Sporty Shoes

A SpringBoot project,
Using H2 database
And Maven
So as to be
fully standalone.

GitHub repository: https://github.com/dedhiah10/SportyShoes.git

1) Project Statement:

As a Full Stack Developer, complete the features of the application by planning the development and pushing the source code to the GitHub repository.

2) Sprint Planning:

A. Product Backlog:

i. **Plan Flow** of the Back-End, Front-End and Object Persistence.
   a. Plan server-side dependency Spring modules, DB, interfaces, JPA, CRUD, Entities, session management etc.
   b. Plan client-side dependency CSS, HTML, JSP, JS.
   c. Plant smart ease of living features.

ii. Start coding **classes** and **interfaces**.
   a. Data members for users, products, transactions, cart, etc and Data Structures required, persistence required and also start coding them.
   b. Different types Annotation mapping, session management, ModelAndViews, repository interface using CRUD/JPA, etc. for the portal
   c. Start coding different annotated HttpServlets and Maintain Entities, Data Structures, add random values for testing.
   d. Start coding in JS, JSP, HTML and CSS for client-side view.
   e. Run and Debug all the code and reduce waste codes.
   f. Test and Iterate code till a certain polish is accomplished.

iii. Run, debug, test and reiterate.
   a. Run and Debug all the code and reduce waste code, data members and structures.
   b. Test and reiterate code till a certain polish is accomplished.

iv. Test Run App.
   a. **Run** and **Test** Application.

GitHub repository: https://github.com/dedhiah10/SportyShoes.git

B. Sprint Table:

Sprint No.	Tasks	Estimation	Status
1.	**Plan Flow** of the Back-End, Front-End and Object Persistence.	4 hours	Pending
2.	Start coding **classes** and **interfaces**. –- **PART I**	10 hours	Pending
3.	Start coding **classes** and **interfaces**. – **PART II**	10 hours	Pending
4.	**Run**, **debug**, **test** and **reiterate.**	4 hours	Pending
6.	**Test** Run App.	2 hours	Pending

C. Sprints:

i. **Plan Flow** of the Back-End, Front-End and Object Persistence.

a. Plan server-side dependency Spring modules, DB, interfaces, JPA, CRUD, Entities, session management etc.:

In this, we plan the dependencies, flow, components, requirements, logical issues, etc to required to make the SportShoes portal or atleast enough to start building it. Here we decided to use spring modules: SpringBoot, devtools, JPA, Hibernate, H2 database, MySQL database. We decided that even though the normal use would be H2 database to make the App truly standalone but the code for connecting to MySQL can be done if needed as it is not much different. Hibernate and JPA are required to persist objects as entities as required. Interfaces like CRUDRepository and JPARepository make things easy by providing the basic SQL queries for getting List<Object> and making the life of programmers easy. We also plan all the Entities i.e. object classes whose objects are to be persisted or stored in the database. We also require to choose one of the many session management tools. We ended chosing HttpSession as the method for maintaining session.

b. Plan client-side dependency CSS, HTML, JSP, JS.:

JS, CSS, JSP and ModelAndViews are the best tools for the job as we plan to offload some view calculation job to client using JS lowering lines of code and also making the website seem to load faster. Here we plan the JSONObjects and JSONArrays to send from the server-side and start with building view on the client-side using the above mentioned JSP, JS, CSS. Model and Views and HttpSession are used on the server-side. We must carefully think about the processing and data we send to the client some data is better of pre-processed as that is more secure than handing down sensitive data. We must also plan additional options for Admin login and make sure on the serverside that normal user cannot access the servlet. Adding more options in the visible DOM for Admin login so that they can access more data was done using JS and data sent from server-side such that even if source code was to be inspected normal user cannot access the page and will be redirected to Home Page.

c. Plant smart ease of living features.:

In this we plan for any kinds of ease of living features that could be added to make portal better. Such as adding products to cart using CartManager, making cart versatile by adding option to remove individual items from the cart, buying all items in the cart using one click. Also, to make user focus out less and enjoy the seemless experience we collect most of the personal data such as address, card-details early on during the registration process so that the Buy option just buys the item.

For admins accessing the site let them view most of the users with their personal data and purchase history along with date and time of purchases made. Also, admins have option to login as normal users so that they can browse portal with users-like experience which is actually just limiting admin to a normal user.

ii. Start coding **classes** and **interfaces**.

a. Data members for users, products, transactions, cart, etc and Data Structures required, persistence required and also start coding them.:

We need to start coding with all the data members, data structures, persistence of an object, classes in mind. Users, Products, Transactions and Carts are the classes which need to be managed using data structures, Hibernate and JPA entities, etc. We decided to add Users and Products to SQL database while using a java data structure to maintain transactions and user's product carts in memory using userIDs and productIDs to map them. This saves on data as Users and Products are mapped rather than hard coding all. i.e. loose coupling is achieved hence, the preference for Spring.

b. Different types Annotation mapping, session management, ModelAndViews, repository interface using CRUD/JPA, etc. for the portal:

In this, we make all the different mappings to functions of different servlets using annotations. We use ModelAndViews, CRUD/JPA repository interfaces, JSONArrays and JSONObjects, HashMaps, ArrayLists and iterator interface to inter play with data and map user to different servlet pages as annotated. CRUD/JPA methods are configured so as to save, update and find entities for Login and Registration. If credentials entered by a client matches of that in repository HttpSession is used to track the user as they move through different sections of the portal. The session is managed unitl they log out by accessing LogOut section.

c. Start coding different annotated HttpServlets and Maintain Entities, Data Structures, add random values for testing.

In this, we start with the rest of the HttpServlet functions that provide ViewAndModel. Here we think carefully about the data sent to the user not being sensitive. We use the aforementioned JSONArrays and JSONObjects and use HashMap to send the JSON objects to client and the client can access them using ${objname} notation with objname being the same as the one configured on the server-side. This also means that changing JSONArray can break the output hence strategically using JSONObjects in their stead, we reduce the risk. We also add Random values to the SQL database so that we can start with the testing of the project as we proceed so we don't waste our time writing false code. We must also refine all the functions that navigate user through the portal. We also decided to maintain user carts and user transactions in java collections which are not backed up in the DB.

d. Start coding in JS, JSP, HTML and CSS for client-side view:

Now we have to code for the client-side view using JSP, CSS and JS majorly. We use JS to Reduce server-side load and also html code. Making the experience feel lightning fast as less data is sent for the same view. We use JS to also make options, popups, disclaimers, etc. show up as needed using server-side HashMap<String, JSONObject> or HashMap<String, JSONOArray> to send and JS innerHTML properties for controlling HTML tags so that view can be calculated rather than typed however clients that don't allow running of .js scripts will face view not working. A CSS is made with all the pages in mind and is linked to all the JSPs. i.e. it is plug and play. We could have made different CSS for different pages, but once CSS is downloaded it can be used throughout

the portal rather than getting different CSS for different Pages. Hence, saving on client data sent as well as client data required.

e. Run and Debug all the code and reduce waste codes:

In this session, we are almost completely done with the major coding part of the SpringBoot application and we only need to minorly adjust the code to make the functioning of the app smooth. We also actively look for any redundant code and make sure we move data we always calculate and mark it with some reference and keep it in memory. This helps save server-side resources and we don't keep calculate things we keep needing again and again. We might also have skipped over some bugs such as a certain one we faced where adding products to cart functioned and so did removing the products would work fine too; on the surface that is. CartMaker which uses data manipulated by CartAdder and CartRemover to send JSON objects and arrays to load the view on client-side using JS would have its functionality to add products broken when products were removed and hence the view on the client-side would not load. The object was unusable by CartAdder after being used by CartRemover due to using String.split() to calculate the document view. It was solved by reiterating the method to not break functionality after being used over and over again by CartAdder and CartRemover than being accessed by CartMaker to build client-side view objects.

f. Test and Iterate code till a certain polish is accomplished:

Testing makes sure that none of the bugs make it out to the finished version of the application hence is very important step. The testing here is done manually which is still better than doing none. All of the outlying border conditions that could be thought of by me and my team including and consisting entirely of me set out to test the application on these border conditions and any inconsistency was ironed out before finalizing the product publishing.

