# ASSIGNMENT 2:
# SIMPLE PICTURE SYSTEM IN REACT

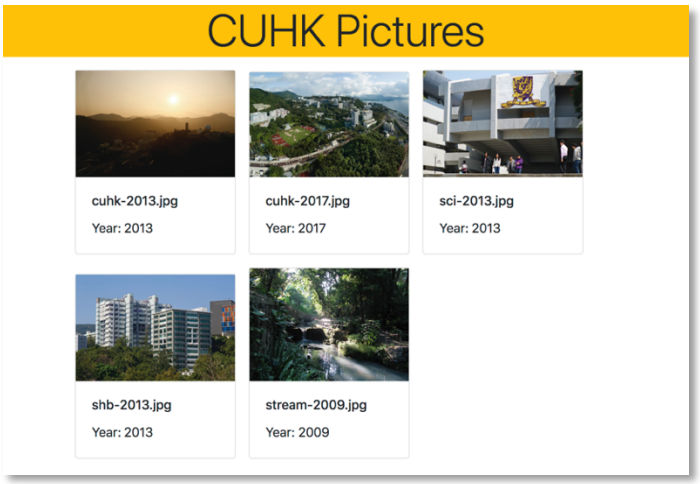
RELEASED: **1 MAR 2021**
DUE: **23 MAR 2021 23:59**

## SYNOPSIS

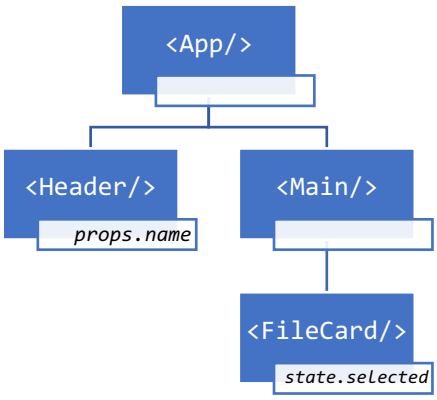
You are going to set up a simple system for showing pictures using React. It includes a slideshow feature with adjustable speed, as well as links for routing within the single-page application.

## BASIC SYSTEM (25%)

*(Done in Lab 4)* Build a simple picture system to display an array of picture files. The file names and relevant information of each file are stored as a variable in an JS array, ***as specified in Lab 4***.



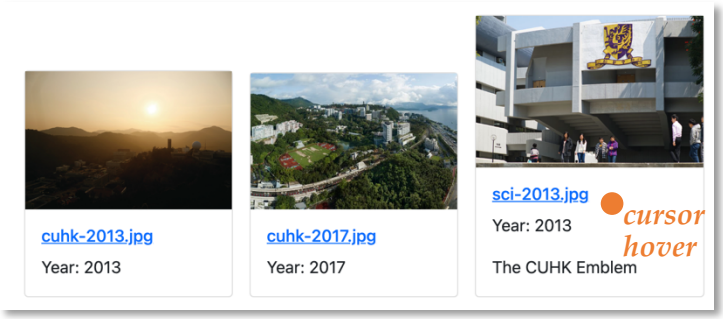

The React components in the basic system are structured in this manner:




The pictures are shown in a group of ***Bootstrap*** cards with width 200px each, using the `map()` function in component **`<FileCard/>`**. The filename and year are shown along with the picture.

## MOUSEOVER EVENT (15%)

In Lab 4, you set up an **`onClick`** event to enlarge the picture card to 100% width when clicked. The *file remarks* are shown as well. This was done using an event handler **`handleClick()`** to change the state **`selected`** to the index of the picture.

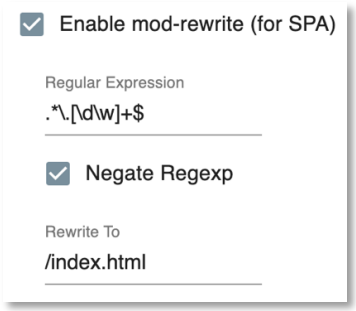



Now, change this to an **`onMouseOver`** event so that when the cursor hovers over the picture card, and enlarge the card to 220px of width instead of 100%. Return to normal when the cursor moves away using **`onMouseOut`**.

## ROUTING MENU (20%)

*(Tried in Lab 5)* Add a navigation menu for routing within the page. Show *three* items:

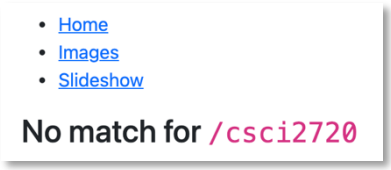
- ***Home***: with path **`/`** should show component **`<Home/>`**
- ***Images***: with path **`/file`** should show component **`<Main/>`** (*i.e., the original picture system*)
- ***Slideshow***: with path **`/slideshow`** should show component **`<Slideshow/>`**




To allow all requests to be handled by **`index.html`**, do these:

1. In ***Web Server for Chrome***, tick *Advanced Options* >> ***Enable mod-rewrite (for SPA)***
2. In the files `index.html` and `app.jsx`, ***use the root path*** by changing all mentioning of **`app.jsx`** to **`/app.jsx`** and **`images`** to **`/images`**

Now even when refreshing a page or typing in a URL with path directly, the correct routed component will be shown. Set up a **`<NoMatch/>`** component to catch all URL not found.




