:**

* Poorly designed architecture that doesn't scale horizontally.

**Real-World Example:**

**E-commerce Platforms** like Amazon or Shopify experience massive spikes during events like Black Friday.

**2. Real-Time Processing Latency**

Real-time systems such as stock trading or video conferencing apps fail to deliver low-latency responses.

**Challenges:**

* High throughput requirements overwhelm the system.

**Real-World Example:**

**Zoom** needs to provide seamless video conferencing for millions of concurrent users globally

**3. Data Consistency Problems in Distributed Systems**

Distributed systems encounter inconsistencies, leading to mismatched data across replicas or regions.

**Challenges:**

* Network partitions or delays in propagating updates.

**Real-World Example:**

* **Uber** synchronizing ride data between drivers and riders in different regions.

**4. Security Vulnerabilities**

Systems handling sensitive data are prone to breaches due to architectural flaws.

**Challenges:**

* Inadequate encryption and access control mechanisms.

**Real-World Example:**

The Equifax Data Breach exposed sensitive user data due to poor patch management.

**5. Monolithic Systems Hindering Agility**

Legacy monolithic applications make it hard to add features or scale specific components.

**Challenges:**

* Coupled services create dependencies, increasing the risk of failure.

**Real-World Example:**

A **banking system** that struggles to integrate modern digital banking solutions due to its monolithic design.

**Task 2**

Solve one of them with solution

**Scalability Issues in an E-Commerce Platform**

An e-commerce platform is experiencing slow response times and server crashes during high-traffic events like Black Friday sales. The current architecture is unable to handle increased traffic, resulting in loss of sales and poor user experience.

**Challenges:**

1. **High Traffic**:
2. **Database Bottleneck**:

**Solution**

**1. Transition to Microservices**

* Break the monolith into smaller, independent services:
  + **Authentication Service**: Handles user sign-ins and permissions.
  + **Product Catalog Service**: Manages product listings and search queries.
  + **Order Management Service**: Handles cart and order processing.

**2. Implement Load Balancing**

* Use a **Load Balancer** (e.g., AWS ELB, NGINX) to distribute traffic across multiple instances of each service.

**3. Optimize Database**

**Step 1**: Use **Read Replicas**:

**Step 2**: **Database Partitioning**:

**Step 3**: Add a **Caching Layer**:

**4. Deploy a CDN for Static Content**

* Store static assets like images, CSS, and JavaScript in a **Content Delivery Network (CDN)** .

**5. Use Asynchronous Processing**

* Implement a **Message Queue** (e.g., RabbitMQ, Kafka) for processing non-critical tasks asynchronously.