**Category**

**Products**

**Cart**

**Order**

**Get cart id**

**After getting successful cart user must login or sign up and authentication should start**

**User login or signup and add to redis**

**Check out**

**Pay**

**After successful payment user redirect to delivery status**

**Project Overview: E-commerce System Workflow**

This project outlines the step-by-step workflow for an E-commerce system that allows users to browse categories, select products, add them to a cart, place an order, make payments, and track delivery. The details are as follows:

**Technologies and Tools Used**

1. **Programming Language:**
   * **Java: The core language for development.**
2. **Framework:**
   * **Spring Boot: Used for building microservices and RESTful APIs.**
3. **Database:**
   * **MySQL: Utilized for data storage, including user details, product catalogs, orders, and delivery addresses.**
4. **Messaging and Notification:**
   * **Kafka: Configured via Docker for sending email notifications.**
5. **Service Discovery:**
   * **Eureka Server: Ensures smooth communication and distribution between microservices.**
6. **API Management:**
   * **API Gateway: Handles routing and load balancing for the microservices.**
7. **Payment Integration:**
   * **Stripe: Integrated for secure and reliable payment processing.**
8. **Delivery Service:**
   * **Delivery addresses are hardcoded in MySQL, restricting service availability to Ukraine and Europe.**

**Note – currently working on AI model to get delivery status by chatgpt**

**1. Browse Categories**

* **Endpoint:** GET http://localhost:8080/category/  
  Retrieve all available product categories. Choose your preferred category from the displayed list.

**2. Search Products by Category**

* **Endpoint:** GET http://localhost:8080/category/searchByCategoryName/  
  Example: Search for products in the "PHONES" category.

Alternatively, you can fetch all products using the endpoint:

* **Endpoint:** GET http://localhost:8080/product/

**3. Add Products to Cart**

* **Endpoint:** POST http://localhost:8085/cart/add  
  Add products to your cart by specifying the product ID and quantity in the request payload.  
  **Example Request:**

{

"item": [

{

"productId": 352,

"quantity": 5

},

{

"productId": 355,

"quantity": 4

}

]

}

**Example Response:**

{

"userId": 2,

"items": [

{

"cartId": 3,

"productId": 352,

"productName": "IPHONE",

"quantity": 5,

"price": 1000.0

},

{

"cartId": 4,

"productId": 355,

"productName": "IPHONE",

"quantity": 4,

"price": 1800.0

}

],

"total": 12200,

"cartCreatedTime": "2024-12-27T19:12:51.1442883"

}

**4. Confirm Cart**

* **Endpoint:** GET http://localhost:8085/cart/getCartById/{userId}  
  Confirm the cart details and proceed to the order service. IT WILL SAVE AND CONFIRM IN DATABASE

**5. Place an Order**

* **Endpoint:** GET http://localhost:8086/order/getCartById/{cartId}  
  Place an order using the cartId.  
  **Example Response:**

{

"orderId": 1,

"cartId": 1,

"orderStatus": "SUCCESSFUL",

"price": 1800

}

**6. Make Payment**

* **Endpoint:** GET http://localhost:9095/pay/{orderId}  
  Pay for the order. A payment URL will be generated.  
  **Example Response:**

{

"status": "SUCCESSFUL",

"message": "HERE IS YOUR LINK TO PAY",

"lineItems": [

{

"adjustableQuantity": {

"enabled": true,

"maximum": 10,

"minimum": 1

},

"price": "price\_1QeJcjFKjxizuB9sqD6XOUIp",

"quantity": 1

}

],

"url": "https://buy.stripe.com/test\_9AQ3d85pY4Sm4X68xU"

}

**Action:** Open the URL in a browser to complete payment.

**7. Delivery Service**

**Register/Login as a User**

* **Signup Endpoint:** POST <http://localhost:8090/user/signup>
* **USER WILL BE INFORMED WHEN SIGNEDUP BY SENDING EMAIL**

**Example Payload:**

{

"userName": "",

"userPhone": "",

"userPassword": "",

"userEmail": "",

"roles": "",

"userHouseNumber": "",

"userStreet": "",

"userLandMark": "",

"city": "",

"state": "",

"country": "",

"postalCode": ""

}

* **Delivery Request Endpoint:** POST http://localhost:8091/deliveryUser/{cartId}/{email}  
  Use this endpoint to finalize the delivery after payment.

**Example Response:**

{

"id": 752,

"createdAt": "2025-01-06T18:58:42.4352385",

"updatedAt": "2025-01-06T18:58:42.4352385",

"userName": "XXXXXX",

"userPhone": "050505050",

"userEmail": "XXXXXX@gmail.com",

"userPassword": "NOT VISIBLE BECAUSE OF PRIVACY REASONS",

"userHouseNumber": "XX",

"userStreet": "XXXXXXX",

"userLandMark": "XXXXXX",

"userCity": "XXXXXXX",

"userState": "XXXXXXX",

"userCountry": "XXXXXXX",

"userPostalCode": XXXXX,

"message": "PARCEL WILL DELIVER IN MAXIMUM 2 DAYS",

"countryDistance": 260, KILOMETERS

"cartId": 1,

"totalAmount": 1800 USD

}

IF USER CANNOT FIND COUNTRY OF CITY THEN CAN CHECK WITH THIS API IF COUNTRY OR CITY EXISTS IN DATABASE OR NOT <http://localhost:8091/Destination/>

1. **Spring Oauth2 authorization server used for security**
2. Error handling done by each repository

**For Returning Customers**

* Delivery service uses a database to calculate the distance between the warehouse and the user’s location based on the country, state, city, and postal code. The estimated delivery time is then provided.