Sprint No.	Tasks	Elapsed	Status
1.	**Plan Flow** of the Back-End, Front-End and Object Persistence.	3 hours	Done
2.	Start coding **classes** and **interfaces**. –- **PART I**	11 hours	Done
3.	Start coding **classes** and **interfaces**. – **PART II**	13 hours	Done

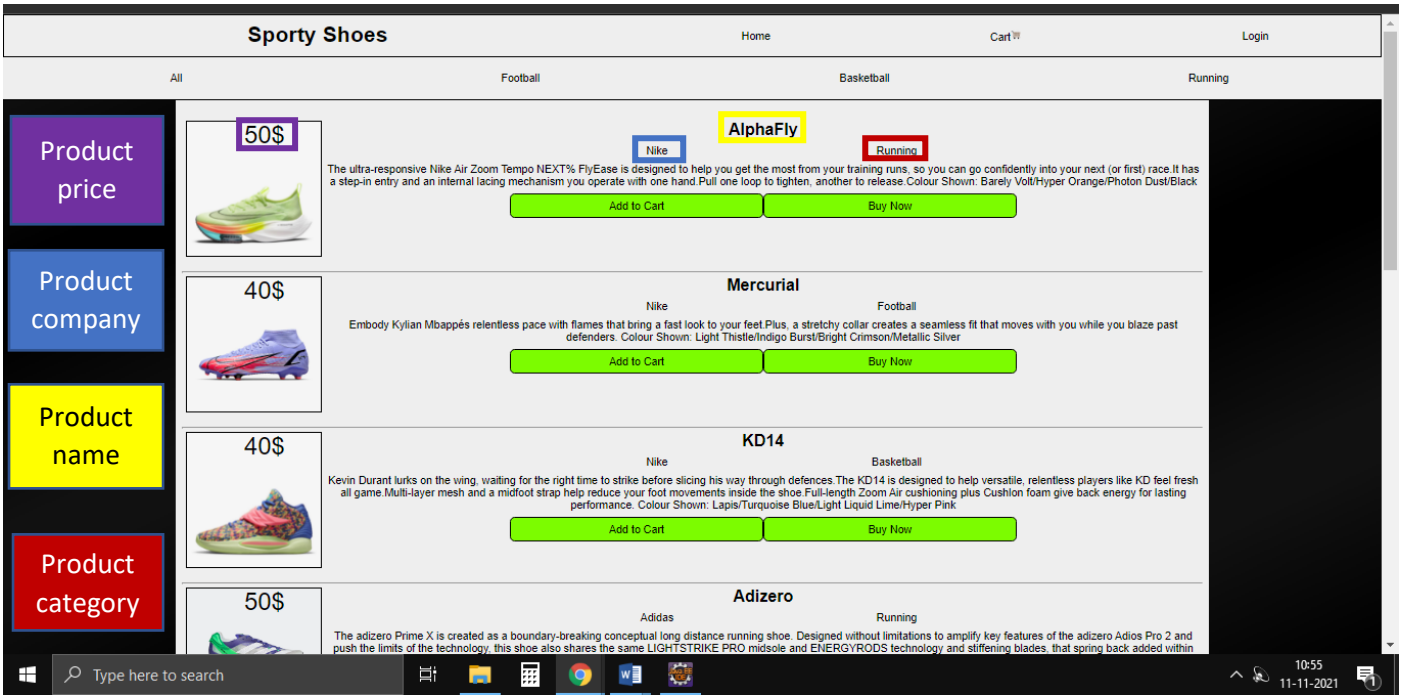
GitHub repository: https://github.com/dedhiah10/SportyShoes.git

4.	**Run**, **debug**, **test** and **reiterate.**	5 hours	Done
6.	**Test** Run App.	2 hours	Done

3) Working of the App:

1) After running the spring boot app:

SportyShoes Home Page:



DataBase SportyShoes initializes values:

GitHub repository: https://github.com/dedhiah10/SportyShoes.git

Run | Run Selected | Auto complete | Clear | SQL statement:

```
SELECT * FROM PRODUCTS
```

PID	COMPANY	INFO	NAME	PRICE	SIZESDB	TYPES
1	Nike	The ultra-responsive Nike Air Zoom Tempo NEXT% FlyEase is designed to help you get the most from your training runs, so you can go confidently into your next (or first) race.It has a step-in entry and an internal lacing mechanism you operate with one hand.Pull one loop to tighten, another to release.Colour Shown: Barely Volt/Hyper Orange/Photon Dust/Black	AlphaFly	50	6_7_8_10	Running
2	Nike	Embody Kylian Mbappés relentless pace with flames that bring a fast look to your feet.Plus, a stretchy collar creates a seamless fit that moves with you while you blaze past defenders. Colour Shown: Light Thistle/Indigo Burst/Bright Crimson/Metallic Silver	Mercurial	40	6_7_8_10	Football
3	Nike	Kevin Durant lurks on the wing, waiting for the right time to strike before slicing his way through defences.The KD14 is designed to help versatile, relentless players like KD feel fresh all game.Multi-layer mesh and a midfoot strap help reduce your foot movements inside the shoe.Full-length Zoom Air cushioning plus Cushlon foam give back energy for lasting performance. Colour Shown: Lapis/Turquoise Blue/Light Liquid Lime/Hyper Pink	KD14	40	6_7_8_10	Basketball
4	Adidas	The adizero Prime X is created as a boundary-breaking conceptual long distance running shoe. Designed without limitations to amplify key features of the adizero Adios Pro 2 and push the limits of the technology, this shoe also shares the same LIGHTSTRIKE PRO midsole and ENERGYRODS technology and stiffening blades, that spring back added within the heel.	Adizero	50	6_7_8_10	Running
5	Adidas	Every time you cross that white line, you enter a whole new world. An alternative reality where you call the shots. The pitch is yours to control. Set your inner freak free in adidas Predator. These football boots have a coated textile upper to help you boss games in comfort. The mid-cut design supports your ankles. A wide expanse of 3D-printed Demonscale elements grip the ball to keep you in charge.	Predator	40	6_7_8_10	Football
6	Adidas	Take control when you step onto the court in these adidas basketball shoes. The mesh upper keeps your feet feeling fresh as you run baseline to baseline. Lightweight cushioning underfoot absorbs impact with every step.	OwnTheGame	40	6_7_8_10	Basketball
7	Puma	Simply put, the Magnify Nitro Mens Running Shoes give you more of the good stuff. First off theres NITRO FOAM in the midsole - thats PUMAs advanced technology designed to provide amazing responsiveness and cushioning. Then theres PUMAGRIP, a durable rubber compound on the outsole that means you get traction on just about any surface. When comfort is the top priority, but you dont want to sacrifice style, these are the shoes to choose.	Nitro	50	6_7_8_10	Running
8	Puma	The PUMA ONE 19.1 Firm Ground/Artificial Ground Mens Football Boots are offering supreme comfort and fit, promoting speed and facilitating ball control. PUMA's evoKNIT technology offers a sock-like feel that adapts to your foot, while the FUSEFIT lacing set-up makes for a fit that is tailored to your foot. The RAPIDSPRINT outsole features a stud configuration perfect for speed, while the soft K-Leather upper ensures traction with the ball.	One	40	6_7_8_10	Football
9	Puma	The Disc Rebirth marks the 30th anniversary of the PUMA Disc System. We have grabbed design details from the original and combined it with modern performance technology to bring the disc back on court in a bigger and better way. Rest assured, you will be shooting hoops in pure PUMA style.	Rebirth	40	6_7_8_10	Basketball

Database product values

SELECT * FROM USER

SELECT * FROM USER;

UID	COUNTRY	DATE_OF_BIRTH	NAME	NUMBER	PASSWORD	ROLE	ZIP_CODE
1	India	1991-02-02	admin	1234567890	admin	TRUE	54321
2	India	1992-05-04	user	9876543210	user	FALSE	65432

(2 rows, 2 ms)

