Faculty of Computer Science

CS425

Website by Microservices

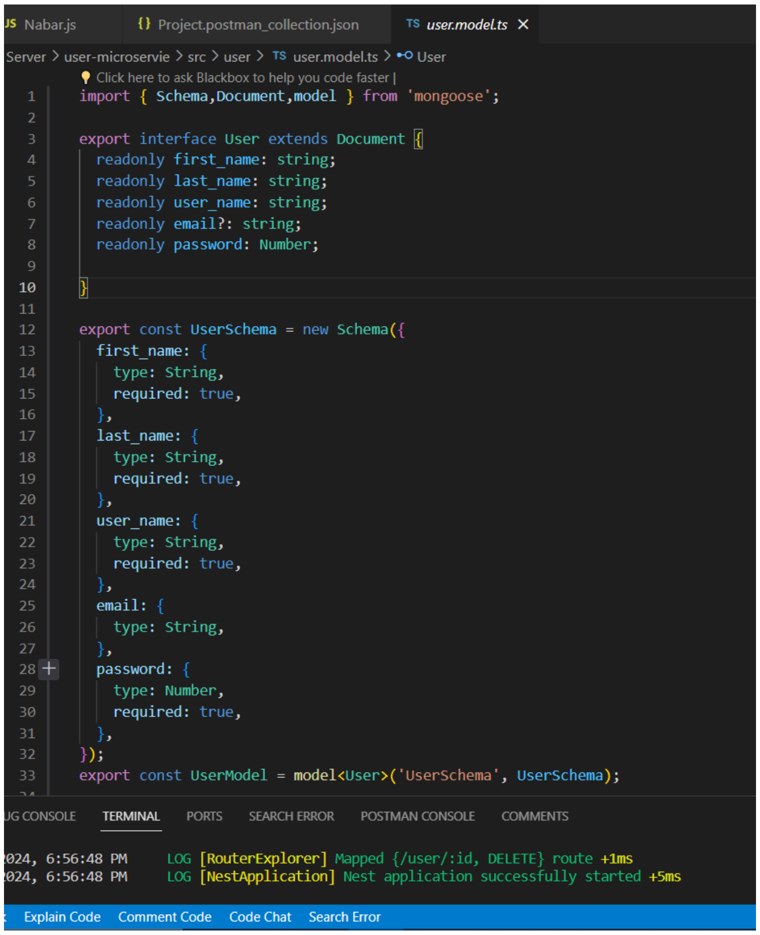
(3 microservices, React.js, Nodejs, Postman)

213673	Rowan Abdelhafiz
213597	Meray Ashraf
210165	Ahmed Adel

❖ User Microservices

- Signup/Login Microservice Schema

https://schema.getpostman.com/json/collection/v2.1.0/collection.json


JS Nabar.js
{} Project.postman_collection.json
TS user.model.ts ×
Server > user-microservie > src > user > TS user.model.ts > User
Click here to ask Blackbox to help you code faster |

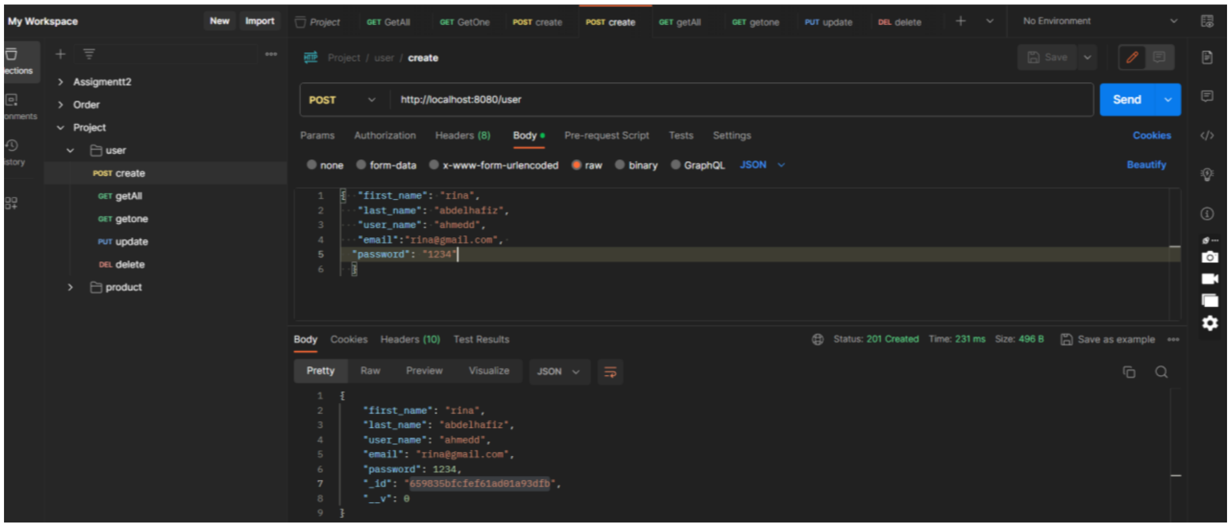
```typescript
import { Schema,Document,model } from 'mongoose';

export interface User extends Document {
  readonly first_name: string;
  readonly last_name: string;
  readonly user_name: string;
  readonly email?: string;
  readonly password: Number;

}

export const UserSchema = new Schema({
  first_name: {
    type: String,
    required: true,
  },
  last_name: {
    type: String,
    required: true,
  },
  user_name: {
    type: String,
    required: true,
  },
  email: {
    type: String,
  },
  password: {
    type: Number,
    required: true,
  },
});
export const UserModel = model<User>('UserSchema', UserSchema);
```

```
024, 6:56:48 PM     LOG [RouterExplorer] Mapped {/user/:id, DELETE} route +1ms
024, 6:56:48 PM     LOG [NestApplication] Nest application successfully started +5ms
```

Here the Signup Schema it Contains 'First name, Last name, Id, Email, password' All of these it's a type of string that can be taken from the user and we got it in the database by POST, GET, DELETE and PUT.

- Signup POSTMAN test.

POST



POST http://localhost:8080/user

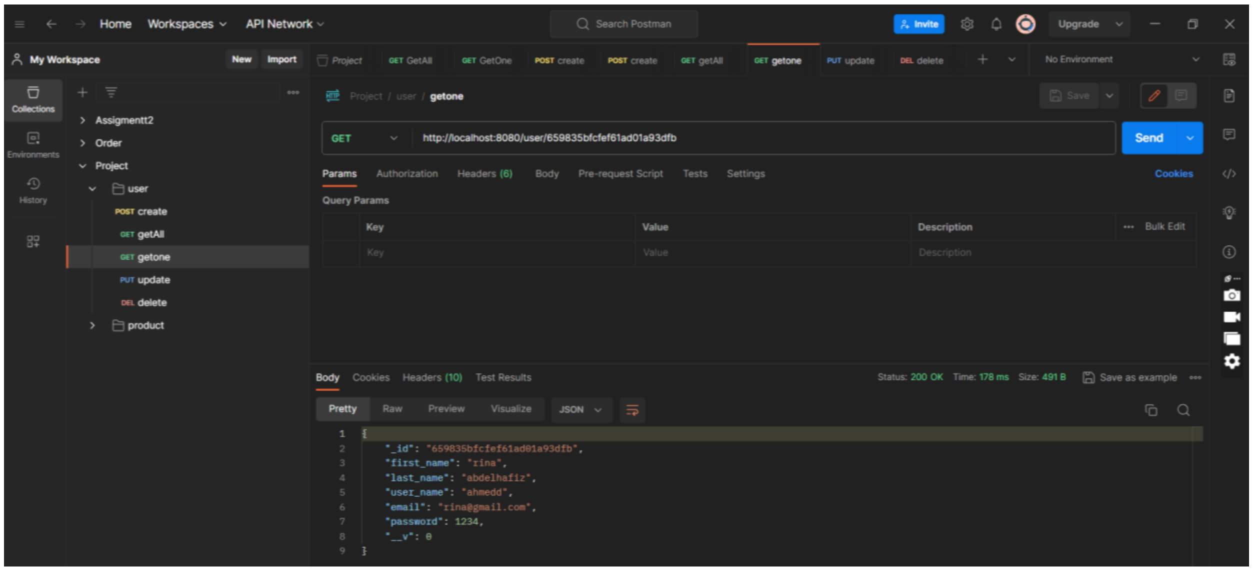
```
{ "first_name": "rina",
  "last_name": "abdelhafiz",
  "user_name": "ahmedd",
  "email":"rina@gmail.com",
 "password": "1234"
 }
```

