

ACTIVITY 9: Mini E-Commerce API + UI

Leader:

Darryl Jedidiah A. Aranzasu

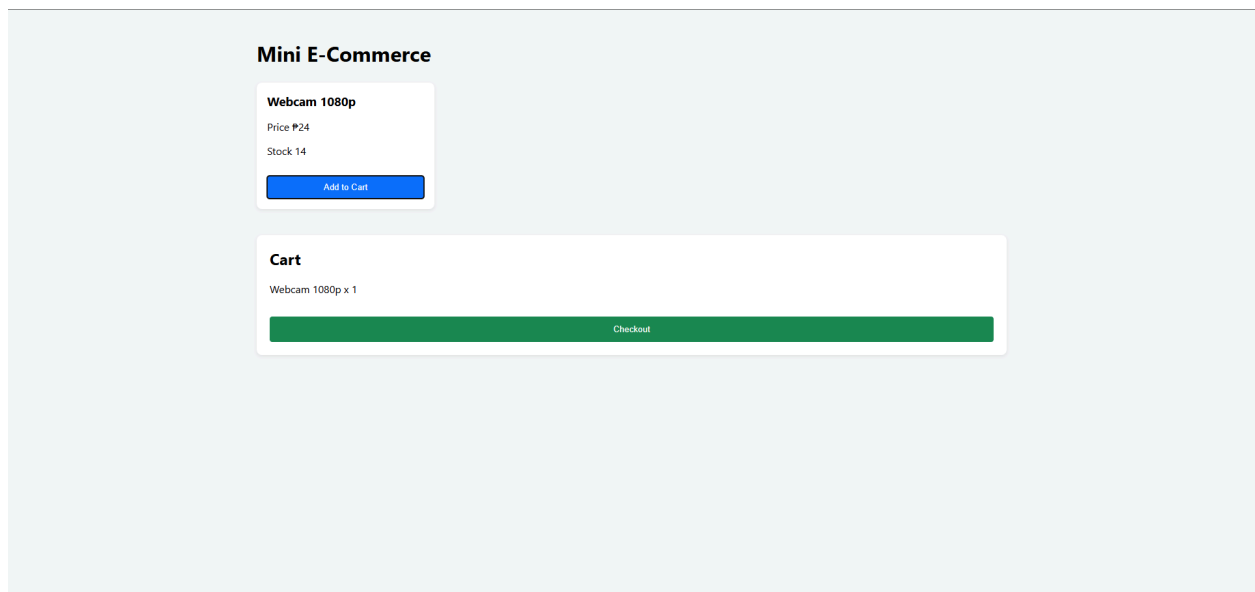
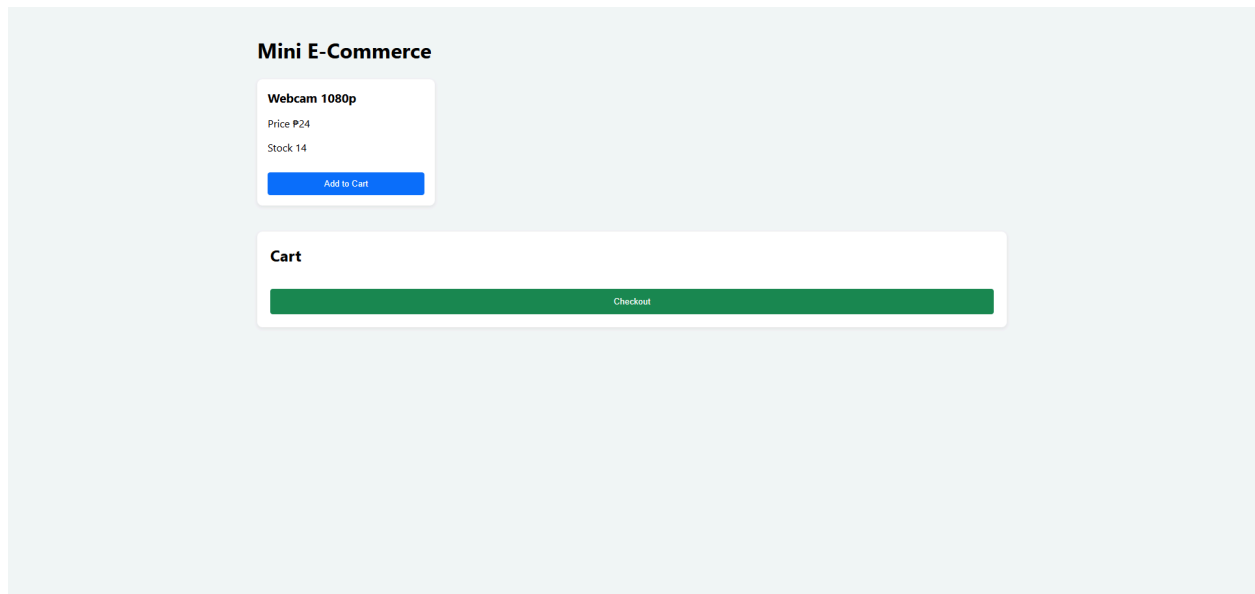
Members:

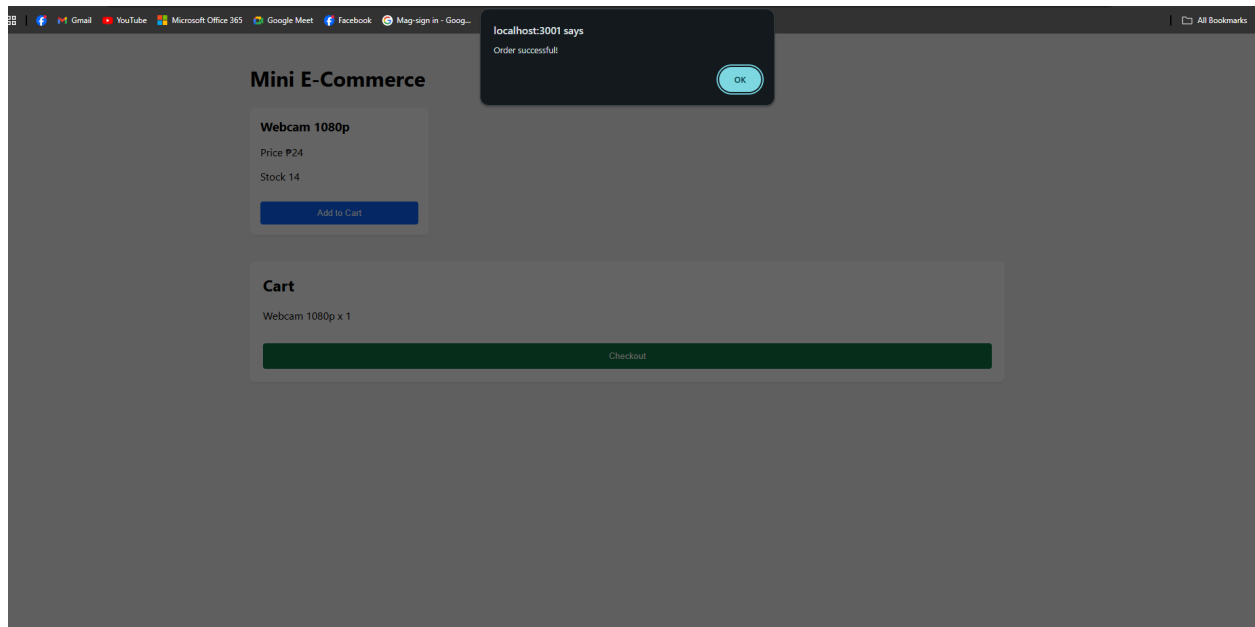
Czyrell Gwen R. Abelita
Darrylene Kaela A. Aranzasu
Demrose Carla C. Gangan
Andrie B. Moral
Nathaniel L. Relox

Activity 9: Mini E-Commerce API + UI

This is a Mini E-Commerce System built with NestJS, TypeORM, and ReactJS. It allows users to browse products, add items to a shopping cart, and place orders. The backend manages CRUD operations for products, cart, and orders, while automatically validating stock and updating quantities. The system is fully Swagger-documented, enabling easy API testing and integration.

