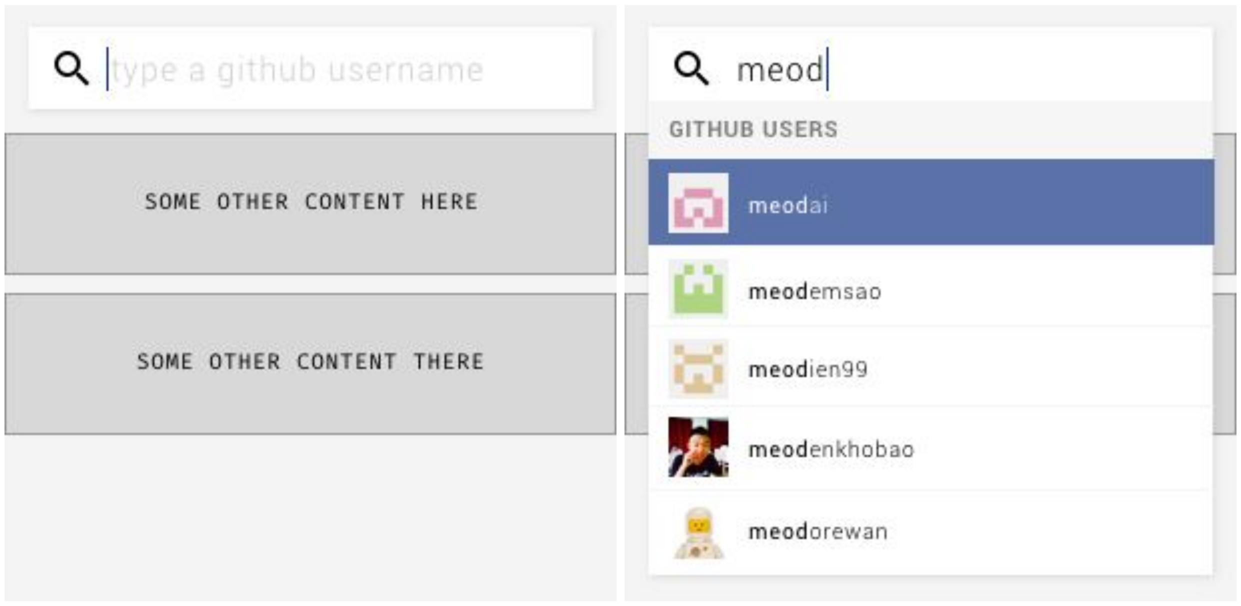
Senior Frontend Developer - Technical Assignment

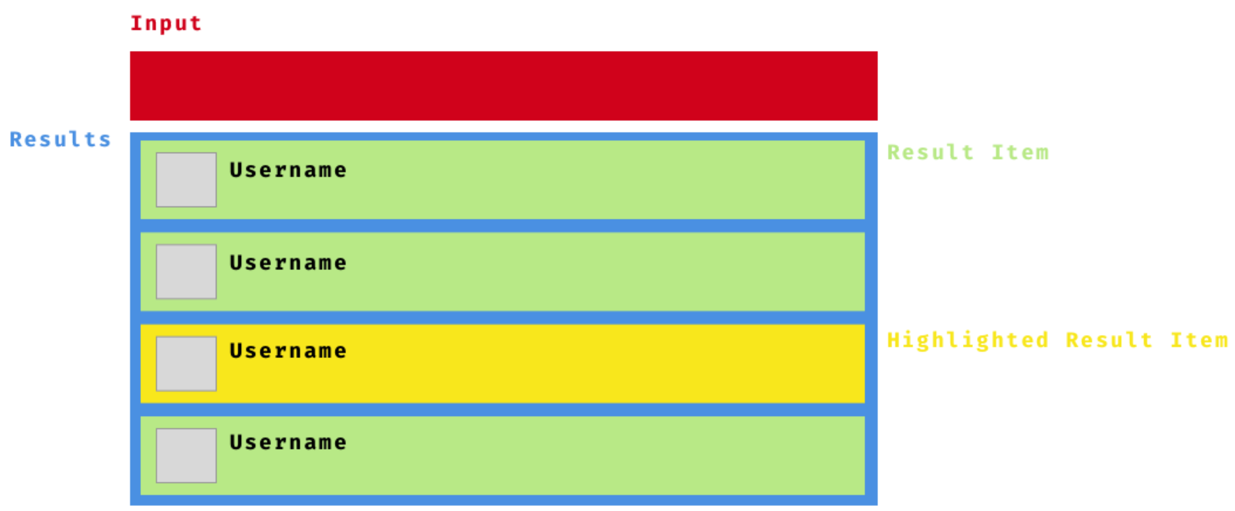
Congrats, we only get to this stage with candidates we're excited about working with. In order to review your technical abilities and get a sense of how you work, we'd like to move forward with a coding assignment. We understand your time is valuable - please don't spend more than 3 hours on the coding exercise - we understand if some things aren't finished, or are missing polish.

