DC1IAP Wright Harison Report 3

Word count: 1829

Harison Wright

2023

# Introduction:

### HLSP Description:

The requirements of the HLSP are as follows. Requirements in blue are stretch goals:

* For users to input their daily routines into the app. For example, details about their eating habits, exercise, work, alcohol consumption and sleep patterns. Also, the user can opt into the collection of some of this data such as step count, sleep data and resting heart rate from a third-party solution such as Apple Health or Fitbit.
* Based on a user’s profile/data the HLSP will suggest relevant resources to attempt to positively impact someone’s health. For example, if the user is overweight it might suggest articles about healthy eating or exercise.
* The HLSP will have set parameters for each of the metrics it collects. These will be based on parameters set based on reliable sources. For example, for alcohol consumption, the NHS recommends that adults should not drink more than 14 units per week so if a user inputs that they drink more than that then HLSP will start to recommend relevant resources around alcohol consumption.
* The HLSP will also run ads for local events or establishments that are related to fitness. This is how HLSP will monetise this web app. For example, it could suggest a free day pass to a local gym or a free yoga session to the user based on the target demographic of the ad. The app will then make money from the referral if the user decides to sign up for the local establishment through the ad creating a win, win, win for the user, Healthy life, and the local establishment.
* As the user builds their profile or it changes over time the HLSP should suggest articles more relevant to their lifestyle choices.
* The articles suggested must be from a reliable source
* The HLSP must not provide any medical advice on how to treat medical conditions of any kind, which includes mental health.
* The HLSP will also have the capability should a user opt-in to send a weekly newsletter containing articles relevant to the user and a local establishments promotion.
* The HLSP will also implement 2FA as the data held on the site is very personal so this should help make sure that only they can log in to it.
* The HLSP will also support 3rd party login support to allow users to sign up and login easily which will hopefully entice more users to sign up. For example, log in with google

### User Scenarios:

Tom is a 20-year-old male who likes to play football and go to the gym. He is a university student. He is also light for his height and age. He is also a heavy drinker. He uses an apple watch and wants the HLSP to suggest articles to him about how he can improve his fitness to perform better in his football games. The HLSP should try to suggest relevant articles about how reducing his drinking habits and healthy eating would help him to achieve his goals. It could also consider his metrics and data from his apple watch if, for example, he wasn’t getting enough sleep or enough quality sleep perhaps due to his heavy drinking to suggest articles about sleep.

Dan is a 27-year-old male software engineer that works from home. He works Monday-Friday and plays sports on Tuesday nights. On weekends he is usually active. Based on the data collected from his Fitbit on working days of the week when he does not play sports his average step count is around 2700 steps, and he often does not leave the house. The HLSP should consider this data and then suggest relevant articles about the importance of trying to get out of the house every day as a young professional. It could also suggest articles about how to find time to be more active as a busy professional who works from home.

Margery is a 67-year-old female who has recently recovered from lung cancer. She is currently unable to walk much further than the end of her street. Before she had lung cancer, she liked to walk around the local reservoir a few times a week. She would like to work on her physical health so she can return to doing this again. The HLSP should try to suggest articles about improving stamina that is specifically aimed at the elderly with very limited mobility. It should under no circumstances provide any medical advice.

# Software Product Implementation:

## Unit 5. Web Application Security:

• **Which security issues are you considering on the client side?**

I am considering the security issues on the client side by using the following methods:

1. Protection against CRSF attacks by using the JWT token inside a cookie to authenticate the user when they make a request to the server.

2. Protection against XSS attacks by sanitizing the input from a user using the sanitize-HTML package. For example, when a user creates a new HL Post the Title, Body and filename of the post are sanitized before being added to the database.

**• Which security issues are you considering on the server side?**

I am considering the security issues on the server side by using the following methods:

1. I am using environment variables to store Secrets and API keys.

2. I am using the .gitignore file to ignore the sensitive files and folders such as the .env file and the MongoDB folder and stop them from being pushed to the remote git repository.

3. I am using an authentication strategy to authenticate the user.

4. I am using an authorization strategy to authorize the user.

5. I am using a low expiration time of 1 day for the JWT.

6. I am using the Mongoose package which helps to prevent NoSQL injection attacks.

7. I am using the bcryptjs package to hash and salt the passwords before being stored in the database.

**• What authentication strategy have you implemented on the server?**

1. Using a username, password and optionally 2FA to authenticate the user.

2. After a user is authenticated, the server will generate a JWT and send it to the client as a cookie.

3. The client will send the cookie with the JWT to the server for every request.

4. The server will validate the JWT for routes that require authorization and if it is valid, the server will process the request.