Status: 201 Created Time: 231 ms Size: 496 B

```
{
    "first_name": "rina",
    "last_name": "abdelhafiz",
    "user_name": "ahmedd",
    "email": "rina@gmail.com",
    "password": 1234,
    "_id": "659835bfcfef61ad01a93dfb",
    "__v": 0
}
```

http://localhost:8080/user

- GET ONE by id



GET http://localhost:8080/user/659835bfcfef61ad01a93dfb

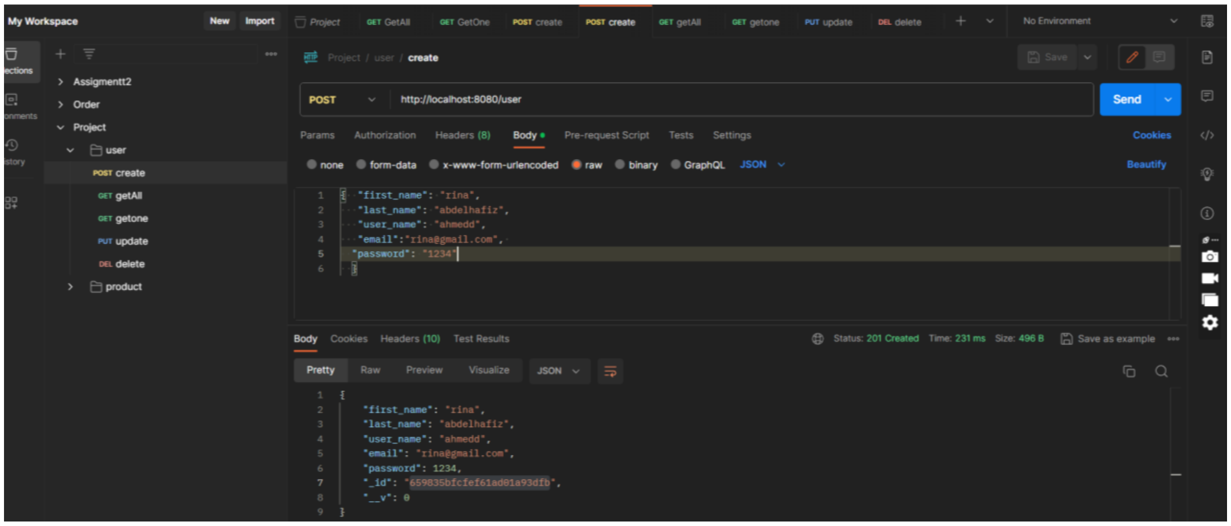
Status: 200 OK Time: 178 ms Size: 491 B

```
{
    "_id": "659835bfcfef61ad01a93dfb",
    "first_name": "rina",
    "last_name": "abdelhafiz",
    "user_name": "ahmedd",
    "email": "rina@gmail.com",
    "password": 1234,
    "__v": 0
}
```

http://localhost:8080/user/659835bfcfef61ad01a93dfb

- Signup POSTMAN test.

POST



POST http://localhost:8080/user

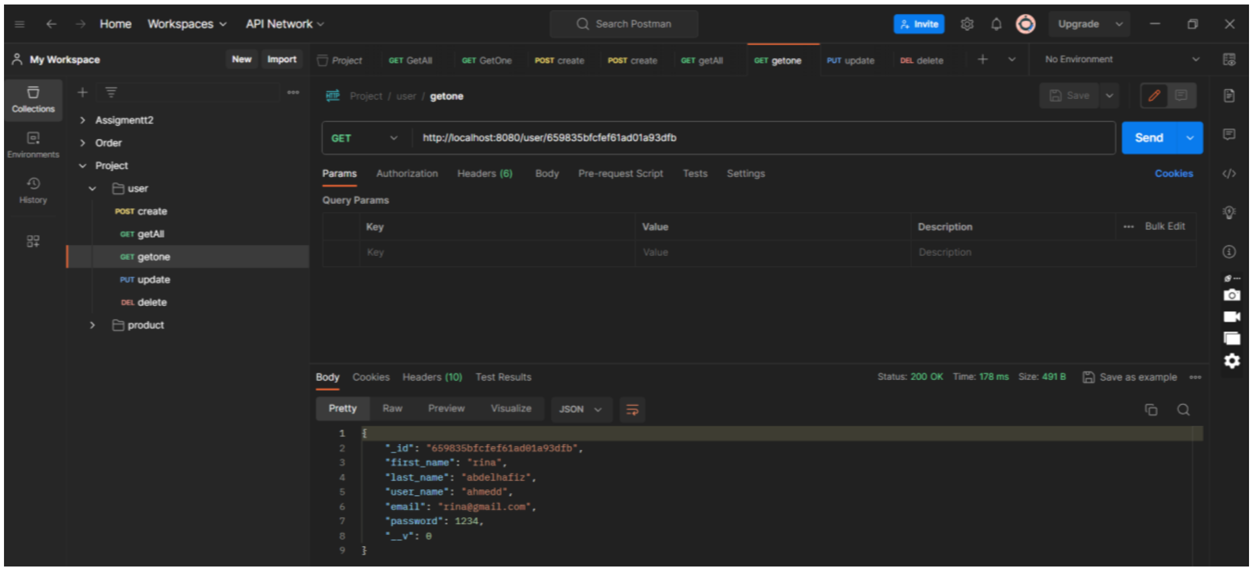
```
{ "first_name": "rina",
  "last_name": "abdelhafiz",
  "user_name": "ahmedd",
  "email":"rina@gmail.com",
 "password": "1234"
 }
```

Status: 201 Created Time: 231 ms Size: 496 B

```
{
    "first_name": "rina",
    "last_name": "abdelhafiz",
    "user_name": "ahmedd",
    "email": "rina@gmail.com",
    "password": 1234,
    "_id": "659835bfcfef61ad01a93dfb",
    "__v": 0
}
```

http://localhost:8080/user

- GET ONE by id



GET http://localhost:8080/user/659835bfcfef61ad01a93dfb

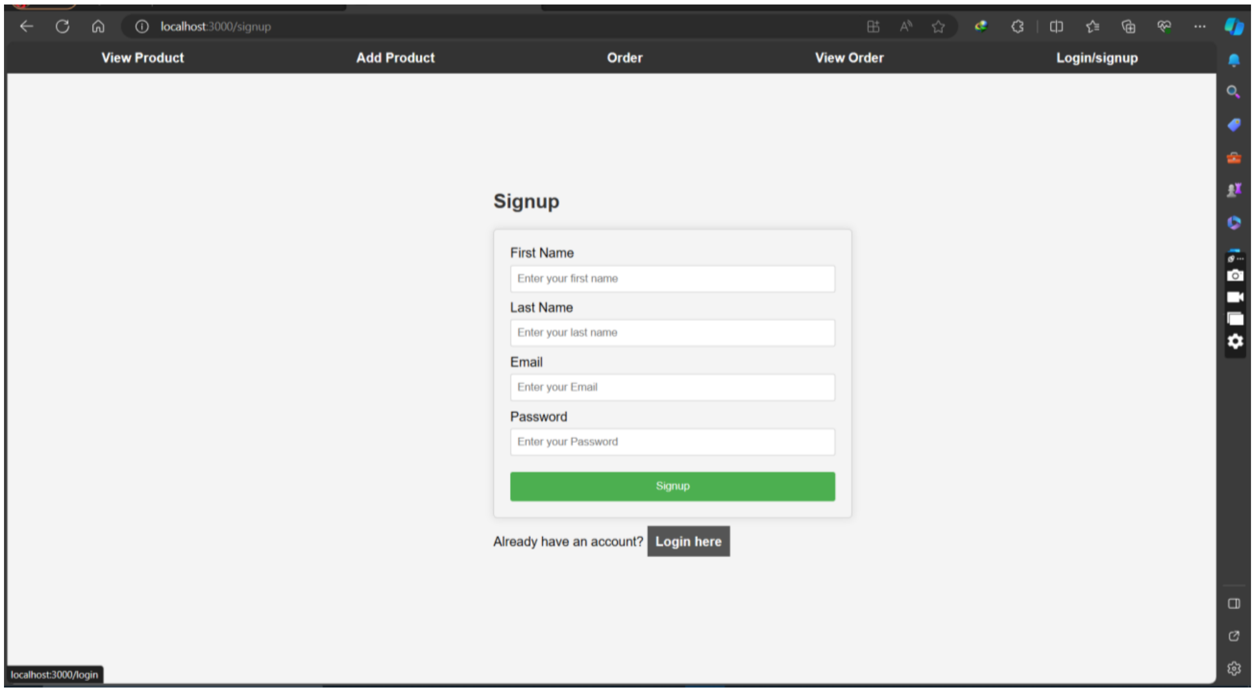
Status: 200 OK Time: 178 ms Size: 491 B

```
{
    "_id": "659835bfcfef61ad01a93dfb",
    "first_name": "rina",
    "last_name": "abdelhafiz",
    "user_name": "ahmedd",
    "email": "rina@gmail.com",
    "password": 1234,
    "__v": 0
}
```

