



Experiment No: 2

Student Name: Keshav Datta
Branch: BE CSE
Semester: 6th
Subject Name: System Design

UID: 23BCS10423
Section/Group: 23BCS_KRG-2_B
Date of Performance: 14/01/2026
Subject Code: 23CSH-314

1- Aim - Design an Online shopping platform similar to Amazon / Flipkart that will allow users to purchase mobiles, laptops, cameras, clothes etc.

2- Requirements:

A- Functional Requirement

- Users should be able to search products using product name or title.
- Users should be able to view product details such as description, image, available quantity, and reviews.
- Users should be able to select quantity and add products to the cart.
- Users should be able to check out and make payments securely.
- Users should be able to check order status after placing an order.
- The system should manage limited stock inventory and handle race conditions during flash sales.

B- Non-Functional Requirement

- The system should support up to 100 million daily active users and handle around 10 orders per second.
- High availability is required for product search and browsing.
- Strong consistency is required for payment, order placement, and inventory management.
- The expected response time should be around 200 ms.
- The system should support horizontal and vertical scaling.

3- Core-entities of System

- User/Client
- Products
- Cart
- Orders
- Checkout followed by Payment

4- API endpoint creation

a) GET API Call: Prod_Search

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

HTTP Req

```
{  
    GET: <iPhone 16>
```

```
}
```

HTTP Res

```
{  
    List<ProductID:iPhone>  
}
```

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

So we will be using Pagination (1,2,3,...next)

b) GET API Call: View Product Details

Https://Local_Host/products/{product_id}

HTTP Req

```
{  
    GET: <Product_id=17>  
}
```

HTTP Res

```
{  
    Product_id=17,  
    Name: iPhone17,  
    Color: Navy Blue,  
    Price: $1009,  
    Image_URL: URL_image  
}
```

c) 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  
}
```

- d) PUT API Call: To update any order in the cart
- e) DELETE API Call: To remove any item from the cart
- f) **POST API Call: for check out & Payment**
Https://Local_Host/checkout -> {post body}

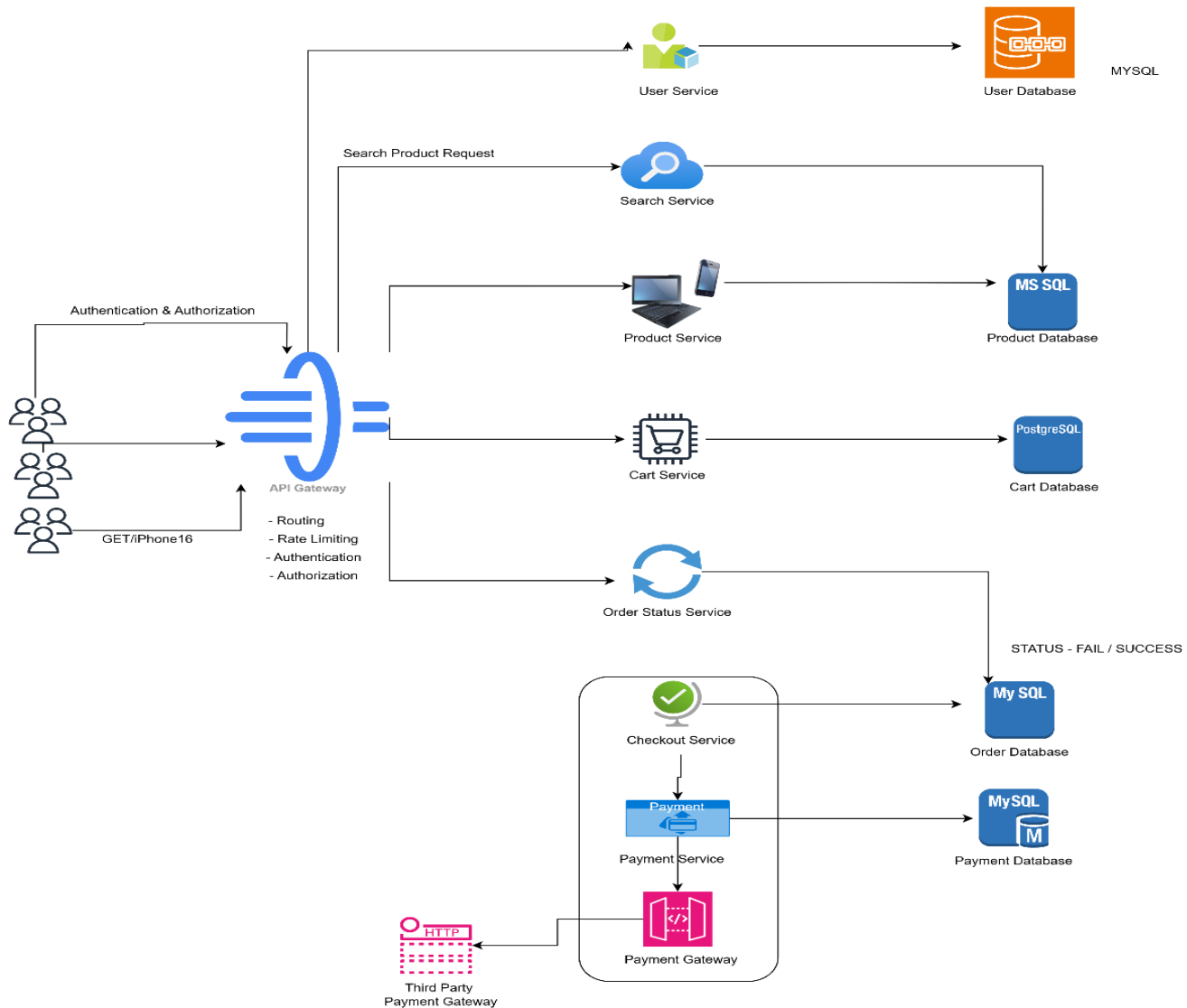
```
HTTP Req  
{  
    All Product Id's,  
    Total Quantity,  
    Total Price  
}  
HTTP Res  
{  
    Order_id  
}
```

