COMP 1004

Introduction

For this project I want to create something that can assist users in managing their time. People can struggle to keep on top of managing their time and this can lead to procrastination and issues with a healthy work-life balance. I want to create something that will take information from the user such as how much time they spend on activities and other lifestyle choices and let them know if they are spending too much or too little time. This report will cover in detail about my project and explain to the user why I chose this project and what I have added to it to make it a user friendly, useful tool.

SDLC

Stages of SDLC:

- Project planning
This stage allows companies to evaluate how much the project is going to cost and what resources they are going to need to complete the project. They can also figure out what the requirements of the software is

- Requirements Analysis
This stage is all about completely analyzing the requirements. All requirements should be analyzed in detail, this can be done by teams asking the clients for specific details about what they want done. This stage is very important as it gives the team a chance to evaluate if the requirements are reasonable/ possible and gives them a clear vision for what to do.

- Design
In this stage the developer will create a design plan for the project and evaluate if it meets all the needs specified. They will also plan out the tools needed to create the project such as what language to use. Stakeholders will also be consulted at this point for their opinion/input on the proposed design.

- Implementation
In this stage the program is coded. The work can be split up into different blocks for different developers to work on at the same time, increasing productivity. Some clients

may request additional features at this point which the developers will have to consider if it is plausible.

- Testing
At this stage the software will be tested for any bugs. Testers will scrutinize the software and try different methods to see if any bugs can be picked out. This could include using valid inputs and invalid inputs. The testing phase will go on and the software will be edited and re-tested until there are no bugs found and meets the requirements laid out..

- Deployment
At this stage the software is deployed for users to use. The size of the project determines the deployment's complexity. Users are provided with documentation on how to use the software. Testing is also run again to check for any issues involved with its release.

- Maintenance
After the project is released users will find issues or updates will be needed. The project will be constantly maintained and regular updates can be made to fix bugs post release.

How I used agile
At the beginning of my project I outlined what I intended to create and planned a rough idea of what I want the final project to look like. I noted down the requirements for my project and planned my sprints around what requirements were left to complete.

I used an agile method for the development of the software. In this approach I broke my development down into 2 week sprints. I planned an initial sprint which outlined what needs to be done within those 2 weeks. At the end of every sprint I planned the next step in development. I used the office365 planner app where I had a backlog to store what I was currently working on. Taking this approach allowed me to be adaptive with my project. I was able to account and adapt to any issues I faced along the way. In total it took me 6 sprints to complete my project.

There was a delay to my planned sprint deadlines as I faced issues with JSON which took me around 3 weeks to figure out. Overall this did not have a massive impact as I still had months to finish my project at this point and was able to carry on with the next few sprints without facing issues.

Project Vision

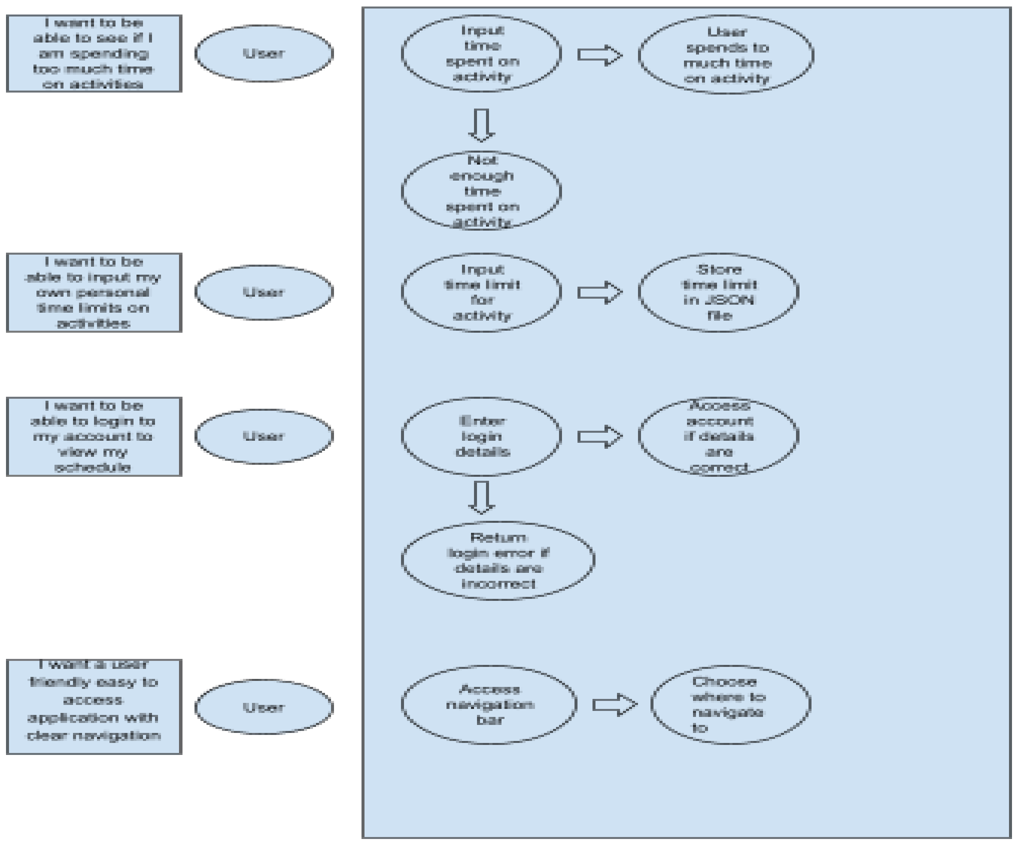
My vision for this project is to create a schedule application that can help users organize their daily tasks. I want users to be able to input activities they want to do and how much time they will spend on those activities. The application should then tell them if they are spending too long on those activities, for example if they input something like ‘gaming’ and say they spend 6 hours on it the application will highlight it red to let them know that they are spending too long on that activity. This should hopefully promote a more healthy balance for users. I will use google and research what is considered too much time on activities and use this for suggestions on my application. I also want to include an account system for my application where users can sign in and have access to their own stored schedules, accounts should not be able to access schedules linked to other accounts. I want to also encrypt the passwords so that the users accounts are kept more secure.

