 # 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.

Requirements

- 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://ndb.nal.usda.gov/ndb/api/doc**

  Reference APIs:
  https://api.nal.usda.gov/ndb/search/?format=json&q=broccoli&sort=n&max=25&offset=0&api_key=<API KEY>

  https://api.nal.usda.gov/ndb/V2/reports?ndbno=45225267&type=b&format=json&api_key=<API KEY>

- A frontend web app where the user can **register/login** to the application, **search for a specific food**, **show list of matching food items, view nutrition content for a selected food item** and **bookmark favourite food** for later reference.
- User can **add a food item into favourite list** and should be able to **view the favourite food items** for user.

Modules

UserService - should be able to manage user accounts.
UI (User interface) - should be able to

1. Search for a specific food item and show list of matching foods
2. View nutrition details of a selected food
3. Add a food to favourite list
4. should be able to see favourite food items
5. UI should be responsive which can run smoothly on various devices

FavouriteService - should be able to store all the favourite foods for a user

Tech Stack
- Spring Boot
- Angular
- CI (Gitlab Runner)

- Docker, Docker Compose

Flow of Modules

Building frontend
- Building responsive views:
    1. Register/Login
    2. Search for a specific food item and show list of matching foods - populating from external API
    3. Show nutrition content for a selected food - populating from external API
    4. Build a view to show favourite foods
- Using Services to populate these data in views
- Stitching these views using Routes and Guards
- Making the UI Responsive
- E2E test cases and unit test cases
- Writing CI configuration file
- Dockerize the frontend

Building the UserService
- Creating a server in Spring Boot to facilitate user registration and login using JWT token and MySQL
- Writing swagger documentation
- Unit Testing
- Write CI Configuration
- Dockerize the application
- Write docker-compose file to build both frontend and backend application

Building the Favourite Service
- Building a server in Spring Boot to facilitate CRUD operation over favourite food items stored in MySQL
- Writing Swagger Documentation
- Write Test Cases
- Write CI Configuration
- Dockerize the application
- Update the docker-compose

Demonstrate the entire application