Recipe Book API

This is the README for my CRUD API project. It is a CRUD (Create, Read, Update, Delete) Application that incorporates
Full stack development, HTML/CSS web design and Database management through the use of H2 Console.

Contents:
This file contains the following:
1. Resources
2. Brief
3. Kanban Board
4. Database Structure
6. Data Stack
7. Testing
8. FrontEnd
9. Further Improvements

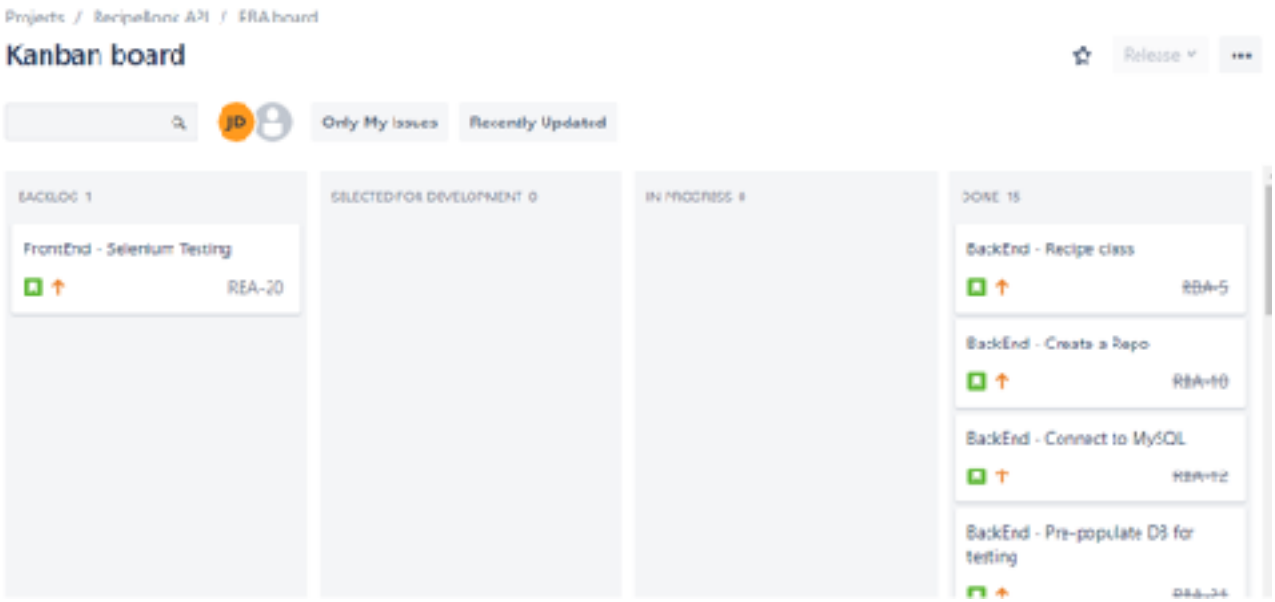
Resources:
BackEnd GitHub Repo:https://github.com/jdayih/RecipebookAPI
JIRA Board : https://my-site-name27.atlassian.net/secure/RapidBoard.jspa?rapidView=3&projectKey=RBA
FrontEnd GitHub Repo: https://github.com/jdayih/FRONTEND_RecipeAPI

Brief:
To create a CRUD application with the utilisation of supporting tools, methodologies and technologies that
encapsulate all core modules covered during training.

This project will involve concepts from all core training modules; more specifically this will involve:
- Project Management
- Databases
- JAVA SE
- Spring Boot
- FrontEnd Development
- Automated Testing

My Approach:
I have created a Recipe Book API database which implements CRUD functionality. On the front end the user can do
the following:
- Create a recipe
- Delete a recipe
- Update a recipe
- See the recipes already saved in the database

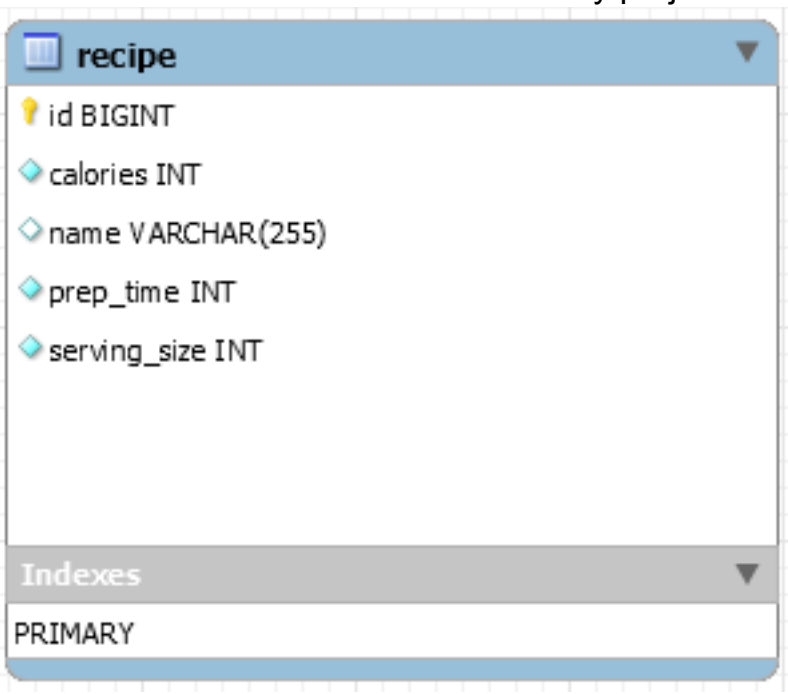
Kanban Board:
My Kanban board for this project can be accessed here:
https://my-site-name27.atlassian.net/secure/RapidBoard.jspa?rapidView=3&projectKey=RBA



The Jira Board is using the Agile Scrum method with a Sprint to fulfill the user stories to complete the criteria set in the brief.

