

Diploma Web Application Development: Introduction

ICT50220 Diploma of Information Technology(Front-End Web Development)

Code	Title
ICTWEB517	Create web-based programs
ICTWEB546	Validate application design against specifications

Recap

Last week we:

1. Installed our Next JS
2. Reviewed out NASA Apod route
3. Discussed dynamic routes

How do 1-3 translate to Next JS

Global CSS & the DOM

When we use the globals.css file and import it in the root layout file.

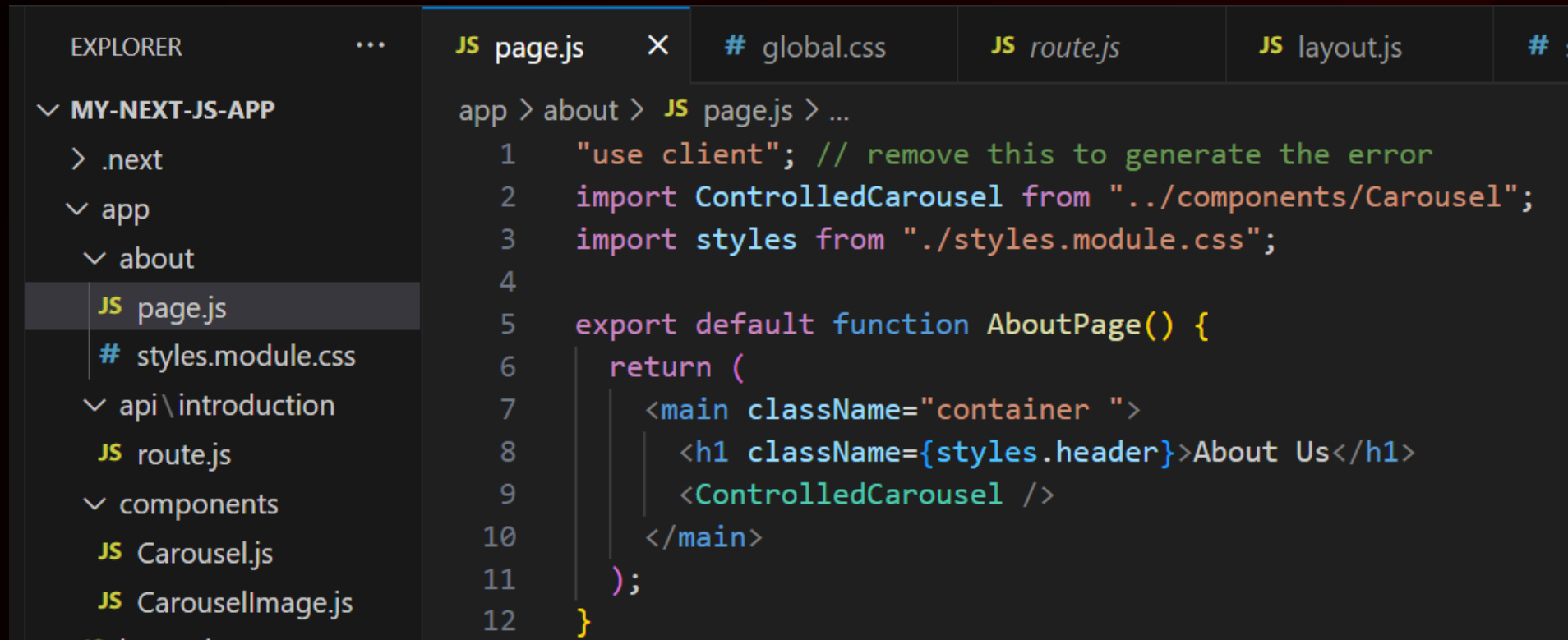
These styles apply globally across the entire app

```
/* styles/globals.css */
@import "~bootstrap/dist/css/bootstrap.min.css";

body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
}
```

CSS for Modules

If we need to we co-locate specific CSS for our route.



The screenshot shows a code editor interface. On the left is the 'EXPLORER' sidebar with a file tree for 'MY-NEXT-JS-APP'. The tree includes a '.next' directory, an 'app' directory (expanded), and an 'about' subdirectory. Inside 'about', 'page.js' is selected. Below 'about', there is a file named 'styles.module.css'. Other files like 'route.js', 'Carousel.js', and 'CarouselImage.js' are visible under a 'components' directory. The main editor area shows the content of 'page.js'. The code includes a comment to remove 'use client' to avoid an error, imports for 'ControlledCarousel' and 'styles', and a default export function 'AboutPage' that returns a JSX element with a main container, a header, and a carousel.

```
EXPLORER  ...
  ▼ MY-NEXT-JS-APP
    > .next
    ▼ app
      ▼ about
        JS page.js
        # styles.module.css
      ▼ api\introduction
        JS route.js
      ▼ components
        JS Carousel.js
        JS CarouselImage.js
        JS ...

JS page.js  X  # global.css  JS route.js  JS layout.js  # s
app > about > JS page.js > ...
1  "use client"; // remove this to generate the error
2  import ControlledCarousel from "../components/Carousel";
3  import styles from "../styles.module.css";
4
5  export default function AboutPage() {
6    return (
7      <main className="container ">
8        <h1 className={styles.header}>About Us</h1>
9        <ControlledCarousel />
10     </main>
11   );
12 }
```

CSS Frameworks & Alternatives

We also have options to:

- use SASS to create variables, mixin logic & nested CSS etc
- tap into JavaScript `.style` property and use the more complex CSS-in-JS
- lastly, we can use CSS frameworks

We have used Bulma to this point but let's install Bootstrap in our application.

<https://nextjs.org/docs/app/building-your-application/styling>

Activity 1: Installing Bootstrap & CSS in Next JS

1. Open my-nextjs-app

```
cd my-nextjs-app
```

2. Install the dependencies, then run the project

```
npm install  
npm run dev
```

Activity 1: Installing Bootstrap

We have two choices here we will look at both:

```
# only the bootstrap css
```

```
npm install bootstrap
```

```
# the widely supported react component library
```

```
npm install react-bootstrap
```

```
npm run dev
```

Activity 1: Update the meta, scripts & dependencies

- whilst we can use bootstrap in single route
- we want it to be available for our entire application.

```
{  
  ▶ Debug  
  "scripts": {  
    "dev": "next dev",  
    "build": "next build",  
    "start": "next start",  
    "lint": "next lint"  
  },  
  "dependencies": {  
    "bootstrap": "^5.3.3",  
    "next": "^14.2.7",  
    "react": "^18.3.1",  
    "react-bootstrap": "^2.10.5",  
    "react-dom": "^18.3.1"  
  }  
}
```


Activity 1: Use the library

1. import the bootstrap minified CSS into the **global.css**
2. import the global.css into the app's root **layout.js**
3. the entire application can now access bootstrap

```
/* styles/globals.css */  
@import "~bootstrap/dist/css/bootstrap.min.css";  
  
/* app/layout.js */  
import "../styles/globals.css";
```

Activity 1: Testing our Bootstrap installation

Add some bootstrap classes to the root layout of the application.

```
// add some bootstrap classes to test the
install

export default function RootLayout({ children
}) {
  return (
    <html lang="en">
      <body className="container">
        <nav className="nav">Navigation</nav>
        {children}
      </body>
    </html>
  );
}
```

Activity 2: Finally getting round to our Theme

Earlier we discussed using a ternary to create a theme.

We discussed using React Context to achieve this.

Bootstrap has some very useful classes for us to use

```
const toggleTheme = () => {  
  setTheme(theme === "light" ? "dark" : "light");  
};
```

```
className={`min-vh-100 ${  
  theme === "light" ? "bg-light text-dark" : "bg-dark text-light"}`  
}
```

Activity 2: Passing a theme using Context

```
const ThemeContext = createContext();
# Ensure all the layouts & pages in the app can access our context
export function ThemeProvider({ children }) {
  # Set the default theme state to light
  const [theme, setTheme] = useState("light");
  # When we want to change update the state
  const toggleTheme = () => {
    setTheme(theme === "light" ? "dark" : "light");
  };
  # Wrap the entire application in the context and pass the state as props
  return (
    <ThemeContext.Provider value={{ theme, toggleTheme }}>
      {children}
    </ThemeContext.Provider>
  );
}
```

Activity 2: Wrapping the Children

- We need to apply our context (& the state it uses) to our root layout.
- This makes both available throughout the application

```
const ThemeContext = createContext();

export function ThemeProvider({ children }) {
  const [theme, setTheme] = useState("light");

  const toggleTheme = () => {
    setTheme(theme === "light" ? "dark" : "light");
  };

  return (
    <ThemeContext.Provider value={{ theme, toggleTheme }}>
      {children}
    </ThemeContext.Provider>
  );
}
```

Client & Server-Side Components

Components can be considered right for the Client side if they use:

- Event listeners
- State
- Manipulate the DOM

NextJS uses server-side components by **default**.

This means **useState** and **useEffect** are not available to us

Component Libraries

To speed up production we can access component libraries, though this comes with dependencies!

- We installed react-bootstrap earlier
- Using the documentation we will implement a Carousel.

<https://react-bootstrap.github.io/docs/components/carousel/>

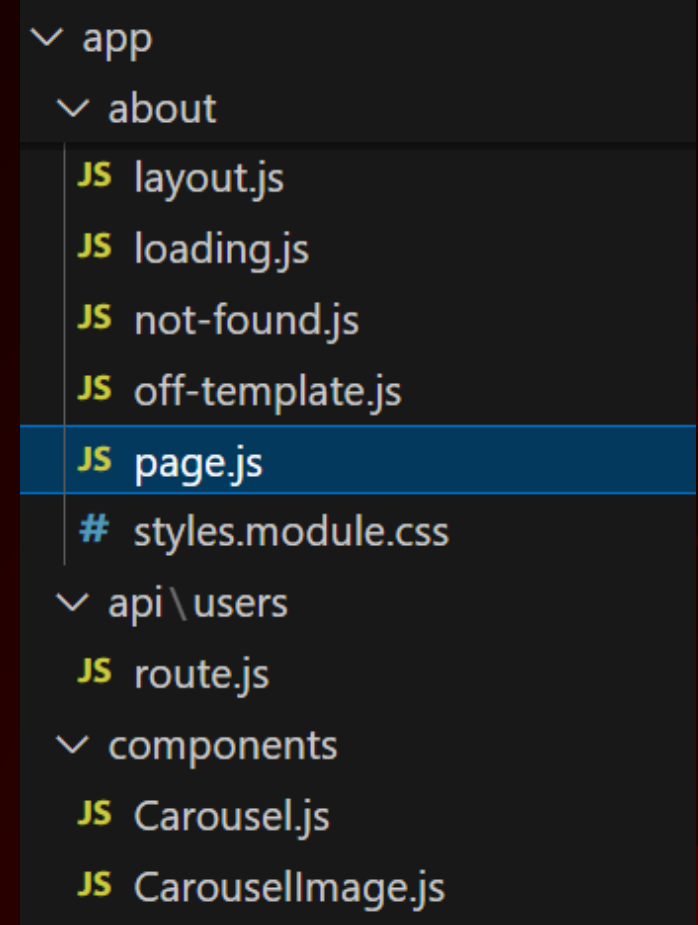
```
▼ app
  ▼ about
    JS layout.js
    JS loading.js
    JS not-found.js
    JS off-template.js
    JS page.js
    # styles.module.css
  ▼ api\users
    JS route.js
  ▼ components
    JS Carousel.js
    JS CarouselImage.js
```

The Carousel

To speed up production we can access component libraries, though this comes with dependencies!

- We installed react-bootstrap earlier
- Using the documentation we will implement a Carousel.

<https://react-bootstrap.github.io/docs/components/carousel/>



Using state to move the carousel

This is very similar in look to our sculpture list.

1. When we click the index is increased by 1
2. This index returns the carousel item from an array
3. Note the default state is 0

```
function ControlledCarousel() {
  const [index, setIndex] = useState(0);

  const handleSelect = (selectedIndex) => {
    setIndex(selectedIndex);
  };

  return (
    <Carousel activeIndex={index} onSelect={handleSelect}>
      <Carousel.Item>
        <CarouselImage text="First" />
        <Carousel.Caption>
          <h3>First slide label</h3>
          <p>Nulla vitae elit libero.</p>
        </Carousel.Caption>
      </Carousel.Item>
    </Carousel>
  );
}
```

Getting the images

The docs' don't specify,
but we can reuse the
react-bootstrap **image**
component

```
import Image from "react-bootstrap/Image";

function CarouselImage({ text }) {
  return (
    <Image
      src={`https://via.placeholder.com/1250x500.png?text=$
           {encodeURIComponent(text)}`}
      alt={text}
      fluid
    />
  );
}

export default CarouselImage;
```

Implement the Carousel in **the /about** route

We can now import the carousel in any of your routes

On this occasion we will use/about.page.js

```
import ControlledCarousel from "../components/Carousel";
import styles from "../styles.module.css";

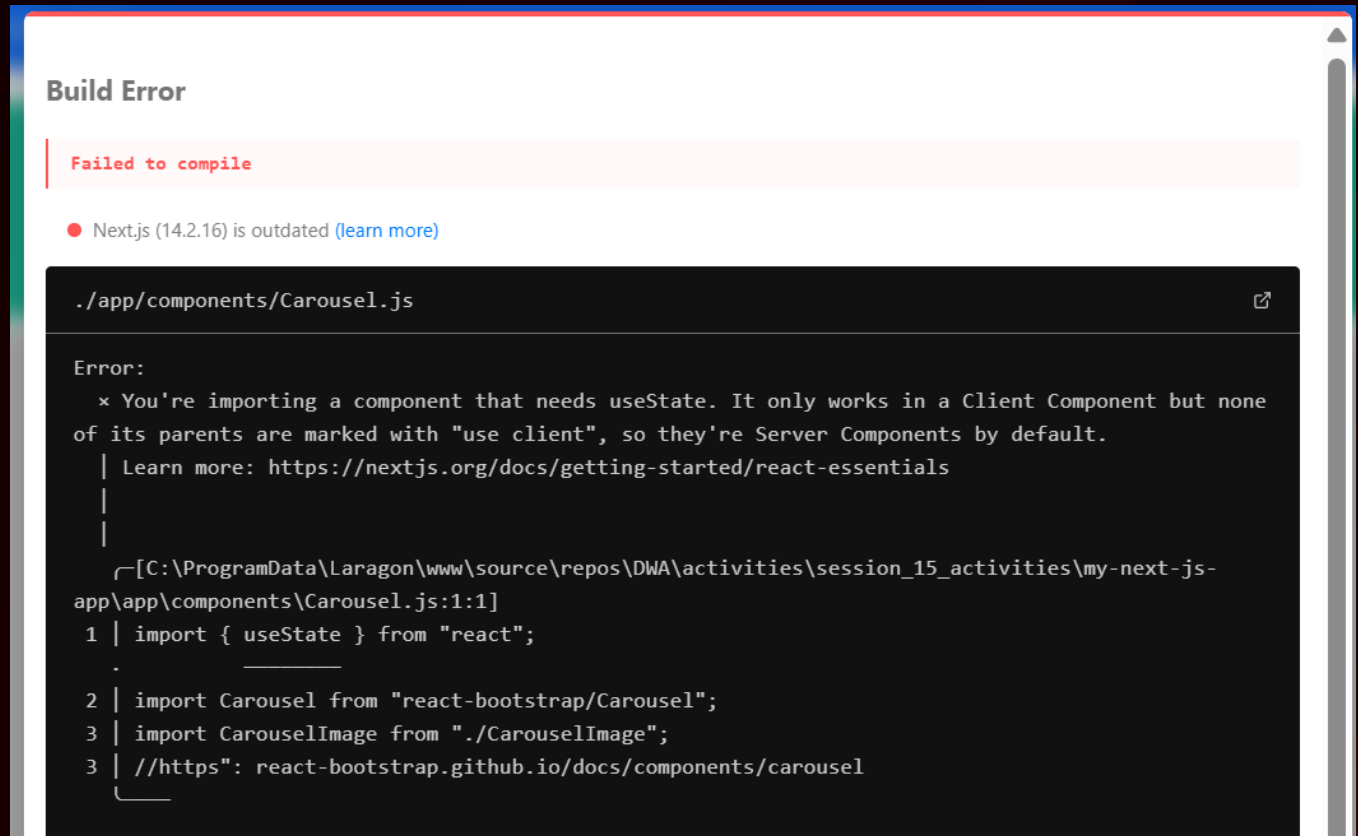
export default function AboutPage() {
  return (
    <main className="container">
      <h1 className={styles.header}>About Us</h1>
      <ControlledCarousel />
    </main>
  );
}
```

Client & Server-Side Components

OOPS – Read the error



We need to specify `use client` as we are using the `useState` hook.



The screenshot shows a 'Build Error' window in a code editor. At the top, a red bar indicates 'Failed to compile'. Below this, a message states 'Next.js (14.2.16) is outdated' with a link to 'learn more'. The main part of the window displays the error details for a file named './app/components/Carousel.js'. The error message is: '× You're importing a component that needs useState. It only works in a Client Component but none of its parents are marked with "use client", so they're Server Components by default. | Learn more: https://nextjs.org/docs/getting-started/react-essentials'. Below the message, the file path is shown: 'C:\ProgramData\Laragon\www\source\repos\DWA\activities\session_15_activities\my-next-js-app\app\components\Carousel.js:1:1'. The code snippet shows imports for 'useState' from 'react', 'Carousel' from 'react-bootstrap/Carousel', and 'CarouselImage' from './CarouselImage'. The error points to the first line of the code.

```
./app/components/Carousel.js

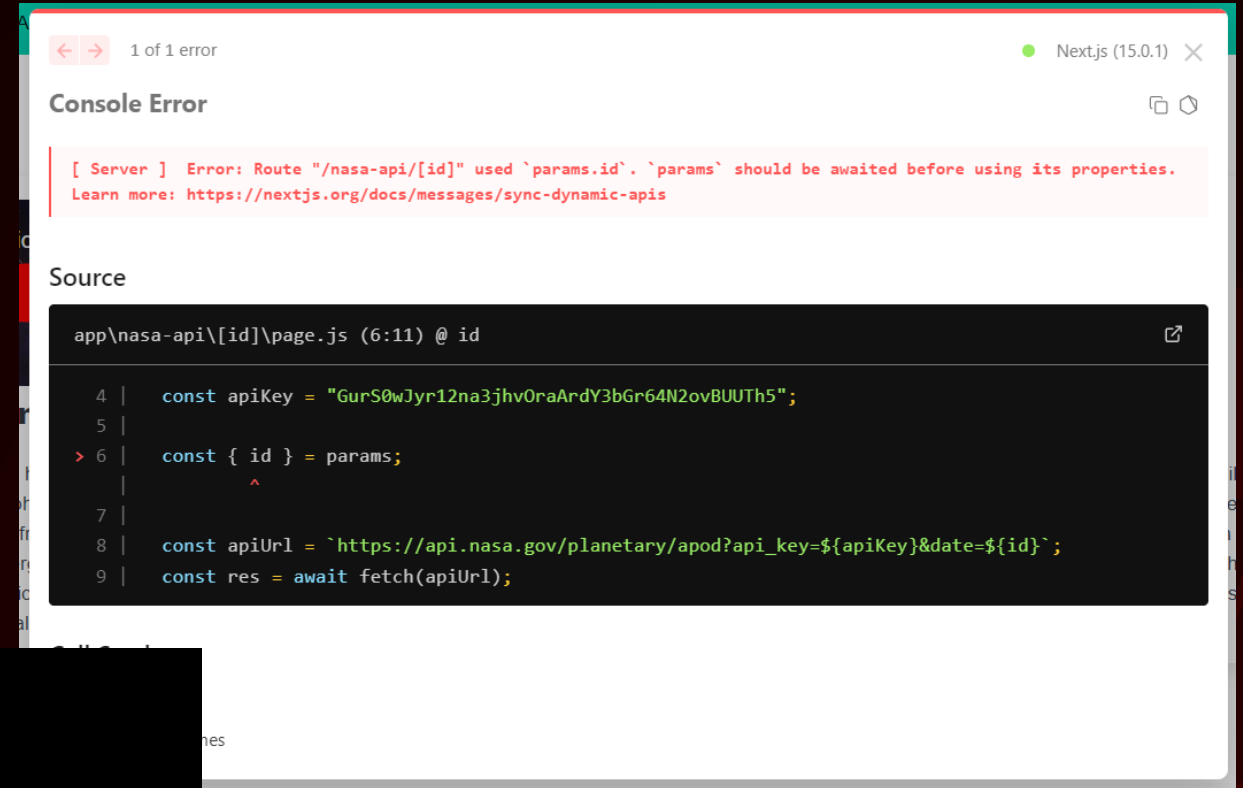
Error:
  × You're importing a component that needs useState. It only works in a Client Component but none
  of its parents are marked with "use client", so they're Server Components by default.
  | Learn more: https://nextjs.org/docs/getting-started/react-essentials
  |
  |
  | [C:\ProgramData\Laragon\www\source\repos\DWA\activities\session_15_activities\my-next-js-
  app\app\components\Carousel.js:1:1]
  1 | import { useState } from "react";
    | _____
  2 | import Carousel from "react-bootstrap/Carousel";
  3 | import CarouselImage from "./CarouselImage";
  3 | //https: react-bootstrap.github.io/docs/components/carousel
    | _____
```

Server-Side Components

Let's also deal with the error in our NASA Component which is rendered from the Server Side

```
const { id } = (await params) || {};
```

```
if (!id) {  
  notFound();  
}
```



NextJS – Images

Using nasa-apod-next-router as our example

Next JS Images

Automatically optimises images at build time and runtime.

1. chooses the best format for the browser(JPEG, PNG, and WebP)
2. generates images in multiple sizes, adapting to screen sizes & resolutions.
3. images are loaded only when they're about to enter the viewport
4. images are cached for faster loading
5. a blurred version is produced for initial loading

Using the Image Component

- We will convert or sculpture gallery component from React to Next JS
- Using the Next JS Image component.

```
✓ data
  JS list.js
✓ gallery
  ✓ [id]
    JS page.js
  JS page.js
> nasa-api
JS global-error.js
```


New props

- We have a number of props built in to help us
- Including the url to the blurred image served by the NextJS server
- Note the `||` which sets a default for a prop

```
<Image  
  src={sculpture.url}  
  alt={sculpture.alt}  
  width={400}  
  height={200}  
  blurDataURL={sculpture.blurDataURL || ""}  
>
```

Gallery Item Dynamic Route

- use the navigation default **notFound()**
- We are also using **React.use()** instead of **useEffect**
- Next JS **suspends** the loading until params is available
- this is just a JS Promise.

```
"use client";

import { notFound } from "next/navigation";
import Image from "next/image";
import { use } from "react";
import { sculptureList } from "../../data/list";

export default function GalleryItem({ params }) {
  const { id } = use(params);

  const sculpture = sculptureList.find(
    (item) => item.id == id
  );

  if (!sculpture) {
    notFound();
  }
}
```