Requirements

1. It will be a single page application that users can interact with.
2. The application must be user friendly and easy for users to navigate.
3. Must input into and output from a json file.
4. A clear navigation bar for users to navigate the application.
5. A user should be able enter information about their daily routine and have a recommended routine change that can be stored as a schedule for them.
6. Users should be able to create an account to store their schedule.
7. Details users provide should be stored safely and not be at risk of being lost, stolen, etc.
8. Use CSS to style my page and make it appealing to users.
9. Use javascript to add interactivity to my application.
10. Provide a guide for how the application works.

UML Diagrams

User Stories


I want to be able to see if I am spending too much time on activities
User
Input time spent on activity
User spends to much time on activity
Not enough time spent on activity
I want to be able to input my own personal time limits on activities
User
Input time limit for activity
Store time limit in JSON file
I want to be able to login to my account to view my schedule
User
Enter login details
Access account if details are correct
Return login error if details are incorrect
I want a user friendly easy to access application with clear navigation
User
Access navigation bar
Choose where to navigate to

Use case description

Name	Update a schedule
Description	User can access existing schedules and make changes to the existing schedule and then store the result
Pre condition	Schedule must be fully complete
Post condition	Full updated schedule will be stored in a JSON file that can be accessed whenever the user wants.
Error situations	Schedule is not complete
System state in event of an error	Ask users to ensure they have finished the schedule and left no incomplete sections.
Actors	User

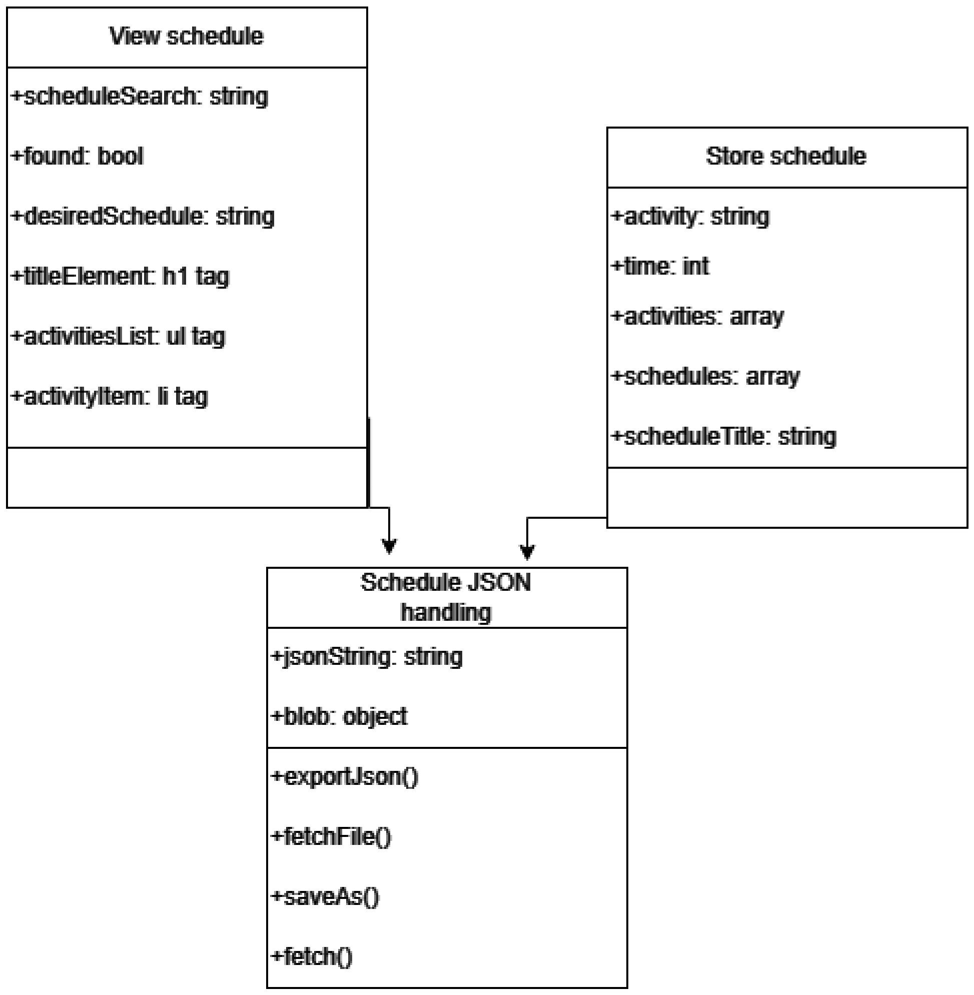
Triggers	User submits an updated schedule
Standard process	Save all details of the schedule to a JSON file in a text format.
Alternative process	No alternative method

Name	Create schedule
Description	User will be asked to set up a schedule and will input data such as how long they want to spend max on different things
Pre condition	User must run the program and sign into their account
Post condition	Any input made will be stored in variables and saved into a file once the user wishes to save their schedule
Error situations	Invalid inputs made e.g. inputting a word when asked for how long you want to spend on a task
System state in event of an error	Ask the user to make a valid input and only store the input if it is valid
Actors	User

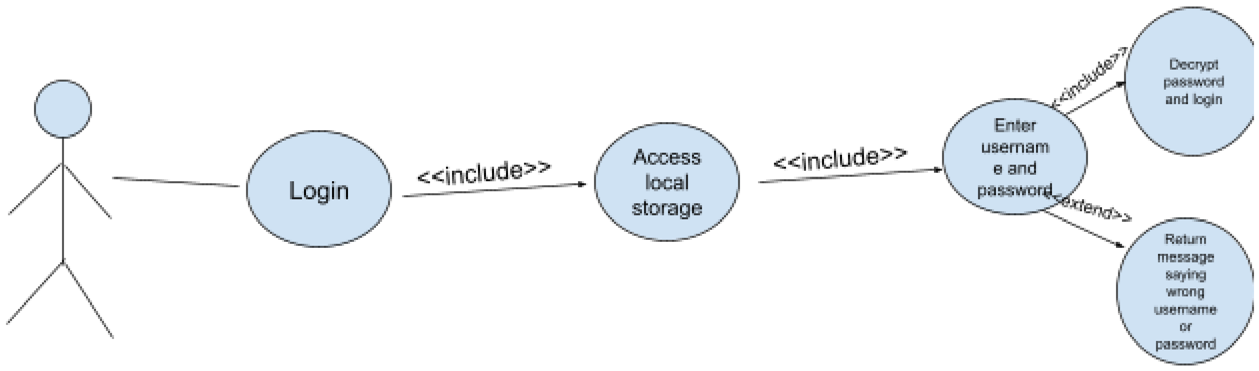
Triggers	User signs into their account and creates a schedule
Standard process	The user will be given an option of activities they want to manage such as leisure, sports, work and will be asked how long they would like to put into activity each day. The program will return suggestions if it thinks they should change any plans but will allow them to not make any changes.
Alternative process	No alternative method

