FooDraft

Target release	28 Jun 2020
Epic	
Document status	**1.6.2.**
Developer	@ Daniel Norris

Objective

Develop an app with an MVP that enables the user to randomly pick a 5-a-side football team from a list of 10 names.

Further stretch goals could be implemented if time allowed.

Installation

Installing the app locally

> This guide assumes that you have **node.js** and **npm** installed. Installing these dependencies are out of scope for this guide but you can refer to the npm docs on how to install these dependencies here.

The app can be installed locally onto your machine by using the following steps.

First, clone the remote repository to your local machine and name the target directory.

```
git clone git@github.com:daniel-norris/football_app.git <dir>
```

Then cd into the project directory.

```
cd <dir>
```

Install the dependencies required for the app.

```
npm i
```

Run the app. When this runs you will redirected to http://localhost:3000/#/.

```
npm start
```

Installing the API locally

> This guide assumes that you have both **PHP**, **Composer**, **Vagrant** and **MySQL** installed. Installation is out of scope for this guide but you can refer to their docs on how to install.

First, clone the remote repository and name the target directory.

```
git clone git@github.com:daniel-norris/football_api.git <dir>
```

Change directory into folder.

```
cd <dir>
```

Run Composer to install dependencies.

```
composer install
```

Make a `homestead.yaml` file.

```
vendor/bin/homestead make
```

Copy the example environments variable file and create a new one.

```
cp .env.example .env
```

Then open the .env file and change the following fields according to your MySQL installation.

```
nano .env

...

DB_DATABASE=<DB name>
DB_USERNAME=<DB username>
DB_PASSWORD=<DB password>
```

Change `homestead.yaml` memory to `512`.

Then spin up a new Vagrant VM.

```
vagrant up
```

SSH into vagrant, generate a new key and migrate the tables.

```
vagrant ssh
cd code
run artisan key:generate
run artisan migrate
```

If you choose to run **both** the app and API locally, then you will need to point the app away from AWS and towards your local Vagrant VM.

Point your local app towards the local instance of your API.

Create an `.env` file in your **app directory** using the instructions below.

Mac OS

You need to point the React app towards the appropriate URL. You can edit this in the `.env` file in the root directory.

```
REACT_APP_API_URL=http://<laravel url here>/api/
```

Then update your `.axios` file in the root directory.

```
baseURL: process.env.REACT_APP_API_URL,
```

Win OS

You need to point the React app towards the appropriate URL. You can edit this in the `.env` file in the root directory.

```
REACT_APP_API_URL=http://localhost:8000/api/
```

Then update your `.axios` file in the root directory.

```
baseURL: process.env.REACT_APP_API_URL,
```

Releases

21 Jun 2020 | v 0.1.0.

The initial release was planned on the Sunday before the week started. Ironically to avoid wasting any time but this plan would later be shelved over concerns about meeting time deadlines. You can review the original plan below.

Click here to view the original plan.

