**Development documentation**

In this documentation, I will talk about the sprints wherein I will talk about the features and changes I intent to make for that days sprint and then document any changes I have made and why.

**08/02/2024**

In this sprint, I will implement the flask app, the database connection class, the login route, the register route, the index route and then the corresponding pages to them all. I will implement some JavaScript for the navigation bar to show which page the user is currently on and then add the corresponding forms for the pages. I will also setup the database with the documentation from task 1 and then test to see If the data is being inserted into the database correctly.

**Changes**

* I created the basic directory structure for the entire solution

A screenshot of a computer

Description automatically generated

* I setup the flask app and configured it inside app.py

A screen shot of a computer program

Description automatically generated

This imports the needed modules, it creates an instance of the flask app, configures the secret key, sets the session lifetime, imports the backend routes and then runs the app if it is being run directly.

* I created the Database class

A black screen with white text

Description automatically generated

* I created the initialisation method

A computer screen with text

Description automatically generated with medium confidence

* I created the connection method

A screen shot of a computer program

Description automatically generated

This will try to connect to the database and then return the connection, if not then it will return the error

* I created the disconnect method

A screen shot of a computer code

Description automatically generated

* I created the query method

A screen shot of a computer program

Description automatically generated

This method will create the connection, create the cursor, execute the command, disconnect from the database and then return the result

* I created the update method

A screen shot of a computer program

Description automatically generated

This method will do the same as the query method but will commit the changes

* Inside routes.py, I performed all the imports

A black screen with white text

Description automatically generated

* I create an instance of the database



* I created the index page route

A screen shot of a computer

Description automatically generated

* I created the register route

A black background with white text

Description automatically generated

* I created the if statement that will get the user data from the form is the user has submitted the form

A screen shot of a computer program

Description automatically generated

* I created the regular expression patterns to validate the user input

A screenshot of a computer program

Description automatically generated

* I added the code to format the height, weight and location above the regular expressions to ensure the format is correct

A screen shot of a computer program

Description automatically generated

* I added in a query to check if the username or email is already in use



* I imported flask onto the end of the flask import line
* I added in the series of if, elif and else statements to check if each input matches the correct form and is the correct length

A computer screen shot of a program code

Description automatically generated

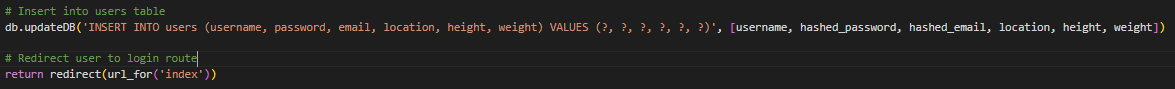
This will check the patterns above for each form input and then check if each form input is within the certain length, if not then it will flash a message to the user

* I hashed the email and password using hashlib

A computer screen shot of text

Description automatically generated

* I created the query to insert the user data and then redirected the user to the index page



* I removed the height and weight from the query to insert the user data because I decided it did not make sense to ask the user to sign up with their weight and height and thought it more appropriate to have the user enter the data on their profile page.
* I removed the checking if statements for the location and the height in the series of if and elif statements since the checking will be performed in another route.
* I removed the height pattern and weight pattern for the same reason.
* I removed the height and weight format statements and the height and weight from request.form for the same reason as above
* A screenshot of a computer program

  Description automatically generatedI added the basics for the login route, same as the register route
* I got the user data from the form

A black screen with white text

Description automatically generated

* I retrieved the user password from the query to find the user where the username is equal to the form username

A black screen with white text

Description automatically generated

* I added in the checks to see if the user account exists and if the password is correct

A screen shot of a computer program

Description automatically generated

With both, if a condition is not met then the user is flashed a password

* In the else statement, if the user login is correct then they are added to the session dictionary and then they are redirected to the home page

A screen shot of a computer program

Description automatically generated

I add the user to the session dictionary to essentially create the illusion that the user is logged in

* I added a check to see if the user is already logged in when they try to access the login page, they will be redirected to the home page