```
Https://Local_Host/payment -> {post body}  
HTTP Req  
{  
    Order_id,  
    Payment Type,  
    Payment_Mode  
}  
HTTP Res  
{  
    Confirmation_Status: Succes / Fail  
}
```

- g) **GET API Call: Order Status**
Https://Local_Host/orde_status = {order_id}

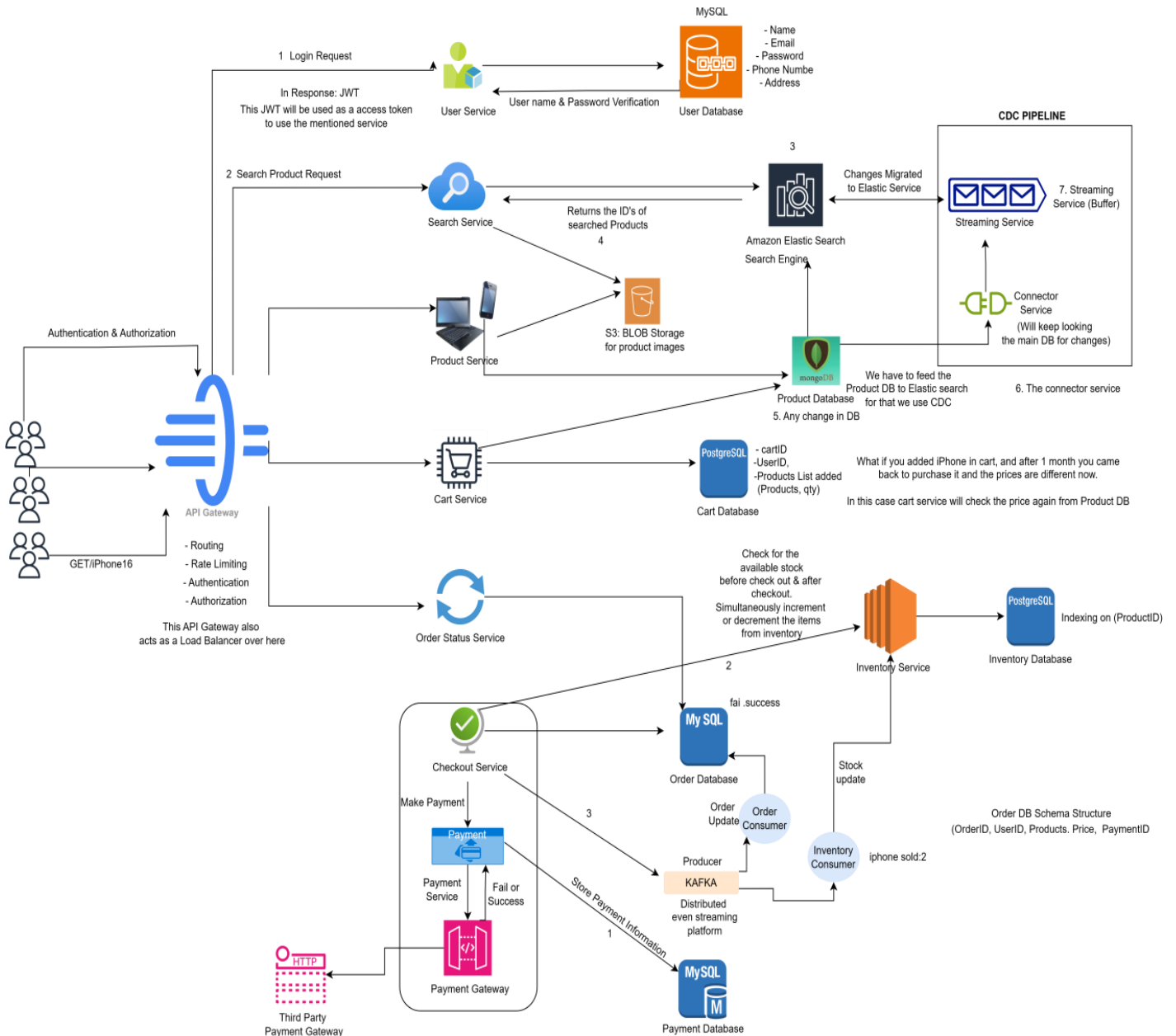
5- High-Level Design

Now according to the functional requirement of the system, we can identify that we have to follow a distributed / micro-services approach not the monolithic one.



This will fulfill all the functional requirements that were listed. Now, we will see the internal implementations of each one of these components in LLD.

6- Low- Level Design





7. Outcome / Result

- A functional Online Shopping Platform design was successfully created.
- The system supports product search, product viewing, cart management, checkout, payment, and order tracking.
- API endpoints were designed to ensure smooth frontend–backend communication.
- Pagination was implemented to reduce latency and improve performance during product search.
- The design handles limited inventory and race conditions, ensuring correct order placement.
- The system is scalable, consistent, and reliable for high user traffic.