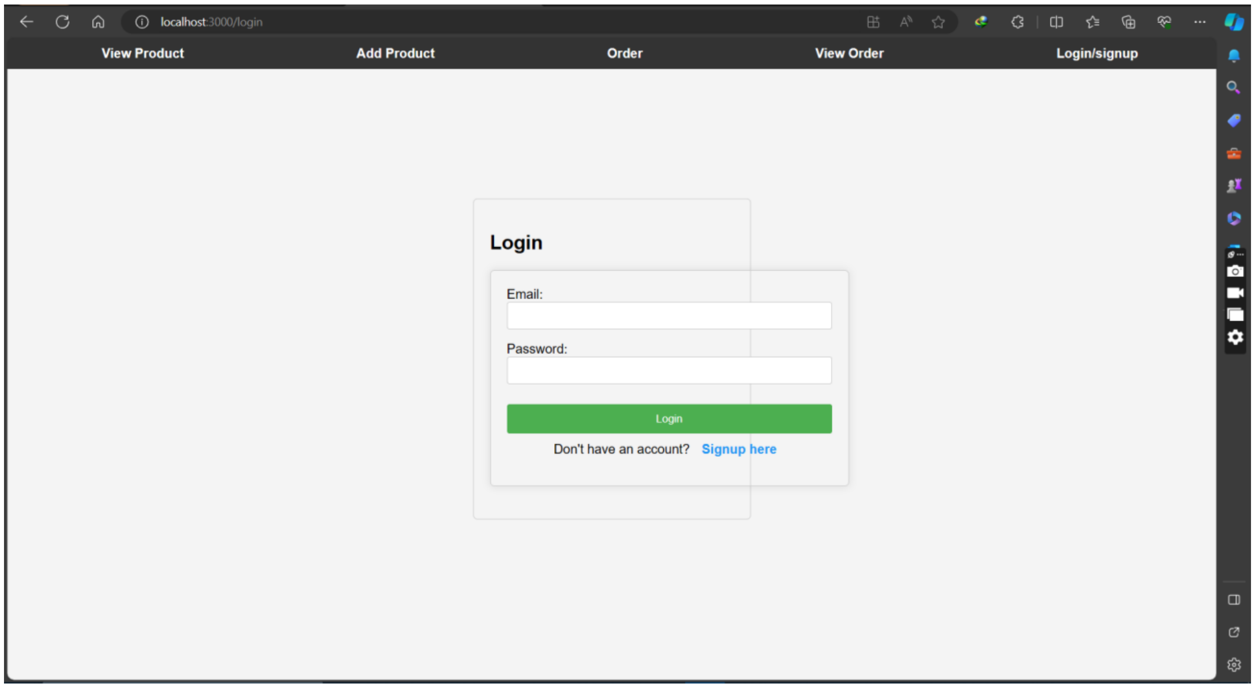
http://localhost:8080/user/659835bfcfef61ad01a93dfb

