Setup Guide: Material UI (MUI) with Next.js App Router

Follow this guide to set up **Material UI (MUI)** with the **Next.js App Router**.

Note: Follow 1, 2 & 5(optional) for basic set up of **Material UI (MUI)** with the **Next.js App Router**.

1. Create a Next.js Project (with App Router)

If you don't have a project yet, create a new Next.js project using the App Router structure:

```
npx create-next-app@latest my-mui-app
```

```
cd my-mui-app
```

Make sure to select **App Router** during setup, as Next.js will ask you whether you want to use the App Router or Pages Router.

2. Install Material UI (MUI) and Dependencies

Install the required Material UI packages by running the following command:

```
npm install @mui/material @emotion/react @emotion/styled
```

- `@mui/material`: The core Material UI library for UI components.
- `@emotion/react` and `@emotion/styled`: Dependencies for styling with Material UI.

3. Create the Custom Theme

Create a new file `lib/theme.js` to define your custom theme:

```
// lib/theme.js

import { createTheme } from "@mui/material/styles";

const theme = createTheme({
  palette: {
    primary: {
      main: "#0070f3", // Primary color
    },
    secondary: {
      main: "#19857b", // Secondary color
    },
  },
  typography: {
    fontFamily: '"Roboto", "Helvetica", "Arial", sans-serif',
    h1: {
      fontSize: "2.5rem",
    },
    h2: {
      fontSize: "2rem",
    },
    h3: {
      fontSize: "1.75rem",
    },
    // Customize more typography styles if needed
  },
  components: {
    MuiButton: {
      styleOverrides: {
        root: {
          textTransform: "none", // Disable button text uppercase
        },
      },
    },
  },
});

export default theme;
```

4. Set Up Global Styling for Material UI

In the Next.js App Router structure, apply global styles and theme providers in `app/layout.js`.

```
"use client"; // Required for client-side rendering

import { CssBaseline, ThemeProvider } from "@mui/material";
import theme from "@/lib/theme"; // Import the custom theme
import "./globals.css"; // Import global styles if needed

export default function RootLayout({ children }) {
  return (
    <html lang="en">
      <body>
        <ThemeProvider theme={theme}>
          {/* CssBaseline to apply Material UI's global reset */}
          <CssBaseline />
          {children}
        </ThemeProvider>
      </body>
    </html>
  );
}
```

5. Using Material UI Components

You can use Material UI components in your Next.js App Router project. For example, add a component in `app/page.js`:

```
"use client"; // Required for client-side rendering

import { Button, Typography } from "@mui/material";

export default function Home() {
  return (
    <div style={{ textAlign: "center", marginTop: "50px" }}>
      <Typography variant="h3" gutterBottom>
        Welcome to My MUI + Next.js App!
      </Typography>
      <Button variant="contained" color="primary">
        Click Me
      </Button>
    </div>
  );
}
```

6. Server-Side Rendering (Optional)

Next.js automatically handles **server-side rendering (SSR)** for SEO and performance, so no extra configuration is needed. Using MUI's `CssBaseline` ensures proper rendering without flickering.

7. Optimize CSS for Production

Next.js purges unused CSS automatically. Make sure to keep the bundle size small by removing any unused styles.

Final Thoughts

- **SEO & Performance**: MUI works seamlessly with Next.js' App Router, maintaining high performance and SEO-friendliness.
- **Tailwind CSS with MUI**: You can use both MUI and Tailwind CSS together, but ensure that conflicting styles are managed properly and unused classes are purged.