

Make an E-commerce Website for Sporty Shoes .

Developer name: Neelkanth Tripathi

Application features:

- Sign up to the website.
- Sign in page for both user and admin.
- Edit username and password.
- Purchase a product.
- View a list of all products. • Search a product by name.
- View details of a product.
- View account details.
- View orders.
- Admin password change.

Core concepts that were used in the project are:

- Database
- Collections
- Exception Handling

Technologies that were used in the project are:

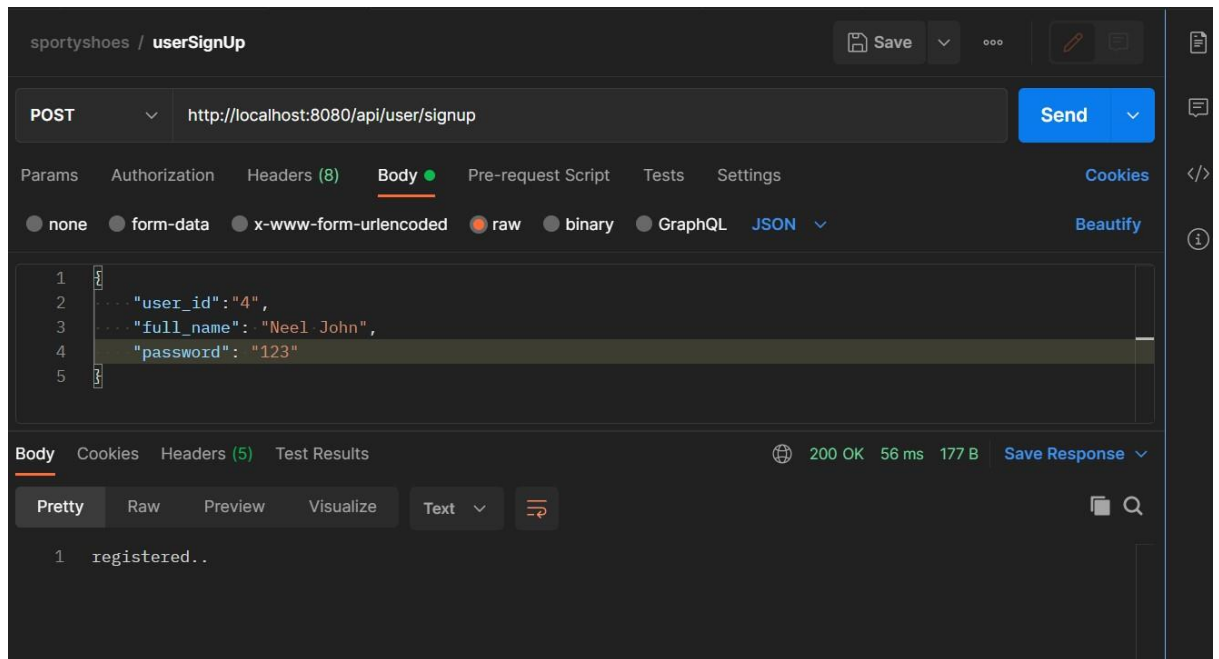
- Spring boot.
- Handling exceptions.
- Spring boot CRUD Api
- JDBC
- Java classes
- MySQL

- Admin views list of all users.
- Admin views details of user.
- Admin searches user by name.
- Admin views all the products.
- Admin adds a new product to website.
- Admin views details of a product.
- Admin updates product details.
- Admin deletes product.
- Admin views all orders sorted by order date and order id ascending and descending.
- Admin views all orders for a specific product.
- Admin views all orders for a specific user.
- Close the application at any time.

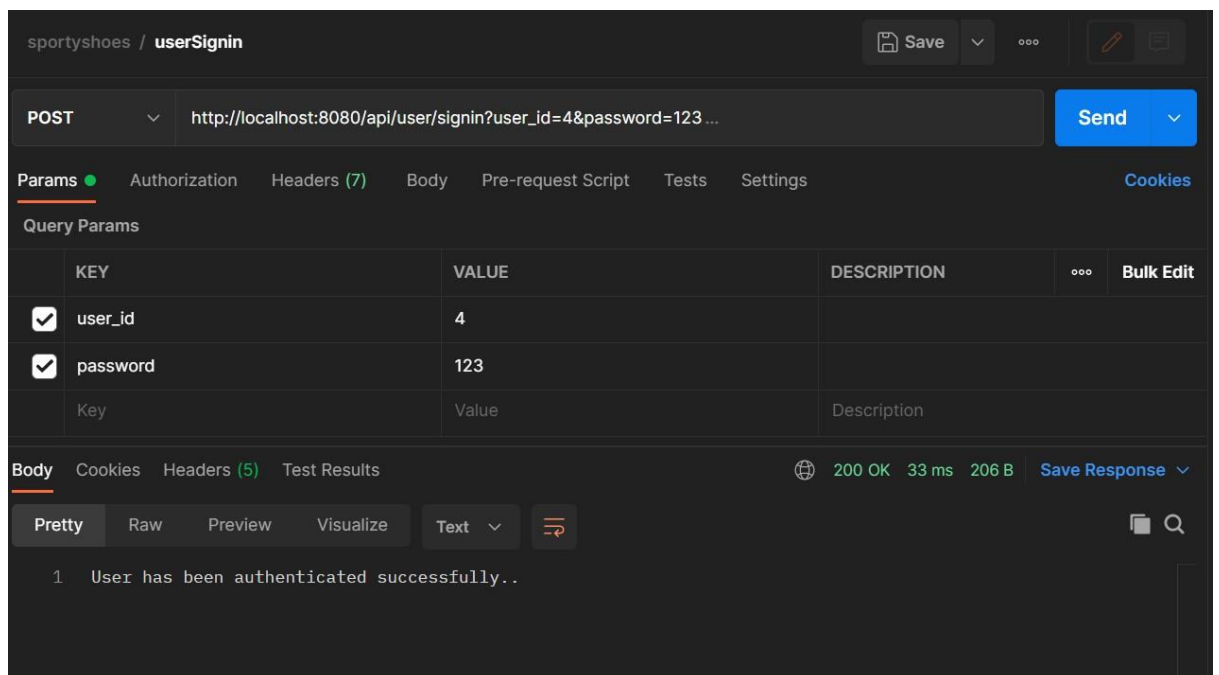
The application was done in four sprints, the first sprint for sprint planning and preparing. Next, the remaining three sprints for developing and testing the application. Moving on to the first sprint we planned the flow of the development and how we are approaching the project.