- **<u>React App (Frontend)</u>**

Sign-Up



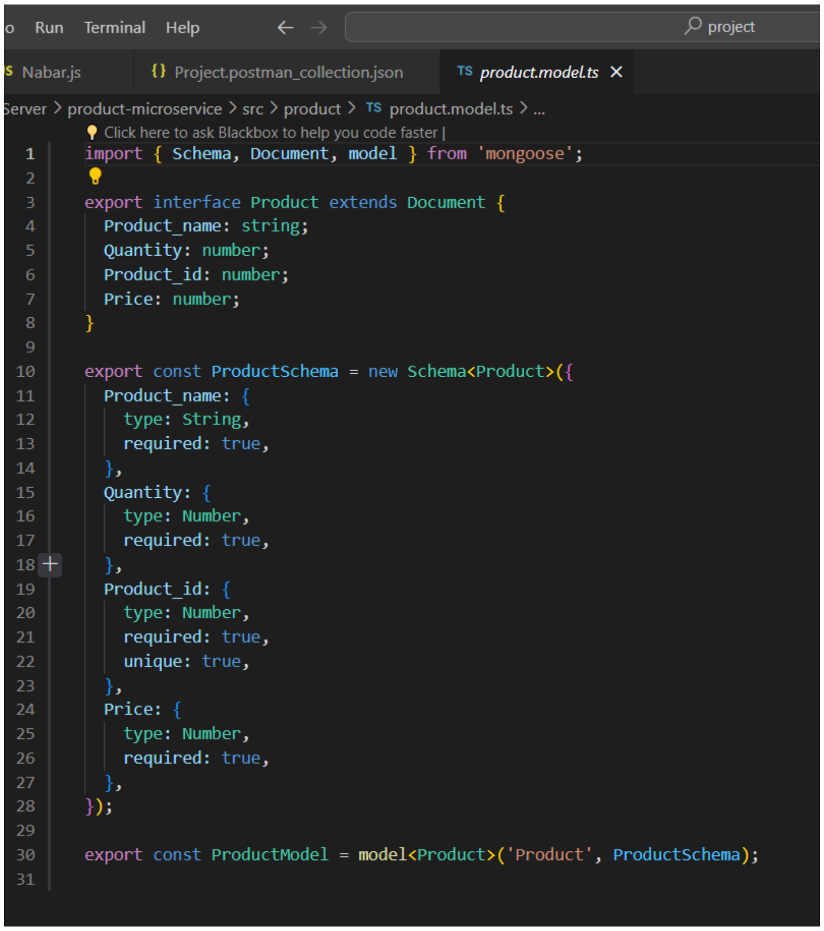
Login-In



❖ Product Microservices

• Product Microservice Schema

https://schema.getpostman.com/json/collection/v2.1.0/collection.json

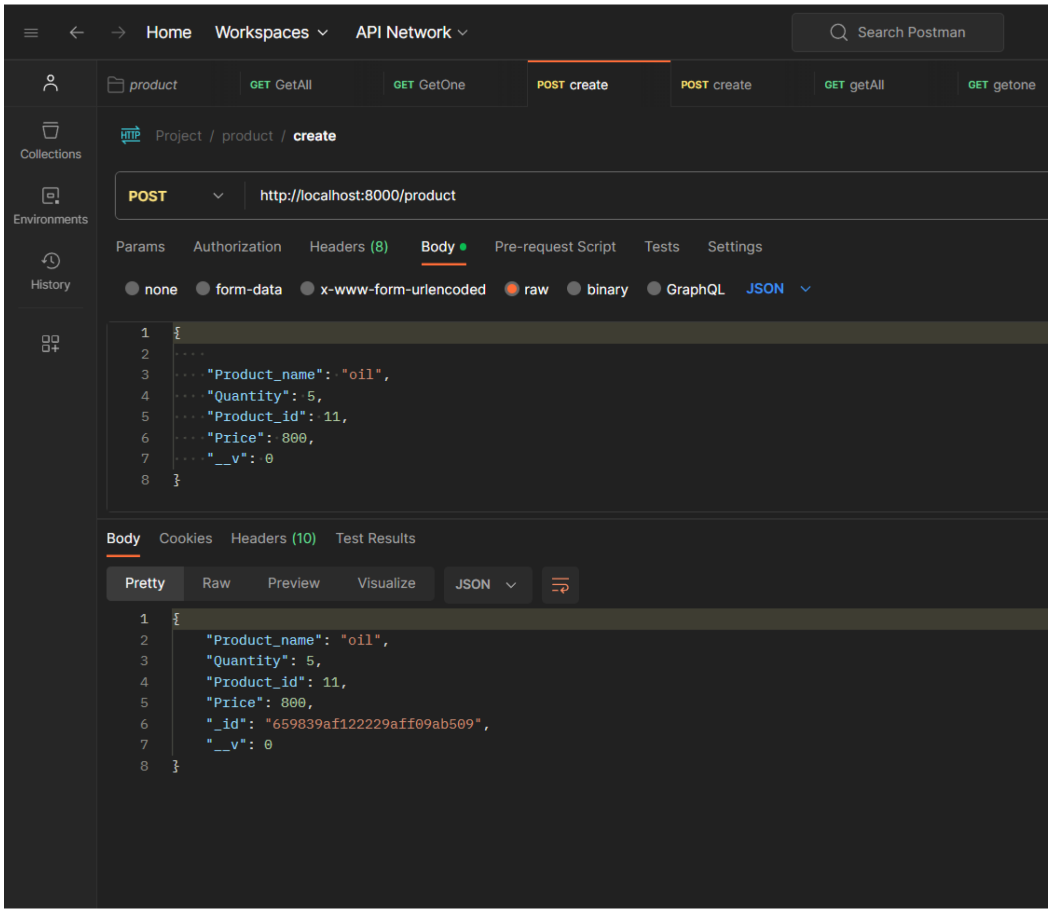

o Run Terminal Help
project
Nabar.js
Project.postman_collection.json
product.model.ts
erver > product-microservice > src > product > product.model.ts > ...
Click here to ask Blackbox to help you code faster

```
import { Schema, Document, model } from 'mongoose';

export interface Product extends Document {
  Product_name: string;
  Quantity: number;
  Product_id: number;
  Price: number;
}

export const ProductSchema = new Schema<Product>({
  Product_name: {
    type: String,
    required: true,
  },
  Quantity: {
    type: Number,
    required: true,
  },
  Product_id: {
    type: Number,
    required: true,
    unique: true,
  },
  Price: {
    type: Number,
    required: true,
  },
});

export const ProductModel = model<Product>('Product', ProductSchema);
```

Here the Product Schema it Contains 'Product name, Quantity, Product-id, Price All of these it's a type of string that can be taken from the user and we got it in the database by POST, GET, DELETE and PUT.

- Product POSTMAN test.

POST



POST http://localhost:8000/product

Body: [✓ raw] JSON

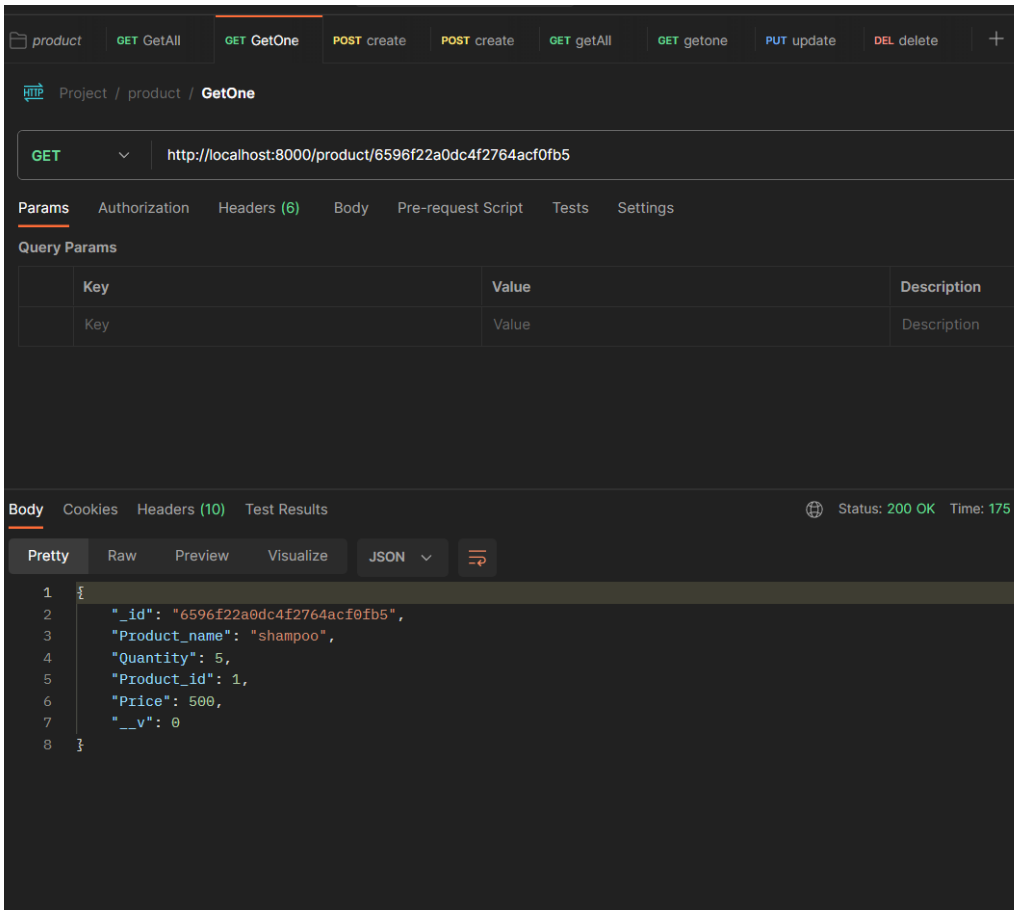
```
{

    "Product_name": "oil",
    "Quantity": 5,
    "Product_id": 11,
    "Price": 800,
    "__v": 0
}
```

Body: Pretty JSON

```
{
    "Product_name": "oil",
    "Quantity": 5,
    "Product_id": 11,
    "Price": 800,
    "_id": "659839af122229aff09ab509",
    "__v": 0
}
```

http://localhost:8000/product

- GET ONE by id


product
GET GetAll
GET GetOne
POST create
POST create
GET getAll
GET getone
PUT update
DEL delete
Project / product / GetOne
Params Authorization Headers (6) Body Pre-request Script Tests Settings
Body Cookies Headers (10) Test Results
Status: 200 OK Time: 175
Pretty Raw Preview Visualize JSON

GET http://localhost:8000/product/6596f22a0dc4f2764acf0fb5

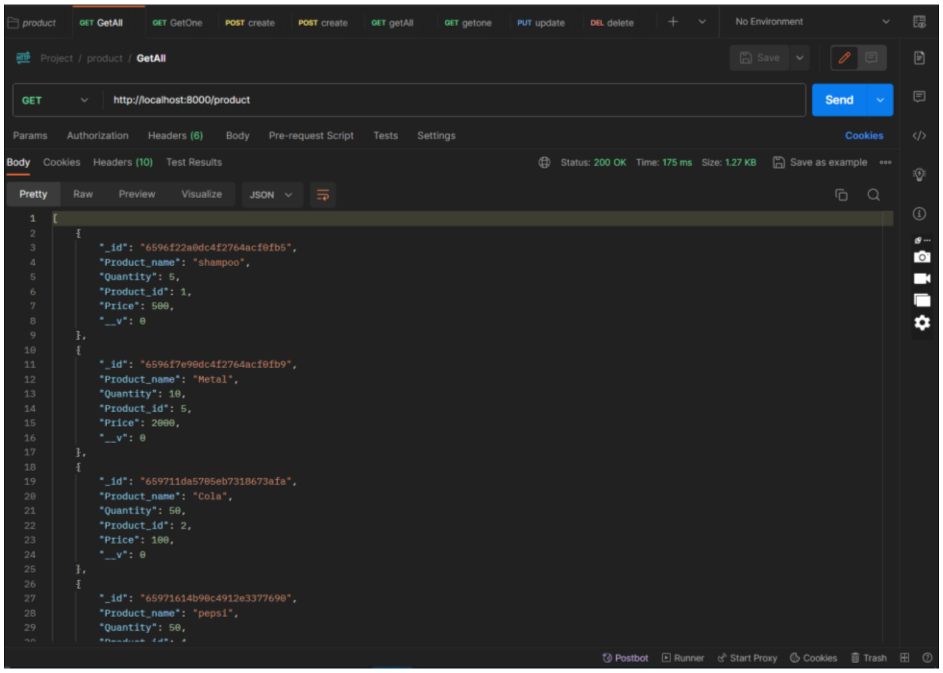
Query Params

	Key	Value	Description
	Key	Value	Description

```
{
    "_id": "6596f22a0dc4f2764acf0fb5",
    "Product_name": "shampoo",
    "Quantity": 5,
    "Product_id": 1,
    "Price": 500,
    "__v": 0
}
```

