

SW Engineering CSC648/848 Summer 2024

## **ScholarEats**

### Team 4:

Angelo Arriaga (Team Lead - aarriaga1@sfsu.edu)

Donovan Taylor (Front-End Lead)

Hancun Guo (Front-End)

Tina Chou (Front-End)

Edward McDonald (Back-End Lead)

Karl Carsola (Back-End)

Sai Bavisetti (Database)

Maeve Fitzpatrick (Docs Editor)

Sabrina Diaz-Erazo (GitHub Master)

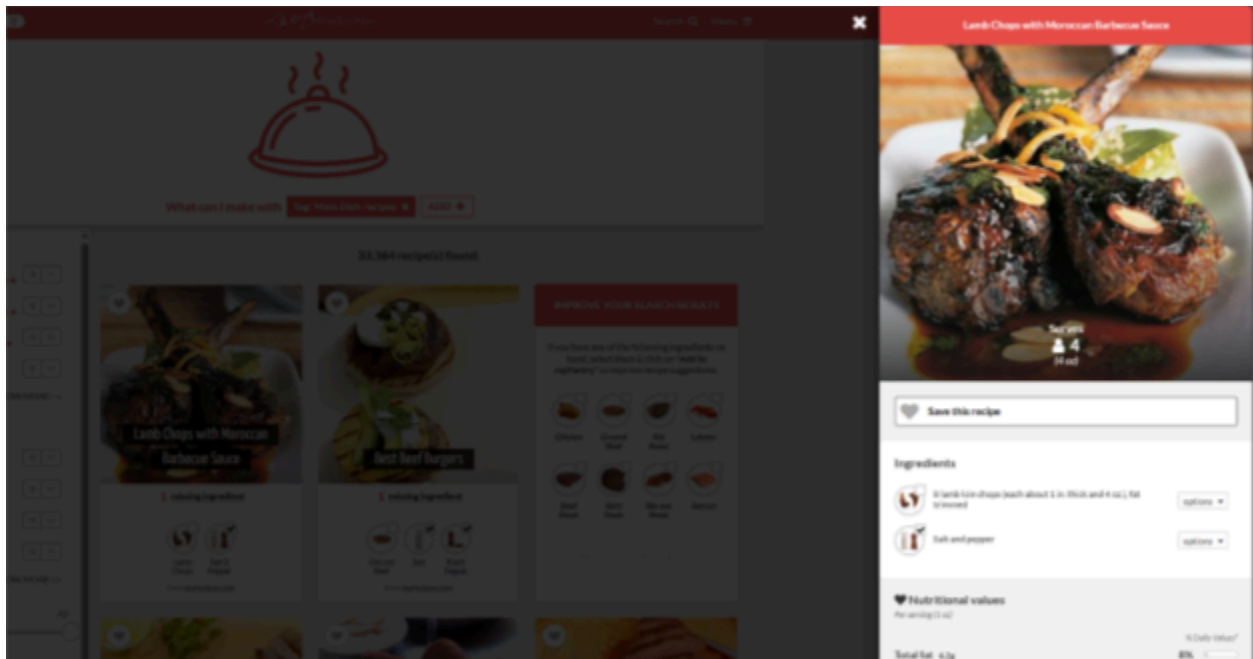
## **Milestone 4**

07/30/24

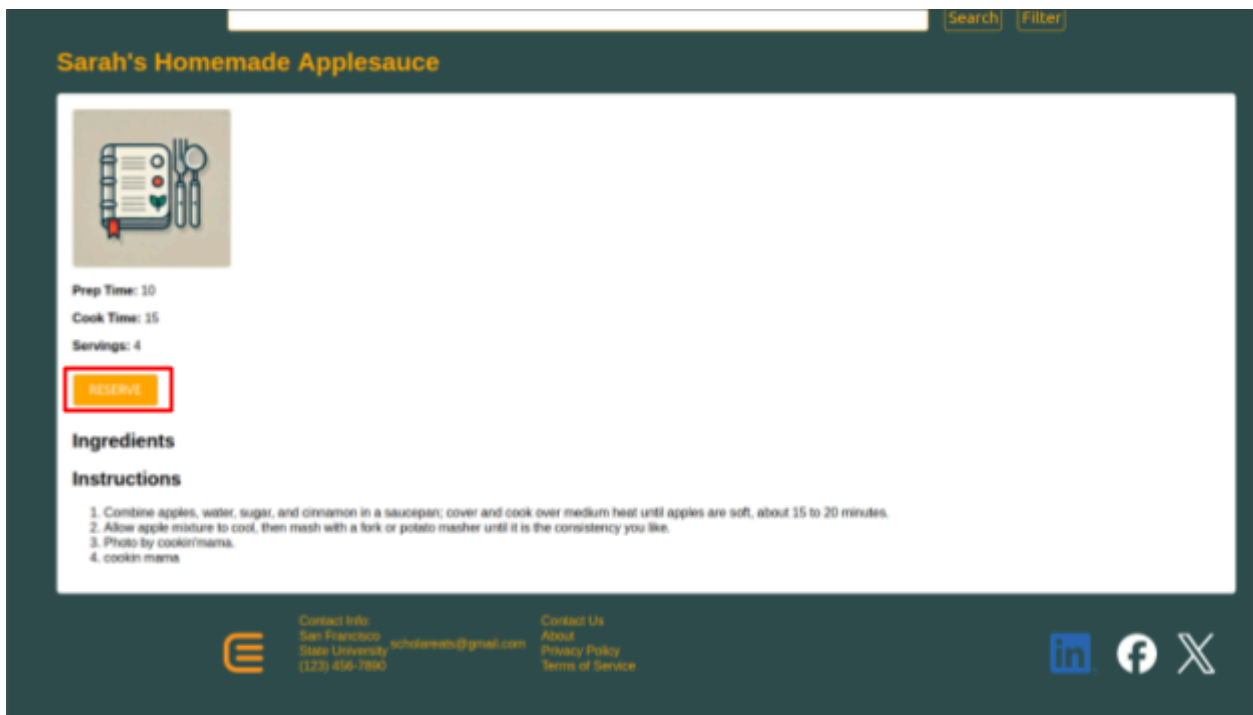
History (Revisions)	
07/30/2024	Milestone 4 Version 1 Submission
07/23/2024	Milestone 3 Version 1 Submission
07/22/2024	Milestone 2 Version 2 Submission
07/09/2024	Milestone 2 Version 1 Submission
06/26/2024	Milestone 1 Version 1 Re-Submission
06/20/2024	Milestone 1 Version 1 Submission

## **1. Product Summary**

- Name: ScholarEats
- List of major committed functions:
  1. ADMINISTRATORS SHALL BE ABLE TO AUTHENTICATE THE USERS.
  2. ADMINISTRATORS SHALL BE ABLE TO BLACKLIST/ WHITELIST A USER
  3. ADMINISTRATORS SHALL BE ABLE TO EDIT THE INGREDIENT LIST
  4. ADMINISTRATORS SHALL BE ABLE TO REMOVE USER ACCOUNTS
  5. ADMINISTRATORS SHALL BE ABLE TO FOLLOW THE EXPIRATION DATE CLOSELY AND TAKE THE NECESSARY ACTIONS.
