

## API Execution Screenshots

Below are the screenshots from Postman demonstrating the successful execution of all required API scenarios for the Order Management System.

### Scenario 1: Create an Order

- **Endpoint:** POST /orders
- **Description:** Successfully created a new order containing multiple items (T-Shirt and Jeans) in a single atomic transaction.
- **Status Code:** 201 Created

The screenshot shows the Postman interface for creating a new order. The left sidebar lists various API endpoints under the 'Hotwax JAVA' collection. The main window shows a POST request to 'http://localhost:8080/orders'. The 'Body' tab is selected, showing a JSON payload for the order creation:

```
1 {
2     "orderDate": "2025-11-28",
3     "customerId": 1,
4     "shippingContactMechId": 1,
5     "billingContactMechId": 2,
6     "orderItems": [
7         {
8             "productId": 1,
9             "quantity": 2,
10            "status": "APPROVED"
11        },
12        {
13            "productId": 2,
14            "quantity": 1,
15            "status": "PENDING"
16        }
17    ]
18 }
```

Below the body, the response is shown as a 201 Created status with a response time of 2.71 s and a size of 401 B. The response JSON matches the payload sent:

```
1 {
2     "orderId": 1,
3     "orderDate": "2025-11-28",
4     "customerId": 1,
5     "shippingContactMechId": 1,
6     "billingContactMechId": 2,
7     "orderItems": [
8         {
9             "orderItemSeqId": 1,
10            "productId": 1,
11            "quantity": 2,
12            "status": "APPROVED"
13        },
14        {
15            "orderItemSeqId": 2,
16            "productId": 2,
17            "quantity": 1,
18            "status": "PENDING"
19        }
20    ]
21 }
```

### Scenario 2: Retrieve Order Details

- **Endpoint:** GET /orders/{orderId}
- **Description:** Retrieved the full details of the created order, including customer ID, addresses, and the list of order items.
- **Status Code:** 200 OK

The screenshot shows the Postman interface for retrieving order details. The left sidebar lists various API endpoints under the 'Hotwax JAVA' collection. The main window shows a GET request to 'http://localhost:8080/orders/1'. The 'Params' tab is selected, showing a 'Query Params' table:

Key	Value	Description
Key	Value	Description

Below the params, the response is shown as a 200 OK status with a response time of 116 ms and a size of 396 B. The response JSON matches the order created in Scenario 1:

```
1 {
2     "orderId": 1,
3     "orderDate": "2025-11-28",
4     "customerId": 1,
5     "shippingContactMechId": 1,
6     "billingContactMechId": 2,
7     "orderItems": [
8         {
9             "orderItemSeqId": 1,
10            "productId": 1,
11            "quantity": 2,
12            "status": "APPROVED"
13        },
14        {
15            "orderItemSeqId": 2,
16            "productId": 2,
17            "quantity": 1,
18            "status": "PENDING"
19        }
20    ]
21 }
```

### Scenario 3: Update Order (Shipping/Billing Address)

- **Endpoint:** PUT /orders/{orderId}
- **Description:** Updated the Shipping and Billing Contact Mechanism IDs for an existing order.
- **Status Code:** 200 OK

The screenshot shows the Postman interface with the following details:

- Collection:** campusmate
- API:** Hotwax JAVA
- Request Type:** PUT
- URL:** http://localhost:8080/orders/1/items/1
- Body (raw JSON):**

```
1 {
2   "quantity": 5
3 }
```
- Response Status:** 200 OK
- Response Time:** 2.87 s
- Response Size:** 108 B

### Scenario 4: Add New Item to Order

- **Endpoint:** POST /orders/{orderId}/items
- **Description:** Added a new product (e.g., Sneakers) to an existing order.
- **Status Code:** 201 Created

The screenshot shows the Postman interface with the following details:

- Collection:** campusmate
- API:** Hotwax JAVA
- Request Type:** POST
- URL:** http://localhost:8080/orders/1/items
- Body (raw JSON):**

```
1 {
2   "productId": 3,
3   "quantity": 1,
4   "status": "APPROVED"
5 }
```
- Response Status:** 201 Created
- Response Time:** 129 ms
- Response Size:** 171 B

## Scenario 5: Delete Order Item

- **Endpoint:** DELETE /orders/{orderId}/items/{seqId}
- **Description:** Removed a specific item from the order.
- **Status Code:** 200 OK

The screenshot shows the Postman interface with the following details:

- Collection:** Hotwax JAVA
- Request Type:** DELETE
- URL:** <http://localhost:8080/orders/1/items/1>
- Headers:** Authorization (7)
- Query Params:** Key: Value, Description: Description
- Body:** Raw, Preview, Visualize
- Response:** 200 OK, 1.93 s, 108 B, Save Response

## Scenario 6: Delete Entire Order

- **Endpoint:** DELETE /orders/{orderId}
- **Description:** Deleted the order header. Due to ON DELETE CASCADE configuration in the database, all associated order items were automatically removed.
- **Status Code:** 200 OK

The screenshot shows the Postman interface with the following details:

- Collection:** campusmate
- Request Type:** DELETE
- URL:** <http://localhost:8080/orders/1>
- Headers:** Authorization (7)
- Query Params:** Key: Value, Description: Description
- Body:** Raw, Preview, Visualize
- Response:** (empty response body)