A black screen with white text

Description automatically generated

* I created the users table in database.db

A computer code with text

Description automatically generated

I did not add the weight or the height like initially planned because after some consideration, I decided that it did not make sense to ask the user to enter their health related details when signing up and thought it was better to ask them to enter that information when they go onto their account page which will be added into the health\_tracking table

* I created the health\_tracking table with the height and weight columns

A screenshot of a computer program

Description automatically generated

* I created the conditions table

A computer code with text

Description automatically generated

* I created the risk\_asessments table

A computer code with blue text

Description automatically generated

* In base.html I added the basic of the HTML file and included the Bootstrap CDN links with the content blocks in jinja2

A black background with colorful text

Description automatically generated

* I changed the navigation bar to black



* I added the following to index.html, login.html and register.html

A screen shot of a computer screen

Description automatically generated

* I added a title inside of the block title

A black background with white text

Description automatically generated

* I removed the excess links from the navigation bar

A screen shot of a computer code

Description automatically generated

* I created base.js and added in the a check to add aria-current = page and active to the class for the page the user is currently added on

A screen shot of a computer code

Description automatically generated

This will wait for the whole page to load and then go through each item in the navigation bar and if it is the current page then it will add aria-current = page to the element as well as active so it will be highlighted

* I added in the script tag to link the JavaScript file to the base HTML file and added a comment

A screen shot of a computer screen

Description automatically generated

* I created base.css and added a selector to underline the current page in the navigation bar

A screen shot of a computer program

Description automatically generated

* I added a link to the base.css file

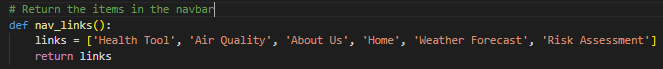


* I added in the login and register routes inside an if loop to check if the user is not logged in

A screen shot of a computer program

Description automatically generated

* I created a function that returns a list of nav items



This will prevent me from having to manually write out each nav item and can instead use one inside of a for loop

* I added in the route for the sole purpose of parsing the items into the base template for the navigation bar

A screen shot of a computer

Description automatically generated

* I removed the base route because it did not work
* I parsed the result of the nav\_links function into index.html instead



* I added the routes list because the links to the pages would not work without the correct routes into the nav\_links function



* I parsed the routes into the index template



* I could not get the for loop to display the items properly so I went back to just listing out all items manually
* I removed the items variable and the routes variable from the index route
* I deleted the nav\_link function
* I added in the logo of the solution to the navigation bar



I changed the logo to a new one rather than use the one in the design because it did not fit on the navigation bar well and it was hardly visible

* I added a comment above the if statement in base.html



* I added in 2 div containers to surround the entire content of the base template and then added a footer

* I added in the code to adjust the footer to stay on the bottom of each page and referenced

A computer screen with green text

Description automatically generated

* I removed two placeholders from the SQL insert query to create an account since it would have caused an error



* I added in register.html the for loop to display flashed messages

A computer screen with text on it

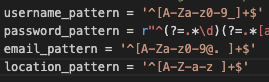
Description automatically generated

This will first check if there is any flashed messages, and if there are it will iterate through all flashed messages and display them in order

* I changed match to search as I could not create an account because re.match was returning None meaning that the username is incorrect so I could see what the output would be



* I removed the print statement in the above image and added a plus to each pattern because it was causing the match function to return none



* I changed the user to be redirected to the login page instead of being redirected to the index page as this was a mistake



* I added in the elif statement to check if the passwords did not match as I forgot this

A black background with white text

Description automatically generated

* I added register inside of the title block in login.html

A black background with white text

Description automatically generated

* I added in the code to display the flashed messages

A computer screen shot of text

Description automatically generated

* I added in the form for login

A black background with text

Description automatically generated

* In the login route I forgot to hash the password and then to compare the hashed password against the password in the database so I added it in

