REACT ROUTER

https://reactrouter.com/

Routing

The act of moving between pages/components of an app to complete tasks

Designing effective navigation = Simplify the user journey

Routing

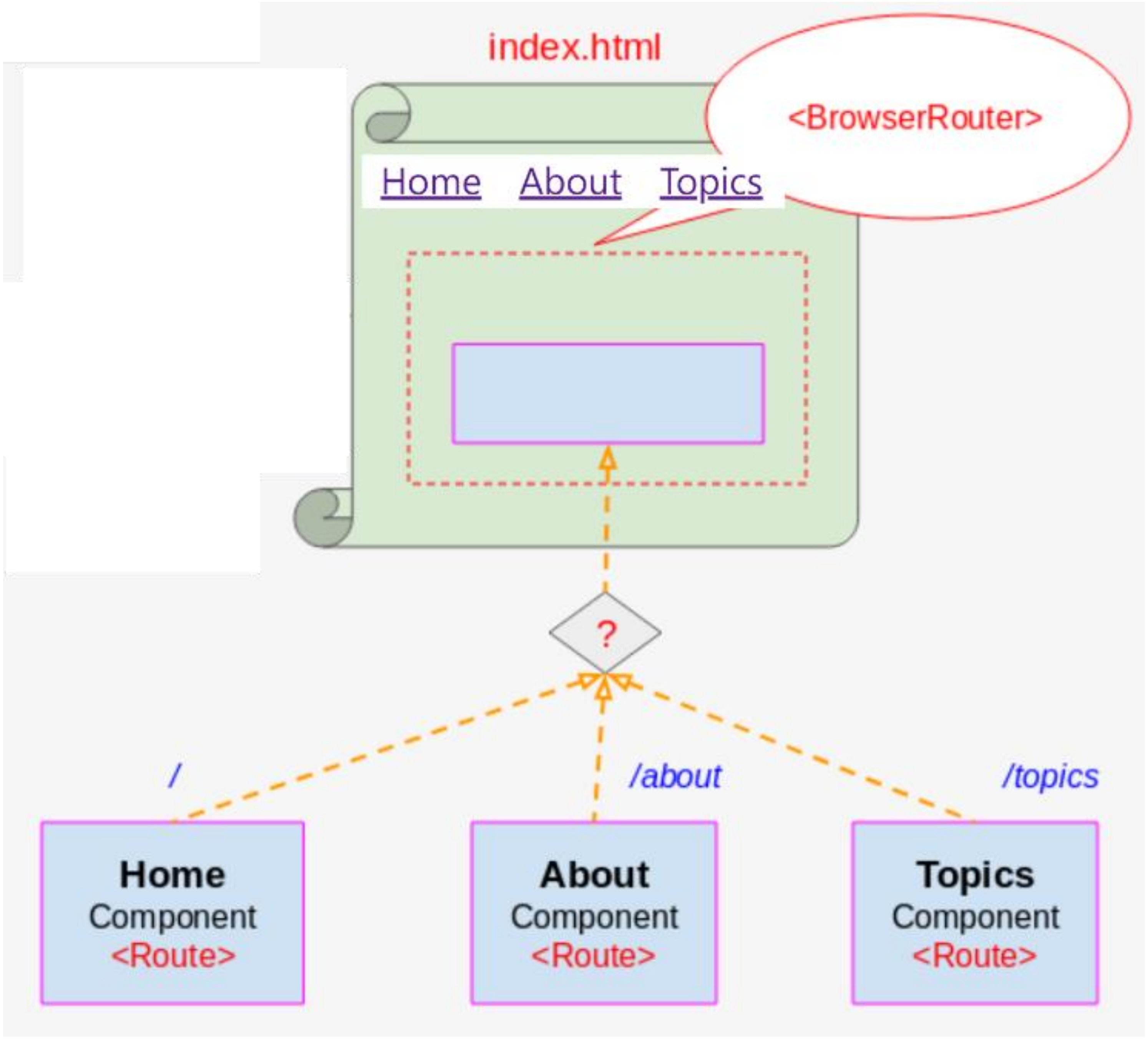
- Component Router ease loading components as the user interacts with the page
- Routing implements client-side navigation for SPA:

  - Configure routes: map relative Url to the corresponding components in a declarative way

  - On URL change the router loads the associated component

- Install

  ```
  npm install react-router-dom@6
  ```


index.html
<BrowserRouter>
Home
About
Topics
?
/
/about
/topics
Home Component <Route>
About Component <Route>
Topics Component <Route>

