SPORTY SHOES

(AN E-COMMERCE WEBSITE)

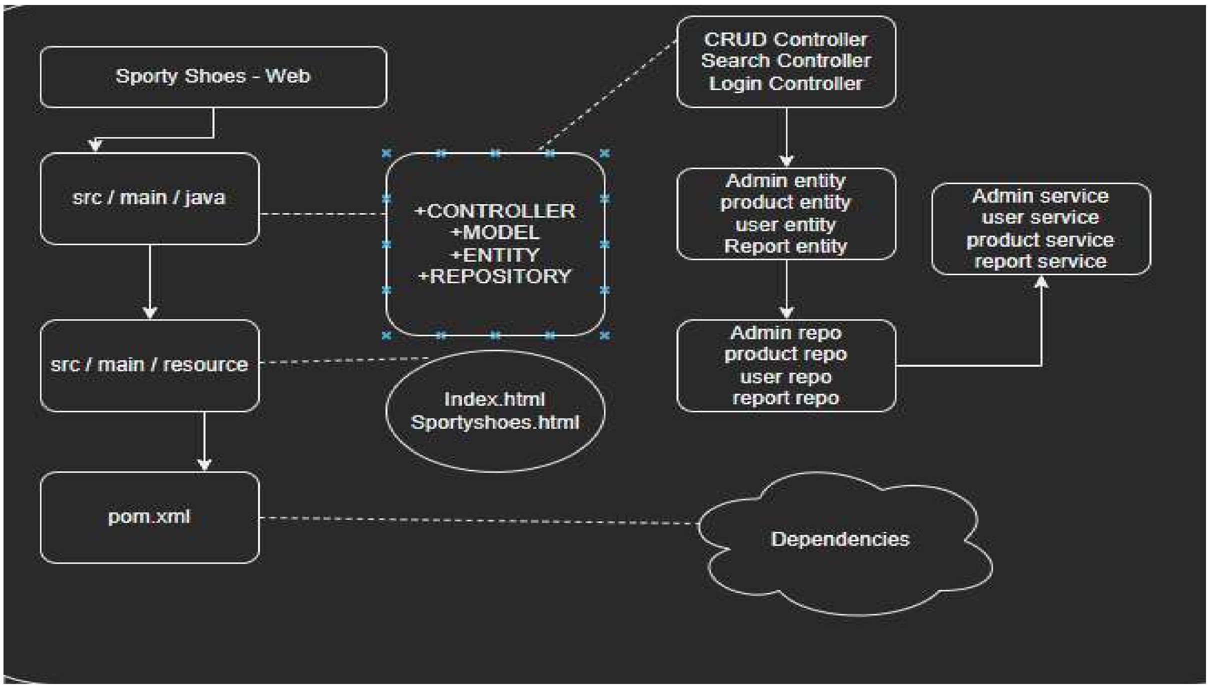
GITHUB LINK: https://github.com/AnishAugustin09/Sporty-Shoes

This document contains the:

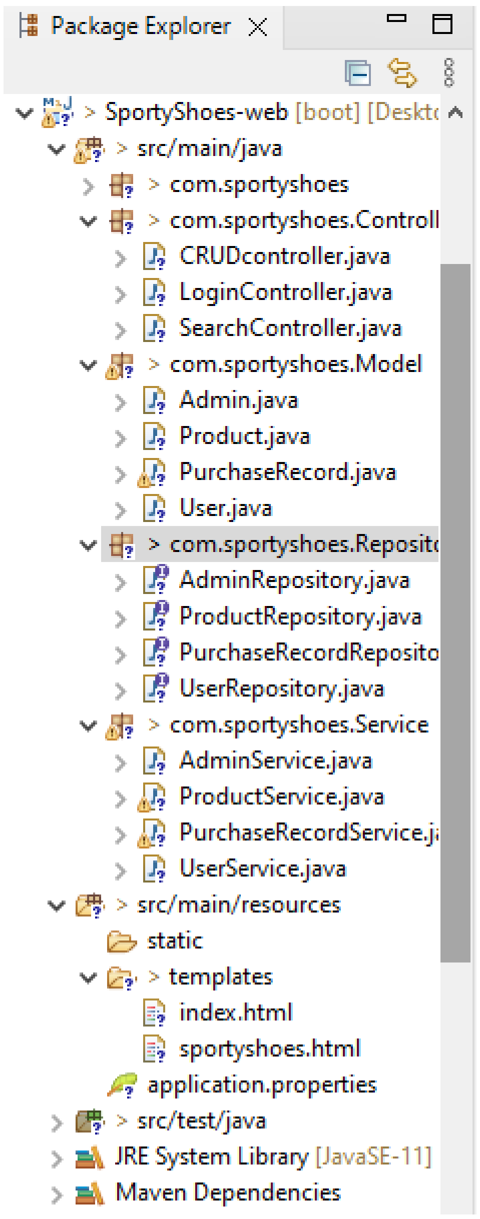
Sprint Planning:

For this project, I've planned to complete the project in 4 Sprints. And the task is completed according to that. And This is a Backend Project so no Frontend Technologies user here

Flow of Project:



Project Flow:

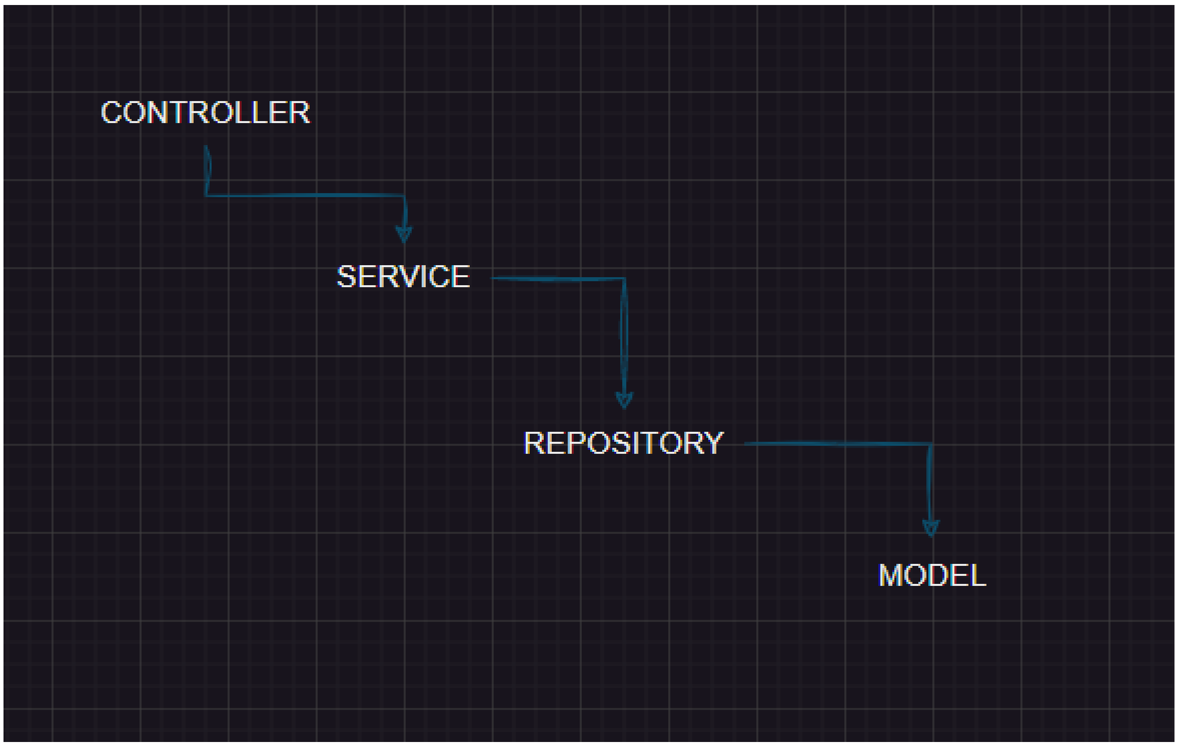

Package Explorer

```
SportyShoes-web [boot] [Deskto
  src/main/java
    com.sportyshoes
    com.sportyshoes.Controll
      CRUDcontroller.java
      LoginController.java
      SearchController.java
    com.sportyshoes.Model
      Admin.java
      Product.java
      PurchaseRecord.java
      User.java
    com.sportyshoes.Reposito
      AdminRepository.java
      ProductRepository.java
      PurchaseRecordReposito
      UserRepository.java
    com.sportyshoes.Service
      AdminService.java
      ProductService.java
      PurchaseRecordService.j
      UserService.java
  src/main/resources
    static
    templates
      index.html
      sportyshoes.html
    application.properties
  src/test/java
  JRE System Library [JavaSE-11]
  Maven Dependencies
```