Furthermore, we developed the database tables and entered products, users' information and orders details. Then, we created the packages and java classes for each table in the database. Moving on, to the developing the repository packages to link the classes and the database code. Then, we created the services and rest api controllers to get to the code. Finally, we tested the website using postman.

The above screenshot displays the welcome page.



The user will signup via entering the above details.



The user will enter his/her credentials to log in.

The screenshot shows a REST client interface for a request named `userSignIn`. The request is a `POST` to `http://localhost:8080/api/user/signin?user_id=4&password=1234 ...`. The `Params` tab is active, showing query parameters:

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	<code>user_id</code>	<code>4</code>	
<input checked="" type="checkbox"/>	<code>password</code>	<code>1234</code>	

The `Body` tab is also active, showing the response status `200 OK` with a message: `1 Incorrect credentials..`

If incorrect credentials are entered the above message will be displayed.

The screenshot shows a REST client interface for a request named `adminSignIn`. The request is a `POST` to `http://localhost:8080/api/admin/signin?admin_id=1&password=admin ...`. The `Params` tab is active, showing query parameters:

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	<code>admin_id</code>	<code>1</code>	
<input checked="" type="checkbox"/>	<code>password</code>	<code>admin</code>	

The `Body` tab is also active, showing the response status `200 OK` with a message: `1 User has been authenticated successfully..`

List of all users.

sportyshoes / **getUserByName** Save ... Edit Copy

GET ▼ http://localhost:8080/api/user/search?name=Fatima Salam Send ▼

Params ● Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

<input checked="" type="checkbox"/>	name	Fatima Salam	
	Key	Value	Description

Body Cookies Headers (5) Test Results 200 OK 24 ms 220 B Save Response ▼

Pretty Raw Preview Visualize JSON ▼ ...

```
1 {
2   "user_id": 1,
3   "full_name": "Fatima Salam",
4   "password": null
5 }
```

Search user by name.

sportyshoes / **getUserById** Save ... Edit Copy

GET ▼ http://localhost:8080/api/user/all/1 Send ▼

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

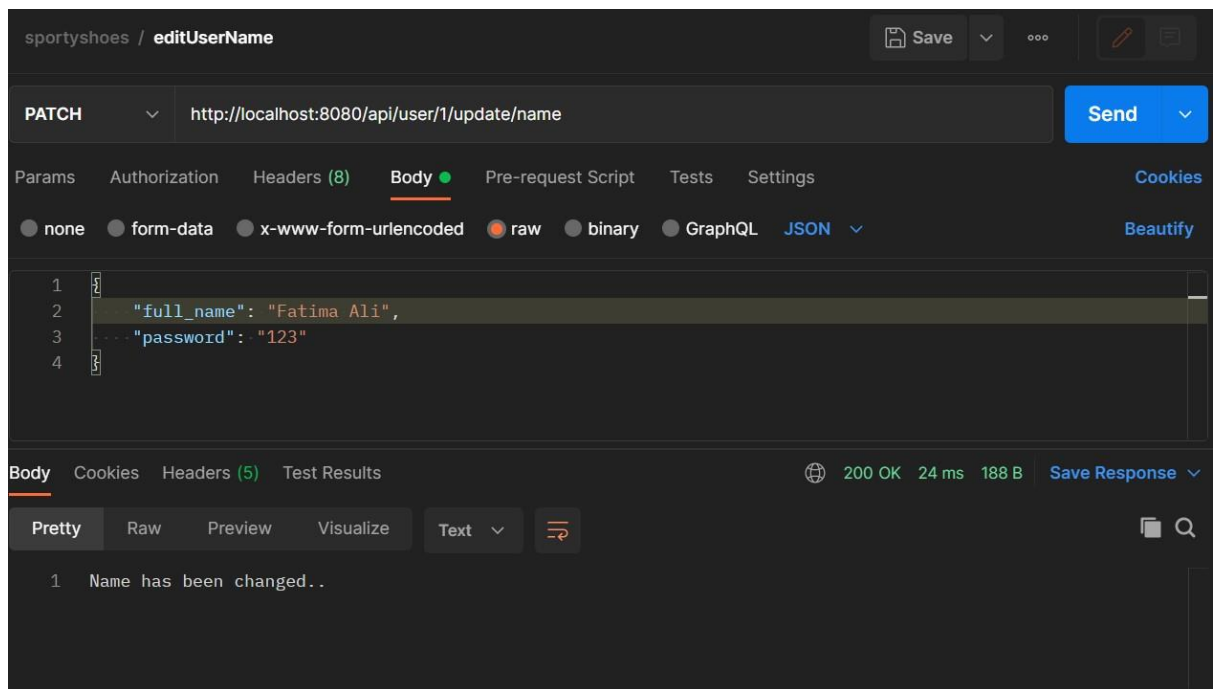
	KEY	VALUE	DESCRIPTION	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (5) Test Results 200 OK 37 ms 222 B Save Response ▼

