# ATG.se frontend interview task

## Requirements

You need to create a React app where user types in horse betting game type and the app should show the **closest upcoming** or **closest results** game (if no upcoming games are available) as well as information about its races and starts.

Game has to be selected from the game schedule table (see `Data source` section below).

Game schedule only contains **game id** and **start time** for each game. So, after game id is selected, the complete game data needs to be loaded from second API endpoint (see `Data source` section below).

### UI

User interface should contain input field to put game type in. Below the input field there should be the area where information about the game is presented.

Information about the game should contain following data: - Game type - Races information (see below)

For each race: - Race number - Race name - Race start time - Starts information (see below)

For each start: - Start number - Horse name - Driver/rider first name + last name - Trainer first name + last name (expanded view) - Name of the horse father (expanded view)

Information about start should be presented in expandable block (collapsed by default). The following information should be available in the expanded view: - Trainer first name + last name - Name of the horse father

## Data source

### Game schedule

**API endpoint:** `https://www.atg.se/services/racinginfo/v1/api/products/<gameType>` - where `<gameType>` is replaced with the value input by the user.

**gameType** *(string)* - allowed values are `V75`, `V65`, `V64`, `V4`

Return value of the game schedule endpoint is following:

```
{
    betType: string, // "V75"
    upcoming: Array<GameInfo>,
    results: Array<GameInfo>
}
```

```
type GameInfo = {
    id: string,
    startTime: string // YYYY-MM-DDThh:mm:ss
}
```

Both `upcoming` and `results` can be empty arrays.

## Game data

**API endpoint**: `https://www.atg.se/services/racinginfo/v1/api/games/<gameId>` - where `<gameId>` is replaced with the value picked up from game schedule table.

**gameId** *(string)* - example `"V75_2018-05-12_6_5"`

Return value of game data API looks like following:

```
// Only relevant fields are stated
type GameResponse = {
    id: string,
    races: Array<GameRace>,
    status: string
};

// Only relevant fields are stated
type GameRace = {
    date: string,
    name: string,
    scheduledStartTime: string,
    starts: Array<RaceStart>
}

// Only relevant fields are stated
type RaceStart = {
    number: number,
    driver: {
        firstName: string,
        lastName: string
    },
    horse: {
        name: string,
        trainer: {
            firstName: string,
            lastName: string
        },
        pedigree: {
            father: {
                name: string
            }
        }
    }
}
```

## How to submit the solution

Aside the actual code the solution should contain `README.md` file with instructions how to run the application locally, run tests, lint, etc. File with instructions must be placed in the project root folder.

The solution can be submitted in any of the following ways: 1. Zip-package with all files and folders that is sent to HR-representative of ATG 2. Link to Github repository (on github.com) (also sent to HR-representative)

## Side notes

It is allowed to use app generators like `create-react-app`. But before submitting the solution all of the configuration needs to be ejected (`npm run eject` or `yarn eject`).

The choice of libraries to use is not restricted. The only mandatory lib to use is `React`.