**Foodbox Project**

DESCRIPTION

Create a dynamic and responsive online food delivery web application for ordering food items of different cuisines from a restaurant.

**Background of the problem statement:**  
Foodbox is a restaurant chain that delivers food items of different cuisines at affordable prices. It was established in 2014 in Bengaluru, India. It had been serving fine all these years, however, the business analysts noticed a decline in sales since 2016. They found out that the online ordering of food items with companies, such as Swiggy and Foodpanda were gaining more profit by eliminating middlemen from the equation. As a result, the team decided to hire a Full Stack developer to develop an online food delivery web application with a rich and user-friendly interface.  
You are hired as the Full Stack Java developer and are asked to develop the web application. The management team has provided you with the requirements and their business model so that you can easily arrange different components of the application.

**Features of the application:**

1. Registration
2. Login
3. Payment gateway
4. Searching
5. Filtering
6. Sorting
7. Dynamic data
8. Responsive and compatible with different devices

**Recommended technologies:**

1. Database management: MySQL and Oracle
2. Backend logic: Java programming, NodeJS
3. Frontend development: JSP, Angular, Bootstrap, HTML/CSS, and Javascript
4. Automation and testing technologies: Selenium, Jasmine, and TestNG
5. DevOps and production technologies: Git, GitHub, Jenkins, Docker, Kubernetes, and AWS

**Project development guidelines:**

* The project will be delivered within four sprints with every sprint delivering a minimal viable product.
* It is mandatory to perform proper sprint planning with user stories to develop all the components of the project.
* The learner can use any technology from the above-mentioned technologies for different layers of the project.
* The web application should be responsive and should fetch or send data dynamically without hardcoded values.
* The learner must maintain the version of the application over GitHub and every new change should be sent to the repository.
* The learner must implement a CI/CD pipeline using Jenkins.
* The learner should also deploy and host the application on an AWS EC2 instance.
* The learner should also implement automation testing before the application enters the CI/CD pipeline.
* The learner should use Git branching to do basic automation testing of the application in it separately.
* The learner should make a rich frontend of the application, which is user- friendly and easy for the user to navigate through the application.
* There will be two portals in the application, namely admin and user portal.

**Admin Portal:**  
The admin portal deals with all the backend data generation and product information. The admin user should be able to:

* Add or remove different cuisines to or from the application to build a rich product line
* Edit food item details like name, price, cuisine, description, and offers to keep it aligned to the current prices
* Enable or disable the food items

**User Portal:**  
It deals with the user activities. The end-user should be able to:

* Sign-in to the application to maintain a record of activities
* Search for food items based on the search keyword
* Apply filters and sort results based on different cuisines to get the best deals
* Add all the selected food items to a cart and customize the purchase at the end
* Perform a seamless payment process
* Get an order summary details page once the payment is complete

## **Table of contents**

* General info
* Programming and deploying the application
* Technologies
* How to run the program locally
* Using the application

## **General info**

This project aims to design and develop an E-commerce website that lets people shop food items of different cuisines at affordable prices and deliver the products to their addresses. It was developed using Angular and Spring boot for the Full Stack Java Developer course as capstone project.

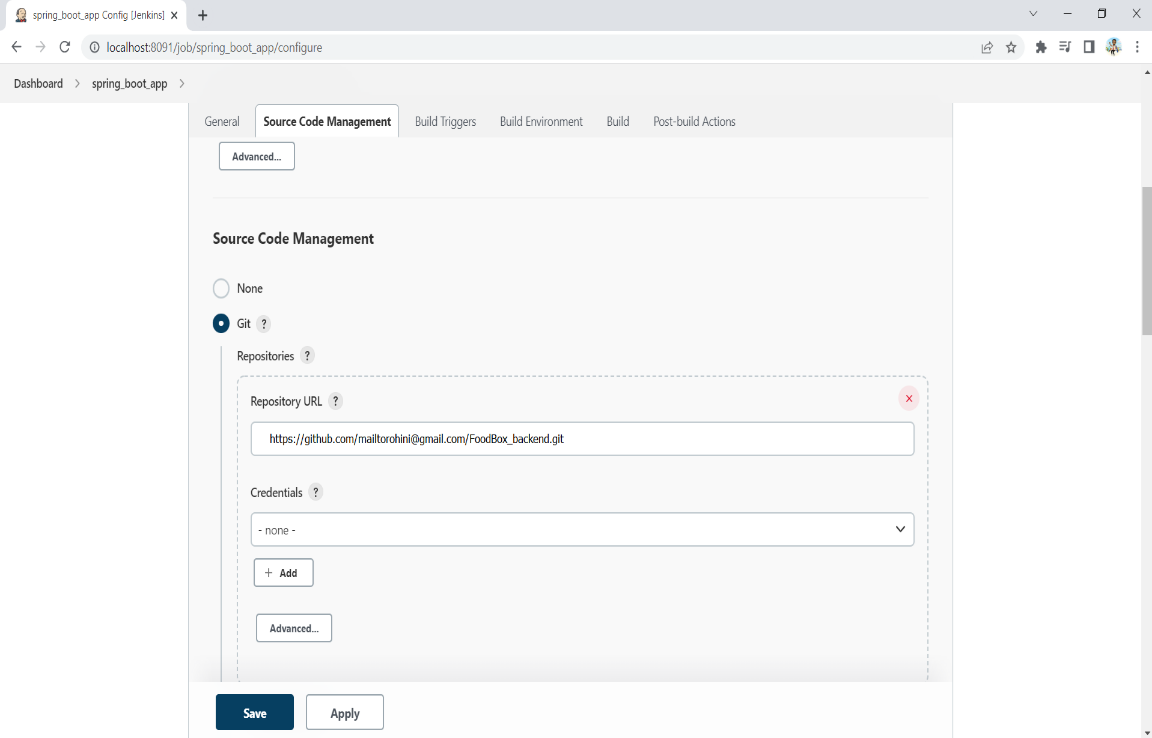
**Programming:**

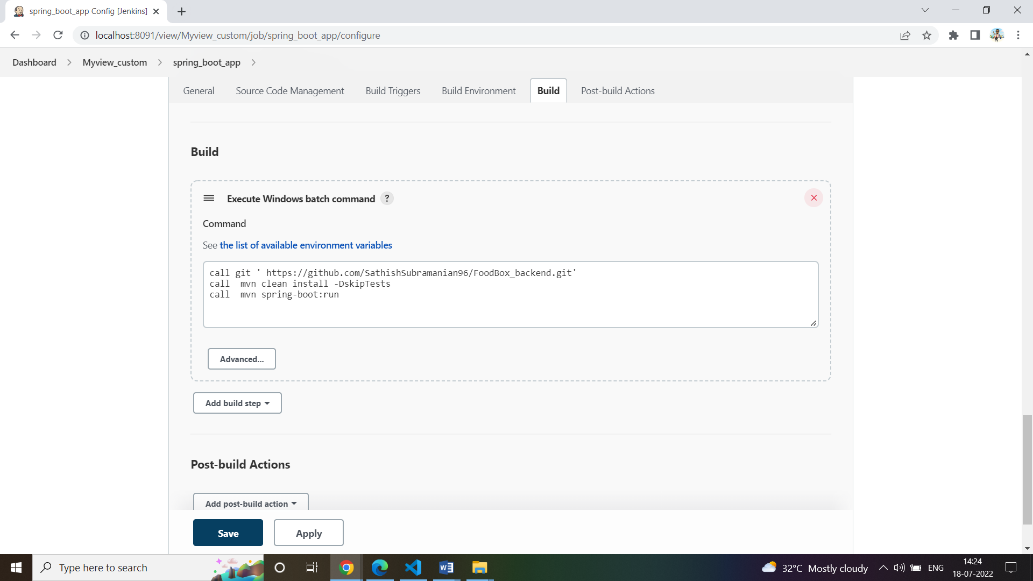
* Create database and tables.
* Add some rows and metadata to the tables
* Initialize a Spring Boot project for the Back-End side.
* Create REST APIs with spring Data JPA Repositories.
* Create desired DAO methods for the Back-End side
* Create a new Angular project for the Front-End side.
* Create login and register pages and components.
* Add cache to the login user
* Logout user and remove cache
* Show all products to the home page.
* Show all products as cards.
* Create a product details component.
* Search a product by a category.
* Search a product by a keyword.
* Add products pages
* Filter by page number

**Deployment:**

* Upload project to GitHub. For jenkins, two separate Github repositories are used for frontend and backend.
* Install Jenkins in the windows as a service.
* Configure all the tolls such as maven, java, git and nodejs globally available by setting the proper system and environment variables or by configuring in jenkins.
* For github integration, two separate repositories have been created since we need to create two jenkins freestyle project.

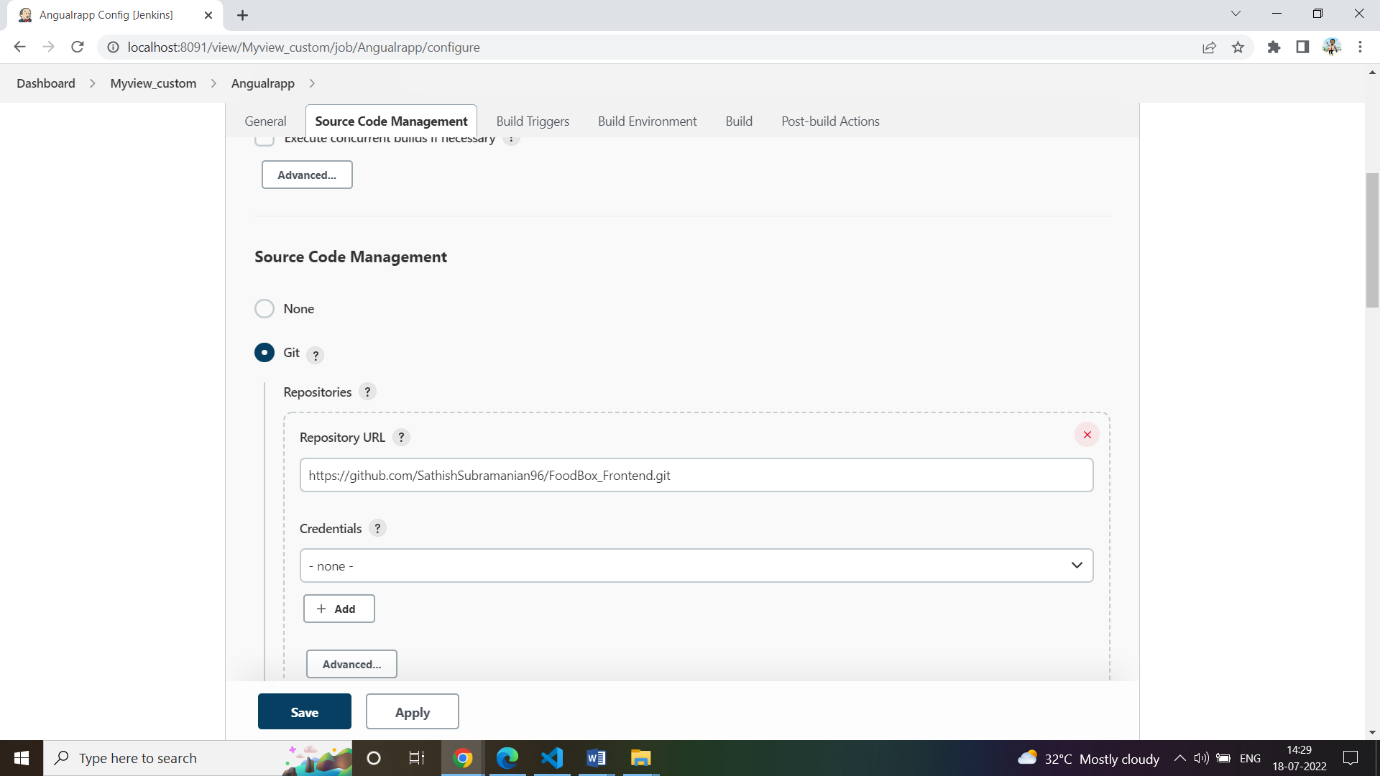
* Starting with jenkins, we need to create two freestyle projects with the following windows batch command for backend,

