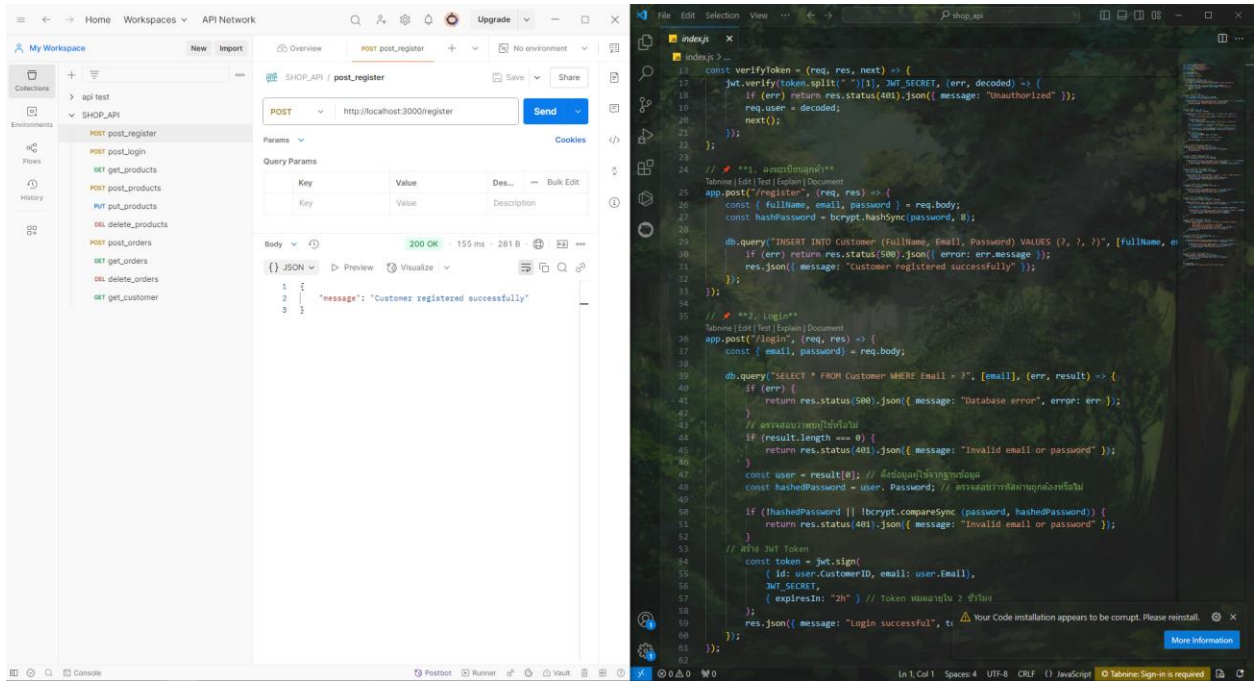
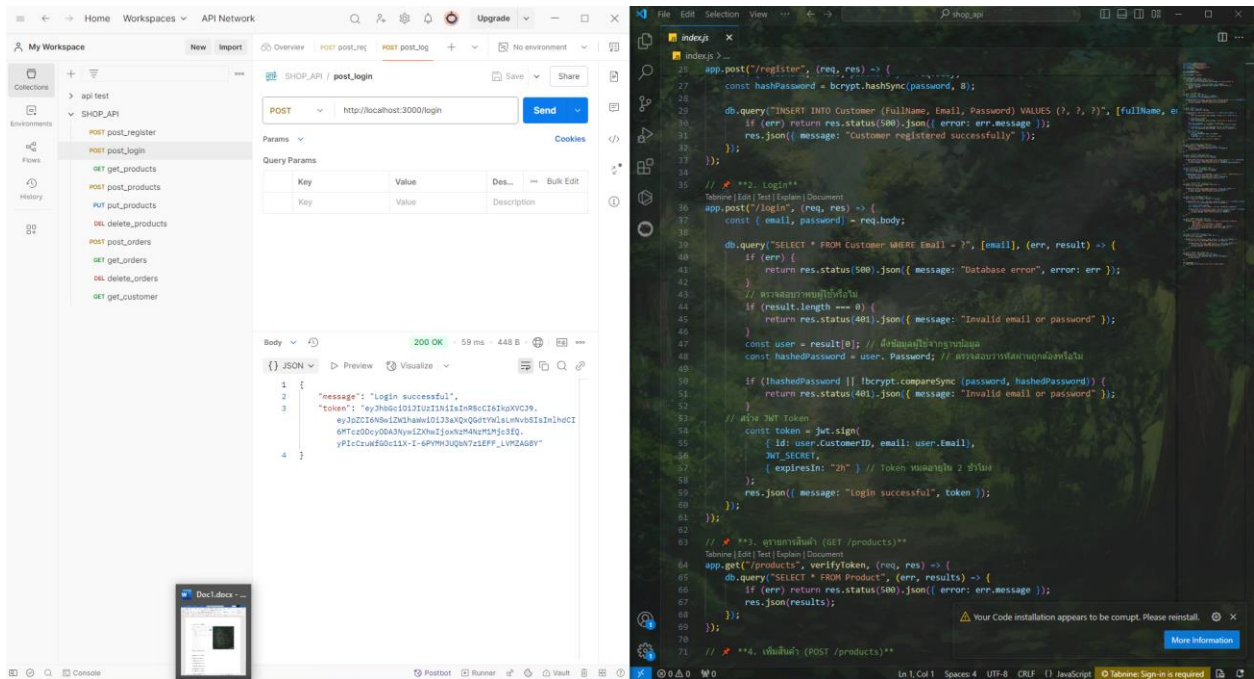