Name	Delete
Description	User can access and then delete an existing schedule
Pre condition	There must be an existing schedule to delete
Post condition	Schedule will be fully deleted and can no longer be accessed
Error situations	No existing schedules to delete
System state in event of an error	Tell user there are no schedules to delete
Actors	User
Triggers	User clicks delete a schedule
Standard process	Remove the entire schedule from the file and ensure it can no longer be accessed
Alternative process	No alternative method

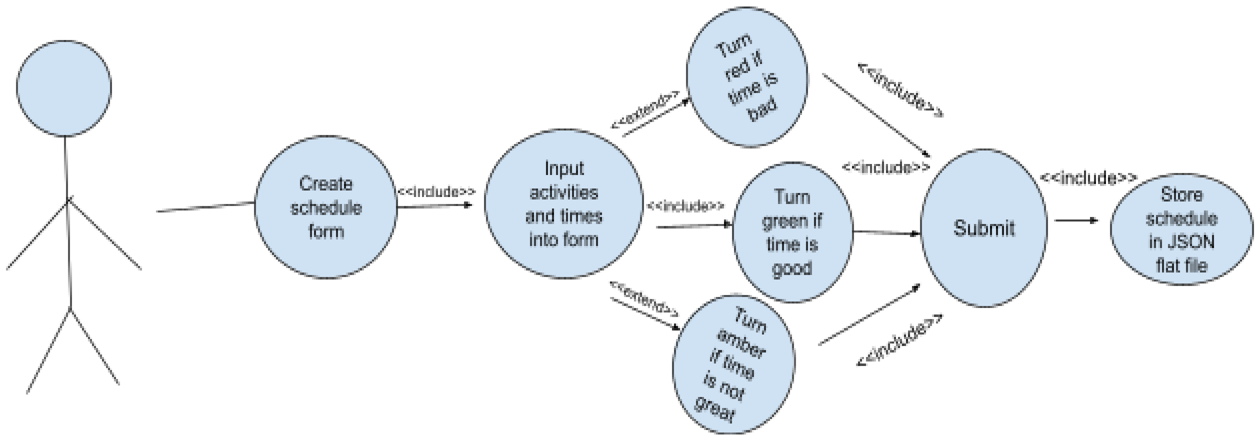
Class diagrams



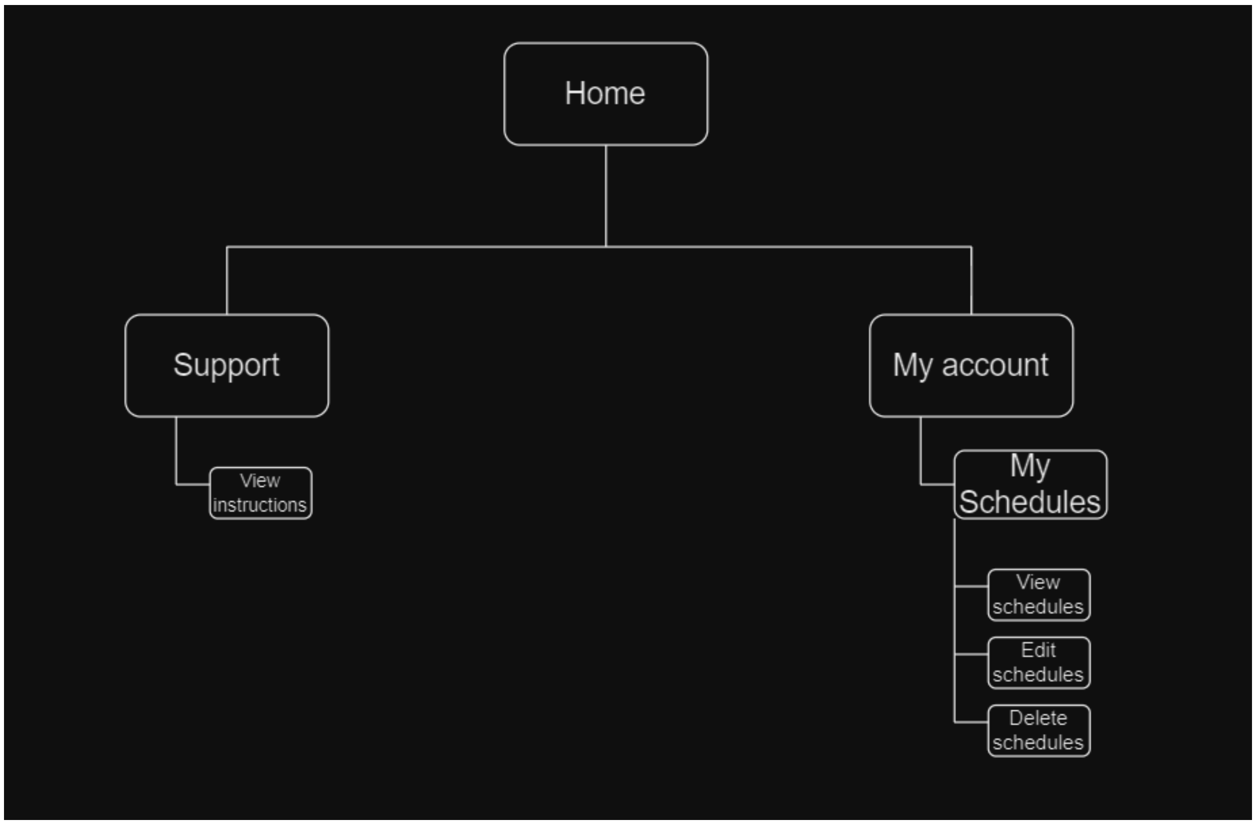
Login


Login
<<include>>
Access local storage
<<include>>
Enter usernam e and password
<<include>>
Decrypt password and login
<<extend>>
Return message saying wrong username or password

