The Picks Inn - PRD

Purpose; the product, who it's for and why you're building it.

The purpose of this product is to enable **NBA fans** to see how good they are at scouting NBA prospects. This is a product for a niche segment of basketball fans who care deeply about player scouting. We're building it because it'd be super fun to measure how effective we are at identifying NBA caliber talent and comparing our scouting performance against (a) the experts in the field (Sam Vecenie @The Athletic, Jonathan Givoney @ESPN, Jeremy Woo @Sports Illustrated, etc.) and (b) our friends.

We intend to build a product that is simple yet informative and, most importantly, fun for users. We will collect user submitted data and data from the draft experts, then merge it with actual player data to produce findings; eg individual user performance, performance vs peers, performance vs experts, and more. The arrangement of all data sources will be hard for individual users to reproduce on their own, let alone compare with others, so we'll do it for them - the passionate community of NBA draft fans.

We're fired up to make it easy for users to find our website. We're fired up to make it super fun to submit a big board. We're fired up to present big board submissions in fun and illustrative ways. We're fired up to serve the community of NBA draft fans. We're fired up to be **The Picks Inn**.

Features; what you're going to build..

The product will consist of five components:

1. Big board; user submits a list of prospects (ie a "big board") via Google Form
2. Player Data; player performance data pulled in via web scraper (BasketballReference)
3. Scouting performance; measure accuracy of user submitted big board vs player performance data and then visualize in a table on web app
4. Leaderboard; compare performance of user submitted big boards vs other user submitted big boards and then visualize in a table on web app
5. DIscovery; enable user to view other user submitted big boards

MVP / Version 1.0; Goals for the release (e.g., functionality).

The MVP should feature a web application that enables the user to sign-in with Google, submit a big board via Google Forms, view/modify the big board, share the big board and view performance of user submitted big board vs other boards.

Component #1: Big board
Goal: User can access a form to submit his/her ranking of the top 20 players in a given NBA draft class

- [ ] Feature #2: Submit a big board via Google Forms
- [ ] Feature #3: View/Modify big board
  - [ ] *Modifications to big board expire at specified date
- [ ] Feature #4: Share big board
  - [ ] User must be able to share big board via SMS or email, post via Twitter or download image file of big board (png, jpeg)

Component #2: Player Data
Goal: User can view an entire draft class ranked by actual player data via select advanced metrics (we will use only "PER" statistic in MVP)

- [ ] Feature #1: Pull player performance data from Stathead or BasketballReference (Player Name, Team, PER)
- [ ] Feature #2: Expose player performance data in web app via table
- [ ] Feature #3: User must be able to sort data in table by any column
- [ ] Feature #4: User must be able to select draft class year and table must update automatically
- [ ] Feature #5: Table must only include the following data; Rank, Player Name, Team, PER
  - [ ] "Rank" = Highest PER to lowest PER of all rookies (auto-descending order)

Actual Top 20					
Draft	**Rank**	**Player Name**	**Team**	**Draft pick**	**PER**
2022	1	Walker Kessler	UTA	22	2.5
2022	2	Tari Eason	HOU	17	1.7
2022	3	Dyson Daniels	CLE	8	0.4
2022	4	Kevon Harris	TOR	Undrafted	0.3
2022		...	...		...
2022	20	Nikola Jovic	LAC	27	-4.8

Component #3: Scouting Performance

Goal: User can view the accuracy of his/her board compared to actual player data via select metrics in a table; correct picks, big board accuracy, sum PER, PER %

Correct picks

- [ ] Feature #1: Count the number of correct picks in a user submitted board vs actual Top 20 players by PER
- [ ] Feature #2: Expose the number of correct picks in Scouting Performance table (below)

Big board accuracy

- [ ] Feature #1: Calculate big board accuracy by dividing the total number of correct picks in user submitted board by twenty;
    - [ ] Accuracy = (# of correct picks / 20)*100
- [ ] Feature #2: Expose big board accuracy in Scouting Performance table (below)

Sum PER

- [ ] Feature #1: Calculate the Sum the PER of all players in a user submitted board by totalling the PER of each player in a big board
    - [ ] Sum PER_User = {UserSelectedPlayerName1_PER} + [UserSelectedPlayerName2_PER}... + {UserSelectedPlayerName20_PER}
- [ ] Feature #2: Calculate the Sum the PER of Top 20 players per PER via actual player data (ie “Actual Top 20”)
    - [ ] Sum PER_Actual = {ActualPlayerName1_PER} + {ActualPlayerName2_PER}... + {ActualPlayerName20_PER}
- [ ] Feature #3: Expose sum PER in Scouting Performance table (below)

