DESIGN DOC

of A Recipe Web Service

By Team DragonFruit:
Xuejuan Zhang, Yihan Tian, Xi Ling

Objective

The goal of this project is to provide a web service for people to share their own recipes.

A user can post recipes to the website, and can also browse others' recipes and rate those recipes. The author of a recipe can edit his/her recipes and also undo any ratings he/she has done before.

Registration of users is provided and necessary before the user can access the main service. A user is able to change the password of his/her account.

Methodology

Front end:	HTML, CSS and JavaScript
Back end:	Java with Spark Web Framework
Database:	MongoDB
Cloud Service Platform:	Heroku
Build Automation Tool:	Gradle

Goals

1. Design Unified Modeling Language (UML).

2. Implement the following RESTful APIs

(1). AutoIncrement API that enables the ID of users to monotonically increase.

(2). User API (userController) where users can not only register/login to obtain access to the main service, but also change passwords that are encrypted.

(3). MarkDelete API that marks specific recipes as deleted. Recipes that are marked as "deleted" would not display to users.

(4). Recipe API where users can create/read/update/delete recipes. For reading recipes operation, there are four parts. The first part is to allow users to view all the recipes in the database; the second part is to allow users to view recipes by a specific category; the third part is to allow users to view a specific recipe; the last part is to allow users to view recipes they posted.

(5). Rating API where users can create/read/delete ratings of related recipes. For reading ratings operation, users can not only view all ratings of a specific recipe, but also view the ratings user rated.

For the deleting recipe operation, if one recipe has been marked as "deleted", meaning delete, the rating would also be deleted simultaneously. For the deleting rating operation, we just delete the rating.

3. Setup a MongoDB database to store User/Recipe/Rating/AutoIncrement/MarkDelete information.

4. Build a frontend application that is connected with the backend to support the project's functionalities.

5. Deploy the web application to Heroku.

Access

Source Code in GitHub:

https://github.ccs.neu.edu/xuejzhang/DragonFruit

Special notes about mongoDB:

When desired, another mongoDB can be used in place of the current one. To do that, simply replace the content of the String variable called “mongoData” to the new database access address. For example:

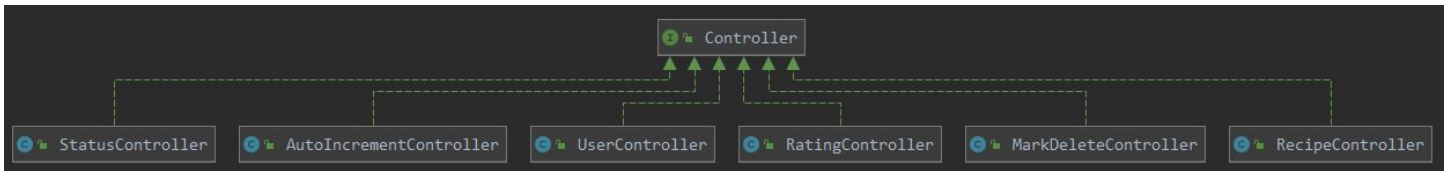
```
public AutoIncrementController() {

    String mongoData =
            "mongodb+srv://admin:admin@cluster0-zdcag.mongodb.net/test?retryWrites=true&w=majority";
```