Concepts used in Project:

- HTML
- JAVA , API, MAVEN
- H2 DATABASE
- SPRING TOOL SUITE 4
- Tomcat Server 9.0
- Git & Github
- POSTMAN

Project Flow:



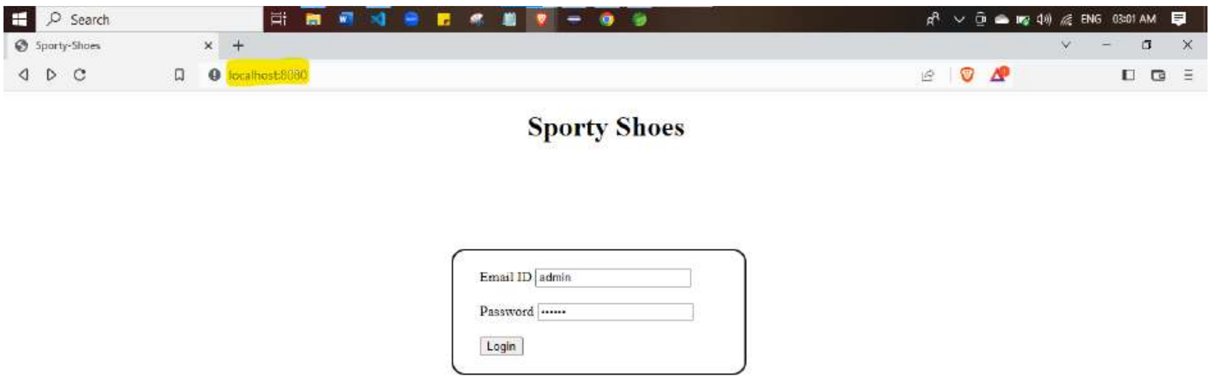
Working on SPRING TOOL SUITE 4 IDE:

Step 1: Creating a new Spring Starter Project in STS

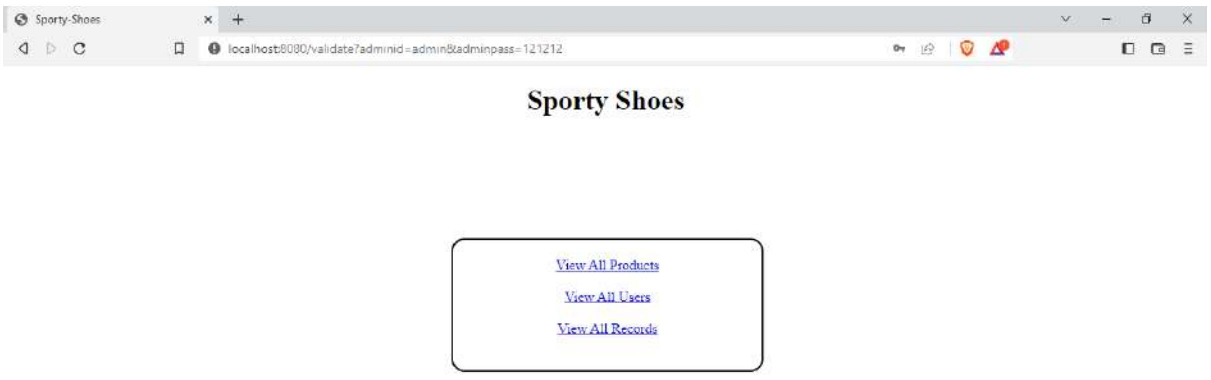
- Open STS
- Go to File -> New -> Spring Starter Project --> Next.
- Enter GroupID & ArtifactID click →Next
- Select your project require dependencies
- click FINISH.

Step 2: Writing a program in JAVA for the entry point of the application (**LOGINCONTROLLER).** This code will redirect to another pager

Step 3: It'll take you to the **INDEX.HTML** page, where the admin can login with credentials.



Step 4: By giving the email id as **admin** & password as **121212,** you can enter into the main page. The main page will have some options to view some methods in browser itself.



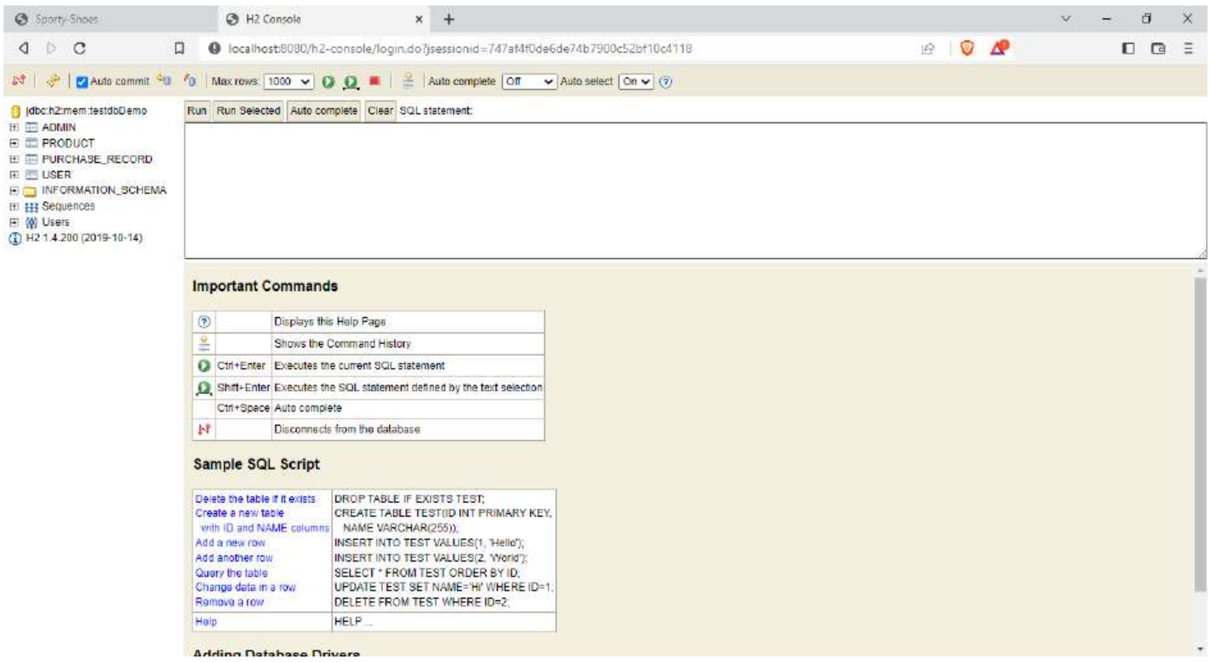
Step 5: Here the admin can access some methods like view report, view users & view product. **However this project is fully based on the Backend API calls so I didn't much focused on frontend.**

Step 6: Once the Admin email id and password has validated, the admin can focus on