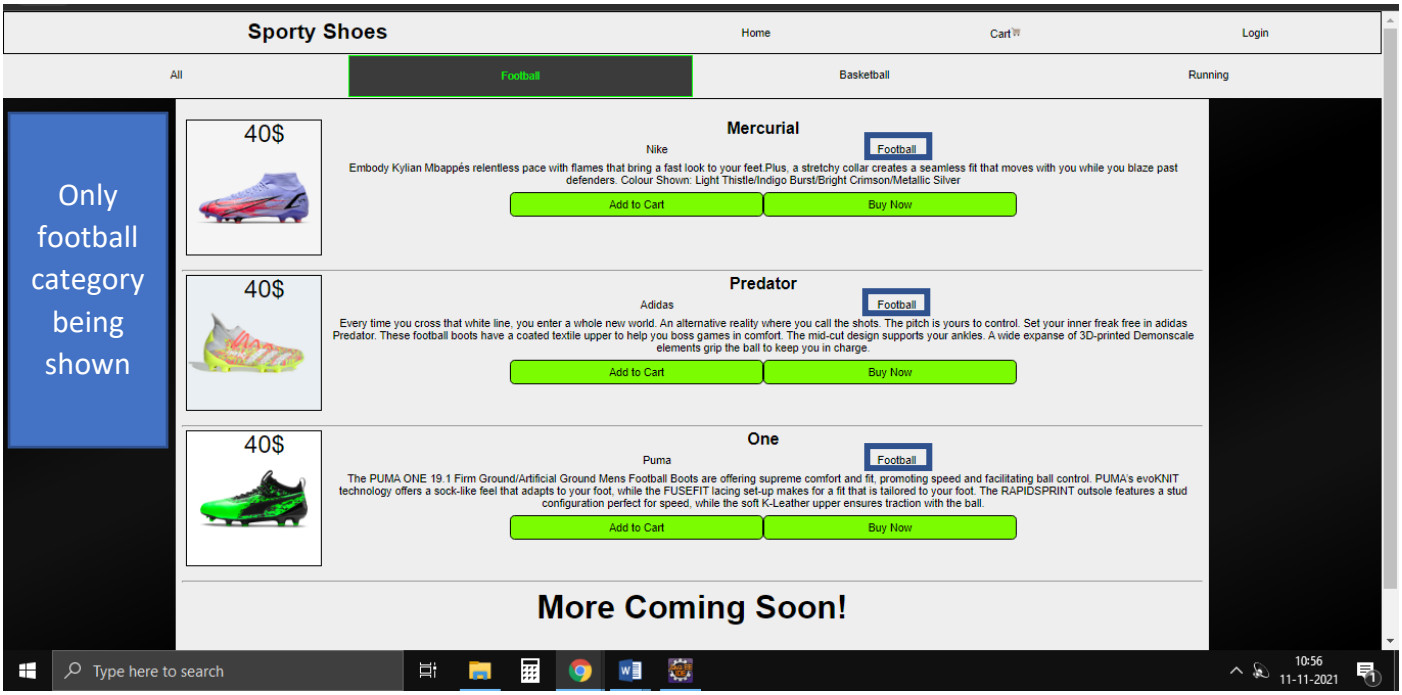
Edit

Database user values

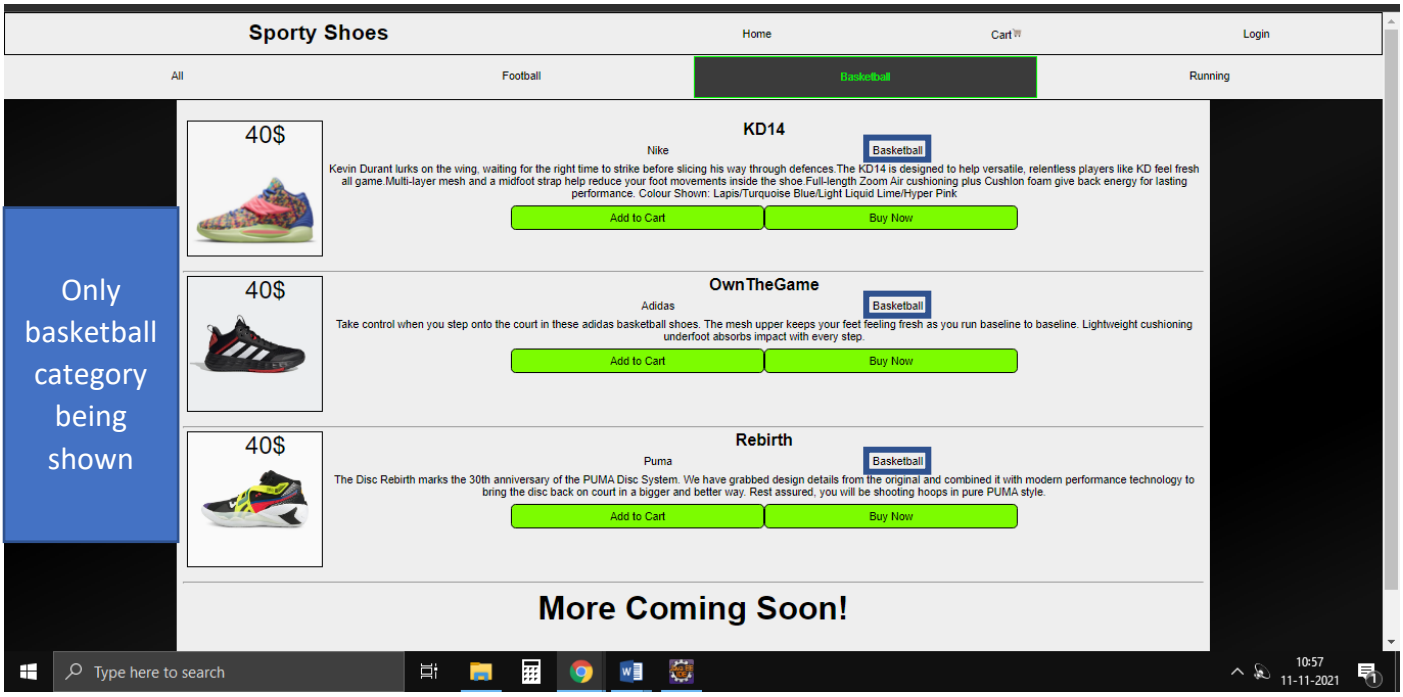
2) Interacting the Home Page:

Using sorting buttons provided:

GitHub repository: https://github.com/dedhiah10/SportyShoes.git



Sorting football products



Sorting basketball products

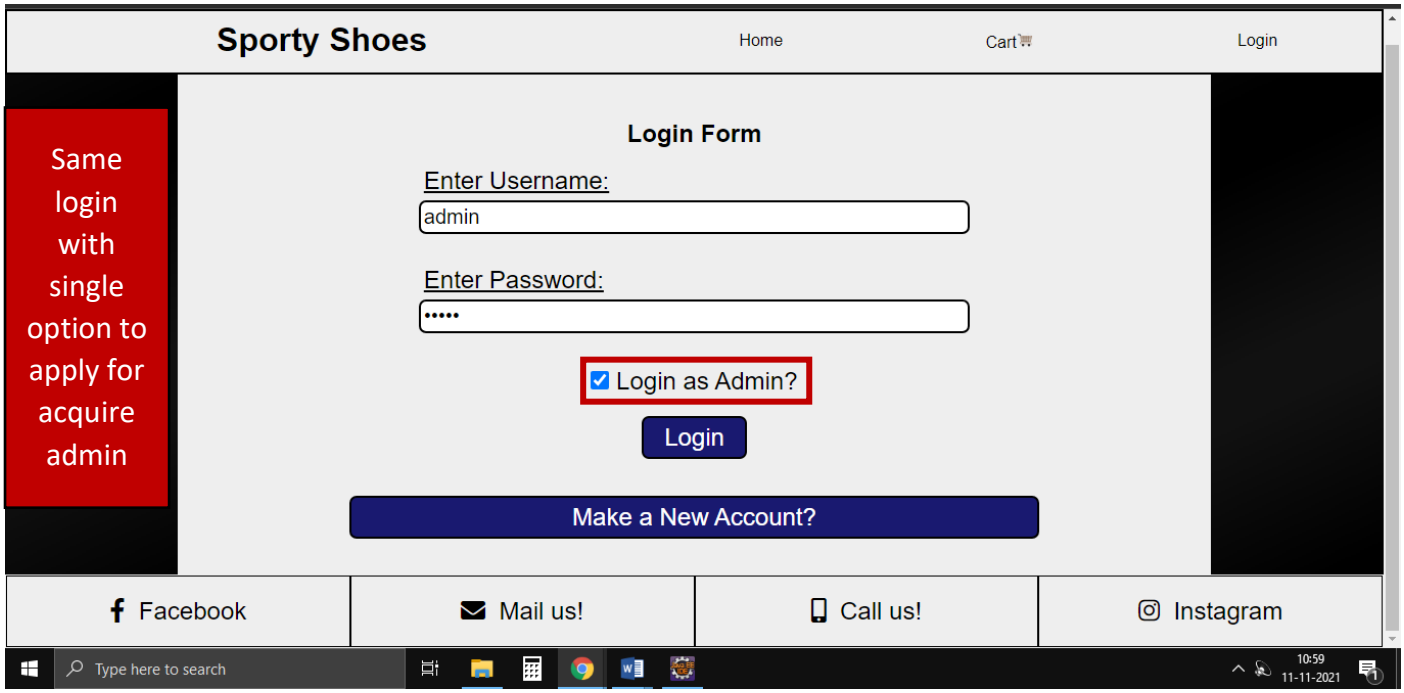
GitHub repository: https://github.com/dedhiah10/SportyShoes.git

Sorting running products

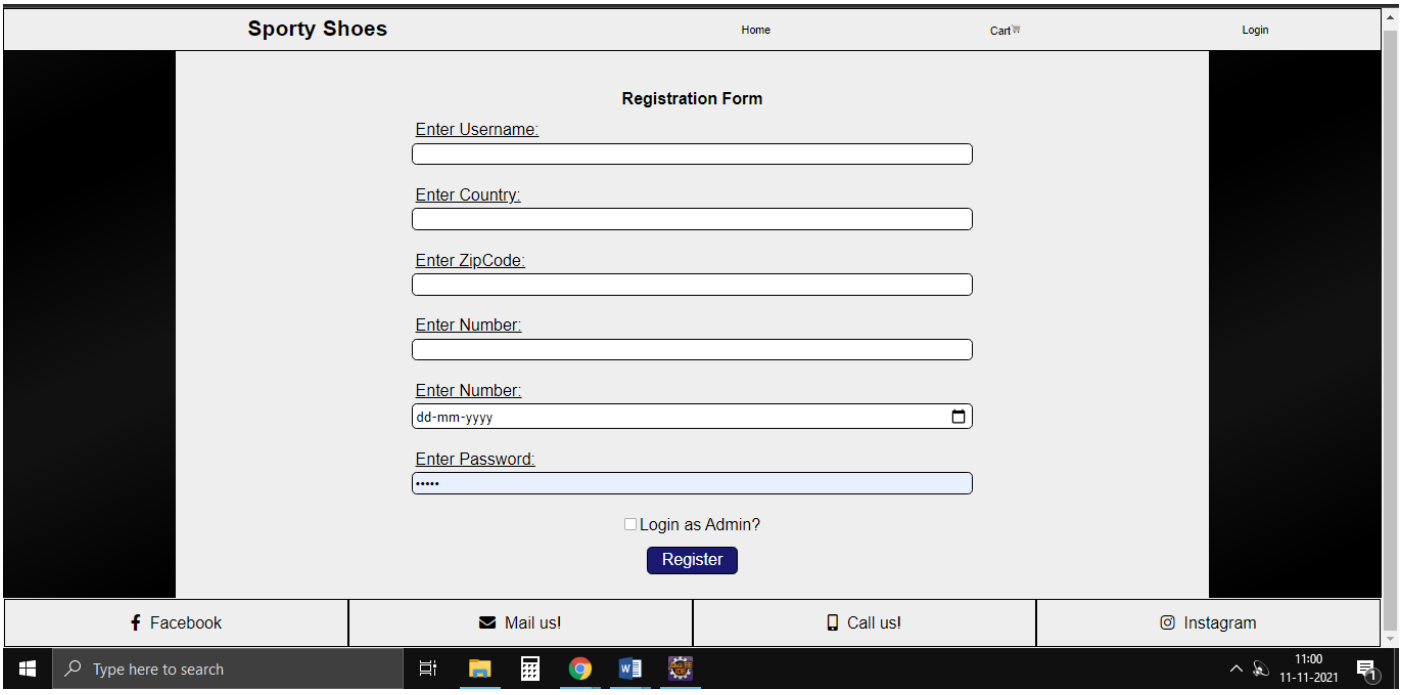
3) Login and Registration:

Login Page:

GitHub repository: https://github.com/dedhiah10/SportyShoes.git



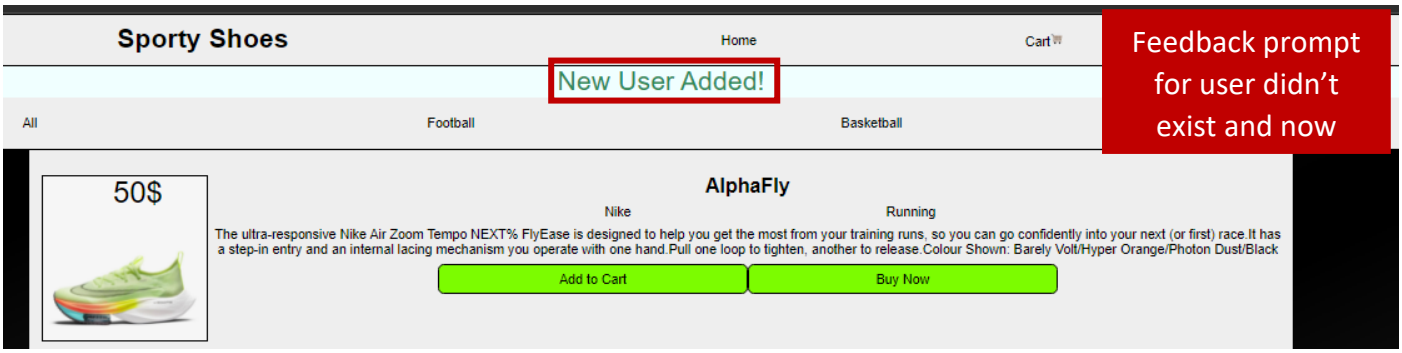
Registration Page:



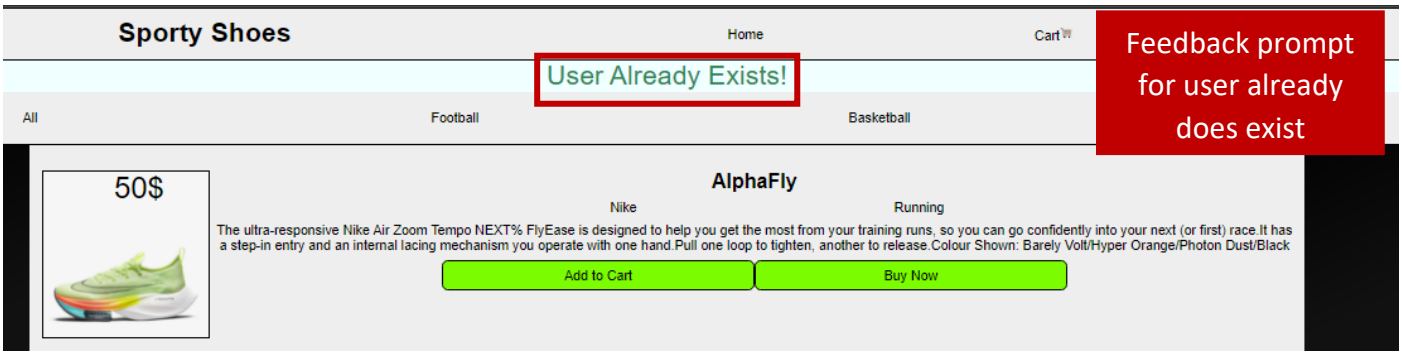
4) Registering:

GitHub repository: https://github.com/dedhiah10/SportyShoes.git

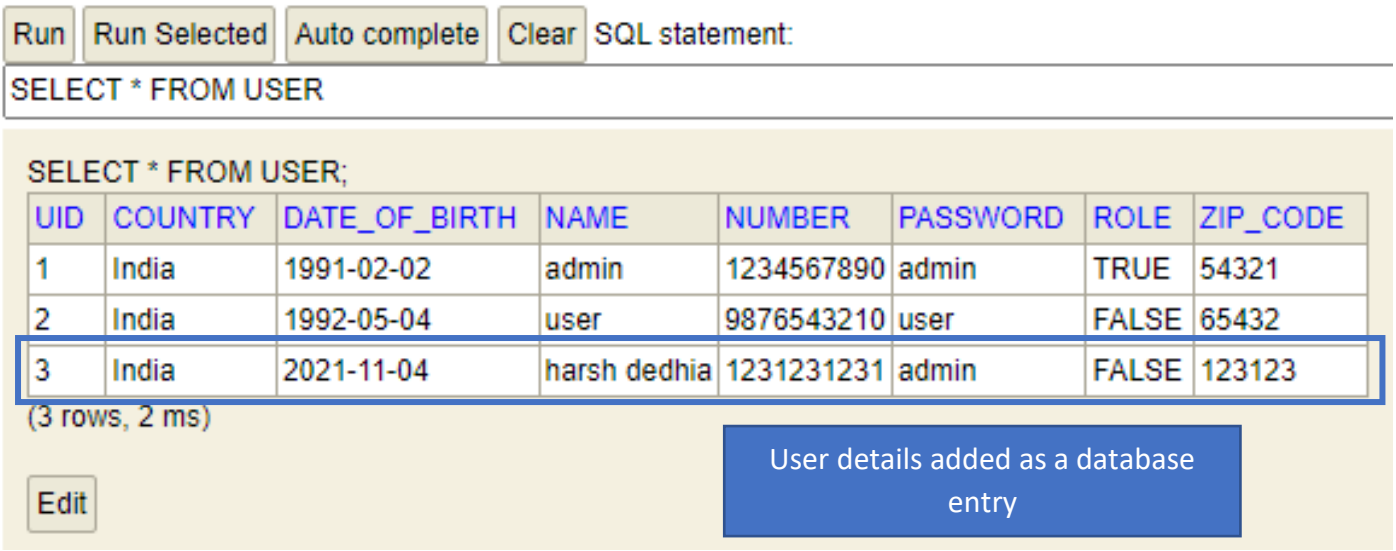
User Added prompt:



User Already Exists prompt:



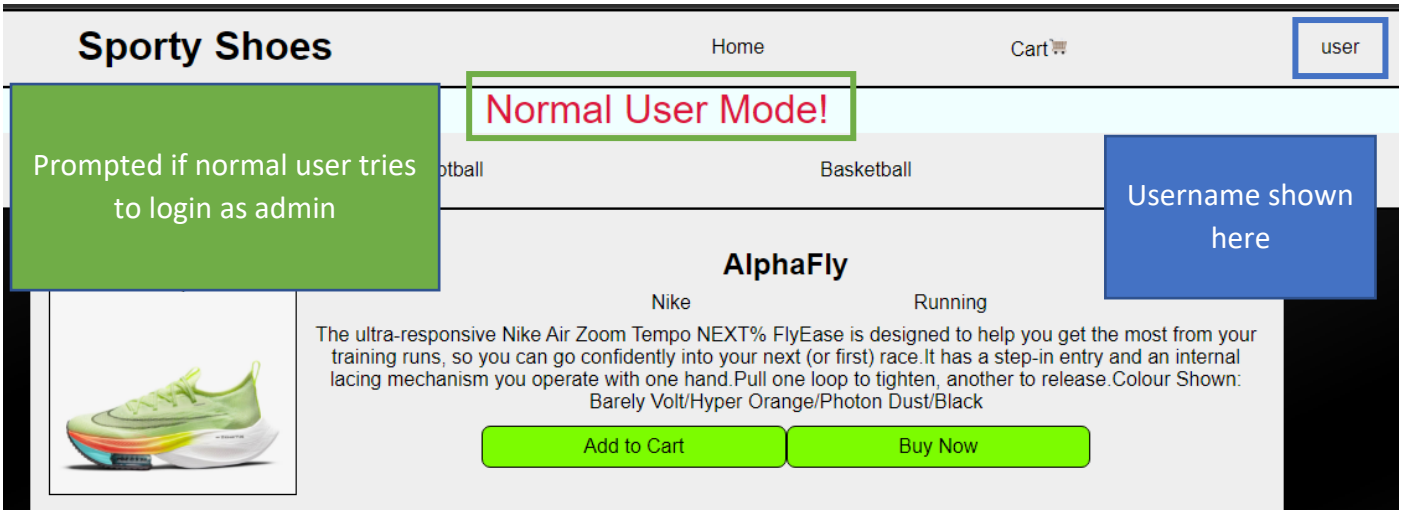
Database user values change:



5) Logging in:

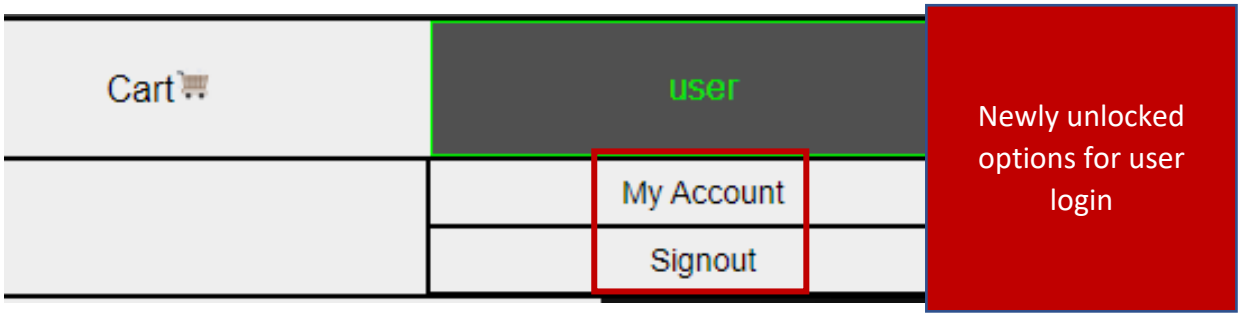
Normal user tries to login as admin:

GitHub repository: https://github.com/dedhiah10/SportyShoes.git

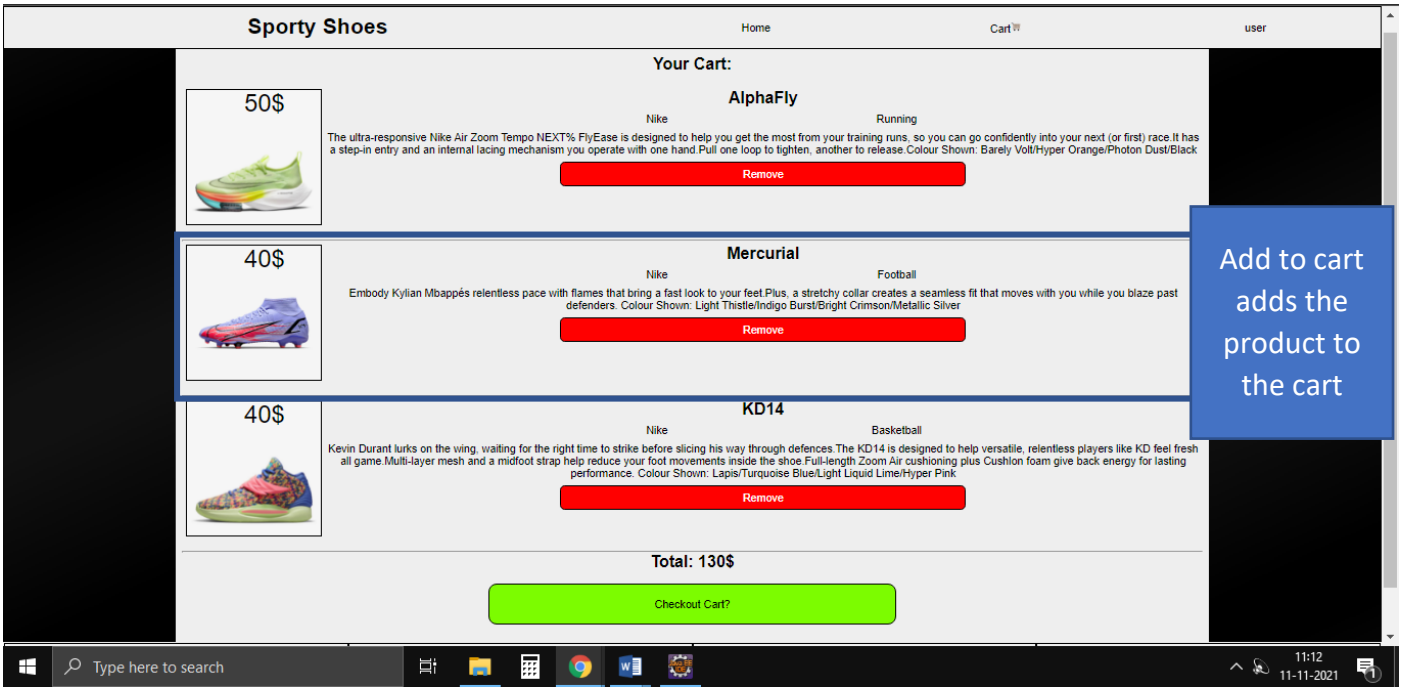


Login user type prompt

Normal user options unlock:

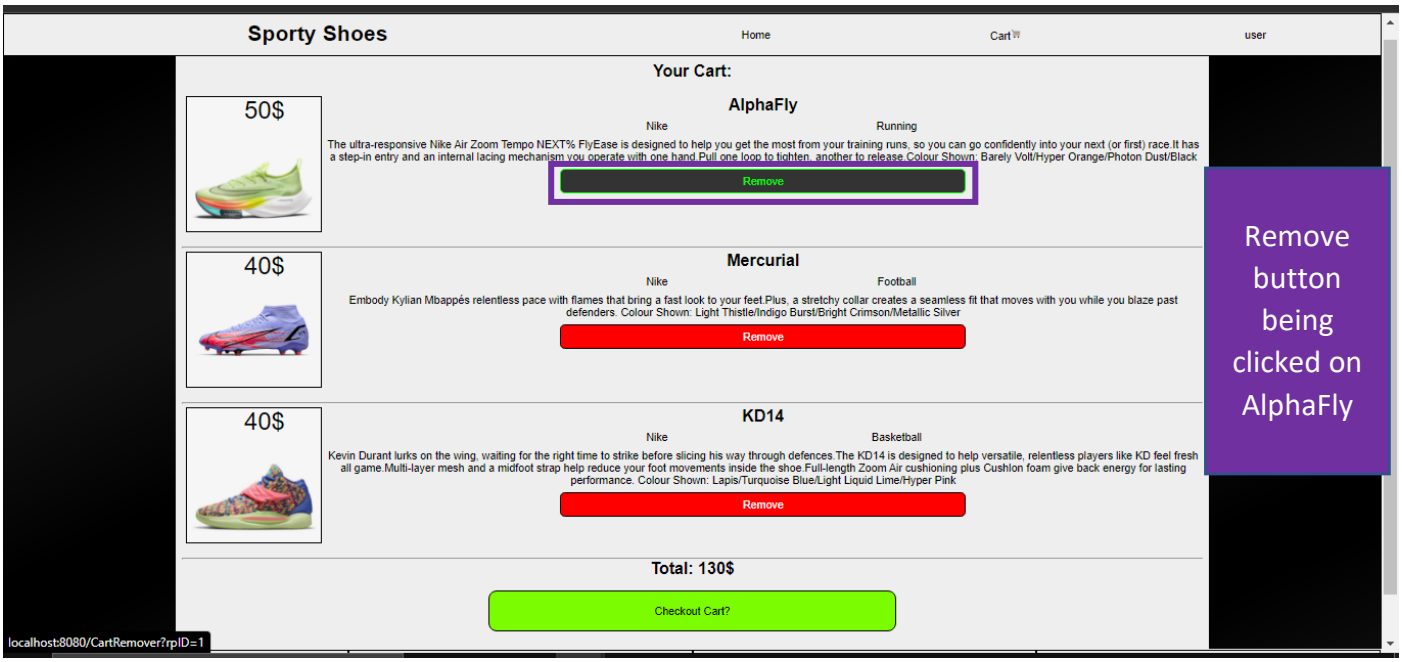


User cart after adding to cart:

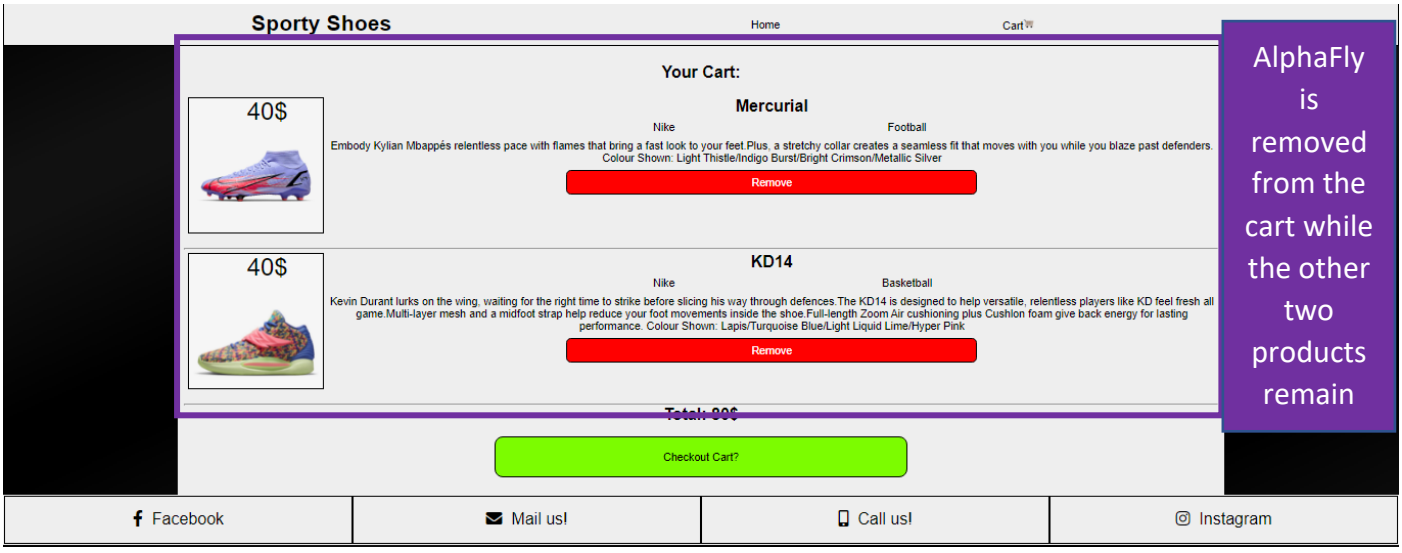


GitHub repository: https://github.com/dedhiah10/SportyShoes.git

Remove option in cart:



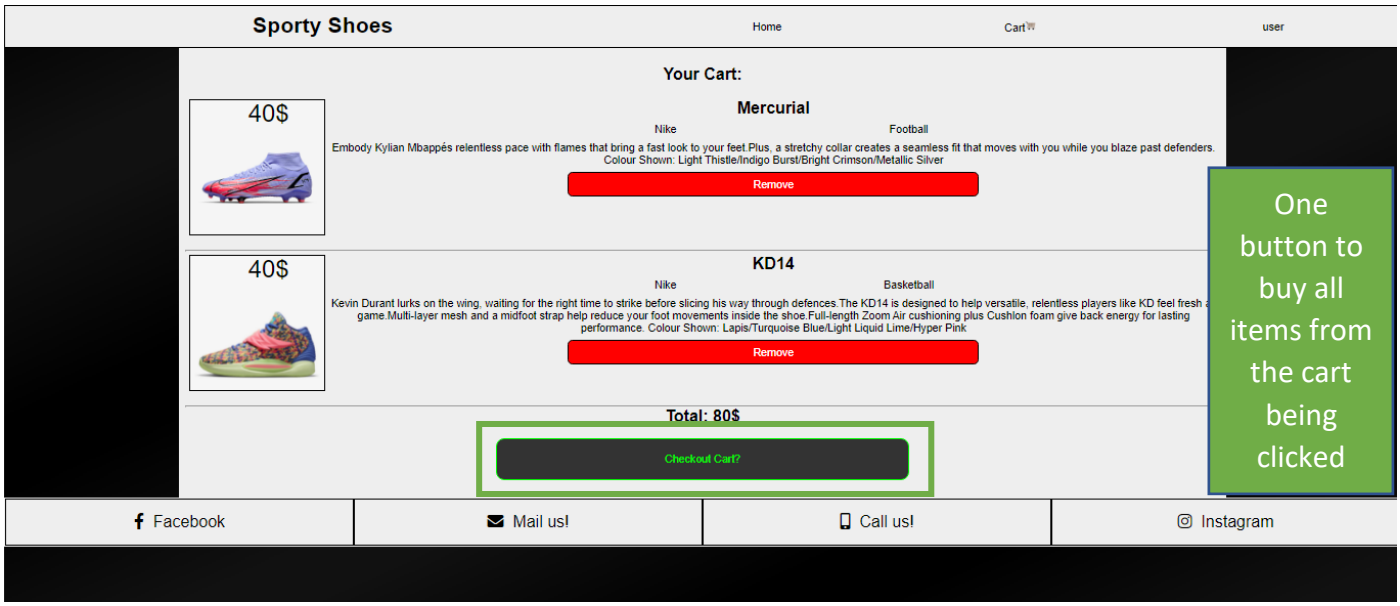
Before clicking



After clicking

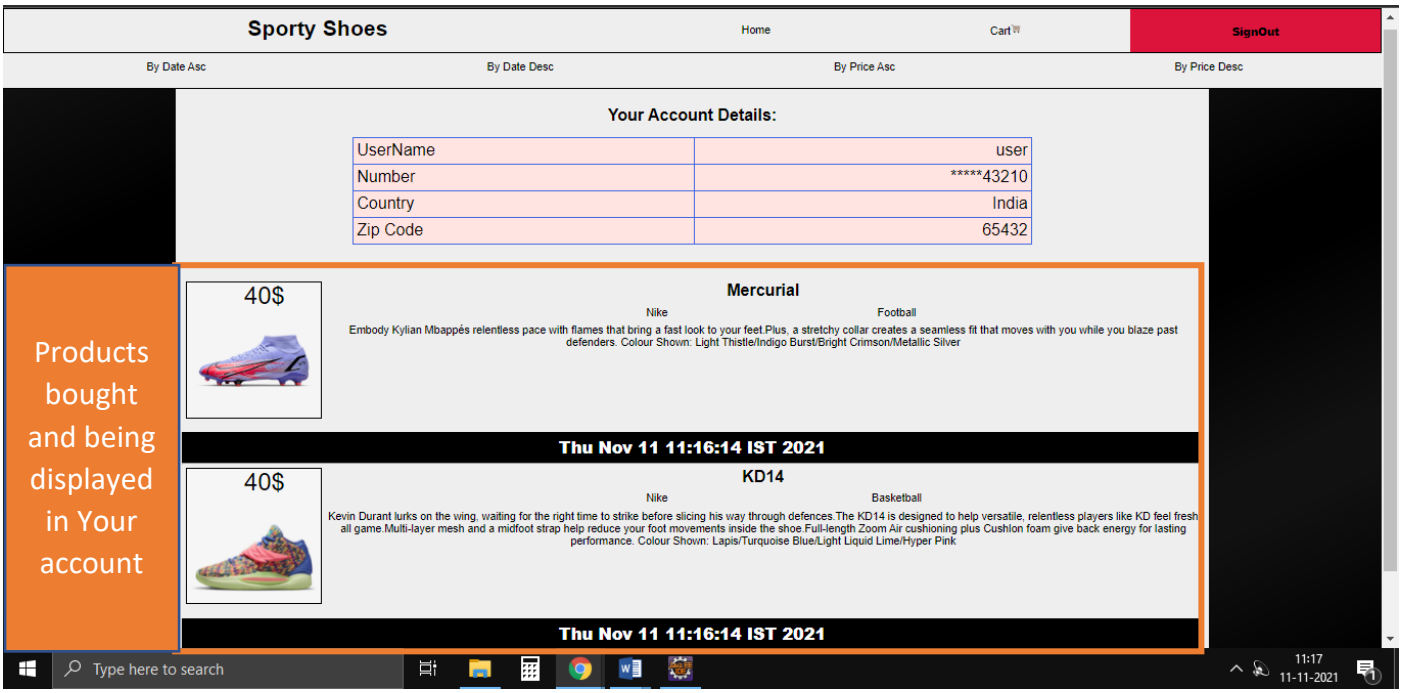
GitHub repository: https://github.com/dedhiah10/SportyShoes.git

Checkout Cart option:



Before clicking

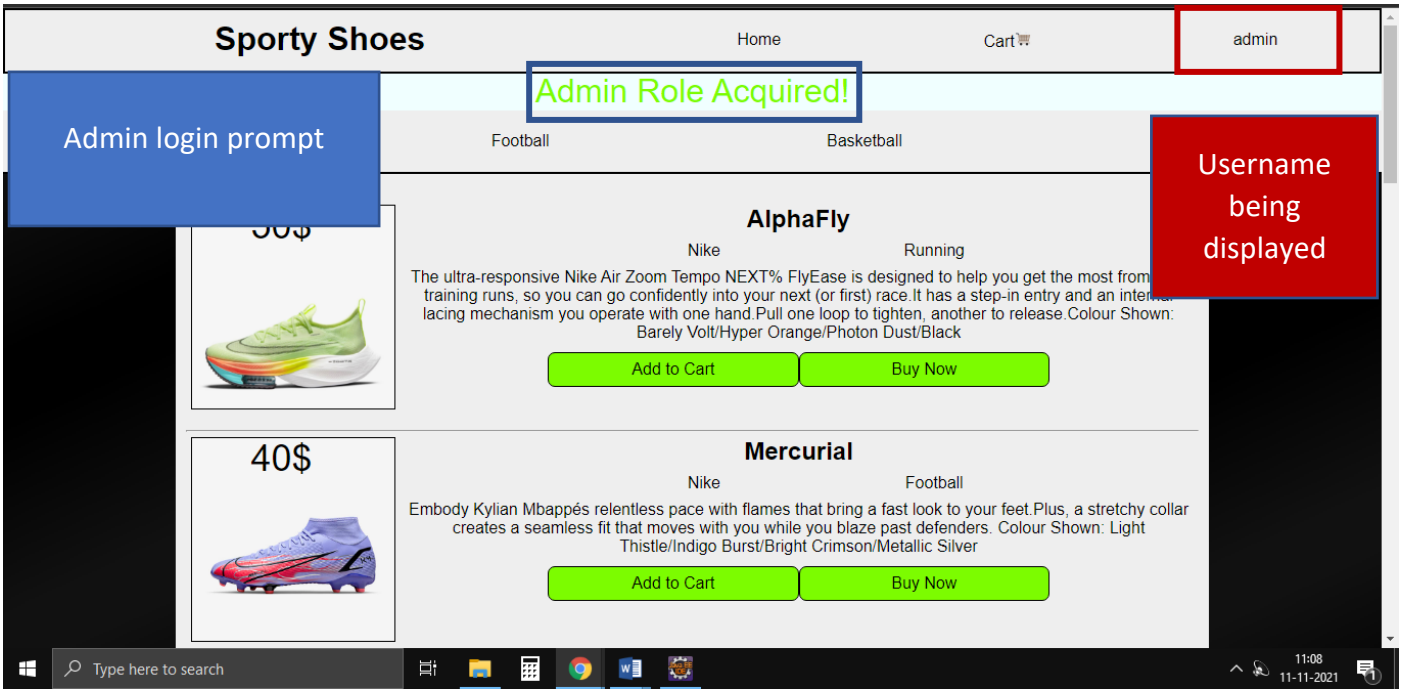
Your account details option:



After clicking

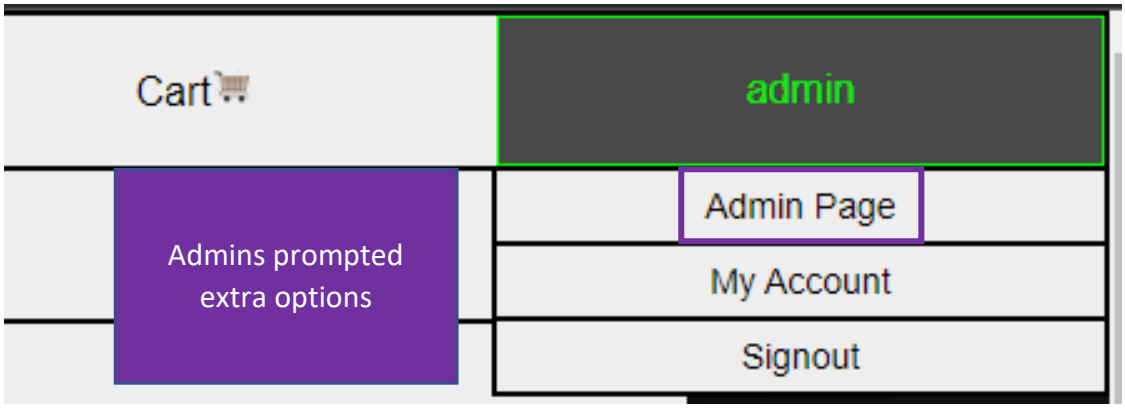
GitHub repository: https://github.com/dedhiah10/SportyShoes.git

Admin tries to login:



Login admin type prompt

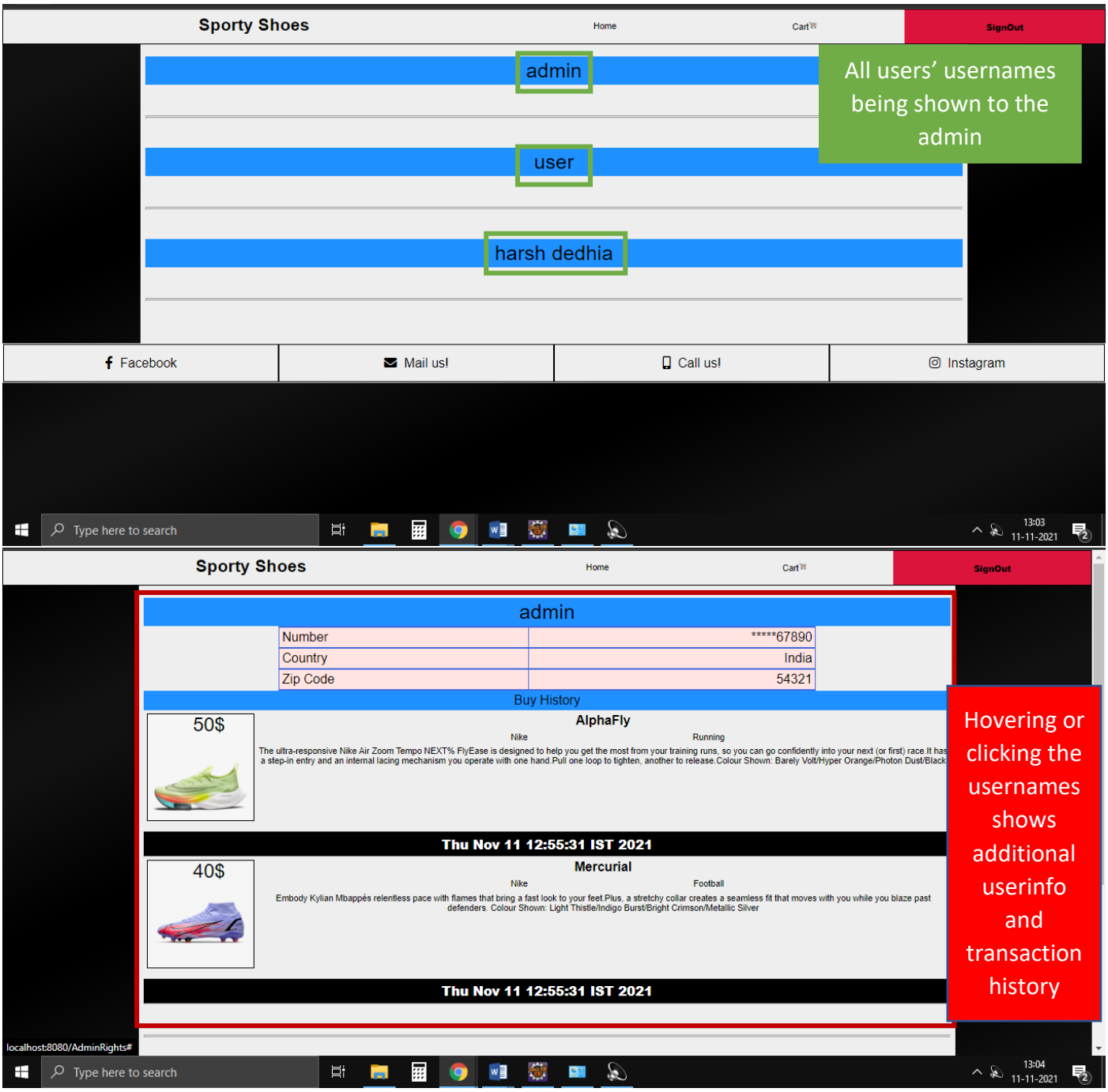
Admin options unlock:



Admin page unlocks for admin login

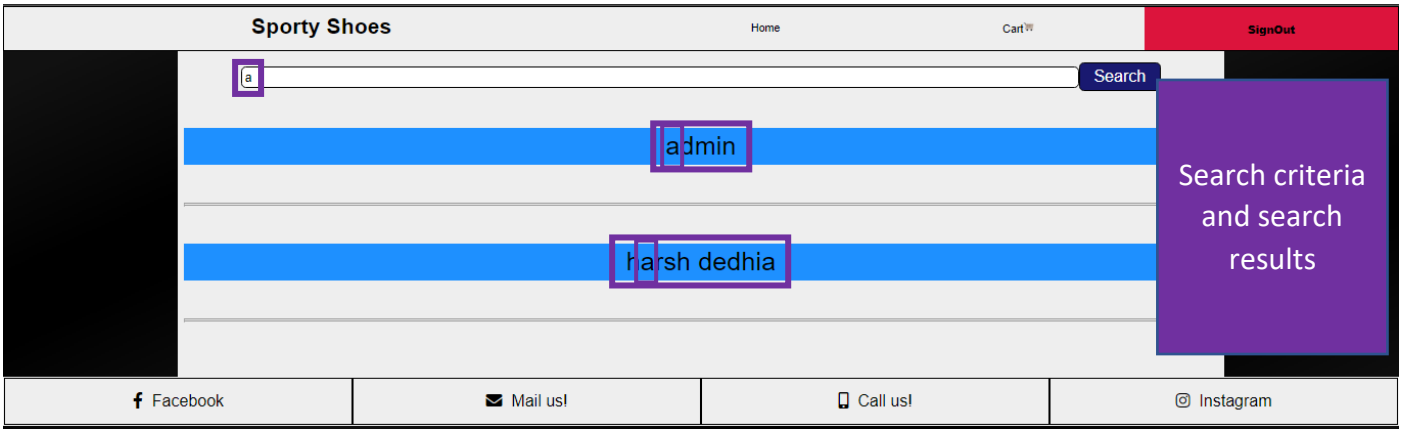
GitHub repository: https://github.com/dedhiah10/SportyShoes.git

Admin Page view:



Username <u>admin</u> upon being clicked shows user details and transactions

Admin Page view:



GitHub repository: https://github.com/dedhiah10/SportyShoes.git

6) Git and GitHub:

Now we link git on PC and git repository on git hub using git bash. Then we use git commit and git push commands to upload to repository and give a commit message.

```
HD@HD MINGW64 ~/Desktop/proj/SL/SportyShoes
$ git init
Initialized empty Git repository in C:/Users/HD/Desktop/proj/SL/SportyShoes/.git
/

HD@HD MINGW64 ~/Desktop/proj/SL/SportyShoes (master)
$ git add .
warning: LF will be replaced by CRLF in TheSpringBootApp/SportyShoes/.gitignore.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in TheSpringBootApp/SportyShoes/.mvn/wrappe
r/MavenWrapperDownloader.java.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in TheSpringBootApp/SportyShoes/.mvn/wrappe
r/maven-wrapper.properties.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in TheSpringBootApp/SportyShoes/mvnw.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in TheSpringBootApp/SportyShoes/mvnw.cmd.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in TheSpringBootApp/SportyShoes/pom.xml.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in TheSpringBootApp/SportyShoes/src/main/ja
va/com/SL/SportyShoes/SportyShoesApplication.java.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in TheSpringBootApp/SportyShoes/src/main/re
sources/application.properties.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in TheSpringBootApp/SportyShoes/src/test/ja
va/com/SL/SportyShoes/SportyShoesApplicationTests.java.
The file will have its original line endings in your working directory

HD@HD MINGW64 ~/Desktop/proj/SL/SportyShoes (master)
$ git commit -m "First iteration commit"
[master (root-commit) 7eaa0c8] First iteration commit
 57 files changed, 2398 insertions(+)
 create mode 100644 SportyShoes.docx
 create mode 100644 SportyShoes.pdf
 create mode 100644 TheSpringBootApp/SportyShoes/.gitignore
 create mode 100644 TheSpringBootApp/SportyShoes/.mvn/wrapper/MavenWrapperDownlo
ader.java
 create mode 100644 TheSpringBootApp/SportyShoes/.mvn/wrapper/maven-wrapper.jar
 create mode 100644 TheSpringBootApp/SportyShoes/.mvn/wrapper/maven-wrapper.prop
erties
 create mode 100644 TheSpringBootApp/SportyShoes/mvnw
 create mode 100644 TheSpringBootApp/SportyShoes/mvnw.cmd
 create mode 100644 TheSpringBootApp/SportyShoes/pom.xml
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/java/com/SL/SportyShoe
s/DatabaseComms/ProductRepository.java
```

GitHub repository: https://github.com/dedhiah10/SportyShoes.git

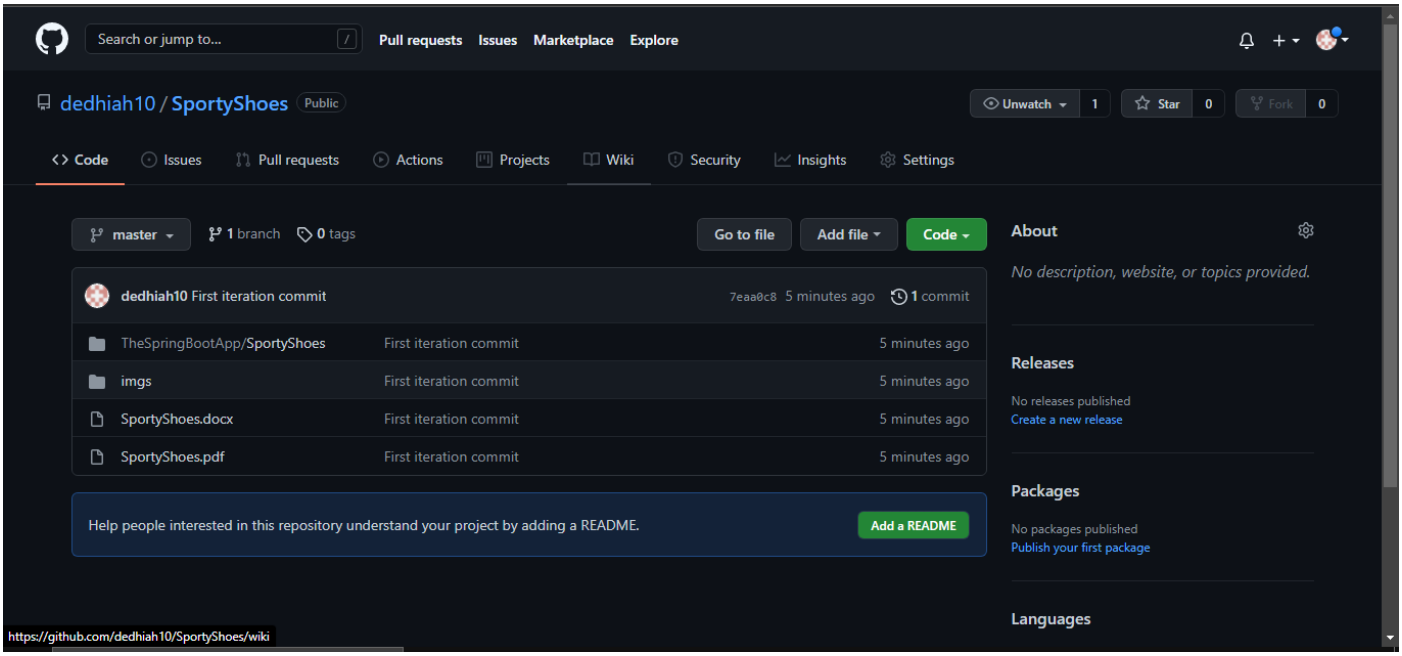
```
s/DatabaseComms/ProductRepository.java
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/java/com/SL/SportyShoe
s/DatabaseComms/UserRepository.java
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/java/com/SL/SportyShoe
s/Entities/Products.java
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/java/com/SL/SportyShoe
s/Entities/User.java
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/java/com/SL/SportyShoe
s/ProductController/ProductController.java
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/java/com/SL/SportyShoe
s/SportyShoesApplication.java
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/resources/application.
properties
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/resources/data.sql
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/CartItems.jsp
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Adidas-B
asketball-OwnTheGame.jpg
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Adidas-F
ootball-Predator.jpg
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Adidas-R
unning-Adizero.jpg
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Cart_Ico
n.png
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Nike-Bas
ketball-KD14.jpg
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Nike-Foo
tball-Mercurial.jpg
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Nike-Run
ning-AlphaFly.jpg
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Puma-Bas
ketball-Rebirth.jpg
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Puma-Foo
tball-One.jpg
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Images/Puma-Run
ning-Nitro.jpg
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Index.jsp
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Login.jsp
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/MyAccount.jsp
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Styles/Home.css
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/Transactions.js
p
 create mode 100644 TheSpringBootApp/SportyShoes/src/main/webapp/UserForm.jsp
 create mode 100644 TheSpringBootApp/SportyShoes/src/test/java/com/SL/SportyShoe
s/SportyShoesApplicationTests.java
 create mode 100644 imgs/AdminLoginAsAdmin.PNG
 create mode 100644 imgs/AdminRightsView.PNG
 create mode 100644 imgs/AfterClickingRemoveinCartManager.PNG
 create mode 100644 imgs/BeforeClickingCheckoutCart.PNG
 create mode 100644 imgs/BeforeClickingRemoveinCartManager.PNG
```

GitHub repository: https://github.com/dedhiah10/SportyShoes.git

```
 create mode 100644 imgs/BeforeClickingRemoveinCartManager.PNG
 create mode 100644 imgs/DBInitiatedRandomProductsList.PNG
 create mode 100644 imgs/DBInitiatedRandomUsersList.PNG
 create mode 100644 imgs/LoginPage.PNG
 create mode 100644 imgs/NormalUserLoginAsAdmin.PNG
 create mode 100644 imgs/RegisteredUserAddedtoDB.PNG
 create mode 100644 imgs/RegistrationPage.PNG
 create mode 100644 imgs/ShowUsersInfotoAdmin.PNG
 create mode 100644 imgs/SportShoesAfterSelectingSortingCategoryBasketball.PNG
 create mode 100644 imgs/SportShoesAfterSelectingSortingCategoryFootball.PNG
 create mode 100644 imgs/SportShoesAfterSelectingSortingCategoryRunning.PNG
 create mode 100644 imgs/SportyShoesHomePage.PNG
 create mode 100644 imgs/UserAddedPrompt.PNG
 create mode 100644 imgs/UserCartManager.PNG
 create mode 100644 imgs/UserExistsPrompt.PNG
 create mode 100644 imgs/UserNewOptionsforAdmin.PNG
 create mode 100644 imgs/UserNewOptionsforUser.PNG
 create mode 100644 imgs/YourAccountAfterClickingCheckoutCart.PNG

HD@HD MINGW64 ~/Desktop/proj/SL/SportyShoes (master)
$ git remote add origin https://github.com/dedhiah10/SportyShoes.git

HD@HD MINGW64 ~/Desktop/proj/SL/SportyShoes (master)
$ git push -u origin master
Enumerating objects: 82, done.
Counting objects: 100% (82/82), done.
Delta compression using up to 4 threads
Compressing objects: 100% (72/72), done.
Writing objects: 100% (82/82), 4.06 MiB | 991.00 KiB/s, done.
Total 82 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), done.
To https://github.com/dedhiah10/SportyShoes.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```



The end

GitHub repository: https://github.com/dedhiah10/SportyShoes.git