NUTRITON ASSISTANT APPLICATION

Contents

[Introduction 1](#_Toc160555198)

[Overview 1](#_Toc160555199)

[Features 1](#_Toc160555200)

[Installation Guide 1](#_Toc160555201)

[Prerequisites 1](#_Toc160555202)

[Backend Setup (Spring Boot with Gradle) 2](#_Toc160555203)

[Frontend Setup (React) 2](#_Toc160555204)

[Accessing the Application 2](#_Toc160555205)

[Usage Guide 2](#_Toc160555206)

[Dashboard 2](#_Toc160555207)

[Meal Planning 3](#_Toc160555208)

[Food Logging 3](#_Toc160555209)

[Nutrition Goals 3](#_Toc160555210)

[Project Structure Overview 4](#_Toc160555211)

[Frontend (React) 4](#_Toc160555212)

[Components 4](#_Toc160555213)

[Services 4](#_Toc160555214)

[Assets 4](#_Toc160555215)

[Backend (Spring Boot) 5](#_Toc160555216)

[Controller 5](#_Toc160555217)

[Model 5](#_Toc160555218)

[Repository 5](#_Toc160555219)

[Service 5](#_Toc160555220)

[Application Configuration 6](#_Toc160555221)

# Introduction

Welcome to the documentation for the Nutrition Assistant project! This documentation provides an overview of the Nutrition Assistant, including its features, installation instructions, project structure, roles, and responsibilities.

## Overview

The Nutrition Assistant is a web application designed to help users plan their meals and maintain a healthy diet. With a focus on simplicity and effectiveness, the Nutrition Assistant offers pre-defined 7-day diet plans, meal suggestions, and calorie tracking to assist users in achieving their nutritional goals.

## Features

The Nutrition Assistant offers the following key features:

* **7-Day Diet Plans**: Provides pre-defined diet plans for each day of the week, including breakfast, lunch, snacks, and dinner.
* **Meal Suggestions**: Offers nutritious meal suggestions for each day, categorized by meal type and accompanied by calorie counts.
* **Customization**: Allows users to customize their meal plans by selecting alternative options or modifying existing meals to suit their preferences and dietary requirements.
* **Calorie Tracking**: Displays daily calorie totals, along with protein, carbohydrate, fiber, fat, and sodium content for each meal plan, enabling users to monitor their nutritional intake.
* **Modern Interface**: Features a user-friendly interface with responsive design for easy navigation and readability across various devices.

# Installation Guide

This installation guide provides step-by-step instructions for setting up the Nutrition Assistant project, which uses Spring Boot with Gradle for the backend, MongoDB as the database, and React for the frontend.

## Prerequisites

Before you begin the installation process, ensure that you have the following prerequisites installed:

* Java Development Kit (JDK)
* Node.js and npm (Node Package Manager)
* MongoDB
* Gradle

## Backend Setup (Spring Boot with Gradle)

1. Clone the repository from the GitHub repository: link-to-repo.
2. Navigate to the backend directory of the project.

bash

cd backend

Open the project in your preferred IDE (e.g., IntelliJ IDEA, Eclipse).

Configure the MongoDB connection in the application.properties file located in src/main/resources.

Build the Spring Boot project using Gradle.

bash

gradle build

Run the Spring Boot application.

bash

1. gradle bootRun

## Frontend Setup (React)

1. Navigate to the frontend directory of the project.

bash

cd frontend

Install dependencies using npm.

bash

npm install

Start the React development server.

bash

1. npm start

## Accessing the Application

Once the backend and frontend servers are running, you can access the Nutrition Assistant application by navigating to http://localhost:3000 in your web browser.

# Usage Guide

The Nutrition Assistant project provides a platform for tracking nutritional goals, managing meals, and monitoring food intake. This guide outlines the usage instructions for different components of the application.

## Dashboard

The dashboard serves as the main interface for users to track their nutrition goals, view meal plans, and manage food intake. Here's how to use the dashboard:

1. Upon logging in, you will be redirected to the dashboard.
2. The dashboard displays nutritional goals, meal plans, and food intake statistics.
3. Navigate through different sections using the sidebar menu.
4. View nutritional goals and progress towards achieving them.
5. Access meal planning tools and food logging features.
6. Customize the dashboard layout and preferences as needed.

## Meal Planning

The meal planning feature allows users to create and schedule meals for each day. Follow these steps to use the meal planning feature:

1. Navigate to the "Meal Planner" section of the dashboard.
2. Choose the desired day for meal planning.
3. Add breakfast, lunch, dinner, and snacks using the provided options.
4. Customize meal details such as ingredients, portion sizes, and calorie counts.
5. Save the meal plan for future reference.
6. Edit or delete existing meal plans as needed.

