Fantasy Football Idea:

* Have rpi to do api requests
* Rpi processes data
  + Can be storing records
  + Make predictions on who will win
* Rpi sends data to a server or sends emails to everyone (I think this is with MQTT)
* We can also get the rpi to generate an image with open ai api request

Website: <https://www.thesportsdb.com/season/4391-nfl/2024>

Espn api: <https://github.com/pseudo-r/Public-ESPN-API>

Final ranking: <https://www.espn.com/nfl/fpi/_/view/projections>

View past snapshots with <https://archive.org/>

Idea:

My friend has a fantasy football league. Each player chooses three teams and ‘owns’ them. The player’s record is the sum of their team’s record. Each week, my friend has to enter the stats into an excel sheet. My idea is to create a server, which handles this for him.

The server is my rpi; it will run a restAPI to hold a database and make outgoing calls. Clients can be anything that can make http calls. We will use my laptop to do so. The client will post initial data (team combinations). The server will store this. Once the client requests an update, the server will use other api requests to gather stats. The server will also run a regression model to see the odds of each player winning. The client can request weekly updates as well.

Rpi posts to a website. Client can click a button to choose which stats to display or to update

Explain why we are using the rpi. Like a buzzer or light. The rpi can make a sounds each time. We just have to justify using the rpi rather than only our computer.

* **Introduction**
* Fantasy‑league commissioners need a clear, data‑driven way to see which manager is most likely to finish atop the standings—and to share those insights automatically. In our system, each league participant manages **three NFL teams**, and their **aggregate record** (sum of wins, losses, ties across all three) determines their rank. We’ll build a pipeline that **collects weekly NFL team results**, computes each manager’s aggregate record in **pandas**, trains a simple ML model to forecast final aggregate standings, and **pushes** those forecasts out via email, MQTT topics, or a Grafana dashboard.
* **Objective**
* **Gather** weekly NFL game outcomes and update each manager’s three‑team aggregate record.
* **Store** historical and current aggregate records in a pandas DataFrame for analysis.
* **Train & predict** using a classification/regression model (e.g., Random Forest or Logistic Regression) to estimate each manager’s chance of finishing with the best record.
* **Notify** league participants of their projected standings and win‑probabilities through one or more channels (email, MQTT, Grafana panels).
* **Data Pipeline & Components**

Laptop is server

RPI collects data and makes buzzer sound for an update

Other laptops and phones can access the url

I should just host the webpage to get hub idk but then the rpi posts the data to the webpage