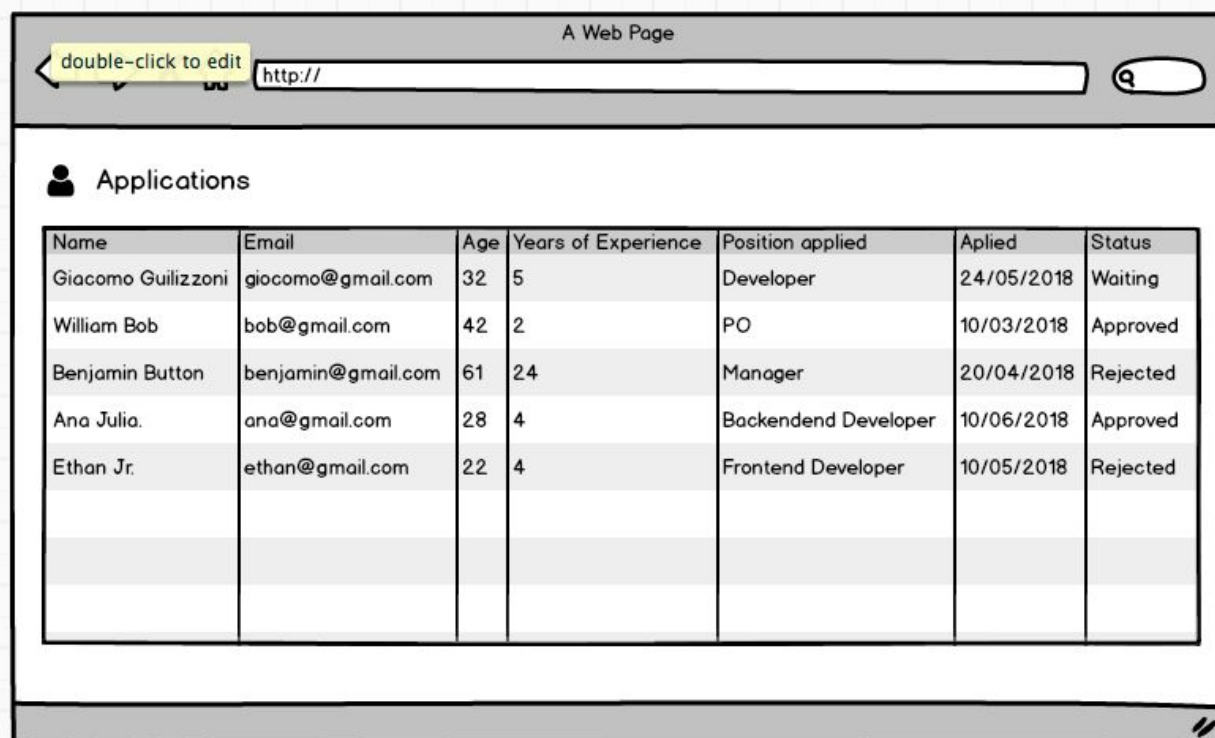


Frontend Engineer - Coding Challenge

Introduction

Maria works at Personio as a recruiter. She has to sift through a lot of applications every day to find the best possible talent for her favourite company! In order to help her do it efficiently, she uses a small application which displays all the relevant candidate information in a table:



Name	Email	Age	Years of Experience	Position applied	Aplied	Status
Giacomo Guilizzoni	giocomo@gmail.com	32	5	Developer	24/05/2018	Waiting
William Bob	bob@gmail.com	42	2	PO	10/03/2018	Approved
Benjamin Button	benjamin@gmail.com	61	24	Manager	20/04/2018	Rejected
Ana Julia.	ana@gmail.com	28	4	Backendend Developer	10/06/2018	Approved
Ethan Jr.	ethan@gmail.com	22	4	Frontend Developer	10/05/2018	Rejected

Lately, Maria has been even busier because Personio started to get more awesome candidate applications. The table got too cluttered with information, and she needs a better way to filter the incoming applications. We've invited you to help her build a new and optimized version of her recruiting application.

After some alignments with Maria, you found out that she would like to:

- See the list with all applications showing the following data:
 - Name (string)
 - Email (string)
 - Age (string - birthday)
 - Years of experience (number)
 - Position applied (string)
 - Date of application (string)
 - Status of the application (string: approved | rejected | waiting)
- Sort by:
 - Position applied

- Years of experience
 - Date of application
- Filter by:
 - Name
 - Status (approved/reject/waiting)
 - Position applied
- Have the URL changed every time she applies sorting or filtering so that she could share this URL with her colleagues
 - Example: If she is filtering by the position applied, the URL should change in a way that if she reloads the page, it should show the same data that she was looking for before.

Maria is opened for you to improve the UI in whatever way you see fit.

Backend engineers offered an endpoint for you to access applicants data:

<http://personio-fe-test.herokuapp.com/api/v1/candidates> . This endpoint is known to be somewhat buggy, and it will return errors and occasional delays, so be ready to process these accordingly!

Nonfunctional requirements

- Use React.js to accomplish the task. This is where you can show off your skills and knowledge about the react ecosystem. Feel free to use state management and routing related libraries. You can use any utility libraries like lodash or underscore.
- Please, do not use any list rendering libraries (e.g. react table) for react. It would be great to see which components you will create yourself to tackle the task.

What has to be delivered

- Provide a deadline by when you'll hand in your solution.
- An archive containing a .git folder. Best would be to have a commit history that represents your natural working process (so, if possible, no history rewrites / rebase).
- Instructions on how to bootstrap your app (in Readme.md of your repository).

What do we look at when checking out the solution

- UX and Functionality
- UI and Styling structure
- Code quality
- Architecture and Scalability

Duration and effort

Before implementation please timebox the implementation effort to maximum six hours in total.

The main thing - have fun!