The Task: Github users autocomplete search

This assignment consists of creating a search field with autocomplete functionality. As the user types, github user will show up in the autocomplete results.



How your assignment can look like at the end.



Visually, this assignment has the following elements:
— **Input:** a input field where the user types search keywords
— **Results:** where the results from the Github API will be shown, it is located below the **Input** element. Note: The results don't affect the rest of page (i.e. they don't push further content down).
— **Result Item:** each username matching the user keywords (which comes from the Github API) is displayed on a Result Item. When clicked, it should go to the user's github profile page. It includes the user avatar and username.
— Highlighted Result Item: the same as a result item, but with a different background. Only one result item is highlighted at a time and there is always an highlighted item (by default, the first on the list).

The autocomplete field should be keyboard accessible and support the following features:

- **Arrow ↓ :** highlight next autocomplete result. If last one is highlighted, don't do anything.
- **Arrow ↑ :** highlight previous autocomplete result. If first one is highlighted, don't do anything.
- **Enter ↩:** go to the highlighted result's user github profile page (same as clicking on it).

Technical Specifications

You are free to use whatever technology or library you prefer. Or even no library at all. Use something that you like or that you have experience with or try something you always wanted to try but never had the chance before.

What will we look for

- Does your solution do what is required and the best for the user as it could be?
- Is your solution simple, readable and clear?
- Are you able to communicate your thoughts through code?
- Does your solution separate concerns?
- Does your file/folder structure read like a table of contents?

Some links that could help

- ITCSS video -> https://www.youtube.com/watch?v=1OKZOV-iLj4
- CSS guidelines -> http://cssguidelin.es/
- Functional Javascript -> https://www.sitepoint.com/introduction-functional-javascript/

Please also answer these questions after you have completed the assignment

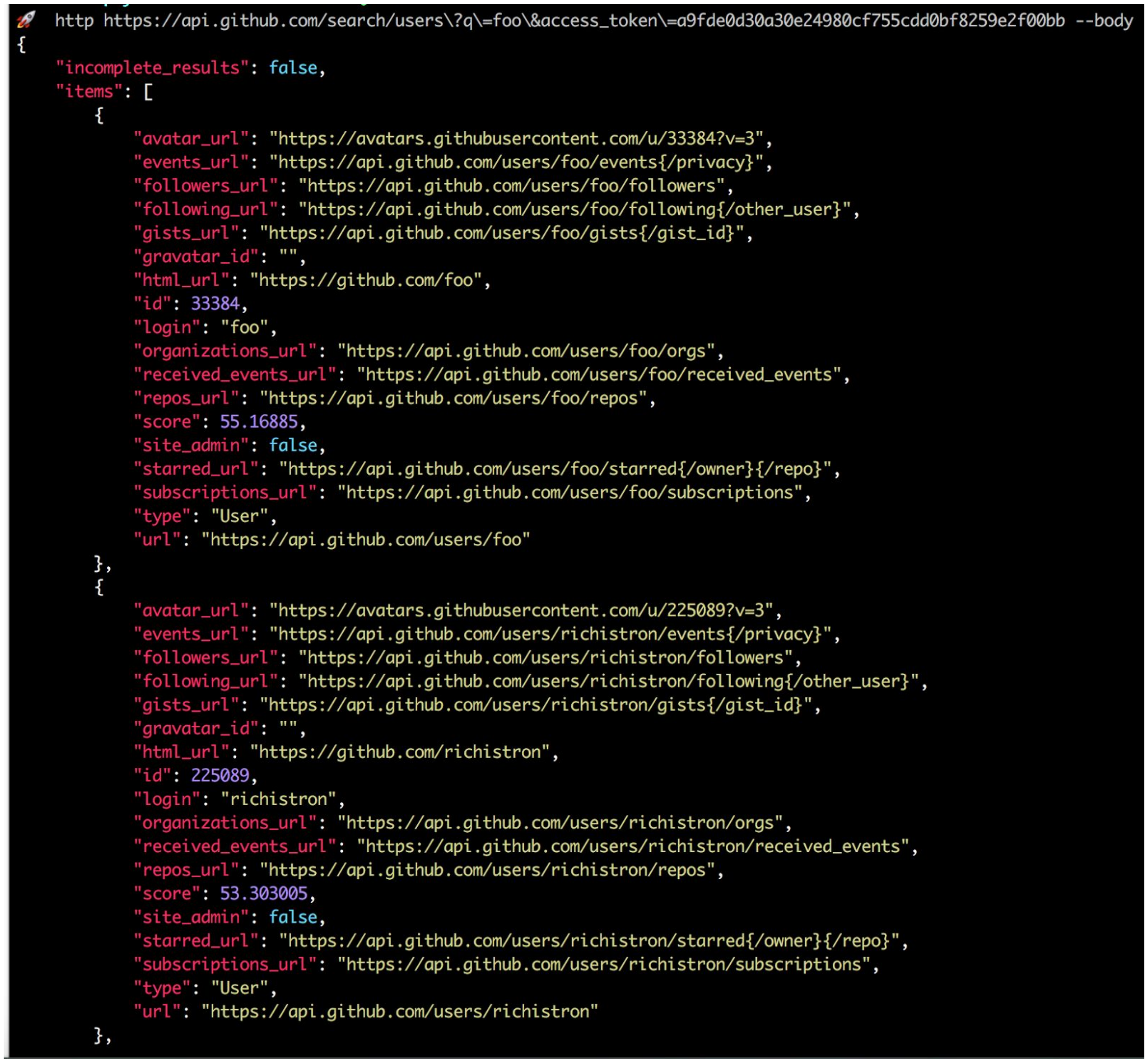
- What do you like about your solution
- What do you dislike about your solution?
- If you had a full day more to work on this, what would you improve?
- If you would start from scratch now, what would you do differently?

