NUTRITIONIST - A Case Study

Problem Statement

Build a system to search for a specific food to find it's nutrition details, show list of matching food, view the nutrition content for a selected food and bookmark favourite food for later reference.

FoodData Central API

- The application needs to search for food and find nutrition data for a selected food by registering with the following link and get API key required to call the APIs.

- https://fdc.nal.usda.gov/api-key-signup.html
- https://fdc.nal.usda.gov/api-guide.html

Modules

User Module

On launching the application the user should get the login page. The login page should have a link for registration using which the user should be able to register.
Upon login, the user should be taken to the home page. Home page should have proper ui elements to search for food based on fooddescription, food type, published date.
Home page should have proper navigation links to logout, fetch favourite food, fetch recommended food etc.
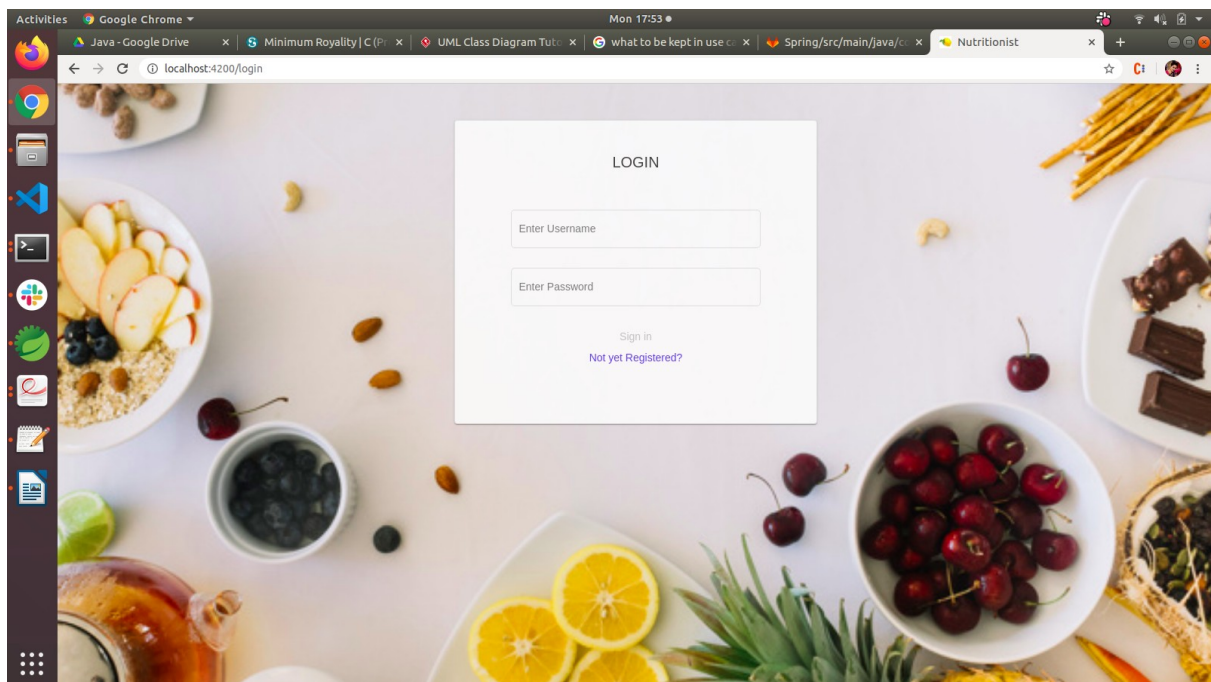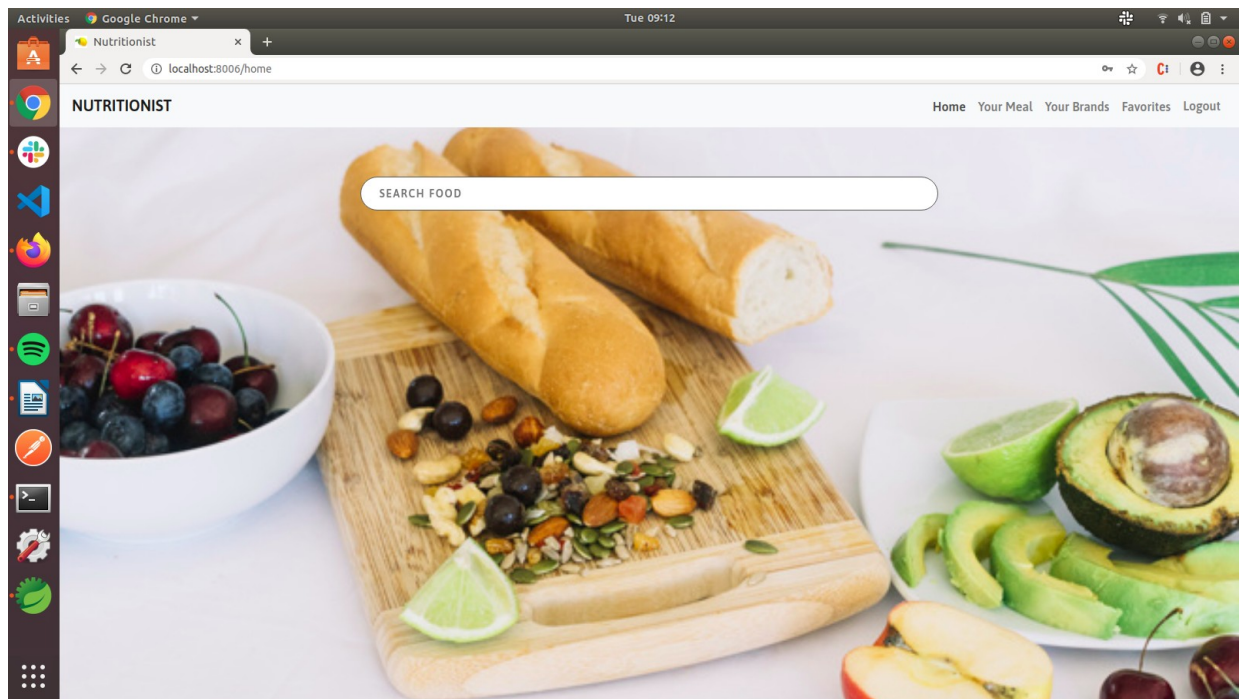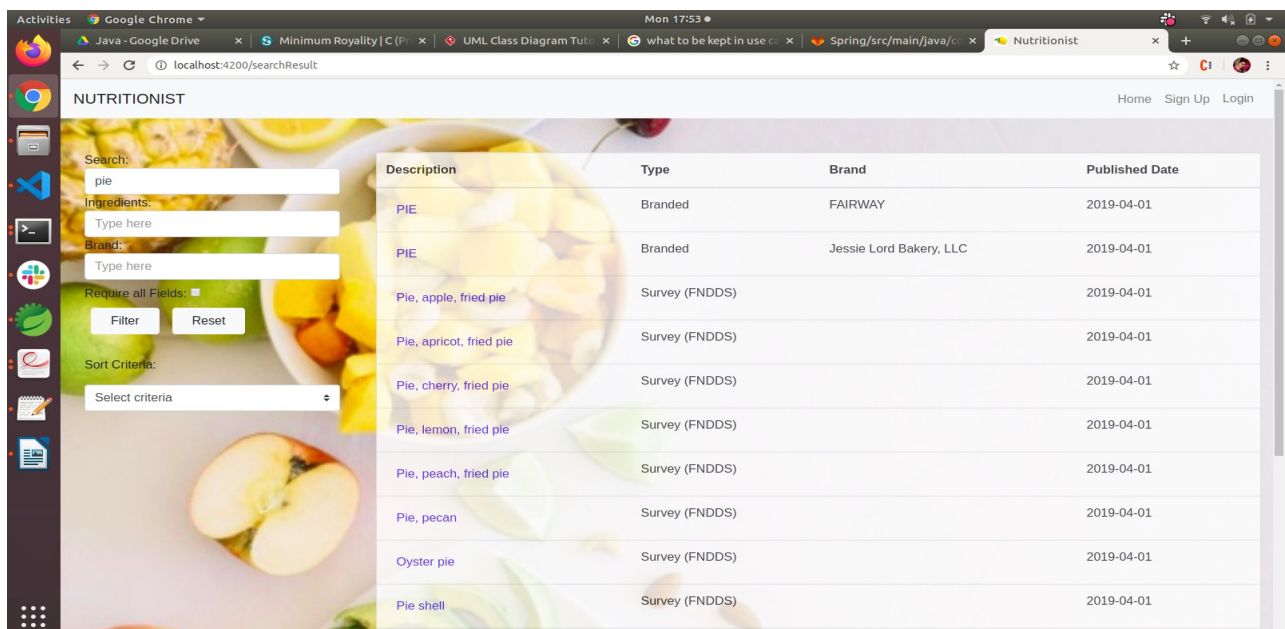
Search Food Module

