Documentation for recipe sharing web application “NomNom”

Main features of the application

In the application users can create their own profile, share recipes, search for recipes and also put comments and likes on the recipes they or others have made.
The app works as a web app which users can access using their browsers on computer or also their phones.
The data is stored in a database.

For a more detailed list of what users can exactly do:
Main features are that Users can

1. view recipes created by other users or themselves
2. register and authenticate in the system
3. change their password
4. add their own user profile image
5. change and delete their profile image
6. create their own recipes and add a title, body and image to the recipe
7. delete the recipes they have made
8. edit the recipes they have made
9. add likes or dislikes to recipes they or others have made
10. add comments to the recipes they or others have made
11. delete the comments they have made
12. search for recipes by their title

Main technologies used

How app is built

The backend of NomNom was written in Java with SpringBoot as the base. It was written following Model-View-Controller pattern. Various dependencies were used to implement and manage aspects of the application. Spring Web to manage the MVC controllers and server. Spring Data JPA and Hibernate to manage the persistence with JPA annotations. H2 database for testing and MySQL database as the real database. Junit and Mockito for testing. And Spring Security to handle user authentication. Also Maven was used for dependency management.

The frontend of the NomNom application was made using Thymleaf templates which receive data from controllers and HTML/CSS/JS with bootstrap library to help with styling and also TinyMCE to allow users to input formatted text for recipes.

List of documentation for the mentioned technologies:

Backend:

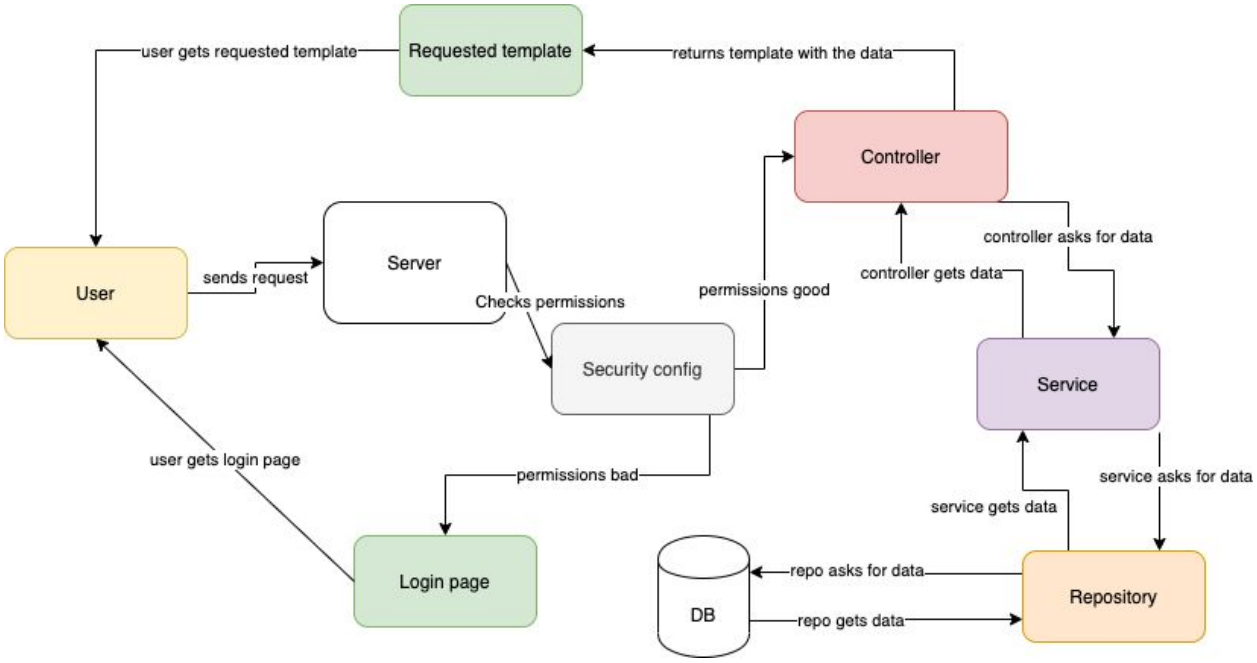
- Java documentation: https://docs.oracle.com/javase/8/docs/
- Spring boot general documentation: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
- Spring Data JPA documentation: https://docs.spring.io/spring-data/jpa/docs/2.3.2.RELEASE/reference/html/#reference
- Spring MVC documentation:
- https://docs.spring.io/spring/docs/current/spring-framework-reference/web.html#mvc
- Spring Security documentation:
- https://docs.spring.io/spring-security/site/docs/5.3.2.RELEASE/reference/html5/
- Maven documentation: https://maven.apache.org/guides/index.html
- Hibernate documentation: https://hibernate.org/orm/documentation/5.4/
- JUnit documentation: https://junit.org/junit5/docs/current/user-guide/
- Mockito documentation: https://javadoc.io/doc/org.mockito/mockito-core/latest/org/mockito/Mockito.html
- Thymeleaf documentation: https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html
- MySQL documentation: https://dev.mysql.com/doc/
- H2 Database documentation: https://www.h2database.com/html/quickstart.html

