What are Components?

What are Components?

Components are the building blocks of the (UI). They allow you to break down the UI into reusable, isolated pieces of code that can be managed, maintained, and composed together efficiently

- There are two main types of Components

Function Components (Preferred)

Components written as JavaScript functions that return JSX.

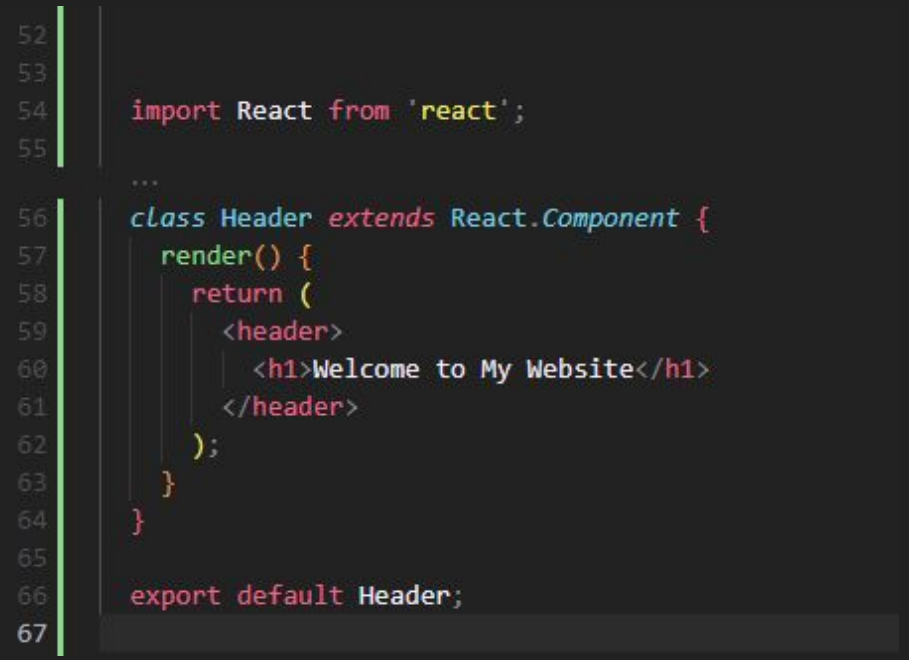
These are the most common types of components in modern React. They are simple, concise, and can use React hooks (e.g., useState, useEffect) to manage state and lifecycle.

```
53
54  function Header() {
55    return (
56      <header>
57        <h1>Welcome to My Website</h1>
58      </header>
59    );
60  }
61
62  export default Header;
63
```

Class Components (Less Common)

Components written as JavaScript classes that extend React.Component and must implement a render() method.

Class components were used more often in older React projects but have been largely replaced by function components, especially with the introduction of hooks.



```
52
53
54  import React from 'react';
55
    ...
56  class Header extends React.Component {
57    render() {
58      return (
59        <header>
60          <h1>Welcome to My Website</h1>
61        </header>
62      );
63    }
64  }
65
66  export default Header;
67
```

Key Concepts in Components

- Props
- State
- Children
- Component Composition
- Conditional Rendering

Props - Key Concepts

Props (Short for "Properties")

Props are arguments passed to components, allowing them to be dynamic and reusable.

They are used to pass data from a parent component to a child component.

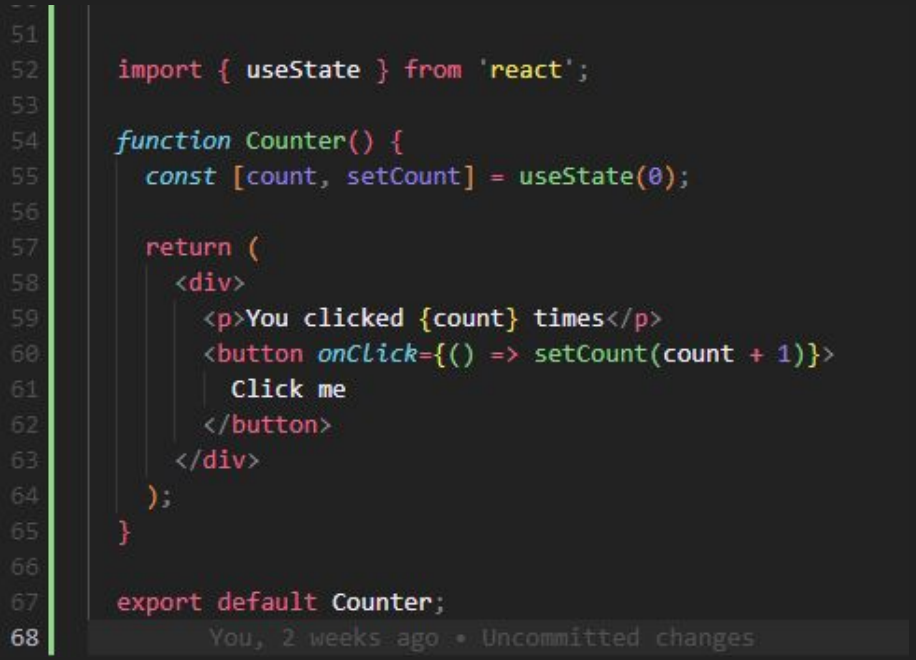
Here, the Welcome component receives a name prop from the App component and uses it inside its JSX.

