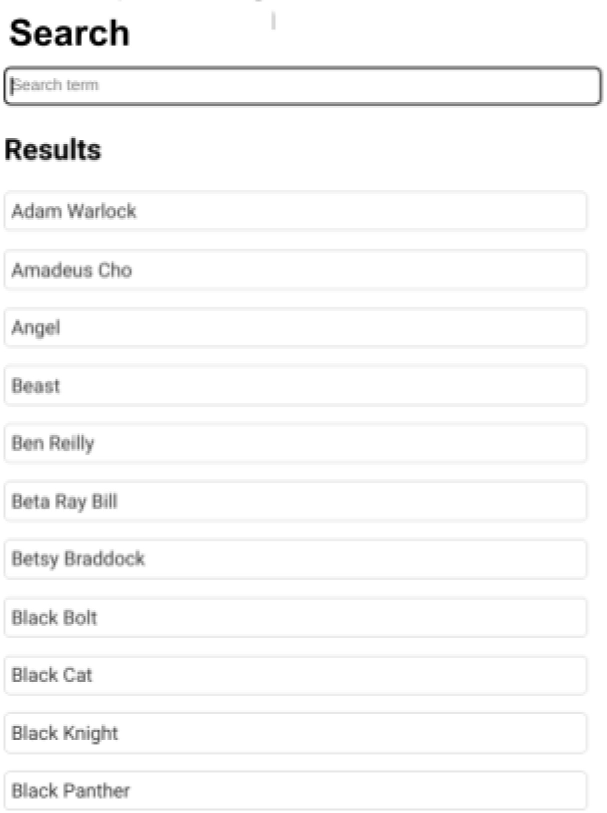
UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIAS DE COMPUTAÇÃO
SCC0219 - Introduction to Web Development

Exercise 10: React or Vue

Note: In all questions, use a local installation of React or Vue. Use React or Vue components to implement the functionalities.

1. A real-time search function means that as soon as you type a term into a search box, the results are displayed in real-time. We are going to implement a search component with this functionality. It has a fixed list of entries and, when you type a term in the search field, the results are displayed. Starting from the `index.html` file (Appendix 1 at the end of this document), make this widget work using React or Vue.

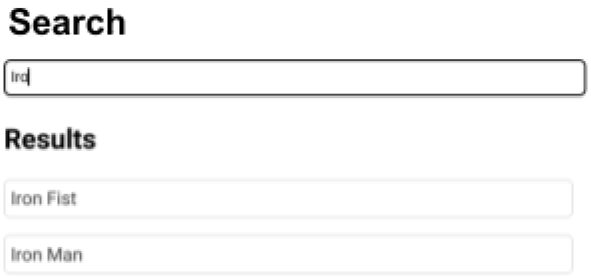


Search

Search term

Results

Adam Warlock
Amadeus Cho
Angel
Beast
Ben Reilly
Beta Ray Bill
Betsy Braddock
Black Bolt
Black Cat
Black Knight
Black Panther



Search

Iro

Results

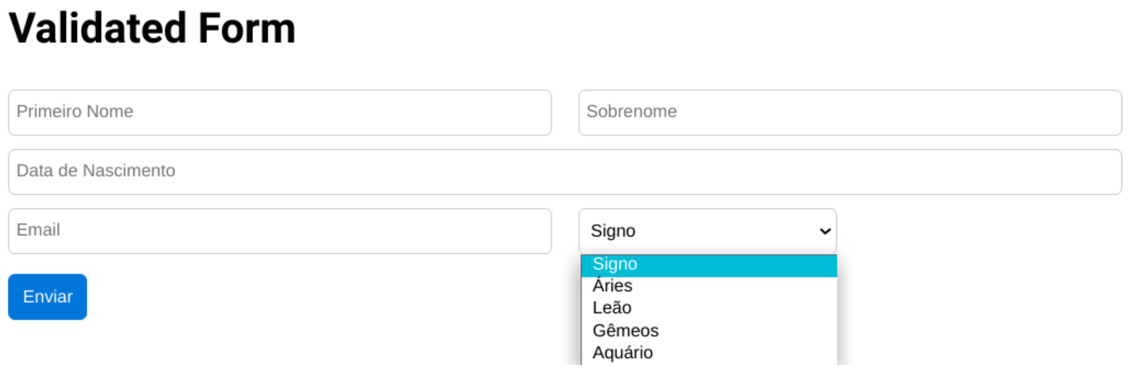
Iron Fist
Iron Man

2. Forms are often indispensable in web applications. In this exercise, you will implement a component (React or Vue) with a form whose fields may have the validation attributes shown below:

    a. `data-required` - If present, the field is required.

    b. `data-type` - It has 2 possible values: date and email. The field has to be a string in a format compatible with the specified type. For date, the format should be `dd/mm/yyyy`, and for email, it should be `name@site.(com|edu|br)`.

    You must implement the validation for the form fields in accordance with these rules. Note that the code doesn't need to communicate with a server, just to validate the form fields. When a field is not valid, in addition to showing an error message, it must turn red. Do not use a third-party validation library. Use the `index.html file` (Appendix 2 at the end of this document) as a starting point.



Challenger (2 pts):

3. For this challenge, you're expected to make a ToDo List-style application using Vue.js or React and component concepts, and everything you've learned so far. In it, there is a ToDo list with unfulfilled tasks and a DONE list with tasks that have already been completed.

    What this application has to do:

    a. Add and delete items from the ToDo list.

    b. There must be a Check system in this list (for example, a checkbox or similar)

    c. When a task from this list is checked, it should go to an area below the main list. A DONE task area!

    d. Tasks from the DONE list can also be deleted, and if they are deselected, they should go back to the ToDo list.

    e. To simulate the existence of other people doing the activities, every minute activity in the ToDo list is checked randomly (in the interface, everything happens as if it had been checked by a user).

Write a nice CSS for your application.

Anexo 1 (`index.html`):

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Search</title>
  </head>
  <body>
    <div id="app" class="container-fluid">
      <h1>Vue.js Search</h1>

      <div class="form-group">
        <input type="text" class="form-control"
               placeholder="Search term">
      </div>

      <div class="result-container">
        <h2>Results<span></span></h2>
        <ul class="list-group">
          <li class="list-group-item" </li>
        </ul>
      </div>

    </div>
  </body>
</html>
```

Anexo 2 (`index.html`):

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Validated Form</title>
  </head>
  <body>
    <div id="app" class="container-fluid">
      <h1>Validated Form</h1>

      <div class="form-container" >
        <div class="form-row">
          <div class="form-group col-6">
            <input type="text" class="form-control" data-required
                   placeholder="Primeiro Nome">
          </div>
          <div class="form-group col-6">
            <input type="text" class="form-control" placeholder="Sobrenome">
          </div>
          <div class="form-group col-12">
            <input type="text" class="form-control"
                   data-required data-type="date"
                   placeholder="Data de Nascimento" >
          </div>
          <div class="form-group col-md-6">
            <input type="text" class="form-control" placeholder="Email"
                   data-type="email">
          </div>
          <div class="form-group col-md-4">
            <select id="state" class="form-control" data-required>
              <option selected>Signo</option>
              <option>Áries</option>
              <option>Leão</option>
              <option>...</option>
              <option>...</option>
            </select>
          </div>
          <div class="form-group col-12">
        </div>
        <button type="submit" class="btn btn-primary">Enviar</button>
      </div>
    </div>
  </body>
</html>
```

Good luck!