  6. SYSTEM SHALL AUTOMATICALLY ENROLL USERS IN CORRESPONDING UNIVERSITY FOOD PROGRAMS
  7. SYSTEM SHALL BE ABLE TO GENERATE RECOMMENDED RECIPES
  8. SYSTEM SHALL TELL ADMINISTRATORS WHEN AN INGREDIENT HAS RUN OUT OF STOCK
  9. SYSTEM SHALL TELL ADMINISTRATORS WHEN FOOD HAS SPOILED
  10. USERS SHALL BE ABLE TO CHANGE THEIR PASSWORD
  11. USERS SHALL BE ABLE TO EDIT THEIR USERNAME
  12. USERS SHALL BE ABLE TO FILTER RECIPES BY COOKING AIDS REQUIRED
  13. USERS SHALL BE ABLE TO FILTER RECIPES BY DIETARY RESTRICTIONS
  14. USERS SHALL BE ABLE TO FILTER RECIPES BY DIFFICULTY
  15. USERS SHALL BE ABLE TO FILTER RECIPES BY INGREDIENTS
  16. USERS SHALL BE ABLE TO MAKE ACCOUNTS WITH VALID UNIVERSITY EMAIL.
  17. USERS SHALL BE ABLE TO SET ALLERGIES
  18. USERS SHALL BE ABLE TO SET DIETARY RESTRICTIONS
  19. USERS SHALL BE ABLE TO SORT RECIPES BY CALORIES
  20. USERS SHALL BE ABLE TO SORT RECIPES BY FAT
  21. USERS SHALL BE ABLE TO SORT RECIPES BY FIBER
  22. USERS SHALL BE ABLE TO SORT RECIPES BY PROTEIN
  23. USERS SHALL BE ABLE TO VIEW AVAILABLE INGREDIENTS
  24. USERS SHALL BE ABLE TO VIEW RECIPES
- What makes our product unique: we automatically enroll eligible academic emails into their school's food pantry (comparison screenshots on next page)



