Assignment -3

|  |  |
| --- | --- |
| **Student Name/ID Number:** | Francis Roel L. Abarca | bdse-0922-113 |
| **Academic Year:** | 2022-2023 |
| **Unit Assessor:** | Archana Sakpal |
| **Project Title:** | Assignment 3 - |
| **Issue Date:** | 4/13/2023 |
| **Submission Date:** | 4/13/2023 |
| **Internal Verifier Name:** | Archana Sakpal |
| **Date:** | 4/13/2023 |

|  |
| --- |
| **Learner declaration** |
| I certify that the work submitted for this assignment is my own and research sources are fully acknowledged.    4/13/2023  Student signature: Date: |

Let’s build on the previous assignment. (Develop “Know-Your-Neighbourhood”) application. The goal of this application is to provide details on all stores in the user’s neighbourhood.

1. **Create a Spring Boot application for “Know-Your-Neighbourhood”.**

Inside Spring Tool Suite 3, click on File then New then Spring Starter Project.

Graphical user interface, application

Description automatically generated

Inside the Spring Starter Project form, fill out necessary information but make sure you set:  
- Type: Maven   
- Java Version: 17  
then click next.

Graphical user interface, text, application

Description automatically generated

After that, look for Spring Web inside the Spring Starter Project Dependencies then click Finish.

Graphical user interface, text, application

Description automatically generated

After your project has been created, open pom.xml and kindly add these required dependencies.   
   
 - Here’s what it should look like after:  


<dependency>

<groupId>org.apache.tomcat.embed</groupId>

<artifactId>tomcat-embed-jasper</artifactId>

</dependency>

<dependency>

<groupId>org.glassfish.web</groupId>

<artifactId>jakarta.servlet.jsp.jstl</artifactId>

<version>2.0.0</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>4.0.1</version>

<scope>provided</scope>

</dependency>

After importing those dependencies above, expand src/main/resources then open application.properties.

Graphical user interface, text, application, chat or text message

Description automatically generated

Lastly, inside application.properties, add these following lines of code:   
  
 - Spring.mvc.view.prefix points to the /views/ directory inside the project which contains your .jsp files.  
 - Spring.mvc.view.suffix means that it should only be reading files with the .jsp as its filename.  
 - Server.port refers to the port that your web server is going to use to host the project.

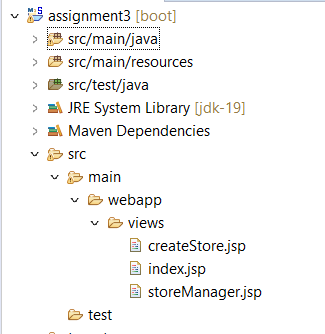
spring.mvc.view.prefix=/views/

spring.mvc.view.suffix=.jsp

server.port=9092

1. **Add support for JSP views and create required folder structure.**

After the Spring Starter Project has created, make sure you create a folder inside src/main as webapp then create a folder named views inside webapp then lastly, import your previous KnowYourNeighborhood jsp files inside src/main/webapp/views folder.



1. **Move already developed classes into this project.**

After importing those jsp files, you will need to copy the .java classes from the previous KnowYourNeighborhood project package then copy them all to the new package inside which for this project, it’s enclosed in com.yeems214.assignment3

Graphical user interface, text, application, chat or text message

Description automatically generated

1. **Develop all components required to view the stores** 
   1. **Add method to existing Controller class to receive the request to fetch stores**

Text

Description automatically generated

* 1. **Add method to existing Service class to process the request (or add method to existing service class)**

Graphical user interface, text, application

Description automatically generated

* 1. **Add method existing repository class to return all available stores.**

Graphical user interface, text, application, email

Description automatically generated

* 1. **Create HTML to view the stores. Show name, phone number and localities it serves for each store.**

Text

Description automatically generated

* 1. **Ensure that view stores request works end-to-end. (i.e., should be able to submit request to view the stores in the browser and get the page back with all stores).**
     + **No Data**

