**Foodbox**

Project and Developer Details

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 It was created as the Capstone Project for the Full Stack Java Developer course.

Sprints and Tasks

Product Backlog:

**Programming**:

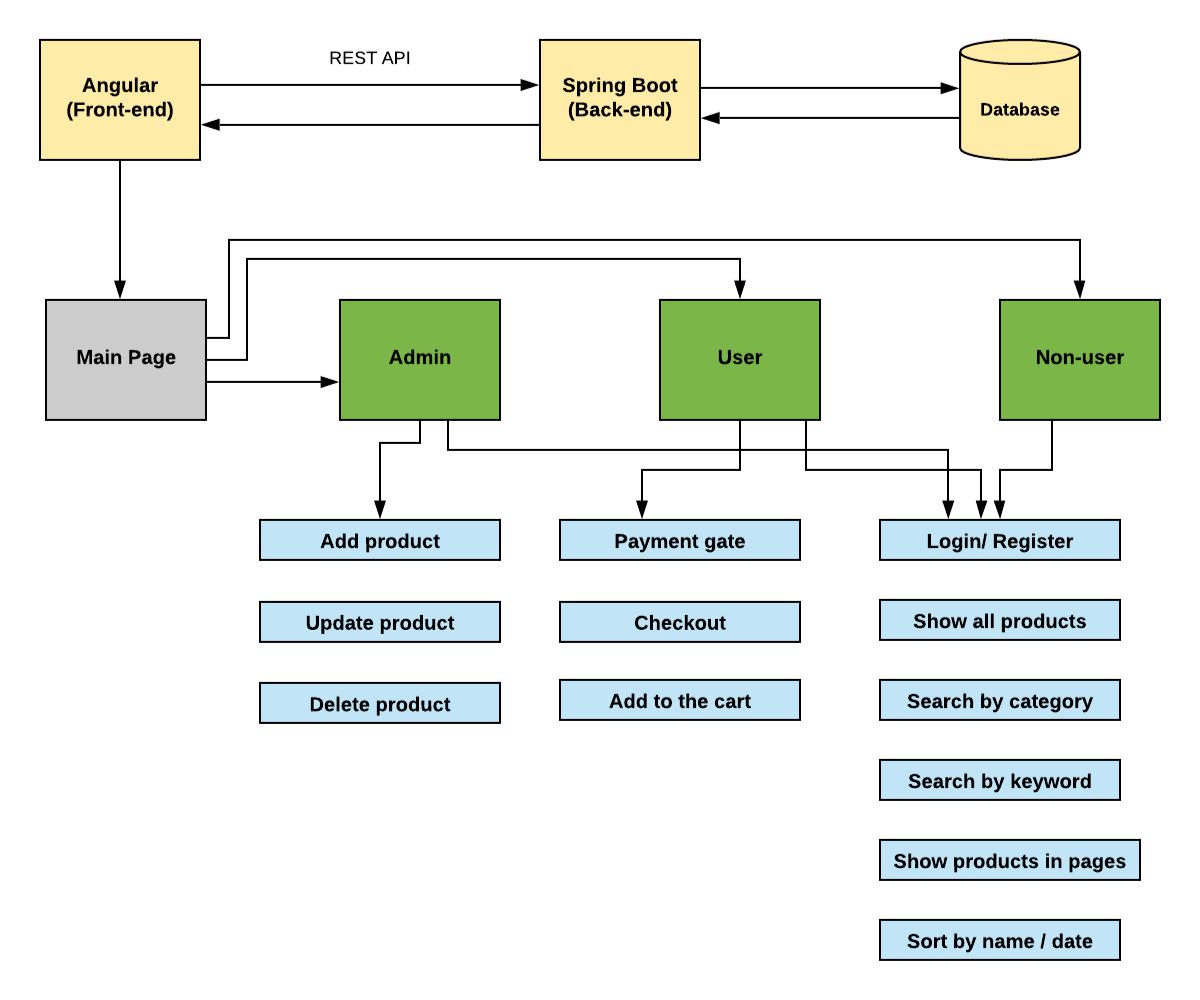
1. Create database and tables.
2. Add some rows and metadata to the tables
3. Initialize a Spring Boot project for the Back-End side.
4. Create REST APIs with spring Data JPA Repositories.
5. Create desired DAO methods for the Back-End side
6. Create a new Angular project for the Front-End side.
7. Create login and register pages and components.
8. Add cache to the login user
9. Logout user and remove cache
10. Show all products to the home page.
11. Show all products as cards.
12. Create a product details component.
13. Search a product by a category.
14. Search a product by a keyword.
15. Add products pages

1. Filter by page number
2. Sort product by different options
3. Add products to the cart.
4. Update total price in the cart status.
5. Show the payment gate and review the list
6. Add and remove products from the review list
7. Update the total price in the payment gate
8. Create the admin view
9. Update/Remove a product for the admin
10. Add a new product for the admin
11. Update the CSS design
12. Add bootstrap and font awesome to the components.
13. Debug and test the project.

**Deployment**:

1. Upload project to GitHub.
2. Create a t3.medium instance for master.
3. Create a t3.micro instance for slave.
4. Create a Pipeline project on Jenkins.
5. Create a Jenkins file.
6. Generate a SSH key for GitHub.
7. Build the pipeline project.
8. Deploy the project.

Flowcharts of The Application



Core concepts used in the project.

* **Object-Oriented**: used to create and model objects for users and their credentials.
* **REST API**: used to communicate between the back-end and the front-end sides.
* **Data Access Object**: to abstract and encapsulate all access to the data source.
* **Object–Relational Mapping**: to map the objects to the database.
* **Databases:** used to store and retrieve data.
* **Data Sources**: used to define a set of properties required to identify and access the database.
* **Collections**: used some collections such Arraylist to store collection of data.
* **Deployment:** to deploy the local project to the end-users.
* **Virtual Machine**: use virtual instances to help to build, deploy and manage websites.
* **Exception Handling**: used to catch problems that arises in the code especially in I/O blocks.
* **Single Web Page**: apply the concept of a website that only contains one HTML page.