http://localhost:8000/product/6596f22a0dc4f2764acf0fb5

- GET ALL



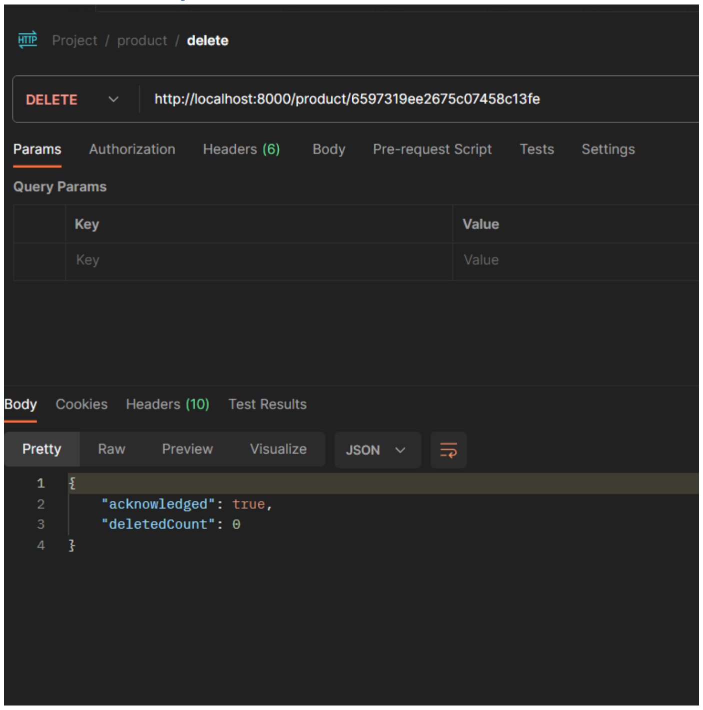
Status: 200 OK Time: 175 ms Size: 1.27 KB

GET http://localhost:8000/product

```
[
    {
        "_id": "6596f22a0dc4f2764acf0fb5",
        "Product_name": "shampoo",
        "Quantity": 5,
        "Product_id": 1,
        "Price": 500,
        "__v": 0
    },
    {
        "_id": "6596f7e90dc4f2764acf0fb9",
        "Product_name": "Metal",
        "Quantity": 10,
        "Product_id": 5,
        "Price": 2000,
        "__v": 0
    },
    {
        "_id": "659711da5705eb7318673afa",
        "Product_name": "Cola",
        "Quantity": 50,
        "Product_id": 2,
        "Price": 100,
        "__v": 0
    },
    {
        "_id": "65971614b90c4912e3377690",
        "Product_name": "pepsi",
        "Quantity": 50,
```

http://localhost:8000/product

- DELETE by id

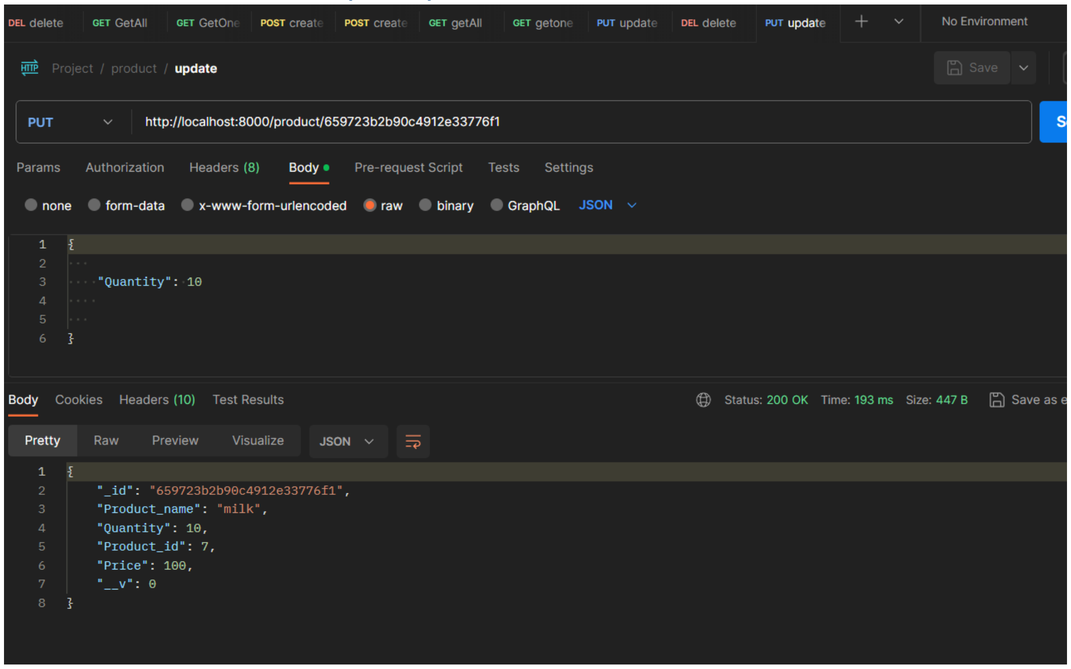


DELETE http://localhost:8000/product/6597319ee2675c07458c13fe

```
{
    "acknowledged": true,
    "deletedCount": 0
}
```

http://localhost:8000/product/6597319ee2675c07458c13fe

- UPDATE(PUT)



PUT http://localhost:8000/product/659723b2b90c4912e33776f1

```
{

    "Quantity": 10


}
```

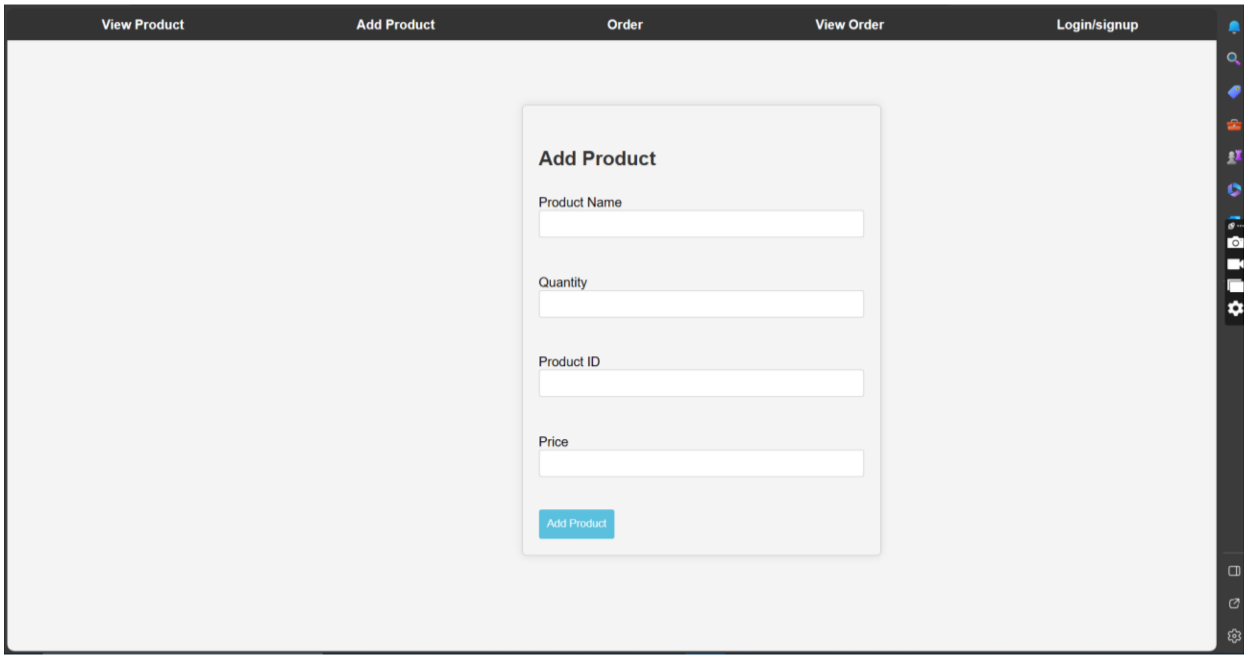
Status: 200 OK Time: 193 ms Size: 447 B

```
{
    "_id": "659723b2b90c4912e33776f1",
    "Product_name": "milk",
    "Quantity": 10,
    "Product_id": 7,
    "Price": 100,
    "__v": 0
}
```