Graphical user interface, text, application

Description automatically generated

* + - **Adding Data**

**Graphical user interface, application

Description automatically generated**

* + - **Store Manager with Data**

**Graphical user interface, text, application, email

Description automatically generated**

1. **Create an HTML page to add a store and link it to the view stores page.**

Text

Description automatically generated

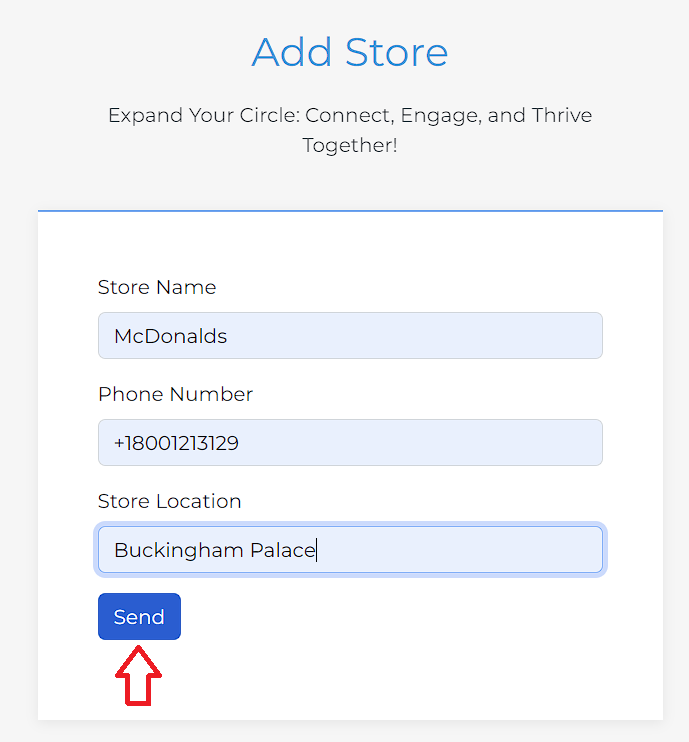
1. **Let the response of add request submission be the view page that shows all stores including recently added store.**

After loading into the page, go to Add then enter your new store data (Name, Phone Number, Localities)