v 0.1.0

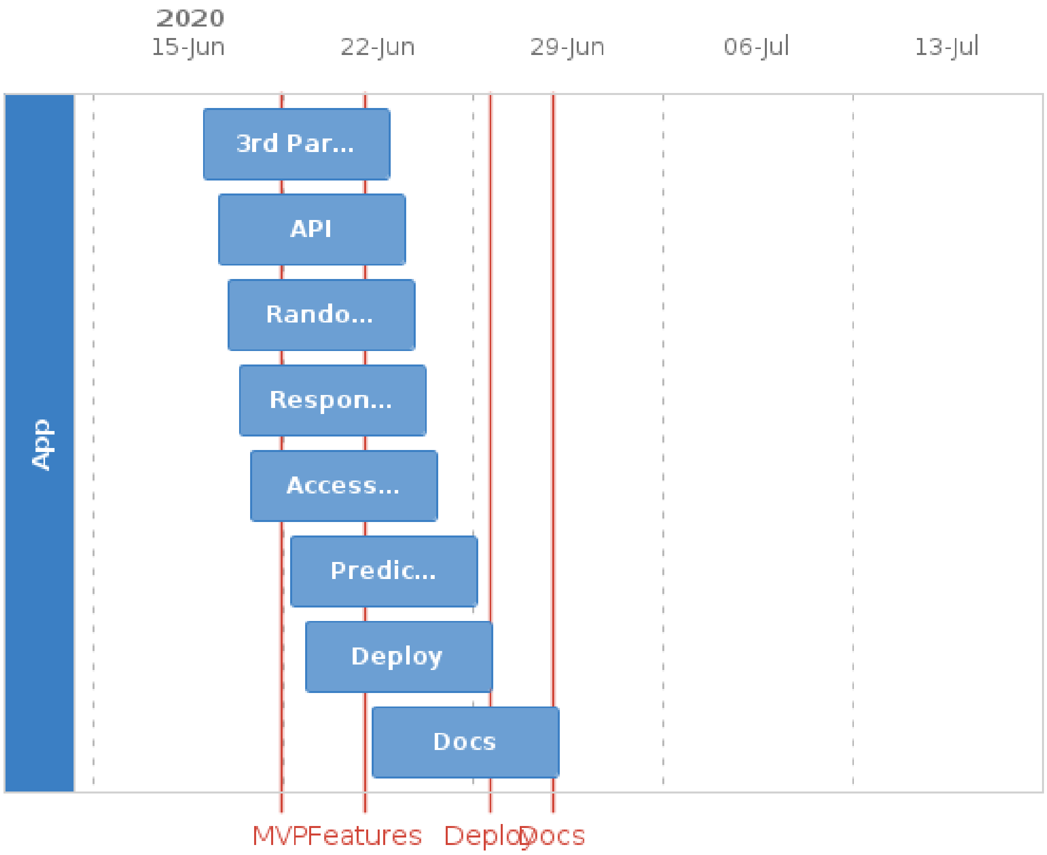
Technology Stack

- Laravel RESTful API to enable the user to review game history and reload
- React and Redux app to enable modular component development and better state management for immediate and future releases
- AWS deployed for a scalable cloud hosted solution

Assumptions

- The app must be deployed and accessible online
- Most users will access this app from either a mobile device, tablet or web browser
- Users will not know what the app is intended to do

Milestones



Requirements

A real focus was on ensuring that I identified user stories that were critical to ensuring a viable MVP first and then identify requirements that could be later added in future releases.

This initial plan aimed to consume a (3rd Party API) and also create a RESTful API using Laravel to store game history and later win / loss history. The app would enable the user to choose premiership teams and players from any team from the 2018/19 season, enabling them to roster their own fantasy football squad for their favourite team.

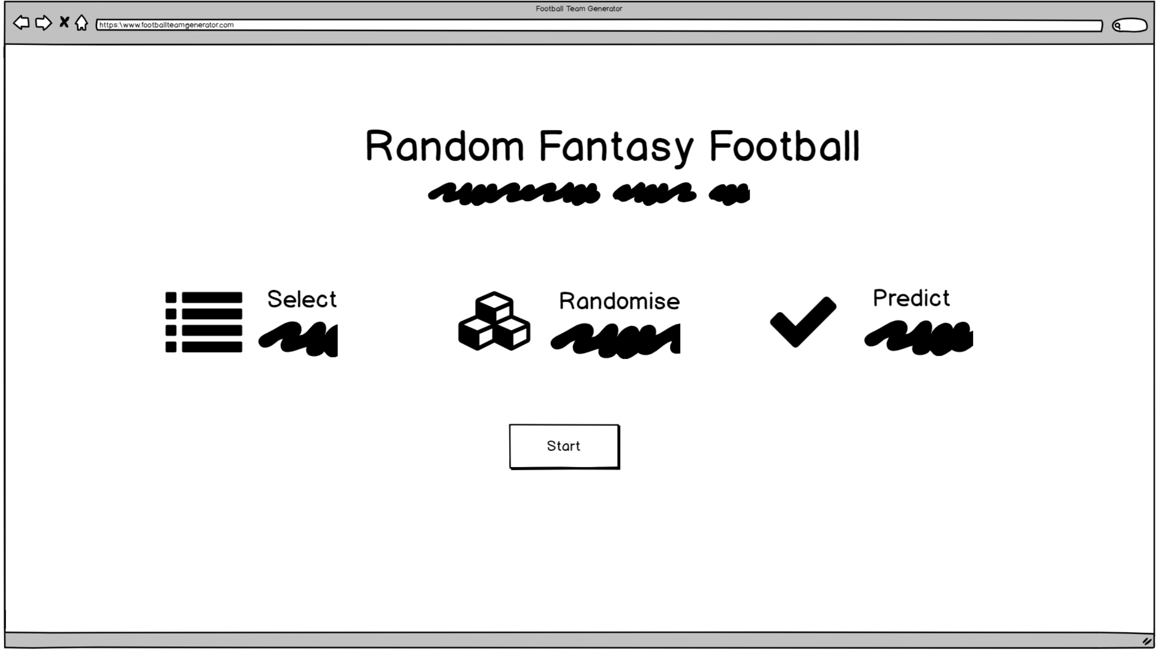
The API would provide a range of data including saves, assists and scoring probability, which would provide great data to use to create a win probability feature and algorithm.