## SLIDESHOW (20%)

In component **`<Slideshow/>`**, display the following:

1. 4 buttons: Start slideshow, Stop slideshow, Slower, Faster
2. Image showing the `[0]` item in the array
3. Image file name

Set up *four* event handlers to deal with the buttons when they are clicked.

- ***Start slideshow***: the next picture (and filename) in the array should be shown one by one, and looping back to the beginning after each round (default interval: every 1000ms)

- ***Stop slideshow***: there should be no more change in the picture
- ***Slower***: the changing interval would be doubled *(×2)*
- ***Faster***: the changing interval would be halved *(÷2)*

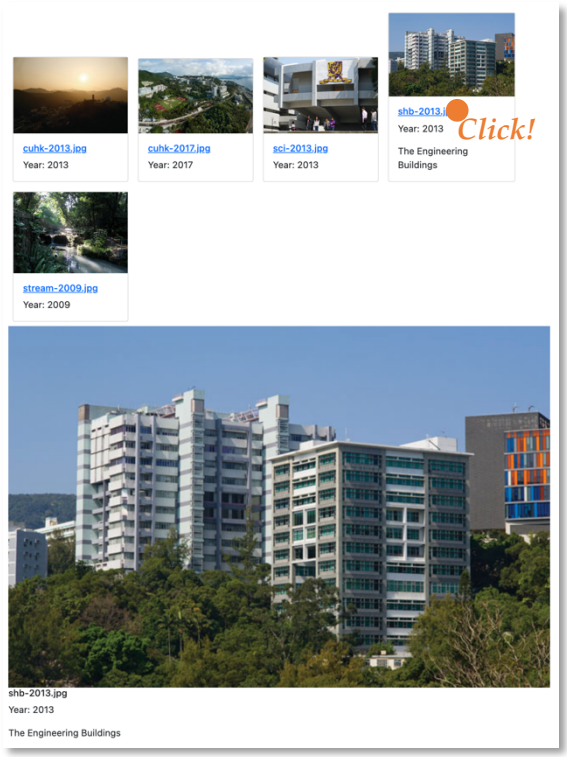

There should be *at least* two state variables being changed to affect the React image display, e.g. **`currentImageID`** and **`currentInterval`**. They should be accessible in *Developer Tools >> Components >> … >> Slideshow*. *You may use more states if needed.*

You may design the page layout to your liking, as long as all required items are clearly shown. We will not test the ***Slower*** and ***Faster*** buttons before starting the slideshow.

## URL PARAMETERS FOR NEW COMPONENT (15%)

*(Tried in Lab 4)* Inside **`<FileCard/>`**, set up a new **`<Router>`** so that the listed picture file names will become a **`<Link>`**. It should link to the path `/file/0`, `/file/1`, etc. basing on the array index of the picture.

Under all pictures, show a new component using **`<Switch>`** and **`<Route>`**. Inside it, show the linked image in 100% width, with the *filename*, *year*, and *remarks*, similar to what is shown here on the right:




***Note**: the* `<NoMatch/>` *component needs not catch* `/file/*`

## COMPONENT DIAGRAM (5%)

Inside **`<Home/>`**, draw a ***tree diagram*** based on *React components you create (not from libraries)* within **`app.jsx`**. There is no requirement on formats and software to use, but a clear hierarchical relationship must be shown in the diagram. List also the ***props*** and ***states*** in each component.

## BONUS FEATURES

Some transition animations can be made for the change of React components, using the library ***React Transition Group*** (*https://reactcommunity.org/react-transition-group/*). You may introduce simple transitions that can be seen obviously, and state in `<Home/>`. ***Use only the CDN method.***

This task is NOT a requirement of the assignment. But if you successfully do it, it contributes to a few bonus points for your course grade:

- *Transitions for both slideshow and router =* ***0.5 points***
- *Transition for only slideshow =* ***0.25 points***

## LIBRARIES, FEATURES AND FRAMEWORKS

You should be using the latest version of React, ReactDOM, Babel, and ReactRouterDOM from the official CDN (*https://reactjs.org/docs/cdn-links.html*) with no other libraries:

- `<script crossorigin src="https://unpkg.com/react/umd/react.development.js"></script>`
- `<script crossorigin src="https://unpkg.com/react-dom/umd/react-dom.development.js"></script>`
- `<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>`
- `<script src="https://unpkg.com/react-router-dom/umd/react-router-dom.min.js" crossorigin></script>`

You should use the ***development version*** for clearer error/warning messages.

Lab 4 has incorporated the use of ***Bootstrap*** and you can keep it. Other than anything specified, there is no cosmetic requirement for this assignment. You are welcome to implement extra styles and features at your own ability. You should contain the code in the same structure as in Lab 4, i.e. not using `create-react-app`. *It is fine for extra components or different hierarchy from required here.*

## SUBMISSION

We will only visit your web page submission using ***Google Chrome*** (almost-latest versions) with extension ***Web Server for Chrome***. Please utilize the ***Developer Tools*** with ***React Developer Tools*** for debugging of your code.

Plagiarism is heavily penalized. Do not attempt sharing code other than those given in labs. Please read this article carefully: *http://www.cuhk.edu.hk/policy/academichonesty*

Include your full name and student ID in ***all code files*** using comments. Zip your files *excluding CUHK images*, and submit it on the course site on Blackboard. ***Only these files should be included:***

- 1 `.html` file
- 1 `.jsx` file
- *1 diagram image*

*Submit NO other image files.* We will use the same set as in Lab 4 in an **`images/`** folder.