# Experiment-2

**Student Name:** Arjun Dubey          **UID:** 23BCS13190

**Branch:** CSE                                  **Section/Group:** KRG_3B

**Semester:** 6$^{th}$                          **Date of Performance:** 14/01/26

**Subject Name:** System Design      **Subject Code:** 23CSH-314

1.  **AIM :** To design an online E-commerce platform similar to Amazon/Flipkart for browsing and purchasing products like mobiles, laptops, cameras, and clothes.
    To implement Kafka, Elasticsearch, and a CDC pipeline for real-time data processing, fast search, and scalability.

2.  **Objective:**

    - To develop a scalable online shopping system for product listing, search, and order management.
    - To use Apache Kafka for real-time event streaming and inter-service communication.
    - To implement Elasticsearch for fast and efficient product search.
    - To integrate a CDC pipeline for real-time synchronization between databases and services.

3.  **Tools Required:**

    - Programming Language: Java / Python / JavaScript
    - Backend Framework: Spring Boot / Express.js / Flask
    - Database: MySQL / PostgreSQL / MongoDB
    - API Testing Tool: Postman
    - Design Tool: Draw.io (for HLD diagrams)
    - Web Browser
    - ElasticSearch
    - Kafka
    - CDC connector

4.  **SYSTEM DESIGN / SYSTEM SPECIFICATION:**
4.1. **Functional Requirements:**
    - User should be able to search and find the products based on product title or names.
    - User should be able to view the details of the product llike description, image, available quantity, review, accessed.
    - User should be able to select the quantity and move the product/item into the cart.
    - User should be able to make the payment and should be able to perform the check out.
    - User shoulde be able to check the status of the order.
    - System should be able to manage purchase of items having limited stocks.

### 4.2. Non-functional Requirements:

- Target Scale: 100Million DAU with 10 orders processed per second.Availability – System should be available 24/7
- Consistency & Availability: Here for this system we need both as per the Target Scale.
  Now we should specifiy that, which part of our system needs what?
- Latency: Required: ~200 ms
- Scaling: Horizontal / VerticalConsistency

### 4.3. Core-Entites of the System:

1. User / Client
2. Product
3. Cart
4. Orders
5. Checkout followed by Payment

### 4.4. API Endpoints Creation:

**1. GET API Call:  Prod_Search**

Https://Local_Host/products/search_item = {Search_keywords}

```
HTTP Req
{
   GET: <iPhone 16>
}
HTTP Res
{
   <ProductID:iPhone>
}
```

Now, on front-end if multiple data of respective product is coming in that case the FE becomes faulty -> ultimately increasing the LATENCY.

 For that: we can use Pagination 1, 2 ,3 ,4 ,----- SO ON

**2  GET API Call: View Product Details**

```
       Https://Local_Host/products/{product_id}
       HTTP Req
       {
          GET: <Product_id = 17>
       }



       HTTP Res
       {
          Product_id:17,
          Name: iPhone 17,
          Color: Navy Blue,
          Price: $1099,
          Image Thumbnail: URL  Image
```

}

## 3.POST API Call: Item add in cart
Https://Local_Host/cart/add_products

HTTP Req
```
{
  Product_id = 17,
  Product_id = 16
}
```

HTTP Req Header
```
{
  User_id: 04
}
```

HTTP Res
```
{
  Cart_id: 101
}
```

## 4. PUT API Call: To update any order in the cart

Let's Suppose you want to add one more product into the cart.

## 5.DELETE API Call: To remove any item from the cart

Let's Suppose you want to delete one more product into the cart.