Our competitor: foodcombo. No reserve option.



Us: reserve option.

- URL: <http://3.148.145.110:3000/>

## **2. Usability Test Plan**

### **Test Objectives**

#### **Function 1: Search Recipes**

- **Test Objective:** Ensure users can quickly and accurately find the recipes they need using the search bar and filters.

#### **Function 2: Reserving Something That is Available**

- **Test Objective:** Ensure users can successfully reserve an available item, such as a cooking ingredient.

#### **Function 3: Update Allergy Information**

- **Test Objective:** Ensure users can conveniently update their allergy information on the account management page.

#### **Function 4: Change Dietary Restrictions**

- **Test Objective:** Ensure users can easily change their dietary restrictions on the account management page.

#### **Function 5: Manage User Profile**

- **Test Objective:** Ensure users can conveniently update and manage their profile information.

### **Test Description**

#### **Function 1: Search Recipes**

- **System Setup:** Latest version of the browser, connected to the internet.
- **Starting Point:** Start from the homepage of the website.
- **Intended Users:** Home cooks and cooking enthusiasts looking for specific recipes.
- **URL of the System:** <http://3.148.145.110:3000/recipes>
- **What to Measure:** Measure the time taken and accuracy of users finding the desired recipes using the search bar and filters.

#### **Function 2: Reserving Something That is Available**

- **System Setup:** Latest version of the browser, connected to the internet.
- **Starting Point:** Start from the ingredients page.
- **Intended Users:** Users who want to reserve cooking ingredients.
- **URL of the System:** <http://3.148.145.110:3000/ingredients>
- **What to Measure:** Measure the time taken and the number of steps for users to complete a reservation.

### **Function 3: Update Allergy Information**

- **System Setup:** Latest version of the browser, connected to the internet.
- **Starting Point:** Start from the account management page.
- **Intended Users:** Users who need to update their allergy information.
- **URL of the System:** <http://3.148.145.110:3000/accountManagement>
- **What to Measure:** Measure the time taken and the number of steps for users to update their allergy information.

### **Function 4: Change Dietary Restrictions**

- **System Setup:** Latest version of the browser, connected to the internet.
- **Starting Point:** Start from the account management page.
- **Intended Users:** Users who need to change their dietary restrictions.
- **URL of the System:** <http://3.148.145.110:3000/accountManagement>
- **What to Measure:** Measure the time taken and the number of steps for users to change their dietary restrictions.

### **Function 5: Manage User Profile**

- **System Setup:** Latest version of the browser, connected to the internet.
- **Starting Point:** Start from the user account page.
- **Intended Users:** Users who want to update their personal information.
- **URL of the System:** <http://3.148.145.110:3000/accountManagement>
- **What to Measure:** Measure the time taken and the number of steps for users to update their profile information.