http://localhost:8000/product/659723b2b90c4912e33776f1

- <u>**React App (Frontend)**</u>

Add Product



Add Product

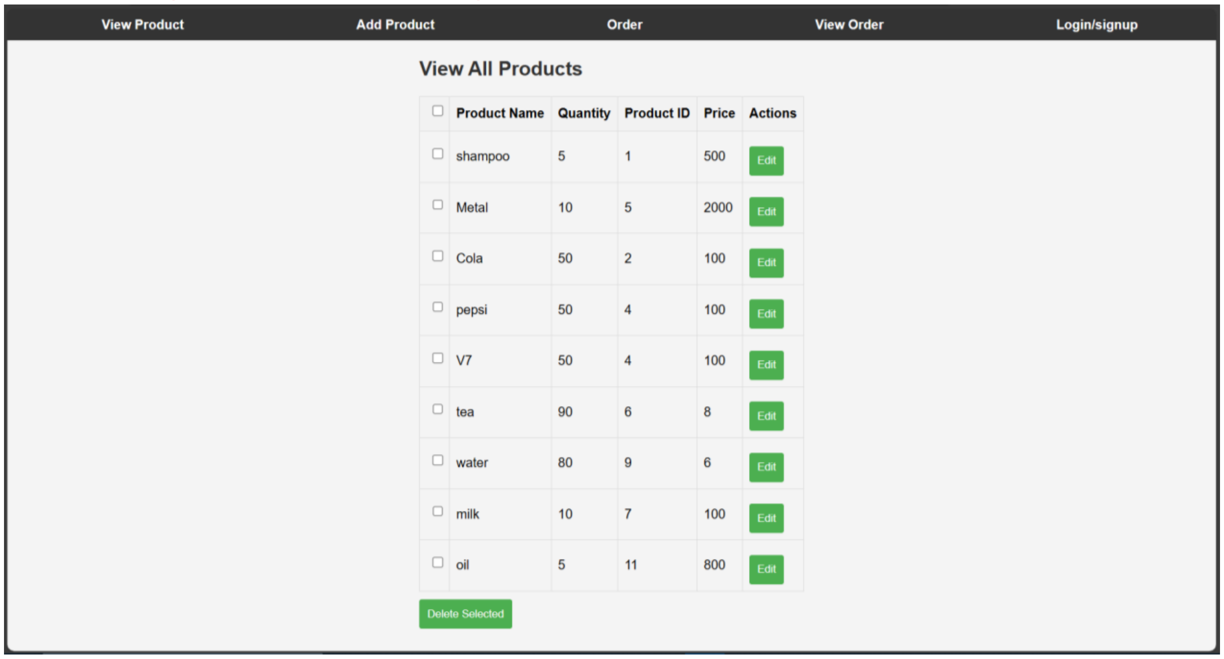
Product Name

Quantity

Product ID

Price

View/update/delete product

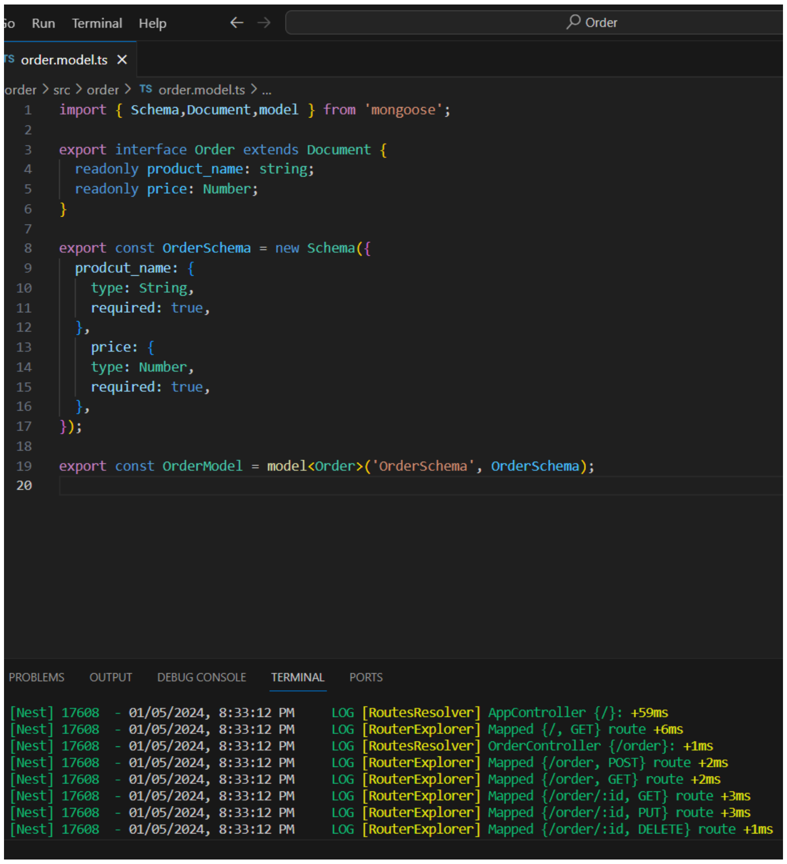


View All Products

☐	Product Name	Quantity	Product ID	Price	Actions
☐	shampoo	5	1	500	Edit
☐	Metal	10	5	2000	Edit
☐	Cola	50	2	100	Edit
☐	pepsi	50	4	100	Edit
☐	V7	50	4	100	Edit
☐	tea	90	6	8	Edit
☐	water	80	9	6	Edit
☐	milk	10	7	100	Edit
☐	oil	5	11	800	Edit

❖Order Microservices

- ### Order Microservice Schema



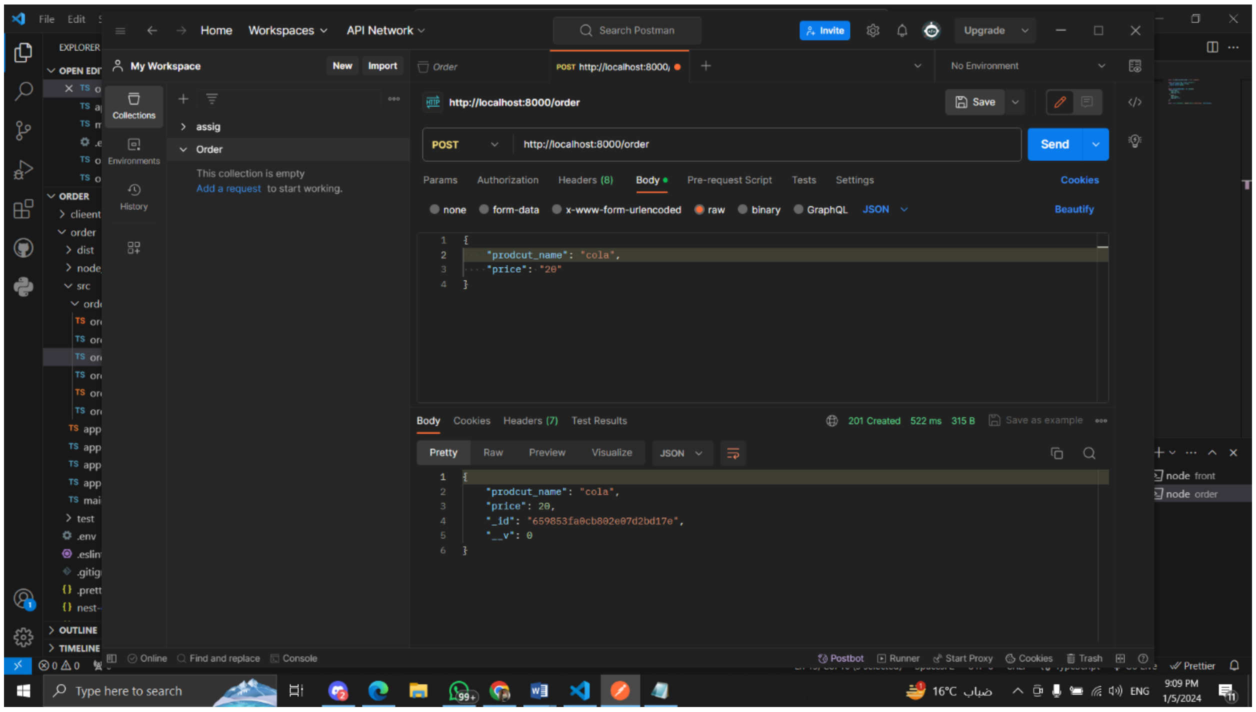
```typescript
import { Schema,Document,model } from 'mongoose';

export interface Order extends Document {
  readonly product_name: string;
  readonly price: Number;
}

export const OrderSchema = new Schema({
  prodcut_name: {
    type: String,
    required: true,
  },
    price: {
    type: Number,
    required: true,
  },
});

export const OrderModel = model<Order>('OrderSchema', OrderSchema);
```

```
[Nest] 17608  - 01/05/2024, 8:33:12 PM     LOG [RoutesResolver] AppController {/}: +59ms
[Nest] 17608  - 01/05/2024, 8:33:12 PM     LOG [RouterExplorer] Mapped {/, GET} route +6ms
[Nest] 17608  - 01/05/2024, 8:33:12 PM     LOG [RoutesResolver] OrderController {/order}: +1ms
[Nest] 17608  - 01/05/2024, 8:33:12 PM     LOG [RouterExplorer] Mapped {/order, POST} route +2ms
[Nest] 17608  - 01/05/2024, 8:33:12 PM     LOG [RouterExplorer] Mapped {/order, GET} route +2ms
[Nest] 17608  - 01/05/2024, 8:33:12 PM     LOG [RouterExplorer] Mapped {/order/:id, GET} route +3ms
[Nest] 17608  - 01/05/2024, 8:33:12 PM     LOG [RouterExplorer] Mapped {/order/:id, PUT} route +3ms
[Nest] 17608  - 01/05/2024, 8:33:12 PM     LOG [RouterExplorer] Mapped {/order/:id, DELETE} route +1ms
```

Here the Order Schema it Contains 'Product name, Price All of these it's a type of string that can be taken from the user and we got it in the database by POST, GET, DELETE and PUT.