## Food Logging

Food logging enables users to track their food intake and monitor nutritional values. Here's how to log food entries:

1. Access the "Food Intake" section of the dashboard.
2. View a list of previously logged food entries.
3. Use the "Add Food" button to log a new food entry.
4. Enter details such as food name, category, serving size, and nutritional values.
5. Save the food entry to the database.
6. Edit or delete existing food entries as needed.

## Nutrition Goals

Nutrition goals help users set targets for calorie intake, macronutrients, and micronutrients. Follow these steps to manage nutrition goals:

1. Navigate to the "Nutrition Goal" section of the dashboard.
2. View current nutrition goals and progress towards achieving them.
3. Edit existing goals or set new targets for calorie intake, protein, carbohydrates, fats, and other nutrients.
4. Save changes to update the nutrition goals.
5. Monitor progress over time and adjust goals as necessary.

# Project Structure Overview

The Nutrition Assistant project follows a structured organization to ensure modularity, scalability, and maintainability. This section outlines the project structure for both the frontend (React) and backend (Spring Boot), along with additional details and explanations.

## Frontend (React)

The frontend of the Nutrition Assistant application is built using React.js, a popular JavaScript library for building user interfaces. React components are organized within the src/components directory, while service functions for API interactions reside in the src/services directory. Static assets such as images and CSS stylesheets are stored in the src/assets directory.

### Components

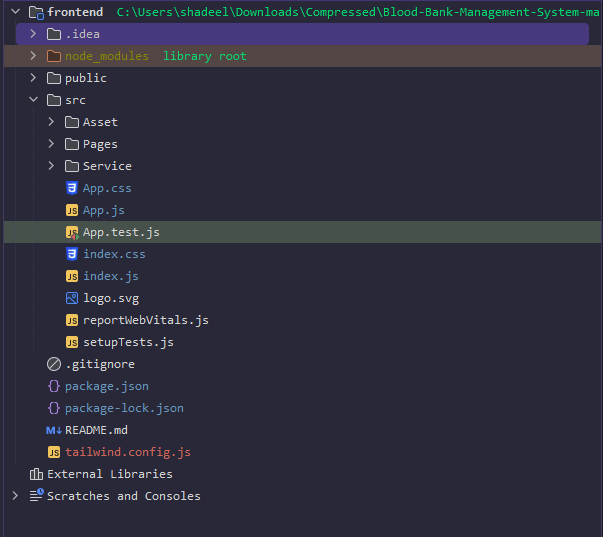
* **Dashboard.js**: This component represents the main dashboard of the application, displaying various sections like meal planner, food intake, etc.
* **MealPlanner.js**: The meal planner component allows users to plan their meals for specific days.
* **FoodIntake.js**: This component enables users to track their food intake and monitor their nutritional consumption.
* **NutritionGoal.js**: The nutrition goal component allows users to set and track their nutrition-related goals.

### Services

* **FoodService.js**: Service functions for fetching food-related data from the backend API.
* **NutritionService.js**: Service functions for managing nutrition goals and related operations.

### Assets

* **images/**: Directory for storing image files used in the application.
* **css/**: Directory for CSS stylesheets used to style the components.



## Backend (Spring Boot)

The backend of the Nutrition Assistant application is implemented using Spring Boot, a powerful framework for building Java-based web applications. MongoDB is used as the database for storing application data. The backend follows the MVC (Model-View-Controller) architecture pattern, with controllers handling HTTP requests, services implementing business logic, and repositories managing data access to MongoDB.

### Controller

* **FoodController.java**: Controller class responsible for handling HTTP requests related to food entities.
* **NutritionGoalController.java**: Controller class for managing HTTP requests related to nutrition goals.

### Model

* **Food.java**: POJO (Plain Old Java Object) representing food entities stored in the database.
* **NutritionGoal.java**: POJO representing nutrition goal entities.

## Repository

* **FoodRepository.java**: Interface defining CRUD (Create, Read, Update, Delete) operations for food entities in the database.
* **NutritionGoalRepository.java**: Interface defining CRUD operations for nutrition goal entities.

## Service

* **FoodService.java**: Interface defining service methods for food-related operations.
* **NutritionGoalService.java**: Interface defining service methods for managing nutrition goals.
* **FoodServiceImpl.java**: Implementation class for food-related service methods.
* **NutritionGoalServiceImpl.java**: Implementation class for nutrition goal service methods.

## Application Configuration

* **application.properties**: Configuration file for Spring Boot application properties, including MongoDB connection details.

