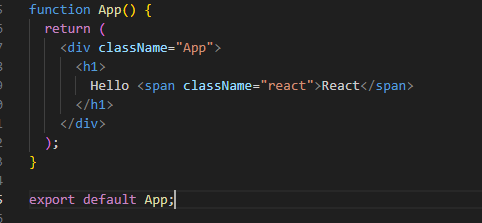
**Exercise 4**

1. Design this website as image below

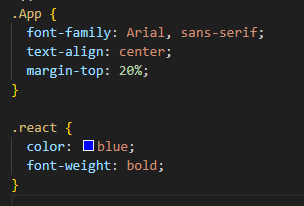


B1: tạo file App.js

B2: viết function App để trả về giao diện JSX

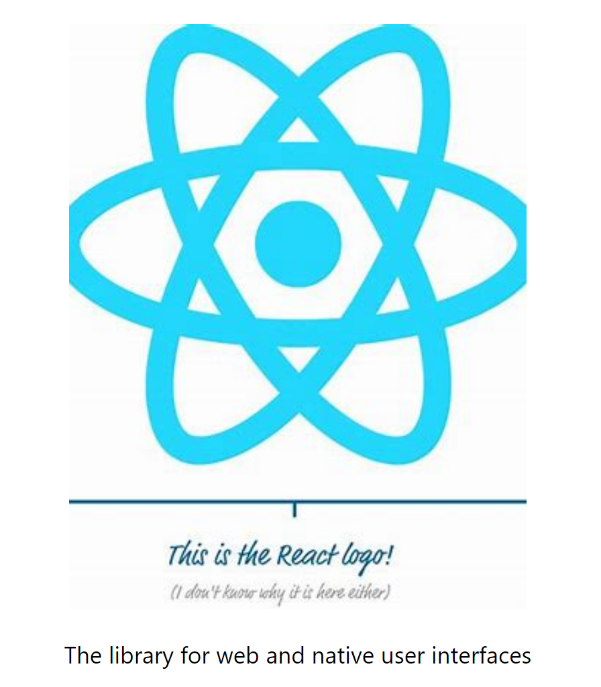


B3: tạo thêm file App.css để thiết kế lại giao diện



B4: nhập lệnh npm start để khởi chạy giao diện sẽ được kết quả



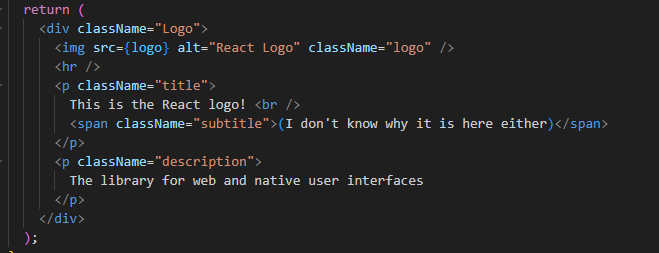
1. Design this website as image below

B1: Tạo 1 dự án mới và tạo file App.js

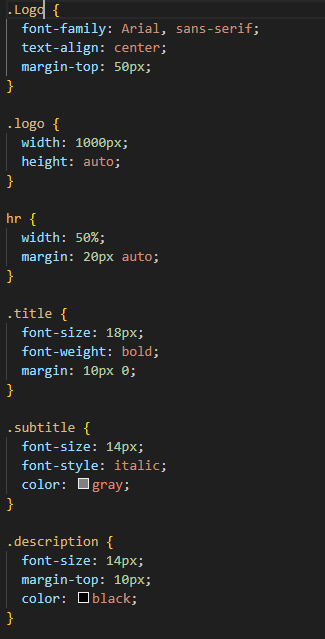
B2: Lưu hình ảnh logo dưới tên logo.svg vào thư mục và import vào để sử dụng