1. ลงทะเบียนลูกค้า POST /register

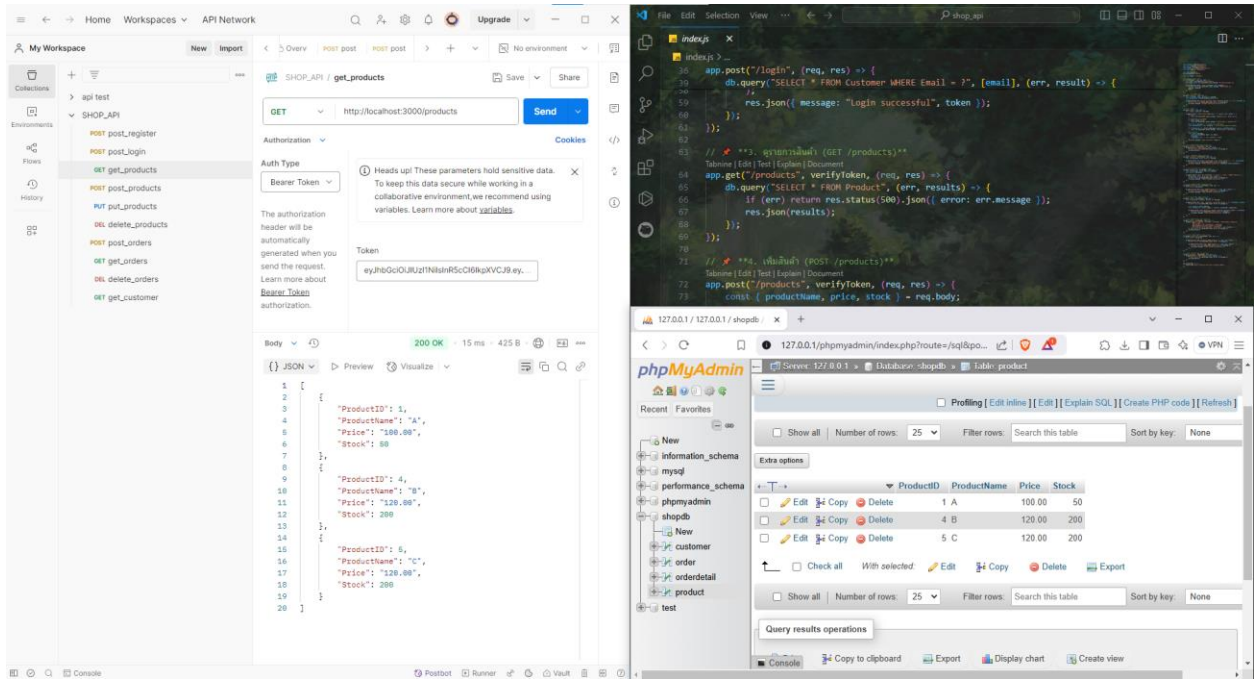


2. เข้าสู่ระบบ (Login) POST /login

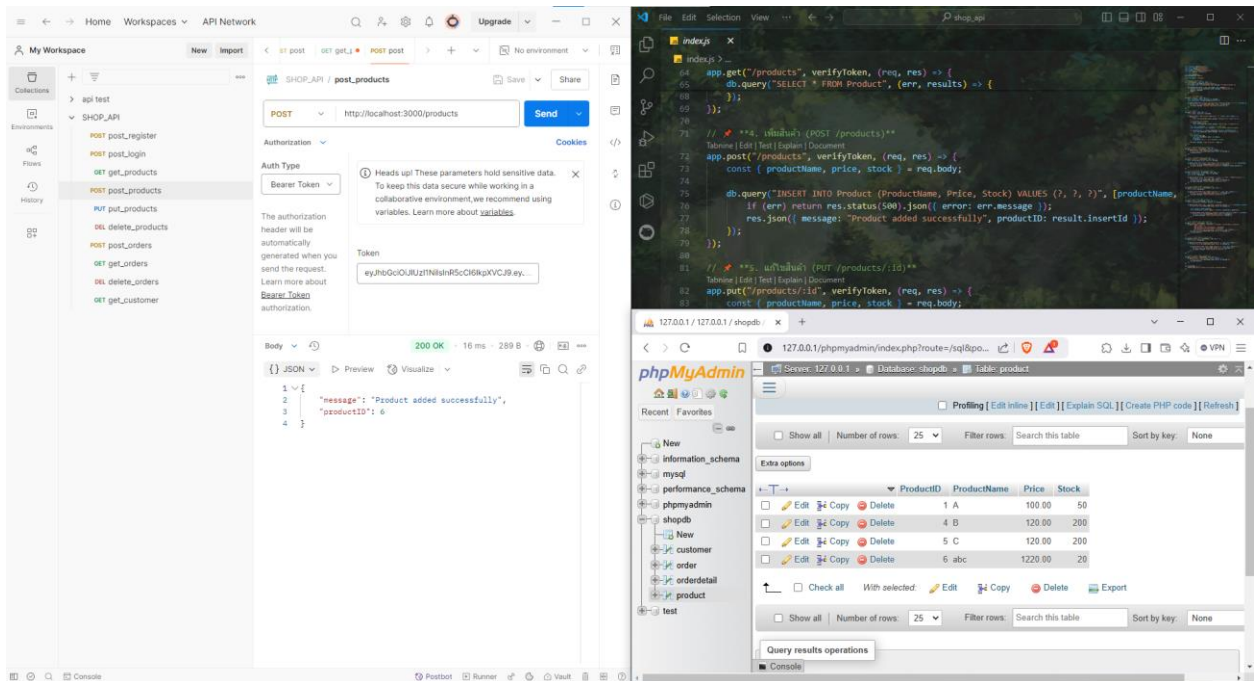
eyJhbGciOiJIUzU1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6NSwiZW1haWwiOiJ3aXQxQGdtYWlsLmNvbSIsIm1hdCI6MTMtczODcyODA3NywiZXBwIjoxOzNzNDM1Mjc3fQ.yPIcCruWfGoc1lX-I-6PYMHJUQbN7z1EFF_LVMZAG8Y



3. ดูรายการสินค้า GET /products



4. เพิ่มสินค้า POST /products



5. แก้ไขสินค้า PUT /products/:id

The screenshot displays two applications side-by-side. On the left is the Postman API client, and on the right is the phpMyAdmin database interface.

Postman: A PUT request is configured to `http://localhost:3000/products/6` with a Bearer Token authorization. The response is a 200 OK status with a JSON body: `{ "message": "Product updated successfully" }`.

Code Editor: The `index.js` file shows the implementation of the PUT endpoint. It verifies the token, updates the product in the database using `db.query("UPDATE Product SET ProductName = ?, Price = ?, Stock = ? WHERE ProductID = ?")`, and returns a success message.

phpMyAdmin: The 'product' table is visible, showing 4 rows of data. The table structure is as follows:

ProductID	ProductName	Price	Stock
1	A	100.00	50
4	B	120.00	200
5	C	120.00	200
6	D	120.00	20

6. ลบสินค้า DELETE /products/:id

The screenshot displays two applications side-by-side. On the left is the Postman API client, and on the right is the phpMyAdmin database interface.