- View Products
- View Users
- View Reports
- Add Product
- Update Product
- Delete Product
- Add user
- Update User
- Delete User
- Add Report
- Update Repost

- Delete report
- View Report by Date
- View Report by Category
- View User by Email &
- The Admin can update password if he/she wants

Step 7: Once the program started running, You can test the H2-CONSOLE embedded with Spring



Important Commands

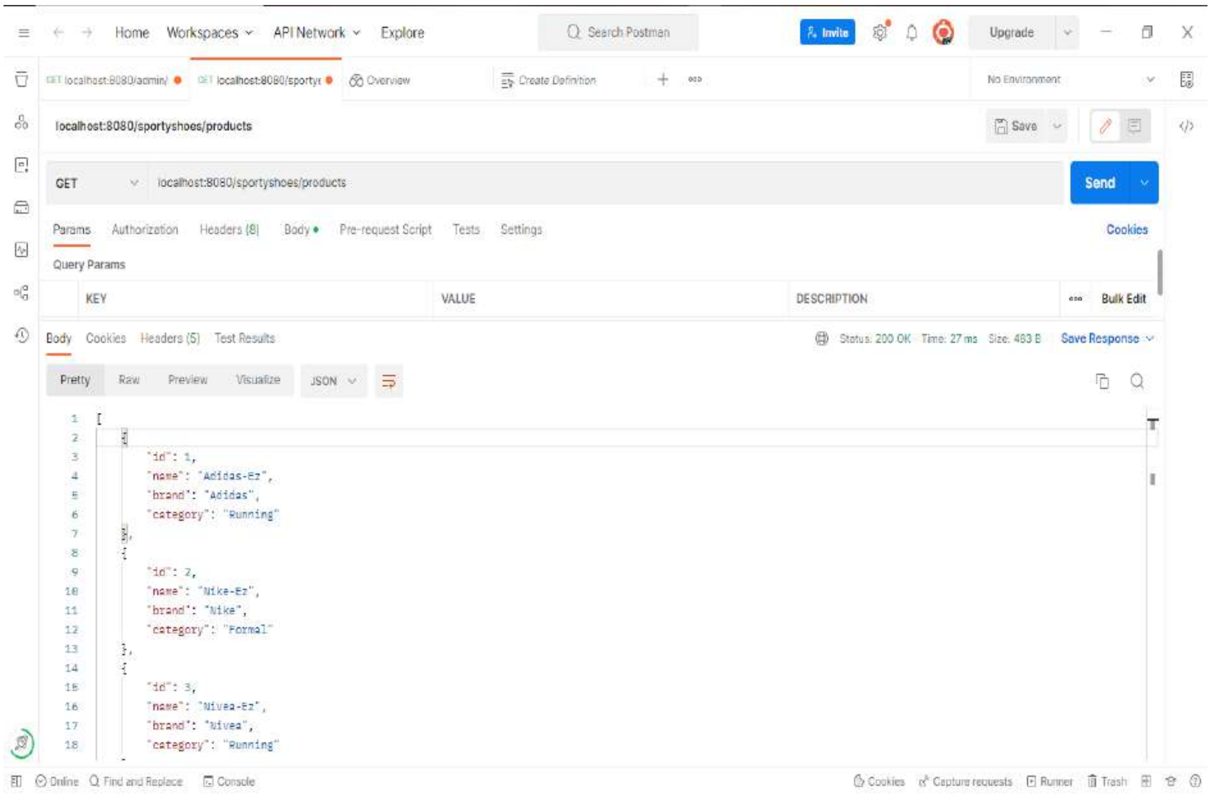
?		Displays this Help Page
		Shows the Command History
	Ctrl+Enter	Executes the current SQL statement
	Shift+Enter	Executes the SQL statement defined by the text selection
	Ctrl+Space	Auto complete
		Disconnects from the database

Sample SQL Script

Delete the table if it exists	DROP TABLE IF EXISTS TEST;
Create a new table with ID and NAME columns	CREATE TABLE TEST(ID INT PRIMARY KEY, NAME VARCHAR(255));
Add a new row	INSERT INTO TEST VALUES(1, 'Hello');
Add another row	INSERT INTO TEST VALUES(2, 'World');
Query the table	SELECT * FROM TEST ORDER BY ID;
Change data in a row	UPDATE TEST SET NAME='Hi' WHERE ID=1;
Remove a row	DELETE FROM TEST WHERE ID=2;
Help	HELP ...

Adding Database Drivers

View Products :

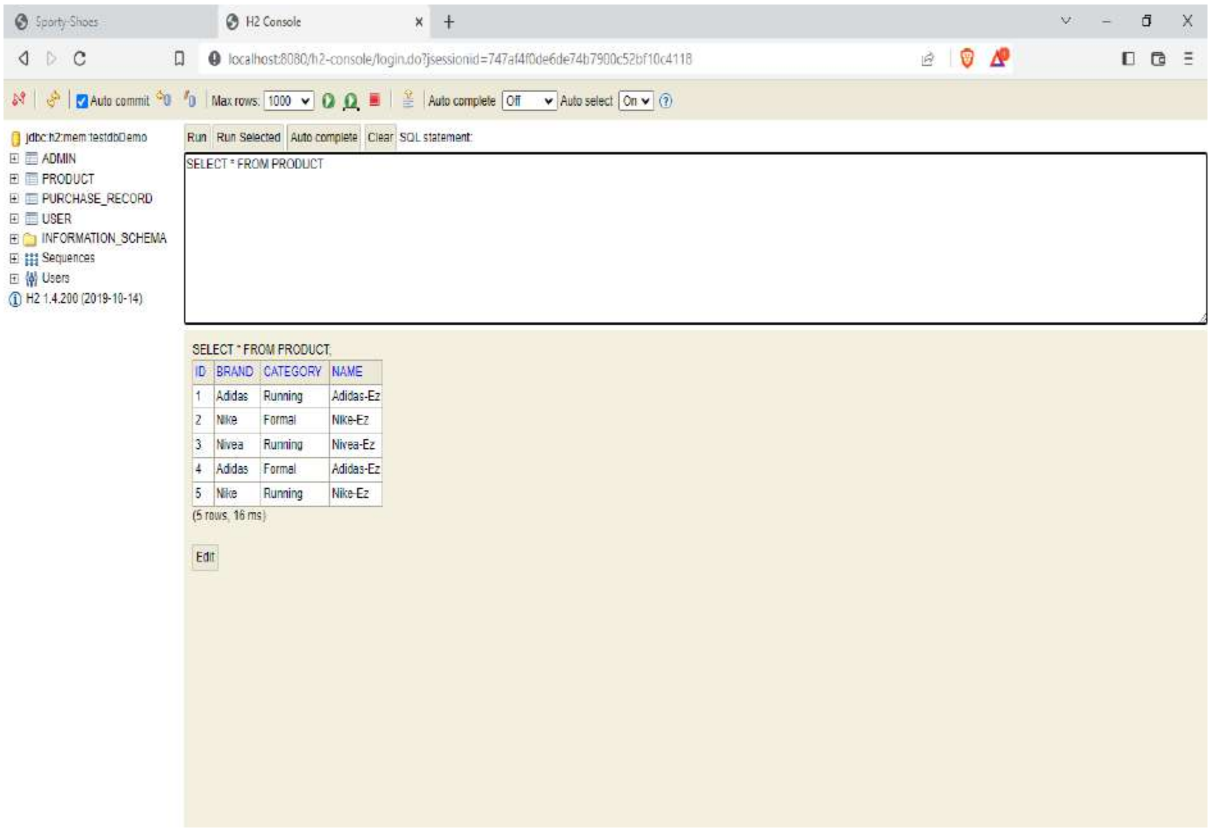


localhost:8080/sportyshoes/products

GET localhost:8080/sportyshoes/products

Status: 200 OK Time: 27 ms Size: 483 B

```
[
    {
        "id": 1,
        "name": "Adidas-Ez",
        "brand": "Adidas",
        "category": "Running"
    },
    {
        "id": 2,
        "name": "Nike-Ez",
        "brand": "Nike",
        "category": "Formal"
    },
    {
        "id": 3,
        "name": "Nivea-Ez",
        "brand": "Nivea",
        "category": "Running"
```



