# **Linking and Routing**

## Basic Types of navigation

Website navigation is the parts of any website that allows you to browse through various pages or links on that website from a single component. There are several practical implementations of this user interface pattern.

The most common navigation components are, a horizontal navigation bar, a vertical navigation menu, a menu hiding behind a button, and a footer navigation menu. The horizontal navigation bar is often referred to as a navbar, and the vertical navigation bar is also known as a sidebar navigation.

*Which of the following statements are correct regarding how navigation is implemented in React? Select all that apply.*

* *In React, the entire app is loaded inside a single div, you're not actually visiting different pages.*
* *Different views are rendered when React makes changes to the Virtual DOM, with React updating the real DOM accordingly*

## The navbar

However, with the default React library, these anchor tags won't work as expected. This is because React can't imitate multi-page websites. However, I can make this possible with the help of another library known as React Router. As you may have guessed from the name React Router gives you more control over the routing of components.

Graphical user interface, text

Description automatically generated

*True or False?*

*You are building an app in React, and have written the following navigation link in your code:*

*<a href="/" className="nav-item">Contact Us!</a>*

*When the user clicks this link, the app will navigate to the ‘Contact Us!’ page.*

* *False*

*Text

Description automatically generated*

After importing the BrowserRouter you need to wrap the app component with in it, as seen above.

Text

Description automatically generated

Next in the app.js file you need to wrap ‘route’ elements within the routes tags, each route has a path and which element (component) will be displayed for the given path.

Text

Description automatically generated

Anchor tags are replaced by ‘Link’ elements as seen above.

## Conditional rendering

In an app, you can render components conditionally based on whether a specific state data has specific values. In other words, when you write the rendering logic in the main app component, you'll need to reference the state of the other components. For example, suppose you have a component that contains code for a button that shows a sidebar, the button controls the state of the toggle sidebar variable, which is set to false. When the button is clicked, the state of the toggle sidebar variable is updated to True, and the sidebar component is displayed.

Graphical user interface, text, application

Description automatically generated

*You are developing a clock app that displays an image of the sun when the time is 6 AM until 6 PM, and an image of the moon for the other hours. Your component code is written as follows:*

*function CurrentImage() {*

*const time = new Date().getHours();*

*return (*

*{hour >= 6 && hour <= 18*

*? <Daytime />*

*: <Nighttime />*

*}*

*Based on this code, what will be returned if the getHours function produces a value of 14?*

* *Daytime*

## Single view conditional updates

*True or false: You can use JavaScript's logical AND operator in a React component's return statement to conditionally render some JSX elements based on whether a value to the right of the AND operator evaluates to true.*

* *False*

# **Using Assets in React**

## What is an asset and where does it live?

In React, assets can be things like images, style sheets, fonts, media files, or basically any file that's needed by your app at runtime. In other words, assets are all the files that your React app needs to have access to in order to work as intended. For example, you might code your app to display specific images or use certain fonts.

Diagram

Description automatically generated with medium confidence

It's important to keep assets easily and readily available to your components. A common way to do this is to add an assets folder inside of a source folder and keep all your apps assets there. Some assets can also be placed inside the public folder. For example, in the default React installation, you'll find that some images such as favicon and Logo 512 are stored there by default. The general rule for asset storage is that if your app can compile without it, you can keep it in the public folder.

*You are developing a web-based clock app that allows the user to set alarms to go off at specific times.*

*Which of the following assets are ideally stored in an Assets folder within the src folder of your project? Choose all that apply.*

* *An alarm sound that plays when the specified time is reached in your app*
* *An image of a clock that is displayed in the app*

*Text

Description automatically generated*

*Graphical user interface, text, application, email

Description automatically generated*

Graphical user interface, text, application, email

Description automatically generated

## Using embedded assets

Text

Description automatically generated

Text

Description automatically generated

An image/asset can be used directly using the require function as seen above.



Graphical user interface, text

Description automatically generated

You can also use a URL/link as seen above.

*In React, you can import an image as you would any other module.*

* *True*

## Audio and video

Text

Description automatically generated

You can use the readily available HTML5 video tag to load a local video asset to your React app. Like loading an image or any other kind of asset file, you can just declare a variable somewhere above a component's return statement and then add it as a JSX expression to the SRC attribute. However, this approach might not work as easily with some major providers of video content, such as one of the several social media and social video sharing platforms.

Text

Description automatically generated with medium confidence

Graphical user interface, text, application, email

Description automatically generated

*When looking for a third-party NPM package to use in the NPM package ecosystem, how would you determine if the package is suitable and the right one you need? Select all that apply.*

* *Check the package’s GitHub page*
* *Perform an internet search for the package name*
* *Check the frequency of updates*

## Create an audio / video component

A screenshot of a computer

Description automatically generated with medium confidence

*True or false?*

*When you render the React player in your app, you can add code that determines whether or not a video starts automatically when the page loads.*

* *True*