The user would then be able to review their previous drafts and their win / loss history by accessing the RESTful API created using Laravel. Later releases would enable the user to login to access this functionality and provide additional features to tailor their experience while on the app.

	Requirement	User Story	Importance	Notes
1	API Integration	A developer wants to integrate with an API so that they can view their randomised team and roster.	MUST HAVE	
2	3rd Party API Integration	A user wants to choose premiership players and teams.	MUST HAVE	
3	Responsive	A user wants to access the app on their mobile, table and desktop.	MUST HAVE	
4	Accessible	A user can access the app when using a screen reader.	MUST HAVE	
5	Randomise	A user wants to randomise their selected roster into two teams.	MUST HAVE	
6	Predict Winner	A user wants to determine who will win out of the two randomised teams.	SHOULD HAVE	
7	Players Per Side	A user wants to determine how large the teams will be.	SHOULD HAVE	
8	Custom Team Creation	A user wants to create custom teams.	MAY HAVE	
9	Custom Player Creation	A user wants to create custom players.	MAY HAVE	
10	User Login	A user wants to login to review drafts made in a previous session.	MAY HAVE	
11	User Login	A user wants to review the win / loss history of a draft.	MAY HAVE	
12	Team History	A user wants to review previous teams they have created in the same session.	MAY HAVE	

Wireframes and UX

The wireframes below include both the initial plan and the later revised plan.



API Routes

The original plan included consuming a 3rd party API and creating a RESTful API using Laravel.

Football Data API

GET /https://api.footystats.org/league-players?key=YOURKEY&season_id=*

```
{
    success: true,
    data: [
            {
                id: 2984,
                competition_id: 161,
                full_name: "Petr ech",
                first_name: "Petr",
                last_name: "ech",
                known_as: "Petr ech",
                shorthand: "petr-cech",
                age: 36,
                league: "Premier League",
                league_type: "Domestic League",
                season: "2017/2018",
                starting_year: "2017",
                ending_year: "2018",
                url: "/players/czech-republic/petr-cech",
                club_team_id: 59,
                club_team_2_id: -1,
                national_team_id: -1,
                position: "Goalkeeper",
                minutes_played_overall: 3040,
                minutes_played_home: 1510,
                minutes_played_away: 1530,
                birthday: 390726000,
                nationality: "Czech Republic",
                continent: "eu",
                appearances_overall: 34,
                appearances_home: 17,
                appearances_away: 17,
                goals_overall: 0,
            }
          ]
}
```

GET /https://api.footystats.org/league-teams?key=YOURKEY&season_id=*&include=stats

```
{
  {
    success: true,
    pager: {
    page: 1,
    per_page: 50,
    total_results: 20
  },
    data: [
              {
              id: 59,
              original_id: 59,
              name: "Arsenal FC",
              cleanName: "Arsenal",
              english_name: "Arsenal FC",
              shortHand: "arsenal-fc",
              country: "England",
              continent: "",
              image: "https://cdn.footystats.org/img/teams/england-arsenal-fc.png",
              flag_element: null,
              season: "2018/2019",
              seasonClean: null,
              url: "https://footystats.org/clubs/england/arsenal-fc",
              stadium_name: "",
              stadium_address: "",
              table_position: 5,
              performance_rank: 5,
              risk: 87,
              season_format: "Domestic League",
              competition_id: 1625,
              founded: "1886",
              full_name: "Arsenal FC",
              alt_names: [ ],
              official_sites: [ ]
              },
          ]
}
```

