Foodie Connector

Sprint 1 Retrospective

Team 25: Zequan Wu, Shuqi Ma, Shiwen Xu, Xiaonan Shen

What went well?

User Story #1

As a user, I would like to register for an account using my email address.

Sub-task	Estimated Time	Responsible Person
Create ‘users’ table on the database	1h	Xiaonan Shen
Implement an API for registering	1h	Xiaonan Shen
Create a user interface for registering on the web client	4h	Shiwen Xu
Unit test - correct and incorrect input	2h	Shiwen Xu

Completed:

We did some research on the register methods, and we think using the email with the username for sign up. E-mail is convenient for users to remember and for future function(verification, reset, etc). We would send the information to the backend for checking whether the email is already used or not. Then the email and password information will auto-filled to the log-in page for quick login.

User Story #2

As a user, I would like to log in.

Sub-task	Estimated Time	Responsible Person
Implement an API for login	1h	Xiaonan Shen
Create a user interface for login on the web client	4h	Shiwen Xu
Unit test - correct and incorrect passwords	2h	Shiwen Xu

Completed:

After we implemented API and user interface for login, users can use their email and password to log in and will be redirected to restaurant list page if the backend sends the success responses, or the user will get the corresponding notification of unsuccess.

User Story #3

As a user, I would like to reset my password if I forget it.

Sub-task	Estimated Time	Responsible Person
Setup SMTP client on the backend so that we can send emails to users	2h	Xiaonan Shen
Implement an API for sending verification codes	1h	Xiaonan Shen
Implement an API for resetting password	1h	Xiaonan Shen
Unit test - correct and incorrect verification code (via email)	3h	Shiwen Xu

Completed:

After the email is verified, the user will be able to reset the password. We will send an 8-digit verification code to the user's email address. Then the user will be able to set a new password using the code. The APIs are protected by throttle limits, so the user will be able to send at most one resetting email per minute.

User Story #4

As a user, I would like to save my address so that I don't have to type them in every time.

Sub-task	Estimated Time	Responsible Person
Create ‘addresses’ and ‘users_addresses’ tables in the database	1h	Xiaonan Shen
Implement an API that returns all the addresses of the current user	1h	Xiaonan Shen
Implement an API for adding addresses	1h	Xiaonan Shen
Implement an API for editing addresses	1h	Xiaonan Shen
Implement an API for deleting addresses	1h	Xiaonan Shen
Setup Google Maps JavaScript API on the web client	4h	Shuqi Ma
Create a user interface for showing address list	2h	Shuqi Ma
Create a user interface for adding addresses	2h	Shuqi Ma

Completed:

We use the table to show the saved address and edit button for users to edit them. And the “ADD” button is for adding the new address. We set up a Google API to get users' current geolocation and converted address and auto-filled when users want to add a new address.

User Story #5

As a user, I would like to change my password.

Sub-task	Estimated Time	Responsible Person
Implement an API for changing password using the old password	1h	Xiaonan Shen
Create a user interface for changing password	5h	Shuqi Ma
Unit test - correct and incorrect old password	2h	Shuqi Ma

Completed:

In the “user profile” page, users could change their password with their old password. And we check the new password to make sure that it is not the same as the old password. And sending the information to the backend to check the old password in our database.

User Story #44

As a user, I would like to save my payment information.

Sub-task	Estimated Time	Responsible Person
Create ‘cards’ table in the database	1h	Xiaonan Shen
Setup PHP library for Stripe API	1h	Xiaonan Shen
Implement an API that returns all the payment methods of the current user	1h	Xiaonan Shen
Implement an API for adding payment methods	1h	Xiaonan Shen
Implement an API for editing payment methods	1h	Xiaonan Shen
Implement an API for deleting payment methods	1h	Xiaonan Shen
Setup Stripe.js on the web client	4h	Zequan Wu
Create a user interface for showing payment methods	4h	Zequan Wu
Create a user interface for adding payment methods	2h	Zequan Wu

Completed:

After implementing those, users can add a new credit card. The frontend will use Stripe API to verify the credit card information and receive a token if correct or error otherwise. The token is what actually sent to the backend. The backend will use the token to verify the credit card via Stripe API and retrieve basic information about the credit card, like last 4 digits, nickname, expiration month, expiration year and zip code.

User Story #6

As a user, I would like to change my email address.

Sub-task	Estimated Time	Responsible Person
Implement an API for changing the email address	1h	Xiaonan Shen
Create a user interface for changing the email address	4h	Shuqi Ma
Unit test - correct or incorrect verification code	2h	Shiwen Xu

Completed:

We add the button(showing current email) in the “user profile” for users to change their email. They will receive an email from us to the new email they want to change we will notify them when the change is successful.

User Story #7

As a user, I would like to have a general address that based on my current geolocation so that I don’t have to type in my address before I decided.

Sub-task	Estimated Time	Responsible Person
Create a functionality that requests current location from the browser	5h	Shiwen Xu
Create API call that requests a human-readable address from Google Maps API	5h	Shiwen Xu

Completed:

If users allowed, when they click the ”current location” button, we will get their current geolocation and converted address will show on the button. We could use this component in the future user story to help us get users’ address easier.

User Story #8

As a user, I would like to see the available restaurants, together with distance, estimated delivery time, order minimum, and delivery fee.