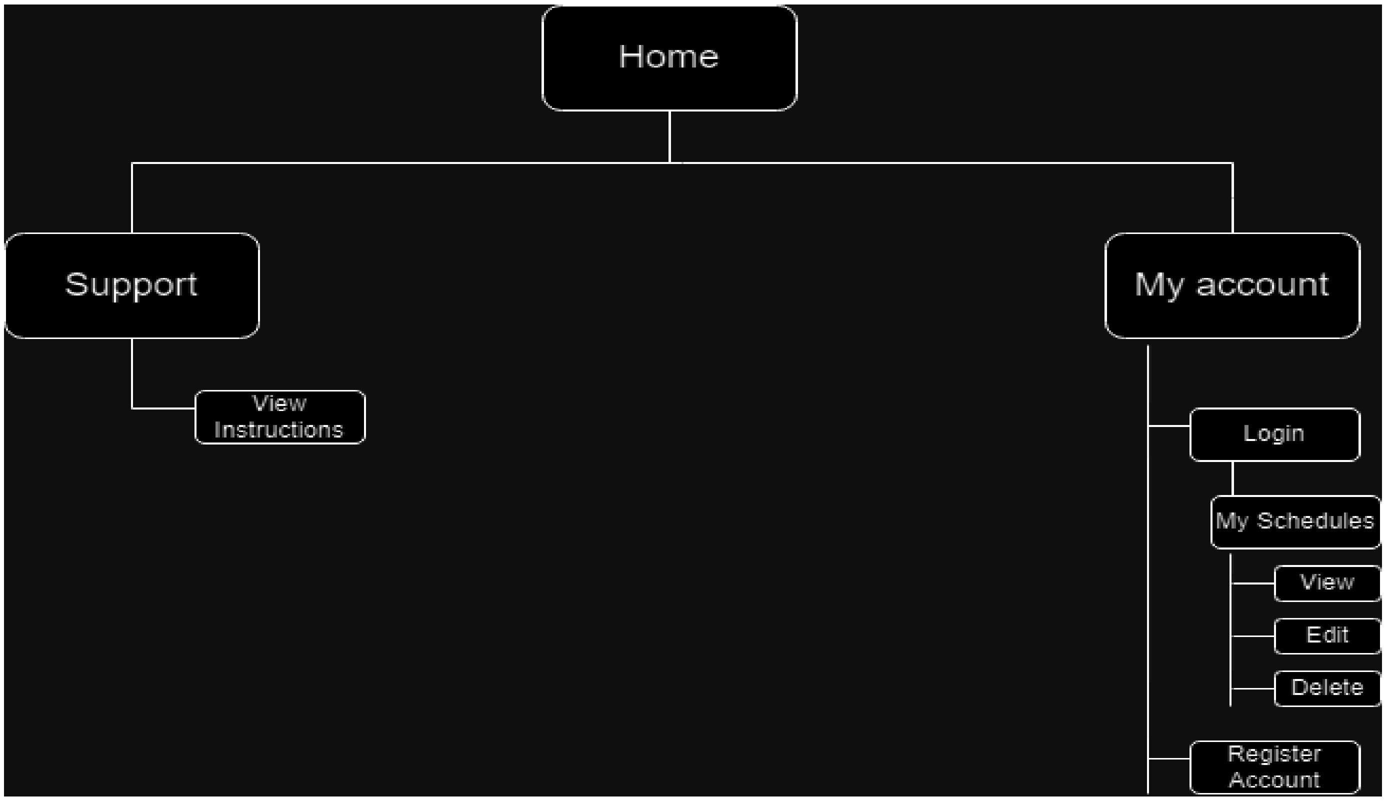
Create Schedule



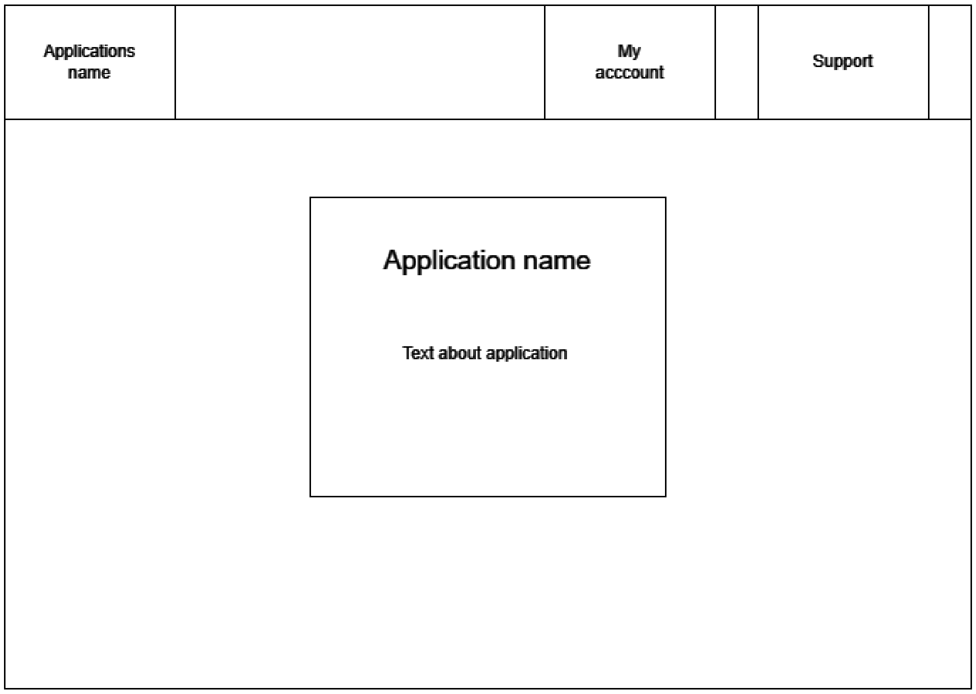
Original Sitemap

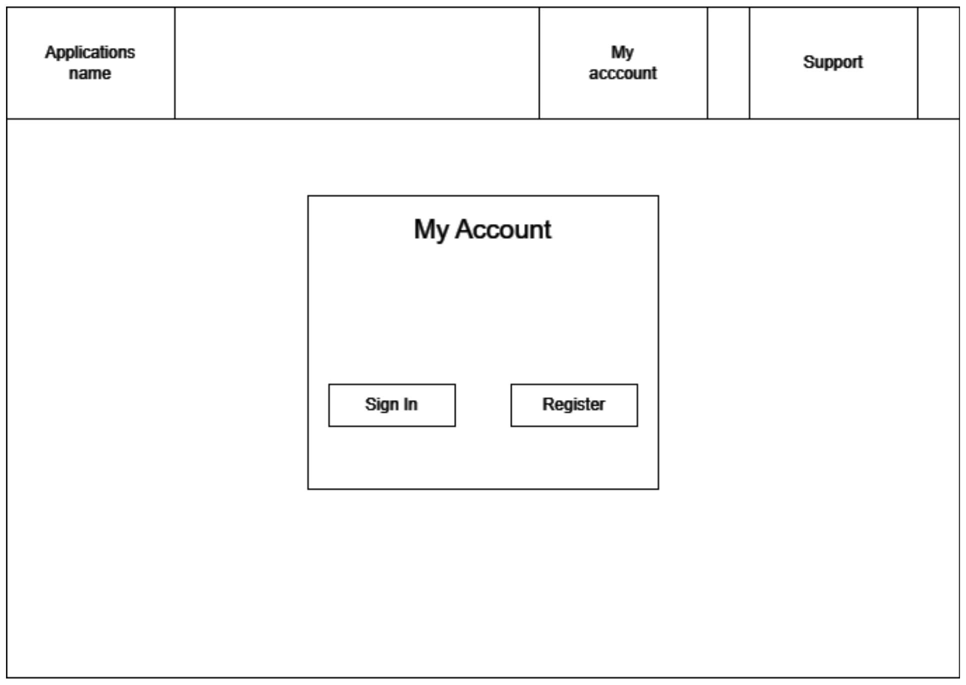