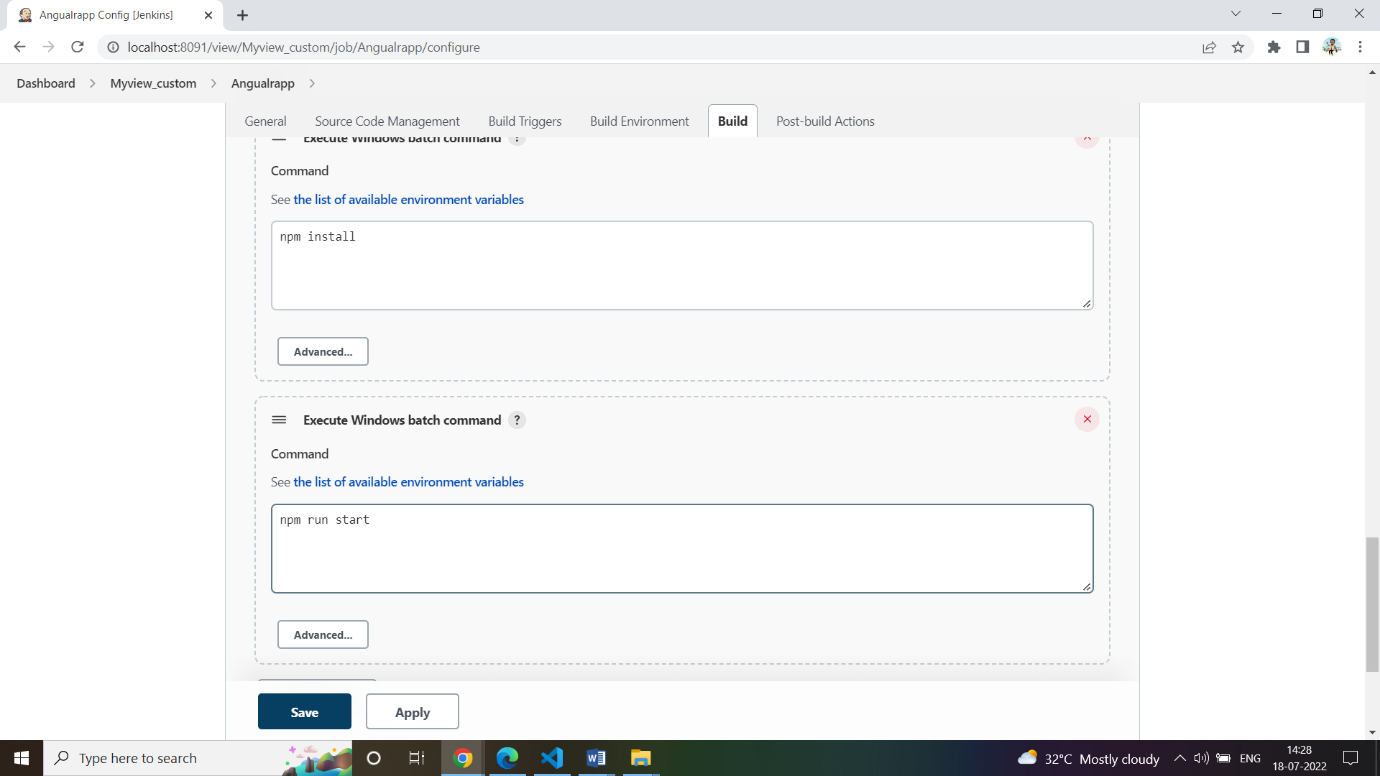




* Next we need to create another freestyle projects with the following windows batch command for frontend,

1. **Npm install**
2. **Npm run start**

****

****

* Now we can run both the jobs simultaneously and we can check the application though the following url- <http://localhost:4200/>

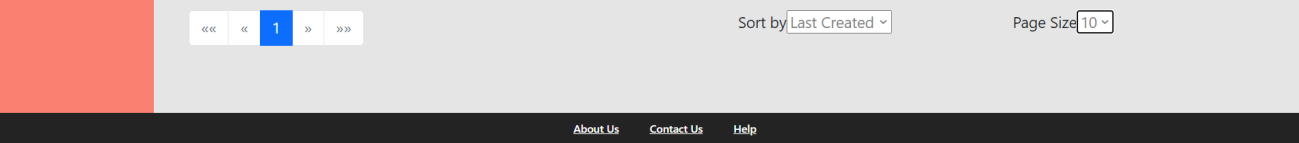
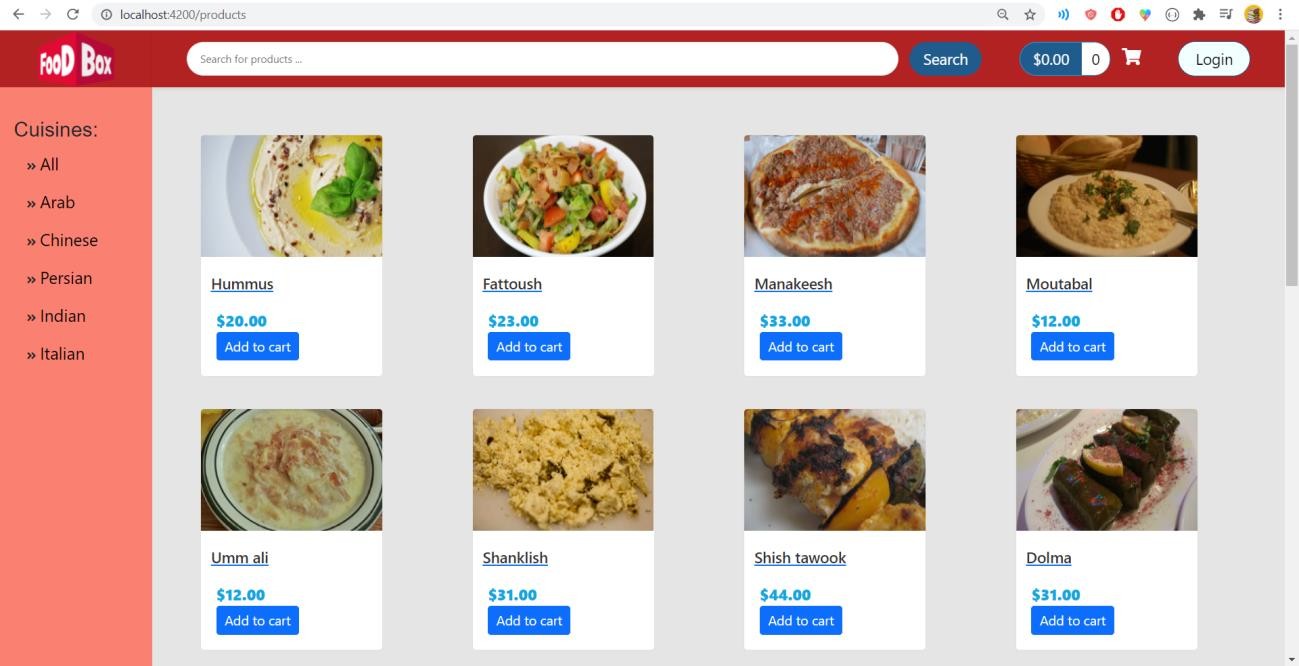
## **Technologies**

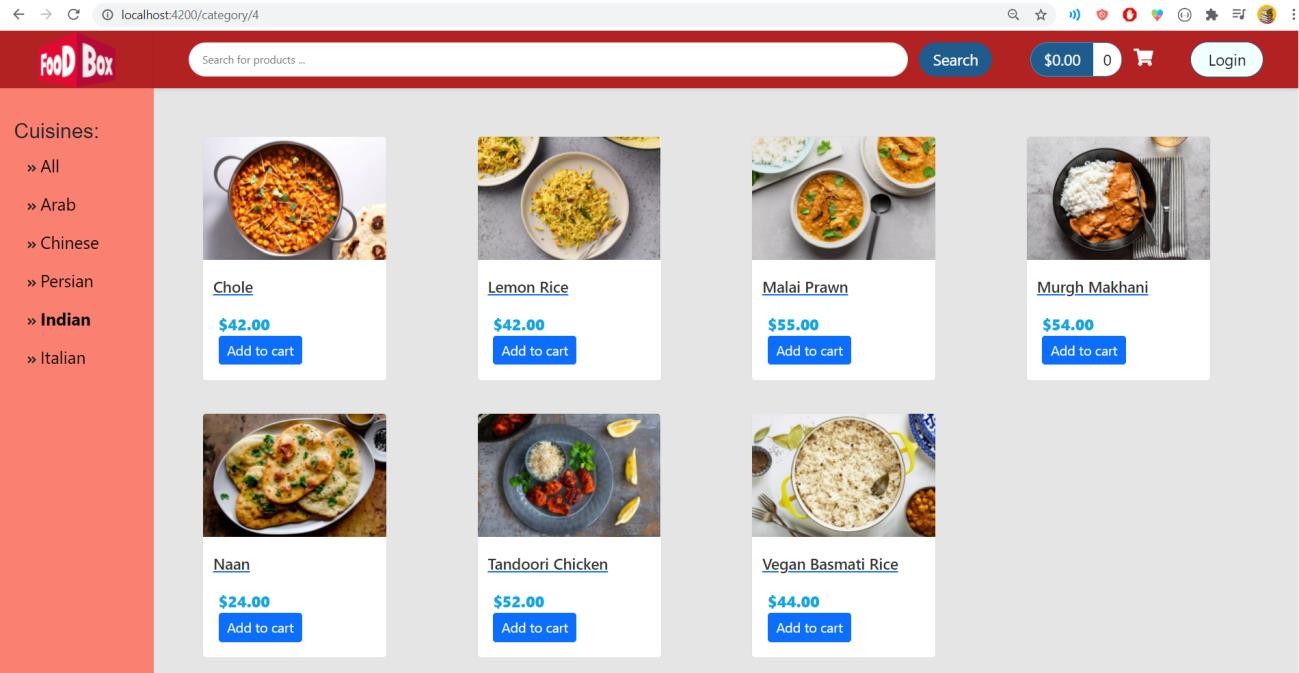
* **Spring Boot:** used in the back-end side to create the REST API and retrieve data from a database.
* **AWS RDS:** to upload the database online.
* **Jenkins:** to build the project from GitHub.
* **GitHub**: to upload the source code of the project.
* **Selenium:** for automation and testing.
* **HTML/CSS**: to create and format the content of the pages.
* **Bootstrap**: to use some CSS and JavaScript designs.
* **Maven**: to manage the project.
* **Visual Studio Code**: to write and run the Angular code.
* **Eclipse**: to write and run the Spring Boot code.
* **MySQL**: to use it as database management system.