Why I have created this project?
I created a Recipe Book API because I enjoy cooking but prefer finding recipes online as opposed to buying several recipe books. And jotting things down is so 2003. Jokes aside, this means that I can have my recipes with me wherever I go at the touch of a button.

Database Structure:
Below is an EER for the database of my project.

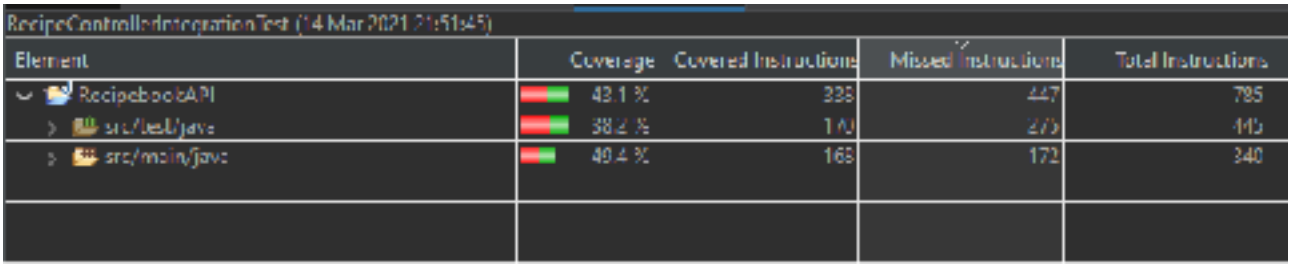


Data Stack:
Database: The database for this project is a local H2 console with a SQL database. H2 is the easiest database to get up and running, it is an in-memory database meant for testing and debugging.

Back-end: I have used Java and the Spring Boot Framework to power the backEnd of my API.\

Front-end: I have used HTML, JS and CSS to create the frontEnd of my API.

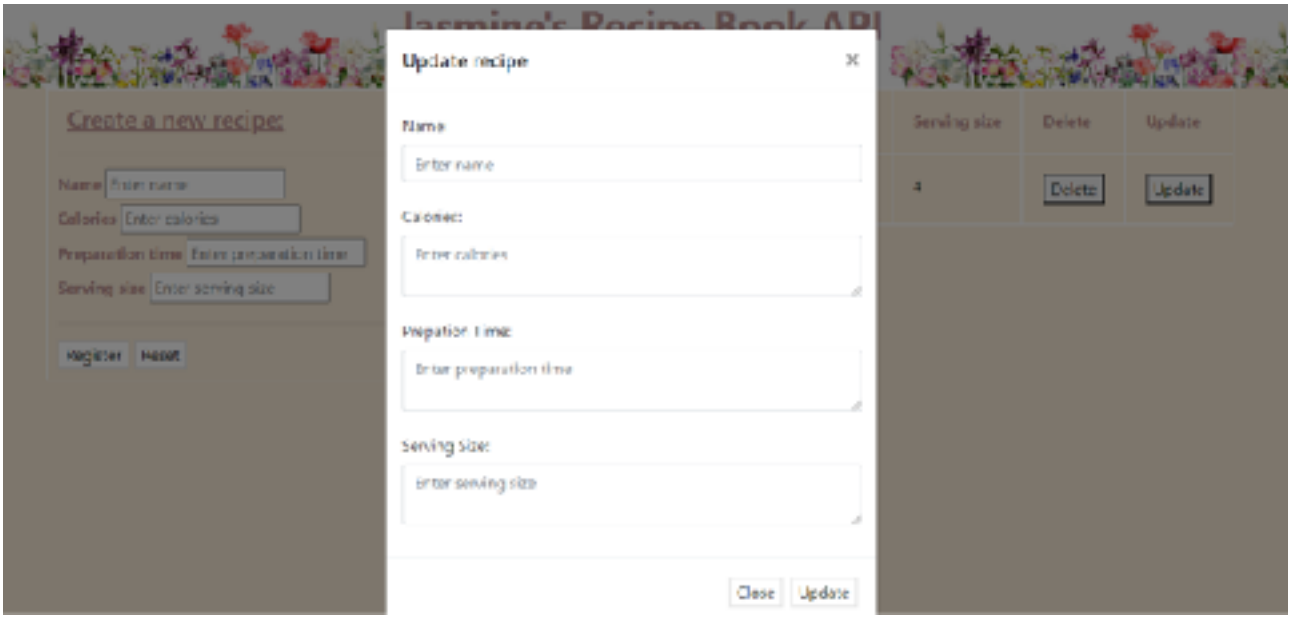
Testing:
My testing coverage can be seen below. Ideally, given more time, I would've increased the testing coverage by including more test scenarios.



Testing for my back-end was done through integration testing (using JUnit) and unit testing (using Mockito). The integration testing mocked the various HTTP requests. The tests were completed by using a mock repo created for testing purposes, passing in the request being tested, the body (if any is needed) for that specific request and the response we expect back out of it. By comparing the response that comes from the request compared to the one we expect, we can see whether the test failed or not.

Front-End:

Below are screenshots of the frontEnd of my Recipe Book API:



Future improvements:

The improvements I would like to make to my project are as follows:

- Enable the user to search for a recipe by name
- A connecting table that holds the recipe's cooking ingredients and instructions
- Increase the testing coverage
- Completed E2E testing on the frontEnd using Selenium