The Trade Desk – Coding Exercise – Recommendations UI

The purpose of this exercise is both for us to get a sense of how you would approach a real problem that you might encounter at The Trade Desk.

Engineers at The Trade Desk take personal responsibility for the code they write. We will evaluate your submission as if your intention was to push this code into production. You are strongly encouraged to take the time necessary to produce something you are proud to deliver.

If you are unsure of the requirements, please reach out with any questions you have during the exercise.

We are not evaluating the solution for cross-browser, cross-device, or cross-operating system compatibility and the web application does not need to be designed with mobile devices in mind.

Keep in mind, however, that testing will likely be done in Chrome on a desktop computer running Windows.

If You Are New to React

We do not require candidates to have prior knowledge of React, but our coding exercise *is* a React application. If you are already familiar with UI development (and, in particular, web development) you should be able to pick up enough to complete the exercise fairly quickly.

We recommend the React documentation, it's well written and brief: https://reactjs.org/docs.

A Guide to the Exercise

The following steps should help you to get started.

How to Run the Exercise

The distributed files contain a pre-built version of the exercise. You should be able to extract the contents of the provided ZIP file and then open the "public/index.html" file in your browser.

Building the Exercise

Your next step should be to make sure you can build the exercise. To do this, you'll need to install Node.js and npm. You can get the latest version for your operating system here:

https://nodejs.org

Then, from your command prompt run:

```
> npm install
```

And then:

```
> npm run build
```

This will rebuild the application and you'll be able to run it by refreshing your browser window with "public/index.html".

Automatically Rebuilding Your Application

During development you can use the following command to automatically rebuild your application when your code changes (although you will still need to refresh the browser to see the modifications).

```
> npm run watch
```

Services

There are three services provided for interacting with recommendations: CampaignCoreSettingsRecommendationService, CampaignAdFormatRecommendationService and CampaignGeoRecommendationService.

You are encouraged to look at the source code for the services (to discover what methods are available) but modifying the services is outside of the bounds of the exercise – we'd like you to keep them intact, as they are.

Requirements

NOTE: Do not pull in additional packages or change the code in the "services" folder.
The reason for not pulling in additional packages is this is that we want to see your work, not the work of someone else. The services are there to simulate a back-end server and enforce some constraints.

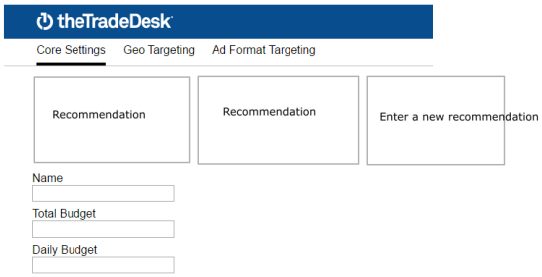
(1) **Fix the software architecture of the Recommendations component.**
The component has many intentional flaws in its implementation. We'd like to see what you would do to fix those flaws, so please feel welcome to change as much as necessary in the application (excepting the "services" folder) to end up with code in the Recommendations component that you would be proud to claim as something you'd written from scratch.

When considering the implementation of the component from a software engineering standpoint, keep in mind two use cases:

- Using the component in a large application (maintained by over 100 engineers, over multiple years) – **not** the toy application in this exercise, where recommendations might be present on many pages.
- Including the component in a library to be distributed to others for use in their own applications (for instance, in a commercial component library).

(2) **Add the ability to dismiss individual recommendations to the Recommendations component.**
Note that each of the provided recommendations services have a "dismiss" function.

(3) **Update the *aesthetic* design for the Recommendations component.**



The component should render with boxes as in the **intentionally rough** mockup above. The *actual* design is up to you – the goal here would be to show us what you would come up with, given *some* constraints, but in the absence of a high-fidelity mockup from our UX team.

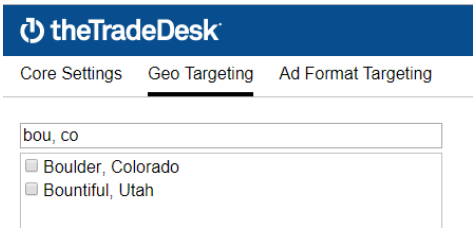
User-entered recommendations are not supported on the "Geo Targeting" page in the current application, but all tabs should otherwise match the above mockup.

(Requirements continued on next page…)

(4) **Investigate a bug on the "Geo Targeting" page.**
Users have reported that they *sometimes* get the wrong results for their search and sometimes it works, even when they search for the same thing.

Their work around is to backspace and then type the last letter of their search a second time. According to the users, this usually fixes the issue.

One of the users has helpfully supplied a screenshot demonstrating the problem:



Since they included "co" after the comma, "Bountiful, Utah" is not a valid result. This bug should be fixed in the submitted solution.

Common Questions

(1) **Can I post my solution online?**

Because we use this exercise for other candidates, **please do not post your solution anywhere publicly available online, including any public repositories** (e.g. GitHub). Thanks :)

(2) **The requirements say that we are not allowed to pull in additional modules, but I'd like to pull in a unit testing framework. Is that allowed?**

Adding modules related strictly to unit testing *is* allowed as an exception to this requirement.

If you feel it will increase the quality of your submission, you are welcome to do so and you won't be penalized in any way. However, adding unit testing can considerably increase the amount of time spent on the exercise, so we don't recommend it.

(3) **Should the changes made by the user persist over browser refreshes?**

No, although changes should persist between tabs.

(4) **I understand that I can't change the code in the "src/services" directory. Can I change other code?**

Yes - you are welcome to make any changes you like under the "src" directory **except** for the "services" folder.