Sub-task	Estimated Time	Responsible Person
Add ‘restaurants’ table in the database	1h	Xiaonan Shen
Implement an API for list all nearby restaurants available at that time	1h	Xiaonan Shen
Create a user interface for view the available	5h	Zequan Wu

restaurants' information		
Unit test - different local time for available restaurants	2h	Shiwen Xu

Completed:

After we implemented this, users need to enter an address to see a list of restaurants with their basic information, like image, name, order minimum, delivery fee, estimated delivery time, category and distance. Also, users can press "get current position" button to auto-fill the address input.

User Story #9

As a user, I would like to search for specific restaurants by name.

Sub-task	Estimated Time	Responsible Person
Add search bar in user story #8 user interface	2h	Zequan Wu
Unit test - existing and non-existing restaurant names	2h	Zequan Wu

Completed:

After we implemented this, users can filter restaurants by entering the name. The list of restaurants will be updated every time the text in name input got changed, which is only used to filter restaurants in the frontend.

User Story #10

As a user, I would like to filter the restaurant list by category, distance, and estimated delivery time.

Sub-task	Estimated Time	Responsible Person
Implement the API for filtering restaurants by given constraints	1h	Xiaonan Shen

Completed:

By giving related URL parameters, the API will be able to return the filtered restaurant list. Filtering by categories, distance, estimated delivery time, order minimum, and delivery fee is supported. Multiple filters can be used at the same time.

User Story #13

As a user, I would like to sort the restaurant list by distance, estimated delivery time, and rating.

Sub-task	Estimated Time	Responsible Person
Implement the API for sorting restaurants by given constraints	1h	Xiaonan Shen
Create a user interface for sorting restaurants	2h	Zequan Wu

Completed:

After we implemented this, users can sort restaurants by different types such as distance, rating, price and etc. The list of restaurants will be updated every time the text in name input got changed, which is only used to sort restaurants in the frontend.

User Story #35

As an administrator, I would like to login to a dashboard.

Sub-task	Estimated Time	Responsible Person
Create ‘admin_users’ table in the database	1h	Xiaonan Shen
Create authentication guard for admin users in the Laravel backend	1h	Xiaonan Shen
Setup Voyager, a Laravel Admin Package	1h	Xiaonan Shen

Completed:

A command line tool called ‘artisan’ can be used to create admin accounts. Administrators can also create new admin accounts directly on the dashboard.

User Story #36

As an administrator, I would like to manage restaurants.

Sub-task	Estimated Time	Responsible Person
Add new BREAD (Browser, Read, Edit, Add, Delete) model for restaurants in the Voyager package	1h	Xiaonan Shen

Completed:

Administrators can manage restaurant categories and restaurants on the dashboard.

What did not go well?

User Story #3

As a user, I would like to reset my password if I forget it.

Create a user interface for resetting password on the web client	5h	Shuqi Ma

Not Completed:

We have implemented the user interface and APIs for resetting the password, but there are some problems causing crashes, which prevent the user from entering a new password on the web client.

User Story #4

As a user, I would like to save my address so that I don't have to type them in every time.

Create a user interface for editing addresses	2h	Shuqi Ma

Not Completed:

We already finished the UI of the edit saved address but there are some problems that making the user unable to edit the information in the text area, we will improve this part in the future sprint.

User Story #10

As a user, I would like to filter the restaurant list by category, distance, and estimated delivery time.

Create a user interface for view the filtered restaurant list returned by the server	4h	Zequan Wu

Not Completed:

We only implemented filtering restaurants by category and didn’t have enough time to finish the rest, since the API for getting the list of restaurants was updated just a few hours before the review and our previous design is to filter restaurants at frontend.

User Story #9

As a user, I would like to search for specific restaurants by name.

Add search bar in user story #8 user interface	2h	Zequan Wu

Not Completed:

This is basically completed, but there is a bug, which only recognizes the lowercase characters of user input. This can be fixed quickly.

User Story #44

As a user, I would like to save my payment information.

Create a user interface for editing payment methods	2h	Zequan Wu

Not Completed:

The user interface for editing payment methods hasn’t been implemented yet. After implementation, users will be able to edit the nickname, expiration month, expiration year and zip code.

How should we improve?

We successfully finished most of the tasks of Sprint 1. However, there are still plenty of bugs and problems.

In the first sprint, we spent a lot of time dealing with merge conflicts. It also caused a lot of bugs. In the next sprint, we should better assign the tasks and improve the structure of the code so that there will be fewer conflicts. Besides, we need to push and merge more frequently, so the conflicts can be detected earlier, and hopefully, easier to solve.

Another issue that needs to be addressed is that we need to spend more time writing automatic tests for our project, especially for the frontend. Such kind of unit or feature tests can prevent us from destroying the finished functionalities by the changes we made to accomplish other tasks. In the Sprint 1, some functionalities were working when we finished, but broken during the review.

Effective communication is also important. Firstly, we need to hold more meetings. All team members are busy and finding a meeting time can be difficult, but meetings are necessary. Besides, we should inform other team members when changes are made in APIs or shared components. It will prevent many unnecessary troubles.

Furthermore, we have to be more self-disciplined, which means that we have to actually spend time on the project bit by bit after finishing the sprint plan document. In the first sprint, we have not fully adapted to the model. We spend most of our time researching on many things rather than actually starting it. The side effect is that we had to catch up the rest of the user stories in the last few days. We could have more spare time to test corner cases to make our product more perfect.