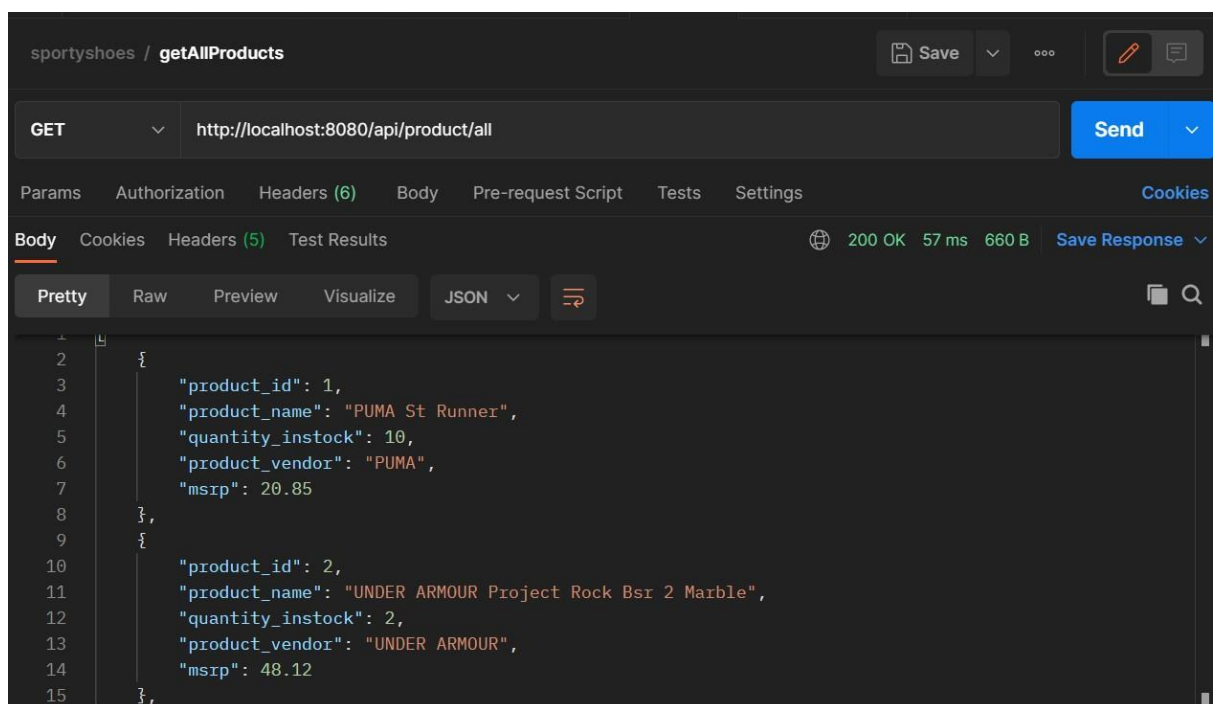
Pretty Raw Preview Visualize JSON ▼ ...

```
1 {
2   "user_id": 1,
3   "full_name": "Fatima Salam",
4   "password": "test"
5 }
```

Find user by user_id.



Edit user name.



Viewing all products.

sportyshoes / **getProductById** Save ... ✎ 💬

GET ▼ http://localhost:8080/api/product/all/1 Send ▼

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (5) Test Results 🌐 200 OK 19 ms 271 B Save Response ▼

Pretty Raw Preview Visualize JSON ▼ ↔ 📄 🔍

```
1  {
2    "product_id": 1,
3    "product_name": "PUMA St Runner",
4    "quantity_instock": 10,
5    "product_vendor": "PUMA",
6    "msrp": 20.85
7  }
```

Viewing all products by product_id.

sportyshoes / **getProductByName** Save ... ✎ 💬

GET ▼ http://localhost:8080/api/product/search?name=PUMA St Runner Send ▼

Params ● Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

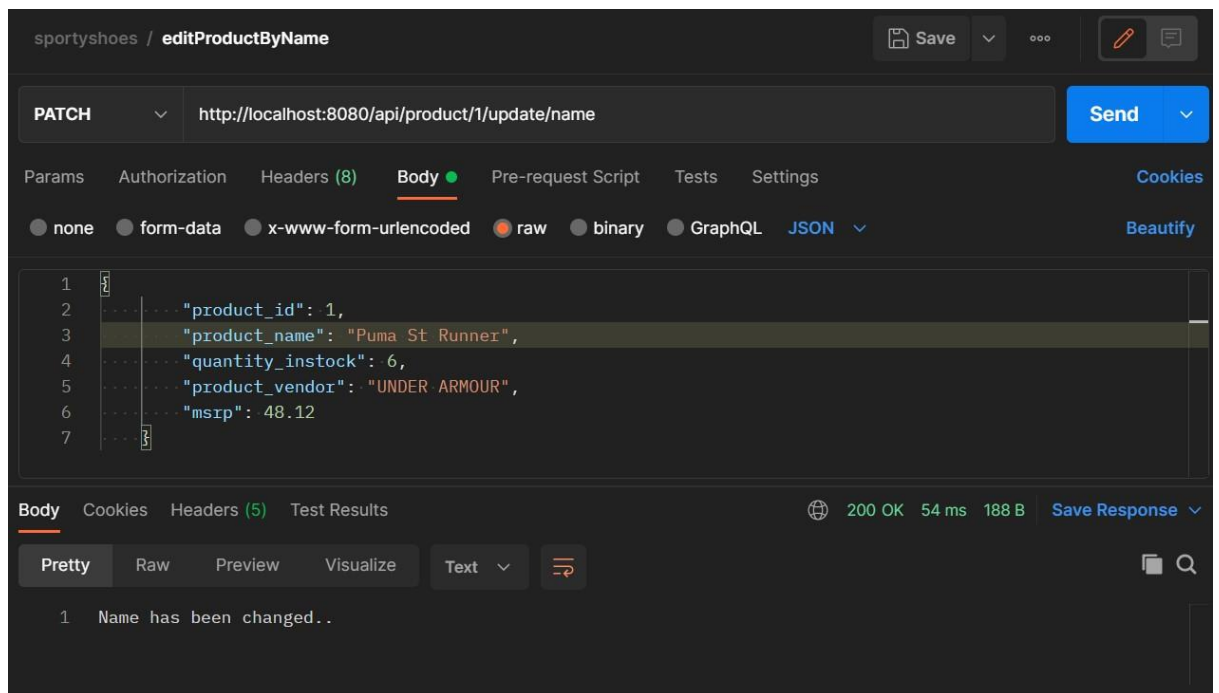
<input checked="" type="checkbox"/>	name	PUMA St Runner	
	Key	Value	Description