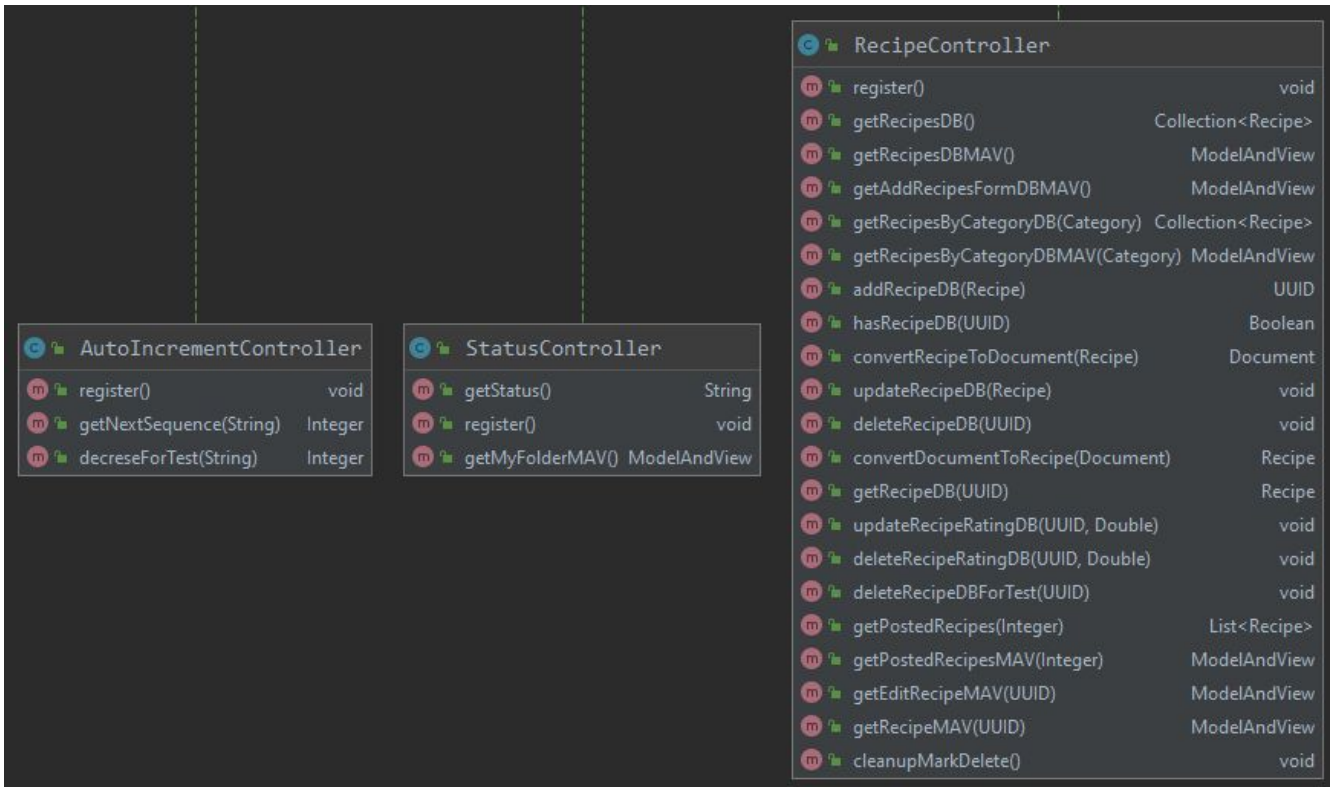
The places where modification are needed are as follows:

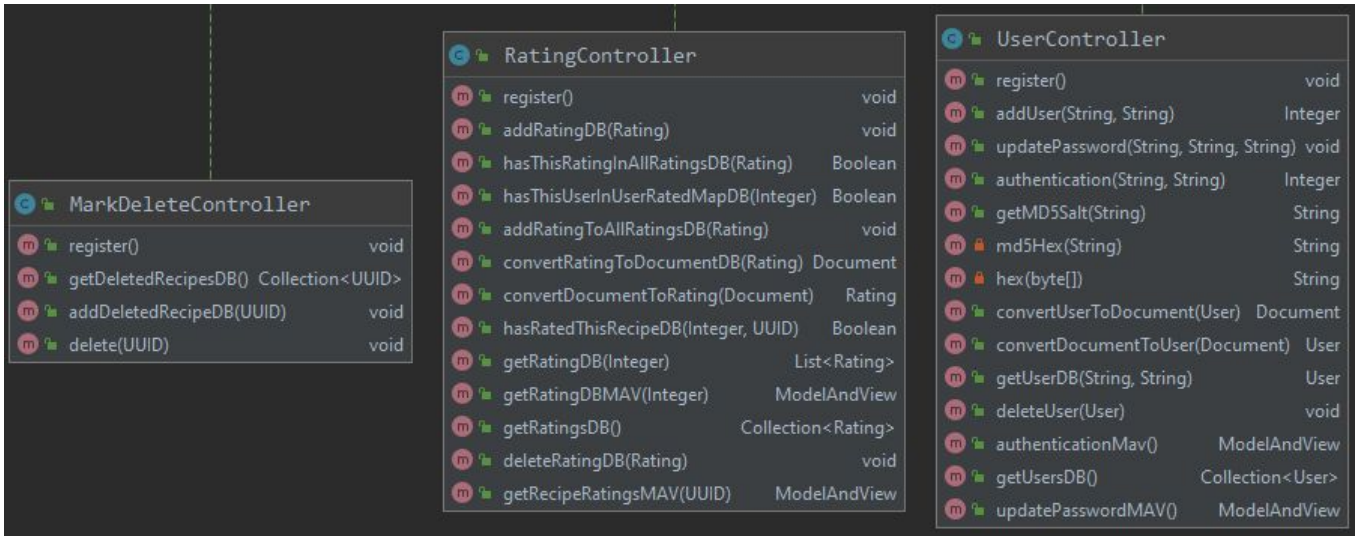
1. AutoIncrementController.java
2. MarkDeleteController.java
3. RatingController.java
4. RecipeController.java
5. StatusController.java
6. UserController.java

