Splatoon 3 Table Turf Battle Deck Builder was an ambitious project, but one that I had a lot of passion towards. This was a project that I would have pursued on my own time, but Intro to ITWS gave me a vehicle to pursue this project with teammates during allotted school time. The project started quite simply, a system for dragging and dropping Table Turf cards into 15 empty slots, being able to download and reupload that as a JSON file, and then being able to share that specific file with other users of the site via a database which stores both the files and the values for display.

This project initially started with the realization that Table Turf had no equivalent within the Splatoon community like Sendou.ink has. The project never really gained much scope outside of that, outside of some fantastic suggestions from the community. The only features that were added within the last semester that wasn't intended are the FAQ page and the Tutorial page. Outside of that, I had a vision, and that vision was fully realized by the end of the semester.

This is not to say that this was all smooth sailing. Our group struggled with work distribution, and also with good Github usage. Many times new features went untested, and features were accidentally made on old branches that should have been deleted. But regardless of that, we ended off with a complete site, and I would like to discuss each aspect and the decisions we made regarding them.

Let's start by discussing what the user sees first, then peel away at the layers until we reach the backend of the site, starting with the aesthetic. The aesthetic of the site was an extremely important element to me, personally, since the Splatoon series has such a distinct visual style, which I wanted to fully capture within the website. I did a lot of studying on how Splatoon presented itself, its layouts, colour contrasts, and so on. I choose the two major colours of the site (purple and yellow) based on the main colours of Splatoon 3[1] since that is the game in which Table Turf was added as a mode.

After that, the user will use the architecture and flow of the site. I tried to keep this simple, there are no pages within pages. There are only 5 main pages: Home, Build Deck, Browse + Share Decks, FAQ, and Tutorial. I formatted the home page to guide the user naturally to the Build Deck, then have the now highlighted Build Deck lead the user's eyes across the other pages the site has to offer. All of the pages were also designed with user vision in mind. Build Deck split all of the functions into 4 quadrants, and had the more prominent ones take up a greater height. Browse was also inspired by many social media sites since this was made for

[1] (note: every Splatoon game has a main color pairing. Splatoon 1 was orange and blue, Splatoon 2 was green and pink, and splatoon 3 is yellow and purple).

searching and scrolling. The FAQ was also quite straightforward. I was particularly inspired by Nintendo's old FAQ page, which differentiated the questions and answers with their two colours colliding, the only difference is that I used JQuery to create a showable answer, instead of the static Q and A that Nintendo's FAQ uses. And finally, Tutorial was designed like a Wikipedia page, which allows users to jump to the section they desire.

And for the aforementioned architecture, we have sorted several of our files into folders for ease of access and organisational reasons. Class Resources is where all of our documents for the class (personas, proposal, presentation, mockup, etc) were stored. Then we have Pages, which is where the sites mentioned in the last paragraph lay. And finally, we have Resources, which holds a lot of files. It has 5 folders CSS, Fonts, Images, JS, and Uploads. These folders store our CSS files, font files, image files, JavaScript files, and uploaded files respectively. This makes our site architecture extremely easy to navigate and allows ease of access when it comes to the relative linking of files.

Now for the backend, let's talk about Javascript first. I used JS for its Drag and Drop capabilities, as well as its ability to create and parse a JSON file. I chose to use Drag and Drop to help simplify the User Experience, it's much easier to just see all of the cards you can use, and also easier to manage what cards have and haven't been used, if they're just there on the site and can be moved around. And for the primary file of choice for the data sharing being a JSON file, it simply made sense as an easy way to store and then later parse data. JSON files are also quite easy to modify/create, so I worried about invalid JSONs being uploaded and stored, but they turned out to be quite easy to check. The small size of the file and general ease of use makes them a great fit for the main file type. My only qualm with JSON files is that you cannot use any of their functionalities on IOS, and when they are uploaded to Discord they display the internals of the file, which can confuse any who isn't in the know. But these are small issues compared to what the JSON file adds.

And finally, the PHP side of TTDB handled the storing of user cards, as well as the displaying of cards to users. All of our PHP is in Browse + Upload Decks, which I originally considered splitting into two sections, but then decided to keep together for simplicity. When you upload a deck, the filename and a total of 0 likes (which we will get into soon) are added. Then, when you wish to display these decks below, they are displayed by their deckID. Out of the tools I was allowed to use for this project, PHP was the best tool for the job. It was possible for us to just save the files to a folder and display them all, but PHP gives us much greater control. It would also be necessary if I wished to implement likes.

This smoothly transitions us to our next topic: things that are coming in the future. Most of this centres around the Browse section, which has the most room for expansion. I hope to enable likes and have PHP sort the decks by the number of likes, which would help the better decks swim to the top, while the lesser decks sink to the bottom. I would also like to implement a search bar in the browse section, which can help users find specific decks they are looking for. Additionally, I would love to allow users to view decks before they download them, or have some way of displaying the cards with the name and description. Finally for the Browse section, some tighter restrictions and “auto-moderation” to allow slurs or profane decks, or ones that aren't formatted properly, to be removed or not uploaded. In terms of other sections, it would be great to add a way to filter cards when building a deck (Weapon Cards, Sub Cards, Idol Cards, or perhaps by rarity or cost). Additionally, when this project is worked on more into the summer, I hope to be able to run ads on the site and use the revenue to buy my own domain and web server. Crowdfunding is another direction in which I could take this site in the future. But that will come much later in development, and the issue of monetization is still one to come.

Many of these future ideas also came from the extensive user testing we did during our development. Before browse and download were complete, I deployed the site to a temporary GitHub Pages account and got user feedback on what is good and bad about the site, and what may need to be added. We got this feedback by first asking for details about the individual (Age, Employment, History with Splatoon, etc). Then we asked them more direct questions, scaling from 1 to 10 how they felt about individual parts of the site. Then finally, I opened the floor for free response questions on what the user would specifically add to the site, or just written complaints and critiques about the site.

But overall, I learned a lot from this project. Not only did I learn tangible skills like HTML, CSS, JavaScript, PHP, UI/UX, and graphic design. I also learned less tangible things, like teamwork, responsibility, site deployment, and how to use user feedback.