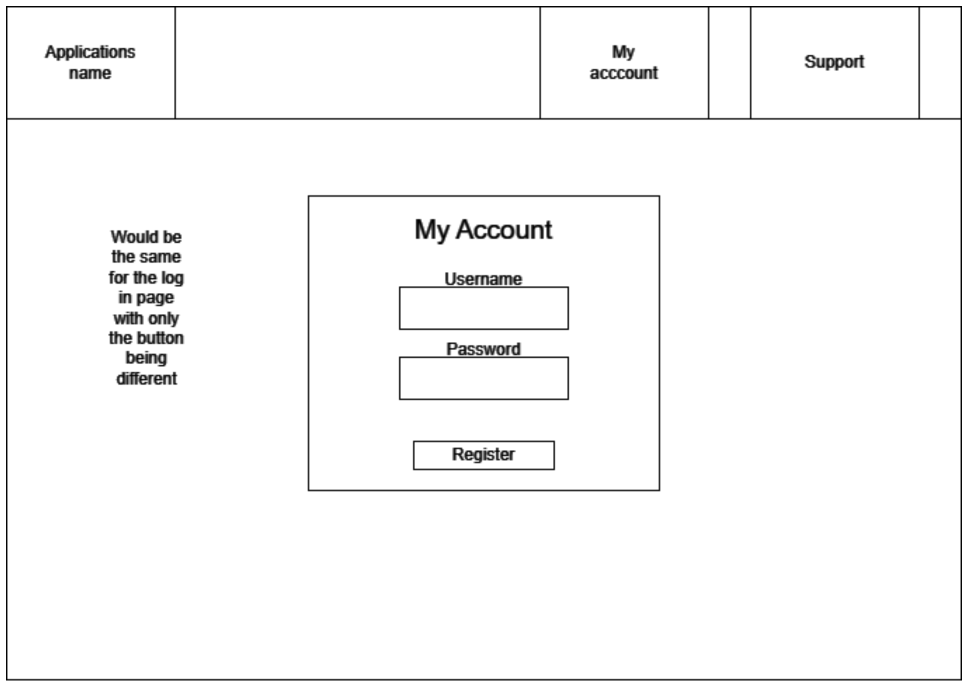
Updated sitemap

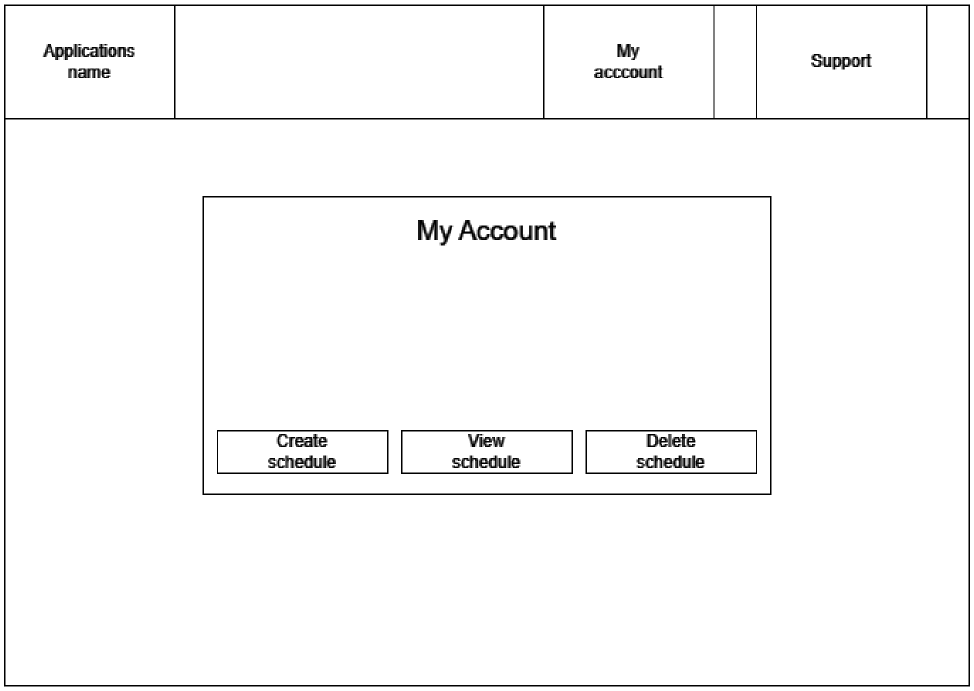

Home
Support
View Instructions
My account
Login
My Schedules
View
Edit
Delete
Register Account