Laravel RESTful API

GET /games

```
{
    "data": [
        {
            "id": 1,
            "players_per_side": 5,
            "winner": "",
            "team_1": {
                "id": 1,
                "name": "Asdf",
                "players": [
                    {
                        "id": 4020,
                        "first_name": "Juan Manuel",
                        "last_name": "Mata García",
                        "age": 31,
                        "height": 170,
                        "weight": 63,
                        "club_team_id": 149,
                        "position": "Midfielder",
                        "nationality": "Spain",
                        "goals_overall": 3,
                        "goals_involved_per_90_overall": 0.35,
                        "assists_per_90_overall": 0.14,
                        "goals_per_90_overall": 0.21,
                    },
                    {
                        "id": 4020,
                        "first_name": "Juan Manuel",
                        "last_name": "Mata García",
                        "age": 31,
                        "height": 170,
                        "weight": 63,
                        "club_team_id": 149,
                        "position": "Midfielder",
                        "nationality": "Spain",
                        "goals_overall": 3,
                        "goals_involved_per_90_overall": 0.35,
                        "assists_per_90_overall": 0.14,
                        "goals_per_90_overall": 0.21,
                    },
                [
            },
            "team_2": {
                "id": 1,
                "name": "Asdf",
                "players": [
                    {
                        "id": 4020,
                        "first_name": "Juan Manuel",
                        "last_name": "Mata García",
                        "age": 31,
                        "height": 170,
                        "weight": 63,
                        "club_team_id": 149,
                        "position": "Midfielder",
                        "nationality": "Spain",
                        "goals_overall": 3,
                        "goals_involved_per_90_overall": 0.35,
                        "assists_per_90_overall": 0.14,
                        "goals_per_90_overall": 0.21,
                    },
                    {
                        "id": 4020,
                        "first_name": "Juan Manuel",
                        "last_name": "Mata García",
                        "age": 31,
                        "height": 170,
                        "weight": 63,
                        "club_team_id": 149,
                        "position": "Midfielder",
                        "nationality": "Spain",
                        "goals_overall": 3,
                        "goals_involved_per_90_overall": 0.35,
                        "assists_per_90_overall": 0.14,
                        "goals_per_90_overall": 0.21,
                    },
                [
            },
        },
        ]
}
```

GET /games/1

As above just returns a single game.

POST /players

```
{
  "players": [
        {
            "id": 4124,
            "first_name": "Juan Manuel",
            "last_name": "Mata García",
            "age": 31,
            "height": 170,
            "weight": 63,
            "club_team_id": 149,
            "position": "Midfielder",
            "nationality": "Spain",
            "goals_overall": 3,
            "goals_involved_per_90_overall": 0.35,
            "assists_per_90_overall": 0.14,
            "goals_per_90_overall": 0.21,
      },
  ]
```

POST /games

```
{
    "id": 1,
    "players_per_side": 5,
    "team_1": {
        "id": 52,
        "name": "Liverpool",
        "image": "https://cdn.footystats.org/img/teams/liverpool.png",
    }
    "team_2": {
        "id": 23,
        "name": "West Ham",
        "image": "https://cdn.footystats.org/img/teams/westham.png",
    }
}
```

22 Jun 2020 | v 0.2.0.

Production version plan is included here.

On Monday morning, I later revised the original plan due to concerns about time constraints and database complexity and opted for a “slimmed” down version which I was confident I could deploy in time and with time to spare. Importantly, the free time provided contingency for unforeseen challenges which the earlier plan definitely would not have done.