5. If the JWT is invalid, the server will return a 401 Unauthorized response.

## Unit 6. Frameworks and Architecture:

**• How did you implement the DRY principle in the HLSP?**

I used JavaScript functions to implement the DRY principle in the HLSP.

For example, I used JavaScript to loop through the posts and display them on the HL Posts page rather than copying and pasting the code and making a section for each post.

I also used JavaScript to populate the Profile page against a dictionary of the users’ info and the corresponding HTML element ID. I used a for loop to loop through the dictionary and set the innerHTML of the HTML element to the correct value from the server. Rather than repeating this code for each HTML element.

I also used one category page to display all the categories. For example, when a user requests the exercise category, they send a request to the server ‘/category.html?category=Exercise’ with the parameter ‘category=Exercise’. Similarly, they can request the health category by requesting ‘/category.html’ route with the category ‘Health’.

**• How did you overcome the STATE problem in the HLSP?**

1. Refreshing the page after a change is made such as when a user uploads a post so the state is updated, and the user can see the post that they just uploaded.

2. I also made changes to the state of the application for example, when a user likes a post, the state of the application is updated, and the number of likes is increased by 1.

3. I also added modal popups to the application, so the state of the application is updated when they click on the modal button and then the modal popup is displayed. For example, the Setup / Edit 2FA modal popup.

4. Generally most client-side actions perform a request to the server and then the state of the application is managed server-side.

**• How did you architect the separation of concerns (e.g., Domain Model vs**

**Presentation)?**

I architected the separation of concerns by using the express framework and the MVC pattern. I mainly use client-side rendering for the dynamic content. The controller on the server sends JSON data to the client which is then rendered using JavaScript.

The Model is on the server side and is responsible for the data and the logic. The View is on the client side responsible for the presentation of the data. The Controller is on the server side and is responsible for the communication between the Model and the View.

# Conclusion:

On reflection, I think my implementation of the Health Life Web Application has been a huge success. I think that my UI looks good thanks to the implementation/customisation done with bootstrap and is also clear and easy to use especially after I added the welcome page to explain some of the site's features to a newly registered user. I have written lots of useful JS to populate my front end with data from my APIs as well as send data back to my APIs. I am quite proud of my APIs functionality, especially my implementation of 2FA. However, I have struggled to add functionality for sign-in with google or apple. This is because I decided to build my own user authentication system but then I found it very difficult to integrate this with google sign in, so I decided to remove it. If it were to do it again, I would use a library like Passport as I have found that it makes user authentication much easier and it has support for services such as google and apple sign in making it much easier to implement. I also found it difficult to find a good free public API to use to pull relevant articles. The current News API I am using leaves lots to be desired as the results aren’t always that relevant as it can only search from the last 30 days in the free tier, and it doesn’t return an image relevant to the articles despite it being in the documentation that is does, so I suspect that this is also a paid feature. In the future, I could investigate web scraping more to see if it would be viable to implement. I think I have secured the application quite well although If I was deploying the application to a production environment, I would create an SSL certificate for the server and use HTTPS (using the https module) to encrypt the data that is sent between the client and the server. I would also like to improve the test coverage of both the front-end and back-end code before deploying this application to the public.

# References:

Display the sign in with google button;|; authentication ;|; google developers (no date) Google. Google. Available at: https://developers.google.com/identity/gsi/web/guides/display-button#html (Accessed: October 27, 2022).

Mark Otto, J.T. (no date) Bootstrap Documentation, · Bootstrap v5.2. Available at: https://getbootstrap.com/docs/5.2/getting-started/introduction/ (Accessed: October 27, 2022).

Apple Developer Documentation (no date). Available at: https://developer.apple.com/documentation/sign\_in\_with\_apple/displaying\_sign\_in\_with\_apple\_buttons\_on\_the\_web (Accessed: October 27, 2022).

# Appendices:

Sites used for research/inspiration on how to layout my site:

[BBC - Home](https://www.bbc.co.uk/) – Layout of news articles

[Health: Trusted and Empathetic Health and Wellness Information](https://www.health.com/) – Layout of articles + newsletter

<https://twitter.com/> - sign in / sign up page

[https://www.facebook.com](https://www.facebook.com/) - sign in / sign up page

Software Product Implementation from Previous Submissions:

## 1.1 Unit 1. HTML5

### Structure / Content of the HLSP:

Example

What is the recommended HTML page structure/content for the HLSP? (Content refers to media, images, links to the webpage, etc.). Note: Justify the chosen HSLP structure and explain the purpose and value of each content element, making clear connections with the HLSP. You may find it useful to include user scenarios2 and related approaches.