localhost:8080/h2-console/login.do?jsessionid=747af4f0de6de74b7900c52bf10c4118

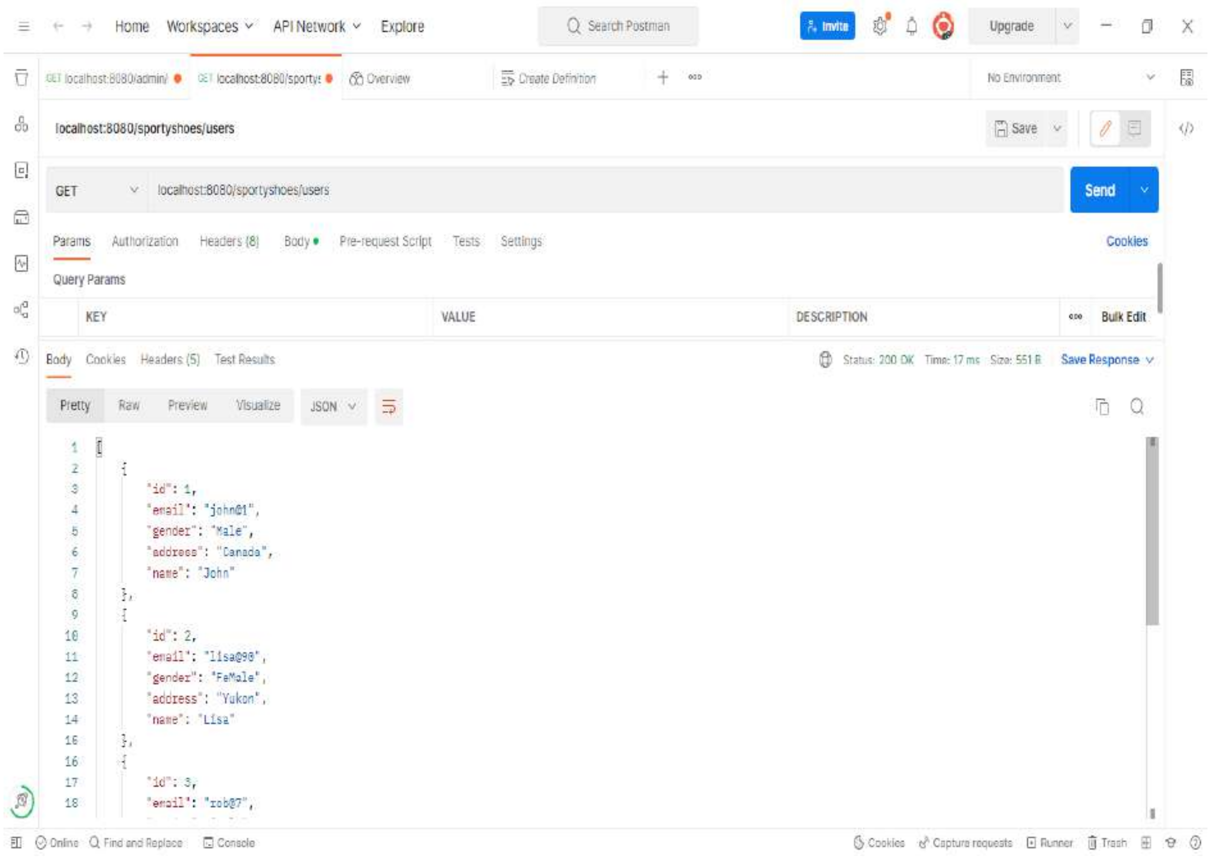
```
SELECT * FROM PRODUCT
```

SELECT * FROM PRODUCT;

ID	BRAND	CATEGORY	NAME
1	Adidas	Running	Adidas-Ez
2	Nike	Formal	Nike-Ez
3	Nivea	Running	Nivea-Ez
4	Adidas	Formal	Adidas-Ez
5	Nike	Running	Nike-Ez

(5 rows, 16 ms)

View Users :

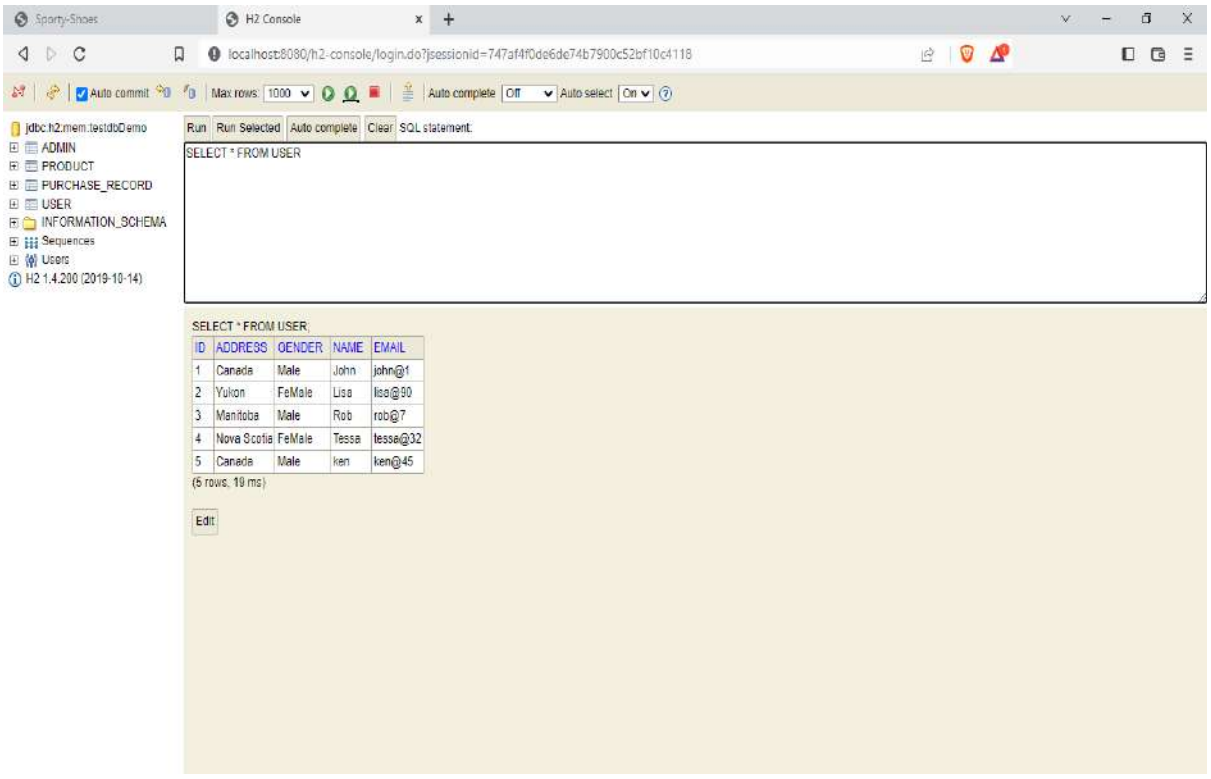


localhost:8080/sportyshoes/users

GET localhost:8080/sportyshoes/users

Status: 200 OK Time: 17 ms Size: 551 B

```
[
    {
        "id": 1,
        "email": "john@1",
        "gender": "Male",
        "address": "Canada",
        "name": "John"
    },
    {
        "id": 2,
        "email": "lisa@90",
        "gender": "FeMale",
        "address": "Yukon",
        "name": "Lisa"
    },
    {
        "id": 3,
        "email": "rob@7",
```