Click here to see the production plan.

v 0.2.0

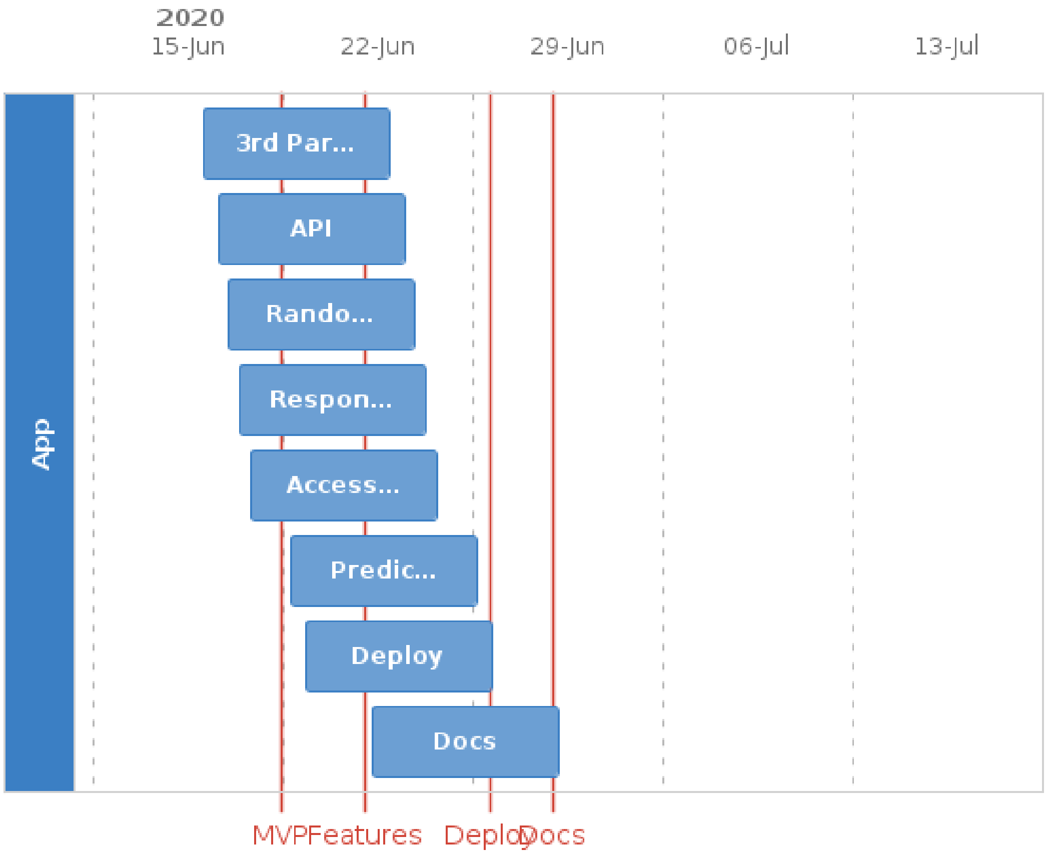
Technology Stack

- Laravel RESTful API to enable the user to review game history and reload
- React and Redux app to enable modular component development and better state management for immediate and future releases
- AWS deployed for a scalable cloud hosted solution

Assumptions

- The app must be deployed and accessible online
- Most users will access this app from either a mobile device, tablet or web browser
- Users will not know what the app is intended to do

Milestones



Requirements

A real focus was on ensuring that I identified user stories that were critical to ensuring a viable MVP first and then identify requirements that could be later added in future releases.