#### Navbar and Footer:

There are consistent on every page to keep with the theme of the site. I also chose a dark theme as this is what I prefer, and it makes the colour that is used on the page stand out more. It contains the site's logo as well as navigation for the user.

#### Index.html

Index.html is the landing page for the website. This is the first page someone will see and therefore it contains the company logo, a summary of what the website is for/about and a button so they can register as a new user for the site. The button is coloured bright blue which stands out from the rest of the site's dark theme to draw attention and entice the user to click.

#### Login.html

Login.html is the page a user sees after clicking either the login or register button. Margery is a 67-year-old and therefore, might not be the best with technology so as a result, it is important, she can find everything easily. As a result, I have made it so that both the login and register forms are on the same page, so it is simple for the user to navigate. Hopefully, new users will find it easy to sign up and therefore they are more lightly to return to the site. There is also an option to register for the newsletter which is selected by default to try and entice users to sign up for the newsletter. It also has a 2FA box so users can securely log in to their accounts. It also contains the signup and login with google buttons for the user’s convenience.

#### Home.html

This is the users’ home page that they visit once they have had a successful login. Here it recommends a top 3 pick of articles or ads for the user as well as 5 articles from 3 categories highlighted to be a benefit/interest to the user. I may also add a banner using JS to ask the user if they would like articles emailed to them via the newsletter if they are not already opted in.

#### Profile.html

Here is the page where a user can update /set their health info. They will be redirected to this page after signup. They can also amend their login user info and subscribe/unsubscribe to the newsletter from this page. They can also set up/edit 2FA on this page using the modal after clicking the 2FA button.

#### Connections.html

Is a page where users can sync their Apple Health or Fitbit info to the site. This will then update parameters such as average step count, average sleep, and resting heart rate automatically for the user. This will be useful for users Tom and Dan as they both have fitness trackers that use either Apple Heath or Fitbit.

I have used this tool to ensure my HTML is correct: [The W3C Markup Validation Service](https://validator.w3.org/)

## 1.2 Unit 2. CSS

**• What features of a well-designed web page have you applied to the HLSP?**

Grid layout, On the home page I used an F-based layout placing the most important/relevant content in this shape. I used visual hierarchies with colour and pop-out boxes to draw attention to the content.

**• In what parts of the HLSP have you applied CSS?**

I have applied CSS using bootstrap to every page. I also applied CSS directly to ensure the footer remains at the bottom of the page even if the page is shorter than the VH.

**• How did CSS support the HTML structure and the HSLP elements’ positions?**

Helps set out the page so it is clear for the user and aesthetic.

**• Have you applied style rules to the HLSP?**

Yes, using bootstrap, I have applied many rules such as columns extending/getting smaller depending on screen size, centring text/content, putting the login /register forms into their respective boxes etc.

**• What tools have you used to verify that your CSS is correct?**

This tool: [The W3C CSS Validation Service](https://jigsaw.w3.org/css-validator/)

**• Which platform(s) does your web implementation support?**

It should work on most platforms, but it is specifically designed to work on both desktop and mobile. For example, as the screen size gets smaller the navbar also puts the navigation into a drop-down list to help usability for the mobile user while keeping a good experience for the desktop user.

**• How do you provide support for different screen widths?**

As I am using the front-end framework bootstrap, I am easily able to code my website to be compatible with many different screen widths using a grid layout. The layout changes as the screen size get bigger / smaller.

**• What front-end framework(s) (e.g., bootstrap) are you using if any?**

I am using bootstrap 5.

## 1.3 Unit 3. JavaScript.:

**• How do you propose the user would interact with the HLSP using JavaScript?**

The user doesn’t see JavaScript. It is loaded when the user loads the page in their browser. Then depending on the implementation, the JavaScript is either run automatically to add to the page or it is run as the user interacts with a page. For example, when they click a button.

**• During your implementation, what advantages/disadvantages have you found**

**regarding the dynamically typed characteristic of JavaScript?**

Advantages:

• Don’t have to change or cast the type of an int to display it in a string you can just use the + operator and it will work. – Reduces the time and effort it takes to code.

Disadvantages:

• Vague error messages can make it harder to find errors compared to statically typed languages. – Increases time to debug broken code

• Less meaningful code completion/suggestions in the IDE. – Increases the probability of making a mistake

• IDE doesn’t highlight where / what the error is as well as it does for statically typed languages.