Body Cookies Headers (5) Test Results 🌐 200 OK 16 ms 271 B Save Response ▼

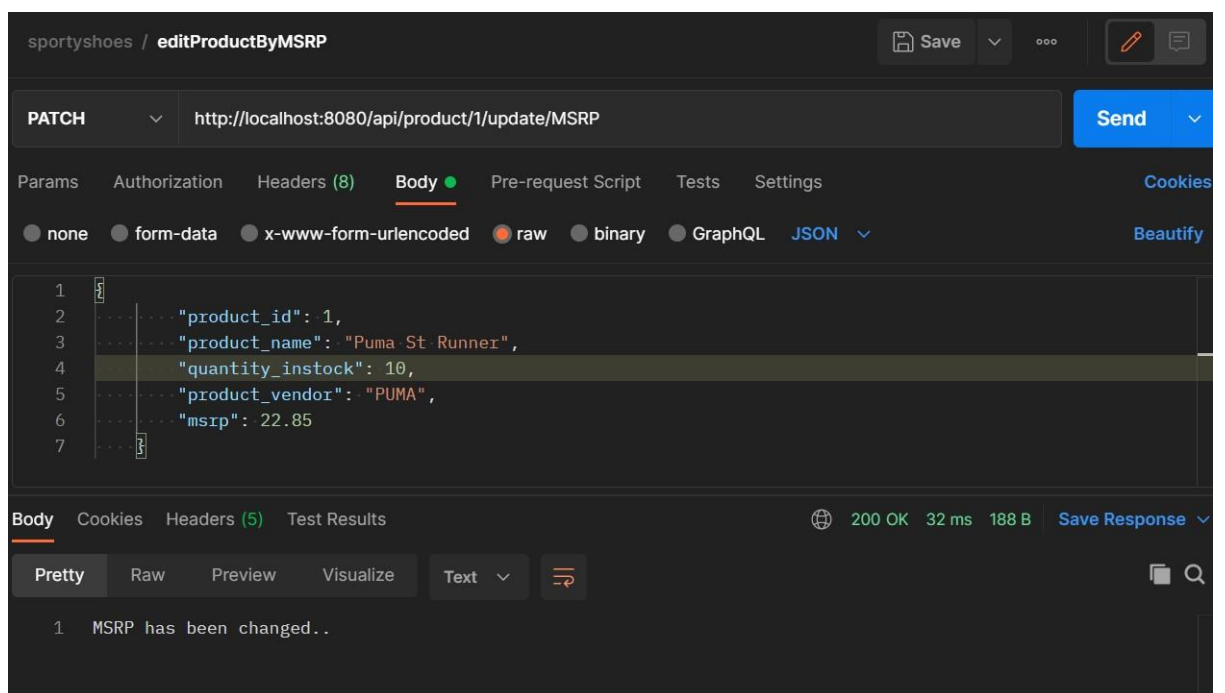
Pretty Raw Preview Visualize JSON ▼ ↔ 📄 🔍

```
1  {
2    "product_id": 1,
3    "product_name": "PUMA St Runner",
4    "quantity_instock": 10,
5    "product_vendor": "PUMA",
6    "msrp": 20.85
7  }
```

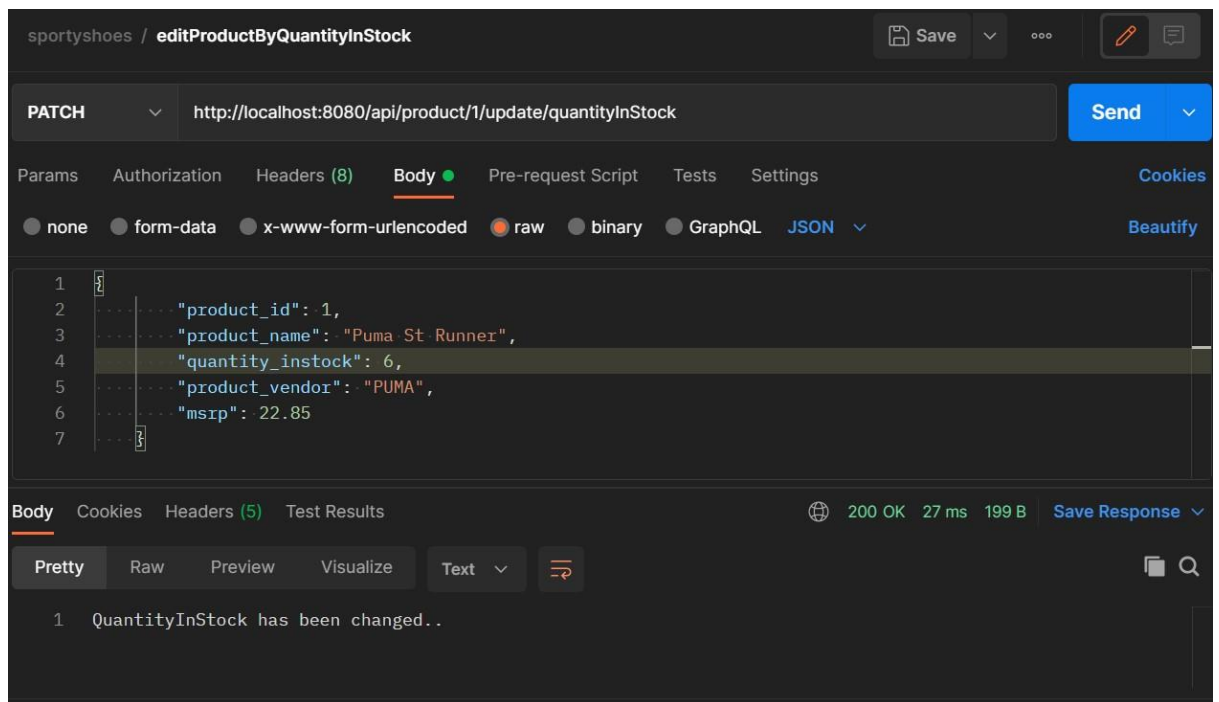
Viewing product by product name.