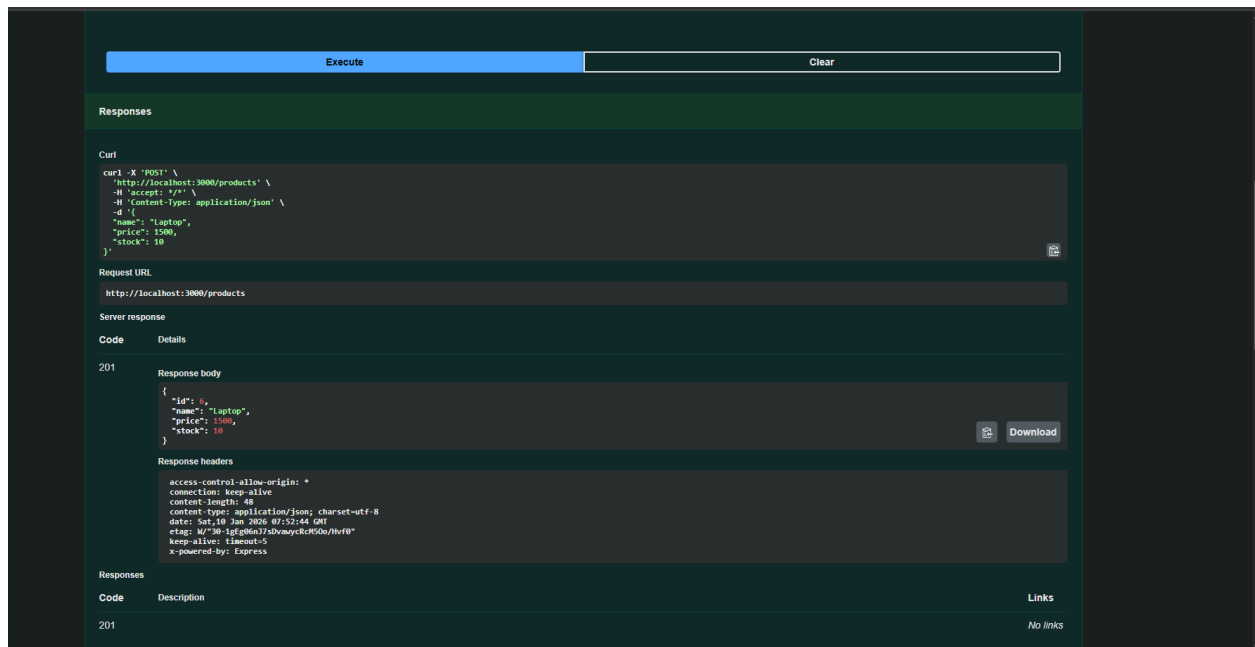
UI Screenshots:





Swagger Screenshot

Products



GET

/products

Get all products

Parameters

Cancel

No parameters

Execute

Clear

Responses

Curl

curl -X 'GET' \n'http://localhost:3000/products' \n-H 'accept: */*'

Request URL

http://localhost:3000/products

Server response

Code

Details

200

Response body

[\n{\n "id": 5,\n "name": "Webcam 1080p",\n "price": 24,\n "stock": 14\n},\n{\n "id": 6,\n "name": "Laptop",\n "price": 1500,\n "stock": 10\n}\n]

Download

Response headers

access-control-allow-origin: *\ncontent-length: 103\ncontent-type: application/json; charset=utf-8

GET

/products/{id}

Get a product by id

Parameters

Cancel

Name

Description

id * required

string

(path)

6

Execute

Clear

Responses

Curl

curl -X 'GET' \n'http://localhost:3000/products/6' \n-H 'accept: */*'

Request URL

http://localhost:3000/products/6

Server response

Code

Details

200

Response body

{\n "id": 6,\n "name": "Laptop",\n "price": 1500,\n "stock": 10\n}

Download

Response headers

access-control-allow-origin: *\nconnection: keep-alive\ncontent-length: 48\ncontent-type: application/json; charset=utf-8\ndate: Sat,10 Jan 2026 07:54:18 GMT\netag: W/"30-1gEg06nJ7SDvaayckn50o/Hvf0"

Responses

Curl

```
curl -X 'PUT' \
  'http://localhost:3000/products/6' \
  -H 'accept: */*' \
  -H 'content-type: application/json' \
  -d '{
    "name": "Flashdrive",
    "price": 1500,
    "stock": 10
  }'
```

