Техническое задание: "Just-Cook-It" — Сайт поиска и сохранения рецептов

Задача

Разработать веб-приложение для управления кулинарными рецептами. Система позволяет пользователям находить блюда по имеющимся продуктам и сохранять их в избранное, а также регистрироваться для сохранения избранных рецептов.

Функциональные требования (поэтапная реализация)

Этап 1: Базовый функционал (Неделя 1-4)
Цель: рабочее backend-приложение на Java с базовой логикой и хранением данных в БД.

1. Управление рецептами:
   - Поля рецепта: название, описание, инструкция по приготовлению, список ингредиентов (продуктов).

2. Управление данными в БД:
   - Поиск и просмотр списка продуктов(внутри базы данных).
   - Каждый продукт имеет: название, единицу измерения (шт., гр., мл., ст.л. и т.д.).
   - Каждый рецепт имеет: Название ,количество необходимых продуктов и интсрукцию по приготовлению.

3. Поиск рецептов по продуктам:
   - Функция, которая принимает список продуктов и возвращает рецепты, которые можно приготовить из этих продуктов.

4. Система хранения данных:
   - Сохранение всех данных о рецептах и продуктах в реляционной базе данных (PostgreSQL).
   - Организовать связи между рецептами и продуктами .

Критерии приемки Этапа 1:
   - Данные сохраняются в БД.
   - Функция для поиска рецептов работает исправно

Этап 2: Расширенная логика и пользователи (Неделя 5-8)
Цель: добавить систему пользователей и избранное.

5. Поиск рецептов по количеству продуктов:
   - Пользователь имеет возможность выбирать количество продуктов и получать список блюд.

6. Система пользователей и аутентификации:
   - Регистрация и аутентификация пользователей.

7. Личный кабинет и избранное:
   - Возможность добавлять рецепты в "Избранное".
   - Просмотр своего списка избранных рецептов.
   - Каждый пользователь видит только свои рецепты и свое избранное.

Критерии приемки Этапа 2:
   - Пользователь может зарегистрироваться и войти в систему.
   - Аутентифицированный пользователь может добавлять рецепты в избранное и просматривать их.
   - Реализован API для управления избранным (добавить/удалить/показать).
   - Написаны тесты для новых функций, включая тесты с аутентифицированным пользователем.

Этап 3: Контейнеризация и финальные штрихи (Неделя 8+)
Цель: подготовить приложение к развертыванию и добавить дополнительные функции.

8. Контейнеризация:
   - Создать Dockerfile для сборки образа Java-приложения.
   - Настроить docker-compose.yml для одновременного запуска приложения и базы данных PostgreSQL в контейнерах.

9. Дополнительные функции (по желанию/времени):
   - Категории рецептов: (завтрак, обед, ужин, десерт).
   - Время приготовления для рецепта.
   - Реализовать два варианта поиска рецептов:
      -*Базовый вариант: рецепты, в которых есть хотя бы один из указанных продуктов.*

      -*Улучшенный вариант: рецепты, для которых у пользователя есть все необходимые продукты.*

Критерии приемки Этапа
   - Все связи между контейнерами работают (приложение подключается к БД).
   - Интеграционные тесты могут запускаться как против локальной БД, так и против БД в контейнере.
   -Пользователь может искать рецепты по количуству продуктов , которые у него имеется.
   -Пользователь может добавлять рецепты в избранное и просматривать их в этом списке

Технический стек:

- Backend: Java 21 + Spring Boot
- База данных: PostgreSQL
- ORM: Spring Data JPA / Hibernate
- Тестирование: JUnit 5, Testcontainers (для интеграционных тестов с БД), Mockito
- Сборка и контейнеризация: Gradle, Docker
- API: REST API