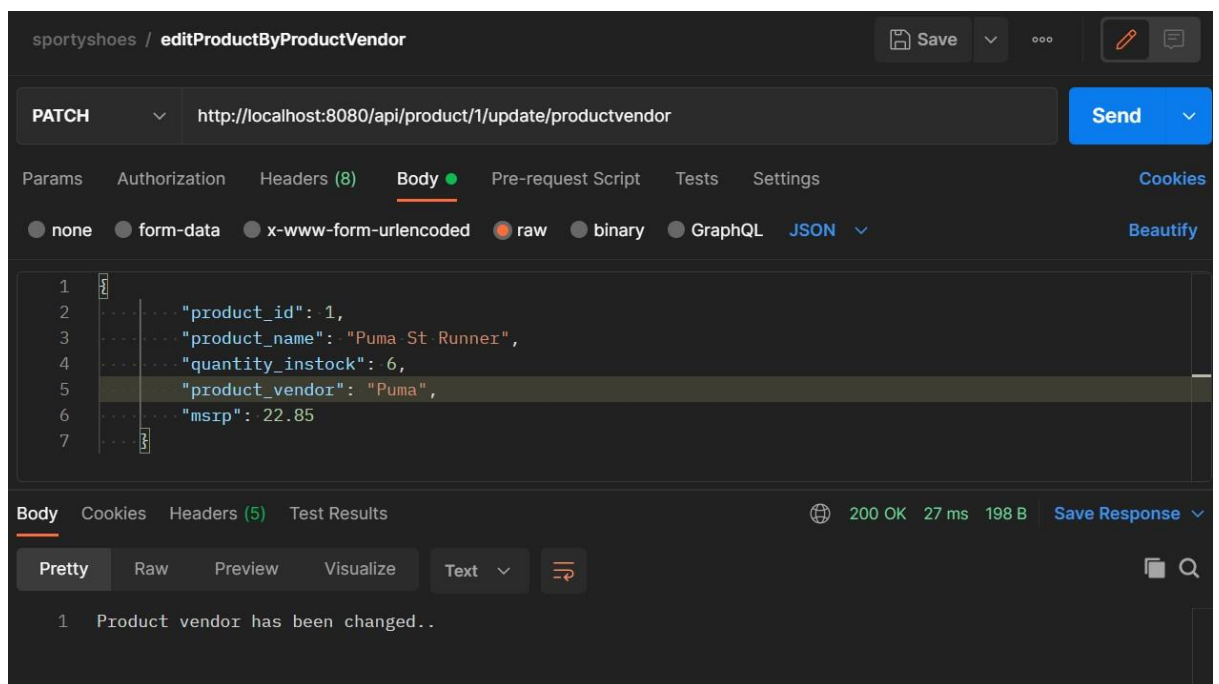
Edit product by name.



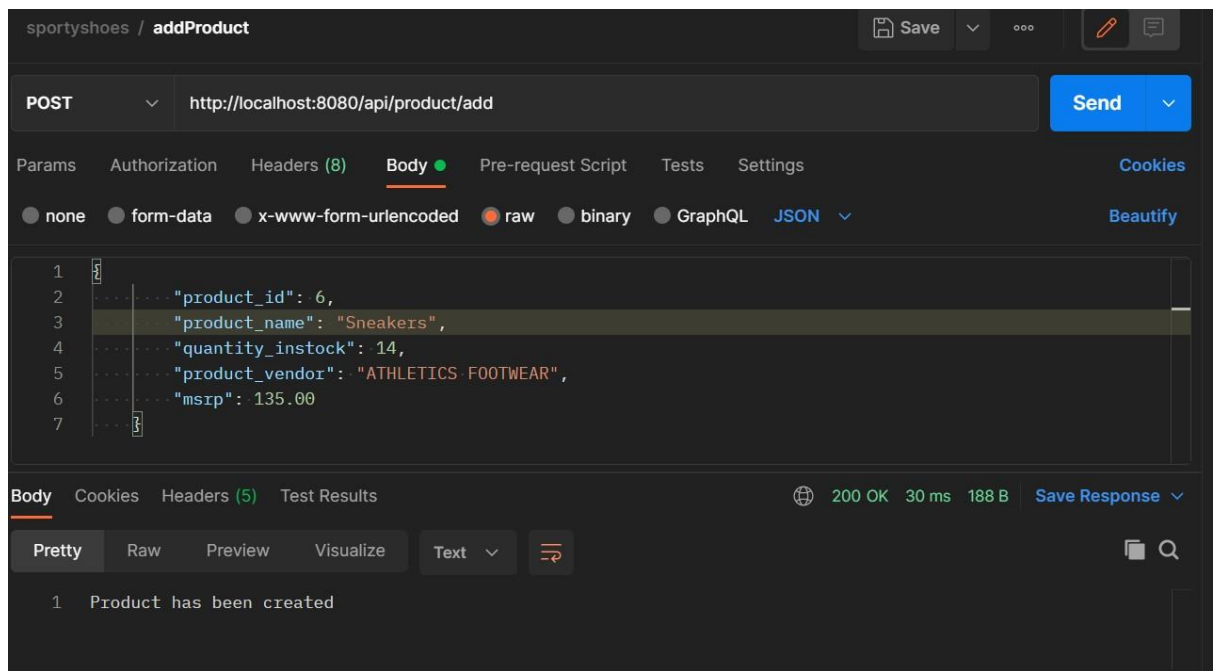
Edit product by MSRP.



