문제 보기

주제

당신은 2020년 상반기에 고양이 사진 검색 사이트를 만들었습니다만, 관리를 잘못하여 사이트가 폐쇄되었습니다.
당신은 지난 실수를 발판 삼아 하반기에는 더 좋은 사이트를 만들려고 하는데요. 이번에는 단순히 사진만을 보여주기보단, 사용성을 개선해 유저가 검색을 쉽게 할 수 있도록 검색어 **추천 기능**을 넣으려고 합니다. 그리고 가능하다면, 성능을 고려하여 여러 가지 부차적인 일도 하고 싶네요. 3시간 내에 사이트를 만들 수 있을까요? 만들고 싶은 결과물은 다음과 같습니다.

과제 설명

- 기본적인 화면을 구성하는 베이스 코드가 포함되어 있습니다. 이 베이스 코드를 활용해, ES6+ 문법을 이용해 작성해주세요.
- 화면을 구성하는 요소를 컴포넌트로 만들어서 코딩합니다.
  - 컴포넌트는 function 문법, class 문법 어느 쪽을 사용해도 괜찮습니다.
  - 각 컴포넌트는 각각의 상태 값을 가지고 있고, render 함수를 통해 현재 상태를 렌더링 합니다.
  - render 함수는 파라미터를 넘겨받는 것 없이 해당 컴포넌트의 상태 변수들만 갖고 렌더링이 일어나도록 설계해야 합니다.
  - DOM 접근은 최소화해야 합니다.
  - 컴포넌트 간의 결합은 최대한 느슨해야 합니다.
- 과제는 chrome, edge, firefox, safari 등 모던 브라우저 최신 버전에서 동작하는 것을 원칙으로 합니다.

수행 기술

- JavaScript(ES6+)
- 설치되어있는 모듈(node_modules) 외에 다른 외부 라이브러리는 사용하지 않도록 합니다. 예를 들어 jQuery, Webpack, Lodash, Axios, Angular, React, Vue, Immutable-js, Ramda 등을 사용할 수 없습니다.

요구사항

중요 말머리로 `[필수]` 가 붙은 요구사항은 반드시 수행해주세요.

검색어 추천

- **[필수]** 추천 검색어 API를 이용해 입력란에 추천 검색어를 보여주세요. 추천 검색어는 엔터를 치는 등 별도의 행위가 없을 때에도 자동으로 보여야 합니다.
  - `keywords` 라는 class 명으로 스타일링 되어있습니다. 자세한 내용은 index.html과 style.css를 참고하시리 바랍니다.
  - API에 대한 설명은 하단에 있습니다.
  - API에서 에러가 나거나, 추천 검색어가 없는 경우에도 검색 기능을 사용하는 데에는 문제가 없어야 합니다.

검색어 추천 - 사용성 개선

- 키보드와 마우스를 이용해서 추천 검색어를 선택할 수 있게 만들어 주세요.
  1. esc를 누르면 추천 검색어 창이 닫힙니다.
  2. 키보드의 위, 아래 키를 누르면 추천 검색어 하이라이트가 옮겨지고 엔터를 누르면 하이라이트가 위치한 검색어가 입력창에 반영되고 사진이 검색됩니다.
  3. 마우스로 다른 곳을 클릭하여 input이 focus를 잃어버리는 경우 추천 검색어 창이 닫힙니다.
  4. 마우스로 추천 검색어를 누르면 커서가 위치한 검색어가 입력창에 반영되며 사진이 검색됩니다.
- 추천 검색어가 로딩되는 중임을 알리는 UI적 처리를 해주세요.

검색

- **[필수]** 검색 시 API로 받은 고양이 사진이 화면에 렌더 되어야 합니다.

검색 - 사용성 개선

- 검색 중 에러가 발생한 경우, 에러가 발생했다는 것을 알리는 UI적 처리를 해주세요.
- 검색 결과가 로딩되는 중임을 알리는 UI적 처리를 해주세요.
- 검색이 일어나면 url 뒤에 `?q={검색어}` 를 붙입니다.
  - 페이지 url에 검색어가 존재하는 경우 페이지에 진입하자마자 해당 검색어로 검색된 결과가 나오도록 합니다.

