Iridian Frontend Technical Assesement

Introduction

Welcome to Iridian Frontend Tech test.

The primary goal of this exercise is to assess how you reason about your ability to write clean, well tested and reusable code. There's no hard rules or tricky questions.

We hope you have fun.

Glossary

- Event - A sports event. A football match between Real Madrid and Barcelona for instance. **An event has 0 or more markets.**
- Market - A betting opportunity available for a specific event. "Team to win" or "Player to Score First" are possible examples of a market.
- Selection - A possible outcome for the market which has a price associated with it. Example: "Real Madrid to win at 1.25" or "Ronaldo to score first at 1.15".
- Betslip - The list of selections added by the user. You can think of this as the shopping cart.

Brief

This exercise is comprised of two views: * Event List - a list of events, markets and selections * Betslip - List of selections that the user is betting on

When the page loads, you should fetch the data to render the **event list**. Each event must contain the event name and a list of markets associated with that event. For each of those markets, its name should be displayed alongside a button for each associated selection. Please see the screens section for more details. **Please be aware that an event may have 0 markets and in that case, the event should be ignored (not displayed).** For the purpose of this exercise, you can consider that every market has at least one selection.

When a selection on the event list is clicked, it should change its color to green. Also, you should add it on the betslip by displaying its name, price and a remove button.

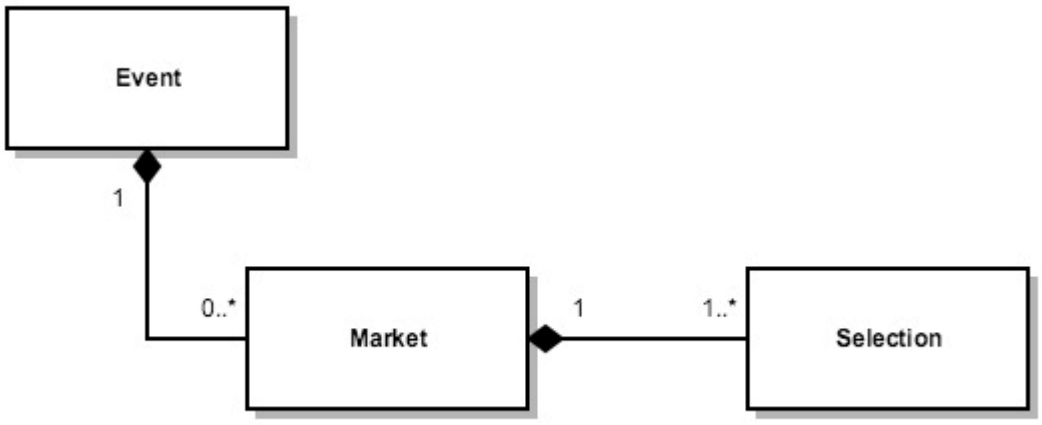
When the remove button for a selection in the betslip is clicked, the selection should be removed from the betslip, and the corresponding button on the event list should switch to its default color. The same behaviour should apply when a user clicks on a previously added selection (green button) on the event list.

Data

We've provided an endpoint that can be accessed in the following URL:

http://www.mocky.io/v2/59f08692310000b4130e9f71

Data Diagram



Although our endpoint provides hierarchical data, we suggest you normalize the data in a way that's more convenient for your application. Please provide justification for your choice.
You can also find a sample of the data data.json provided in the zip file

Types

```
/** The response type of the API call */
type ResponseType = EventsType[];

/** Event Entity */
type EventType = {
   id: string;
   name: string;
   markets: MarketType[];
};

/** Market Entity */
type MarketType = {
   id: string;
   name: string;
   markets: SelectionType[];
}

/** Selection Entity */
type SelectionType = {
   id: string;
   name: string;
   price: number;
}
```

Technology

At Iridian, we're big fans of React and Angular, right now we are looking for an Angurlar developer but we also accept React solutions.

We like to write clean code using proper programming patterns and JavaScript best practices and expect you to strive to do the same.

Screens

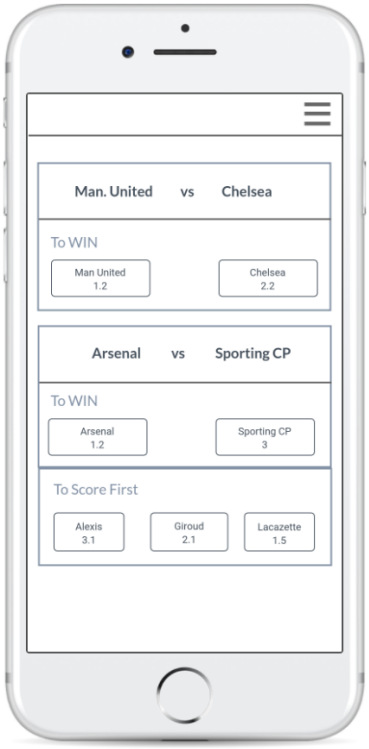
You can find a prototype outlining the basic functionality here: Prototype

The provided screens are a wireframe, not a final design for the application. You are expected to implement the user interface by yourself. The use of component libraries, like Material UI, might make this easier for you and is encouraged.

Event List View

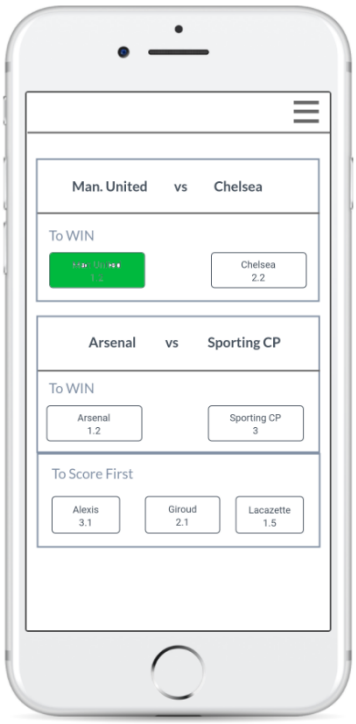
- Initial View

Event List initial view



- After a selection has been added

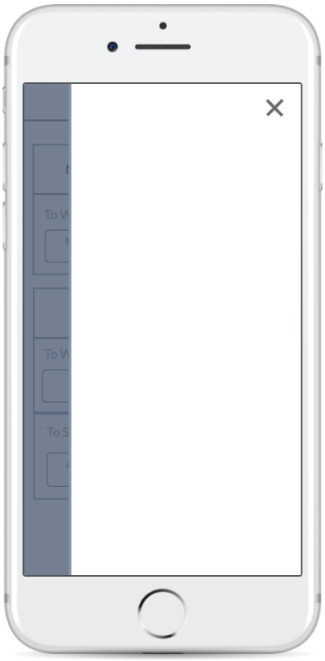
Event List - Selection Added



Betslip View

- Empty

Betslip – Empty



- With Selections

Betslip - With Selection

Tests
It is not required, but is a great if you present some kind of test. We like using cypress.js

The Deliverable

- A zip file or repository with your solution
- A README.md file explaining the decisions you've made solving this task including technology and library choices.
- Any instructions required to run your solution and tests.