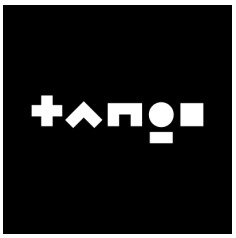

React Dev - GoT API app

Background

Your task is to implement a simple React app for displaying data from An API of Ice And Fire. That API uses no authentication and is available over HTTPS. The application would be deployed using static files hosting.

For simplicity, the application should only display 2 data types: **Characters** and **Houses**.

Business requirements

The application, as the entry point, should display the Table of Characters page.

From the Table of Characters link each House displayed to a dedicated page with the House Details.

From the House Details page it should be possible to go back to the Table of Characters page.

Table of Characters

Columns

Character

Concatenate character `name` and all `aliases` using comma.

Alive

Based on `born` and `died` fields:

- Display "Unknown" if both `born` and `died` are not provided.
- Display "No" if `born` is not provided.
- Display "No, died at X years old" when Character has died, where X is his age in years at the time of death.
- Display "Yes" when Character has not died.

Gender

Based on the `gender` field.

Culture

Based on the `culture` field.

Display "Unknown" when not provided.

Allegiances

Display list of House `id` fields based on what is found in `allegiances` field.

Link each House displayed to a dedicated page with House details.

Display “No allegiances” when not available.

Filtering

It should be possible to filter the list of Characters returned from API by Gender.
Use select or dropdown with “Any”, "Male" and "Female" as options.

It should be possible to filter the list of Characters returned from API by Culture.
Use text input.

Pagination

The API returns a single page of 10 results by default. It should be possible to paginate the results and the page should show 25 results per page by default instead.

The API returns the total number of pages for a given query in response headers. Display this number in the interface.

Application users should also be able to change the number of results per page.
Use select or dropdown with 10, 25 and 50 as options.

Pagination should have the following controls:

- First page
- Previous page
- Next page
- Last page

House Details

The page with the House Details should display the following data:

- Name of the House
- Region
- Coat of Arms
- Words
- Titles
- Seats
- Has died out
- Has overlord
- Number of Cadet Branches

Each House should be accessible by a dedicated URL.

Technical requirements

The solution should be implemented with React using Create React App.
You can use either JavaScript or TypeScript as your technical stack.
You can also use any open source dependencies you want, but be considerate – dependencies for network requests, routing, timedate manipulations and design systems should be sufficient.

Create a code repository on Github using Git.

Create a simple static file deployment at your discretion.

Tips

The goal of this task is to provide a discussion context for the subsequent technical interview and is not meant to be time consuming. Although it is intended for all levels of React Developers, whether it's Junior, Regular or Senior React Developer, we expect more attention to detail the more experienced you are.

Using solutions recommended by the framework is preferred. You can emphasise on certain aspects of the task to showcase your skills, either through cleverly using React built in mechanisms or leaving `TODO` notes for more advanced solutions that come to your mind or when taking shortcuts to reduce development time.

The time to complete this project will depend on your experience level, but based on our own developers executing this task, we estimate that it should not take more than 8 hours for Junior level, 5 hours for Regular level and 3 hours for Senior level. We understand and honor that you have a life outside work so we recommend that you do not exceed the above mentioned limit.

Do not invest your time into implementing additional business requirements that are not listed or into creating sophisticated look for the application since there is no design provided.

We would like to see one unit test (of different types of code) being showcased, but by any means please do not try ensuring 100% coverage – while it's never a waste of time in real projects, we don't want you to waste it for homework assignment.

Evaluation criteria

There are certain aspects that will be considered when evaluating your solution:

- code quality
- implementation according to the specifications
- expertise in using React
- expertise in using external dependencies
- expertise in testing your own code
- expertise in working with HTML5 and Web APIs
- expertise in working with CSS
- expertise in calling external APIs