DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
Discover. Learn. Empower.

# Experiment No: 02

Student Name: Archita Srivastava
UID: 23BCS12459
Branch: BE-CSE
Section/Group: KRG_2B
Semester: 6th
Date of Performance: 14/01/26
Subject Code: 23CSH-314
Subject Name: System Design

**Aim:** To design and implement system design of an e-commerce website.

## Objective:

- To understand the overall architecture and workflow of an e-commerce website.
- To design a scalable and reliable system for product browsing, order processing, and payments.
- To create High-Level Design (HLD) and Low-Level Design (LLD) diagrams using draw.io.
- To analyze key system design decisions and their impact on performance, scalability, and reliability.
- To understand how components such as frontend, backend services, database, and third-party integrations interact in an e-commerce platform.

## Procedure:

- Study the working principles of an e-commerce website and identify its core components such as user interface, product catalog, order management, payment processing, and database.
- Design the High-Level Design (HLD) representing the overall system architecture, including frontend, backend services, database, and third-party integrations, using draw.io.
- Create the Low-Level Design (LLD) illustrating database schemas, APIs, service interactions, and internal workflows using draw.io.
- Define all the functional and non-functional requirements.
- Analyse system latency, throughput, and response time under varying load conditions.
- Analyse the collected performance metrics and evaluate how system design choices impact scalability, performance, and reliability.

## Functional Requirements:

I. The system shall allow users to register, log in, and log out securely.

II. The system shall allow users to browse products by category and view product details such as price, description, and availability.

III. The system shall provide a search functionality to find products using keywords.

IV. The system shall allow users to add products to a shopping cart and update or remove items from the cart.

V. The system shall allow users to place orders by completing the checkout process.

VI. The system shall support online payment processing through integrated payment gateways.

VII. The system shall allow users to view order history and order status.

VIII. The system shall maintain inventory levels and update stock after each order.

Non Functional Requirements:

I. The system shall support a target scale of 100 million daily active users (DAU) with 10 orders processed per second.

II. The system shall balance consistency and availability based on the criticality of each module.

III. The product search module shall ensure high availability to allow users to browse products efficiently at all times.

IV. The payment and order placement modules shall ensure strong consistency to prevent incorrect transactions.

V. The inventory management module shall maintain high data consistency to avoid stock mismatches.

VI. The system shall maintain an average response latency of approximately 200 ms.

VII. The system shall support both horizontal and vertical scaling to handle traffic growth.

Outcome:

I. Designed a scalable microservices-based e-commerce architecture covering search, cart, order, payment, and inventory modules.

II. Implemented secure authentication and authorization using an API Gateway and User Service.

III. Achieved high availability for product search using Elasticsearch and CDC-based data synchronization.

IV. Ensured strong consistency for critical workflows such as order placement, payment processing, and inventory updates.