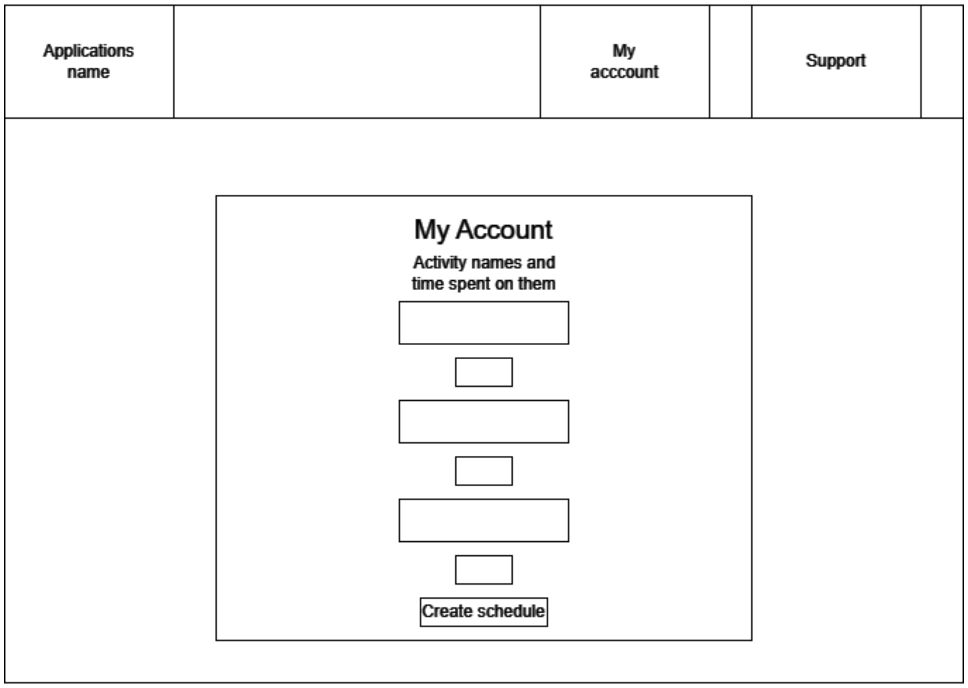
Wireframes

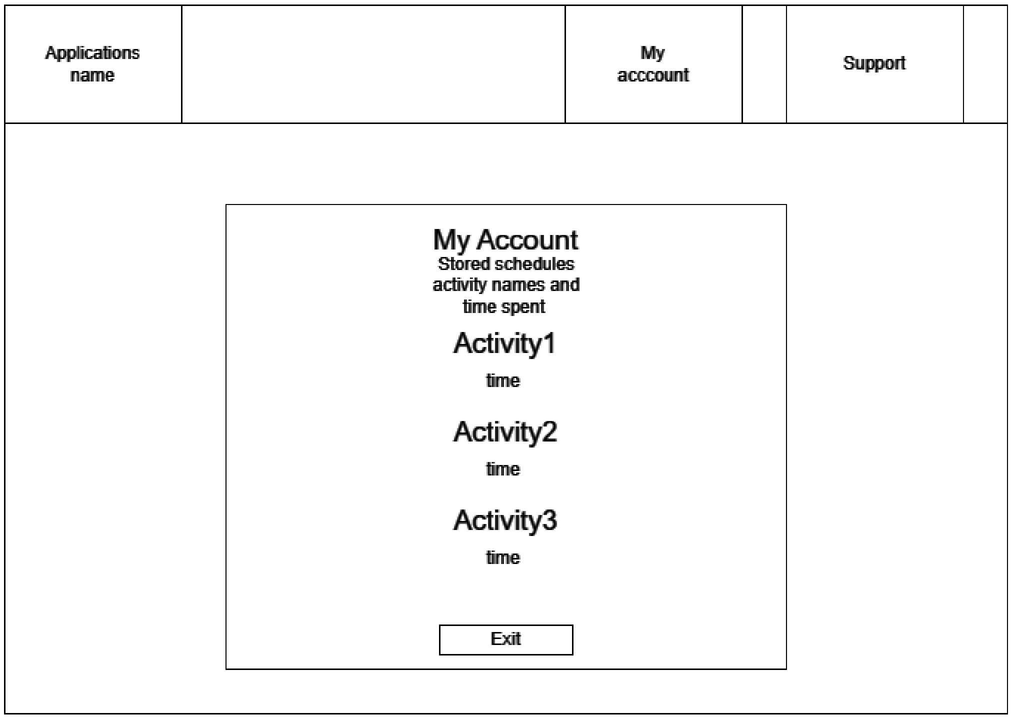

Applications name
My acccount
Support
Application name
Text about application

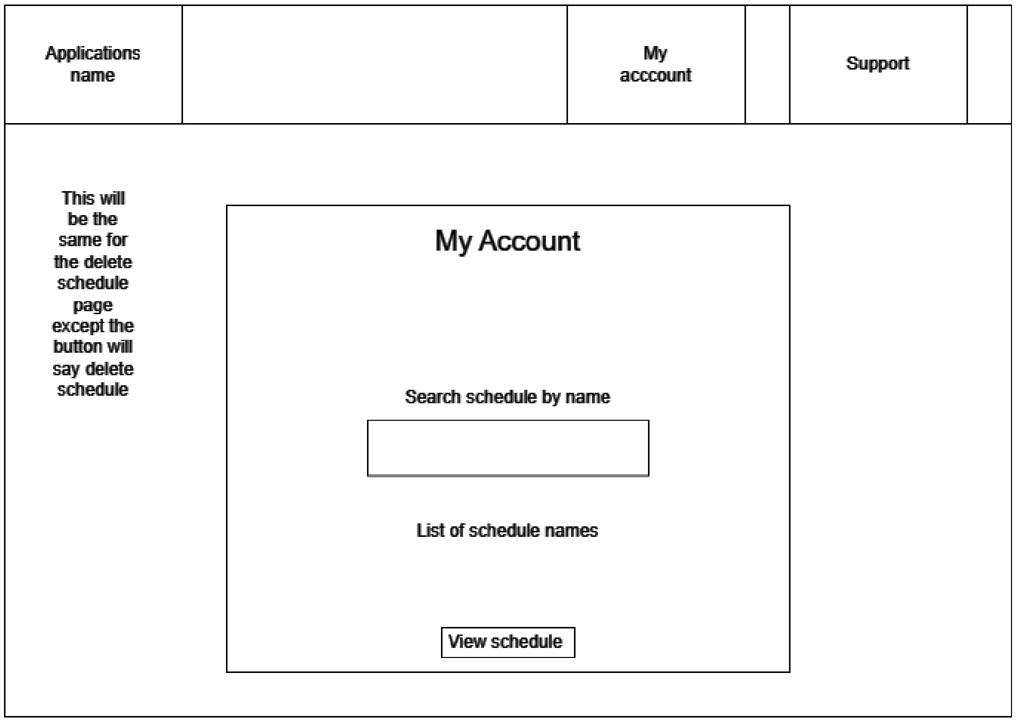

Applications name
My acccount
Support
My Account
Sign In
Register


Applications name
My acccount
Support
Would be the same for the log in page with only the button being different
My Account
Username
Password
Register