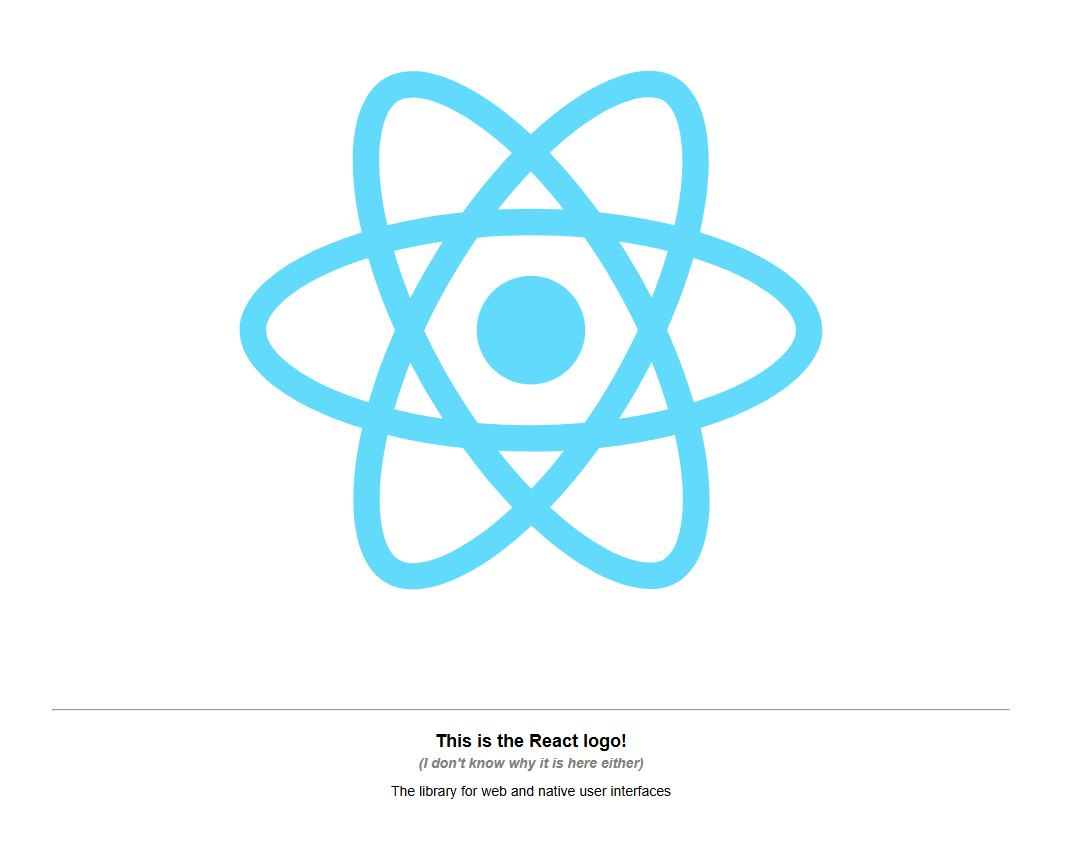
B3: Tạo các class Logo, title và description để hiện thị các nội dung lên giao diện, tất cả được bao bọc trong các thẻ <div>



B4: tạo thêm 1 file App.css để design lại các class và thẻ cho hoàn chỉnh



B5: khởi chạy bằng lệnh npm start và được kết quả



1. Create a navbar as image below with JSX



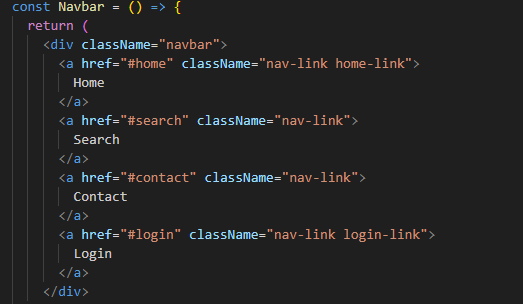
B1: Tạo 1 dự án mới và tạo file với tên Navbar.js và Navbar.css

B2: Trong Navbar.js import thêm CSS file Navbar.css

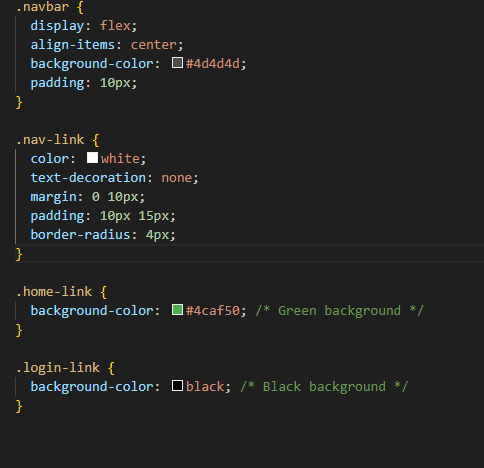


B3:

Tạo biến Navbar và trong biến có những class từng button nằm trong các thẻ <a>. Các class và thẻ <a> được bao bọc bởi 1 thẻ <div>



B4: Trong file Navbar.css sẽ chứa những câu lệnh để design lại giao diện cho các button



B5: Khởi chạy bằng lệnh npm start và được kết quả



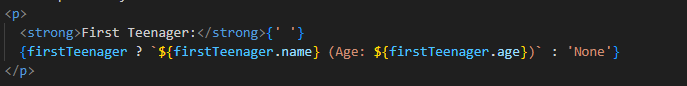
Ex4-1.

* Find the first person off the people array is teenager (age >=10 and age <=20)

B1: Tạo 1 biến firstTeenager, sử dụng hàm find để tìm kiếm và trả về phần tử đầu tiên trong mảng thoả mãn điều kiện là độ tuổi nằm trong khoảng từ 10 đến 20. Nếu không sẽ trả về undefined.



B2: First Teenager sẽ hiện thị tên và độ tuổi của người đầu tiên là thiếu niên, nếu không có sẽ hiện thị “None”.



B3: nhập lệnh npm start và kết quả sẽ được hiện thị

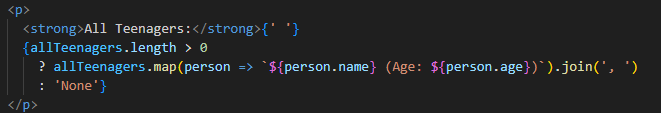


* Find the all person of the people array is teenager (age >=10 and age <=20)

B1: Tạo 1 biến allTeenagers để tìm tất cả các phần tử thoả mãn điều kiện nằm trong khoảng từ 10-20. Sử dụng hàm filter để lọc ra các phần tử thoả mãn các điều kiện. Biến allTeenagers sẽ lưu các kết quả trong mảng people có độ tuổi từ 10-20.



B2: All Teenagers sẽ liệt kê tất cả tên và độ tuổi của những người là thiếu niên. Nếu không có sẽ hiện thị “None”.



B3: nhập lệnh npm start và kết quả sẽ được hiện thị

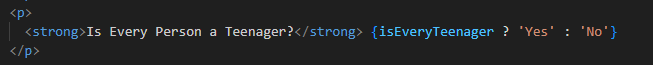


* Check if every person of the people array is teenager (age >=10 and age <=20), which should return true or false

B1: Tạo 1 biến isEveryTeenager. Sử dụng hàm every để kiểm tra xem tất cả các phần tử trong mảng có thoả mãn điều kiện hay không. Nếu có ít nhất 1 người trong độ tuổi từ 10-20 thì hàm sẽ trả về true hoặc ngược lại là false.



B2: Is Every Person a Teenager sẽ kiểm tra xem tất cả mọi người có phải là thiếu niên hay không. Nếu đúng thì kết quả sẽ hiện thị “Yes” hoặc “No”.

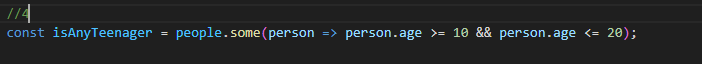


B3: nhập lệnh npm start và kết quả sẽ được hiện thị

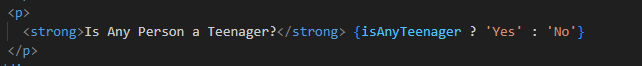


* Checks if any person of the people array is teenager (age >=10 and age <=20), which should return true or false.

B1: Tạo biến isAnyTeenager. Sử dụng hàm some để kiểm tra xem có ít nhất 1 phần tử nào thoả mãn điều kiện không. Nếu có thì hàm sẽ trả về true hoặc ngược lại là false. Biến isAnyTeenager sẽ là true nếu có ít nhất 1 người là thiếu niên, nếu không sẽ là false.



B2: Is Any Person a Teenager sẽ kiểm trả xem có ít nhất 1 người là thiếu niên, nếu có sẽ hiện thị “Yes” hoặc ngược lại là “No”.



