Project Management 3.7/3.8 Final Reflection

Throughout the development of my booking program, I used Trello as a project management tool. This was my first time using Trello, and I loved how it streamline the agile development process. I had set up three different lists, which help me organise what tasks I had to do, what tasks I was doing, and what tasks I had to do. For each of these tasks, I used a Trello card. These Trello cards were excellent. By being able to add comments to each card, this greatly allowed me to document the changes and refinements to each aspect of the program, which is a key part of the Agile Development process.
Additionally, the ability to add checklists to each Trello card greatly improved my ability to plan ahead and organise what I need to do next. Any feedback I received could be added as a comment to these cards. The feedback I received from my peers was incredibly useful itself, from aesthetic points such as the colours used, usability feedback on the spacing of buttons and input elements, and functionality comments which helped refine the JavaScript scrolling when the user moved between steps. Each time I received some of this feedback, I added it as a “To-Do” in a checklist. This breaking of the program in tasks made my development process methodical and structured. I could logically move from refining on aspect of the program to the next.
Compared to the Power Points I used to use to organise my projects, Trello biggest benefit was facilitating effective planning so I wasn’t wasting any time figuring out what my next step is.

I followed the Agile Development process throughout creating this program. Agile Development is a process which has a focus on feedback, so I aimed to receive a range of feedback to improve its quality. During ideation, I got around 15 people to provide their opinions on fonts and colour combinations. I received more feedback on my wireframe as well. More importantly, I periodically received feedback during the coding of the website, especially on the programs GUI. This meant that my development process was largely driven by this feedback I received. One particular example of this is the feedback I received from my teacher to change the appearance of my reset and submit buttons. Her reasoning was that my current colour palette went against UX conventions. Having a red disabled submit button made it look like a reset button, and having a reset button style in the same way as my navigation buttons meant there was the potential for a user to click it thinking it as something else. By changing the button to red, and the submit button to a grey when disabled, the user can identify the function of these buttons by their colour alone, which should decrease the likelihood that they misunderstand these button’s function.

Like trying Trello as a project management tool for the first time, this project also saw me change IDE to Visual Studio Code. I previously used Microsoft Expression Web to write my HTML and CSS, which is much less text based than VSC, so there was a bit of a learning curve. However, VSC has the advantage of being highly modular due to various plugins you can install. One example of this being advantageous to me is that I can code all 3 of my languages in one IDE, whereas with Expression Web I would have had to use a separate editor for JavaScript. Additionally, these plugins allowed me to streamline several steps of the development process into the coding process. For example, a beautifier plugin cleaned my code every time I saved, and by installing the Java Development Kit, I had W3C

validation in my IDE in real-time. In the past, I had to use the Dirty Markup website and the W3C Validator website, copying and pasting the content of my each of pages each time. Now, I can save this time by automatically cleaning my code and fixing my mistakes as I go, not later, which leaves me more time for things like feedback and refinement.

I made several decisions to improve the functionality of my GUI. One such was the way I attached the event listener of main JavaScript function. Using a for loop and the addEventListener method, this function runs on input. Because all my error messages, booking output divs updates, and button updates stem from this function, each time the user types they will visually see an update to the GUI. This is important, as it reduces the amount of time that the user has to spend going back to fix their errors. Instead of having to hit the submit button and being told their inputs are invalid, instead the user can receive feedback as they go, allowing them to correctly input information as they go.

One example of a refinement to the GUI is the outputting of the user's booking information. In my JavaScript, the user's booking information is stored in a JSON object, as this is how the information is sent to firebase. As a proof of concept and a way to test that my inputs were storing correctly, I iterated through each key and value pair of the object, adding them to a string with line breaks. This was then displayed as the innerHTML of the output div. When I wrote this, I intended to delete it later, and then instead use individual divs to output each value, as this was what I thought would be the easiest to style. The main problem I had with the method at the time was that I had to use camelCase for my key names, which don't look pretty when displayed to the user. However, I realised with string methods and regex I can format the camelCase string values to be capitalised and spaced. Combining this outputting the JSON object as an HTML Table object instead of string allowed me to keep the same efficient coding method, while still refining it to be more aesthetically pleasing and more usable in an easier to read state.