Routing with React Router

```
import React from "react";
import { BrowserRouter as Router, Route, Link } from "react-router-dom";
function RouterBasicExample() {
  return (
    <Router>
        <div>
            <ul>
                <li> <Link to="/">Home</Link> </li>
                <li> <Link to="/about">About</Link> </li>
                <li> <Link to="/topics">Topics</Link> </li>
            </ul>
            <hr />
           <Routes>
              <Route exact path="/" element={Home} />
              <Route path="/about" element={About} />
              <Route path="/topics" element={Topics} />
           </Routes>
        </div>
    </Router>
  ); }
```

Configuring Routes

- React-Router allows you to declaratively define routes using the `<Route>` component
- **<Route>** component renders the component mentioned in the `element` prop when the path value mentioned in the `path` prop matches the browser's URL location

Route Parameter

- A <Route> can be configured to accept URL parameters
  - e.g, to display product info for a given product, the URL path could look like '**/products/1**' for a product with id of 1, and **/ products/123** for a product with id of 123
- A <Route> component can be configured to accept the dynamic portion in the URL prefixed with a colon (:)

```
<Route path="/products/:productId" element={SingleProduct}/>
```

- Use **useParams** hook to get the passed parameter from the SingleProduct component

```
const { productId } = useParams();
```

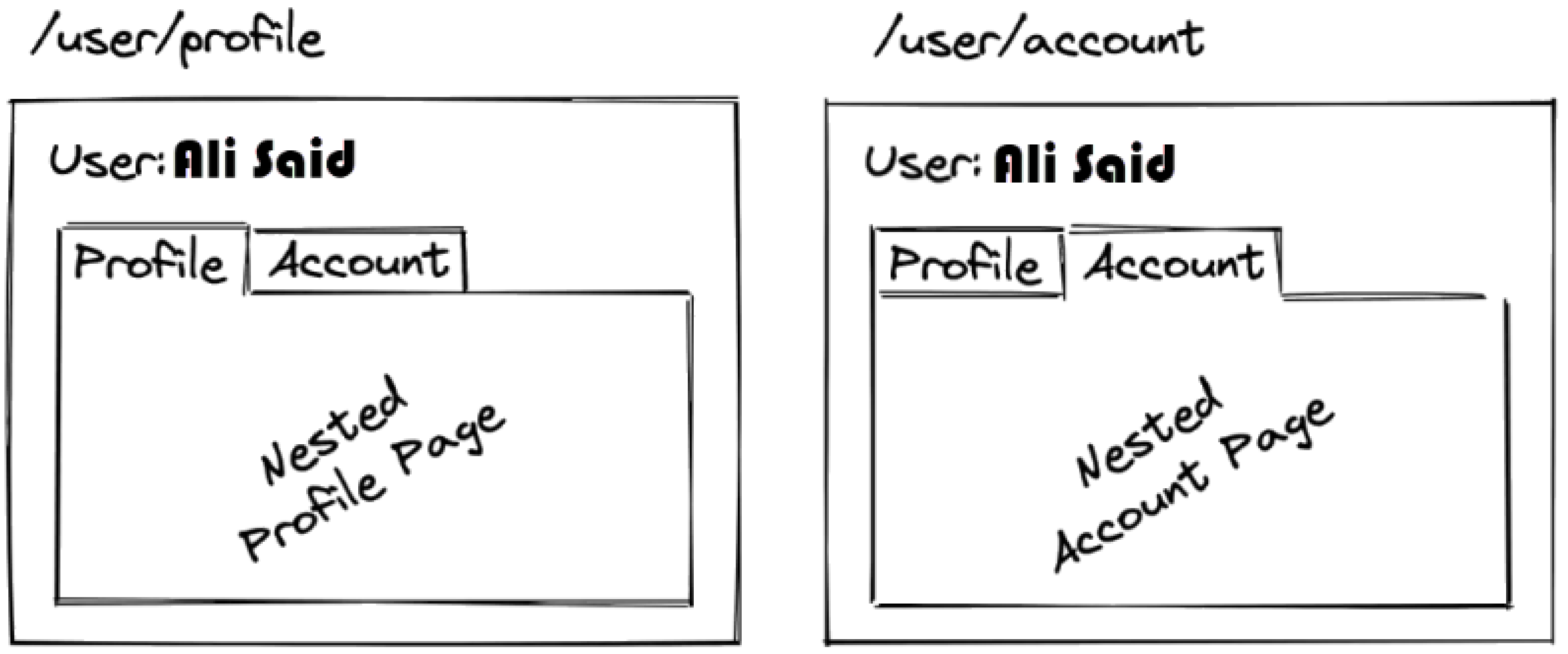
Router programmatic access

- Request the router to navigate to a Url programmatically using **useNavigate**

```
const navigate = useNavigate();
navigate('/dashboard');
```