- Order POSTMAN test.

POST



POST http://localhost:8000/order

Body (raw, JSON):

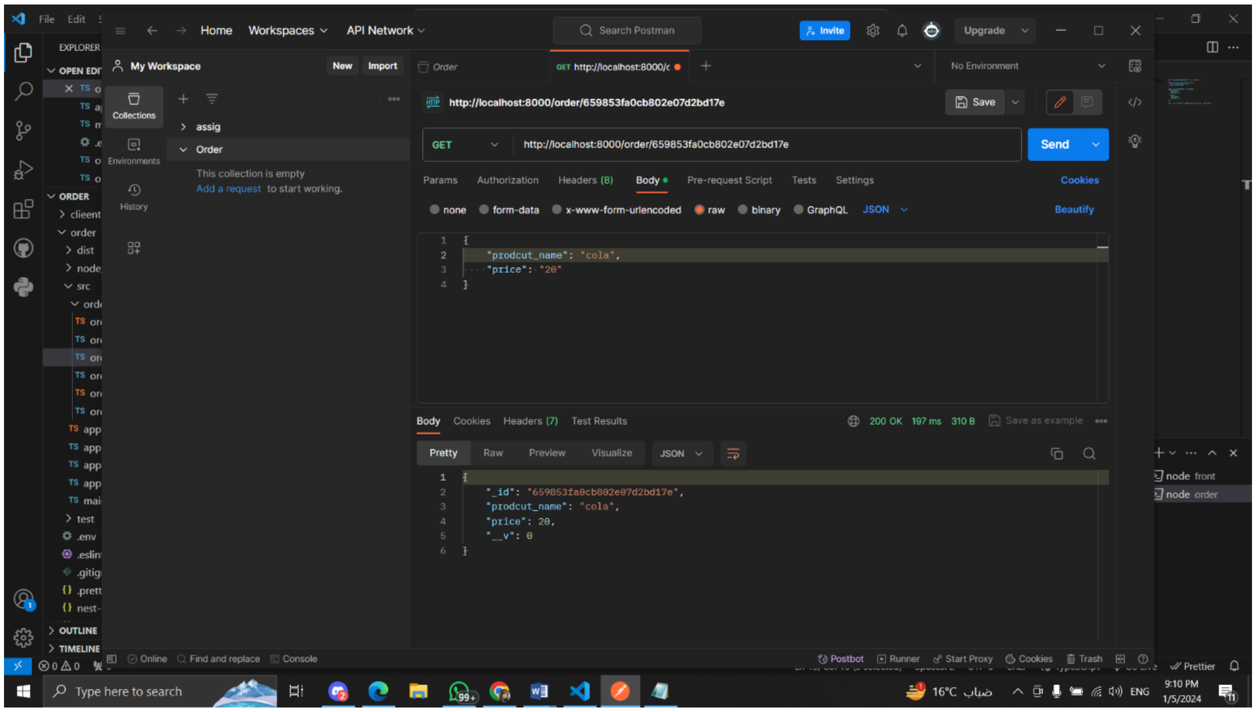
```
{
    "prodcut_name": "cola",
    "price": "20"
}
```

201 Created 522 ms 315 B

```
{
    "prodcut_name": "cola",
    "price": 20,
    "_id": "659853fa0cb802e07d2bd17e",
    "__v": 0
}
```

http://localhost:8000/order

- GET ALL



GET http://localhost:8000/order/659853fa0cb802e07d2bd17e

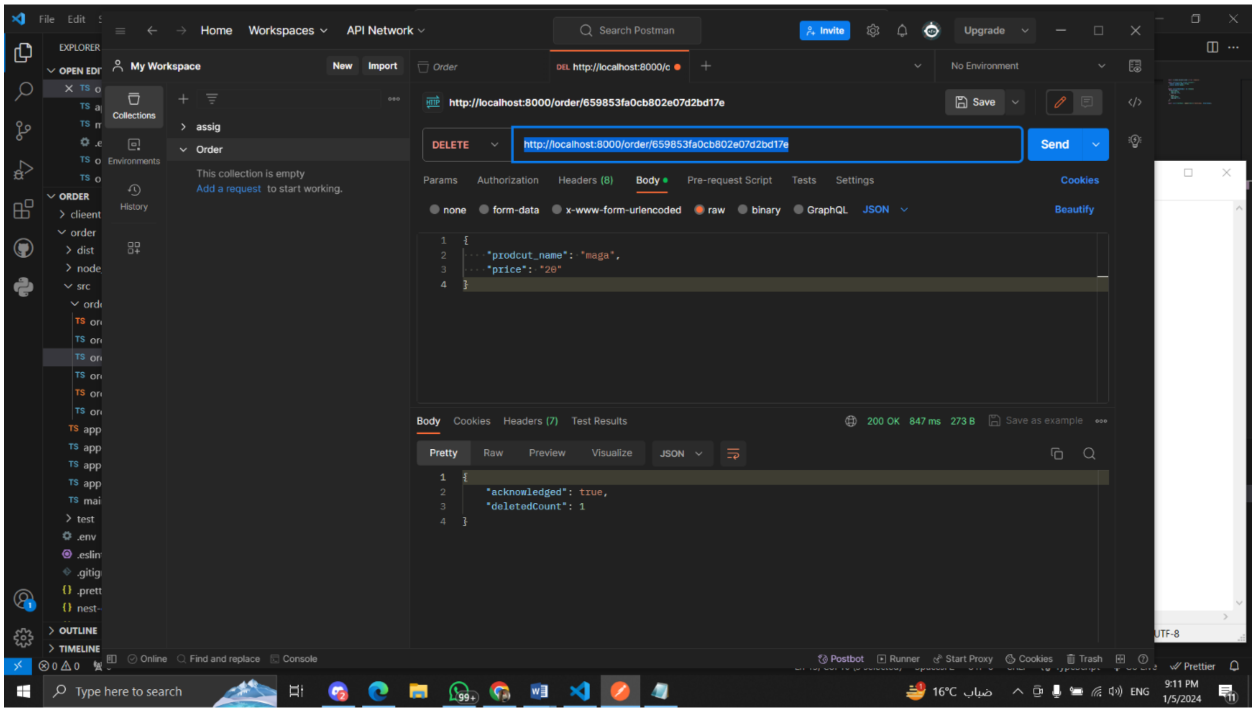
```
{
    "prodcut_name": "cola",
    "price": "20"
}
```

200 OK 197 ms 310 B

```
{
    "_id": "659853fa0cb802e07d2bd17e",
    "prodcut_name": "cola",
    "price": 20,
    "__v": 0
}
```

http://localhost:8000/order/659853fa0cb802e07d2bd17e

- DELETE by id



This collection is empty
Add a request to start working.