UML Diagrams (Controllers General Look)

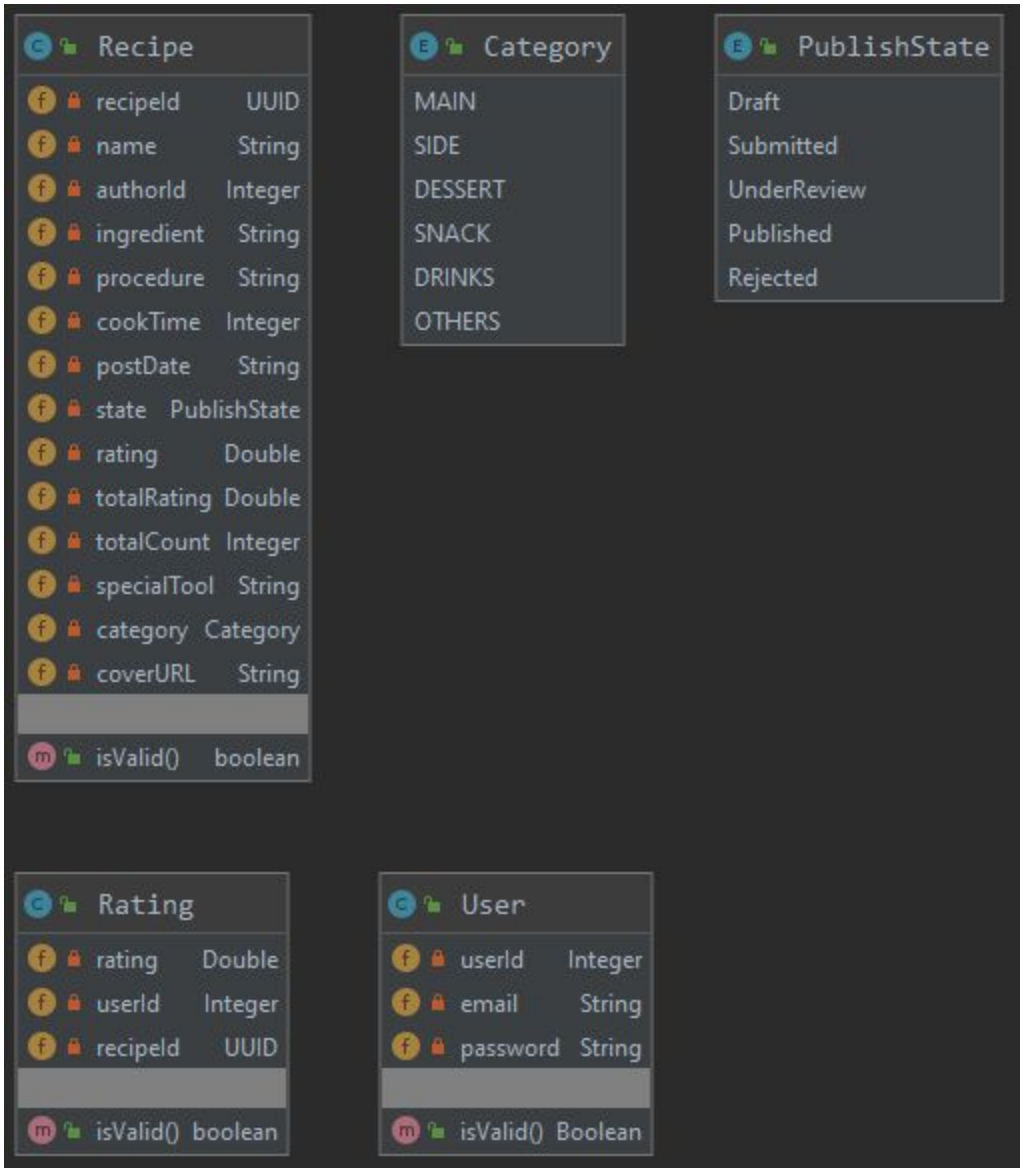


UML Diagrams (Controllers Close-up Details)