A computer screen with text

Description automatically generated

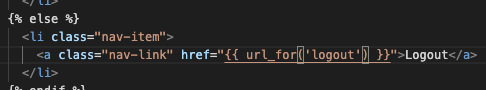
* I added in a logout route

A screen shot of a computer code

Description automatically generated

This will iterate through each session key and then remove it from the session, logging out the user in the process

* I added an extra else statement and then added the nav item to create a button in the navigation bar to enable the user to lop out



* I added in the return statement to return the index page as this was causing an error when the user logged out



* I got an error when trying to login with an account that did not exist as the query would result nothing and then the code would try to access the password of the empty query, causing an error so I moved the organisation around to fix this

A computer screen with text and symbols

Description automatically generated

The code first checks if the account exists, if it does then it will access the query

* The same error was still happening because the code will still check the query if the query returned nothing so I added an else statement

A computer screen with text on it

Description automatically generated

**09/02/2024**

In todays sprint, I will focus on the front end and make the login page and the register page look like the wireframes. I will also add in the remaining needed files for each page and fix the base nav-item links so the navigation bar has got all of its correct links.

**Changes**

* I created register.css and login.css

A screenshot of a computer

Description automatically generated

* I created a div with class form-container and added a link to register.html so I can then adjust the positioning of the form



* I make the new div into a flex box and centred it and moved it down

A screen shot of a computer code

Description automatically generated

* I changed the width of the form



* I create a div and put a h1 in it for the title of the page

A screen shot of a computer

Description automatically generated

* I added the css to turn the h1 div container into a flex box and then position it, I also adjusted the positioning of the form-container upwards

A screen shot of a computer program

Description automatically generated

* I added in the link to the login.css file inside login.html



* I added in the container for the h1 title and the container for the form in login.html just like register.html

A screen shot of a computer screen

Description automatically generated

* I copy and pasted the code form register.css and adjusted it slightly for login.css

A screen shot of a computer code

Description automatically generated

* I changed the form button to say login instead of register



* I added in the routes to return the remaining html pages

A screen shot of a computer program

Description automatically generated

* I created the remaining html pages

A screenshot of a computer

Description automatically generated

* Inside of the html templates I added the following code

A screen shot of a computer

Description automatically generated

* I added in the corresponding url\_for values in the navigation bar

A screenshot of a computer program

Description automatically generated

* I added an alt tag into the navigation bar image



* I added in the website logo



* I changed the form text on the register page to remove the text that was not needed

A screen shot of a computer code

Description automatically generated

* I added in a link that lets the user go to the login page if they already have a register page

A black background with white text

Description automatically generated

* I added in a link that lets go to the register page if they do not have an account on the login page

A screen shot of a computer

Description automatically generated

* I changed the text under the password field on the register page



**11/02/2024**

In todays sprint, I will add the terms and conditions page to ensure the solution is adherent to the law and regulations identified during task 1, when the user registers for an account they will have to agree to the terms and conditions before they can actually create their account. I will also make the home page look like it should in the wireframe. If time permits, then I will add the weather forecast APIs to the weather forecast page.

**Changes**

* I changed the < in the register route to <= for all the form inputs that needed it as this was not allowing the user to register with extreme data such as 4 letters for the username

A screen shot of a computer code

Description automatically generated

* I created tandc.html as the terms and conditions page



* I added in the basic jinja2 code

A screenshot of a computer program

Description automatically generated

* Inside of the block content I added two divs and a h1 tag with the terms and conditions template inside

A screenshot of a computer screen

Description automatically generated

* I changed the template to be specific to Health Advice Group

A black screen with white text

Description automatically generated

* I added the route for terms & conditions

A screen shot of a computer program

Description automatically generated

* I added in the link in the footer



* I caused this error as I forgot to add in the quotes

A black text with black letters

Description automatically generated

* I added the quotes in the parenthesis around tandcs
* I created a tandc css file to style the page and then linked it



* I added classes to the divs



* I then used the classes in the CSS file and turned main-container into a flex box to center it in the page

