# Lab 4

**Due** Mar 28 by 11:59pm **Points** 100 **Submitting** a file upload **File Types** zip

# CS-554 Lab 4

## React Marvel API

For this lab, you will create a Marvel API Single Page Application using React. **You can use either Class Based Components for this Lab or Function Components and Hooks. I would advise using functional components with hooks as that is what is becoming the standard. Class Components are still good to know because many existing and legacy systems still use them but functional components are now more the norm for new applications.**

You will be creating a Single Page Application using React that implements the following routes using **create-react-app** (https://github.com/facebookincubator/create-react-app) and **react router 6** (https://reactrouter.com/docs/en/v6/getting-started/overview).

You will be using the **Marvel API** (https://developer.marvel.com). You will need to register and sign up for an API key. You will not be able to make requests to the API without signing up and getting an **API key** (https://developer.marvel.com/account). You will use the **Characters** (https://gateway.marvel.com/v1/public/characters?ts=1592417963445&apikey=a8f9ccf932bf29fd379ef00e11668673&hash=f061194023791a1593a0ea861a27da67), **Comics** (https://gateway.marvel.com/v1/public/comics?ts=1592417963445&apikey=a8f9ccf932bf29fd379ef00e11668673&hash=f061194023791a1593a0ea861a27da67), and **Series** (https://gateway.marvel.com/v1/public/series?ts=1592417963445&apikey=a8f9ccf932bf29fd379ef00e11668673&hash=f061194023791a1593a0ea861a27da67) listings. Please look at the data returned so you know the schema of the data and the objects it returns (the links to Characters, Comics and Series above work but using my API key. DO NOT use my API key. Please register for your own. You will need to compose the URL with your API key, a ts (time stamp) and a hash.

You can use the following code to construct the URL. (this example is for characters, you would do the same for series and comics) you can read more about AUTHORIZING AND SIGNING REQUESTS from the link below

**https://developer.marvel.com/documentation/authorization** (https://developer.marvel.com/documentation/authorization)

```
const md5 = require('blueimp-md5');
const publickey = 'your_public_key(API KEY) from Marvel dev portal';
const privatekey = 'your private key from Marvel dev portal';
const ts = new Date().getTime();
const stringToHash = ts + privatekey + publickey;
const hash = md5(stringToHash);
const baseUrl = 'https://gateway.marvel.com:443/v1/public/characters';
const url = baseUrl + '?ts=' + ts + '&apikey=' + publickey + '&hash=' + hash;
```

# Pages

### /

The root directory of your application will be a simple page explaining the purpose of your site (to talk about Marvel, the API etc..).

This page will have a `<Link>` to the Characters Listing ( `/characters/page/0` ), The Comics Listing ( `/comics/page/0` ), and the Series Listing ( `/series/page/0` )

### /characters/page/:page

This route will render a paginated list of Marvel Characters. It will use the :page param to determine what page to request from the API. If you are on page 0, you will show a button to go to the *next* page. If you are on page 1+, you will show a *next* button and a *previous* button, that will take you to the previous page. **If the Page does not contain any more Characters in the list, the SPA will redirect to a 404 page.**

### /characters/:id

This route will show details about a single Character. **If the Character does not exist, the SPA will redirect to a 404 page.**

### /comics/page/:page

This route will render a paginated list of comics. It will use the :page param to determine what page to request from the API. **If the Page does not contain any more comics in the list, the SPA will redirect to a 404 page.**

### /comics/:id

This route will show details about a single comic. **If the comic does not exist, the SPA will redirect to a 404 page.**

### /series/page/:page

This route will render a paginated list of series. It will use the :page param to determine what page to request from the API. You will also show some sort of pagination UI (see below). **If the Page does not contain any more series in the list, the SPA will redirect to a 404 page.**

### /series/:id

This route will show details about a single series. **If the series does not exist, the SPA will redirect to a 404 page**.

# Pagination

Pagination can be done using an external library from NPM, such as **react-bootstrap's pagination** (https://react-bootstrap.github.io/components/pagination/) or **react-paginate** (https://github.com/AdeleD/react-paginate).

The minimum you must provide for a pagination UI:

- **If you are on page 0, you will show a button to go to the *next* page.**
- **If you are on page 1+, you will show a *next* button and a *previous* button, that will take you to the previous page.**

**Pagination HINT: Look at the data.. You can also do the pagination by using the following fields in the data: "offset": 0, "limit": 20, "total": 1493, "count": 20**

# HTML, Look, and Content

Besides the specified settings, as long as your HTML is valid and your page runs passes a **tota11y (http://khan.github.io/tota11y/)** test, the general look and feel is up to you. If you want to update it, you may; if you do not, that is fine.

I do, however, recommend using **React-Bootstrap (https://react-bootstrap.github.io/getting-started/introduction/)** to make your life easier if you need any UI elements.

As for the data that you chose to display on the details pages, that is up to you. You should show as much of the data as possible. For example, a single character has comics associated with it. It would be nice if you add links to the comics details page for a comic that a character appears in, also perhaps link to the characters that are in a comic on the comic details page.

# Extra Credit

If you add search functionality either to characters, comics or series you will get 5 points extra for each that you implement so you have the potential to get 15 points extra credit.

# General Requirements

1. Remember to submit your `package.json` file but **not** your `node_modules` folder.
2. Remember to run HTML Validator and tota11y tests!
3. Remember to have fun with the content.
4. Remember to practice usage of async / await!