Trainee (Web) BBC News JS Coding Test

Creating an articles ranker

We would like to assess your abilities to write modern, clean and quality-driven JavaScript. This is a coding exercise to be done at home.

Your task is to design and implement a news site that renders random articles, and has the ability for the user to rank the articles once they have read them all. You are free to design the UI in any way you like.

If you are struggling for time, it would be much better to focus on finishing what you are most comfortable with, rather than leaving multiple things unfinished. If you do not have time to fully implement some functionality, but have a clear plan for it, feel free to give a concise summary of your plan.

You have up to one week for this exercise. We will arrange the timings with you; please let us know as early as possible if you cannot find this time for personal reasons and we will make whatever accommodations we can for you but please understand we need to maintain a fair process for all applicants.

Task Information

The site should load and display one article at a time, and provide a way for the user to move to the next article. We have provided you with JSON data for 5 articles, you must present at least 3 of these articles. The dataset is here: https://github.com/bbc/news-coding-test-dataset

When the user gets to the end of the process, they should be presented with a way to submit a ranking of the articles that they read. It does not matter how they are ranked.

Required

- Use of modern JS throughout.
- Use of Git throughout. The solution must be hosted on GitHub, Bitbucket, or similar, and be sent to us with a link that is accessible to us

- Aside from library code, all code should be created by you and not taken from other projects
- There should be the following HTTP requests (could be stubbed, see guidance):
    - 3-5 GET requests for the article data
    - 1 POST request submitting the ranking
- Semantic HTML
- A basic level of styling
- Handling of errors, such as network unavailability or slow connection
    - Provide a way of inducing such errors if you stub the server
- An attempt at ranking

Guidance

- No actual server logic is required - feel free to stub any HTTP requests with a local, asynchronous function
- Feel free to use Babel and any other preprocessors for CSS etc, and any libraries that aid an implementation
- We are purposefully not requiring the use of React or similar library, so you can implement in your preferred way. This extends to the use of packages that handle build configuration, e.g. create-react-app. A manual build configuration would be a bonus.

Bonus considerations

- Performance
    - Preloading the subsequent article when on an article page
- Accessibility
- Testing - a minimum of unit tests. Could include integration, end-to-end, performance, etc
- Considering the user experience
- Scalability of CSS