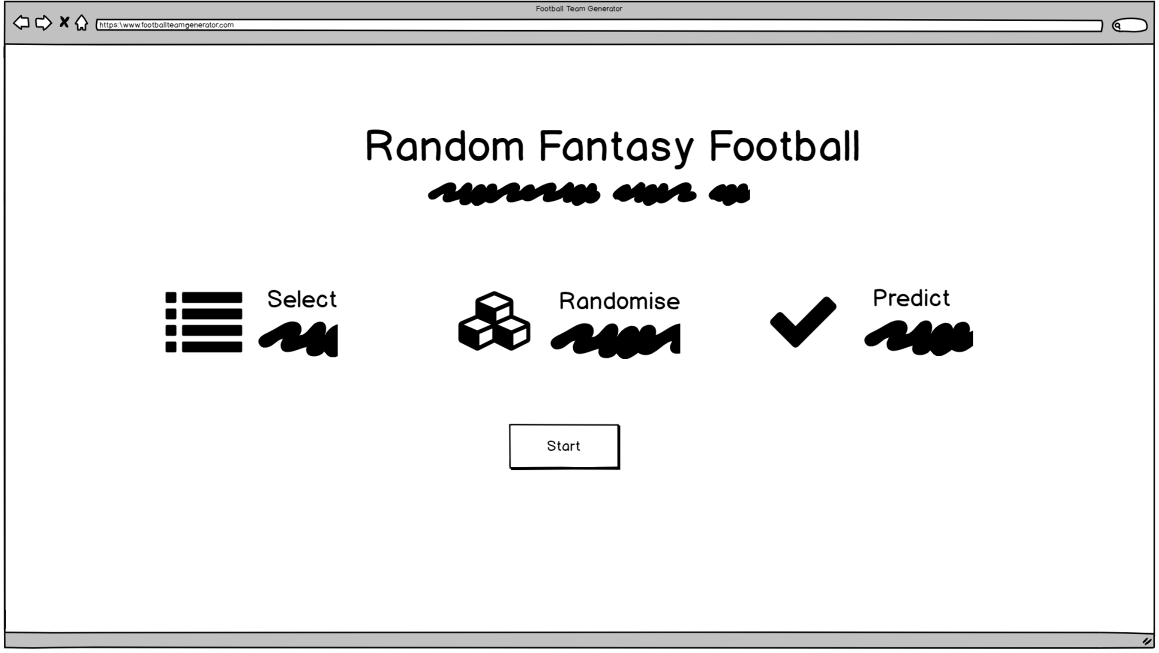
The user starts on the landing page and the app provides a brief overview of what functionality the app provides. The user can then create custom football team names, select team kit preferences, sizes and colours. Then the user can create team players and select attributes including skill, name and position.

The user can then randomise their team rosters and is presented with two randomised teams. Later releases plan to also predict a winner based on the teams skill rating difference and provide a full game history for the user.

	Requirement	User Story	Importance	Notes
1	API Integration	A developer wants to integrate with an API so that they can view their randomised team and roster.	MUST HAVE	
2	Responsive	A user wants to access the app on their mobile, table and desktop.	MUST HAVE	
3	Accessible	A user can access the app when using a screen reader.	MUST HAVE	
4	Randomise	A user wants to randomise their selected roster into two teams.	MUST HAVE	
5	Custom Team Creation	A user wants to create custom teams.	MUST HAVE	
6	Custom Player Creation	A user wants to create custom players.	MUST HAVE	
7	Predict Winner	A user wants to determine who will win out of the two randomised teams.	SHOULD HAVE	
8	Players Per Side	A user wants to determine how large the teams will be.	SHOULD HAVE	
9	User Login	A user wants to login to review drafts made in a previous session.	MAY HAVE	
10	User Login	A user wants to review the win / loss history of a draft.	MAY HAVE	
11	Team History	A user wants to review previous teams they have created in the same session.	MAY HAVE	

Wireframes and UX

The wireframes below include both the initial plan and the later revised plan.



API Routes

Laravel RESTful API

GET /games

Returns all game data available on the API. This was developed with the intention to implement a history feature.

```
{
    "data": {
        "id": 60,
        "players_per_side": 2,
        "winner": 0,
        "team_1": {
            "id": 119,
            "name": "Asdf",
            "players": [
                {
                    "id": 805,
                    "first_name": "Asdf",
                    "last_name": "Asdf",
                    "full_name": "Asdf Asdf",
                    "skill": "3",
                    "age": 21,
                    "position": "Forward"
                },
                {
                    "id": 806,
                    "first_name": "Asdf",
                    "last_name": "Asdf",
                    "full_name": "Asdf Asdf",
                    "skill": "5",
                    "age": 21,
                    "position": "Forward"
                },
            ]
        },
        "team_2": {
            "id": 120,
            "name": "Asdf",
            "players": [
                {
                    "id": 803,
                    "first_name": "Asdf",
                    "last_name": "Asdf",
                    "full_name": "Asdf Asdf",
                    "skill": "5",
                    "age": 21,
                    "position": "Forward"
                },
                {
                    "id": 804,
                    "first_name": "Asdf",
                    "last_name": "Asdf",
                    "full_name": "Asdf Asdf",
                    "skill": "1",
                    "age": 21,
                    "position": "Forward"
                },
            ]
        }
    }
}
```

GET /games/1

As above just returns a single game.