퍼포먼스 향상

- 검색어별 추천 검색어를 로컬에 캐싱해서 사용하도록 합니다.
- 추천 검색어 API 호출 중 새로운 검색어 입력이 감지되면 기존의 ajax 요청을 취소하고 새로운 검색어를 기준으로 API 요청을 보내주세요.
- Debounce를 구현합니다. 이를 통해 검색어가 입력될 때마다 서버에 요청이 일어나지 않게 합니다.
- 키워드별 검색 결과를 캐싱하여 사용합니다. 단, 캐싱된 데이터는 브라우저를 닫으면 사라져야 합니다.

코드 구조 관련

- 각 컴포넌트 간의 연동은 가급적 직접 호출하지 말고, App 컴포넌트를 만든 뒤 이 컴포넌트가 콜백 함수를 이용해 조율하는 형태로 만

듭니다.

- ES6 module 형태로 코드를 변경합니다.
  - `webpack` , `parcel` 과 같은 번들러를 사용하지 말아 주세요.
  - 해당 코드 실행을 위해서는 `http-server` 모듈을(로컬 서버를 띄우는 다른 모듈도 사용 가능) 통해 `index.html` 을 띄워야 합니다.
  - 터미널에서 `npm start` 명령어를 이용해 로컬 서버를 띄운 후 작업을 합니다.
- API fetch 코드를 `async` , `await` 문을 이용하여 수정해주세요. 해당 코드들은 에러가 났을 경우를 대비해서 적절히 처리가 되어있어야 합니다.
  - 에러 발생을 알리는 UI적 처리까지 하면 더욱 좋습니다.
  - 서버에서는 아주 가끔 데이터 모양이 이상하게 내려오기도 합니다. 이를 대응해주세요.
- API의 status code에 따라 에러 메시지를 분리하여 작성해야 합니다.
- 컴포넌트 내부의 함수들이나 Util 함수들을 작게 잘 나누어주세요.

API 설명

1. 추천 검색어 API

URL

```
GET https://jf3iw5iguk.execute-api.ap-northeast-2.amazonaws.com/dev/api/cats/keywords?q={keyword}
```

Parameter

- q: (필수) 검색어

Response

다음과 같은 추천 검색어로 구성된 배열

```
["name1", "name2", "name3"]
```

예시

검색창에 **헤어**를 입력하면 아래와 같이 요청합니다.

```
curl 'https://jf3iw5iguk.execute-api.ap-northeast-2.amazonaws.com/dev/api/cats/keywords?q=%ED%97%A4%EC%96%B4'

// response
[
  "브리티쉬 롱헤어",
  "브리티쉬 숏헤어",
  "컬러포인트 숏헤어",
  "엑조틱 숏헤어",
  "오리엔탈 숏헤어"
]
```

2. 검색 API

URL

```
GET http://jf3iw5iguk.execute-api.ap-northeast-2.amazonaws.com/dev/api/cats/search?q={keyword}
```

Parameter

- q: (필수) 검색어

Response

다음과 같이 검색결과로 이루어진 배열

```
{
  "data": [
    {
      "id": "id",
      "url": "url",
      "name": "name"
    }
  ]
}
```

예시

`브리티쉬 롱헤어` 로 검색했을 경우

```
curl 'http://jf3iw5iguk.execute-api.ap-northeast-2.amazonaws.com/dev/api/cats/search?q=%EB%B8%8C%EB%A6%AC%ED%8B%B0%I
```

대략 아래와 같은 응답이 내려옵니다.


```
{
  "data": [
    {
      "id": "MTc5NDU2MQ",
      "url": "https://cdn2.thecatapi.com/images/MTc5NDU2MQ.jpg",
      "name": "British Longhair / 브리티쉬 롱헤어"
    },
    {
      "id": "5cc",
      "url": "https://cdn2.thecatapi.com/images/5cc.jpg",
      "name": "British Longhair / 브리티쉬 롱헤어"
    },
    {
      "id": "bbr",
      "url": "https://cdn2.thecatapi.com/images/bbr.jpg",
      "name": "British Longhair / 브리티쉬 롱헤어"
```