Nested Routes

- Nested routes allowing replacing a specific fragment of the view based on the current route
  - E.g., a user page can present multiple tabs (e.g. Profile, Account) to navigate through a user's information. By clicking these tabs, the URL in the browser will change, but instead of replacing the whole page, only the content of the tab gets replaced



Nested Routes

```
<Route path="user" element={<User />}>
  <Route path="profile" element={<Profile />} />
  <Route path="account" element={<Account />} />
</Route>
```

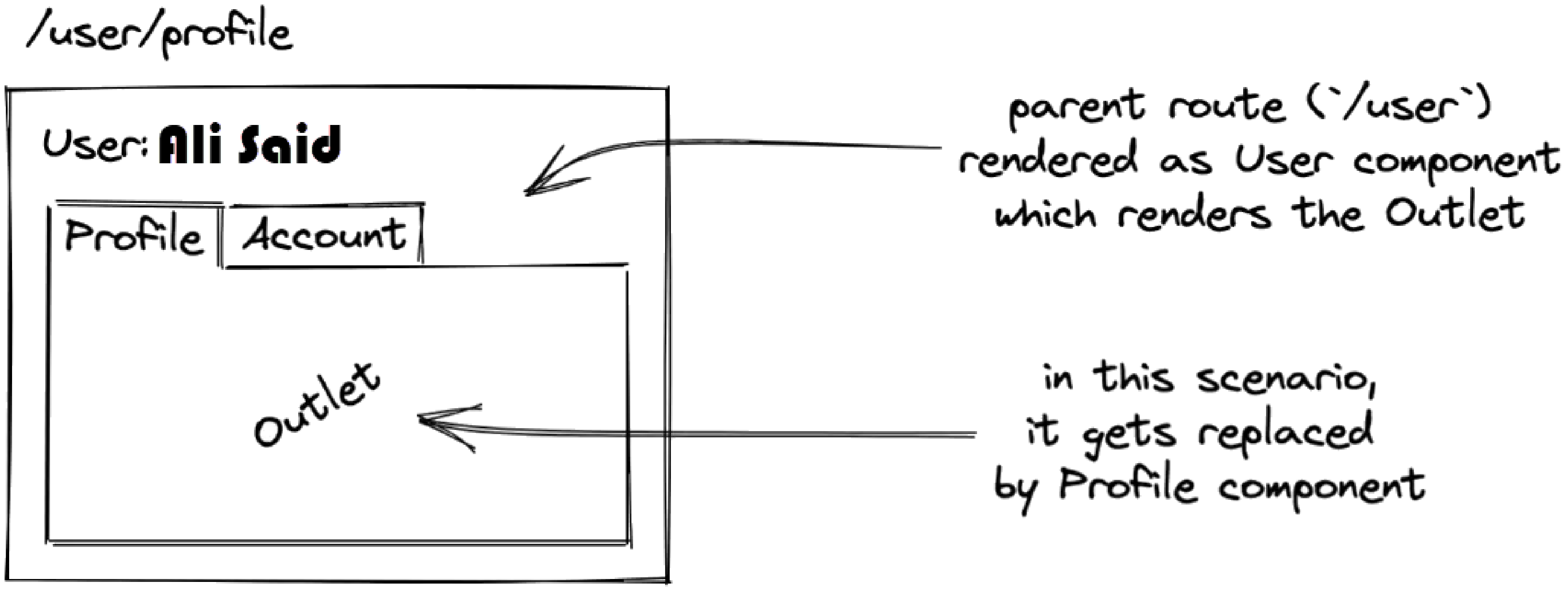
```
import { Routes, Route, Link, Outlet } from 'react-router-dom';

...

const User = () => {
  return (
    <>
      <h1>User</h1>

      <nav>
        <Link to="profile">Profile</Link>
        <Link to="account">Account</Link>
      </nav>

      <Outlet />
    </>
  );
};
```