V. Designed an event-driven system using Kafka for reliable order and inventory state propagation.

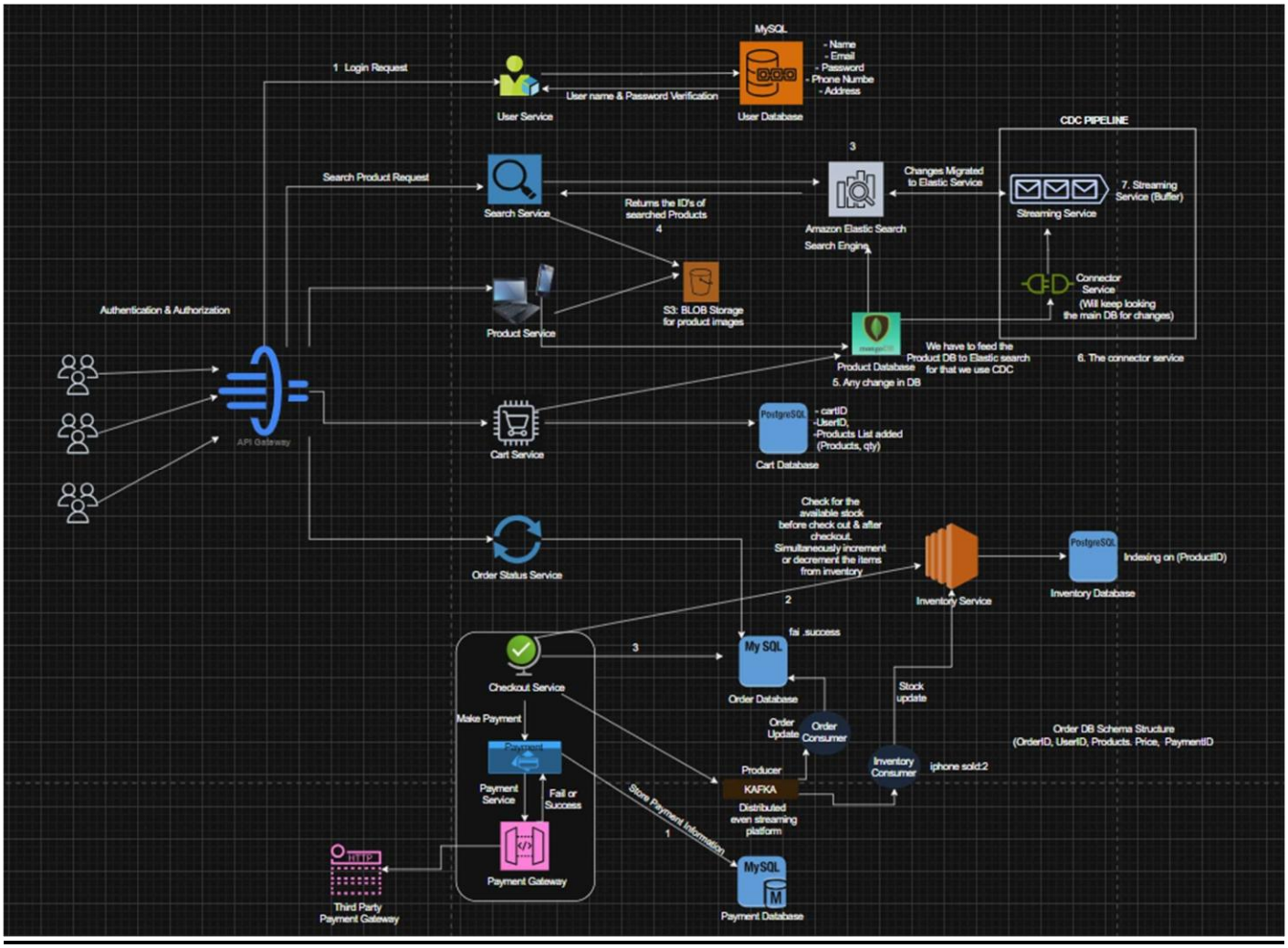
VI. Enabled efficient product discovery using indexed search and optimized data storage.

