Time	Activity
Thursday, December 10th	
15:00 - 18:00	Research on forums and discussion boards online about missing features on job and employment portals, users would like to be implemented soon. I took this as a starting point, to try to get inspired about whick kind of web app to create.
Friday, December 11th	
9:00	In-depth visit of Torre.co web app, genome structure and content, design patterns, search and filtering methods, and content in general. Purpose: Get familiarized with graphic style of the site, and the way the information is organized and displayed on it.
10:30	Creation of React project and bootstrapped application, using “create-react-app” Node module. Addition of third party libraries. Addition to a GitHub repository.
11:00	Code customization on base project files. Creation of environment variables to let app to be deployed on Netlify hosting service.
12:00	Setup of NVM (Node Version Manager) on host machine, to let some third party libraries added to project to work properly on Netlify. Version of Node.js used on project: 12.16.1. Break.
14:00	Testing of API calls to given endpoints and studying structure of responses (using Postman). Several trials trying to get desired results, and looking to data obtained, to get insights and inspiration about what to create.
15:00	Creation of a base project in React, to grab data from the API for a specific user, manage it inside application using a global state manager (React Context) and display it in a simple way (as a starting point). Main libraries installed: Axios (REST API management based in Promises) and Ed-Grid (layout manager based in Flexbox and CSS Grid).
17:00	Getting issues trying to properly display data on app, due to CORS origin blocking from server side of API. Online research was made to find a suitable solution.
18:00	Looking at forums and StackOverflow posts, finally the cause of the problem was discovered: The API was rejecting the request to given endpoints sent from my localhost, even when browser and Postman requests were being done properly answered, because call didn't include a secure https header. Best solution: Include a proxy server in the middle of request, between localhost and endpoint. To avoid creation of a proxy from scratch (having to run a second Node.js app alongside), I choose a service hosted on Heroku servers to emulate a proxy request. After include it on code, problem was finally solved.
19:00	Import of GitHub project to Netlify webapp hosting service. Creation of a redirect file (added to project) to let Netlify handle routes defined on cade using React Router library. Build and deployment. Testing of live app on various browsers, and on mobile, to check responsiveness. Break.
21:00	Testing API calls to endpoints from React, getting data and trasnforming it into arrays and objects, for easier manipulation. In the meantime, wondering what to do with the data. My goal: Create something simple (as per the time given to complete the test), but well done, easy to use, meeting coding good practices, following graphic style of Torre site, and including responsiveness to be used from a smartphone. I was indecise about what to do. Some ideas came to my mind... still not translated to code, yet. Break.
Saturday, December 12th	
10:00	Throughout the morning, I finally get inspired: Why not to get the main info from an user, but to display it like the navigation menu on a videogame console, using the keyboard or touchscreen? I'm a big fan of videogames, so the idea immediately appealed to me, and would be a good way to show off my skills in React, while keeping the app simple. Started to work on it.
18:00	Unfortunately, I hadn't time during the day to work on the application; I was outside home. Was able to return until night. Started coding immediately.
19:00	Main page re-design started. After several attempts (and a lot of CSS trial and errors), finally I got it to look as I wanted.
21:00	Taking care of layout design details. I want it to look as good as possible, in every screen size.
23:00	Working on navbar and navigation. I decided to use useState hook and pass a function as a prop to change screens on the SPA, instead of use React Router library.
Sunday, December 13th	
0:00	Adjusting layout responsiveness. Working on Sass file.
2:00	Search component completed. Functionality debugged and tested several times.
4:00	Working on user bio highlights page. Data from API call is stored on global state, and the request itself is triggered as soon the component mounts, using the useEffect hook.
6:00	All pages completed. Doing some refactoring to keep everything as clean and concise as possible.
7:00	Teste completed. Form submitted.