Applications name
My acccount
Support
My Account
Create schedule
View schedule
Delete schedule

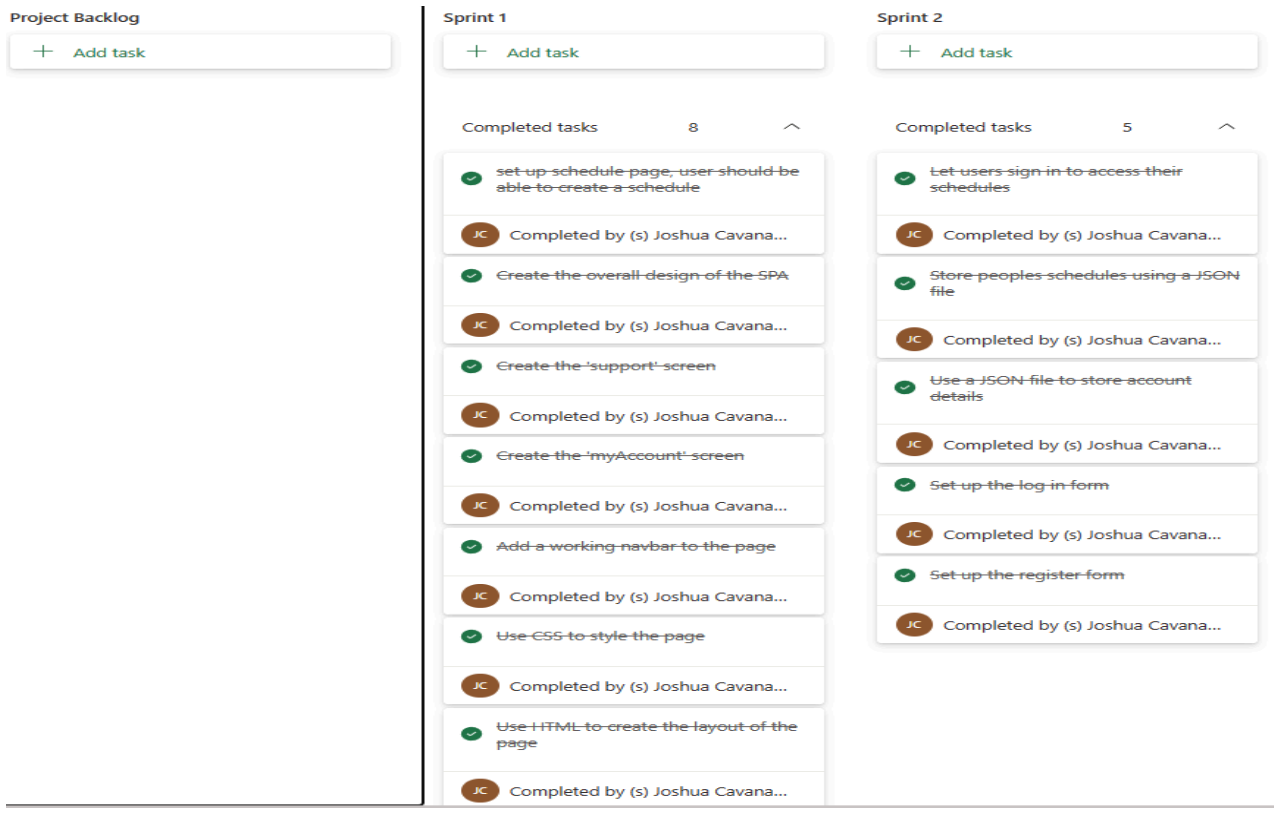

Applications name
My acccount
Support
My Account
Activity names and time spent on them
Create schedule


Applications name
My acccount
Support
My Account
Stored schedules activity names and time spent
Activity1
time
Activity2
time
Activity3
time
Exit


Applications name
My acccount
Support
This will be the same for the delete schedule page except the button will say delete schedule
My Account
Search schedule by name
List of schedule names
View schedule

Sprints

Sprint Planner



Project Backlog

+ Add task

Sprint 1

+ Add task

Completed tasks 8

~~set up schedule page, user should be able to create a schedule~~

Completed by (s) Joshua Cavana...

~~Create the overall design of the SPA~~

Completed by (s) Joshua Cavana...

~~Create the 'support' screen~~

Completed by (s) Joshua Cavana...

~~Create the 'myAccount' screen~~

Completed by (s) Joshua Cavana...

~~Add a working navbar to the page~~

Completed by (s) Joshua Cavana...

~~Use CSS to style the page~~

Completed by (s) Joshua Cavana...

~~Use HTML to create the layout of the page~~

Completed by (s) Joshua Cavana...

Sprint 2

+ Add task

Completed tasks 5

~~Let users sign in to access their schedules~~

Completed by (s) Joshua Cavana...

~~Store peoples schedules using a JSON file~~

Completed by (s) Joshua Cavana...

~~Use a JSON file to store account details~~

Completed by (s) Joshua Cavana...

~~Set up the log in form~~

Completed by (s) Joshua Cavana...

~~Set up the register form~~

Completed by (s) Joshua Cavana...

At the start of each sprint I would place the current tasks in the project backlog and move them over to the sprint section once they are complete.

Sprint1:
Plan:
For this sprint I am going to create the layout of my SPA. I will design the home screen, account screen and support screen. I will also create a login/ register form that you can access through the account screen. I will also create a schedule form that will be used later.
Result:
All went well with this sprint, there were no issues faced and all tasks have been completed. You are able to navigate the different screens using a navbar and will be greeted with a log in or register button on the account page.

Sprint2:
Plan:

Make use of JSON. I will add functionality to the forms, for example when you register an account it will store your account details in a JSON file. When you fill out the login form you will retrieve the stored account data and if it is correct you will be logged in and taken to a schedule form.

