SplatBase

SplatBase is an ongoing project that is a response to a growing issue within the Competitive Splatoon community.

In the competitive game splatoon, there are players who are free agents or players who are without a team. Players are also in the dark as to what weapons some members specialize in. Lastly, there are an ever-shifting number of teams that are disbanding, rebranding or are constantly changing their members. SplatBase’s goal is to update itself using user input on the community’s status over time.

SplatImport

To get starting data, I ran a survey that gathered several pieces of information. Data such as Username, Nintendo Network ID, proficiency in three weapons, whether they are a free agent or name of their team, name of their captain, and the past teams they have been on. The results of survey, which was done on google forms, was converted to a csv file and then parsed using csv-parser module. a and entered into a Postgres database.

SplatBack

SplatBack serves as the backend of SplatBase and houses both the server and the schema for the postgres tables.

In SplatDataBase.sql, We create several tables to handle the members, teams, weapons and a table to take care of the many to many relationship between member and weapons called memberhasweapon. Every member can have more than just one weapon so memberhasweapon remedies this situation by pairing each member by ID to each weapon by ID. The pairing of tables itself is stored into this table.

In SplatServer.js we establish our server for the SplatBase program. This file handles a series of HTTP GET requests that it receives from the react redux frontend action files, enters the corresponding route parameter and pulls up the correct information from the database based on the query it receives. These responses between the actions files, server and database allow the database to give new information to the front end on request.

SplatFront

Splatfront serves as the frontend of SplatBase and it is powered by React and Redux, focusing on making the website a single page application to ensure a pleasant user experience. it connects to the backend via the action files where it makes ajax requests, sending a url, method and jsonified data to the POST route parameters in the backend to add to the Database. if the user is searching something in the Database, they make a GET request to the backend’s route parameters via ajax calls to pull up information they wan according to their queries. The React website then adjusts the contents displayed onto the page according to what the user types in the search field, updating the contents in real-time. these changed values are then updated onto the store after the reducer applies the appropriate logic onto them. In Index.js, Display of the contents are managed by the component files which are wrapped by their containers. the containers are then housed in a JSX ReactRedux.Provider tag which houses all of the containers that SplatBase uses. This makes sure that all the containers have access to the store, made possible by the Provider, or the import of react-redux. Links are associated with each of the containers and made possible through import of'react-router'. Index.js Also houses all the reducers of the program and puts them in a mother of all reducers called reducers and it is applied to the store. App Layout is where we lay down the routing of the program.