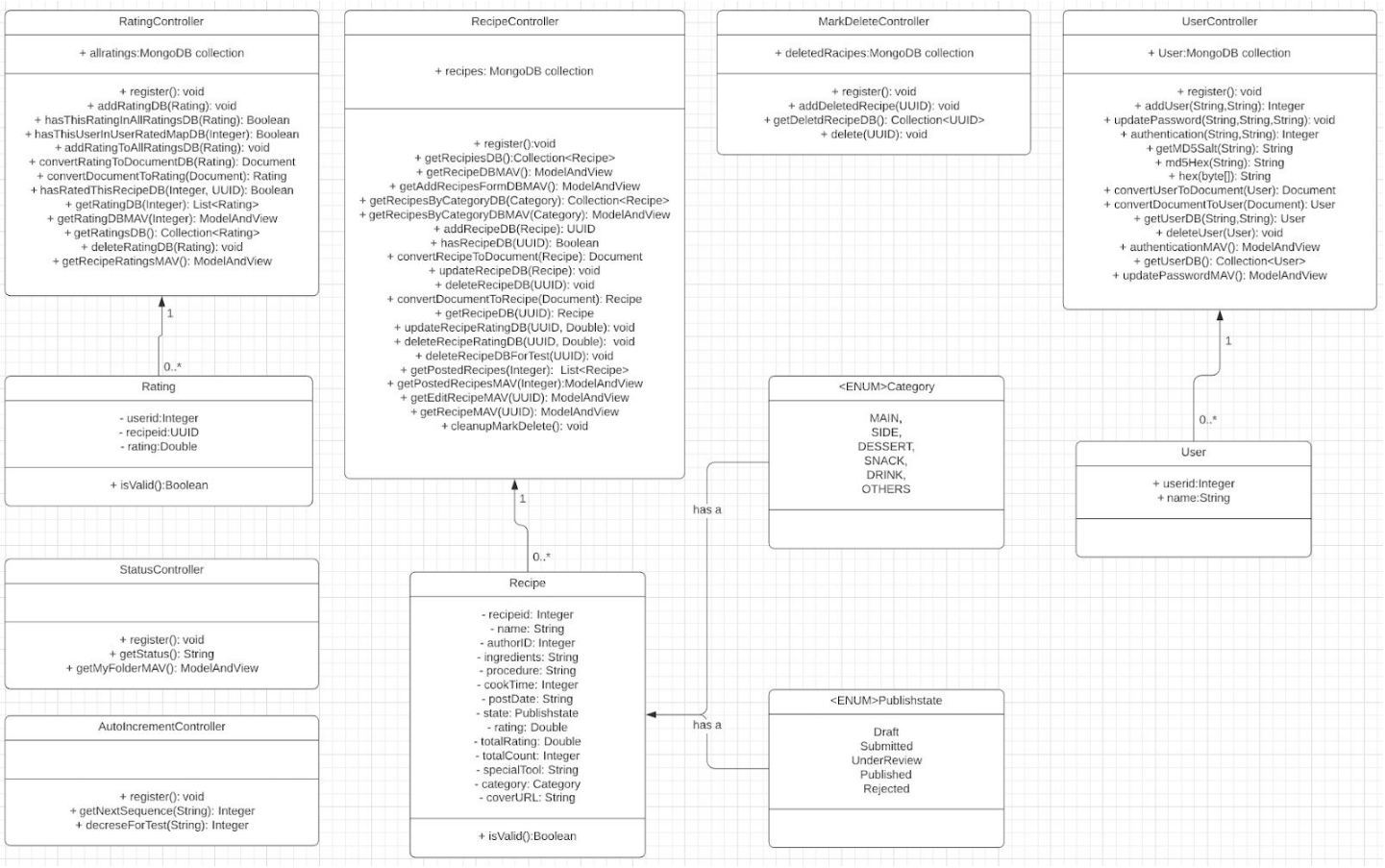




UML Diagrams (Models)



ER Diagrams



ER Diagram is also available at:

https://www.lucidchart.com/documents/edit/2e7cb07d-82d2-4e28-a2d3-f8e0b49ab9c0/0_0?shared=true

Presentation ER Diagram is also available at:

https://www.lucidchart.com/invitations/accept/b2d7672b-f64b-427d-afa9-621eed211904

API Specification

1.Recipe API

URL:	/recipemav
HTTP Request Type:	GET
Functionality:	Display all recipes in the database. Noted that those recipes that are marked as “deleted” will not be shown.

URL:	/recipe
HTTP Request Type:	POST
Functionality:	Add a new recipe to the database.

URL:	/recipe/:id
HTTP Request Type:	GET
Functionality:	Get a recipe by recipe id.

URL:	/recipe/catagory/:category
HTTP Request Type:	GET
Functionality:	Get all recipes under the given category.

URL:	/editrecipe/:id
HTTP Request Type:	GET
Functionality:	Redirect to a webpage where a recipe with given id can be edited.

URL:	/updaterecipe
HTTP Request Type:	POST
Functionality:	Edit an existing recipe with the given recipe id. It is done by replacing the posted data with that of in the database.

URL:	/postedrecipe/:userId
HTTP Request Type:	GET
Functionality:	Redirect to a webpage where all recipes posted by a given user are listed.

URL:	/deleterecipe
HTTP Request Type:	POST
Functionality:	Delete a recipe by marking it deleted, but still keep its database entry.

URL:	/test_delete_recipe
HTTP Request Type:	POST
Functionality:	For testing purposes ONLY. Delete a recipe completely.

2.MarkDelete API

URL:	/test_cleanup_marked
HTTP Request Type:	POST
Functionality:	For testing purposes ONLY. Check and ensure the consistency of the databases after testing.

