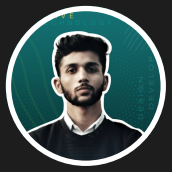
How to implement
Lazy Loading in REACT



M. SHAHZEB RAZA
REACT WEB DEVELOPER

Here's a regular **"App.js file"** with Routes implemented for various componnent

```
import { Switch, Route } from './react-router-dom';
import ComponentA from './components/ComponentA';
import ComponentB from './components/ComponentB';
import ComponentC from './components/ComponentC';

function App() {
    return (
        <Switch>
            <Route path='/a' exact>
                <ComponentA />
            </Route>
            <Route path='/b' exact>
                <ComponentB />
            </Route>
            <Route path='/c' exact>
                <ComponentC />
            </Route>
        </Switch>
    )
}
```

But there is a Problem.

React imports all the components as soon as the Application starts.

React takes some **time to load all components** and the App is accessible only after all components have been imported.

The visitor may not visit all these routes and therefore **importing all these components is bad for performance.**

So, what can we do?

React offers a solution!
We can choose to lazy load the components only when they're required.

With Lazy Loading, components are imported by React App only when the relevant route is visited.

There are 5 simple steps to implement lazy loading.

Step 01: Import lazy & Suspense

```
1 // Step 01: import 'lazy' & 'Suspense' from 'React',
2 import { lazy, Suspense } from 'react';
3
```

Step 02: Use import function

```
1 // Step 02: Use "import function" instead of "import statements" to
  import components i.e.
2 const yourComponent = () => import('./components/yourComponent');
3
```

Step 03: Wrap imports in lazy()

```
1 // Step 03: Wrap your imports in lazy()
2 const yourComponent = lazy(() =>
  import('./components/yourComponent'));
3
```

Step 04: Wrap returns in Suspense()

```
// Step 04: Wrap the returns from your component in Suspense()
function App() {
    return (
        <Suspense >
            {/* all return components */}
        </Suspense>
    )
}

```

Step 05: Specify fallback for Suspense()

```
// Step 05: Specify the fallback JSX for your lazy components
<Suspense fallback={<div>Loading ... </div>} >
    {/* all return components */}
</Suspense>

```

That’s it! We’ve successfully implemented Lazy Loading for React.

Your component should look like this.

```jsx
import { lazy, Suspense } from 'react';
import { Switch, Route } from './react-router-dom';
const ComponentA = lazy(() => import('./components/ComponentA'));
const ComponentB = lazy(() => import('./components/ComponentB'));
const ComponentC = lazy(() => import('./components/ComponentC'));


function App() {
    return (
        <Suspense fallback={<div>Loading...</div>} >
            <Switch>
                <Route path='/a' exact>
                    <ComponentA />
                </Route>
                <Route path='/b' exact>
                    <ComponentB />
                </Route>
                <Route path='/c' exact>
                    <ComponentC />
                </Route>
            </Switch>
        </Suspense>
    )
}
```