State - Key Concepts

State (Used only in Function Components with Hooks)

State allows components to manage dynamic data internally.

You can update the state inside the component, and it will re-render the UI when the state changes.



```
51
52  import { useState } from 'react';
53
54  function Counter() {
55    const [count, setCount] = useState(0);
56
57    return (
58      <div>
59        <p>You clicked {count} times</p>
60        <button onClick={() => setCount(count + 1)}>
61          Click me
62        </button>
63      </div>
64    );
65  }
66
67  export default Counter;
68
        You, 2 weeks ago • Uncommitted changes
```

Children

The children prop allows components to nest other components or JSX elements inside them.

```jsx
function Layout({ children }) {
  return (
    <div>
      <header>Header Section</header>
      {children}
      <footer>Footer Section</footer>
    </div>
  );
}

export default Layout;

// Usage in a page:
function HomePage() {
  return (
    <Layout>
      <main>Main Content</main>
    </Layout>
  );
}
```

Component Composition

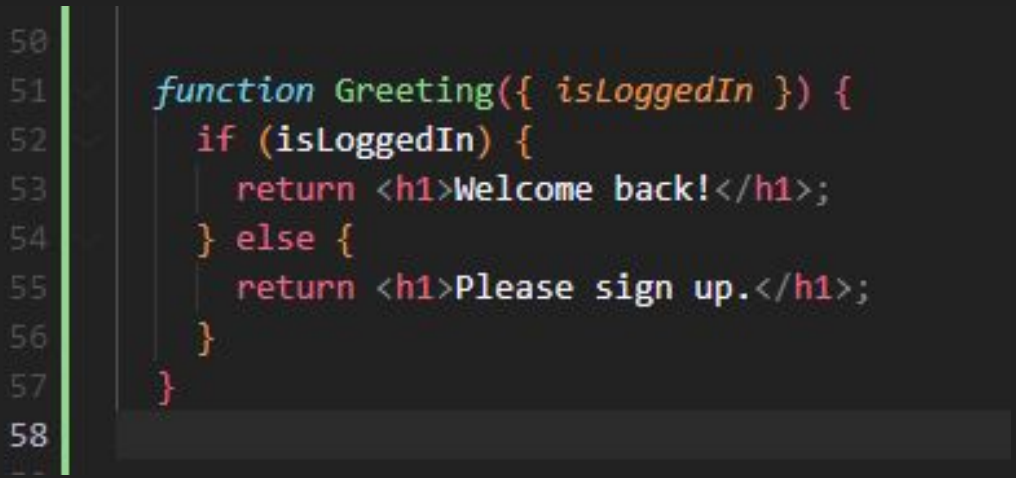
Components can be combined to build complex UIs.

This process is called composition, where smaller components are composed into larger components.

```
50
51  function Card({ title, content }) {
52    return (
53      <div className="card">
54        <h2>{title}</h2>
55        <p>{content}</p>
56      </div>
57    );
58  }
59
60  function App() {
61    return (
62      <div>
63        <Card title="Card 1" content="This is the first card" />
64        <Card title="Card 2" content="This is the second card" />
65      </div>
66    );
67  }
68
69
```

Conditional Rendering

Components can display different outputs based on conditions.

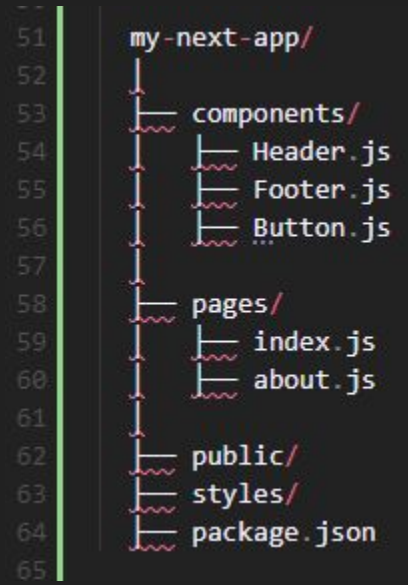


Organizing Components in a Next.js Project

It's common practice to store components in a dedicated folder, typically called components/. This structure helps maintain the separation between page logic (in pages/) and UI components (in components/).

`Header.js` and `Footer.js` could be used across multiple pages,

ensuring that the common parts of the UI are reusable.



QUESTIONS?

Components are reusable pieces of code that return UI elements.

Function components are the most common, supporting hooks for managing state and effects.

Props allow data to be passed between components, while state handles local dynamic data.