Postman: A DELETE request is configured to `http://localhost:3000/products/6` with a Bearer Token authorization. The response is a 200 OK status with a JSON body: `{ "message": "Product deleted successfully" }`.

Code Editor: The `index.js` file shows the implementation of the DELETE endpoint. It verifies the token, deletes the product from the database using `db.query("DELETE FROM Product WHERE ProductID = ?")`, and returns a success message.

phpMyAdmin: The 'product' table is visible, showing 3 rows of data. The table structure is as follows:

ProductID	ProductName	Price	Stock
1	A	100.00	50
4	B	120.00	200
5	C	120.00	200

7. สร้างคำสั่งซื้อ (Order) POST /orders

The screenshot displays the API client interface on the left and the phpMyAdmin database interface on the right. The API client shows a POST request to `http://localhost:3000/orders` with a JSON body containing order details. The response is a 200 OK status with a success message and the order ID. The phpMyAdmin interface shows the 'order' table with three rows of data.

API Client Request:

```
POST http://localhost:3000/orders
```

API Client Response:

```
{
  "message": "Order placed successfully",
  "orderId": 4
}
```

phpMyAdmin 'order' Table:

OrderID	OrderDate	CustomerID
1	2025-01-29	0
3	2025-01-29	1
4	2025-02-05	2

8. ดูรายการคำสั่งซื้อ GET /orders

The screenshot displays the API client interface on the left and the phpMyAdmin database interface on the right. The API client shows a GET request to `http://localhost:3000/orders` with a Bearer Token. The response is a 200 OK status with a JSON array of order details. The phpMyAdmin interface shows the 'order' table with two rows of data.

API Client Request:

```
GET http://localhost:3000/orders
```

API Client Response:

```
[
  {
    "OrderID": 1,
    "OrderDate": "2025-01-19T17:00:00.000Z",
    "FullName": "kon",
    "ProductID": "Item A",
    "Quantity": 0
  },
  {
    "OrderID": 2,
    "OrderDate": "2025-02-02T17:00:00.000Z",
    "FullName": "nospavit",
    "ProductID": "Item B",
    "Quantity": 2
  }
]
```

phpMyAdmin 'order' Table:

OrderID	OrderDate	CustomerID
1	2025-01-20	1
2	2025-02-03	2

9. ลบคำสั่ง DELETE /orders/:id

The screenshot displays the API Network interface on the left and the phpMyAdmin database interface on the right. In the API Network, the endpoint `DELETE /orders/:id` is selected, showing a successful response with a status of 200 OK and a message: `"message": "Order deleted successfully"`. The phpMyAdmin interface shows the `shopdb` database with the `order` table. The table contains two rows: one with `OrderID` 1 and `OrderDate` 2025-01-29, and another with `OrderID` 5 and `OrderDate` 2025-02-05. The `customer` table is also visible, showing a row with `CustomerID` 1.

10. ดูรายการลูกค้า GET /Customer

The screenshot displays the API Network interface on the left and the phpMyAdmin database interface on the right. In the API Network, the endpoint `GET /Customer` is selected, showing a successful response with a status of 200 OK and a message: `"message": "Customers retrieved successfully"`. The phpMyAdmin interface shows the `shopdb` database with the `customer` table. The table contains six rows, each with a `CustomerID`, `Fullname`, `Email`, and `Password`. The rows are: 1. `CustomerID` 1, `Fullname` kon, `Email` kon@gmail.com, `Password` \$2a\$05\$amwRC/Hcp FLch9uk/Nltv/; 2. `CustomerID` 2, `Fullname` noppawit, `Email` noppawit@gmail.com, `Password` \$2a\$05\$E7FgPwVWw go/h9yVhhuo; 3. `CustomerID` 3, `Fullname` saelao, `Email` saelao@gmail.com, `Password` \$2a\$05\$M.BolccWtMMyDUZRTpbuJ; 4. `CustomerID` 4, `Fullname` wt, `Email` wt@gmail.com, `Password` \$2a\$05\$QFv9yO5u6pnhXKpOe3ul; 5. `CustomerID` 5, `Fullname` wt1, `Email` wt1@gmail.com, `Password` \$2a\$05\$QFv9yO5u6pnhXKpOe3ul; 6. `CustomerID` 6, `Fullname` wt1, `Email` wt1@gmail.com, `Password` \$2a\$05\$QFv9yO5u6pnhXKpOe3ul.