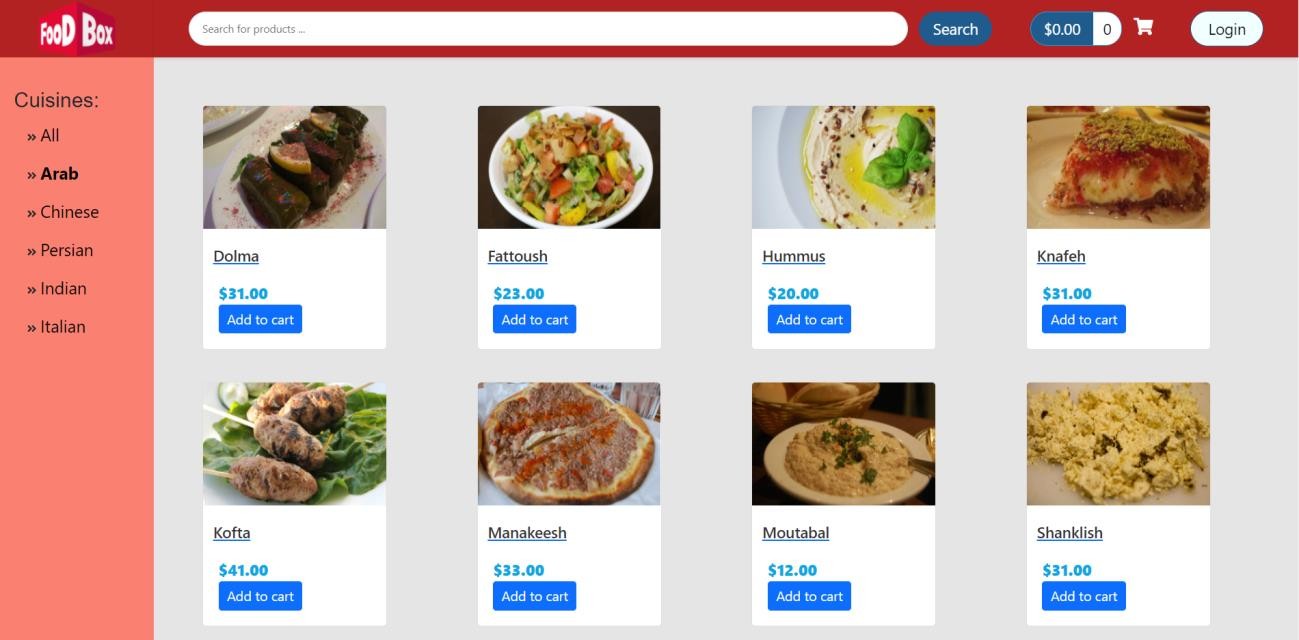
**How to run the program locally**

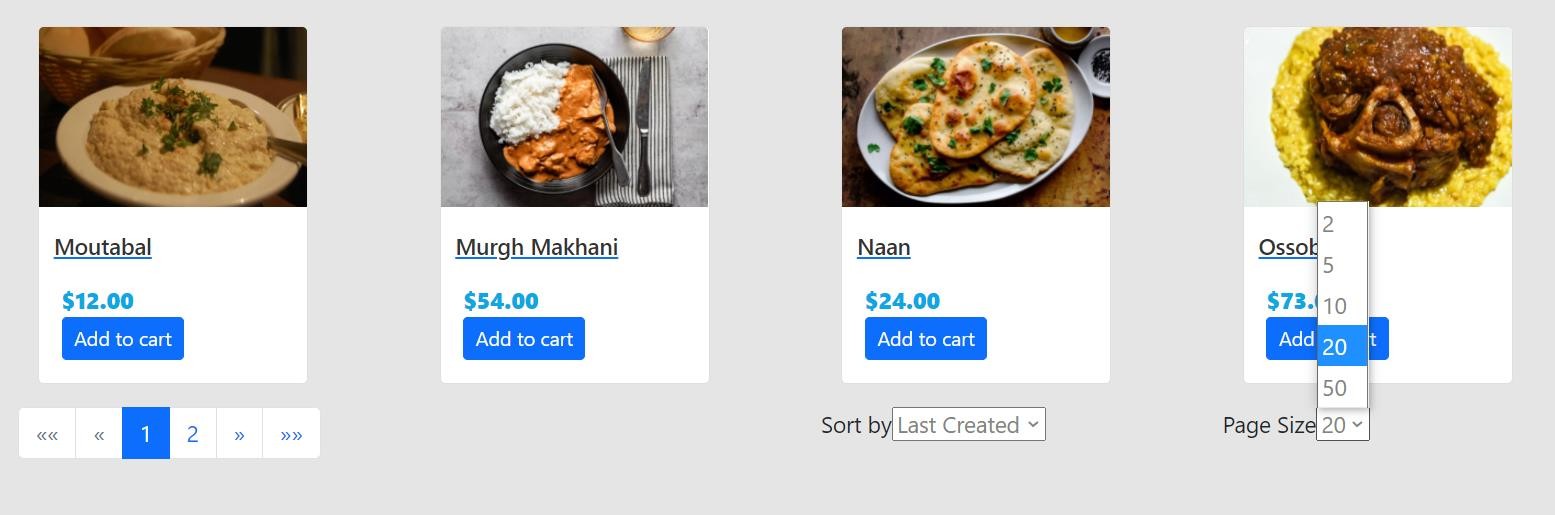
* Clone project
  + clone git : git clone
* Import the “\Back-End\foodbox\database\foodbox.sql” file to your database administration tool.
* Go to “\Back-End\foodbox\src\main\resources\application.properties” file, open it.
* Edit some values of the database’ properties to be suit to your database administration tool(for localhost).
* Run the back-end project as a maven project:
  + cd to your project “Back-end\foodbox”
  + mvn compile -Dskiptest
  + mvn exec:java -Dexec.mainClass=com.simplilearn.foodbox (or)
  + mvn spring-boot:run
* Open another command line for the front-end part.
* cd to your project “Front-end-end\foodbox”
* Install the following:
  + npm install --save-dev
  + npm install @angular/localize --save
  + npm install bootstrap --save
  + npm install font-awesome –save
* Run using ng serve –open
* It would be displayed in <http://localhost:4200/>

**Using the application:**

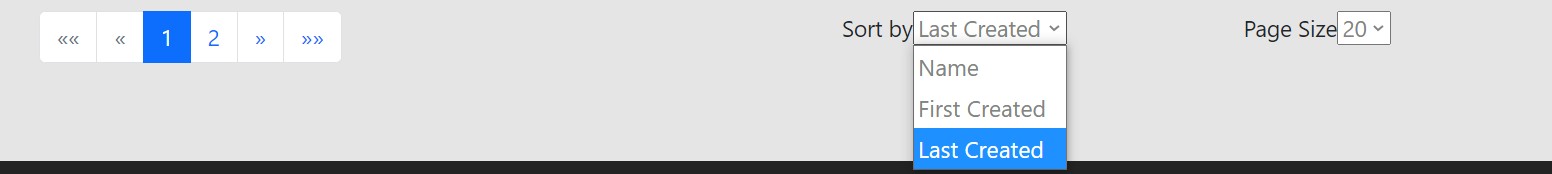
• General look of the store:

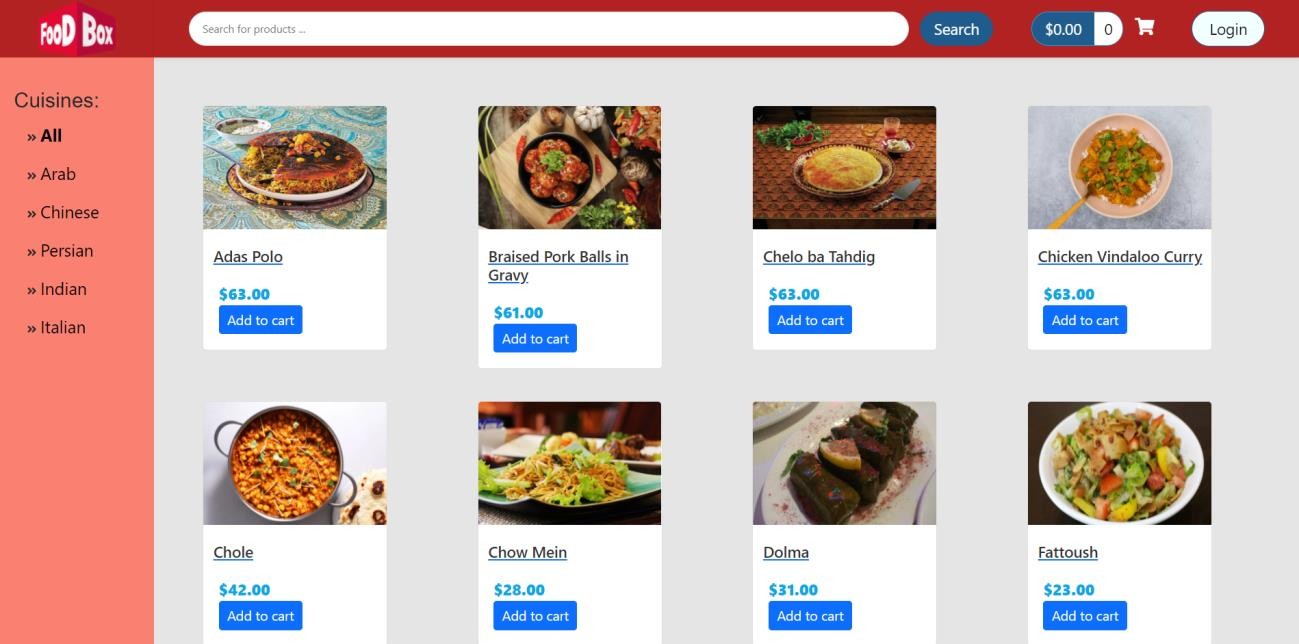
**•** Filter the product according to the cuisine



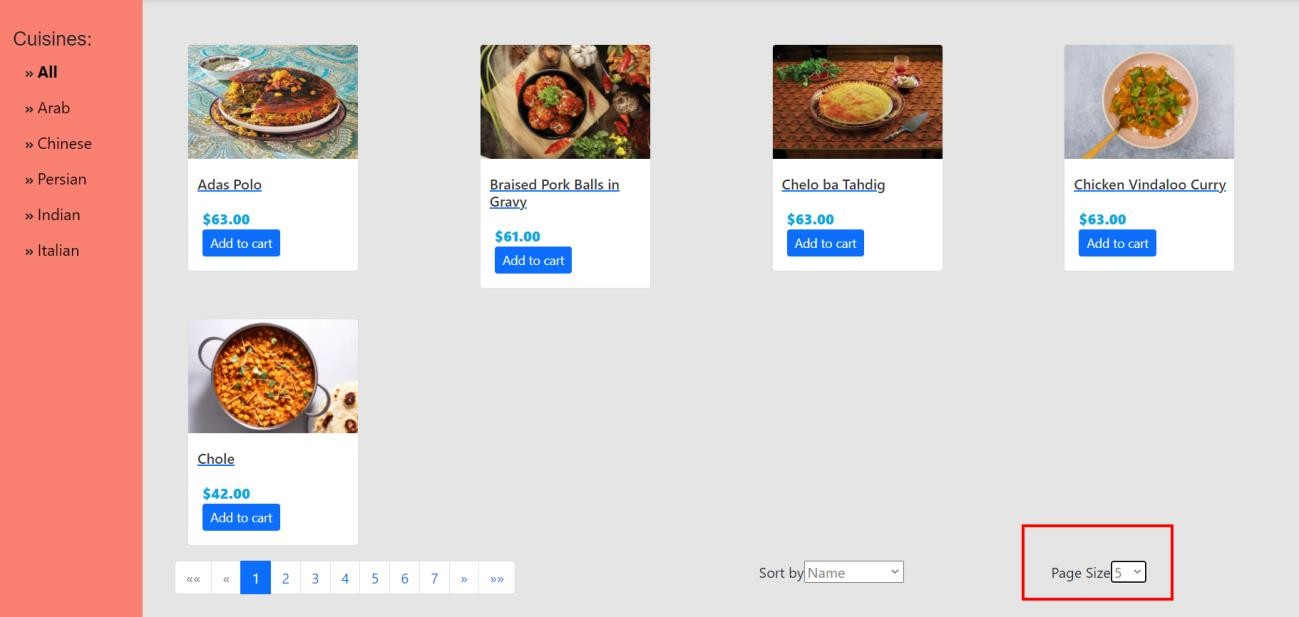
• Filtering by page and sorting are available

