FUN-H FINAL REPORT

Dartagnan Birnie, Anthony Santos, Austin Snow, Jason MacMillan

DEPLOYED URL: http://20.241.139.241:5000/

FEATURES:

Our first highlighted feature is users/authentication. We made it possible to create a user account by entering a name, email, and password through custom authentication. Some JavaScript was also used to make this possible.

Our second highlighted feature is the use of update and delete operations. Users can create an account and add FUN-H Plates to their account. These plates can be edited/updated and also be deleted by the user. The delete operation is also used when deleting an account.

We also use CSS to have a clean UI.

PRODUCT / PROJECT DESCRIPTION:

FUN-H will allow students to pre-plan a healthy meal in the sense of a user selected list which will save crucial time in a college student's day. A digital list of food will be produced called a "FUN-H Plate". This "FUN-H Plate", or meal, will follow the recommended eating habits of the *Wildcat Plate* standards (One serving of each… *Fruits, Whole Grains, Vegetables, Lean Protein)*. Both dining halls, Philbrook and Holloway Commons, will be included in the FUN-H program, with menus being updated in respect to the data provided by the kitchen team.

REPORT / PRESENTATION

INTRODUCTION

The FUN-H team wanted to base our product around a problem and 3 statements which could be achieved to fulfill the purpose of our product. The idea for this project came from each member's shared experiences with the UNH dining hall system, and how the inefficiencies of food selection and dining hall app deficiencies affect our diets and busy schedules. Without having parental figures to plan out sufficient meals anymore, it becomes easy to fall into unhealthy habits without the proper planning skills having yet to be established. FUN-H set out to create a dietary planning service to break these unhealthy habits upon the following goals and values.

MOV:

Make meal planning at the UNH dining halls more efficient and healthy by having 50 “FUN-H” plates, following the Wildcat Plate standard, created by May 5th, 2023.

MISSION STATEMENT:

There are different menus at each dining hall, which could make selecting certain options confusing for users. To counteract this, users will select their preferred dining hall first, and they will be prompted with healthy menu options from the hall they selected. Another issue is item location, users might not know where to find certain selections. To get around this, information on where to find all of the sections will be provided (main line, etc.).

VISION:

With FUN-H, the project team aims to create a healthier campus at UNH, while also making it easier for students and faculty to plan and find food options at both dining halls (HoCo, Philly). Through FUN-H can healthy meals be planned in a straightforward manner online before arriving at a dining hall.

PROBLEM:

New students or students with certain diets/dietary restrictions may have a difficult time finding and locating meals at the UNH Dining Halls.

REQUIREMENTS:

Before beginning development on our service, first understanding the target audience and the features in which they would desire became crucial. To achieve this, the FUN-H team created an extensive list of user stories, targeting the community members from UNH, young to old. We believed we were able to grasp the desires of all in terms of features, further than the needs we saw with ourselves. Through these same user stories, it became more obvious in how the service needed to be presented in terms of UI and UX and the process of creating a FUN-H plate. Here you can find some of our target audience's \ user stories in which much of the creative process stemmed from.

USER STORIES:

"As a [role], I want to [goal], so that [benefit]"

As a member of the UNH weightlifting club, I want to save my previous plates so that I can plan them out ahead of time.

As a professor, I want to be able to use the web interface easily so that I can find the food I want quickly.

As a senior citizen of Durham, I would like clear instructions when using the app so I don't get confused and spend too much time using the app.

As a casual student, I would like to see which dining hall has better options so that I can choose accordingly.

As a new member of the UNH community, I want to know where the dining halls are located so I don't have to spend time finding them.

EXPANDED USER STORIES:

Persona:
Joe Schmo - Incoming Freshman at UNH

User Story:
As an incoming freshman at UNH, I want to be able to understand the variety and nutritional value of food offered at the UNH dining halls so I can pick out healthy meals without the influence of my parents.

Persona:
John Doe - Student at UNH CEPS

User Story:
As a UNH student, I want to be able to exclude specific food groups when creating my Wildcat Plate so I don't accidentally consume something I shouldn't.

Persona:
Dr. Steve - An older Professor at UNH

User Story:
As an older professor at UNH who has never used the dining halls before, I want to be able to plan out my meals to lessen the time on my feet with a friendly UI experience while also meeting my dietary standards.

QUALITY ATTRIBUTES:

Usability
Clean and easy to use UI and UX. We added a lot of CSS to our website and made sure it was simple and easy for users to navigate the website.

Security
Creates accounts using authentication. Passwords are encrypted on the backend. When trying to log into an account, the inputted password is encrypted and checked with the encrypted associated with the email in our MongoDB database. If either the email or password are incorrect the same error message is prompted.

Reliability
System has error checks in place to make sure system performs consistently and
does its intended function.

Ex: Users can't go to the dashboard page to view their account and plates if they aren't logged in.
Ex: Users can't create an account or log in to an account if they are already signed into an account.
Ex: If users try to make a plate from selected menu items without

being logged into an account, they will be redirected to a page to create an account or log into one.

Maintainability

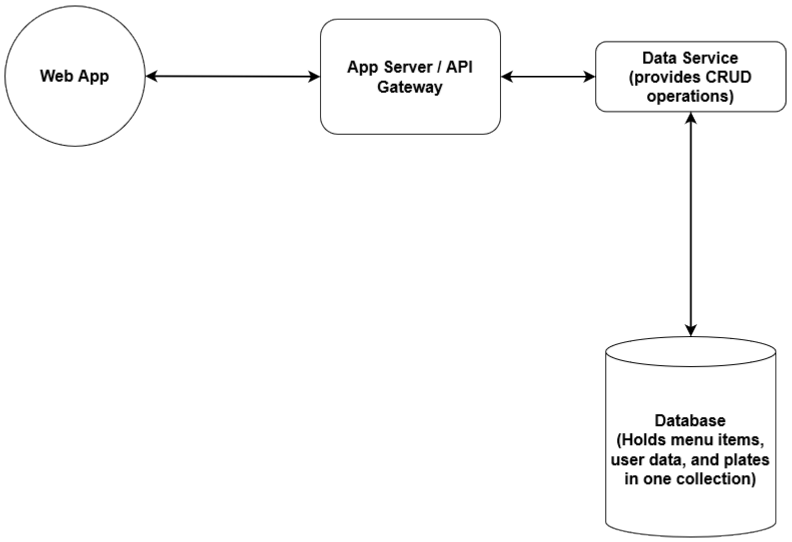
Can easily update system overtime.

Ex: Central CSS file and Central html file for the navigation bar so they don't have to be updated for every separate file

DESIGNS:

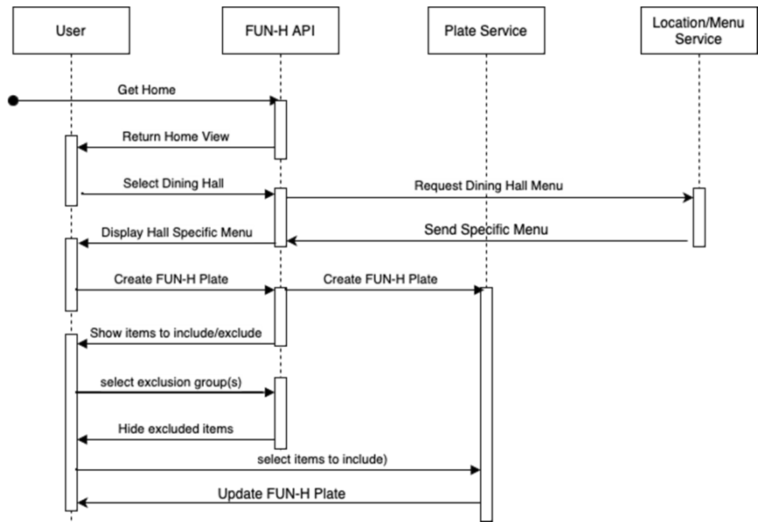
Now having written our desired and required features into text, finding the connections and how each feature can become intertwined becomes the next element. Through this, two diagrams (Architectural / Sequential) were created which showcase how each separate element, such as the WebApp to the database, interacts with one another. Our original sequential diagram sets a roadmap for how we originally thought how each of the features should play-out in order, and how the average user would interact with the service. Our API would end in a state much more intricate than first thought necessary much to the result of a smooth UX and security options.

ARCHITECTURAL DIAGRAM:


Web App
App Server / API Gateway
Data Service (provides CRUD operations)
Database (Holds menu items, user data, and plates in one collection)

Above is our product's architectural diagram. Our users interact with the product through our web app interface at http://20.241.139.241:5000/. When the user presses any buttons or submits any forms, the web app will interact with our API which may call upon one of our Azure CRUD operations (Create, Read, Edit, Delete) that interacts with our Mongo Database. Our Database has only a single collection, but holds all of the current and previous menu items with locations, dates, and meal types, as well as all user data, which contains a name, email, encrypted password, and created plates with pertinent information.

SEQUENTIAL DIAGRAM / API SPECIFICATIONS:



Above is our original sequential diagram that we created near the beginning of the semester. We ended up adding many routes in our app.py file making our API a lot more intricate than we originally thought. Instead of making a new sequential diagram for all used routes, we decided to go into short detail about what each route does. The routes will be listed by the name of the functions they are associated with.

USER AUTHENTICATION SERVICE:

This service manages user authentication and user-related functionality. This involves tasks for user creation, authentication, session management, and account deletion. Functions do use CRUD operations and access the database. All functions here primarily interact with the data service, except for createAccount which is web-focused.

signup:	This triggers a function that makes a user account, and adds it to the database. The POST method is used to send user details, specifically a name, email, and password. A session is also created for the user. If this process fails, an error message is prompted.
signout:	This triggers a function that would sign out the current user. This works by clearing the current session. Function will only work if user is signed in. User is redirected to the home page.
login:	This triggers a function that would log a user in. This uses the POST method to take login credentials. The user is searched in the database by email, and then the password is verified. A session is created for the user. An error is prompted if login credentials are wrong or if something else went wrong.
createAccount:	Renders “createAccount.html”. This is the page where users can create an account and login to an account. Users can input an email and password to log in to an account or create an account by inputting a name, email, and password. Both the signup function and login function are used in this page. A user can not be signed into an account when accessing this page.
removeAccount:	This triggers a function that deletes a user’s account. Deletes the account in the current session data from the MongoDB database. If the account was removed successfully, the user is redirected to the home page. If the account was not able to be deleted, the user is redirected to the dashboard page.

FILE SERVING SERVICE:

Serves static files, like images, CSS files, and JavaScript files, to the client. This ensures that necessary files are sent to the client when requested so that web pages are displayed properly. All functions here are web-focused.

serve_images:	This is used to send the image (png) file for the logo of FUN-H.

serve_css:	This is used to send the css file used for styling across our website. The route is accessed for all html pages.
serve_jquery:	This is used to send a js file that includes the jQuery framework which is then utilized in the scripts.js file. This script is only referenced/used in createAccount.html.
serve_scripts:	This is used to send a js file to act as a script in the createAccount.html. The scripts file is used to handle form submission when creating an account, sending information through the signup route.

STATIC PAGE SERVICE:

Renders static HTML pages to the client. These pages display static information aand provides an entry point to the application. Functions here are web-focused.

index:	Renders “index.html”. This acts as our home page.
contact:	Renders “contact.html”. This page shows contact info for the team behind FUN-H.

MENU SERVICE:

This service is responsible for managing menus and displaying menu information to users. Users also have the ability to select items off a menu through using the date function. These functions use CRUD operations to access the database. Functions here retrieve menu information from the database through the data service, and then display that information in their respective web pages.

phillyMenu:	Renders “phillyMenu.html”. This gathers all menu items from Philbrook Dining Hall for the current date, and has it sent to the webpage to be displayed.
hocoMenu:	Renders “hocoMenu.html”. This gathers all menu items from Holloway Commons Dining Hall for the current date, and has it sent to the webpage to be displayed.

date: Uses the GET method to render “date.html”. A user can submit a date on the webpage and the date is retrieved in this function by using the POST method. Menu items for both Philbrook Dining Hall and Holloway Commons Dining Hall are collected for that date and sorted. “date.html” is then rendered and displays all menu items for that date.

USER PLATE / ACCOUNT SERVICE:

Manages plates of a user and other account-related functionality. This service provides functionality to users to view their account information, create plates, delete plates, and edit plates. The dashboard is a central hub to interact with a user’s account and to manage plates. The dashboard function is web-focused. Both createplate and deleteplate primarily interact with the dataservice. edit_plate retrieves and updates information about a plate from a user by using the data service and also displays a web page where a user can update a plate.

dashboard: Renders “userplates.html”. This acts as a user dashboard and displays information about the user’s account. All the user’s plates are also displayed here. A user must be logged into an account to access this page.

createplate: This takes form data in the form of a list of menu items selected in date.html. The items are made into a FUN-H plate and added to the user’s account. The user is then redirected to the dashboard page. This uses the POST method. The user must also be logged in.

deleteplate: This takes the requested plate from a user, removes it from their account, and deletes it from the database. The plate is an argument/query parameter following the route/URL path for deleteplate. The plate is retrieved by using request.args.get(). The user must be logged in to.

edit_plate: The requested plate will be retrieved from the user’s session and “editPlate.html” will be rendered. The plate is an argument/query parameter following the route/URL path for edit_plate. The plate is retrieved by using request.args.get(). If the plate was not retrieved the user is sent back to the dashboard page. The plate information will be filled in on a form on the webpage so the user can make changes to it. When using the POST method, the plate will be updated in the database with the new information that the user inputted into the form. The user is then redirected to the dashboard page.

BRANDING:

FUN-H Logo Above

FUN-H hopes to push across the message that UNH your health is just as important as academics.

Through our branding, we incorporate the widely-known UNH main logo, with the 4 colors representing the Wildcat Plate food groups. Listed below is the main color scheme for the FUN-H brand, partnered with the respective HEX codes.

Fruits	Grains	Vegetables	Proteins	UNH Blue
#d71d25	#e26f1f	#72b744	#5c4690	#003591

Slogan:

"Find a healthier self with FUN-H"

<u>DEVELOPMENT & TESTING:</u>

Testing for the "Create a WildcatPlate" process was mostly internalized at this point in the service. We believe the product is in a state of functionality for user testing, having incorporated security services and the main purpose. Plates can be successfully created, edited, deleted and viewed by other users via the FUN-H platform. While our functionality seems to work without any glaring bugs and misdirections, incorporating actual user testing with real users in the next phase would allow for improvements and suggestions in which the original team may not have thought about. In order to receive these comments and recommendations, a “Contact Us” page has been added to the WebApp, linked with Email links to each individual member.

TESTING:

Feature Testing / Interaction Testing

Make sure the feature is implemented as expected. Interaction tests

Unit Testing

Test individual components. Usually done on CRUD operations in Data Service

User Testing

Test usefulness & usability. We would mimic users when using our webapp

System Testing

Test the entire system. A lot of tedious manual testing and making sure web app is working properly through the eyes of the user

Most of our testing was conducted by the developer team themselves.

PHASE 5:

Goals / Status

Created UNH Dining Hall Web Scraper based on date and location to fetch menu items

Prioritized and completed implementing update records, delete records, and user authentication

DEMONSTRATION:

First, go to [http://20.241.139.241:5000/]

CONCLUSIONS & FUTURE WORK:

While the team is extremely proud of all efforts put towards producing and developing the FUN-H service, there are still plenty of features which we initially hoped to implement that became ambitious and difficult tasks as the project went on. It wasn't until phase 5 where the team actually went about gathering plate information to store in the database, where the task of grabbing nutritional content of foods put a stop to the feature's development. The nutritional content associated with various foods from the target dining hall's online menu was inaccurate to an atrocious degree as well as difficult to web scrape. We did not meet our end goals as we had hoped since we did not necessarily gather any users to test and mainly tested the product ourselves, particularly due to time constraints. Throughout the development cycle, the group learned a few key lessons when it came to working as a team, like communication and planning skills, as well as balancing workload and documentation skills.

FUTURE WORK / EVOLUTION:

FUN-H is still in the early days of development and aspiration, we aim to implement the following:

- Allergen information
- Calorie, Protein, Carb, & Fat values per item
- Location of foods specific to dining halls
- Dining hall speciality days/events

End Goal:

- Receive feedback from 5 users related to the performance of the FUN-H application
- Have 50 Wildcat Plates created on FUN-H by May 5th, 2023.
- See a 10% returning user rate on the FUN-H platform.

REFLECTIONS / LESSONS LEARNED:

Front End:

- Branding and creating an "image" for a product
- Graphic Design / GIMP
- Understanding the target audience
- HTML / Website Engineering
- CSS
- JavaScript

Back End:

- Flask implementation
- MongoDB and Pymongo
- Docker
- Azure
- Python

Team Process:

The team for FUN-H had as smooth of a development process as could've hoped for. Every team member was committed to putting their effort into our final product, and each of us found a way to work on the features that fit their strengths most. On Mondays and Wednesdays all members were present to work on the project together, and if someone wasn't able to meet for the allotted time, they made sure the other group members would be alerted of it. We frequently found ourselves using the Discord and Snapchat group chat to share plans, progress and meeting times, even outside of the scheduled class times. The main takeaway from the team process was how important communication, frequent progress-checks and documentation became. Anthony, who hosted our product / database, was also very helpful in keeping the most recent version of our web service available through the IP link and GitLab Repo. From the very beginning, this group was able to talk its way into an end goal that fit the expectations of all, while outlining what would be expected by each member in terms of workload.

Into bullet points:

- Communication skills
- Documentation
- Frequently sharing progress with other group members
- Planning / Scheduling

MEETING MINUTES: https://docs.google.com/document/d/1TVZ5yTxdf-vEatMeej_cMihittJbXBFXKYr0y0HejuI/edit

GITLAB ISSUE BOARD:
https://gitlab.cs.unh.edu/cs518/spring2023/lab01/group-b/main/-/boards

GITLAB MAIN BRANCH:
https://gitlab.cs.unh.edu/cs518/spring2023/lab01/group-b/main