B3: nhập lệnh npm start và kết quả sẽ được hiện thị



Ex4-2.

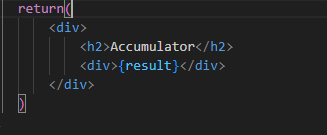
* Applies a function passed as the first parameter against an accumulator and each element in the array (from left to right), thus reducing it to a single value. The initial value of the accumulator should be provided as the second parameter of the reduce function.

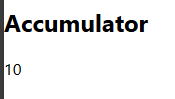
B1: Tạo 1 biến là Accumulator và khai báo các phần tử trong mảng array



B2: Khai biến result, array sử dụng phương thức .reduce() (phương thức này dùng để thu gọn mảng thành 1 giá trị duy nhất và ở đây sẽ là tính tổng tích lũy các phần tử. a là giá trị tích lũy, b là giá trị hiện tại của mảng, 0 là giá trị khởi tạo a ban đầu sẽ = 0. Sau khi tính xong sẽ lưu vào result

  
B3: Trả về giao diện in ra result sẽ cho ra kết quả

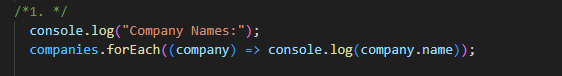




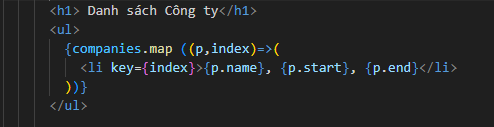
Ex4-3.

* Print the name of each company using forEach

B1: Sử dụng hàm forEach để in ra tên của các công ty ở trong mảng companies. Đây là cách để duyệt qua các phần tử trong mảng và thực hiện hành động với từng phần tử.



B2: Để hiện thị danh sách tất cả các công ty thì sử dụng map() để tạo ra các phần tử <li> cho từng công ty. Mỗi công ty sẽ hiện thị các thông tin như tên, ngày bắt đầu và ngày kết thúc.

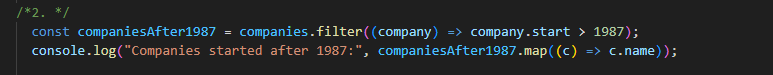


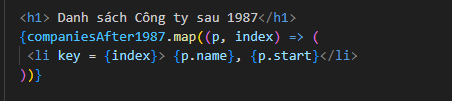
B3: Đây là giao diện được hiện thị sau khi khởi chạy



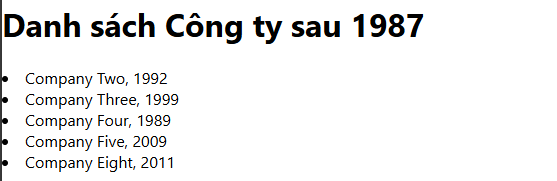
* Print the name of each company that started after 1987

B1: Tạo biến companiesAfter1987. Sử dụng hàm filter để lọc các công ty có ngày bắt đầu lớn hơn 1987. Sau đó sử dụng map(), ta chỉ lấy tên các công ty này và in ra.



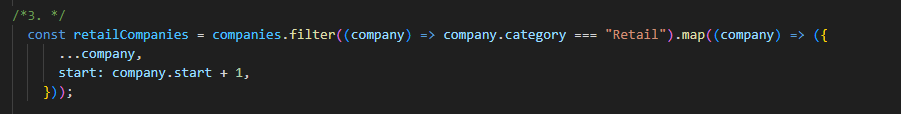
B2: Cũng như câu trên, để hiện thị danh sách tất cả các công ty thì sử dụng map() để tạo ra các phần tử <li> cho từng công ty. Mỗi công ty sẽ hiện thị các thông tin như tên và ngày bắt đầu.