Function	Task Description	Success Count	Total Count	Success Rate
Search Recipes	Can users successfully find the desired recipe?	9	10	90%
Reserving Something That is Available	Can users successfully reserve an available item?	8	10	80%
Update Allergy Information	Can users successfully update their allergy information?	7	10	70%
Change Dietary Restrictions	Can users successfully change their dietary restrictions?	9	10	90%
Manage User Profile	Can users successfully update their profile?	10	10	100%

Function	Task Description	Average Time (seconds)	Average Steps
Search Recipes	Time taken to find the desired recipe	20	3
Reserving Something That is Available	Time taken to reserve an available item	30	4
Update Allergy Information	Time taken to update allergy information	15	3
Change Dietary	Time taken to change	10	3

Restrictions	dietary restrictions		
Manage User Profile	Time taken to update profile information	10	4

#### Function 1: Search Recipes

1. I found this function easy to use. **Strongly Agree**
2. I am satisfied with the experience of using this function. **Agree**
3. I find the design of this function reasonable. **Agree**

#### Function 2: Reserving Something That is Available

1. I found this function easy to use. **Agree**
2. I am satisfied with the experience of using this function. **Agree**
3. I find the design of this function reasonable. **netural**

#### Function 3: Update Allergy Information

1. I found this function easy to use. **Agree**
2. I am satisfied with the experience of using this function. **Agree**
3. I find the design of this function reasonable. **Agree**

#### Function 4: Change Dietary Restrictions

1. I found this function easy to use. **Agree**
2. I am satisfied with the experience of using this function. **Strongly Agree**
3. I find the design of this function reasonable. **Netural**

#### Function 5: Manage User Profile

1. I found this function easy to use. **Strongly Agree**
2. I am satisfied with the experience of using this function. **Strongly Agree**
3. I find the design of this function reasonable. **Strongly Agree**

### 3. QA Test Plan

#	Description	Input	Expected Output	Pass /Fail
1	Test new user registration.	User navigate to register page and enter "abc123" as username, "Abc45678" as password.	User successfully registered. Admin can see user in backend.	Pass
2	Test user login.	User navigate to login page, enter "abc123" as username, "Abc45678" as password, submit login form	User is successfully logged in.	Pass
3	Test viewing the list of available ingredients.	Navigate to the ingredients section	List of ingredients is displayed.	Pass
4	Test searching for a specific ingredient/ recipe.	Enter "apple pie" in the search bar. Submit the search query.	Relevant search results are displayed, such as "apple pie filling, apple pie by Grandma Ople..."	Pass
5	Test viewing the list of available recipes.	Navigate to the recipe section.	List of recipes is displayed.	Pass
6	Test viewing detailed information for a recipe.	Click on "apple pie filling" to view its details.	Recipe details, including ingredients and cooking instructions, are displayed.	Pass
7	<u>Test Reserve button in single recipe page.</u>	<u>On "apple pie filling" recipe page, click on reserve button.</u>	<u>Recipe reserved by user.</u>	<u>Pass</u>
8	<u>Test editing, and</u>	<u>On admin tools page,</u>	<u>Firstly, "abc123" user is</u>	<u>Pass</u>



	<u>deleting users.</u>	<u>firstly change “abc123” user’s username to “abc456”. Then delete user.</u>	<u>gone, “abc456” user comes up. Then “abc456” user is gone.</u>	
9	<u>Test adding, editing, and deleting ingredients.</u>	<u>On admin tools page, firstly add an ingredient called “random”, then edit its name to “random2”, then delete this ingredient.</u>	<u>Firstly, ingredient “random” is added to ingredient page. Then “random” is gone and “random2” is added to ingredient page. Lastly, “random2 is removed from ingredient page.</u>	<u>Pass</u>
10	<u>Test adding, editing, and deleting recipes.</u>	<u>On admin tools page, firstly add a recipe called “random”, then edit its name to “random2”, and add instructions to it, then delete this recipe.</u>	<u>Firstly, ingredient “random” is added to recipe page. Then “random” is gone and “random2” is added to recipe page with instructions. Lastly, “random2 is removed from recipe page.</u>	

## 4. Code Review

### 4a) Our Coding Convention:

#### **Naming of Variables:**

Ideally variables will be named in a single word that can capture their identity where possible.

However in cases where a variable must be named with multiple words in order to properly convey its purpose, variables are to be named in the “**camelCase**” manner.