The Outlet component renders the matching child route

Outlet component

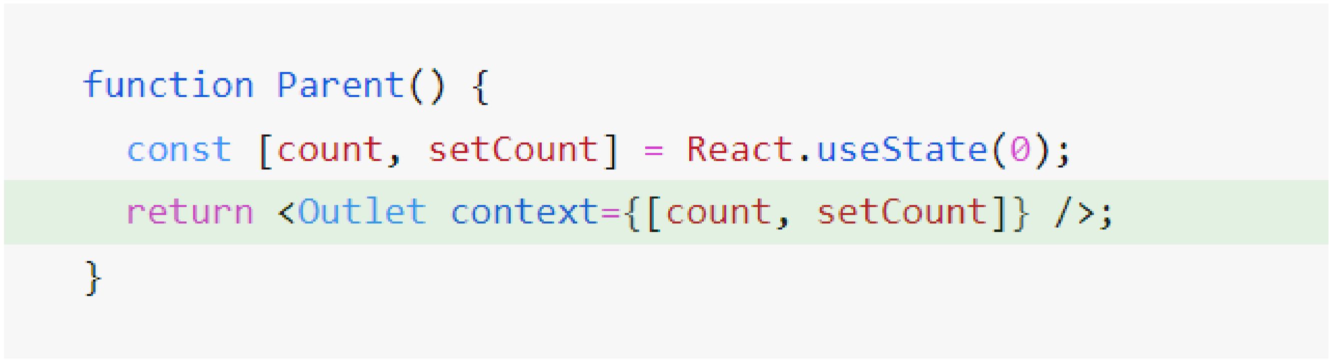
- The Outlet component renders the matching child route

  E.g. `/user/profile`

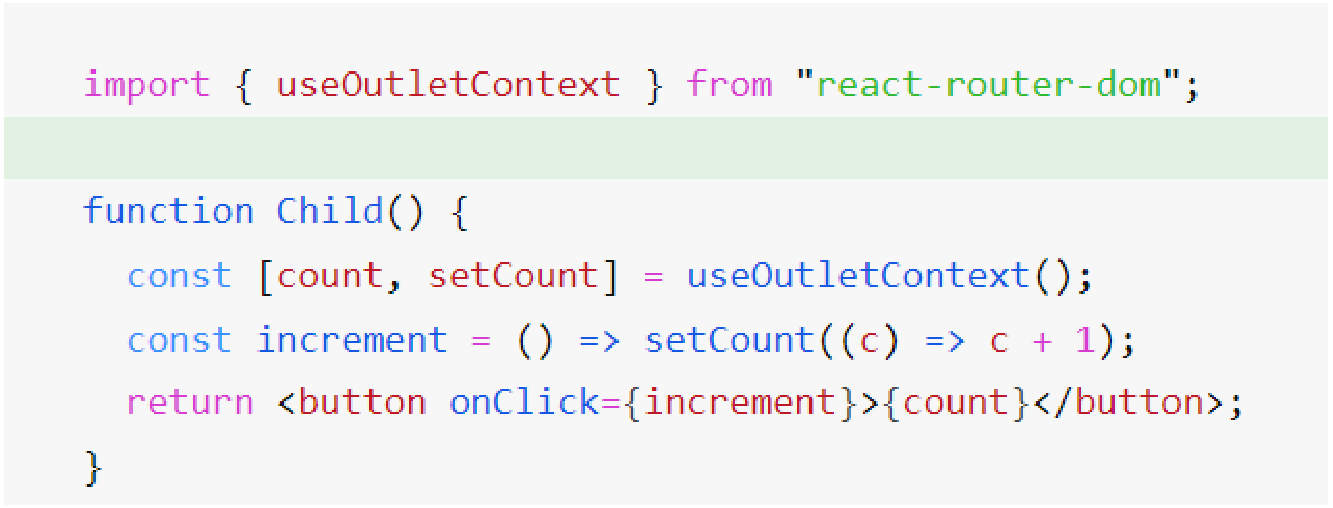

/user/profile
User: Ali Said
Profile
Account
Outlet
parent route (`/user`) rendered as User component which renders the Outlet
in this scenario, it gets replaced by Profile component

Outlet Context

- Often parent routes manage state or other values you want shared with child routes. You can use the built-into context of the outlet component <Outlet />



```
function Parent() {
  const [count, setCount] = React.useState(0);
  return <Outlet context={[count, setCount]} />;
}
```



```
import { useOutletContext } from "react-router-dom";

function Child() {
  const [count, setCount] = useOutletContext();
  const increment = () => setCount((c) => c + 1);
  return <button onClick={increment}>{count}</button>;
}
```