A screen shot of a computer code

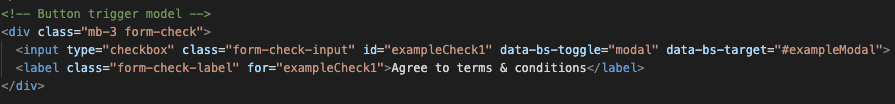
Description automatically generated

* I positioned both the h1 and the text-container

A screen shot of a computer code

Description automatically generated

* I modified the tick box on the register page to trigger the modal when it is clicked



* I modified the modal body to contain the terms & conditions text of the terms & conditions page

A screen shot of a computer screen

Description automatically generated

* I created a register.js file in order to add the functionality to force the user to agree to the terms & conditions before being able to register for an account



* I linked the file in the html file



* I added ids for the modal button and the register button to use in the register.js file

A computer screen with text on it

Description automatically generated

* I added in the code to disabled the registerButton until the user agrees to the terms & conditions

A screen shot of a computer program

Description automatically generated

* This will disable the register button by default, and then wait for the agree button in the modal to be clicked, when it is clicked it will enable the register button and then get rid of the modal enabling the user to register their account
* The above code was not working as I did not link the file correctly, so I changed the link line



* The modal was still not going when the user clicked the agree button so I removed the unnecessary code

A computer screen shot of a program code

Description automatically generated

* I added the code to dismiss the modal when clicked



* I added the disabled attribute to the register button so it is disabled until the user agrees to the terms and conditions



* I created a index.css page and linked it to index.html



* I added in the div tags to then style each container separately

A screenshot of a computer

Description automatically generated

* I added in the background image within the body tag and then positioned the background image

A screen shot of a computer

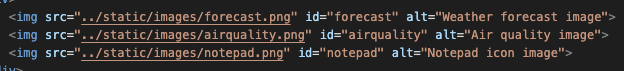
Description automatically generated

I decided to not stick with the original design of the home page as I felt this homepage looked much lower quality and did not give the best representation of Health Advice Group. I believe this background image alone will significantly improve the quality and the look of the website. It will also mean I can lay the content out much better and in a much more natural and fluent way, which will make the home page look a lot more professional

* I added a h1 tag into the div main-container



* I added the 3 images into the div tag



* I adjusted the height of the notepad icon

A black screen with text

Description automatically generated

* I created a class for the div inside of the main-container

A screen shot of a computer

Description automatically generated

* I moved the h1 tag inside sub-container
* I hid any overflowing content inside of the main-container

A black screen with text

Description automatically generated

* I adjusted the height of the main-container as it was causing visual issues

A screen shot of a computer code

Description automatically generated

* I added an extra icon of a house onto the home page and changed the order of the code

A screen shot of a computer

Description automatically generated

I added the image icon because I thought that the homepage was looking too spacious and I thought that adding the icon of the home to show that Health Advice Group do home risk assessments would both look better and convey their services better

* I adjusted the height of the house icon

A black background with white text and colorful letters

Description automatically generated

* I put the images and their h3 tags into two different div tags to put all 4 images in 2 sets of 2 on 2 lines

A screen shot of a computer program

Description automatically generated

* I put the h3 tags under the images

A computer screen shot of a program

Description automatically generated

* I added extra contains in order to centre the individual text and images without affecting the images around

A screen shot of a computer program

Description automatically generated

* I centred each individual div containing the respective text and image

A screen shot of a computer program

Description automatically generated

* I changed the order of the weather forecast and air quality monitoring

A screen shot of a computer program

Description automatically generated

* I added a gap between the divs in both overall container divs

A screen shot of a computer code

Description automatically generated

* I added a gap between the text and the images for each container

A screen shot of a computer program

Description automatically generated

* I adjusted the height of the main-container

A screen shot of a computer code

Description automatically generated

* I adjusted the positioning of the containers

A screen shot of a computer program

Description automatically generated

* I wrapped each image in an anchor tag to make them all clickable and then added a href attribute to redirect to their respective pages

A screen shot of a computer program

Description automatically generated