For example, the case in which the sum or count of a “thing” is needed to be stored you would write it as **sumThings** or **countThings** (*note we omit “Of” in this*)

GLOBAL VARIABLES ARE ALLCAPS

#### **Method Names:**

Similar to above, we will name methods in “**camelCase**”.

Note that for many things specifically in the “front-end” domain they exist using arrow-functions with things like **app.use(...)** and call existing things like **res.render(...)**.

This will largely apply to helper functions like the ones defined in the “back-end”, if any.

#### **Script Names:**

Related to the above, a lot of things, like EventListeners, will be implemented by simply creating a series of statements in a .js file and including the script within the .hbs through a **res.render()** call.

In this case the script should follow that naming convention.

#### **Database Specifics:**

We should probably use **db.execute(...)** over **db.query** based on security concerns as it helps prevent SQL injections as execute does the database operation on the back-end whereas query does the operation on the front-end.

Note that usage is similar however if we do use this then we need to use “await” and “promises”. I can give examples if needed.

Names will be snake\_case for column names (i.e. snake\_case)

### **Naming Convention of Branches:**

For branches we utilize names consisting of only lowercase letters and underscores “\_”.

Any branches currently not following this convention will be changed to do so, **keep in mind that this will add a step to your next push as you will have to update what remote repository branch you are pushing to**

Guidance can be found at [this link here](#).

---

## **Design Conventions**

### **Routing Structures:**

We should have **ALL** routes within the main routes folder which is /application/routes within the repository

We can however separate “front-end” and “back-end” for each route like “userRoutes\_FE” and “userRoutes\_BE”

### **Testing Features:**

Do **NOT** make separate app.js’s and commit them to branches that you are trying to merge into main/master.

Master branch should be production ready code, free of testing aids. If you feel the need to include something to demonstrate functionality and it doesn't exist, implement a view and use routing to make it possible to access in the same application as a separate page.

Additionally, doing so will better demonstrate how the code would actually be implemented in existing pages via the existing architecture we developed, (i.e. dynamically feeding in scripts via `res.render()`)

Again, do not make separate applications with their own launching conditions or packages / `package.json` / `package-lock.json`.

**Failure to do so will result in your PR being denied until the branch is in compliance**

#### **Handlebars and CSS:**

We should be moving to utilize the “grid-area” form of setting up the layouts. This will enable us to better support displays of all dimensions

As always, we should have a singular “layout” that maintains consistency throughout all pages.

The unique and different pages will be done via “views” which will be inserted within the **body** of the html.

Note that each page also has a “title” that we alter via our `res.render()` calls by passing in a value. For the sake of consistency this field should be a “header” tag within the “body” tag but it should be **defined within the layout** and not within each page's independent views. This is something that we are currently not complying with and hurts the consistency of our pages

Cascading Style Sheet (.css) will be passed in via the `res.render()` as we have decided upon before, this will allow easier implementation of “themes” or additional/conditional formatting. (i.e. dark mode)

## updated SQL to match updated DB terminology

### #46

Edit

<> Code

Merged

sderazo merged 3 commits into master from SQL\_Update 2 days ago

Conversation 2

Commits 3

Checks 0

Files changed 21

+80 -1,026

EdwardMcD commented 3 days ago

IN TABLE expired\_products

- renamed Name to name

IN TABLE notifications

- added column university

IN TABLE recipes

- renamed Unnamed: 0 to recipe\_id
- renamed dietary restrictions to dietary\_restrictions
- renamed cooking tip to cooking\_tip
- deleted MyUnknownColumn

IN TABLE store

- renamed Name to name

TABLE Users

- changed Users to users

Reviewers

sderazo

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

Notifications

Customize

Unsubscribe

You're receiving notifications because you modified the open/close state.

