Test Plan:

Test Type	Nature of Test	Example
Check that the program is website that can easily be accessed by any internet user	A person can enter a URL into a browser and find the website with the program	A person enters a copied URL into google and is taken to the website
Check that program has an easy way to input data types and amounts, and outputs easily understood results.	A person can choose a data type from a set list (Field Goals, Rebounds, etc.) and enter an amount into an input field, where it is then displayed in a table	Two connected bars are present on the website where one lists types of data, and the other can have an amount inputted. When both are filled, the pair are transported to a table recording the statistics
Check that the program can take basic statistics (rebounds, points) and calculate more complex information (effective field goal percentage, efficiency).	A person can input multiple types of data into the table, and the program can create new complex statistics without further work from the person.	A person inputs their three point and two point record, and the program takes the percentages to create their effective field goal percentage, then shows it in the table
Check that the program can take in multiple game stats over time and synthesize them into a player average over the course of a season.	A person can enter multiple records of statistics separated by game to form multiple rows on the table, including a bottom average row	A person inputs a 3/6 free throw record for one game, then a ⅝ record for another, creating two separate rows, along with a third average row with a 57% free throw average.
Check that the program can store separate data for multiple players at once	A person can switch between different player tables easily and alter them both independently without leaving/reloading the website	A person sets their assists for one game that was entered recently, then goes back to a previous game to fix a data entry mistake
Check that the program can store player information throughout multiple visits to the website.	A person can use the website with a personal login that allows them to save their data between visits	A person leaves the website once inputting some data, then returns later with their passcode to add on to the same data without needing to re-enter

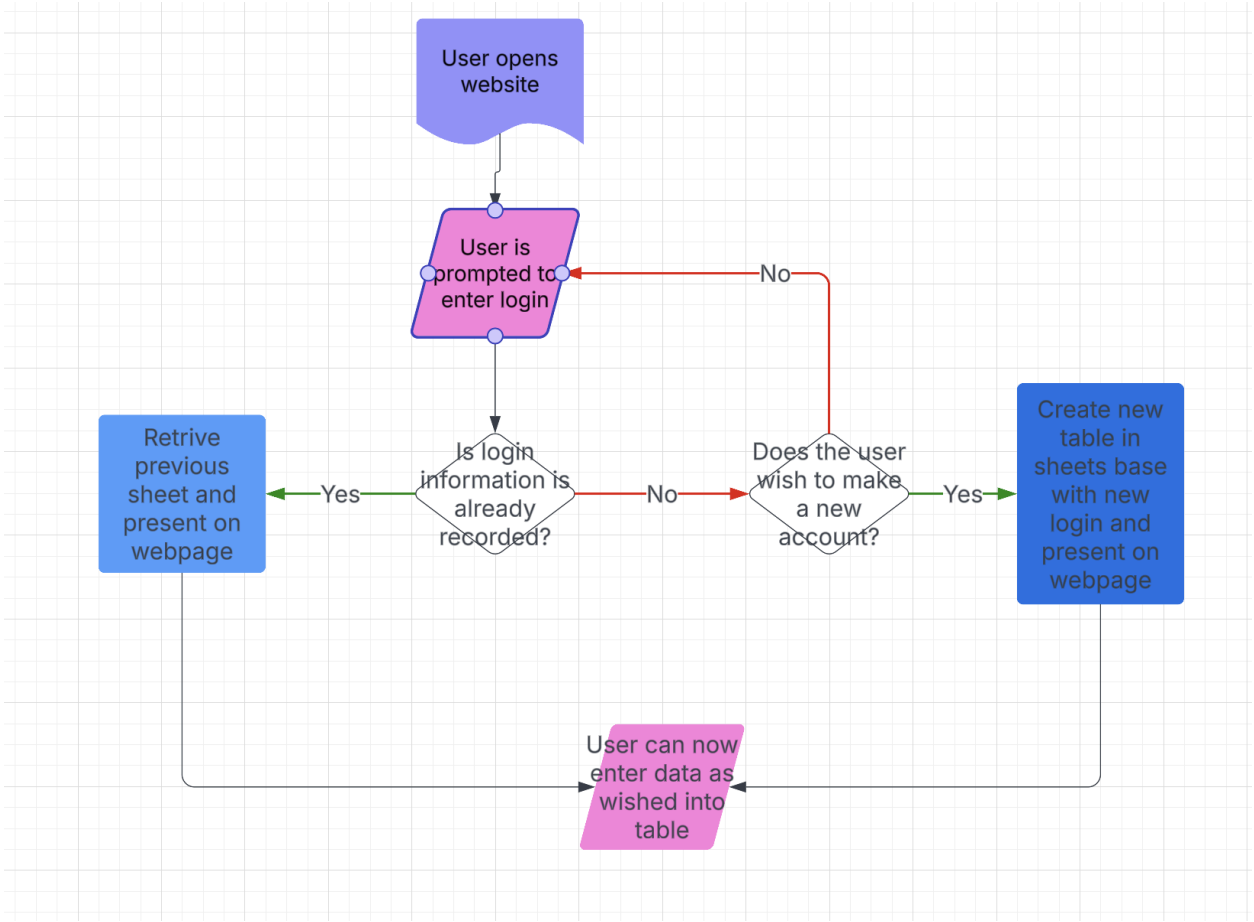
Overview of Design:

My website product will be a web app connected to Google Sheets that will be able to be accessed by any browser user through a sharable link, and have an opening form that will separate data between different players, while still keeping user information between site visits. The data will be separated by existing in different locations in the backend sheets.

