Entry React developer TEST

Welcome to Scandiweb Junior Developer test assignment!

Thank you for your interest and time! Since 2003 we have been figuring out what is the minimum skills and knowledge level to be successful at your training and further career at scandiweb.

This test requires showcasing this minimum skill level by completing practical tasks. It helps you to check whether your level is already good to join us as a Junior Developer.

Overview

This task will put you face-to-face to some common tasks in the world of React development and possibly will get you acknowledged with a bit of new technologies.

You are be expected to fetch data from the GraphQL endpoint and to provide an interface to view and interact with this data. You can find the endpoint [here] (https://github.com/scandiweb/junior-react-endpoint), along with instructions on how to launch it.

The solution should get implemented as per design, which is available at [this link].

Have any questions? Please check out our Frequently Asked Questions page!

❓ **React Developer test - FAQ/frequently asked questions**

Before you start

The provided endpoint is a (GraphQL -https://graphql.org/learn/) endpoint. If you are not familiar with GraphQL - don't worry, only entry-level functionality is utilized. You can use any GraphQL client, for example, our own minimalistic: (**Opus** - https://www.npmjs.com/package/@tilework/opus) or more widespread and heavyweight (**Apollo** - https://www.apollographql.com/docs/react/).

The functionality should be implemented exactly as-per-design. Not necessarily pixel-perfect though. Pay close attention to all of the details, there should not be any noticeable mismatches between the designs and your implementation both in terms of design and functionality.

Think about the implementable functionality in scale, as in a real-life project. What will happen if you have more than 4 products? The pagination is not expected though - it is not in the design.

Toolkit requirements

When selecting a toolkit for this assignment, following these guidelines is mandatory.

Required:

- React. Anything not written in React will be rejected automatically. This is a React position, we expect a React solution.
- Class components. Due to our [work specifics] (https://docs.scandipwa.com/stack/override-mechanism/extending-javascript) we utilize them heavily.
- Create-react-app to scaffold the application.

Allowed:

- State management libraries (e.g. Redux, Recoil)
- CSS-in-JS approach allowers (e.g. styled-components)

Prohibited:

- UI libraries (e.g. Tailwind, Material UI, Ant Design)
- Functional components, due to the reasons described above.

Functionality requirements

- PLP - product listing page, a.k.a. category page
- PDP - product description page, a.k.a. product page
- Cart page + Cart overlay (minicart)

Details

See some more specific information on the main requirements below. Remember - if something is in the design, but is not explicitly described here or above, it should be implemented anyways.

- Ability to add/remove products and change their amounts in cart - on the cart page itself, PLP and PDP should be provided.
- For products that have various options (attributes) - the options should be selected.
- The selected options of added to cart products should be visible in cart overlay and in cart page.
- If an attribute is a swatch attribute (type = swatch), a representation of the value should be rendered on PDP and PLP, rather than text description (e.g. the color itself, not "Blue" or "0000FF")
- Filtering products by category name for all of the categories from BE
- The descriptions provided in HTML format should be parsed and presented as HTML, not as plain text
- Ability to change the currency of the store to one of the available currencies

Last, but not least

If you have any questions about the assignment, feel welcome to contact us with your questions!