localhost:8080/h2-console/login.do?jsessionid=747af4f0de6de74b7900c52bf10c4118

jdbc:h2:mem:testdbDemo

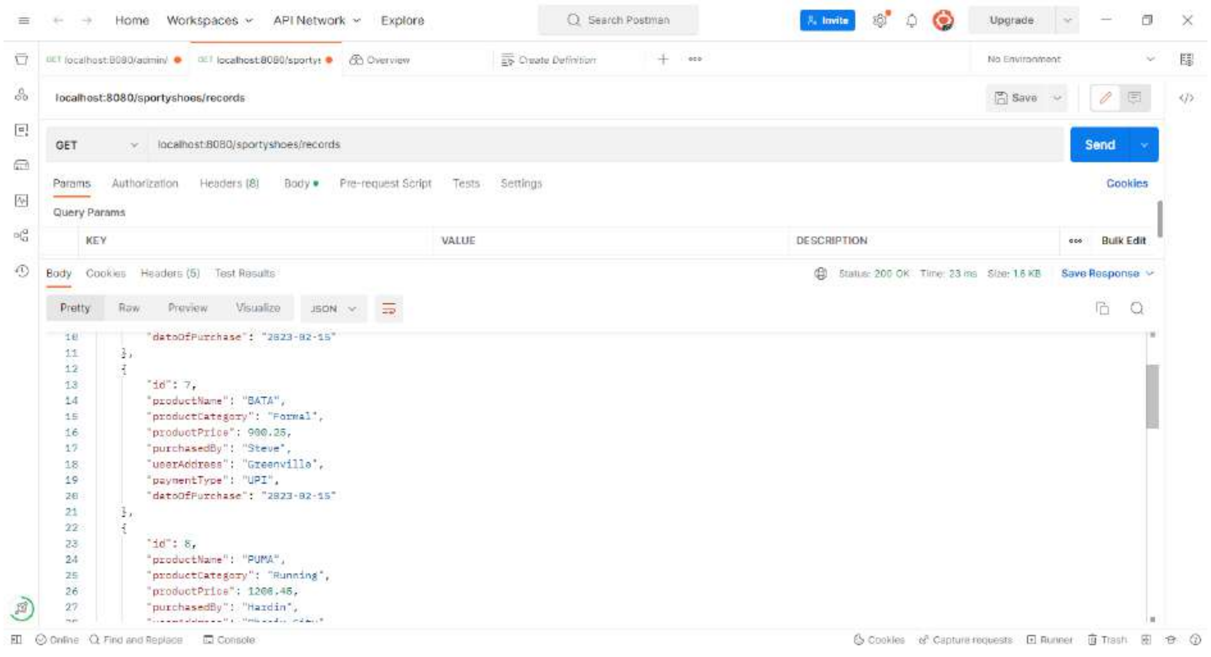
```
SELECT * FROM USER
```

SELECT * FROM USER;

ID	ADDRESS	GENDER	NAME	EMAIL
1	Canada	Male	John	john@1
2	Yukon	FeMale	Lisa	lisa@90
3	Manitoba	Male	Rob	rob@7
4	Nova Scotia	FeMale	Tessa	tessa@32
5	Canada	Male	ken	ken@45

(5 rows, 19 ms)

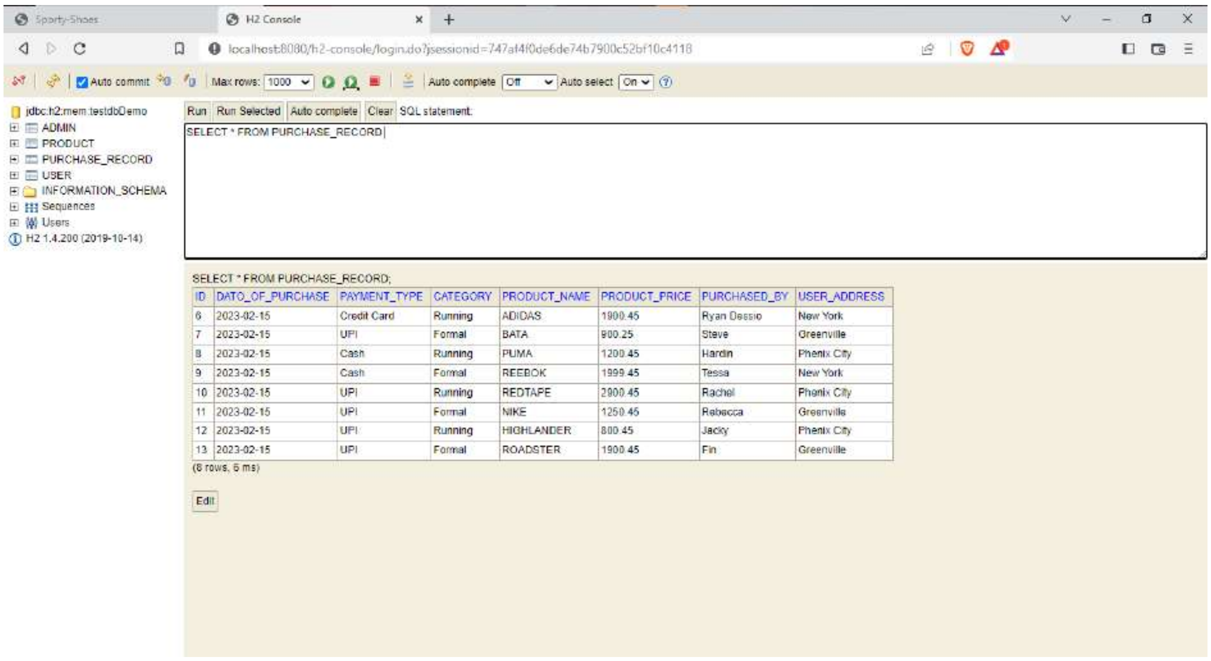
View Reports :



GET localhost:8080/sportyshoes/records

Status: 200 OK Time: 23 ms Size: 1.6 KB

```
        "datoOfPurchase": "2023-02-15"
    },
    {
        "id": 7,
        "productName": "BATA",
        "productCategory": "Formal",
        "productPrice": 900.25,
        "purchasedBy": "Steve",
        "userAddress": "Greenville",
        "paymentType": "UPI",
        "datoOfPurchase": "2023-02-15"
    },
    {
        "id": 8,
        "productName": "PUMA",
        "productCategory": "Running",
        "productPrice": 1200.45,
        "purchasedBy": "Hardin",
```