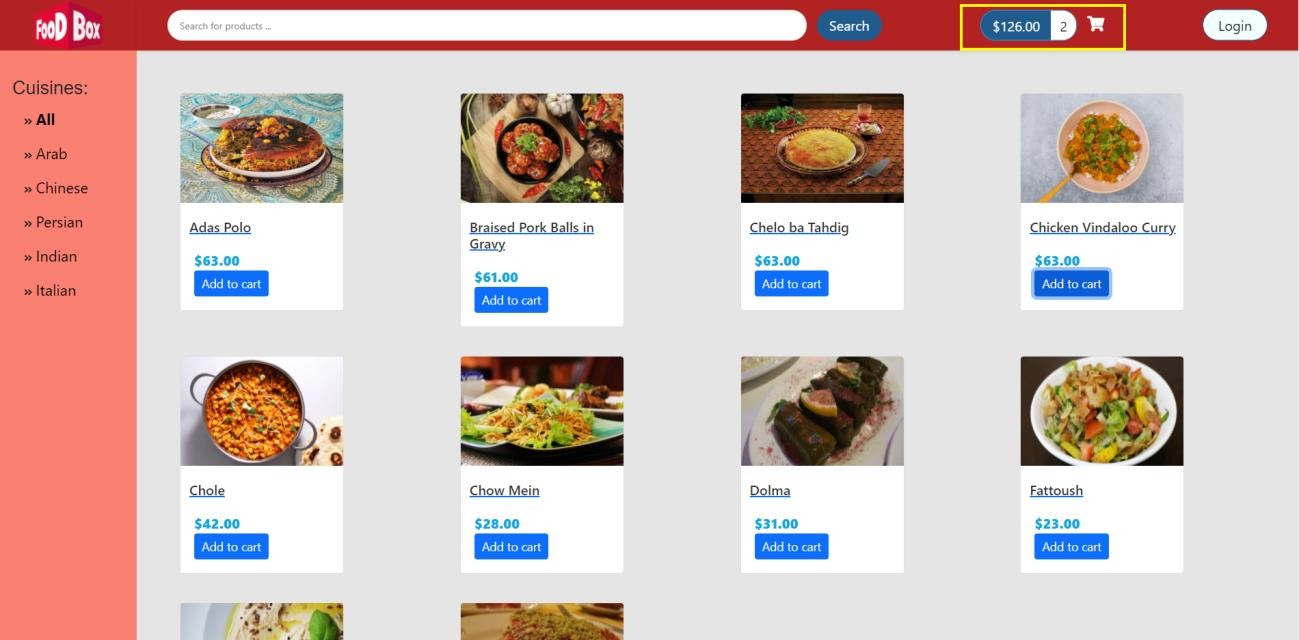
• Sorting can be in one of these three options



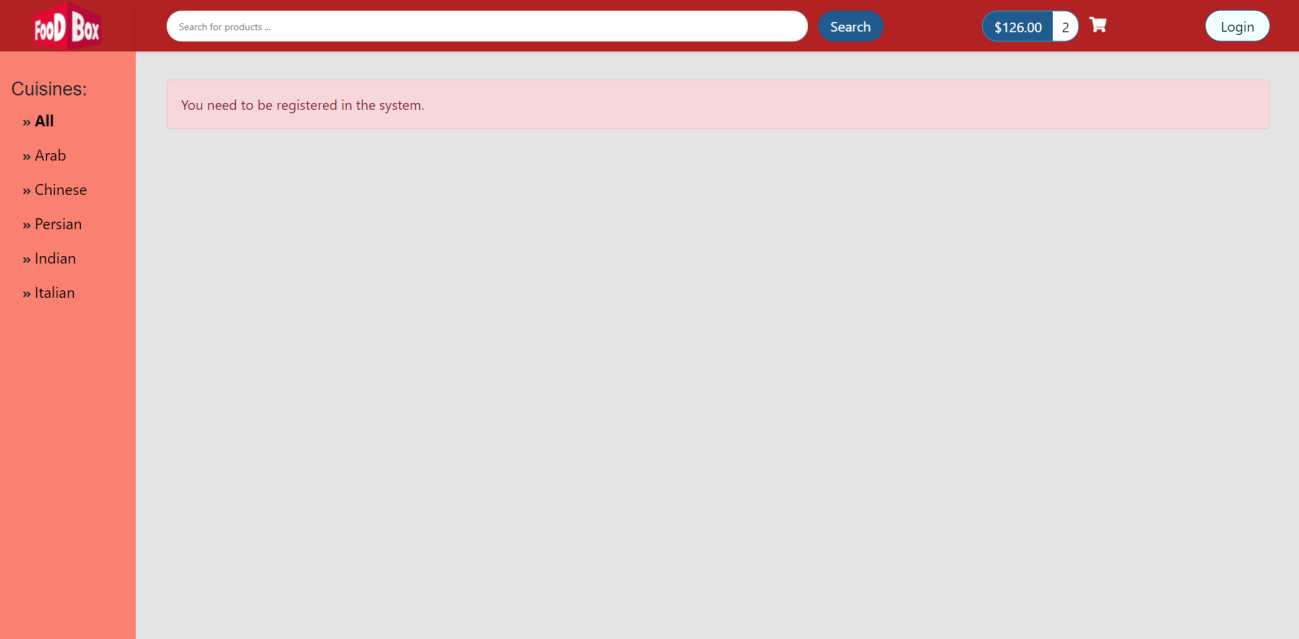
• Sorting by name

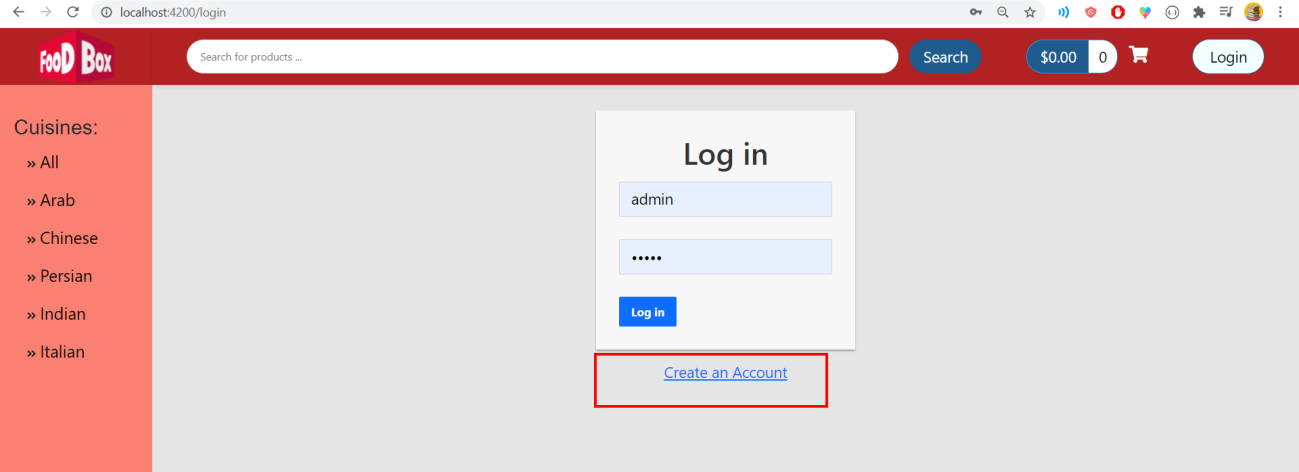
• Choosing the number of each displayed products in each page