Frontend:

- Bootstrap documentation: https://getbootstrap.com/docs/4.5/getting-started/introduction/
- Jquery documentation: https://api.jquery.com/
- TinyMCE documentation: https://www.tiny.cloud/docs/

High level design of the application

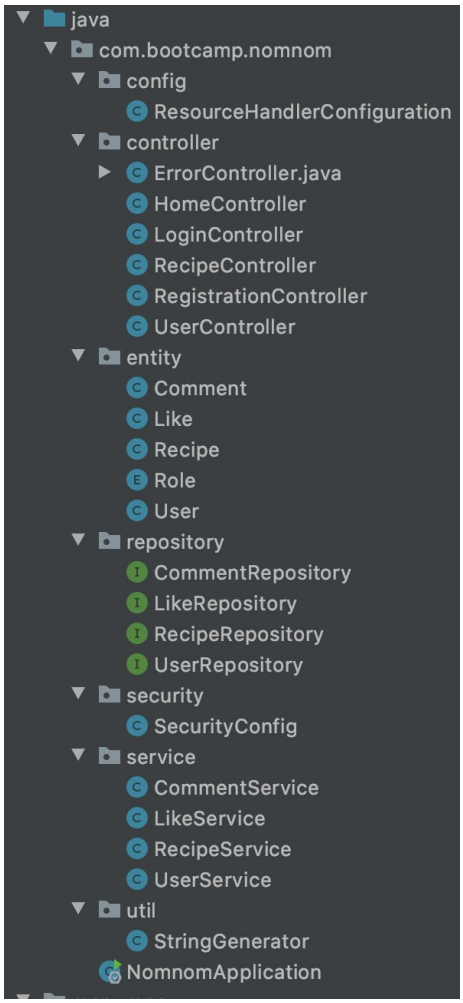
High level data flow



The NomNom application uses an MVC pattern where the user sends a request to the server. It forwards the request to the controller. Controller calls service methods where the main logic of the code is located. Service methods can access data from the database by using Repository methods. Which contain all the logic to retrieve, modify and store data from and to the database.

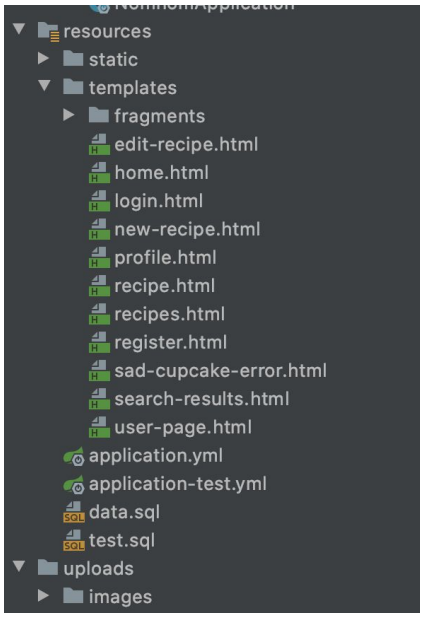
Also all the requests the user makes are checked by security configuration. If a user tries to access a resource or a page that requires authentication for example they will be redirected to the login page instead.

Class and Package structure of the project

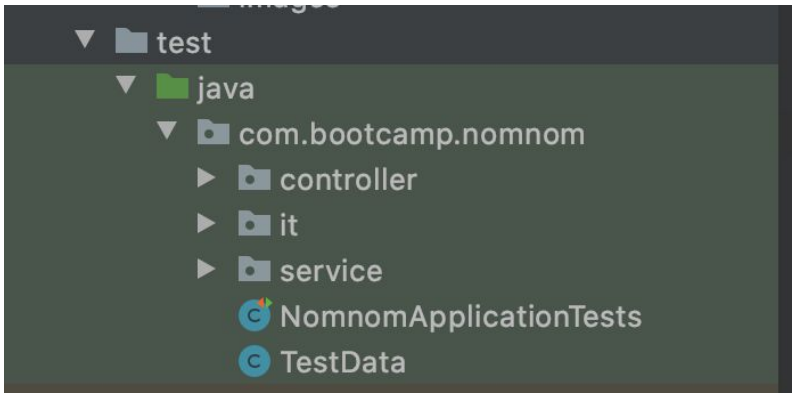


This is the package and class structure used in the NomNom application.
The controller package contains controller classes that are responsible for handling incoming user requests.
The Entity package contains JPA annotated entities by which database schema is created and updated
The Repository package has classes that have logic in them of how to retrieve, update and delete things from database.
The Security package has security related classes and configurations
The Service package contains all the backbone logic of the application. Where the data that has been received from controllers is handled and the appropriate data is handed back to the controller.
The Util package has utility classes that are needed for the application.
General naming convention is to name classes by what they are responsible for and then adding a descriptive title like “service”. Only exception is Entity classes.