Result:

I faced a lot of issues this sprint which resulted in it taking longer than the planned 2 weeks. I was unable to figure out how to write to a JSON flat file as every method I tried did not work, this will be discussed in my reflection. All the forms worked as intended and I faced no issues. Local storage was used as a placeholder for the flat file so that I could continue developing my site.

Sprint 3:

Plan:

For this sprint I plan to work on the schedule part of my site. I will add a page that displays buttons to either view, create or delete your schedules. I will also add a feature that will link schedules to accounts so only the accounts linked can access them.

Result:

There were no issues with this sprint and everything is back on track. When you press the view schedule button you will be given a search bar to search for your schedules. The create schedule button will give you a form to fill in. When submitted it is stored in local storage in a JSON format. The username is also stored with the schedules to create a link between accounts and schedules. The delete schedule page is the same as the search page except it deletes the schedules you enter.

Sprint 4:

Plan:

In this sprint I will add a feature that displays all existing schedules linked to the user's account on the search and delete page so it is easier to find the schedule you need. I will also add some security in the form of the passwords being changed to axtrix's while being entered. Finally I will add a routine to check users schedules and change the colour of the input fields if too much time is being spent on an activity.

Review:

All went well with this sprint. You can see a list of schedules you can search for to view or delete. Working on the password feature was a bit difficult to figure out but by taking in key inputs as an input to build the password to store for the account. This meant I could replace all the characters inputted into atrix's without facing any issues logging in or registering an account. The routine to monitor time spent on activities and provide feedback works perfectly fine as well. I researched recommended times for different common activities and implemented this into the time spent checks.

Sprint 5:
Plan:
For this sprint I have quite a few tasks to get done. I am going to attempt to fix my JSON issues. When submitting data in the schedule form it will be written to a JSON flat file and I will also use this file to view schedules. I will also add encryption to the stored passwords for extra security. Finally I will add a logout button so users can sign into different accounts.
Result:
The logout button was easy to implement and users can easily log out, it appears on the same screen with the schedule buttons. For the encryption of the passwords I used a simple caesar cipher. It's not over the top but will still add an extra layer of security to my site. User accounts are kept more secured and harder to access. In an actual project that would be publicly published I would take the time to learn more complex algorithms. Finally I worked on writing to and reading from a JSON flat file. I am able to write to the JSON flat file. I do this by taking a javascript object and converting it into a JSON format. A temporary anchor element is created which will initialize the download of the json file when clicked. You can also read from the created JSON file. When a schedule is exported I create a copy of all existing schedules and add the new schedule to the list and then store it back into the file. Right now I am having an issue where it will create a new file every time I do this which messes with the reading of the JSON file. I will fix this in my next and final sprint.

Sprint 6:
Plan:
Fix JSON file issue where multiple files keep getting created.
Result:
The reason multiple different files kept getting created was because the method I was using to write to a file can only create new files and not overwrite existing files. I had a fixed file name where I would write the data to. Everytime a schedule is created it would create a ‘filename(1).json’ etc. This meant that I would only be able to read the original file with the outdated schedule list. To fix this I would set the schedule name as the filename and only write one schedule per file. After this sprint I have finished my program's development. I will make sure to keep checking my program for any bugs I may have missed before submission.

Final testing results
I have gone over my program to check for any final bugs I have missed. I found a few bugs that needed to be fixed before I could submit my program. Firstly I found a bug with entering your password. The method I had used where I would take key inputs and

build the password as one string would not include the last inputted character unless another key was pressed after it. I had been pressing enter to submit the password which meant another key was pressed so it worked but when you click the submit button it would not work. To fix this I would check if the current length of the input field is greater than the current password built. If so I would add the last character of the input field to the password. And for the asterisk display I would set it to increase by 1 every time the input size increased.

I was having issues with forms as well. When deleting a schedule it would refresh the page. I found the problem was that I was allowing the default form behavior to run so to fix this I prevented the forms default behavior from running using the event.preventDefault() function. I had the same issue with the search form but for the search form it was also an issue linked to fetching the file. I had to change it so that the form would not submit until the fetch operation is complete. This stopped the error of not being able to read the content of the file. I used this line of code to disable and enable the form submission before and after the fetch operation is complete.
$("#scheduleSearch-submit").prop("disabled", false);

Reflection

After the completion of my project I have to say I am happy with what I have achieved. Going into this I had little knowledge of html and css and had never worked with javascript before in a project. I put the effort in to learn some basic javascript in the weeks leading up to the start of my project which I think helped me keep things moving at a good pace. JQuery was also something very new to me but after working with it in the project I found that it was very easy to learn and was an extremely useful tool that made the development of my project go a lot faster and smoothly.

I have met all my requirements I set for my project and also met the minimum requirements stated for this project by the university. I have created something that can be interacted with easily by users and you are able to write to a JSON flat file and take an input from the same JSON flat file. I have also added security for my site as when an account is made the password that is stored for that account will be encrypted, it is only a very simple encryption but if I were to take the development of my site further I would 100% work on learning a more complex and secure encryption method to ensure the safety of my users accounts. In terms of the schedule part of my site when a user is creating their schedule each field will change colour to indicate if the amount of time spent on an activity is too much or too little and once the schedule is created it is exported to a JSON file which can then be read from to display the schedule. I have also made a feature for the user to delete their schedules if they wish to, this just means unused schedules won't be wasting space for the user.

On the other hand JSON was a completely new and difficult process to figure out for me. I first began by trying to read and write to a flat file but found I could only read from the file as the other write methods I used just would not work, this was very frustrating as it took me more than one sprint to try and fix. The JSON was holding me back so to keep things moving I used local storage for the meantime and stored my data in a JSON format to practice how I will write to the flat file. Once everything else with my project was finished I focused my effort on outputting data to a flat file in a JSON format and then taking JSON data as an input.
The only other issue I faced was how I wanted to design my project. I wanted it to look appealing and I have never dealt with proper web design before so I tried my best to make something I thought users would like the look of. I went with a simple colour design and wanted it to appear something like a task board you'd find in real life. I am not sure if that is the impression users would get but I feel like the final design is still something a user would find visually appealing.

Github repo link
https://github.com/Gilberto5309/COMP-1004-Coursework.git

References

Betsol. (n.d.). 7 stages of SDLC: How to keep development teams running. https://www.betsol.com/blog/7-stages-of-sdlc-how-to-keep-development-teams-running/