PER %

- [ ] Feature #1: Calculate the PER% by dividing the sum PER of user submitted board vs. sum PER of Actual Top 20
    - [ ] Sum PER = (User submitted big board Sum PER / Actual Top 20 Sum PER)*100
- [ ] Feature #2: Expose PER % in Scouting Performance table (below)

Select user	**OochieDad**

Scouting Performance								
Metric	**2020**	**[Rank]**	**2021**	**[Rank]**	**2022**	**[Rank]**	**2023**	**[Rank]**
Correct picks	*16*	37th	12	T-128th	18	...	*15*	...
Accuracy	80%	32nd	60%	T-128th	90%	...	75%	...
Sum PER	18.23	30th	16.43	115th	19.85	...	14.32	...

PER%	82.23%	30th	69.46%	111th	91%	...	79%	...

Component #4: Leaderboard

Goal: User can view the accuracy of his/her board vs. other user submitted boards

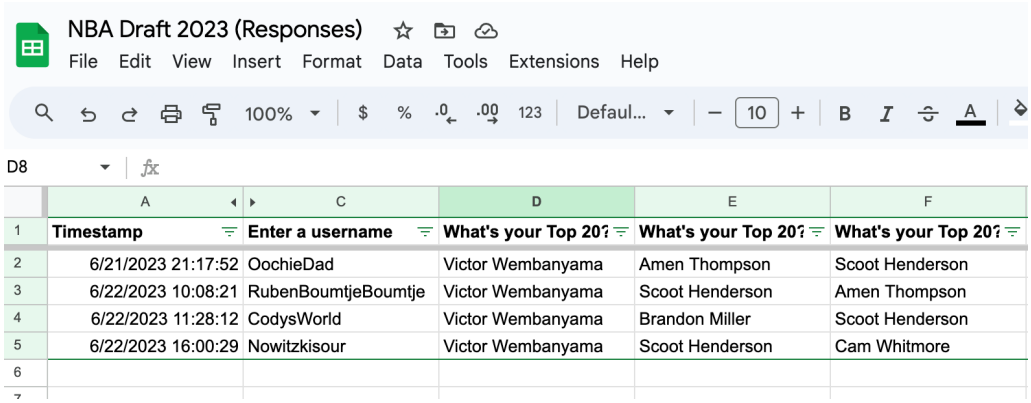
- [ ] Feature #1: Enable to user to view a table of all user submitted Big Boards arranged from most accurate to least accurate that updates dynamically (sample below)
  - [ ] User must be able to sort by *Picks correct*, *Accuracy*, *Sum PER* & *PER%*
- [ ] Feature #2: Enable user to select a given draft year and table updates automatically

(Sample) Leaderboard						
Draft Year	**Rank**	**Username**	**Picks correct**	**Accuracy**	**Sum PER**	**PER%**
2022	*0*	*Actual*	*20*	*100%*	*16.23*	*100%*
2022	1	Blancogrande	16	80%		
2022	2	OochieDad	14	70%		
2022	3	Kevin O'Connor_TheRinger	10	50%		
...	...	...		...		
2022	X	Sam Vecenie_The Athletic	4	20%		

Component #5: Discovery

Goal: User can view big boards submitted by other users.

- [ ] Feature #1: Enable user to access other user submitted big boards in Google Sheets
  - [ ] Google Sheet should be a modified copy of the Google Form Responses sheet
- [ ] Feature #2: Enable user to sort + filter Username column
- [ ] Feature #3: Google Sheet must be protected
- [ ] Feature #4: Google Sheet must only be "Viewable" by others



Technical details

Overview

- The application will feature a backend graphql API and a frontend web application
- The frontend and backend will communicate using GraphQL
- Data will be stored in a postgres database
- The backend will be deployed in a typical cloud provider via a docker container
- The frontend will be deployed on a web app hosting provider like Vercel or Netlify

Backend

- The backend will be written in python
    - It should use python version 3.11
    - It should make use of python type hints
    - It should use poetry for dependency management
    - It should automatically format code using black
- The backend will be implemented using the django web framework
    - It should use django 4.2 (the latest LTS version of django)
    - It should use graphene for graphql
    - It should use JWT based authentication

Frontend

- The frontend will be written in typescript
    - It should use the latest version of typescript (v5)
- The frontend will be build using the latest version of React
- The frontend will be build using Material UI
- The frontend should be built using next.js
- It should feature Single Sign-On using next-auth
    - It should implement Google as the default and only sign-on provider
- It should use Apollo client to communicate with the backend
    - The Apollo client should attach the next-auth provided JWT in graphql requests so that the frontend can authenticate with the backend