ASSIGNMENT 2:
SIMPLE IMAGE SYSTEM IN REACT

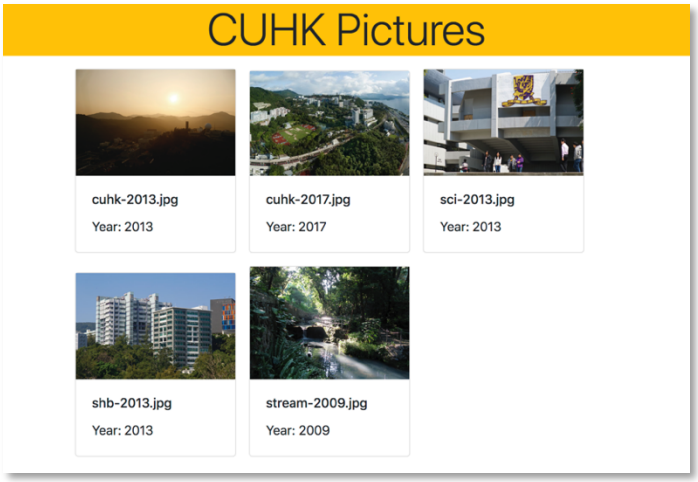
RELEASED: **28 FEB 2022**
DUE: **14 MAR 2022 23:59**

SYNOPSIS

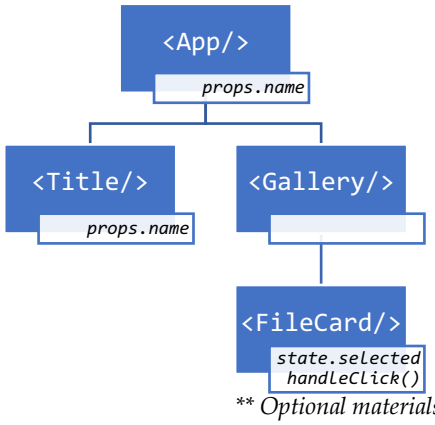
You are going to set up a simple system for showing images using React. It includes a slideshow feature with adjustable speed, as well as links for routing within the single-page application.

BASIC SYSTEM (25%)

(Done in Lab 4) Build a simple image system to display an array of image 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:

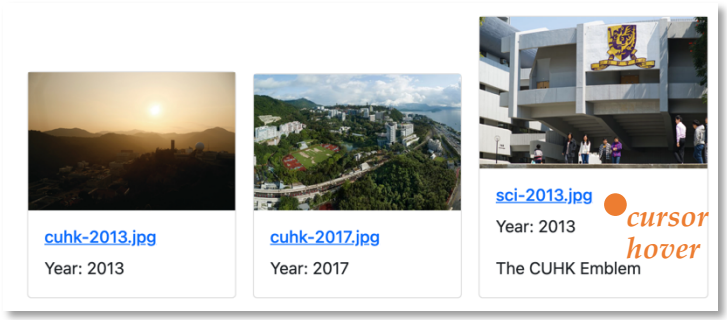


** *Optional materials*

The images are shown in a group of ***Bootstrap*** cards with width 200px each, using the `map()` function to render multiple **`<FileCard/>`** components. The filename and year are shown along with the image.

MOUSEOVER EVENT (25%)

In the optional materials in Lab 4, you are instructed to set up an **`onClick`** event to enlarge the image card to 100% width when clicked. The file remarks are shown as well. This is be done using the **`onClick`** event to change the state **`selected`** to the index of the image.



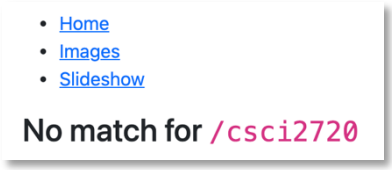
Now, change this to an **`onMouseOver`** event so that when the cursor hovers over the image card, and enlarge the card to ***220px*** of width instead of 100%. Return to normal when the cursor moves away using **`onMouseOut`**.

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 **`/gallery`** should show component **`<Gallery/>`** (*i.e., basic image system*)
- ***Slideshow***: with path **`/slideshow`** should show a new component **`<Slideshow/>`**

In ***Web Server for Chrome***, all access should be redirected to **`index.html`**. 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 that cannot 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 image (and filename) in the array should be shown one by one, and looping back to the beginning after each round (default interval: every 1500ms)
- ***Stop slideshow***: there should be no more change in the image
- ***Slower***: the changing interval would be increased by 250ms
- ***Faster***: the changing interval would be decreased by 250ms

  Note: the interval should not be decreased lower than 250ms

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 are recommended to use a class component to implement `<Slideshow/>` for simplicity with states handling.

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.

COMPONENT DIAGRAM (10%)

Inside **`<Home/>`**, display a ***tree diagram*** basing on the React components *created by you (excluding those from libraries e.g., `<Route/>`)* within **`app.jsx`**. The diagram should be a png/jpg image, and there is no limitation on the software to use for creating the diagram image. A clear hierarchical relationship must be shown. List also the ***props***, ***states, and event handlers*** in each component.

CHALLENGE REQUIREMENTS (20%)

For ESTR2106 students: This part is compulsory. This assignment has a full score at 120%.

For CSCI2720 students: If you finish this part correctly, a 0.5% bonus will be given to the "Assignment" course assessment, capped at 30% maximum in the course total.

In <Home/>, show a short paragraph to briefly describe the work here, with ~100 words.

1. To enhance development, enable **`React.StrictMode`** for your `<App/>` component (with its children). See: *https://reactjs.org/docs/strict-mode.html*
2. Other than `<Slideshow/>`, all React components in the system must be built as React ***functional components***. Be careful with the use of props, states, and event handlers. *Yet, this is not required for the `<Slideshow/>` component.*
3. All the images in the `<Gallery/>` should be ***lazy loaded***. Please use the `IntersectionObserver` method, without using `loading="lazy"`. You may refer to an example here: *https://codepen.io/chuckjee/pen/bGYmgeM*. Instead of as an event handler for `DOMContentLoaded`, the observer setup can be implemented with useEffect() in `<Gallery/>`.

LIBRARIES, FEATURES AND FRAMEWORKS

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

- `<script src="https://unpkg.com/react/umd/react.development.js" crossorigin></script>`
- `<script src="https://unpkg.com/react-dom/umd/react-dom.development.js" crossorigin></script>`
- `<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>`
- `<script src="https://unpkg.com/history@5/umd/history.development.js" crossorigin></script>`
- `<script src="https://unpkg.com/react-router@6/umd/react-router.development.js" crossorigin></script>`
- `<script src="https://unpkg.com/react-router-dom@6/umd/react-router-dom.development.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. Check that the file name and formats are correct. 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 a similar set of images as in Lab 4 in an **`images/`** folder.