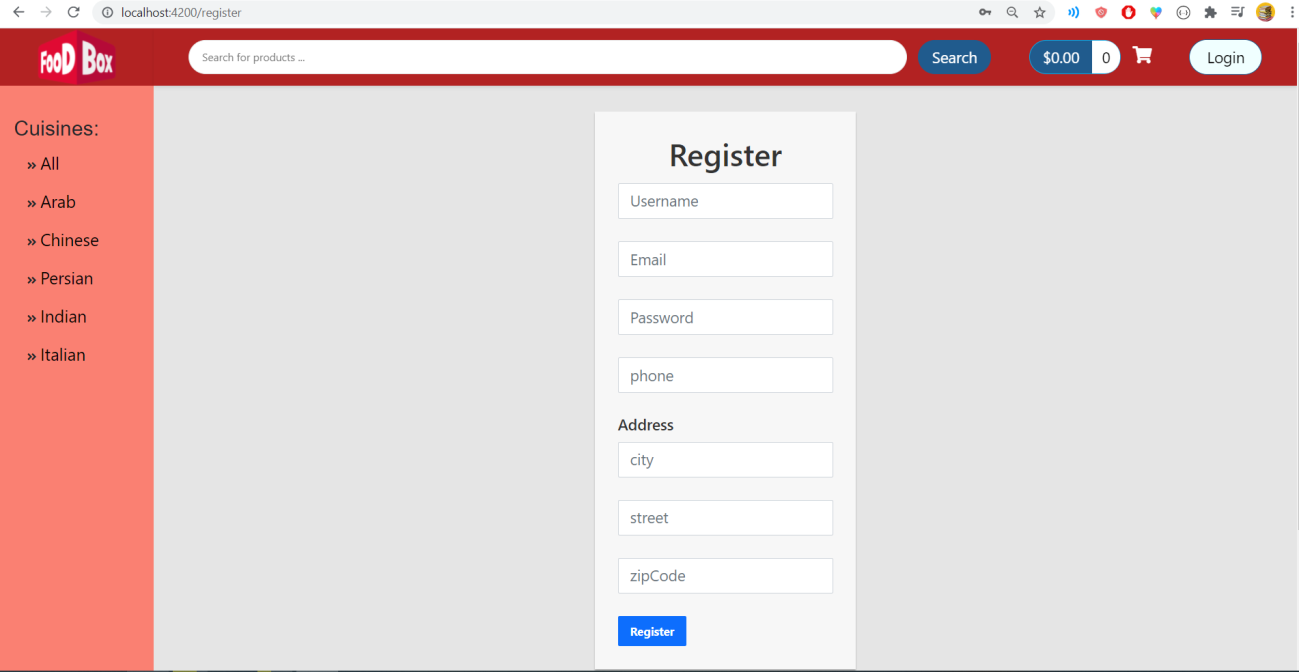
• You can add products to the cart

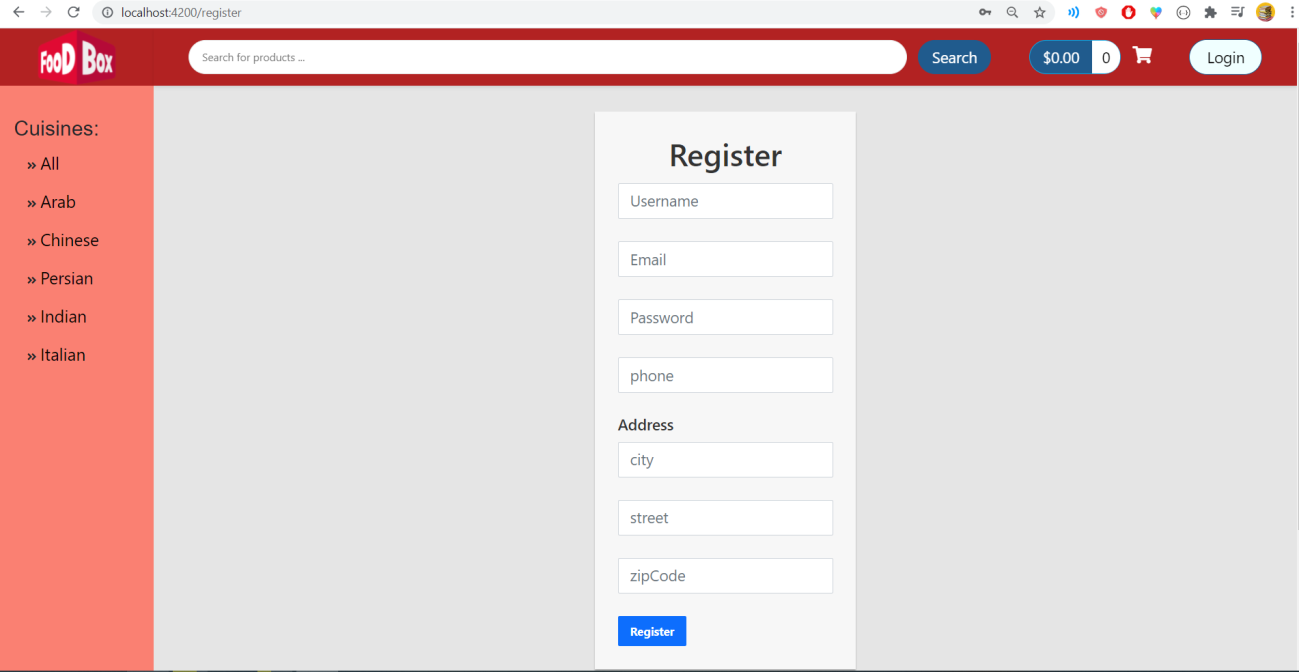
• However, you need to be registered to buy

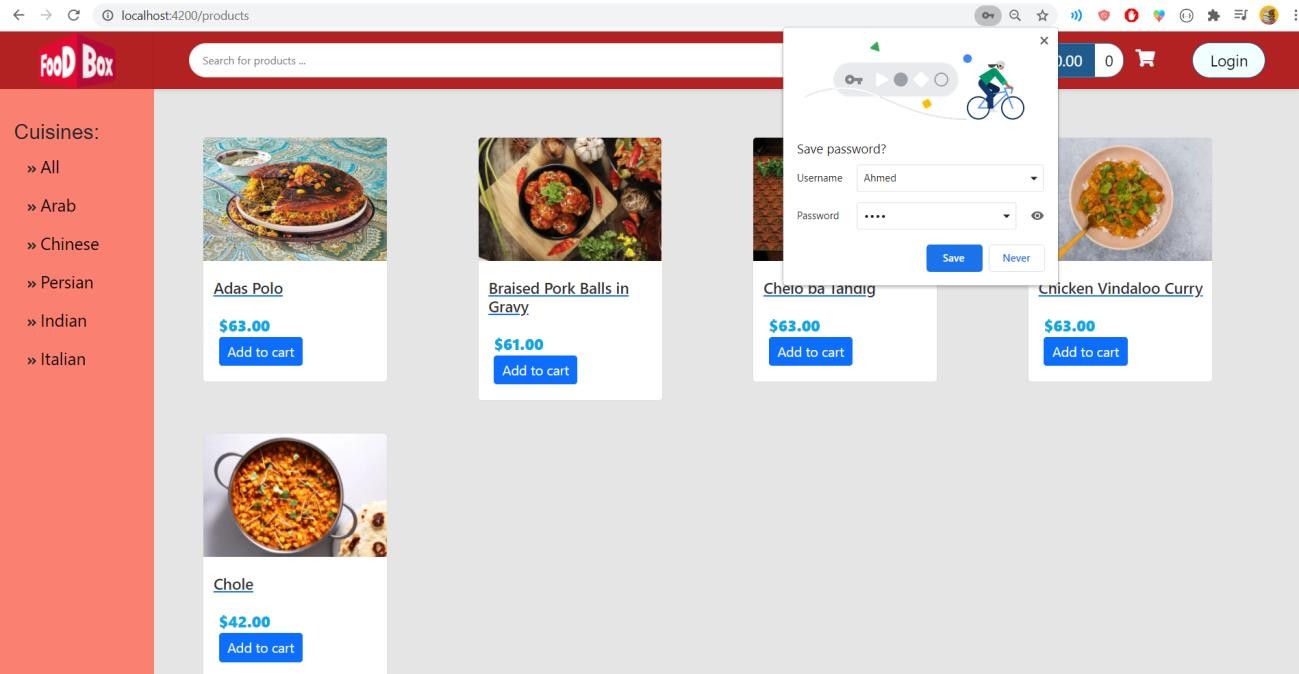


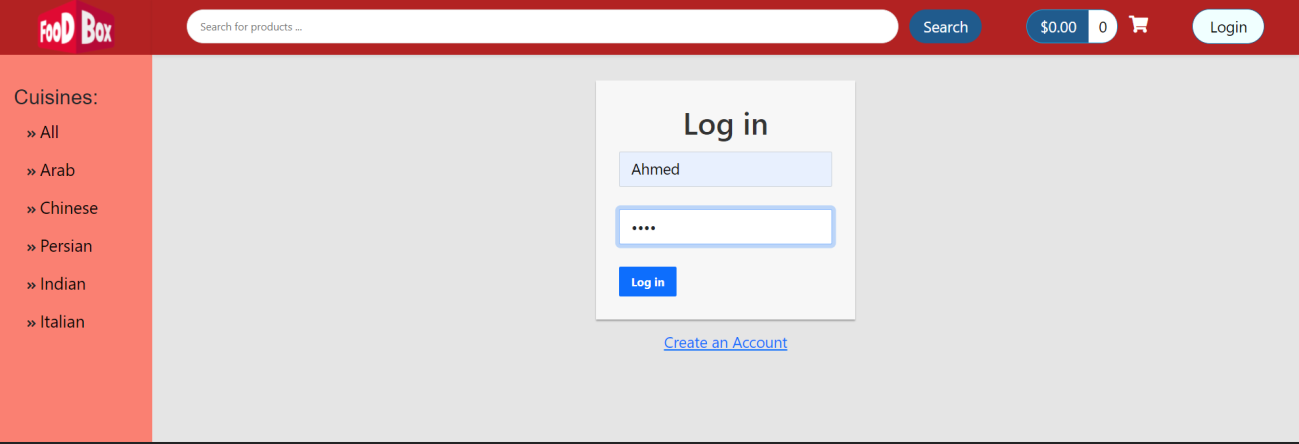
• Login and then create account

• Register form



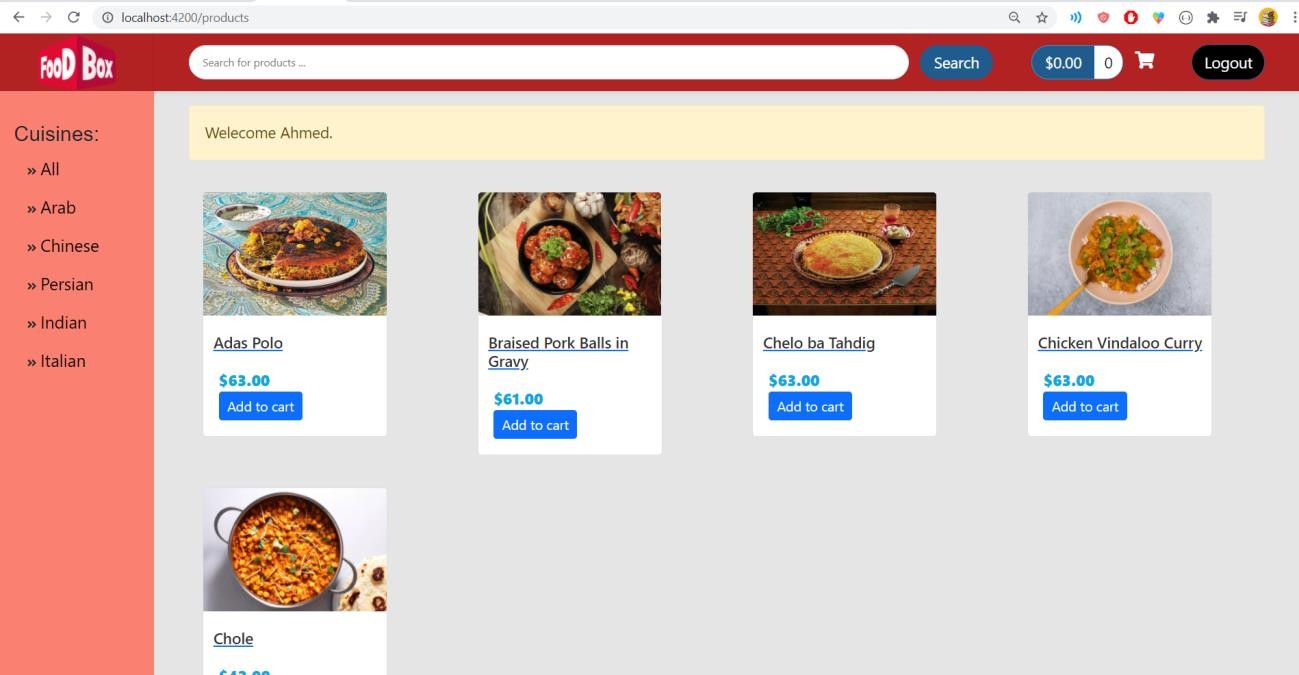
*  The User must Register to order food from the Food Box App