http://localhost:8000/order/659853fa0cb802e07d2bd17e

DELETE http://localhost:8000/order/659853fa0cb802e07d2bd17e

none form-data x-www-form-urlencoded raw binary GraphQL JSON

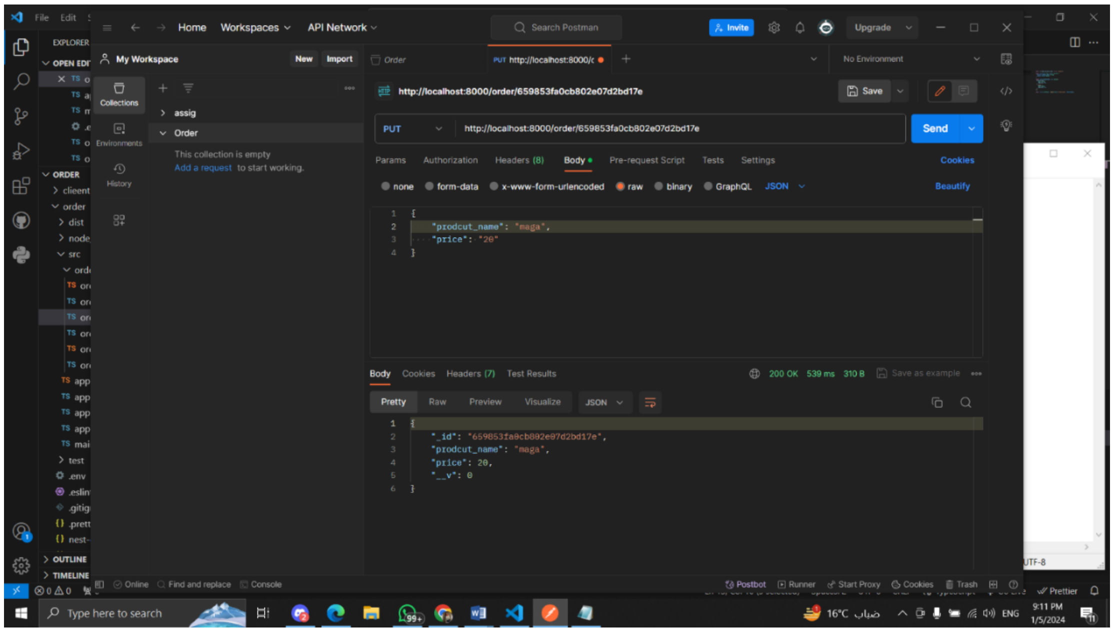
```
{
    "prodcut_name": "maga",
    "price": "20"
}
```

Body Cookies Headers (7) Test Results 200 OK 847 ms 273 B Save as example

```
{
    "acknowledged": true,
    "deletedCount": 1
}
```

http://localhost:8000/order/659853fa0cb802e07d2bd17e

- UPDATE(PUT)



This collection is empty
Add a request to start working.

http://localhost:8000/order/659853fa0cb802e07d2bd17e

PUT http://localhost:8000/order/659853fa0cb802e07d2bd17e

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
{
    "prodcut_name": "maga",
    "price": "20"
}
```

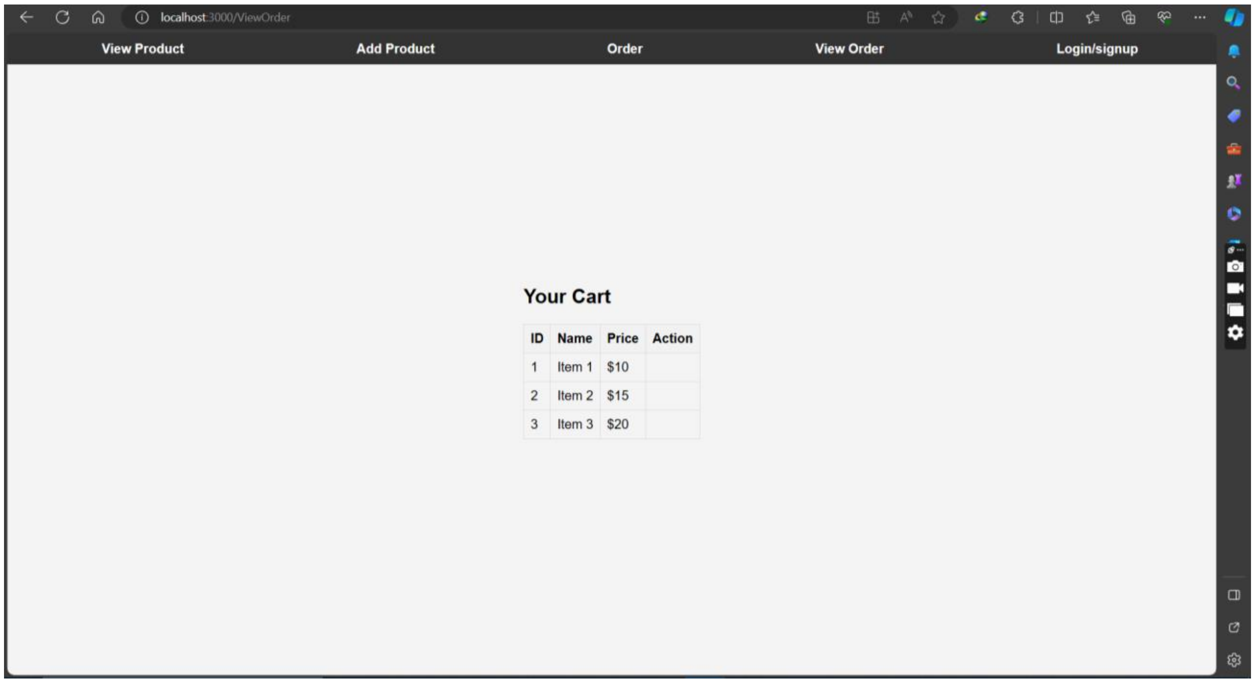
Body Cookies Headers (7) Test Results 200 OK 539 ms 310 B Save as example

```
{
    "_id": "659853fa0cb802e07d2bd17e",
    "prodcut_name": "maga",
    "price": 20,
    "__v": 0
}
```

http://localhost:8000/order/659853fa0cb802e07d2bd17e

❖ **React App (Frontend)**

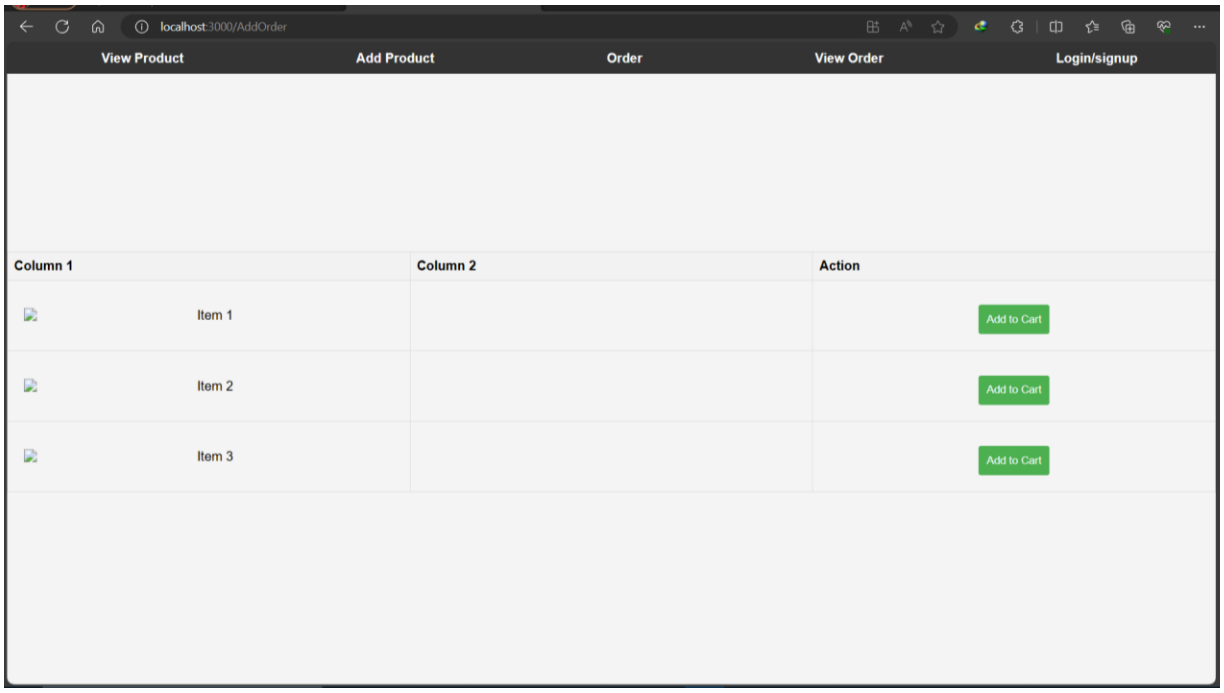
View Cart



Your Cart

ID	Name	Price	Action
1	Item 1	$10	
2	Item 2	$15	
3	Item 3	$20	

Order



Column 1	Column 2	Action
Item 1		Add to Cart
Item 2		Add to Cart
Item 3		Add to Cart

❖ **Apps we used:**

- **Visual studio code** (React.js For Frontend, Nest.js For microservice)
- **Postman** (For test API)

❖ **GitHub Link:**
https://github.com/rowanabdelhafiz/cs425