Introduction

The idea of the test is to see how you think about building components and applications with React Native or ReactJS. It also shows your coding style and the way you solve problems. Would be nice to see the code in Typescript, but it's not mandatory.

Description

We want you to **fetch data and then manipulate it and eventually render it** on a screen. The data is coming from a free to use API that you have to access the endpoints and treat it in order to show the correct data.

We want you to implement a mobile app where we can see all the free-to-play games (coming from the API) shown in a list. This app needs to first have a **main screen, with a fixed header on top and a list of the games below it**, that can be scrolled.

The way that you will implement the UI is totally up to you, as long as it has in the main screen the header, and the header has a title, that could be the name of the app, and also, the options to use some filters and sorting options.

In the filter options, the user needs to be able to **filter the games by platform and category**. In the sorting options, the player needs to be able to **sort by popularity, relevance or alphabetical order**.

The **endpoints** that you need is:

Description	Method	Endpoint
Live games list	GET	https://www.freetogame.com/api/games

For this endpoint, the **URL parameters** to filter and to sort are:

Description	Parameter	Values
Games by platform	platform	pc, browser or all
Games by category	category	mmorpg, shooter, strategy, action, racing, sports, mmo, survival or social
Sort games by release date, alphabetical or relevance	sort-by	release-date, popularity, alphabetical or relevance

API source: https://www.freetogame.com/api-doc

Notes

While sending requests, there are some rate limits, so please **avoid sending more than 4 requests per second**.

The possible code responses are:

- 200: Success
- 404: Object not found: Game or endpoint not found
- 500: Something wrong on the API's end (unexpected server errors)

While creating the UI, the list should show at least the following details of each game: name, thumbnail, platform and category.

After finishing the project, please **upload it to a public repository** like github or gitlab and send us the link.

Please also create a "Readme" file in the root of the file with the **instructions on how to proceed to setup the project locally** so we can test it.

Solve the extra problem

I want to know how you would solve some common problems we have here in our daily work. This is another topic, an isolated issue. So below you need to write a snippet that solves the following problem:

> I have one object that represents the progress stats for each player with mission related data, where each mission belongs to a game.
>
> a. This object has the key/values where the key is the mission ID and for each key the value properties are: gameId, playTime, wins and fails.
> b. The properties playTime, wins and fails are numbers. And the gameId property is a string that represents the game.

Example:

```
{
      "archery-mission-lvl": {
            "fails": 9,
            "gameId": "archery-world-tour",
            "playTime": 5291.706,
            "wins": 8,
      },
      "archery-world-mission-1": {
            "fails": 9,
            "gameId": "archery-world-tour",
            "playTime": 981,
            "wins": 6,
      },
      "bubble-woods-mission-1": {
            "fails": 19,
            "gameId": "bubble-woods",
            "playTime": 1206,
            "wins": 9,
      },
      "bubble-woods-mission-lvl": {
            "fails": 1,
            "gameId": "bubble-woods",
            "playTime": 100,
            "wins": 2,
      },
      "candy-bubble-mission-lvl": {
            "fails": 6,
            "gameId": "candy-bubble",
            "playTime": 1558,
            "wins": 6,
      }
}
```

I want you to group all the missions from the same game and accumulate the values of the "wins", "fails" and "playTime" into one single array with objects where for each game/key the properties will be playTime, gameId, wins and fails.

Example of output:

```
[{
      "fails": 18,
      "gameId": "archery-world-tour",
      "playTime": 6272.706,
      "wins": 14,
}, {
      "fails": 20,
      "gameId": "bubble-woods",
      "playTime": 1306,
      "wins": 11,
}, {
      "fails": 6,
      "gameId": "candy-bubble",
      "playTime": 1558,
      "wins": 6,
}, ]
```

You should build a script that logs this output and send it together with the email or add it into your previous project repository. You can implement this script in your own way, the important thing is that the output is the same as the example above.