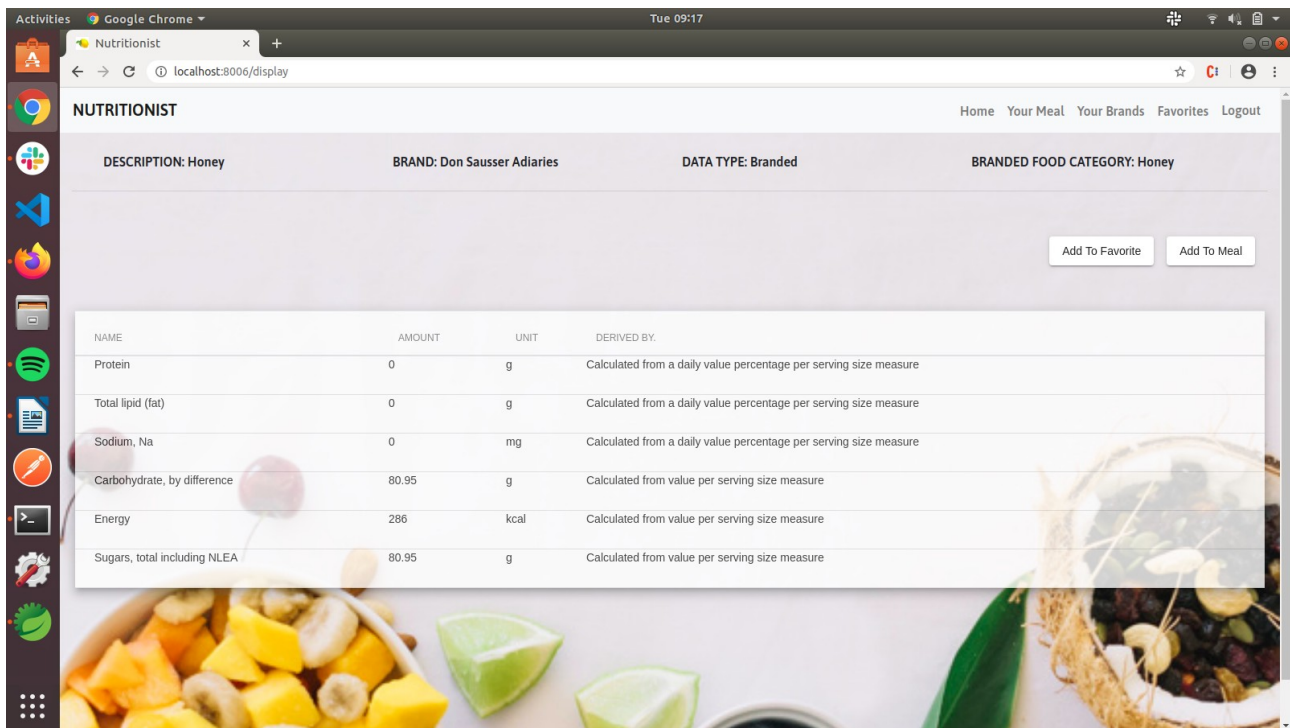User should able to search with different crieteria (based on availability of rest end point) like
a)ingredients
b)brandOwner
c)requireAllWords

Pagination should implemented sorting shoulld be implemented.

