Fullstack JavaScript assignment

Description

This page describes the assignment for a FullStack JavaScript position.

Duration

The duration of this assignment is 2 days.

Goals

The goal of the assignment is to check whether you are able to:

- Create an entire application from scratch, using:
  - `React` for the front-end,
  - `Node.js` for the back-end,
- Develop a user interface that will match a given design.
- Use git with readable and descriptive commits.
- Use best practices when it comes to development, project structure and code quality.
- Write clean, modular, well-organized and reliable code.
- Cover your code with unit / e2e tests with different test scenarios.
- Handle exceptions and errors appropriately.
- Write clear and concise documentation with the consumers of your code in mind.

Application to develop

You must develop an application that will allow an user to consult historical exchange rates for a specific currency.

Requirements

Setup

Create a private git repository to which you will give us access to, in order for us to review your work.

Back-end implementation

Create an API using `Node.js` and Express.js library.

Endpoints

Define new endpoints in order for the API consumer to be able to:

- Retrieve the list of currencies.
- Retrieve the real-time exchange rates for a set of currencies.
- Retrieve the historical exchange rates for a time-period for a specific currency.

To retrieve the exchange rates data, you must use the CurrencyScoop public API:

- You will need to create a free account to retrieve an API key that will allow you to query their API.
- Note that the free subscription:
    - Won't allow you to query their time-series endpoint.
    - Allows you to use maximum 5000 requests per day. **Bonus**: You can come up with a strategy to save roundtrips to their API.

Notes

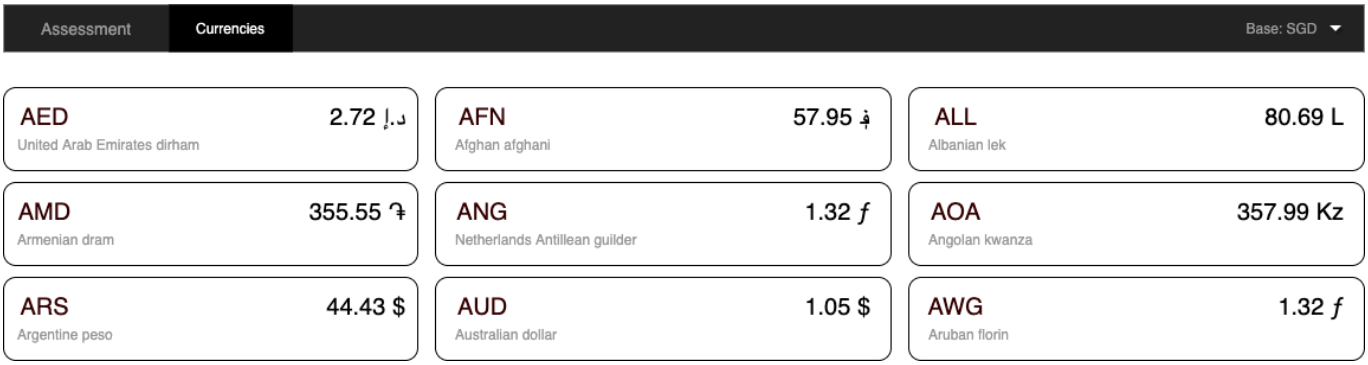
- The above endpoints must return the appropriate `HTTP` status code (`200`, `400`, `401`, `404`, `500`, ...) and data.
- You are allowed to add other endpoints if needed.
- You are allowed to use third-party libraries.
- The code should be covered by tests.

Front-end implementation

Create a web application using `ReactJS` library. The application should contain two pages.

Page 1 - list of currencies

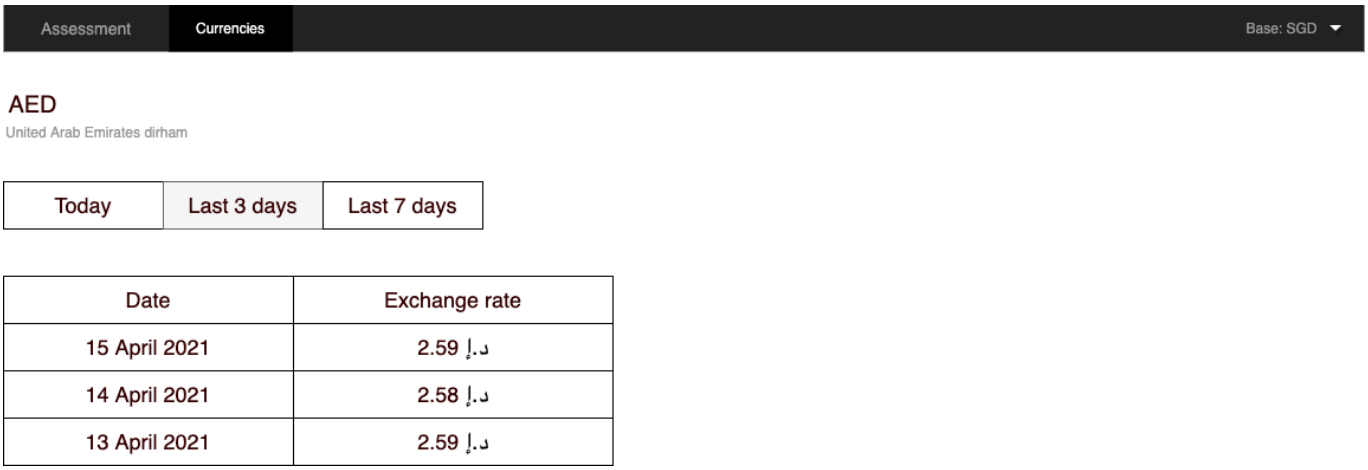
Sample mockup



- The page should display the list of the available currencies.
- For each currency, the following should be displayed:
    - Its currency code,
    - Its name,
    - The lastest exchange rate based on the selected base currency.
        - If the user change the base currency (dropdown located at the top rignt corner of the screen), this piece of data should be updated. The base currency the user selects should be saved in the local storage, so that his selection is not lost if he/she comes back to the app later.
        - You may use currency-symbol-map library to display the appropriate currency symbol.

Page 2 - currency detail

Sample mockup



- The page should display the currency details (code and name), as well as the historical exchange rates for the selected time-period. The exchange rates should be based on the selected base currency.
- The user should be able to select one of the three time-periods: Today (default), last 3 days, and last 7 days.
- Whenever the user selects a time-period or update the base currency, a request should be sent to your API. The table should be updated after receiving the response.
- **Bonus**: display the data as a chart instead of a list.

Notes

- The code should be covered by tests.
- The design of the application should follow the mockups.
- You are allowed to use third-party libraries to help you with the design layout (Bootstrap, SemanticUI, etc.), basic components (Dropdown, ...) and state management (Redux).

Documentation

Make sure your document covers the following:

- How your application is structured and the design choices you've made.
- The available endpoints defined in your API.
- How to setup and start the application after we clone the repository.

Remarks

Feel free to add everything you think meaningful to your application (improvement, new feature, ...), as long as the above requirements are met.