Füdstops

Sprint 1 Planning

Young Yu, Aaron Kim, Veer Sahani, Reuben Varghese

1.0 Sprint Overview

In this sprint, we will create a working client-server relationship between the front and back ends. Following a working client-server relationship, we will create the login and authentication features which will allow the user to login. We will also establish a connection between the backend server and the databases we plan to use, and use the database to store the user data. All of the login and personal account features will have been established in this sprint. We will create the tabs on the bottom of the screen that the user can click on to navigate to different pages within the application.

The user will be able to handle the core features of the app minus the recommendation system after this sprint. The user should be able to select their dietary preferences and restrictions, view menu items and their details (e.g. nutrition facts), save menu items, rate menu items, and also see whether or not the item will be served on near future dates (as well as which dates).

Scrum Master: Young Yu
Meeting Plan: MWF @ 8 pm
Risk and Challenges:
Some potential risks and challenges in this sprint may arise from the fact that no one on the team has worked with the MERN stack. The biggest challenge for us this spring is connecting all of the components together. Once all of the components are connected together we will also have to get used to coding in languages like Node and React which we are not familiar with. We have the logic for the code down, but translating it into Node and React will be difficult for us. Another challenge that may arise is unit testing our code to ensure that our functionality is correct.

2.0 Current Sprint Detail

2.1 User Story #1

As a user, I would like to log in using my email or phone number

#	Description	Estimated Time (hrs)	Owner
1	Create backend service to allow users to students to register	3	Young
2	Create backend service to allow students to log in via phone number or email	3	Young
3	Create unit tests to ensure that logging in performs as expected	2	Veer
4	Create user interface to allow users to register into the application	3	Veer

Acceptance Criteria:

- Given that the backend service works correctly, users should be able to create an account via phone number or email, and be able to login
- Given that the user interface works correctly users should be able to see that they have registered for an account
- Given that the user interface works, after registering for an account the user should be able to login via phone number or email and see that they have successfully logged in

2.2 User Story #2

As a user, I would like to be able to reset my password associated with my account if I registered using email or phone number and forgot my password.

#	Description	Estimated Time (hrs)	Owner
1	Create backend service that will allow users to reset password	3	Young
2	Create front end service that will allow students to reset password	3	Young
3	Unit test code to ensure it works	2	Young

Acceptance Criteria:

- Given that the backend service works correctly, users should be able to reset the password and login with their new credentials
- Given that the user interface works correctly the user should see that they have reset their password successfully

- Given that the user interface works correctly, the user should see that they have logged in successfully even after resetting the password

2.3 User Story #3

As a user, I would like to be able to create a username associated with my account

#	Description	Estimated Time (hrs)	Owner
1	Create back end service to allow user to have a username	3	Young
2	Create front end component to allow users to enter their username via buttons	3	Young
3	Create unit tests to ensure that their username is associated with their phone number or email	2	Young

Acceptance Criteria:

- Given that the backend service works correctly, users should be able to create a username and be able to use that as their login credentials
- Given that the front end service works, user should visually see that they have created a username for their account
- User should be able to login from the front end with their newly created username

2.4 User Story #4

As a user, I would like to be able to create a profile picture

#	Description	Estimated Time (hrs)	Owner
1	Create UI Panel to display profile picture	2	Veer
2	Create buttons to allow user to add, change, and delete profile picture	2	Veer
3	Create logic that stores/deletes profile picture in/from database when user adds, changes, and deletes their profile picture	4	Veer
4	Create unit tests for adding, deleting, and changing profile pictures	2	Veer

Acceptance Criteria:

- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database is set up correctly), the user's profile picture should be correctly displayed.
- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database is set up correctly), when the user interact on the UI corresponding with adding/changing profile pictures, my profile picture should be properly updated in the database and I should be able to see the new profile picture on the UI.
- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database is set up correctly), when the user interacts with the UI corresponding with deleting my profile picture, my profile picture should be removed from the database and the user should have a default empty profile picture.
- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database is set up correctly), when the user submits a profile pic of a size too large, an error message should be displayed to the user.

2.5 User Story #5

As a user, I would like to be able to edit my personal information

#	Description	Estimated Time (hrs)	Owner
1	Create UI panel to display user's existing personal information	2	Veer
2	Create buttons and interactive UI for user to modify personal information	2	Veer
3	Create logic that will update the database when user modifies their personal information in the UI	5	Veer
4	Create unit tests for modifying personal information	2	Veer

Acceptance Criteria:

- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database is set up correctly), when the user interacts with the UI to modify their personal information, their personal information should be updated in the database.
- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database is set up correctly), when the user interacts with the UI to modify their personal information, the changes should be reflected in the UI.

- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database is set up correctly), when the user enters their user information in an invalid format, an error message should be shown to the user.

2.6 User Story #6

As a user, I would like to be able to access Füdstops on both mobile and desktop devices.

#	Description	Estimated Time (hrs)	Owner
1	Create logic that detects the device type and adjusts the UI (if necessary) for mobile devices	5	Veer
2	Create unit tests for detecting device type	2	Veer

Acceptance Criteria:

- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database is set up correctly), the application should work on mobile for the user and the UI should be adjusted if necessary to fit the mobile screen.
- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database, the application should work on the desktop for the user.
- Given that the frontend, backend, and database work correctly (and that the logic between the backend and database, any changes that the user makes through mobile should be reflected on the desktop UI as well and vice versa.

2.7 User Story #7

As a user, I would like there to be a help feature in order to report a problem.

#	Description	Estimated Time	Owner
1	Create buttons and a UI where the user can report a problem	2	Veer
2	Create logic in backend that stores the information that the user writes	5	Veer

3	Create unit tests to ensure that they can report a problem	2	Veer

Acceptance Criteria:

- Given that the backend works as expected the users report should be displayed
- Given that the backend works as expected, when I interact with the user interface my reported problem should be displayed
- Given that the backend works correctly I should be able to add multiple reports and should have this option on the user interface

2.8 User Story #8

As a user, I would like all my data associated with the app to be stored in my account that I login with

#	Description	Estimated Time	Owner
1	Create logic that updates the database whenever the user performs an action that requires updating the user's info in the database	2	Aaron
2	Establish link between database(s) on backend	4	Aaron
3	Create unit tests that ensure communication between the database and backend is working (add/update/delete operations)	2	Aaron

Acceptance Criteria:

- Given that the database is functioning as expected, all user associated data should be seen in the database
- Given that the database is functioning as expected, users should be able to add to their data, and the changes should be reflected in the database

- Given that the database is functioning as expected, users should be able to edit their data, and the changes should be reflected in the database
- Given that the database is functioning as expected, users should be able to delete their data, and the changes should be reflected in the database

2.9 User Story #10

As a user, I would like to be able to view what menu items are available at each dining court on a specific day during breakfast, lunch, and dinner hours.

#	Description	Estimated Time	Owner
1	Create buttons and UI so the user can click on any of the dining courts	2	Young
2	Create buttons and UI so that when the user clicks on the button to display information, then the information pops up	5	Young
3	Retrieve all of the dining court information via the Purdue dining menu API and store it into the database once per day	5	Young
4	Retrieve dining data from database and display it to user on the UI	2	Young
5	Create unit tests to verify communication between each component, and to ensure the dining API is returning the correct data.	2	Young

Acceptance Criteria:

- Given that the frontend, backend, and database all function properly, users should be able to view the dining court hours for each specific day
- Given that frontend works properly the user should be able to scroll on the page if too much information is displayed
- Given that the database works as expected, the hours of the dining court should be stored in the database so that the information can be retrieved at anytime

2.10 User Story #11

As a user, I would like to have access to the nutrition facts for each food item on the menu.

#	Description	Estimated Time	Owner
1	Create necessary buttons and UI interactivity so users can select specific food items	5	Aaron
2	Implement logic that parses dining data from Purdue dining API and stores it in database	5	Aaron
2	Create logic that retrieves relevant nutrition information from database and displays it to user on the UI	5	Aaron
3	Create unit tests to ensure the nutrition facts are all accurate	2	Aaron

Acceptance Criteria:

- Given that the frontend, backend, and database all function properly, users should be able to view the nutrition information for any food item at all times of the day
- Given that frontend works properly the user should be able to scroll the page if there is too much information displayed for the screen
- Given that the database and web scraping portion of the backend works as expected, the proper nutrition facts should be stored in the database for all corresponding food items

2.11 User Story #13

As a user, I would like to see whether a food item will be served on future dates.

#	Description	Estimated Time	Owner
1	Create necessary UI elements to see serving dates	2	Aaron
2	Retrieve relevant serving dates from database of any entries that match the menu item, and display them to the user on the UI	5	Aaron
3	Create unit tests to ensure all future serving dates are accurate	2	Aaron

Acceptance Criteria:

- Given that the frontend, backend, and database all function properly, users should be able to view when food items will be served on future dates
- Given that the database works as expected, the dates when certain food items are planned to be served should be correct and up to date
- Given that the frontend, backend, and database all function properly, the database should contain menus in the future as far forward into the future as the API provides, which should be at least 5 days in advance.