Testing and resources



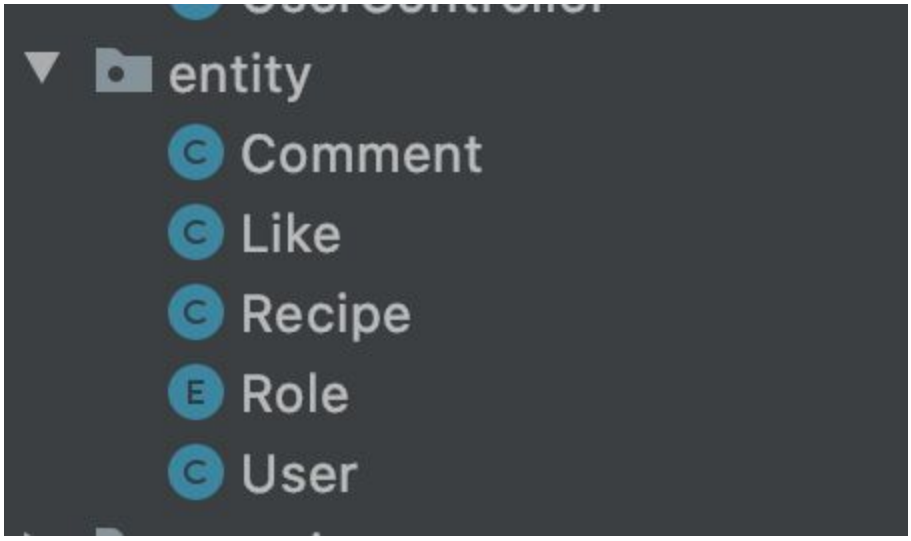
The resources are located in a resource folder. In the template folder we define all the thymleaf templates which are the ones the controllers give data to. Also in the static component we have the js and css styling files to style our templates. Also in the data.sql file we have the data we use to populate the database for testing purposes.



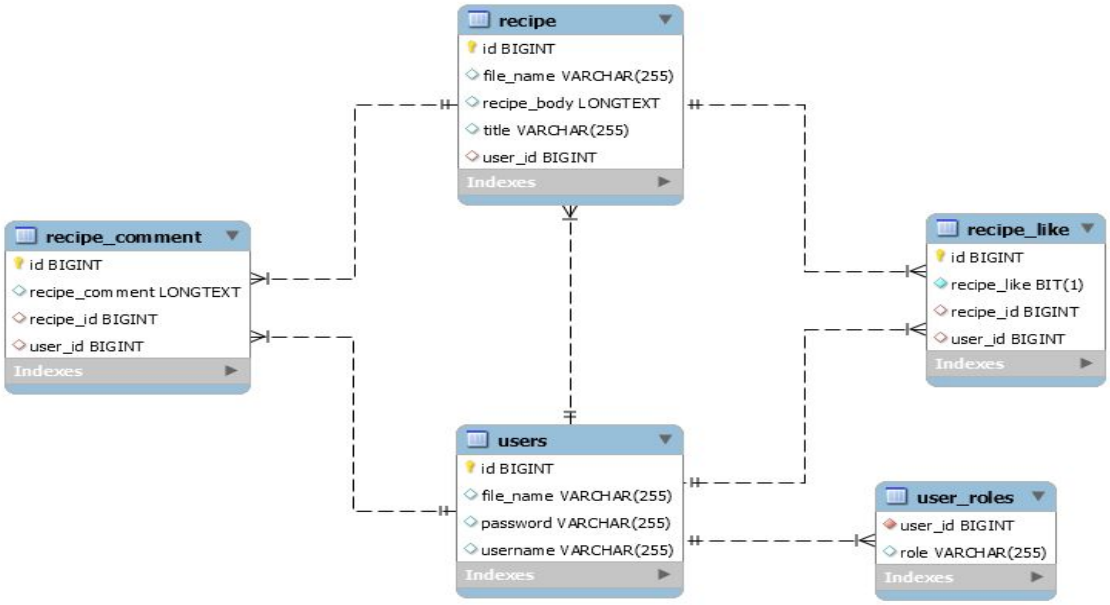
Our test classes are located in the Test directory. We have unit tests for service and controller methods and also a package for our integration tests. Generally naming convention for testing is to describe what the test does and put “Test” in the end.

Data persistence

The data in the NomNom application is stored in an SQL database. For testing and development we use H2 in-memory database but we also have implemented a config switch to MySQL database.
The schema was initially created and is updated by JPA annotations that written in the classes in the Entity package



The resulting database schema looks like this.



Also a part of data persistence is the recipe and user image saving. Those are kept in a folder on the web server. These images are accessible to users and have been given automatically generated names.