**• What elements of the DOM have you used to read and modify your HTML pages in the HLSP?**

I have used many elements of the DOM to read and modify my HTML pages. For example, I have used an event listener attached to the ‘DOMContentLoaded’ function to ensure my JS isn’t run before the DOM is loaded. I have also used functions such as getElementById to get elements from the DOM and either manipulate them or add functions to be run when they are interacted with. For example, I have also used the innerHTML property to change the text of an element and the style property to change the style of an element. Also, I have used the appendChild function to add elements to the DOM.

**• What tool(s) have you used to fix errors in your JavaScript code?**

I have used the console in Edge's dev tools to fix errors in my JavaScript code as well as the debugger in Edge to step through my code.

**• Where did you use Event Driven Programming to trigger JavaScript**

**dynamically in the HLSP?**

• When the user clicks on the save button on the profile page the JavaScript will run to save the user’s profile information.

• When the user clicks on the 2FA button on the profile page the JavaScript will run to get the user's 2FA secret and display it to the user then again after the user enters a 2FA code to verify it and update the database to say the user has successfully set up 2FA.

• When the user clicks on the login button on the login page the JavaScript will run to check the user's credentials with the backend API and then either log them in if they are correct or display an error message if they are not.

• When the user clicks on the register button on the registration page the JavaScript will run to check the user's credentials with the backend API and then either register them if they are valid or display an error message if they are not.

• When functions are run after the DOM is loaded such as the function to add articles to the cards on the home page

**• How did you test your JavaScript code? Did you use a JavaScript framework? What was your testing strategy?**

I used the console in Edge's dev tools to test my JavaScript code. I would run the code and then check the console to see if there were any errors. If there were errors or the function didn't work as expected I would make amendments to the code.

I wrote my backend API first and then used Postman to test my API endpoints to ensure they were working as expected. Then I could be confident that it was the front-end JS that wasn't correct when coding.

I also used Jest and Supertest to test my login and register API endpoints however I plan to expand upon this in the future to get to 100% code test coverage.

## 2.1 Unit 4. Node.js:

**• What is your proposal to use the Node.js execution environment as a Web**

**Server?**

I used Node.js as the webserver to serve static files such as HTML, CSS and JavaScript files using Express. This highlights a big advantage of Node which is that it has a large package library to use which can help you code and implement things faster. For example, I also used the JWT package to create JSON Web Tokens.

I also used Node.js to run the backend API that would handle the requests from the front end and update the mongo database.

Node.js uses the JavaScript runtime environment and I can use it to run JavaScript code on the server side.

This means that I can use the same language for both the front end and the back end which saves time as I only have to learn one language and it also makes it easier to reuse and maintain code.

**• What features/services will be provided by the server?**

• Connection to the Mongo database

* User API endpoints will handle requests from the front end for things such as logging in, registering, getting articles, saving articles, getting user profile information, saving user profile information, setting up 2FA, verifying 2FA code, etc.

• Articles API endpoints to handle requests from the front end for things such as getting articles.

• Posts API endpoints to handle requests from the front end for things such as getting posts, saving posts, etc.

• Serve static files such as HTML, CSS, and JavaScript files

**• What are the advantages/disadvantages of using Express as a Web Application**

**Server?**

Advantages:

• Express is a popular framework which means there is a lot of material online to help you learn it.

• Node.JS / express is fast because it is non-blocking and asynchronous which means it can handle a lot of requests at the same time.

• Node.JS has many other packages that can be used to help you code and implement things faster.

Disadvantages:

• Sometimes there can be so much material and so many approaches to solving a problem that it can be hard to know what the best way is to do something.

**• What database have you selected to achieve data persistency in the HLSP?**

I have selected MongoDB as the database to achieve data persistence in the HLSP.

I chose MongoDB because it is a NoSQL database which means it is a document database.

It is a good fit for the HLSP as the data is not relational and it is easier to store and retrieve data from a document database.

**• What are the advantages/disadvantages of your selected database to offer data**

**persistence in the HLSP?**

Advantages:

• NoSQL databases are easier to use and scale than relational databases like MySQL.

• Node has the package Mongoose which makes it easy to connect to MongoDB and use it in Node.

• MongoDB is a popular database which means there is a lot of material online to help you learn it.

Disadvantages:

• MongoDB is not a relational database, so it would not a good fit for the HLSP if the data was relational.

• Couldn't store the post images in MongoDB as they are too large, so I had to use a file system to store them and then store the path to the image in MongoDB.