## 6.POST API Call: for check out & Payement
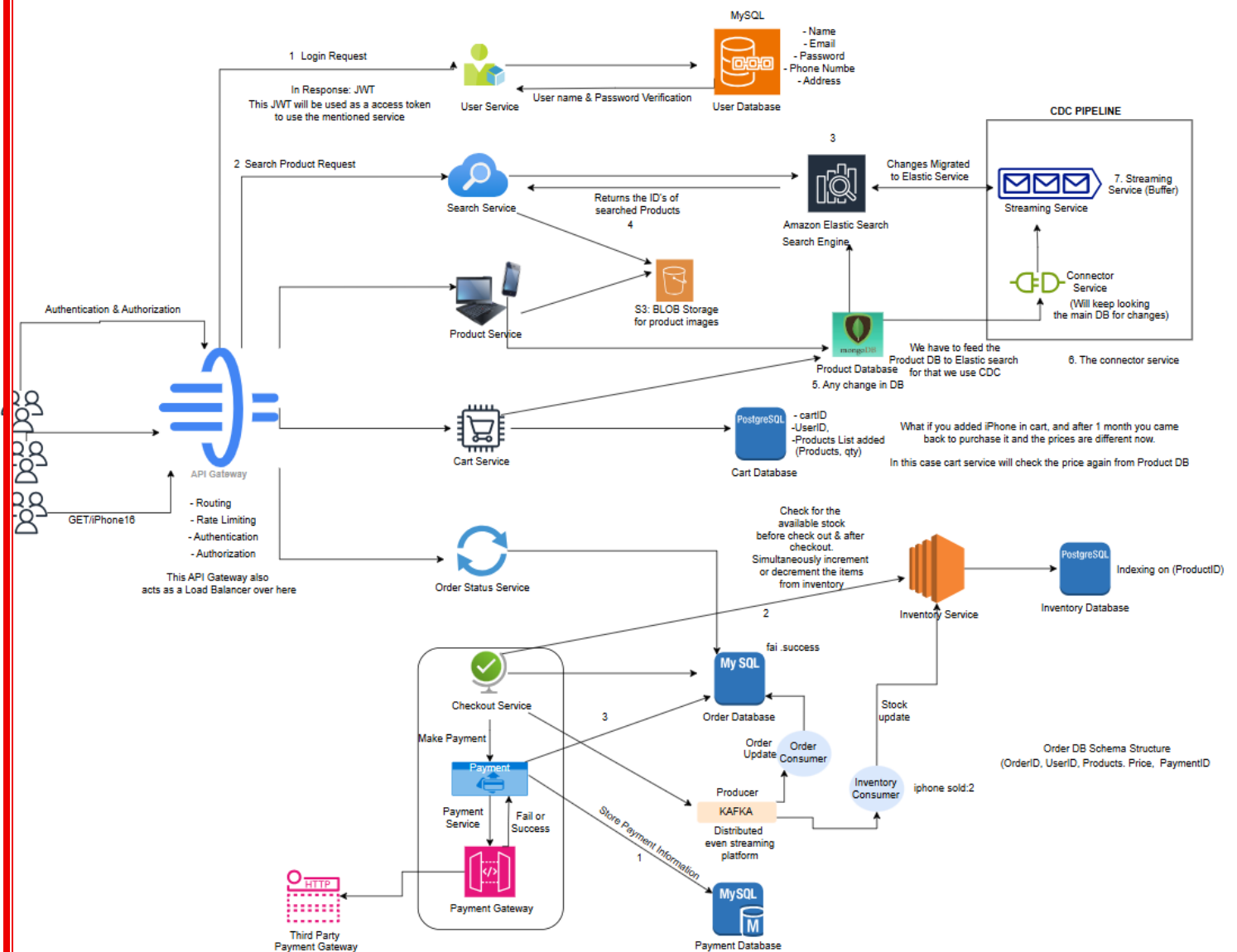Https://Local_Host/checkout -> {post body}
HTTP REQ
```
{
      All products ID's,
      Total Quantity,
      Total Price
}
```
HTTP RES
```
{
      Order_ID
}
```

## 5. HLD(High Level Design):



## 6. Learning Outcomes

- Understand the design and architecture of a scalable E-commerce platform.

- Gain hands-on experience with Apache Kafka for real-time data streaming.

- Learn to implement fast and efficient search using Elasticsearch.

- Understand Change Data Capture (CDC) pipelines for real-time data synchronization.

- Develop skills in integrating distributed systems for high availability and scalability.