Request URL

http://localhost:3000/products/6

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{ "id": 6, "name": "Flashdrive", "price": 1500, "stock": 10 }</pre></div><div><div>Download</div></div></div> <div><div>Response headers</div><div><pre>access-control-allow-origin: * connection: keep-alive content-length: 52 content-type: application/json; charset=utf-8 date: Sat, 10 Jan 2026 08:01:45 GMT etag: W/"34-115C7s02TEU78FvXc1Ks3M/XOM" keep-alive: timeout=5 x-powered-by: Express</pre></div></div>

Responses

Code	Description	Links
200		No links

DELETE

/products/{id} Delete a product

Responses

Curl

```
curl -X 'DELETE' \
  'http://localhost:3000/products/6' \
  -H 'accept: */*' \
```

Request URL

http://localhost:3000/products/6

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{ "id": 6, "name": "Flashdrive", "price": 1500, "stock": 10 }</pre></div><div><div>Download</div></div></div> <div><div>Response headers</div><div><pre>access-control-allow-origin: * connection: keep-alive content-length: 52 content-type: application/json; charset=utf-8 date: Sat, 10 Jan 2026 08:02:35 GMT etag: W/"34-115C7s02TEU78FvXc1Ks3M/XOM" keep-alive: timeout=5 x-powered-by: Express</pre></div></div>

Responses

Code	Description	Links
200		No links

Cart

Responses

Curl

```
curl -X 'POST' \
  'http://localhost:3000/cart' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "productId": 5,
    "quantity": 2
  }'
```

Request URL

http://localhost:3000/cart

Server response

Code	Details
201	<div><div>Response body</div><div><pre>[{ "productId": 5, "name": "Mexican 1080p", "price": 24, "quantity": 2 }]</pre></div><div><div>Response headers</div><div><pre>access-control-allow-origin: * connection: keep-alive content-length: 63 content-type: application/json; charset=utf-8 date: Sat,10 Jan 2026 08:03:33 GMT etag: W/"3f-38aLmIIFW119vy4haZbG/8d0g4" keep-alive: timeout=5 x-powered-by: Express</pre></div></div></div>

Responses

Code	Description
------	-------------

Parameters

No parameters

Execute

Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:3000/cart' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json'
```

Request URL

http://localhost:3000/cart

Server response

Code	Details
200	<div><div>Response body</div><div><pre>[{ "productId": 5, "name": "Mexican 1080p", "price": 24, "quantity": 2 }]</pre></div><div><div>Response headers</div><div><pre>access-control-allow-origin: * content-length: 63 content-type: application/json; charset=utf-8 date: Sat,10 Jan 2026 08:04:08 GMT etag: W/"3f-38aLmIIFW119vy4haZbG/8d0g4" x-powered-by: Express</pre></div></div></div>

Responses

Code	Description	Links
------	-------------	-------

Responses

Curl

```
curl -X 'PUT' \
  'http://localhost:3000/cart/5' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "productId": 5,
    "quantity": 1
  }'
```

Request URL

http://localhost:3000/cart/5

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{ "productId": 5, "name": "Webcam 1080p", "price": 24, "quantity": 1 }</pre></div><div><div>Download</div></div></div> <div><div>Response headers</div><div><pre>access-control-allow-origin: * connection: keep-alive content-length: 61 content-type: application/json; charset=utf-8 date: Sat,10 Jan 2026 08:13:48 GMT etag: W/"3d-1H0tshZXIDahwI3IA0kj96U18A" keep-alive: timeout=5 x-powered-by: Express</pre></div></div>

Responses

Code	Description	Links
200		No links

Responses

Curl

```
curl -X 'DELETE' \
  'http://localhost:3000/cart/5' \
  -H 'accept: */*' \
```

Request URL

http://localhost:3000/cart/5

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{ "productId": 5, "name": "Webcam 1080p", "price": 24, "quantity": 1 }</pre></div><div><div>Download</div></div></div> <div><div>Response headers</div><div><pre>access-control-allow-origin: * connection: keep-alive content-length: 61 content-type: application/json; charset=utf-8 date: Sat,10 Jan 2026 08:15:28 GMT etag: W/"3d-1H0tshZXIDahwI3IA0kj96U18A" keep-alive: timeout=5 x-powered-by: Express</pre></div></div>

Responses

Code	Description	Links
200		No links

Orders