VII. Incorporated fault tolerance and resilience through asynchronous processing and service isolation.

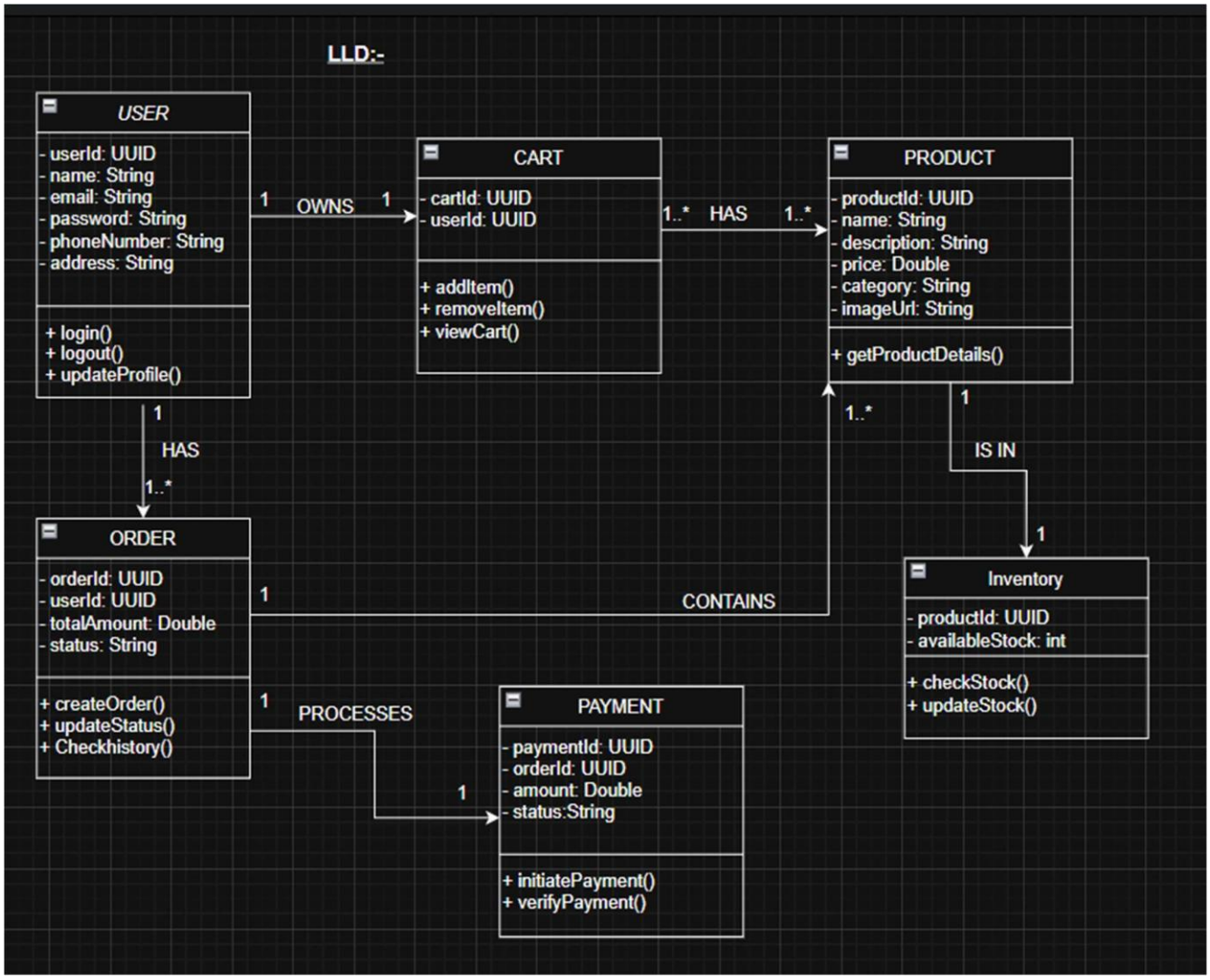
VIII. Analyzed system performance in terms of latency, throughput, and scalability under high user load.

IX. Demonstrated practical application of HLD and LLD concepts using industry-standard design patterns. X. Gained hands-on experience in designing production-grade distributed systems.

High Level Design:-




Low Level Design:-


LLD:-
USER
- userId: UUID
- name: String
- email: String
- password: String
- phoneNumber: String
- address: String
+ login()
+ logout()
+ updateProfile()
1
OWNS
1
CART
- cartId: UUID
- userId: UUID
+ addItem()
+ removeItem()
+ viewCart()
1..*
HAS
1..*
PRODUCT
- productId: UUID
- name: String
- description: String
- price: Double
- category: String
- imageUrl: String
+ getProductDetails()
1
HAS
1..*
1..*
IS IN
1
ORDER
- orderId: UUID
- userId: UUID
- totalAmount: Double
- status: String
+ createOrder()
+ updateStatus()
+ Checkhistory()
1
CONTAINS
1
PROCESSES
1
PAYMENT
- paymentId: UUID
- orderId: UUID
- amount: Double
- status:String
+ initiatePayment()
+ verifyPayment()
Inventory
- productId: UUID
- availableStock: int
+ checkStock()
+ updateStock()