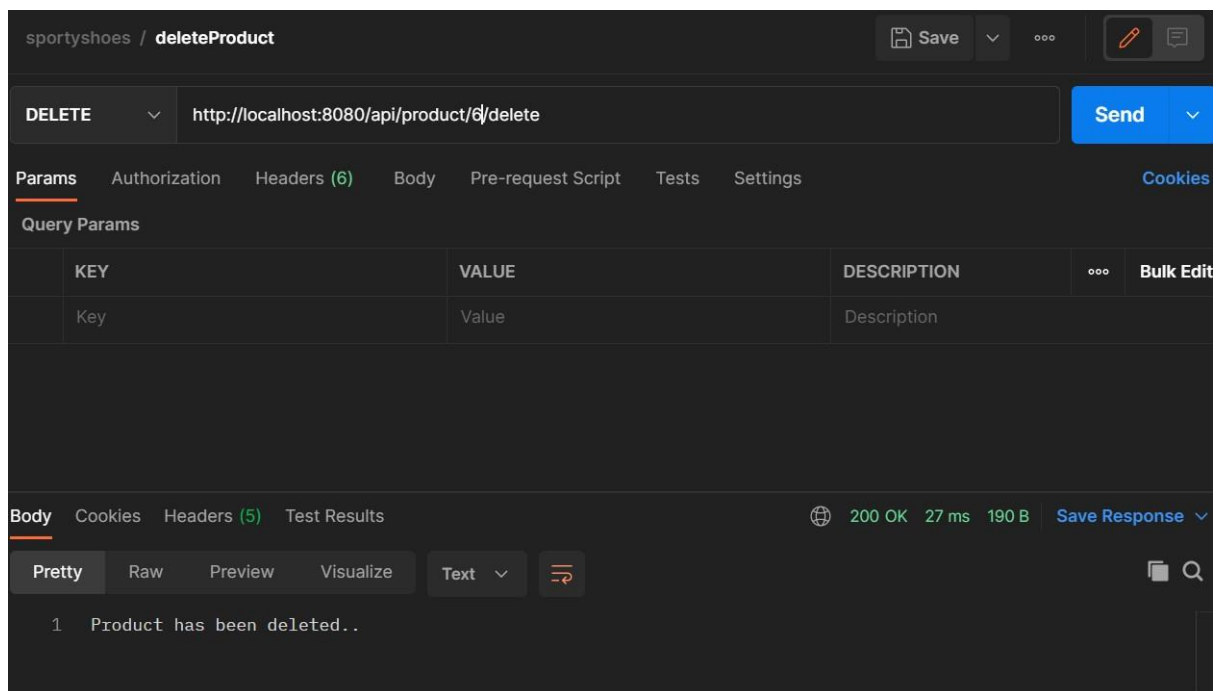
Edit product by quantity in stock.



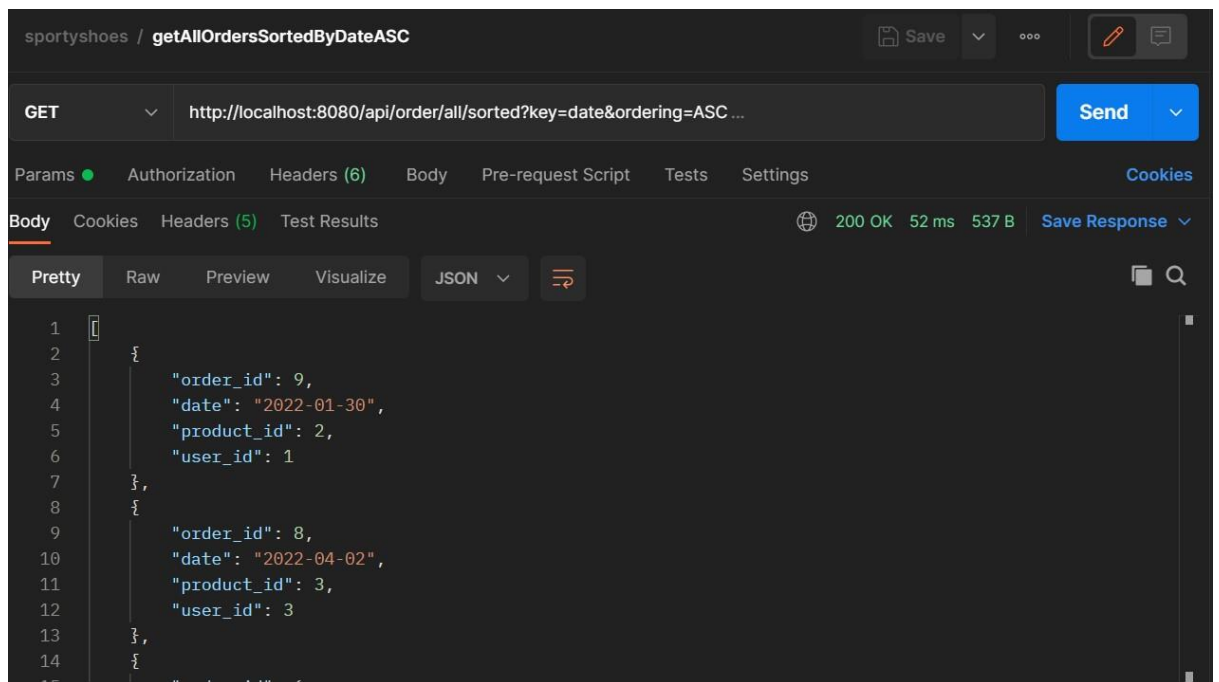
Edit product by product vendor.