localhost:8080/h2-console/login.do?jsessionid=747af4f0de6de74b7900c52bf10c4118

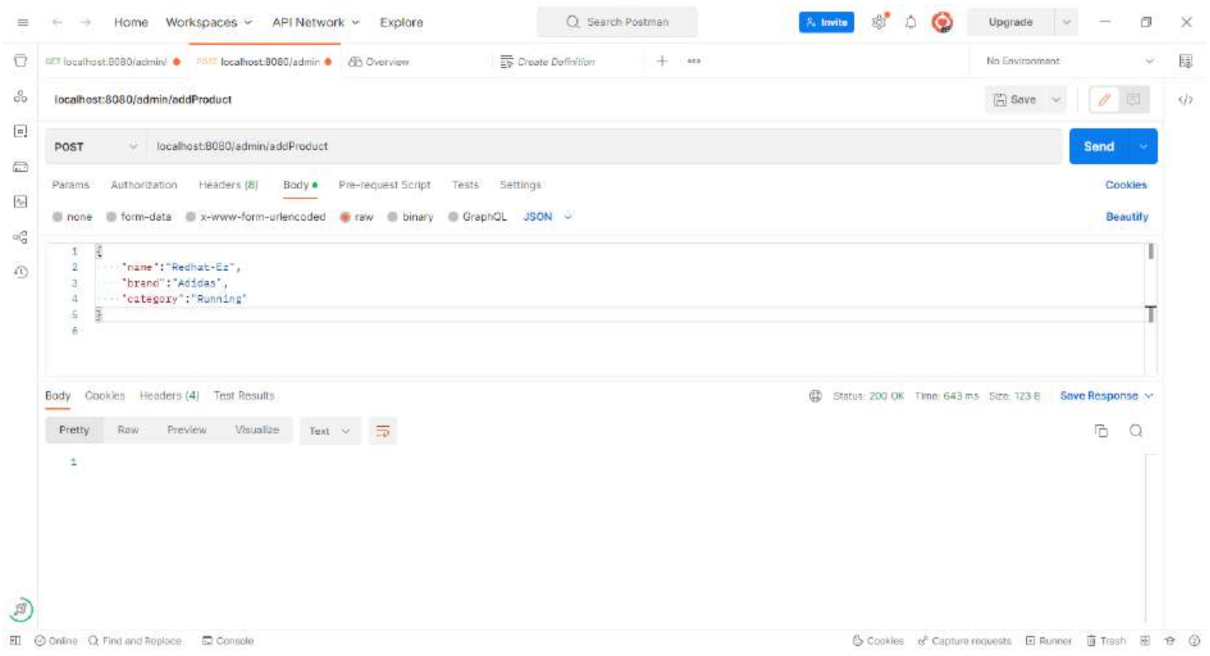
jdbc:h2:mem:testdbDemo

```
SELECT * FROM PURCHASE_RECORD;
```

ID	DATO_OF_PURCHASE	PAYMENT_TYPE	CATEGORY	PRODUCT_NAME	PRODUCT_PRICE	PURCHASED_BY	USER_ADDRESS
6	2023-02-15	Credit Card	Running	ADIDAS	1900.45	Ryan Dessio	New York
7	2023-02-15	UPI	Formal	BATA	900.25	Steve	Greenville
8	2023-02-15	Cash	Running	PUMA	1200.45	Hardin	Phenix City
9	2023-02-15	Cash	Formal	REEBOK	1999.45	Tessa	New York
10	2023-02-15	UPI	Running	REDTAPE	2900.45	Rachel	Phenix City
11	2023-02-15	UPI	Formal	NIKE	1250.45	Rebecca	Greenville
12	2023-02-15	UPI	Running	HIGHLANDER	800.45	Jacky	Phenix City
13	2023-02-15	UPI	Formal	ROADSTER	1900.45	Fin	Greenville

(8 rows, 6 ms)

Add Product :


localhost:8080/admin/addProduct

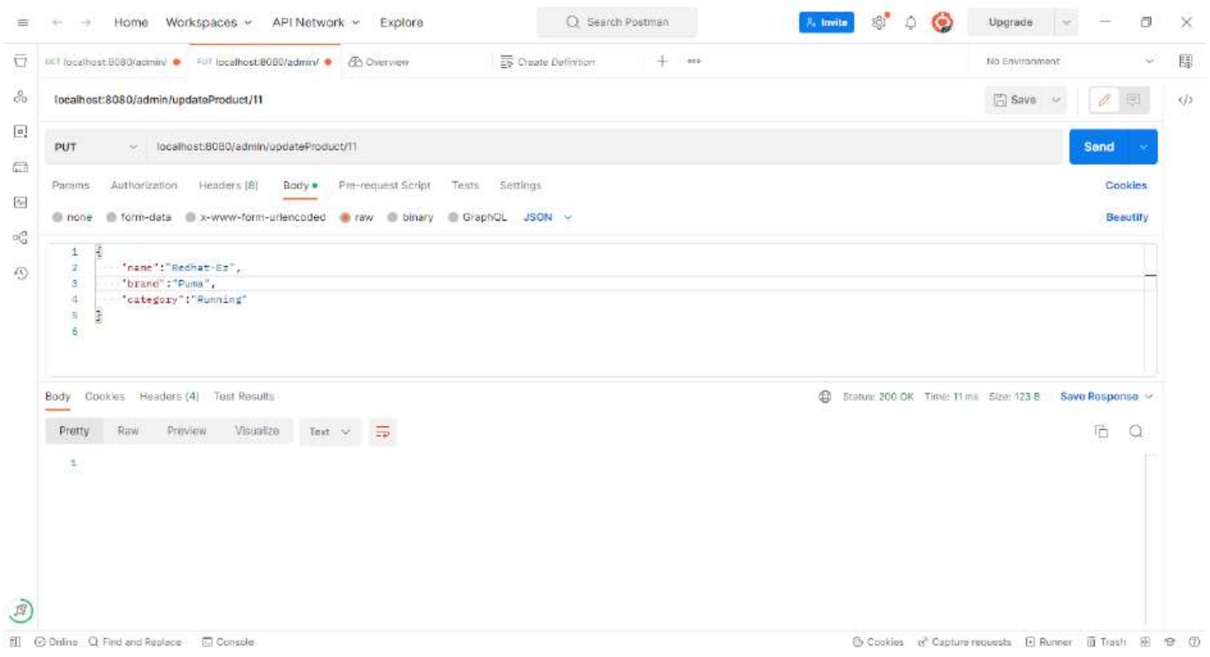
POST localhost:8080/admin/addProduct

Body: raw JSON

```
{
    "name":"Redhat-Ez",
    "brand":"Adidas",
    "category":"Running"
}
```

Status: 200 OK Time: 643 ms Size: 123 B

Update Product :


localhost:8080/admin/updateProduct/11

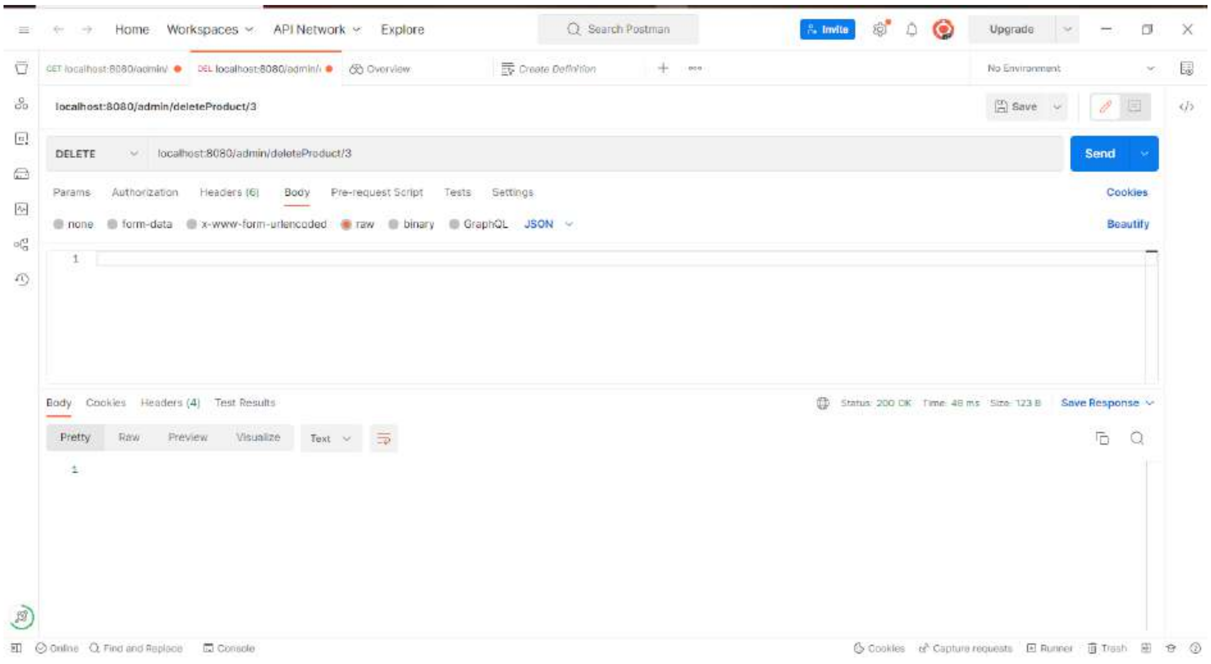
PUT localhost:8080/admin/updateProduct/11