B3: Đây là giao diện được hiện thị sau khi khởi chạy

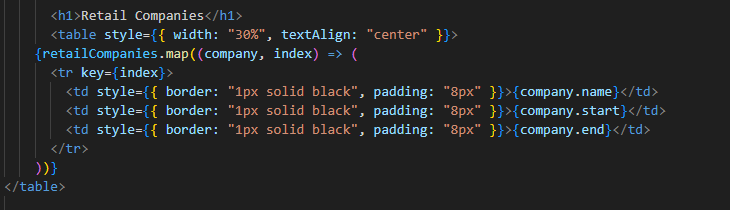


* Get only the companies that have category Retail, increment their start by 1 and append in the DOM a div that has the name, the category, the start and the end in paragraphs elements

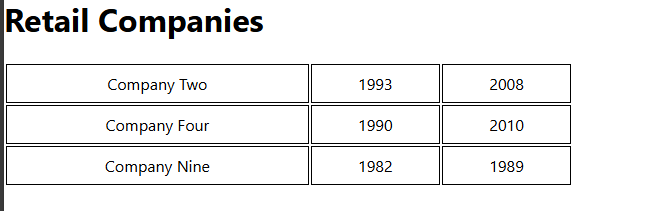
B1: Tạo biến retailCompanies. Sử dụng filter() để lọc các công ty có thể loại “Retail”. Sau đó, dùng map() để tạo một bản sao của mỗi công ty và tăng giá trị năm bắt đầu lên 1.



B2: Sử dụng <table> để hiện thị các công ty trong thể loại “Retail”. Mỗi công ty có một dòng <tr> và <td> chứa tên, ngày bắt đầu và ngày kết thúc.



B3: Đây là giao diện được hiện thị sau khi khởi chạy

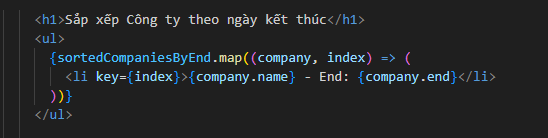


* Sort the companies based on their end date in ascending order

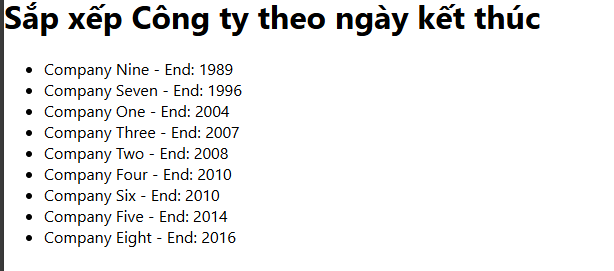
B1: Tạo biến sortedCompaniesByEnd. Dùng phương thức sort() để sắp xếp các công ty theo năm kết thúc (end). Để tránh thay đổi dữ liệu gốc, ta sao chép mảng companies bằng cách sử dụng spread operator ([...]).



B2: Để hiện thị danh sách tất cả các công ty được sắp xếp theo năm kết thúc thì sử dụng map() để tạo ra các phần tử <li> cho từng công ty. Mỗi công ty sẽ hiện thị các thông tin như tên và ngày kết thúc.



B3: Đây là giao diện được hiện thị sau khi khởi chạy

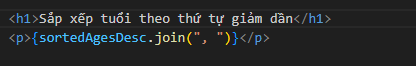


* Sort the ages array in descending order

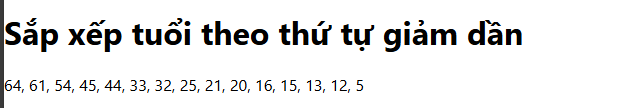
B1: Tạo biến sortedAgesDesc. Mảng ages được sao chép và sắp xếp theo thứ tự giảm dần bằng sort().



B2: Độ tuổi được hiển thị dưới dạng một chuỗi, phân tách bằng dấu phẩy, theo thứ tự giảm dần.

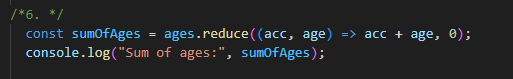


B3: Đây là giao diện được hiện thị sau khi khởi chạy



* Print the sum if you add all the ages using reduce

B1: Tạo biến sumOfAges. Dùng reduce() để tính tổng tất cả các giá trị trong mảng ages. Hàm này nhận vào một hàm phụ có hai tham số: acc (tổng tạm thời) và age (mỗi độ tuổi), cộng dồn các giá trị của độ tuổi lại với nhau.



B2: Tổng độ tuổi được hiển thị trong một đoạn văn (<p>).



B3: Đây là giao diện được hiện thị sau khi khởi chạy