POST /players

```
{
    "players": [
        {
            "first": "Bill",
            "last": "Smith",
            "age": 23,
            "skill": 1,
            "height": 173,
            "position": "Forward",
            "team_id": 2
        },
        {
            "first": "Jim",
            "last": "Smith",
            "age": 23,
            "skill": 1,
            "height": 173,
            "position": "Forward",
            "team_id": 1
        }
    ]
}
```

POST /games

```
{
    "players_side": 5,
    "team_1": "test1",
    "team_2": "test2"
}
```

Database Structure

Game

Field	Type	Null	Key	Default	Extra
id	bigint(20) unsigned	no	pri	null	auto_increment
players_side	int(11)	no		null	
created_at	timestamp	yes		null	
updated_at	timestamp	yes		null	

Player

Field	Type	Null	Key	Default	Extra
id	bigint(20) unsigned	no	pri	null	auto_increment
first_name	varchar(100)	no		null	
last_name	varchar(100)	no		null	
age	int(11)	no		null	
height	int(11)	no		null	
skill	enum('1', ‘2', ‘3', ‘4’, '5’)	no		null	
position	varchar(30)	no		null	
team_id	bigint(20) unsigned	no	mul	null	
created_at	timestamp	yes		null	
updated_at	timestamp	yes		null	

Team

Field	Type	Null	Key	Default	Extra
id	bigint(20) unsigned	no	pri	null	auto_increment
name	varchar(3)	no		null	
created_at	timestamp	yes		null	
updated_at	timestamp	yes		null	
game_id	bigint(20) unsigned	no	mul	null	

24 Jun 2020 | v 1.0.0.

- App provides the user the ability to create teams and players and randomise functionality with an API.

25 Jun 2020 | v 1.0.2.

- Fixing cross-browser compatibility for custom form inputs including select picker.

25 Jun 2020 | v 1.2.0.

- Improved the original player field concept and implemented customisable marker feature.
- Marker now iterates over nth players per side.

26 Jun 2020 | v 1.2.2.

- Fixed a bug that was causing the timing of asynchronous GET /game and POST/players to render without specific data

26 Jun 2020 | v 1.6.0.

- Updated the API to include a GET /game/1 route for specific game data.
- User testing of app led to changes to layout and validation logic on form inputs for better UX.
- Added a tally feature to player creation.
- Removing the player attribute height from the app and database.
- Removed team kit feature from team creation.

27 Jun 2020 | v 1.6.2.

- Fixed bug causing footer to move off the bottom of the screen on some different aspect ratios.
- Improved unit testing across the board.
- Fixed deployment issue with Mockery dependency.

TBC | v 2.0.0.

- Implement 3rd party API integration to enable user to pick premiership teams and players.
- Enable user to pick between creating a custom team and premiership teams above.
- Implement user authentication and authorisation.
- Provide user game history and win / loss history.
- Implement a style guide and documentation including props for the app using Storybook.
- Improve the UX / UI:
    - Implement CSS transform and opacity keyframe animation to the main elements and field markers on app navigation for better visual experience.
    - Introduce A / B user testing.

TBC | v 3.0.0.

- Monetise the app by either:
    - Introducing affiliate links enabling users to place similar predictions with affiliated betting partners
    - Introducing affiliate links enabling users to purchase merchandise from affiliated premiership football clubs
    - Introducing a subscription system…

Out of Scope

- Implement a Phaser.js side scrolling game. Enabling users to play as a player from one of the teams they have created.
- Implement WebGL using a library like three.js to create a 3D version of the field which turns according to the part of the team or player creation process the user is currently viewing.