2 participants

Lock conversation

updated SQL to match updated DB terminology

04a53e3

sderazo requested changes 2 days ago

View reviewed changes

sderazo left a comment

- changes made in each file are necessary for variables to have the same naming convention
- in-line comments are good and descriptive
- in recipeRoutes\_BE.js: is the commented out block of code still needed?
- please include at least a minimal code header in each file

EdwardMcD added 2 commits 2 days ago

Added headers to files the backend team has worked on

f6f3dc9

Deleted unnecessary folder "Backend"

a0e1291

sderazo approved these changes 2 days ago

View reviewed changes

sderazo left a comment

- the files look good (each one has a code header and the commented out code was deleted)


sderazo merged commit f2d9aa0 into master

2 days ago

Revert

## Backend development #47

[Edit](#) [Code](#)

 Merged

 sderazo merged 9 commits into `master` from `backend_development` 17 minutes ago

 Conversation 2

 Commits 9

 Checks 0

 Files changed 12

+1,045 -498



EdwardMcD commented yesterday

- Added support for multiple universities
- You only see the recipes available for the ingredients belonging to your own university, and you only see ingredients available for your own university
- You can now click "reserve" when viewing a recipe, which will notify the admins associated with your university that you have reserved a recipe
- You can now only see notifications that pertain to you



Reviewers



sderazo



Assignees

No one—[assign yourself](#)

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

Notifications

Customize

 Unsubscribe

You're receiving notifications because you modified the open/close state.

2 participants



 Lock conversation




EdwardMcD commented 9 hours ago

Author

PR Update:

- If you are logged out, the notification page does not crash
- Notifications now show the correct recipe
- If your university does not have an admin, you cannot reserve that recipe.
- After reserving a recipe, it now directs you back to that recipes page
- Image now properly shows up on the individual ingredients page (NOTE: this will need to change (and will most likely break) when sai finishes updating the database, and I do not have the ability to edit the table(s) that this is effected by)



 sderazo approved these changes 18 minutes ago

[View reviewed changes](#)

sderazo left a comment

- in-line comments are detailed
- formatting is consistent and neat
- name of variables follow agreed-upon convention

files that need code header:

- authentication.js
- blacklist.js
- users.js
- individual\_recipes\_view.hbs

Edward's PR update (in conversation tab) addresses the problems I ran into when I first reviewed backend\_development's changes on the instance. Note: I listed out the problems I ran into in Discord instead of having them inside of this PR review. This is bad practice, so next time when I review a PR, I will comment on the instance in here.

issues from app on instance that will be addressed in a separate PR:

- admin tools shows up as unfinished button
- buttons in account management page need to be smaller (too long) (frontend will handle this)



 sderazo merged commit 46fbc14 into `master` 17 minutes ago

[Revert](#)










Pull request successfully merged and closed


You're all set—the `backend_developm...` branch can be safely deleted.

[Delete branch](#)



4) c



 Sabrina Diaz-Erazo



  Reply  Reply all  Forward   ...

To:  Michelle Nguyen

Sun 7/28/2024 3:56 PM

 ae0421c6-97c4-42ff-aa39-8c... 57 KB 


 recipeRoutes\_BE.txt 15 KB 







2 attachments (72 KB)  Save all to OneDrive - San Francisco State University  Download all


Hello Michelle,

My name is Sabrina (from Team 4) and I will be conducting your team's external code review. I have attached the file that our team would like for your team to review to this email (I had to change the extension from .js to .txt since outlook wouldn't let me send it). I have also included a document outlining the coding and design conventions we followed.



Best,  
Sabrina Diaz-Erazo  
GitHub Master  
Team 4



 Michelle Nguyen



  Reply  Reply all  Forward   ...

To:  Sabrina Diaz-Erazo

Mon 7/29/2024 8:50 PM

 Progress.txt 15 KB 

 UsersController.java 2 KB 

2 attachments (17 KB)  Save all to OneDrive - San Francisco State University  Download all

Hello Sabrina,


Hi Team 4,







I will be conducting your team's external code review.


Attached, you will find the file that our team would like your team to review. Please review our code for the back and front end. The files should be called: "progress.js" and "UserController.java".

We appreciate your time and effort in reviewing our code. Your feedback is valuable to us, and we look forward to your constructive comments.

Best,  
Michelle  
Team03

 Sabrina Diaz-Erazo

      ...

To:  Michelle Nguyen

Mon 7/29/2024 9:02 PM

Hello Michelle,

According to the Milestone 4 Document, it states the following: "Team's code review can be performed to only one file or script from your project." Please let me know which single file you and your team would like for me to review. Thank you for taking the time to review our code. We look forward to see how we can improve it.

