Front-End Technical Challenge

Introduction

For this technical challenge, we would like you to provide an interface for creating a role based Todo List. This is an exercise in permissions management where users have different roles and therefore will be exposed to different content and levels of interaction based on those roles.

Technologies

This challenge requires you to make a single page application. Below are the technologies we require you to use as this will be the same technologies you will be working with in this position:

- React
- React-Router (v4)
- Redux
- Babel
- Webpack

About Boilerplates

For the purpose of this challenge, we ask that you do not use any third party boilerplates to start your application (ie. create-react-app, etc.), but rather start from scratch. If you use a boilerplate that you yourself created, please make a note in your submission and provide a link to your boilerplate repo on github and make sure it is public. We want to make sure you have a clear understanding of an application setup by doing it yourself.

Specifications:

1. This application should have four routes:
    a. **'/login'**
        i. This is a standard login form and serves as the main entry into the rest of the app. This form can be fairly dumb for our purposes, there is no need to have actual user accounts, all that is needed is for someone to enter the application as either a **'Manager'** or an '**Employee'.** If any of the other routes are accessed without someone logging in first, redirect them to the login page.
    b. **'/todos'**
        i. This is the page a user is taken to after logging in. It is a list of todos that can be styled any way you want (with good UI/UX in mind). A user should also be able to filter by all, completed, or incomplete todos.
        ii. A Todo must have:
            1. Title
            2. Todo Summary
            3. Indication if task is complete or not
            4. For the '**Employee'** Role:

a. Only a regular '**Employee'** can change that completion status of a task ('**Managers'** should not be able to change if a task is complete or not).

5. For the **'Manager'** Role**:**
   a. A task should have an 'edit' button that can only be seen by someone logged in as a **'Manager'**. Clicking the 'edit' button should redirect to the '/edit' route, with the selected-for-edit todo data pre-populated.
   b. A task should have a 'delete' button that can only be seen by someone logged in as a **'Manager'.** Clicking the 'delete' button will remove the todo from the list.
   c. A **'Manager'** should also see a 'create' button somewhere on this page. Clicking this button will redirect to the '/create' route.

c. '/create'

i. This route should only be accessible by a user with the **'Manager'** role. If the **'Employee'** role tries to access this route either redirect them back to '/todos' (or '/login' if no one has logged in yet) or display a 'permission denied' screen of your own design.

ii. A **'Manager'** should be able to create a todo and add it to the todo list. There should also be a way to navigate back to the '/todos' route where the new todos will be rendered.

d. '/edit

i. This route should only be accessible by a user with the **'Manager'** role. If the **'Employee'** role tries to access this route either redirect them back to '/todos' (or '/login' if no one has logged in yet) or display a 'permission denied' screen of your own design.

ii. A **'Manager'** should be able to edit a todo. There should also be a way to navigate back to the '/todos' route where the edited todo has been updated.

2. The '/todos', '/create', and '/edit' routes should all have 'logout' buttons. Logging out should bring you back to the '/login' page after deleting the logged in user's data - make sure other routes aren't accessible until a user has logged in again. Persist the 'todo' data locally (Redux), so that the same todo list is shown between logins and logouts (this is antithetical to usual logout practices, but is useful here for us to examine the todo list while switching roles. If you decide to implement and persist data in a back-end, then logout should delete all local data, and login initiates the getting of todo data).

Bonus Points:

1. Use TypeScript! (We use TypeScript so you would have to get used to it anyway.)
2. Give your todos a backend!
   a. Create an API to fetch and update your todos
   b. Start associating todos with actual users

    c. Save your todos in a database. (In a relational way)
3. Use Sass for you style sheet.
4. Test your application using any framework you wish.
5. Deploy it, and do it live!

What we are looking for:

1. That your application works! It should be easy for us to clone your repo, install dependencies, put in any environmental variables you require, and fire it up. A clear and concise Readme will go a long way for this process; as well as making sure your application is not dependant on your local machine's environment.
2. Codebase quality. We like spaghetti but not spaghetti code. We will be taking a look at your source code to see how easy it is to read, how modular and reusable it is, and how you handle separation of concerns and DRYness.
3. Did you complete the specifications. Did you do everything we asked you to do? Did you take the initiative and take on the bonus points. Completing the bonus points and adding in your own features is an excellent way to make yourself stand out from other applicants.
4. Excellent UI/UX. Functional apps are great, but apps that are easy and enjoyable to use are even better. We are looking forward to seeing what you come up with!

Submissions:

Create a Github repo with a clear Readme about how to start your project locally. Respond to the e-mail thread from which you received this challenge with the repo link. If you deployed your app, include that link as well. Everything you submit must be your own work, if any portion of your submission was completed by another individual, you will be immediately disqualified.

If you have any questions about the challenge itself, submission process, due date, or anything else, do not hesitate to reach out and ask on the thread from which you received this challenge. Best of luck!