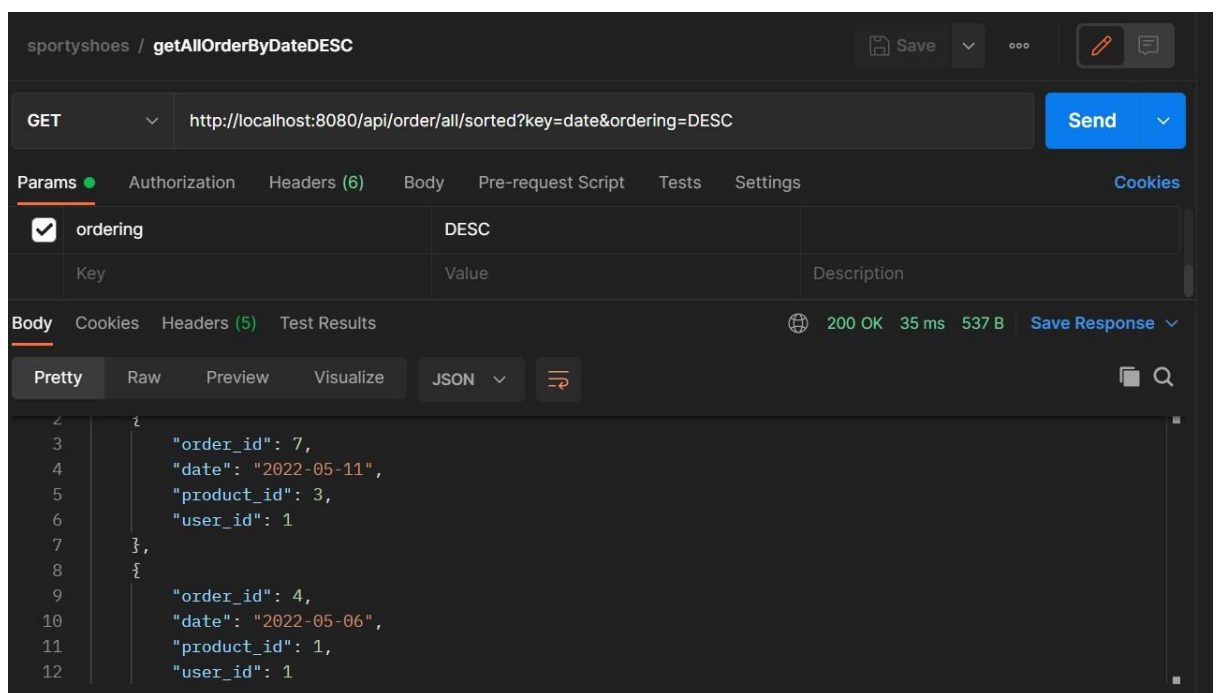
Add a product.



Delete a product.



All orders sorted by order date ascending.



All orders sorted by order date descending.

sportyshoes / **getAllOrderByIdsASC**

GET **Send**

Params **Authorization** Headers (6) Body Pre-request Script Tests Settings Cookies

ordering	ASC
Key	Value
Description	

Body Cookies Headers (5) Test Results 200 OK 46 ms 537 B Save Response

Pretty Raw Preview Visualize JSON

```
2 {
3   "order_id": 4,
4   "date": "2022-05-06",
5   "product_id": 1,
6   "user_id": 1
7 },
8 {
9   "order_id": 5,
10  "date": "2022-05-05",
11  "product_id": 3,
12  "user_id": 2
```

All orders sorted by order id ascending.

sportyshoes / **getAllOrdersByIdDESC**

GET **Send**

Params **Authorization** Headers (6) Body Pre-request Script Tests Settings Cookies