Notes:

Github User Search

To get users from the github api, just perform a GET request to `https://api.github.com/search/users` and the parameters:

- q : query
- access_token: your access token. You can generate one in https://github.com/settings/tokens



```
http https://api.github.com/search/users\?q\=foo\&access_token\=a9fde0d30a30e24980cf755cdd0bf8259e2f00bb --body
{
    "incomplete_results": false,
    "items": [
        {
            "avatar_url": "https://avatars.githubusercontent.com/u/33384?v=3",
            "events_url": "https://api.github.com/users/foo/events{/privacy}",
            "followers_url": "https://api.github.com/users/foo/followers",
            "following_url": "https://api.github.com/users/foo/following{/other_user}",
            "gists_url": "https://api.github.com/users/foo/gists{/gist_id}",
            "gravatar_id": "",
            "html_url": "https://github.com/foo",
            "id": 33384,
            "login": "foo",
            "organizations_url": "https://api.github.com/users/foo/orgs",
            "received_events_url": "https://api.github.com/users/foo/received_events",
            "repos_url": "https://api.github.com/users/foo/repos",
            "score": 55.16885,
            "site_admin": false,
            "starred_url": "https://api.github.com/users/foo/starred{/owner}{/repo}",
            "subscriptions_url": "https://api.github.com/users/foo/subscriptions",
            "type": "User",
            "url": "https://api.github.com/users/foo"
        },
        {
            "avatar_url": "https://avatars.githubusercontent.com/u/225089?v=3",
            "events_url": "https://api.github.com/users/richistron/events{/privacy}",
            "followers_url": "https://api.github.com/users/richistron/followers",
            "following_url": "https://api.github.com/users/richistron/following{/other_user}",
            "gists_url": "https://api.github.com/users/richistron/gists{/gist_id}",
            "gravatar_id": "",
            "html_url": "https://github.com/richistron",
            "id": 225089,
            "login": "richistron",
            "organizations_url": "https://api.github.com/users/richistron/orgs",
            "received_events_url": "https://api.github.com/users/richistron/received_events",
            "repos_url": "https://api.github.com/users/richistron/repos",
            "score": 53.303005,
            "site_admin": false,
            "starred_url": "https://api.github.com/users/richistron/starred{/owner}{/repo}",
            "subscriptions_url": "https://api.github.com/users/richistron/subscriptions",
            "type": "User",
            "url": "https://api.github.com/users/richistron"
        },
```

Example request for query “foo”.