Sorting the results should also be possible. Based on selected field – like description, food type, published date.

The users should be able to select a specific item, which should display the nutrients of the particular item (Data fetched form the API).

Favorite Food Module

User should able to add food as favorite.
User should able to delete food as favorite
User should able to fetch all favorite food items display into categories (brand owner)

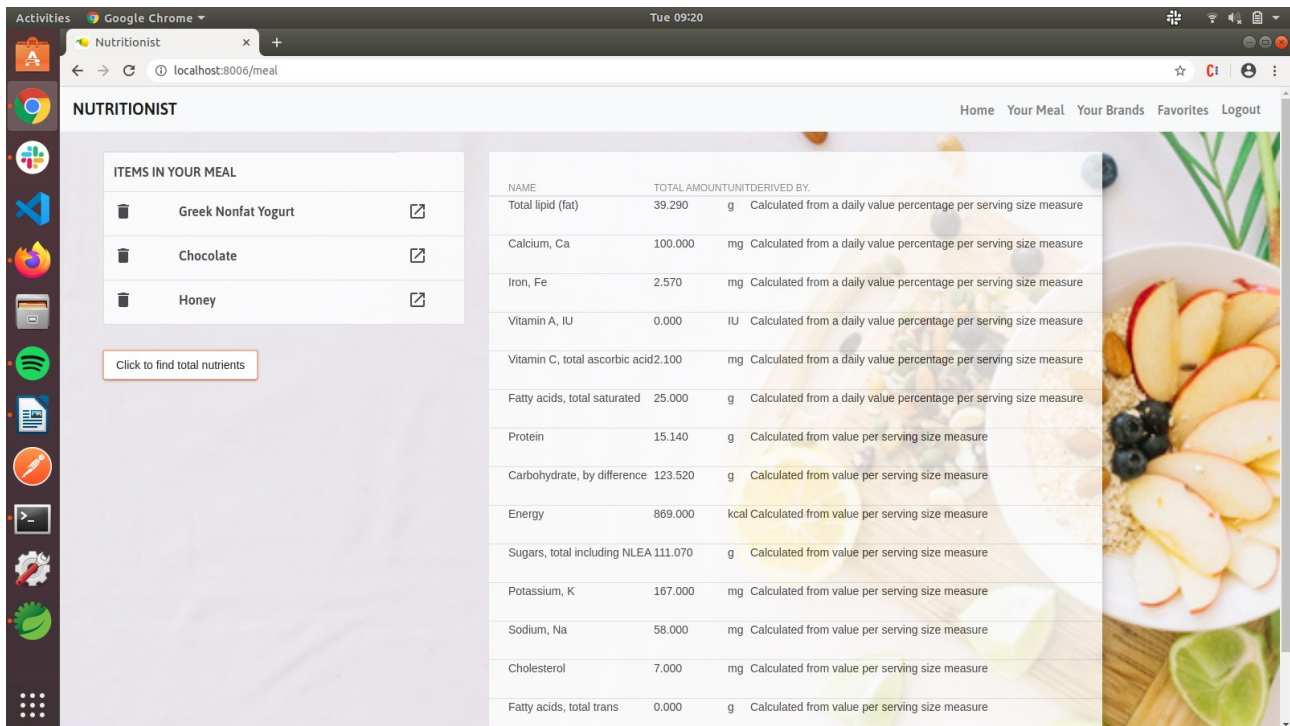### Recomended Food Module

Fetch all the food items based on the brand of food which the user ordered mostly.

##Meal Module

  We have added a Meal module to display the total nutrients of total Food items together.

Tech Stack

- Spring Boot
- MySQL
- Angular- CI (Gitlab Runner)
- Docker, Docker Compose

Demonstrate the entire application

1. Make sure all the functionalities are implemented
2. Make sure both the UI (Component and Services) and server side (For all layers) codes are unit tested.
3. All the Services are up and running using docker (Docker should be used for running them)
4. E2E tests should be executed and shown.
5. Application is completely responsive in nature.