2.12 User Story #23

As a user, I would like to be able to rate food items I have consumed.

#	Description	Estimated Time	Owner
1	Create buttons for rating items and UI elements for viewing ratings	2	Reuben
2	Implement logic for updating ratings in	5	Reuben

	database and viewing updates in real time in the UI		
3	Create unit tests to ensure ratings and updates to ratings are functioning properly and in real time	2	Reuben

Acceptance Criteria:

- Given that the frontend, backend, and database all function properly, users should be able to rate food items positively or negatively and view the average rating of a food item
- Given that the backend works as expected, the ratings of food items should be able to be updated and changed in real time
- Given that the database works as expected, the ratings of food items should be stored and accessible for retrieval at any time of the day

2.13 User Story #25

As a user, I would like to be able to save menu items I enjoyed.

#	Description	Estimated Time	Owner
1	Create backend service that will send saved menu items to be recorded in the database for that user's account	3	Reuben
2	Create frontend buttons to save menu items and display an icon to user that indicates the item has been saved	3	Reuben
3	Create logic that saves menu items to the database when the user saves them in the UI	5	Aaron
4	Create unit tests to	3	Aaron

	check if saved item appears in user's account in database, check if saved indication displays, check if backend call sends correct item		

Acceptance Criteria:

- Given that the frontend, backend, and database all function properly, users should be able to save menu items that they enjoyed and it should be clear to them (in the form of an icon) that their item was successfully saved.
- Given that the backend works as expected, the user's account in the database should now contain an updated "saved list" so that the front end will properly update to indicate the saved item.
- Given that the database works as expected, the user's saved menu item list will be correct with whichever items the user has saved.

2.14 User Story #27

As a user, I would like to be able to select my dietary preferences.

#	Description	Estimated Time	Owner
1	Create backend service that updates database with user's dietary preferences	2	Reuben
2	Create frontend component that allows user to select dietary preference (options such as vegan/pescetarian)	2	Reuben
3	Create logic that updates database when user enters their dietary preferences in the UI	5	Reuben
4	Create unit tests that will check whether	2	Reuben

	the backend call properly updates the database and that the frontend component functions as intended.		

Acceptance Criteria:

- Given that the frontend, backend, and database all function properly, users should be able select their dietary preferences for their profile and it should be clear to them (in the form of checkmarks/selections on the preferences) that their profile has those dietary preferences now.
- Given that the backend works as expected, the user's account in the database should now contain an updated "dietary preferences list" so that the front end will properly update to indicate them.
- Given that the database works as expected, the user's dietary preference list will be correct with whichever preferences the user has selected.

2.15 User Story #28

As a user, I would like to be able to select my dietary restrictions.

#	Description	Estimated Time	Owner
1	Create backend service that updates database with user's dietary restrictions	2	Reuben
2	Create frontend component that allows user to select dietary restrictions (options such as peanut allergy/lactose intolerance)	2	Reuben
3	Create logic that updates database when user enters their dietary restrictions in the UI	5	Reuben
4	Create unit tests that will check whether	2	Reuben

	the backend call properly updates the database and that the frontend component functions as intended.		

Acceptance Criteria:

- Given that the frontend, backend, and database all function properly, users should be able input their dietary restrictions for their profile and it should be clear to them (in the form of checkmarks/selections on the restrictions) that their profile has those dietary restrictions now.
- Given that the backend works as expected, the user's account in the database should now contain an updated "dietary restrictions list" so that the front end will properly update to indicate them.
- Given that the database works as expected, the user's dietary restrictions list will be up to date with whichever restrictions the user has selected.

3.0 Remaining Backlog

3.1 Functional Requirements

3.1.1 Login, User Profile, General

1. ~~As a user, I would like to log in using my email or phone number~~
2. ~~As a user, I would like to be able to reset my password associated with my account if I registered using email or phone number and forgot my password.~~
3. ~~As a user, I would like to be able to create a username associated with my account~~
4. ~~As a user, I would like to be able to create a profile picture~~
5. ~~As a user, I would like to be able to edit my personal information~~
6. ~~As a user, I would like to be able to access Füdstops on both mobile and desktop devices.~~
7. ~~As a user, I would like there to be a help feature in order to report a problem.~~
8. ~~As a user, I would like all my data associated with the app to be stored in my account that I login with~~

3.1.2 Viewing Active Hours of Dining Courts

9. As a user, I would like to be able to view how busy a dining court is at all hours during the day.

3.1.3 Viewing Menu Items

10. ~~As a user, I would like to be able to view what menu items are available at each dining court on a specific day during breakfast, lunch, and dinner hours.~~

11. ~~As a user, I would like to have access to the nutrition facts for each food item on the menu.~~
12. As a user, I would like to be able to view popular menu items at each dining court (algorithm to calculate this TBD).
13. ~~As a user, I would like to see whether a food item will be served on future dates.~~

3.1.4 Filtering Food Spots

14. As a user, I would like to be able to filter food spots by only those that align with my dietary preferences/restrictions (e.g. vegetarian, soy-free, gluten-free, etc).
15. As a user, I would like to be able to filter dining courts by those that align with dietary preferences/restrictions that I select in a filter menu.
16. As a user, I would like to be able to order dining courts by their busyness at the current hour.
17. As a user, I would like to be able to order dining courts by their name in alphabetical order.

3.1.5 Filtering Menu Items

18. As a user, I would like to be able to filter menu items by only those that align with my dietary preferences/restrictions (e.g. vegetarian, soy-free, gluten-free, etc).
19. As a user, I would like to be able to filter menu items by those that align with particular dietary preferences/restrictions that I select in a filter menu.
20. As a user, I would like to be able to order menu items by their name in alphabetical order.
21. As a user, I would like to be able to filter menu items by relevant aggregation categories (e.g. low-calorie, low-fat, dessert, etc).
22. As a user, I would like to be able to filter menu items by their cuisine.

3.1.6 Rating Menu Items

23. ~~As a user, I would like to be able to rate food items I have consumed.~~
24. As a user, I would like to vote for commonly clicked on items in the menu so that they appear in dining courts.

3.1.7 Saving Menu Items

25. ~~As a user, I would like to be able to save menu items I enjoyed.~~
26. As a user, I would like to be able to find recommendations for food similar to menu items I saved.

3.1.8 Dietary Preferences/Restrictions

27. ~~As a user, I would like to be able to select my dietary preferences.~~
28. ~~As a user, I would like to be able to select my dietary restrictions.~~

3.1.9 Receiving Recommendations

29. As a user, I would like to receive recommendations for dining courts based on my food preferences/restrictions.
30. As a user, I would like to receive recommendations for dining courts based off of my distance for each dining court.
31. As a user, I would like to have a tab I can click on to generate recommendations on where I should eat for my next meal and the item(s) I should eat from that spot (breakfast, dinner, lunch)
32. As a user, I would like to have a tab I can click on to generate recommendations on where I should eat for my next meal (breakfast, dinner, lunch)
33. As a user, I would like to be able to sort my recommendations by rating, relevance, and distance.

3.1.10 Viewing Food Spot Information

34. As a user, I would like to be able to view the opening and closing hours of a dining court
35. As a user, I would like to be able to view the contact info of a dining court
36. As a user, I would like to be able to get the directions/address to a dining court
37. As a user, I would like to view when breakfast, lunch, and dinner hours are at each dining court

3.1.11 Searching for menu items and food spots

38. As a user, I would like to be able to search for a term, and have all food items matching that term be displayed, as well as the dining court its being served in, the time of day, and on which day

3.1.12 Notifications

39. As a user, I would like to receive text messages (if I opt in) when menu items I enjoyed are offered soon in a dining court
40. As a user, I would like to receive text messages (if I opt in) of recommendations of menu items that I may enjoy based on what I like and my dietary preferences/restrictions.

3.1.13 Retail (if time allows)

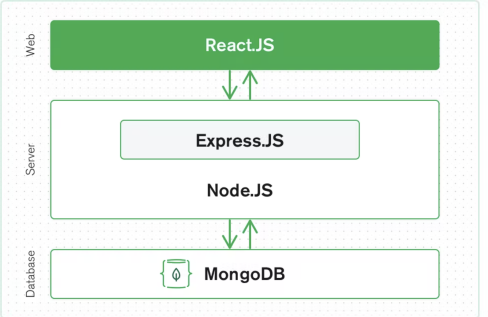
41. As a user, I would like to be able to have a separate tab for retail restaurants
42. As a user, I would like to be able to order retail restaurants by their distances from me
43. As a user, I would like to be able to filter retail restaurants by their busyness at the current hour
44. As a user, I would like to be able to view how busy a retail restaurant is at all hours during the day.
45. As a user, I would like to be able to filter retail restaurants by whether they contain foods following my dietary preferences/restrictions

46. As a user, I would like to see an about section for each retail restaurant
47. As a user, I would like to get directions or the address to the restaurant
48. As a user, I would like to see the hours of the restaurant
49. As a user, I would like to be able to view the menu of the retail restaurant
50. As a user, I would like to have access to the nutrition facts for each food item on the menu.
51. As a user, I would like to be able to see when meal swipes are accepted at the retail restaurant during which hours on which days
52. As a user, I would like to be able to view popular menu items at each retail restaurant (algorithm to calculate this TBD).
53. As a user, I would like to be able to filter retail restaurants by only those that align with my dietary preferences/restrictions (e.g. vegetarian, soy-free, gluten-free, etc).
54. As a user, I would like to be able to retail restaurants by those that align with dietary preferences/restrictions that I select in a filter menu.
55. As a user, I would like to be able to order retail restaurants by their name in alphabetical order.
56. As a user, I would like to be able to filter retail restaurants menu items by only those that align with my dietary preferences/restrictions (e.g. vegetarian, soy-free, gluten-free, etc).
57. As a user, I would like to be able to filter retail restaurants menu items by those that align with dietary preferences/restrictions that I select in a filter menu.
58. As a user, I would like to be able to order retail restaurants menu items by their name in alphabetical order.
59. As a user, I would like to be able to filter retail restaurants menu items by relevant aggregation categories (e.g. low-calorie, low-fat, dessert, etc).
60. As a user, I would like to be able to rate retail restaurants food items I have consumed.
61. As a user, I would like to be able to save retail restaurants menu items I enjoyed.
62. As a user, I would like to be able to find recommendations for food similar to retail restaurants menu items I saved.
63. As a user, I would like to receive recommendations for retail restaurants based on my food preferences/restrictions.
64. As a user, I would like to receive recommendations for retail restaurants based off of my distance for each establishment.
65. As a user, I would like to be able to view the contact info of a retail restaurants
66. As a user, I would like to be able to search for a term, and have all food items matching that term be displayed, as well as the retail restaurant which is serving that item, and its hours on the current day
67. As a user, I would like to be able to filter retail restaurants by their cuisine.

3.2 Non-Functional Requirements

3.2.1 Architecture

We plan on completely separating the frontend and backend of our application. This is in the interest of intersystem independence as well as for the ease of developing the software.
We will be using the MERN stack (MongoDB, Express, React, Node.JS) to create our web application, mainly with JavaScript and JSON. The frontend will be made using React, and will receive dining data from the backend with API calls. Also, we will use ChakraUI for access to components to make a clean and well-designed web-app.



Frontend - React & ChakraUI , Web Server - Express & Node, Database - MongoDB

1. As a user, I expect to use a web app where my login information/preferences are saved (database).
2. As a user, I expect the website to be straightforward to use and clean to view (React framework + Chakra UI components).

3.2.2 Performance

A key component of this application is that it should be usable on-the-go so that users may quickly receive information or recommendations for dining establishments. As such, it is imperative that the application be as responsive as possible. The application should be able to handle many different requests at once in order to handle spikes in usage during peak dining times.

1. As a user, I expect a response time within 400 milliseconds during idle dining periods.
2. As a user, I expect the web app to have a response time within 600 milliseconds during peak dining periods (12pm - 1pm, 6pm - 7pm).
3. As a user, I expect the web app to be accessible 24/7, with a max down time of 3d 15h 39m 29s (99% SLA).

3.2.3 Security

Security is essential in the development of our software because we will be handling potentially confidential/sensitive user information. This includes, but is not limited to, their authentication information and location. Our software will also be utilizing databases for the storage of multiple types of data. So, preventative measures must be taken to prevent any such exploits, which include maintaining a keen awareness of the security of our app when developing, hiding the

visibility of database errors on our product, and sanitizing all user input, including login forms, by removing any potentially malicious code elements such as single quotes.

3.2.4 Usability

Because this application is intended for general use by the Purdue student body, it is necessary that it be as accessible for people of varying technical familiarity and computer literacy. The interface must be intuitive and easy to navigate, with important information such as the most convenient or highest-recommended location being easy to spot.

3.2.5 Scalability

Our software will be extendable to different types of food service locations, not just Purdue University Dining Courts but to other universities if they allow the app to be used for and access their residential dining court information and menus. The software will also be applicable to retail locations, made possible by Google's Place API (tentative).

3.2.6 Hosting & Deployment

For demonstration purposes we intend to run a local server. MongoDB Atlas will allow us to host MongoDB for our application. The React frontend will be hosted using Netlify and the Node backend will be hosted using Heroku, based on the following reference material: https://niruhan.medium.com/deploying-mern-fullstack-application-on-the-web-for-free-with-netlify-and-heroku-87d888012635. If we deploy the application, we will look into using AWS Web Servers for hosting the webapp for public use.