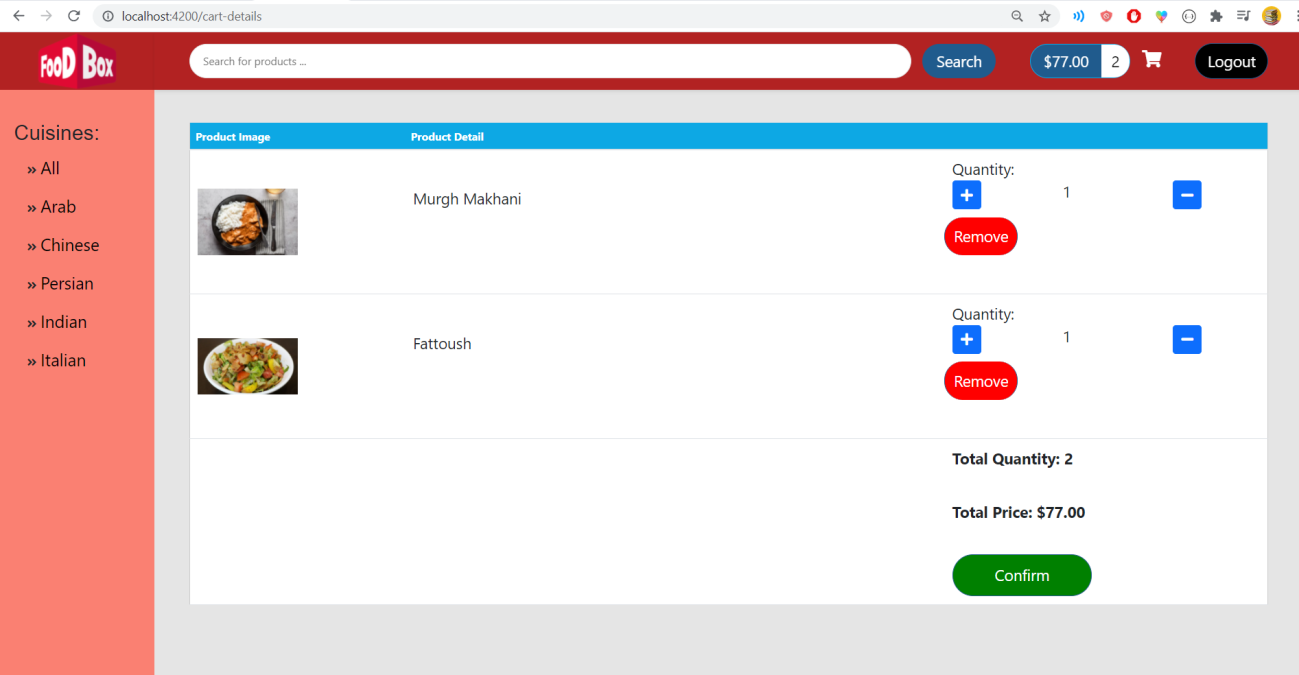


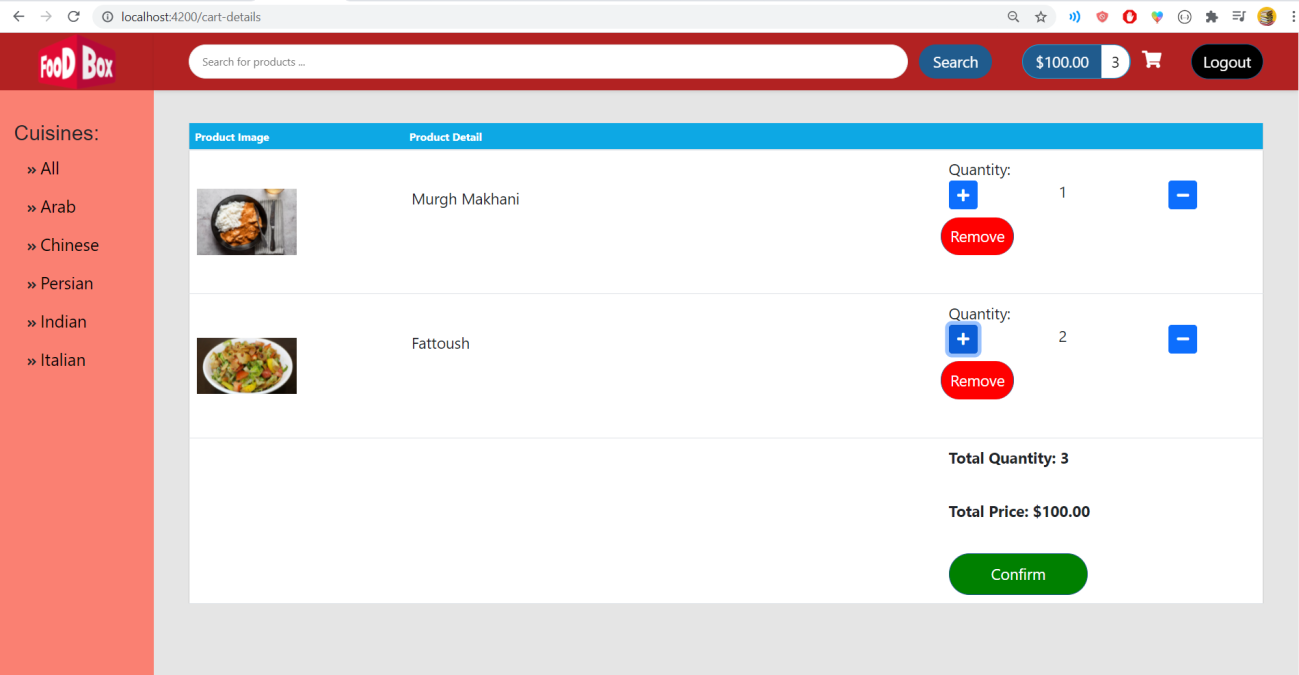
* If the User is already registered, then they can login directly as user by entering login details

• User signed in to the application.

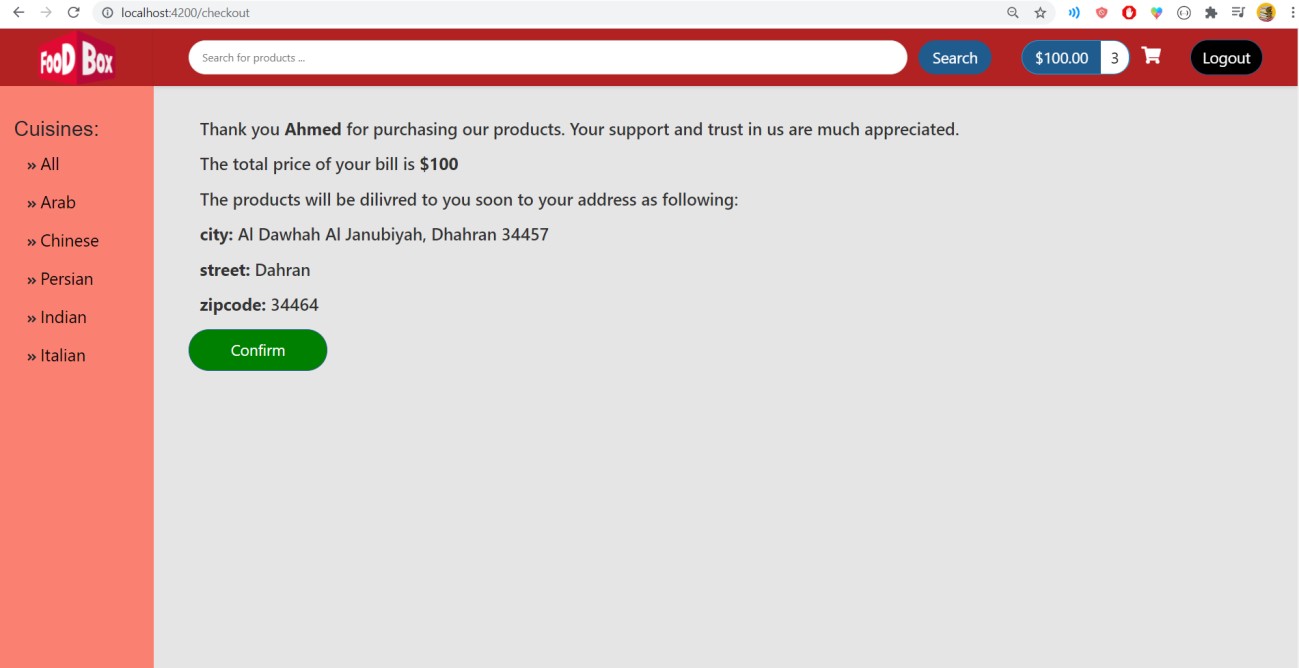


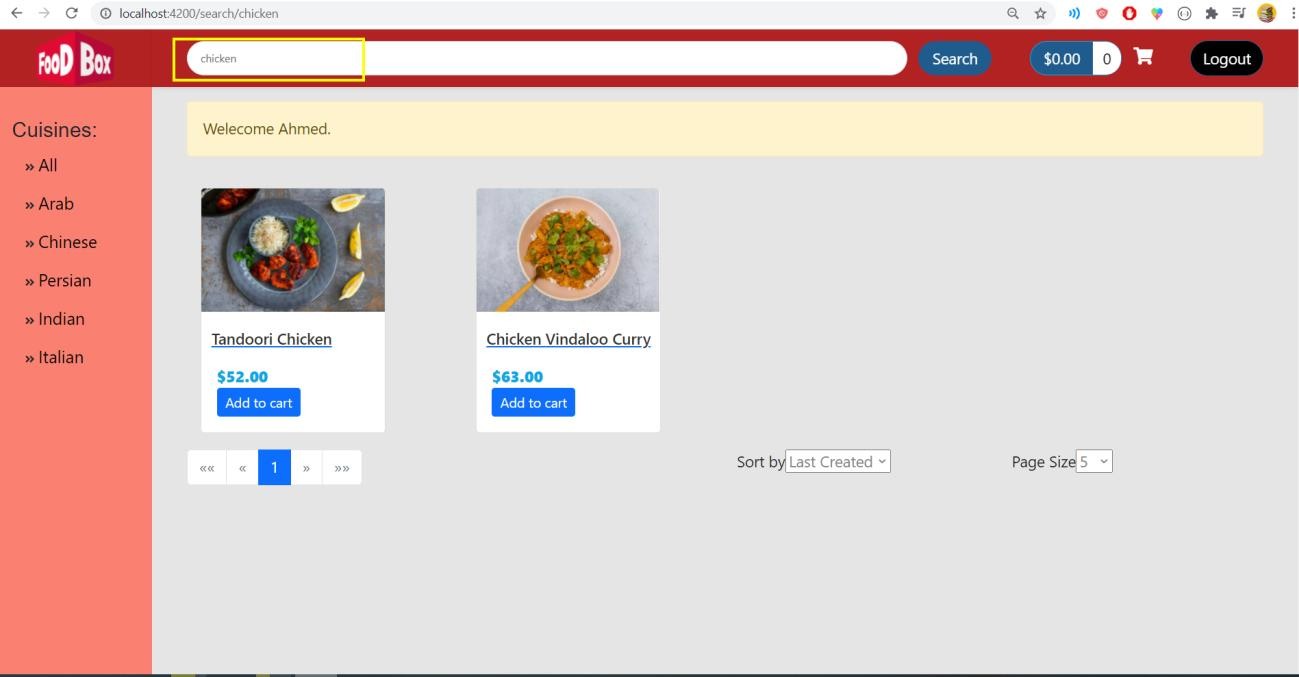
• User can now buy (Payment gateway)