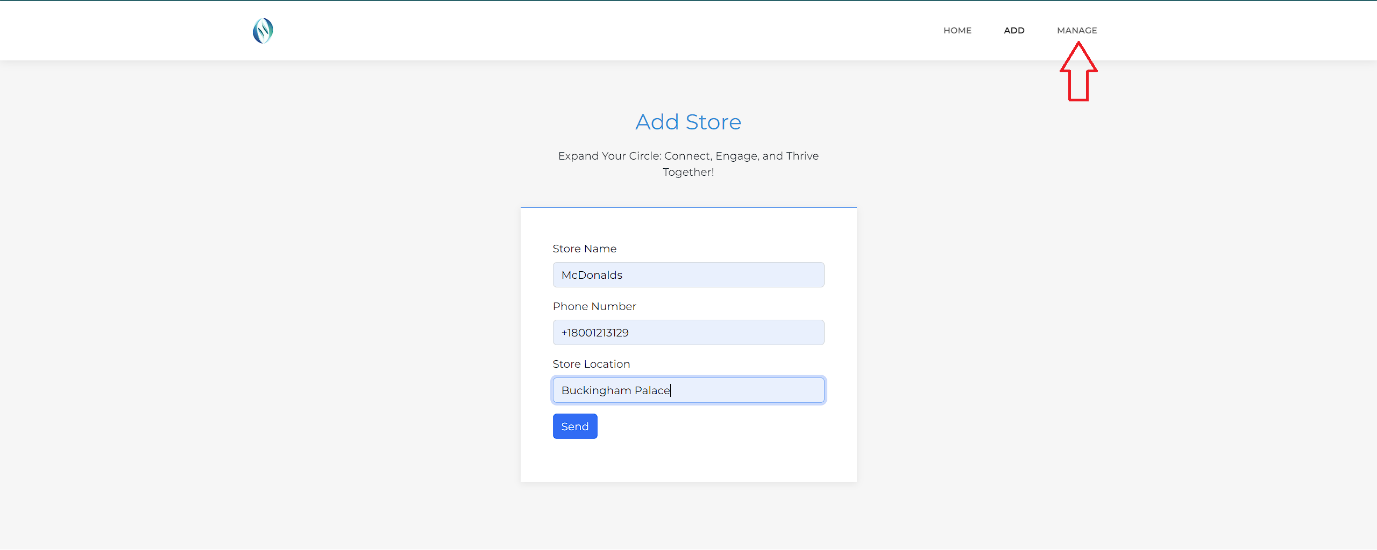
**Graphical user interface, application

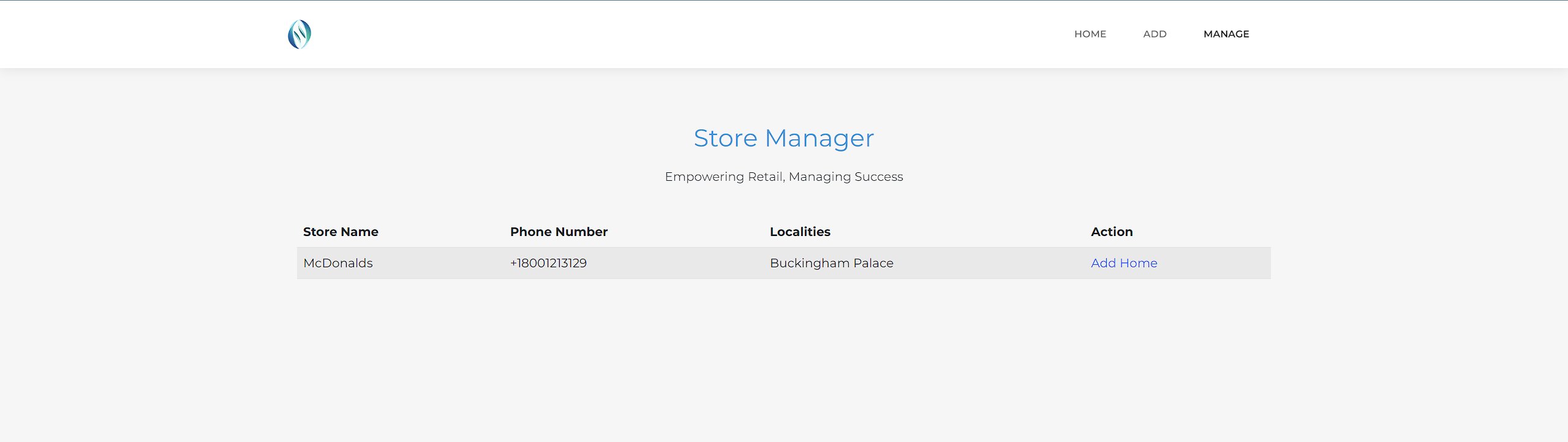
Description automatically generated**

Submit the store form after completing.

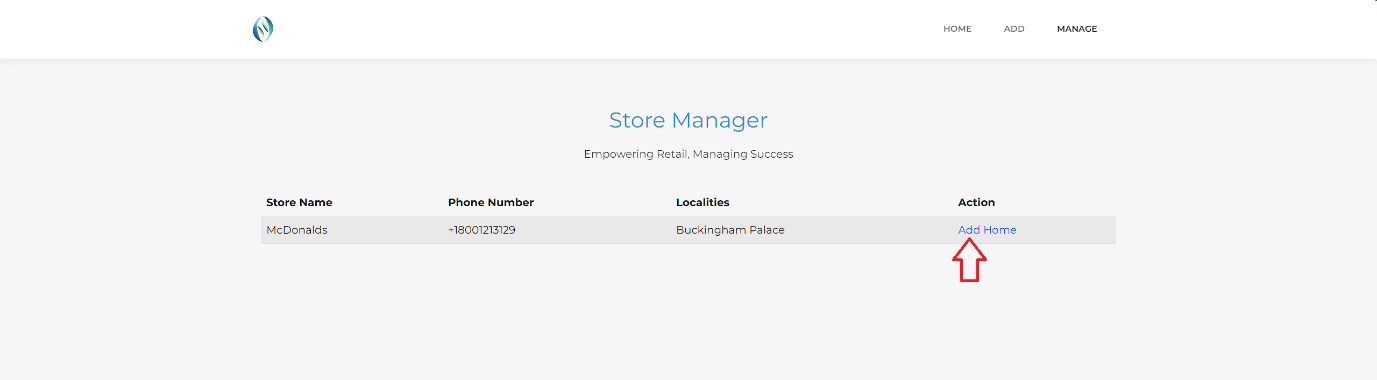


After that, you will need to go to the Store Manager page.





Click the “Add” button inside Store Manager to add a new store.



After adding another store, go back to the Store Manager page which then shows the newly added store.

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated