2. React components (functional components)

2.1 Introduction to react components: why do we build Apple website using react components?

- **Component**: A React component is one of the major building blocks of React. Every application you will develop in React will be made up of independent and reusable pieces code, called components
    - **Difference between a component and a plain JavaScript file**: A component has the same purpose as a JavaScript function has, except that a component returns a React element via a render function
    - **Component name must start in uppercase**: The name of a component should always start with a capital letter. This is done to differentiate a component tag from HTML tags in our JSX. For instance, if you name your component as “button” and render this component as <button/> in your App.js, instead of rendering your <button /> component, React ignores it and renders a vanilla <button> element instead. This is because the button component is not capitalized when referenced in ReactDOM.render() method
    - **Keep one component in one file**: There is a convention to keep only one component in one JavaScript file
    - **Keep the file and component/function name the same**: Another convention is to make the function name and JavaScript file names the same
- **Why use React components to build websites like Apple**: Rather than building the whole UI of a website under one single file, it is better to divide all the sections into smaller independent pieces called components. Breaking down the UI into multiple individual components is important for the following reasons:
    - Different developers can work on the components independently and merge them all in a parent component which will be the final UI
    - These components can be reused
    - Allows testing and debugging to be done on each component in isolation

2.2 React components: definition and types

- **Two Types of Components**: There are two kinds of components in React; class components and function components. In this class, we will concentrate on function components and will go through class components in detail on the coming classes. **Note**: The recommended component type in React is functional components
- **What is a functional Component?** The simplest way to define a component is to write a JavaScript function. A functional component is simply a JavaScript function may or may not take a data as a parameter but returns a React element (JSX)
    - **React element**: We have covered this in our previous class. To revise it, a React element is an object that virtually describes the DOM Nodes that a component represents
    - **Example of a functional component**

```
function JustPractice() {
    return <div> JustPractice </div>;
}
```

- **Example of a functional component (using arrow function)**

```
const JustPractice = () => {
    return <div>JustPractice</div>;
 };
```

2.3 Function based components: steps to create functional components

- **Steps to create a functional component**:
    - 1. We have seen in our last class how to create a React app using "create-react-app". Make sure to create a React app first
    - 2. Go to the “src” folder and create a new JavaScript file for your component
    - 3. Import all the necessary files/components and libraries (such as “react” library from “react”) to the component

- To work with jsx in JavaScript file we will have to import the React library. **Syntax**: import React from 'react';
- If we want to use a component our newly created component, we will import that component. **Syntax**: import Footer from "./Footer";
    - **Note**: No need to mention the ".js" file extension when importing your Footer.js component because build workflow automatically considers the import as a ".js" or ".jsx" file type by default. If file is of different type, example a ".css" file, then we will need to mention the extension of the file importing it

o 3. Create a functional component by simply creating a JavaScript function

- Type the "rfce" abbreviation/snippet without the quotation marks. This is basically a shortcut to create a functional component structure/boilerplate. The "rfce" shortcut will import the React library, create your functional component and export your component for other components to use it. **Note**: Make sure you have the "ES7+ React/Redux/React-Native snippets" extension installed on your VSC to access the above shortcut from your component

o 4. Export the component: To be able to use a component later, we need to first export it so we can import it and use it somewhere else

- **Syntax to export:** export default NameOfComponent

o 5. Render your component in your App.js

- **Syntax**: < NameOfComponent />

- **Sample functional component using the above steps**:

// JustPractice.js

```
import React from 'react'

function JustPractice() {

return <div>JustPractice</div>;

}
```

```
export default JustPractice;
```

2.4 Building Apple website using React (functional component)

- Please watch the class demo and build apple.com's homepage using React (functional component)