The main data entry will be by form, where a user can choose a parameter (type of stat), and number, and enter them into a table that records the information over multiple games. The user can enter their simple data into the table, and the program will output more complex information through mathematical algorithms, as well as averages over multiple games once that data is created.

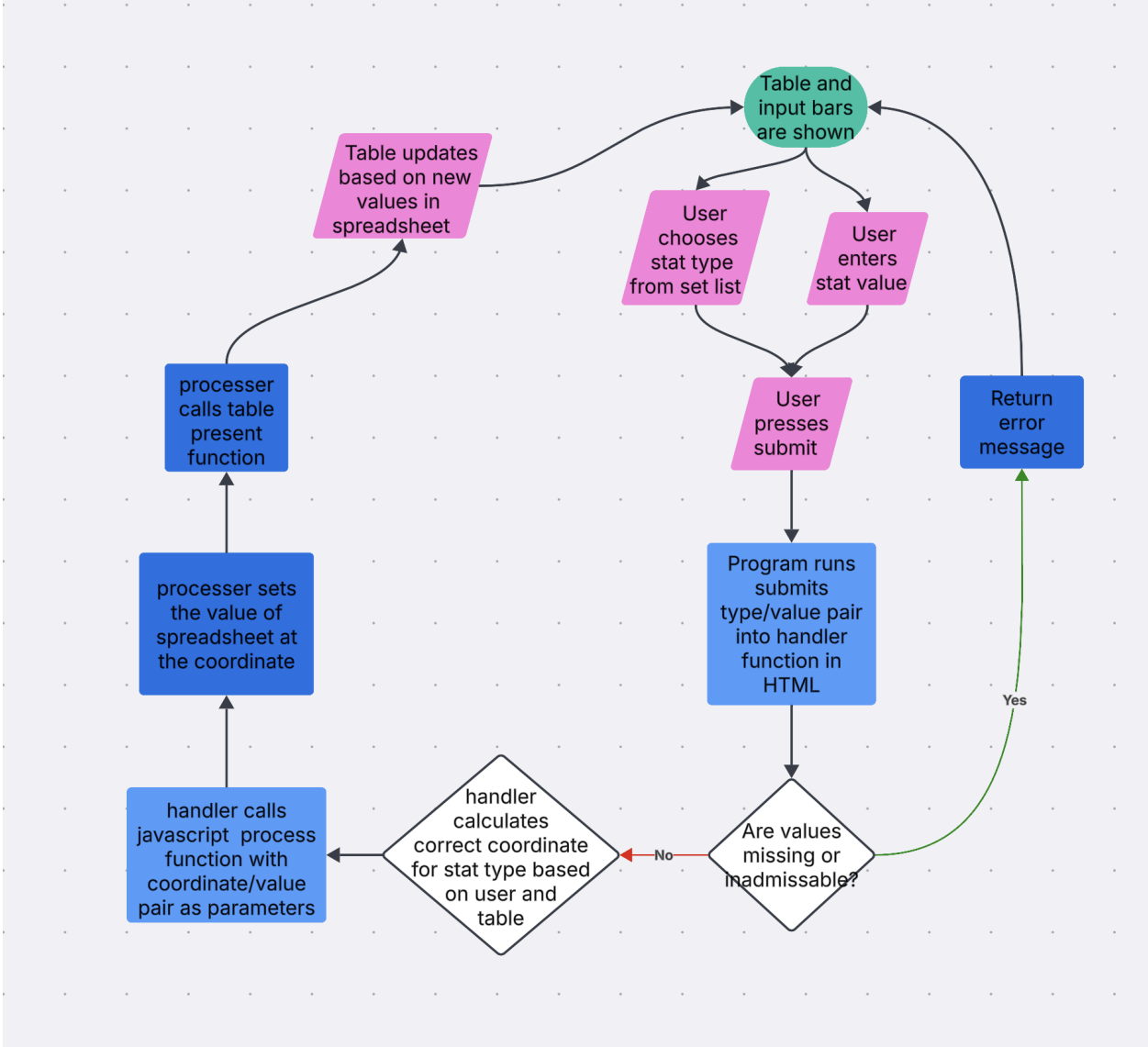
Flowcharts:

Flowchart of login process:



This flowchart demonstrates the planned login process for a website user, where upon loading the page, a login screen would be presented, and the user would enter a name/password and either be given the stat table associated with that login, or have a new sheet/player created if necessary. After both options, a table for a player would be available to be seen and altered as wished.

Flowchart of stat input process:



This flowchart demonstrates the planned process of data input for the user, and the resulting actions on the backend. Shown in green is the starting state of the page, pink represents actions seen by the user, and blue/white represents the backend code elements. Basically, inputting the wished data into a form sends the data into html, where it is confirmed to be usable (sending an error message if otherwise) and assigns coordinates and values to the inputs. The connected coordinates and values are then sent to javascript, where a called function sets the new values in the base sheets, and calls a second function once complete to take the sheet data and present it back to the user in a readable format.