Body: raw JSON

```
{
    "name":"Redhat-Ez",
    "brand":"Puma",
    "category":"Running"
}
```

Status: 200 OK Time: 11 ms Size: 123 B

Delete Product :



localhost:8080/admin/deleteProduct/3

DELETE localhost:8080/admin/deleteProduct/3

Body: none, form-data, x-www-form-urlencoded, raw (selected), binary, GraphQL, JSON

Status: 200 OK Time: 48 ms Size: 123 B

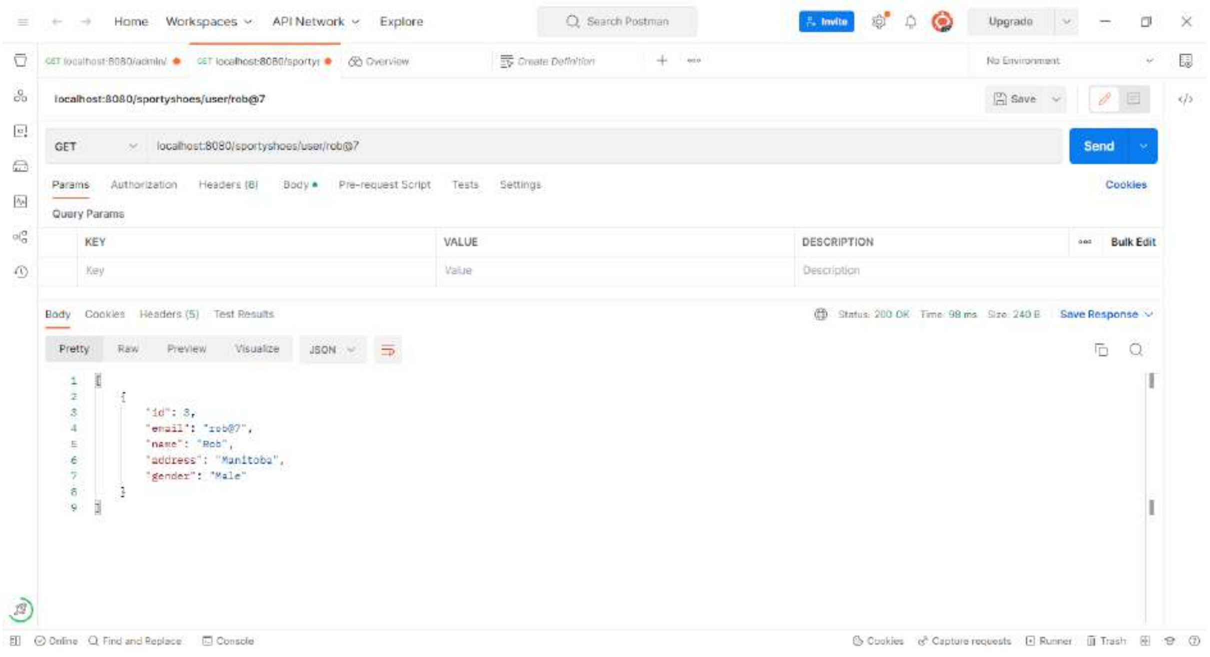
Main Conditions from the Document

- search users
- purchase reports filtered by date and category
- There will be an admin to manage the website. An administrator login will be required to access the admin page.

This conditions has done in project using the NATIVEQUERY concept.

SEARCH USERS :

By using Native Query, the search user option can enable. This Search user will search by the email of the use which we're Searching.

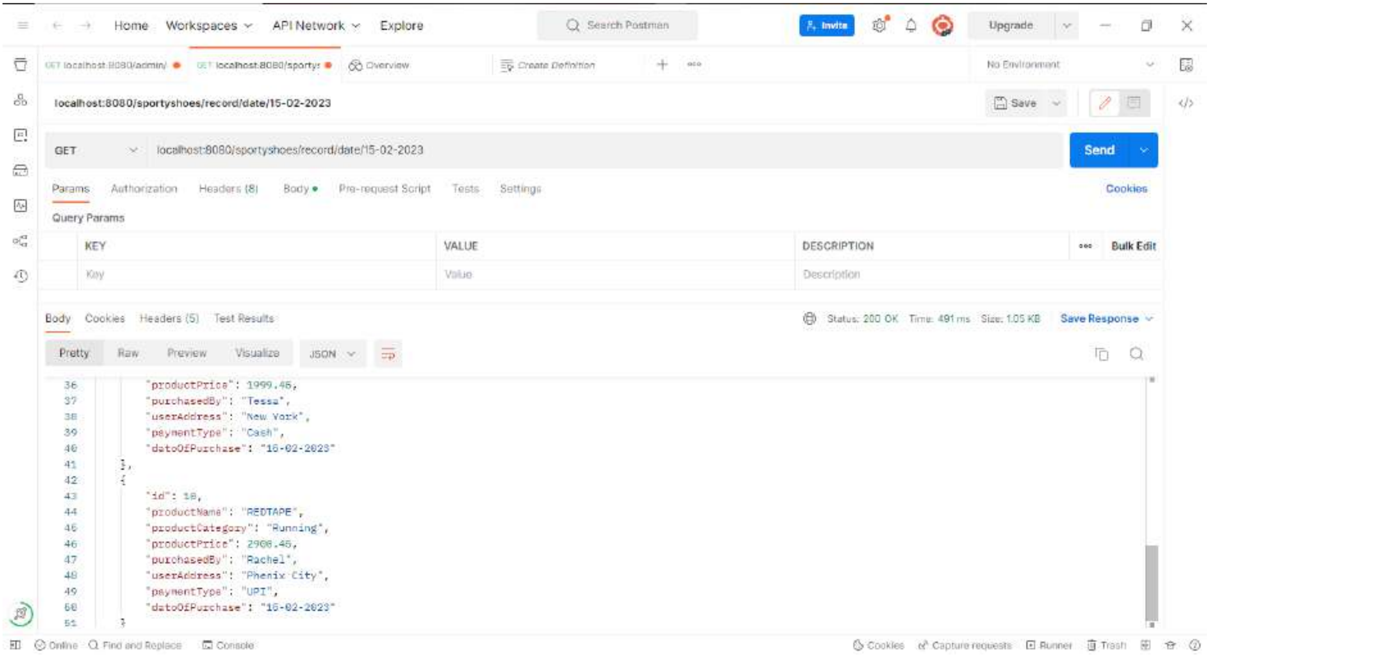

localhost:8080/sportyshoes/user/rob@7

GET localhost:8080/sportyshoes/user/rob@7

Status: 200 OK Time: 98 ms Size: 240 B

```
[
    {
        "id": 3,
        "email": "rob@7",
        "name": "Rob",
        "address": "Manitoba",
        "gender": "Male"
    }
]
```