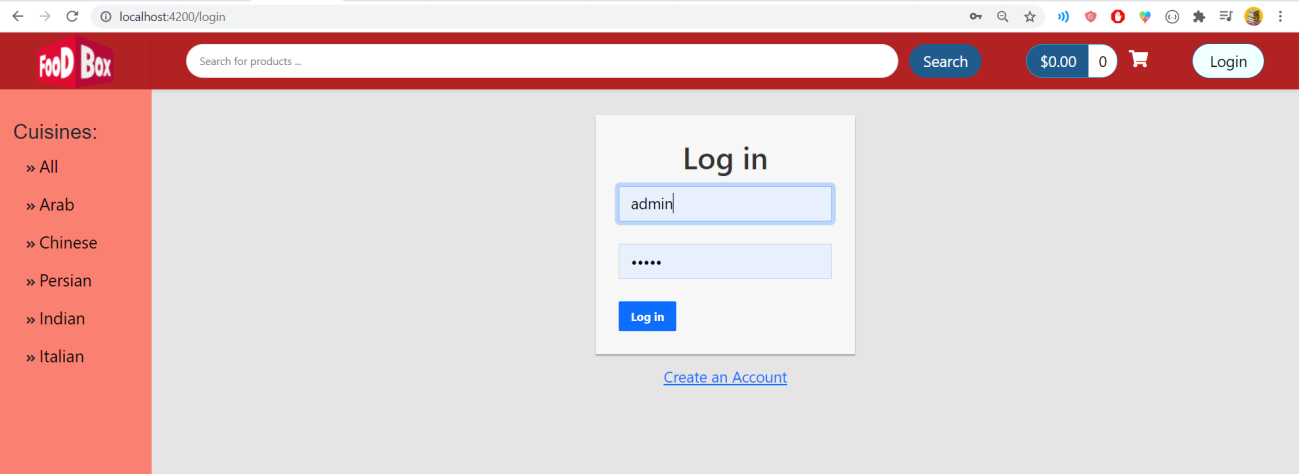
• Can add/remove products during review

• Check-out and confirm

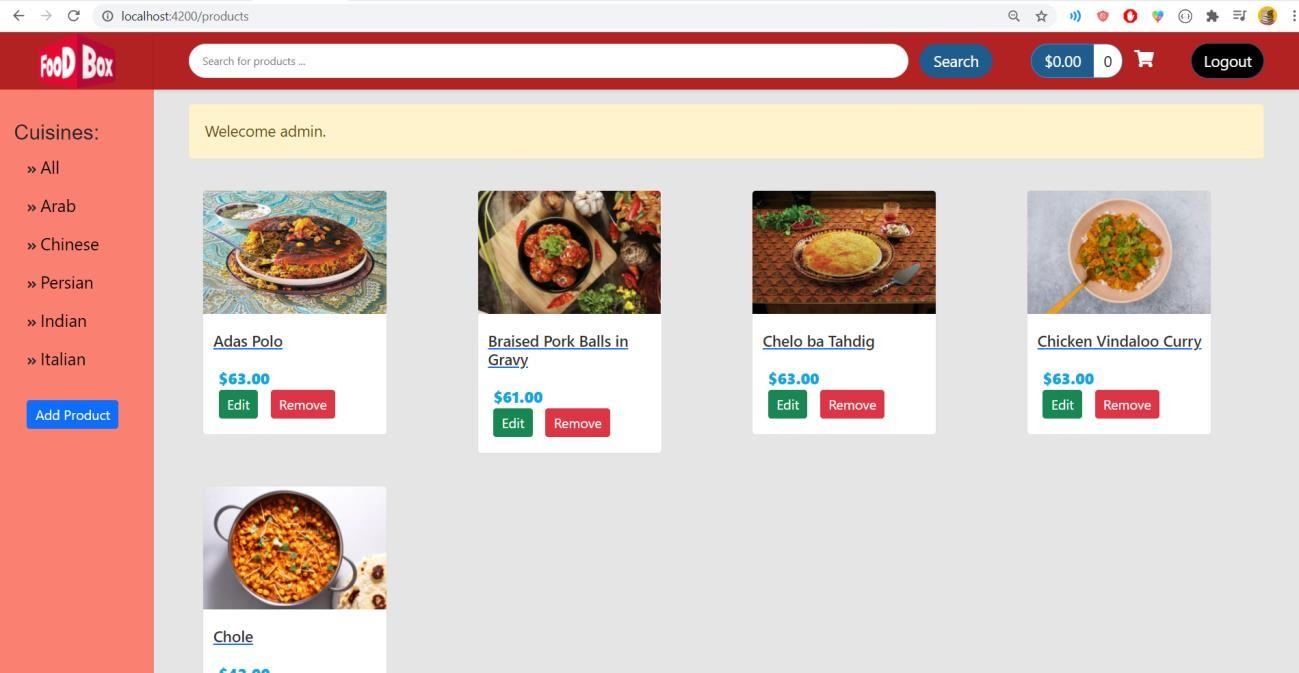


• Search products by Name

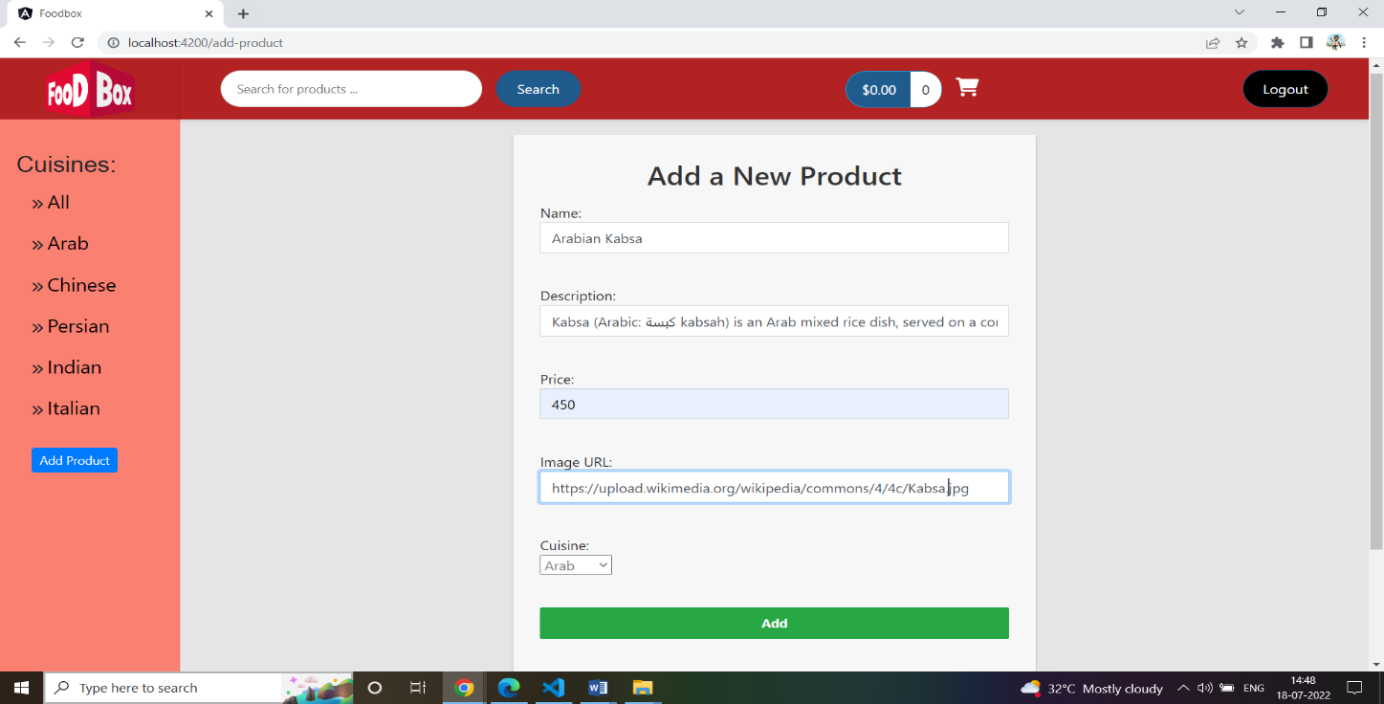
• Now, let’s log in as admin

• Admin/admin

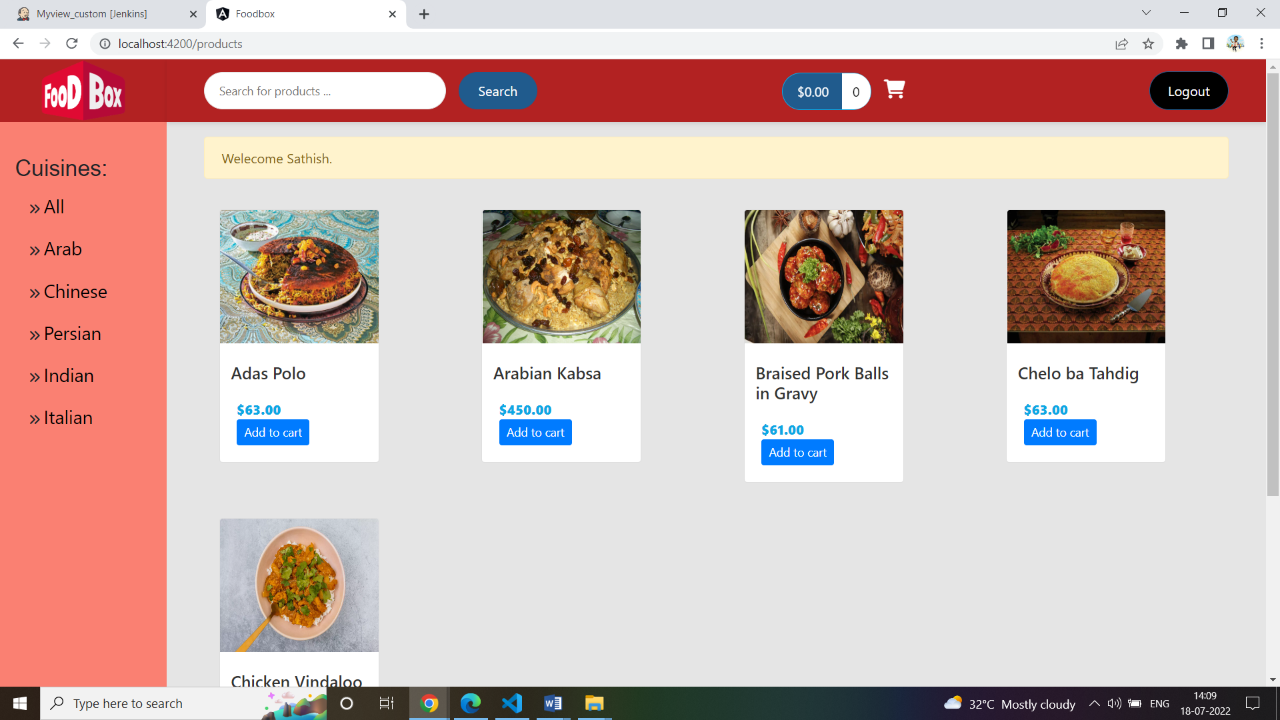
• Admin can:

* Create product
* Edit product
* Delete product

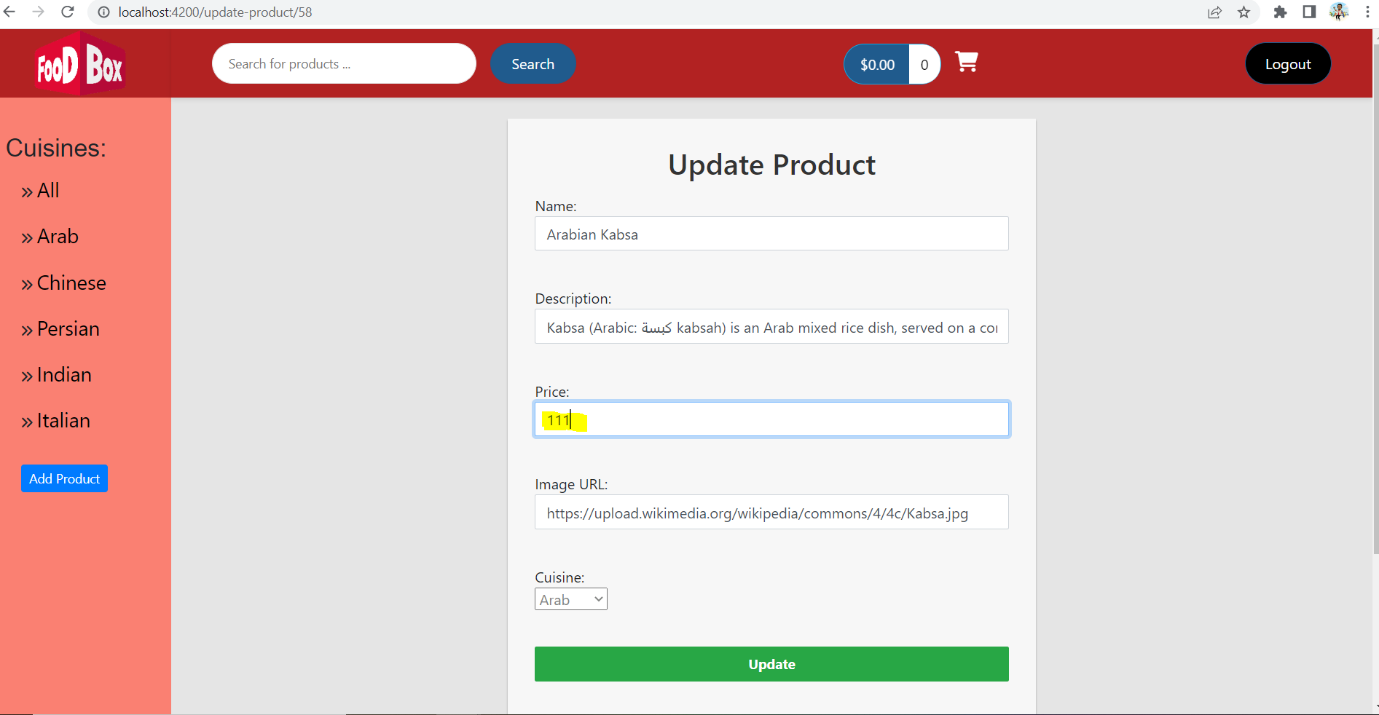
• Create a new product



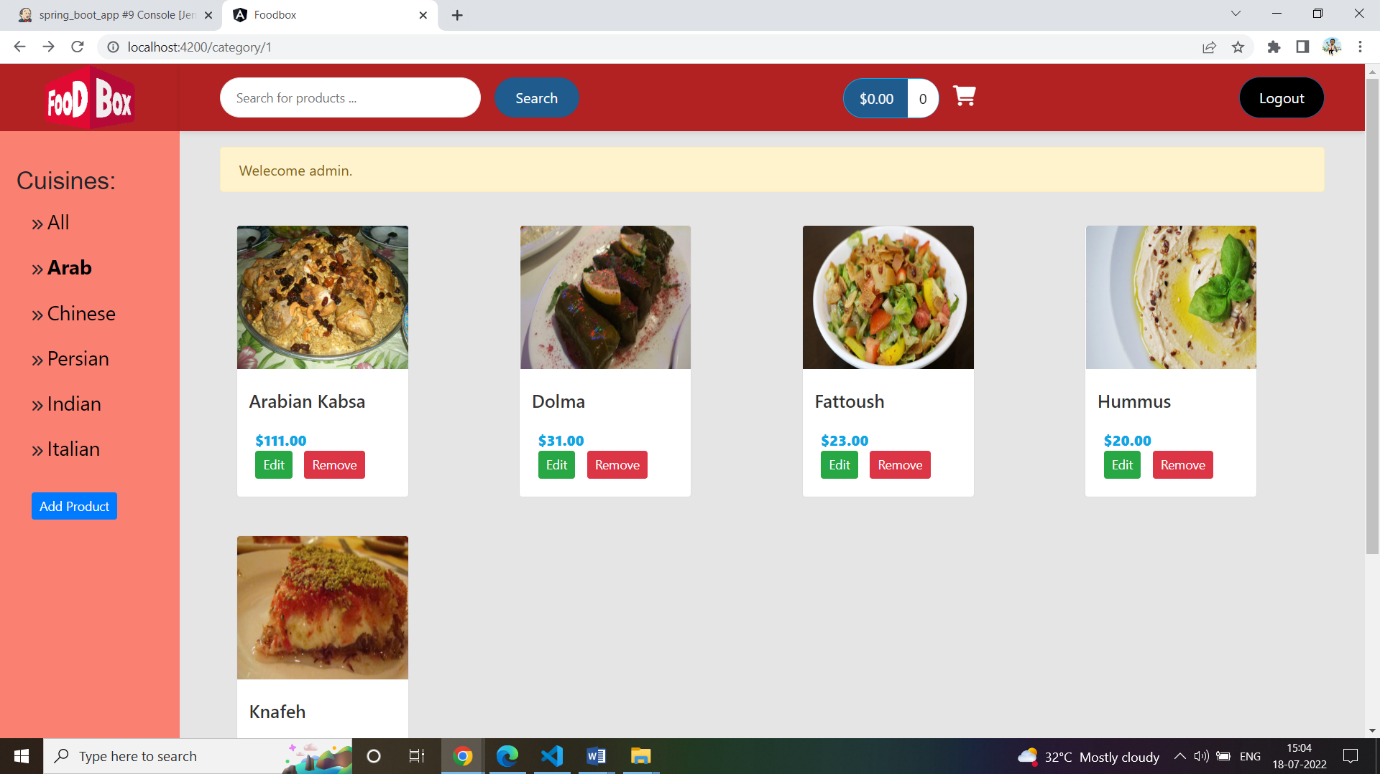
• The new product was added



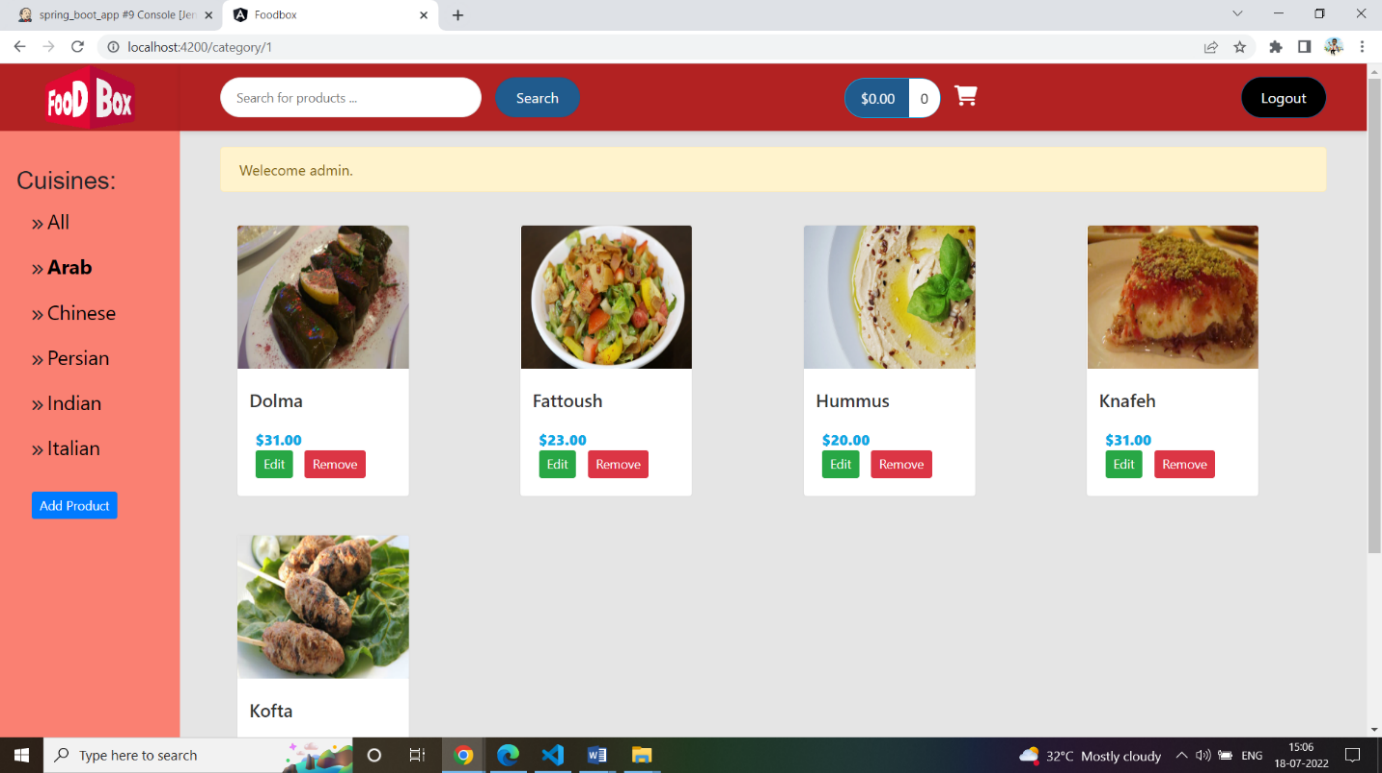
• Let’s update it



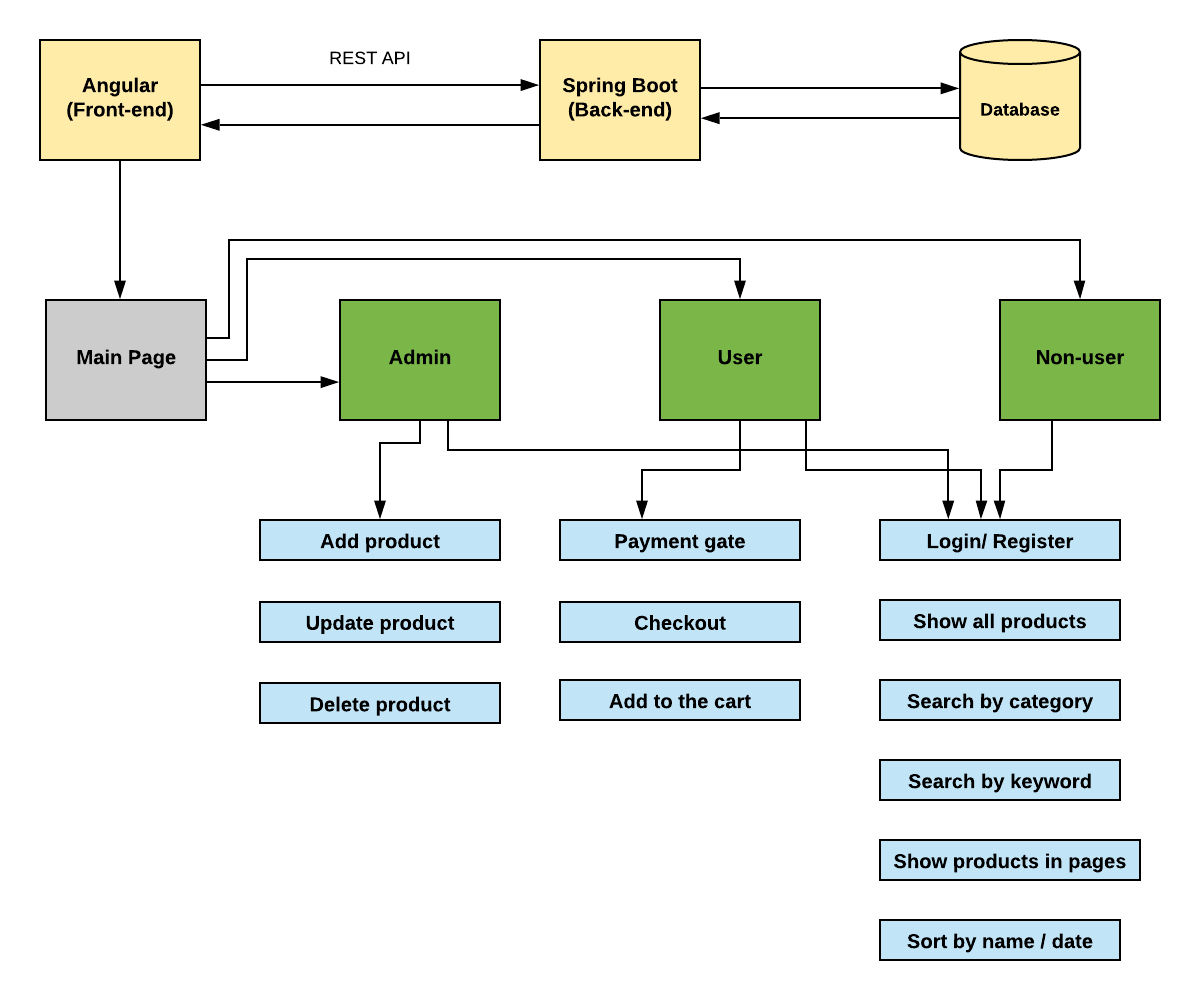
* Updated Product



• Click Remove to remove it



Food Box Project Flowchart:



Conclusion:

A dynamic and responsive online food delivery web application for ordering food items of different cuisines from a restaurant is created.

Pushing the code to your GitHub repositories:

● Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

● Initialize your 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 “Changes have been committed.”

● Push the files to the folder you initially created using the following command:

git push -u origin master