Best,  
Sabrina Diaz-Erazo  
Team 4  
GitHub Master

...



Michelle Nguyen

To: @ Sabrina Diaz-Erazo



Mon 7/29/2024 9:47 PM

Please review any. I recommend the backend UserController.

...



Michelle Nguyen

To: @ Sabrina Diaz-Erazo

Cc: @ Uzair Hamed Mohammed



Mon 7/29/2024 11:20 PM

Hello Sabrina,

Our team (Uzair and I) reviewed your code that you gave us and here are some feedback we saw in your code. You all need a better layout/design of your application. For example we chose the MVC design pattern for the backend, and thanks to it, our files are very organized and easy to find. We are not sure how javascript projects work, but it looks like you all dumped everything in one file?

Additionally...

1. Avoid hardcoding credentials in the code.
2. Use environment variables or a secure vault to manage sensitive information
3. The error message in the res.status(400).json response should be more user-friendly and avoid exposing internal error codes directly to the user. Consider logging the detailed error internally and providing a generic message to the user
4. The comments are helpful, but ensure they are up-to-date and accurate. The comment about IS\_LOGGED\_IN is unclear without additional context.

I hope we were able to help you review your code. Let us know (Uzair and I) if you have any concerns or questions.

Best,

Michelle & Uzair

Team03

...





Sabrina Diaz-Erazo

To: 🌐 Michelle Nguyen



Mon 7/29/2024 11:40 PM

Hello Michelle,

The following are my comments for UserController.java:

- The code header and in-line comments are missing from this file so it took me a bit of time to understand the purpose of this code.
- I see a "/login" endpoint, is there a matching "/logout" endpoint; if so, shouldn't it be included within this class?
- line 57: function for /login endpoint:  
return ResponseEntity.ok().body("User authenticated successfully");
  - this message would be more fitting for someone registering for an account for the first time
  - an alternate message could be "User logged in successfully"
- Good job with consistency in naming variables and functions with camelCase.
- Message for not logging in successfully follows good practice.

I hope that I was able to help review your code. Please let me know if you have any questions.

Best,  
Sabrina Diaz-Erazo  
Team 4  
GitHub Master

↩ Reply

➡ Forward

## 5. Self-Check on Best Practices for Security - ½ Page

We are currently protecting database login credentials, passwords, and SQL queries.

Database login credentials are not hard coded into each file, and are instead stored in the environment, separate from the code, and loaded through `dotenv`. The file is named `process.env`, and it is stored at the root of our application.

An example of loading variables from the .env file

```
const connection = mysql.createPool({
  host:      process.env.DB_HOST,
  user:      process.env.DB_USER,
  password:  process.env.DB_PASS,
  database:  process.env.DB_NAME
});
```

We are encrypting passwords using 'bcrypt', a node.js library to hash (and salt) passwords before putting them in the database. Currently, we are hashing the password with 10 distinct salts, like so:

```
const hash = await bcrypt.hash(password, 10);
```

For example, in practice, if a user creates an account with the password "password," it appears in the database like so:

username	password_hash
tester1234	\$2a\$10\$SCaFII/V.J.uRDHIhNV55eJ52Y0MFycF...

To protect SQL queries, we are using parameterized queries and await execute() rather than query(), even when we are just returning values. This is how we validate input data in the search bar. In the example below, we directly inject the search input into the query.

```
// Search input from the search bar
if (searchInput) {
  query += ` AND recipe_name LIKE ${searchInput}`;
```

This is bad practice, as it allows for SQL injections. Instead, we should push the search input to the array of query parameters, and execute them individually utilizing SQL's ? placeholder variable as shown below.

```
// Search input from the search bar
if (searchInput) {
  query += ' AND `recipe_name` LIKE ?';
  queryParams.push(`%${searchInput}%`);
}
```

Along with hashing and salting, we also verify that user information meets the criteria during the registration process. Usernames must be at least 5 characters long, passwords must be at least 8 characters long, and the two password fields must match during registration.