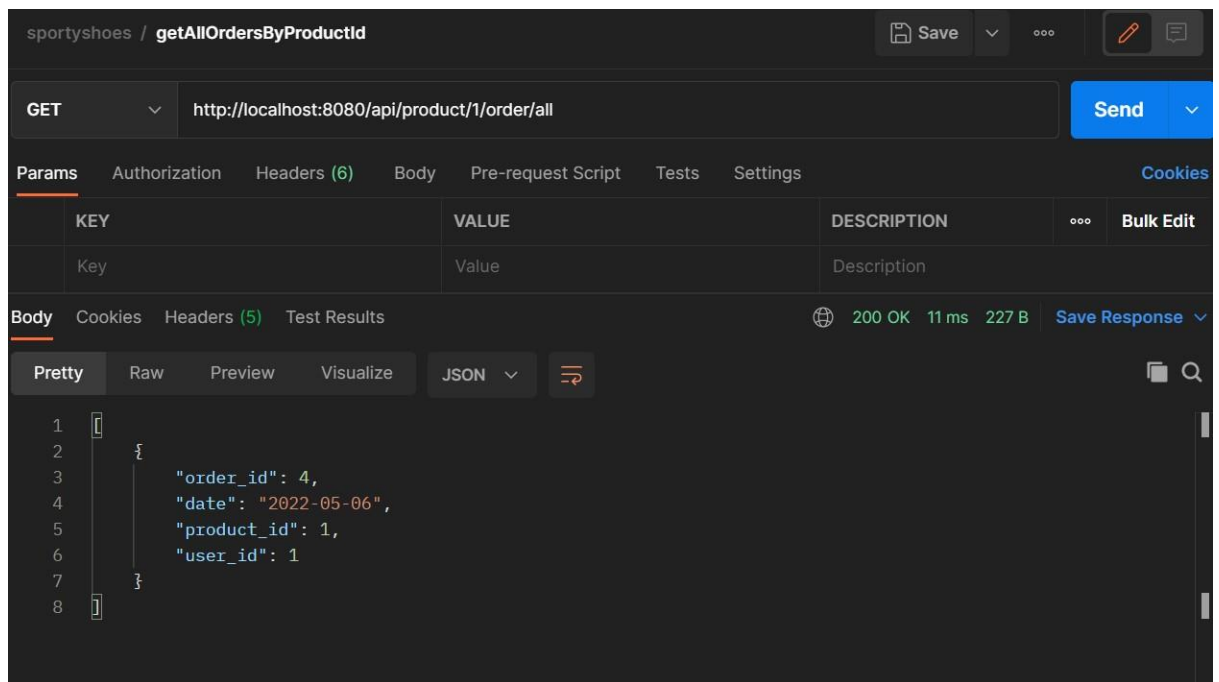
ordering	DESC
Key	Value
Description	

Body Cookies Headers (5) Test Results 200 OK 18 ms 537 B Save Response

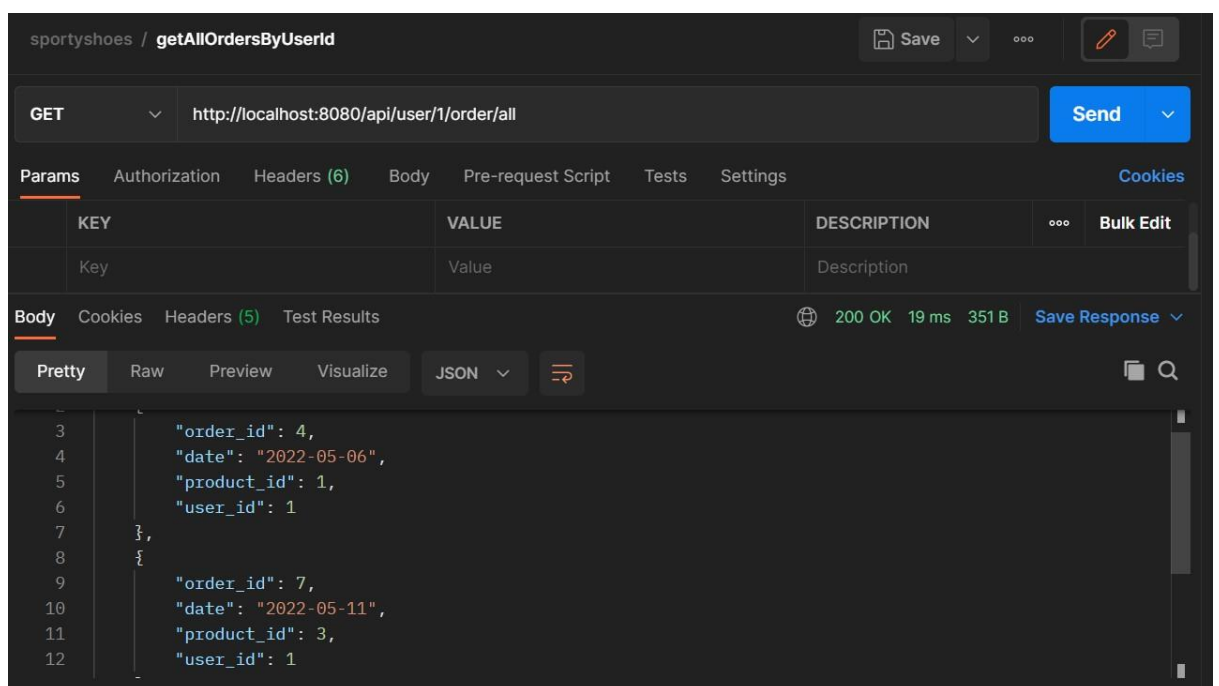
Pretty Raw Preview Visualize JSON

```
3 {
4   "order_id": 9,
5   "date": "2022-01-30",
6   "product_id": 2,
7   "user_id": 1
8 },
9 {
10  "order_id": 8,
11  "date": "2022-04-02",
12  "product_id": 3,
13  "user_id": 3
```

All orders sorted by order id descending.



All orders by product id.



All orders by user id.

To conclude, this project will assist the admin a lot in his/her everyday use. It is user-friendly and easy to learn. In the future, we would like to add a frontend website so that it's easier for the admin to use and update stocks. This project aims to design and develop a

backend sporty shoes website spring boot framework. The goal of this project is to apply JDBC concepts and spring boot.

This is the link to github:

<https://github.com/Neelzzzz/SportyShoes>

the code of the database below.

```
CREATE SCHEMA `sporty_shoes` ;
```

```
use sporty_shoes;
```

```
CREATE TABLE `customers` (  
  `user_id` int NOT NULL AUTO_INCREMENT,  
  `full_name` varchar(200) NOT NULL,  
  `password` varchar(55) NOT NULL,  
  primary key (`user_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
INSERT INTO `customers` (`full_name`, `password`) VALUES  
( 'Surendra Kumar', '*****'),  
( 'Kislay Sinha', '*****'),  
( 'Alaa Azmi', '*****');
```

```
CREATE TABLE `admin_details` (  
  `admin_id` int NOT NULL AUTO_INCREMENT,  
  `admin_name` varchar(55) NOT NULL,  
  `password` varchar(55) NOT NULL,  
  primary key (`admin_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
INSERT INTO `admins_details` (`admin_name`, `password`) VALUES  
( 'Neelkanth Tripathi ', '*****');
```

```
CREATE TABLE `products_in_stock` (
  `product_id` int NOT NULL AUTO_INCREMENT,
  `product_name` varchar(200) NOT NULL,
  `MSRP` decimal(5,2) NOT NULL,
  `quantity_instock` int NOT NULL,
  `product_vendor` varchar(55) NOT NULL,
  primary key (`product_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
INSERT INTO `products_in_stock` (`product_name`, `MSRP`, `quantity_instock`, `product_vendor`) VALUES
( ' Nike Runner', 120.85,100,'NIKE'),
( 'Adidas walking shoes', 28.12, 65,'ADIDAS'),
( 'SKECHERS Summits',105.40 ,29,'SKECHERS');
```

```
CREATE TABLE `customers_orders` (
  `order_id` int NOT NULL AUTO_INCREMENT,
  `date` date NOT NULL,
  `user_id` int NOT NULL,
  `product_id` int NOT NULL,
  primary key (`order_id`),
  foreign key (`user_id`) references `users`(`user_id`), foreign
key (`product_id`) references `products`(`product_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
INSERT INTO `customers_orders` (`date`, `user_id`, `product_id`) VALUES
( '2022-11-6', 1,6),
( '2022-7-5',2,4), (
'2022-12-1',3,5),
( '2022-05-11', '1', '3'),
('2022-04-2', '3', '3'),
('2022-01-30', '1', '2');
```