```
curl -X 'GET' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "userId": 1,
    "items": [
      {
        "productId": 5,
        "quantity": 2
      }
    ]
  }'
```

Request URL

http://localhost:3000/orders

Server response

Code	Details
201	<p>Response body</p> <pre>{ "id": 1, "userId": 1, "items": [{ "id": 1, "product": { "id": 5, "name": "Webcam 1080p", "price": 24, "stock": 12 }, "quantity": 2 }] }</pre> <p>Response headers</p> <pre>access-control-allow-origin: * connection: keep-alive content-length: 114 content-type: application/json; charset=utf-8 date: Sat, 10 Jan 2026 08:16:12 GMT etag: W/"72-qPPgu2F1fNX81tc:x+2Ng1fd2fw" keep-alive: timeout=5 x-powered-by: Express</pre>

Responses

Code	Description	Links
201		No links

Responses

```
curl -X 'GET' \
  'http://localhost:3000/orders' \
  -H 'accept: */*'
```

Request URL

http://localhost:3000/orders

Server response

Code	Details
200	<p>Response body</p> <pre>{ { "id": 1, "userId": 1, "items": [{ "id": 1, "product": { "id": 5, "name": "Webcam 1080p", "price": 24, "stock": 12 }, "quantity": 2 }] } }</pre> <p>Response headers</p> <pre>access-control-allow-origin: * connection: keep-alive content-length: 116 content-type: application/json; charset=utf-8 date: Sat, 10 Jan 2026 08:17:16 GMT etag: W/"74-0uuVichPaDf1AmG0u17kinn0788" keep-alive: timeout=5 x-powered-by: Express</pre>

Responses

Name	Description
id * required string (path)	<input type="text" value="1"/>

Execute
Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:3000/orders/1' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:3000/orders/1
```

Server response

Code	Details
200	<p>Response body</p> <pre>{ "id": 1, "userId": 1, "items": [{ "id": 1, "product": { "id": 5, "name": "Webcam 1080p", "price": 24, "stock": 12 }, "quantity": 2 }] }</pre> <p>Response headers</p>

Request URL

```
http://localhost:3000/orders/1
```

Server response

Code	Details
200	<p>Response body</p> <pre>{ "id": 1, "userId": 1, "items": [{ "id": 2, "product": { "id": 5, "name": "Webcam 1080p", "price": 24, "stock": 13 }, "quantity": 1 }] }</pre> <p>Response headers</p> <pre>access-control-allow-origin: * connection: keep-alive content-length: 114 content-type: application/json; charset=utf-8 date: Sat, 10 Jan 2026 08:18:56 GMT etag: W/"772-5e98xtZM01df1fx2r800ut3L+8" keep-alive: timeout=5 x-powered-by: Express</pre>

Responses

Mini E-Commerce Project: Step-by-Step Setup & Run

Install prerequisites

1. Install Node.js (v18+ recommended) from nodejs.org.
2. Install MySQL and start the server.
3. Install Git to clone the project.

Create the database

1. Open MySQL (MySQL Workbench, phpMyAdmin, or terminal).
2. Create a database called activity9_db:

```
CREATE DATABASE mini_ecommerce;
```

3. Make sure username & password in backend/src/app.module.ts match your MySQL credentials:

```
username: 'root',  
password: '',  
database: 'activity9_db',
```

Set up the backend

1. Open a terminal in the backend folder.
2. Install dependencies:

```
npm install
```

3. Start the backend server:

```
npm run start:dev
```

4. You should see:

```
[Nest] ... - Nest application is running on: http://localhost:3000
```

5. Open Swagger API docs in your browser:

```
http://localhost:3000/api
```

Here you can test all Products, Cart, and Orders endpoints.

Set up the frontend

1. Open a new terminal in the frontend folder.
2. Install dependencies:

```
npm install
```

3. Start the frontend:

`npm start`

4. Your browser will open at:

`http://localhost:3001`

You can now browse products, add to cart, and checkout orders.

Test the system

1. **Products:**
 - Add, edit, or delete products via **Swagger**.
2. **Cart:**
 - Add items to cart, change quantity, or remove items.
3. **Orders:**
 - Checkout cart items to create orders.
 - Update or delete orders; stock will adjust automatically.

Notes

- Cart is in-memory, so it resets if the backend restarts.
- Orders are saved in MySQL.
- TypeORM synchronize: true auto-creates tables for you.
- Swagger is very useful for testing APIs without the frontend.

`localhost:3000/api#`