3.Rating API

URL:	/ratings
HTTP Request Type:	GET
Functionality:	Get all ratings of a given recipe.

URL:	/rating:userid
HTTP Request Type:	GET
Functionality:	Get all ratings posted by a given user.

URL:	/addRating

HTTP Request Type:	POST
Functionality:	Add a rating to a recipe.

URL:	/deleterating
HTTP Request Type:	POST
Functionality:	Delete a rating.

URL:	/rating
HTTP Request Type:	POST
Functionality:	For testing purposes ONLY. Add a rating to a recipe.

URL:	/myfolder
HTTP Request Type:	GET
Functionality:	Redirect to a webpage which allows the user to edit his/her posted recipes, ratings and user info.

URL:	/
HTTP Request Type:	GET
Functionality:	Redirect to the main landing of the website which is the user login page.

4.User API

URL:	/fakelogin
HTTP Request Type:	GET
Functionality:	For testing purposes ONLY. Generate a fixed user id for the ease of testing.

URL:	/login
HTTP Request Type:	POST
Functionality:	Verify a user's email and password. If valid, let the user login and redirect to our main page that displays all recipes in the database.

URL:	/signup
HTTP Request Type:	POST
Functionality:	Allow a user register, and obtain the access to the main service.

URL:	/updatepassword
HTTP Request Type:	POST
Functionality:	Verify user's identity. Allow him or her to change the account password.

URL:	/authentication
HTTP Request Type:	GET
Functionality:	Display user authentication interface.

URL:	/updatepasswordmav
HTTP Request Type:	GET
Functionality:	Display the interface where the user can change the account password.

URL:	/deleteuserfortest
HTTP Request Type:	POST
Functionality:	For testing purposes only. Delete the user that has registered to the main service.