## 6. Self-Check: Adherence to Original Non-Functional Specs

### List of Non-Functional Requirements

- Privacy & Security:
  - Users must check and confirm their preferences regarding personal information upon account creation.
    - **DONE**
  - Passwords MUST be encrypted and NOT PLAINTEXT for security reasons
    - **DONE**
  - Emails must be validated (be an email from the universities domain)
    - **DONE**
  - WE WILL NOT SELL YOUR DATA TO AI
    - **DONE**
- Expected Load:
  - Support 200,000 unique students **ON TRACK**
- Response Time:
  - System shall respond visually within 7 seconds
    - **DONE**
  - File Size of Photos shall not exceed 10MiB
    - **ISSUE:** Our team has decided that the users do not have the ability to upload their own photos on our website.
  - File formats for photos shall include:
    - .png
    - .jpeg
    - .heic
    - .heif
    - **ISSUE:** Our team has decided that the users do not have the ability to upload their own photos on our website..
- Stylistic:
  - Each page shall has the program's logo visible in the header
    - **DONE**
- Hardware:
  - Display dimensions supported(css):
    - Phone **ON TRACK**
    - Tablet **ON TRACK**
    - Monitor **DONE**
- Software:
  - We will support:
    - Firefox **DONE**
    - Chromium **DONE**

**7. List of contributions to the document and contributions score for all the team members. Also email with feedback/complaints. If you do not have complaints, then at least provide some feedback about the dynamics of the team for this milestone.**

1. Donovan Taylor (Score 10/10)
  - a. Adjusted landing page to have a more distinct blurb describing the site
  - b. Fixed the admin tools routes
  - c. The true Index page"/" loads the Landing Page instead of the Contact Us Page
2. Hancun Gao (Score 7/10)
  - a. Registration .css repaired
  - b. "Reserve" Button on Recipe page added
  - c. Addressed QA Test Plan in the Milestone 4 Document
  - d. Ensured we updated the statuses of all Non-Functional Requirements in Section 6 of Milestone 4 Document
3. Edward McDonald (Score 10/10)
  - a. Hooked up functionality to the Recipes page to update the database
  - b. Added the usage of notifications to notify user of available ingredients to pick up
  - c. Examined the repository to ensure coding practices were followed
  - d. Retool-ed queries and routes to support different universities to demonstrate the "per-university" functionality of the product
  - e. Made examples for how to show separate universities's data and ingredients within the site to her per-university functionality
  - f. Hooked up Sabrina's dark-mode implementation to the database
  - g. Documented how we validated user input in forms and in SQL queries and the steps we took to avoid malicious efforts
  - h. Went through repository to ensure we utilized db.execute over db.query
  - i. Documented how passwords were encrypted
  - j. Worked to better Auto-Complete so it only worked on what was available
4. Karl Carsola (Score 8/10)
  - a. Implemented "Forgot Password" Functionality (check)
  - b. Moved the project away from using the alert function as some browsers prevent its usage
  - c. Incorporated Captcha
  - d. Addressed QA Test Plan in the Milestone 4 Document
  - e. Ensured we updated the statuses of all Non-Functional Requirements in Section 6 of Milestone 4 Document
5. Sai Bavisetti (Score 6/10)
  - a. Deleted Erroneous Entries
  - b. Assisted in the development of the Admin Tools used to modify the university's possessed food store.

6. Maeve Fitzpatrick (Score 7/10)
  - a. Reworked coloring of .css sitewide to maximize readability and usability
  - b. Reworked .css sitewide to dynamically change with screen width between standard, tablet, and mobile views
7. Sabrina Diaz-Erazo (Score 8/10)
  - a. Privacy Policy when registering is accessible
  - b. Assisted Edward in hooking up “Dark Mode” to the user-preferences table (CHECK)
  - c. Conducted the Code-Review process with other teams and attached the screenshots to the Milestone 4 document
  - d. Adjusted Privacy Policy to accommodate non-functional requirement stating we will not sell user-data for AI purposes
8. Tina Chou (Score 7/10)
  - a. Reworked .css to have navbar buttons be less rounded and have margins between options
  - b. Made adjustments to have user set dietary restrictions on registration
  - c. Contributed to the Usability Test Plan