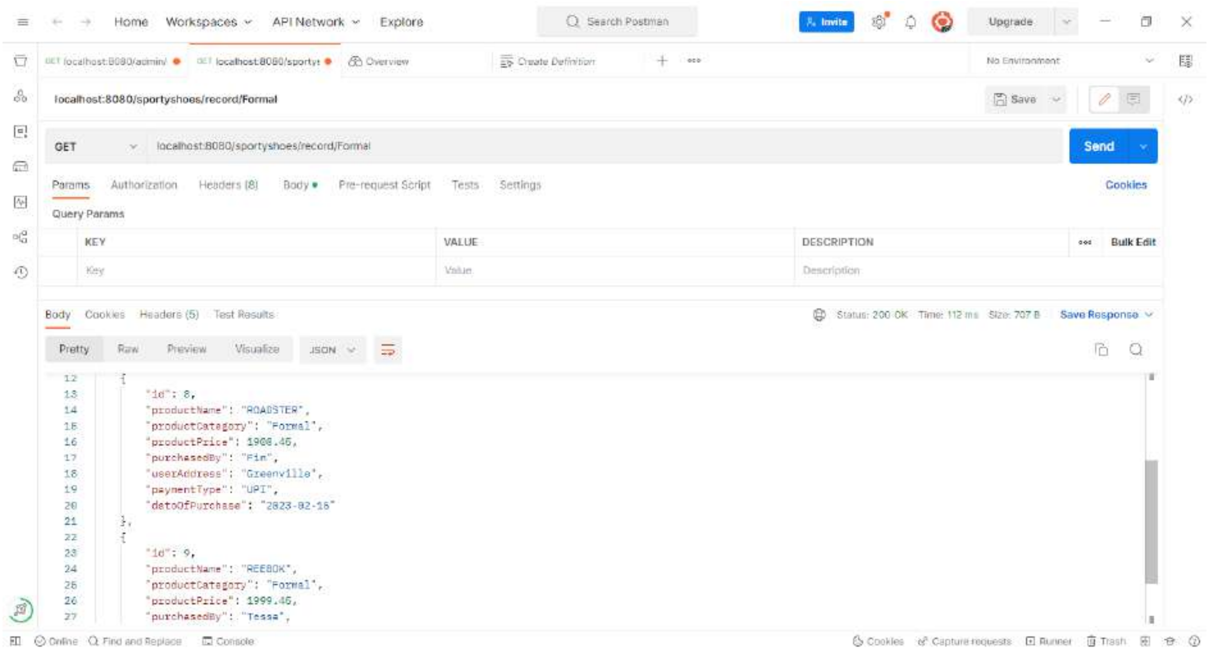
Purchase Reports Filtered By Date And Category :


localhost:8080/sportyshoes/record/date/15-02-2023

GET localhost:8080/sportyshoes/record/date/15-02-2023

Status: 200 OK Time: 491 ms Size: 1.05 KB

```
        "productPrice": 1999.45,
        "purchasedBy": "Tessa",
        "userAddress": "New York",
        "paymentType": "Cash",
        "dateOfPurchase": "15-02-2023"
    },
    {
        "id": 10,
        "productName": "REDTAPE",
        "productCategory": "Running",
        "productPrice": 2900.45,
        "purchasedBy": "Rachel",
        "userAddress": "Phenix City",
        "paymentType": "UPI",
        "dateOfPurchase": "15-02-2023"
    }
```

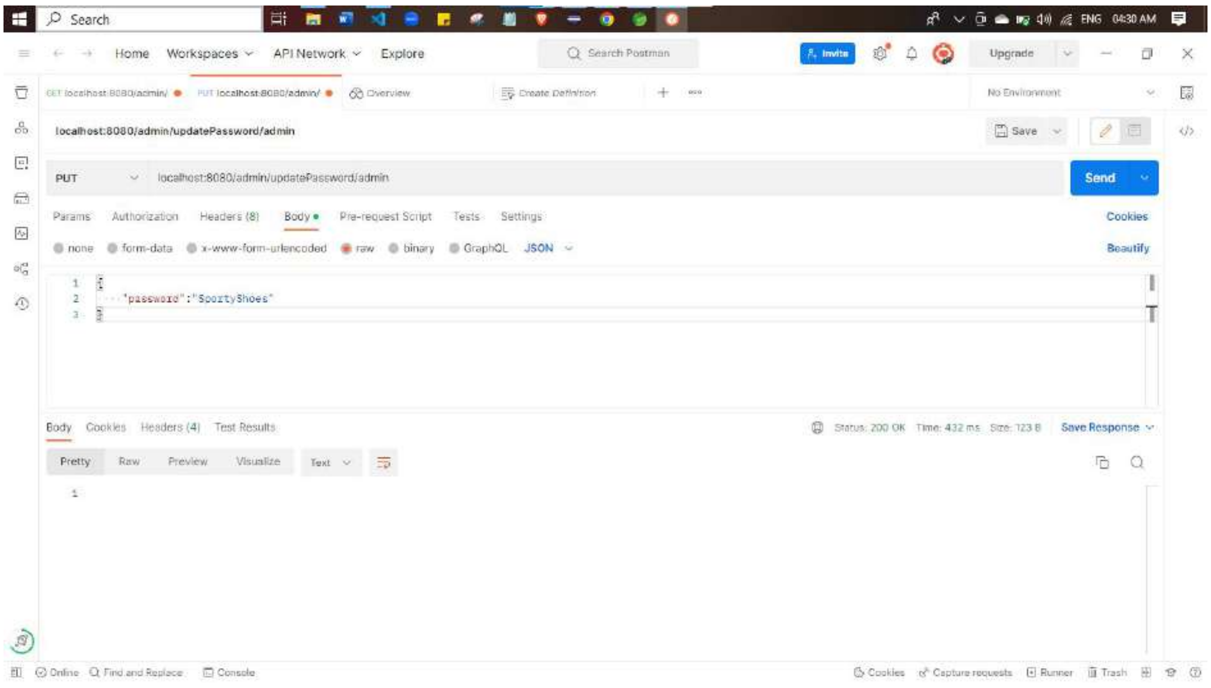


GET localhost:8080/sportyshoes/record/Formal

Status: 200 OK Time: 112 ms Size: 707 B

```
    {
        "id": 8,
        "productName": "ROADSTER",
        "productCategory": "Formal",
        "productPrice": 1908.46,
        "purchasedBy": "Fin",
        "userAddress": "Greenville",
        "paymentType": "UPI",
        "dateOfPurchase": "2023-02-16"
    },
    {
        "id": 9,
        "productName": "REEBOK",
        "productCategory": "Formal",
        "productPrice": 1999.46,
        "purchasedBy": "Tessa",
```

The Admin can update password if he/she wants :



PUT localhost:8080/admin/updatePassword/admin

```
{
    "password":"SportyShoes"
}
```

Status: 200 OK Time: 432 ms Size: 123 B

```
1
```

The goal of the company is to deliver a high-end quality product as early as possible. So, In this Project you can perform

- Admin login
- View Products
- View Users
- View Reports
- Add Product
- Update Product
- Delete Product
- Add user
- Update User
- Delete User
- Add Report
- Update Repost
- Delete report
- View Report by Date
- View Report by Category
- View User by Email &
- The Admin can update password if he/she wants

FINAL SETTING : Upload files to GitHub Repository

- Open your command prompt and navigate to the folder where you have created your files.
- cd <folder path>
- Initialize repository using the following command:
- git init
- Add all the files to your git repository using the following command:
- git add .
- Commit the changes using the following command:
- git commit . -m <commit message>
- Push the files to the folder you initially created using the following command:
- git push -u origin master

Unique Selling Points:

1. This application is use to store & manage the Company products, users & Purchase Record
2. This application will keep on running until user close the program.
3. It allows the admin to add, update , delete, for the products, Users & Purchase Record.
4. It allows the admin to Update Password, Search Reports by date and Category & Search user by email.

Conclusion:

Further enhancements to the application can be made which may include

Admin should able to add many data’s in the DB

To create Many methods using Native Query

To deliver better Front-end.

THANK YOU