Next.js Fundamentals

Layouts

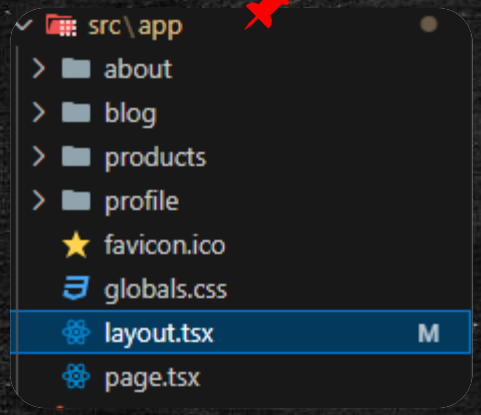
npx create-next-app@latest layout-demo

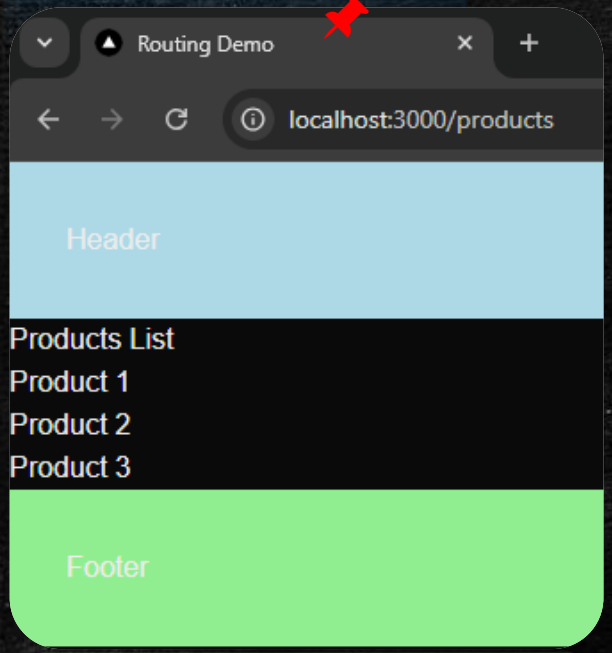
Layouts

- layout.tsx is a component that defines the page layout in Next.js.
- It contains shared structures like headers, footers, or sidebars.
- If you delete layout.tsx, Next.js will automatically create a new one.
- Every Next.js app requires a default layout for proper structure.
- This ensures a consistent look across all pages.
- It avoids repeating the same components on each page.
- The user experience stays clean and seamless.

https://github.com/oguzhandilek

Scenario 1 (Header-Body-Footer)



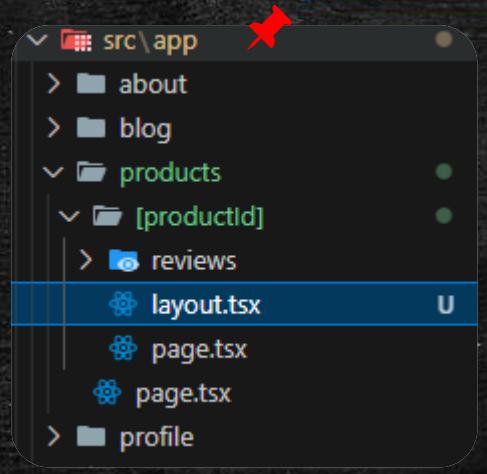


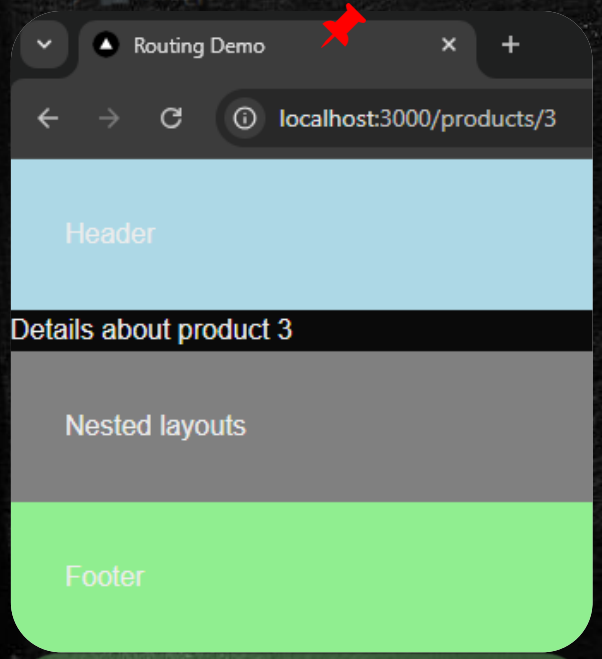

```tsx
import type { Metadata } from "next";
import "./globals.css";
export const metadata: Metadata = {
  title: "Routing Demo",
  description: "Routing Demo",
};

export default function RootLayout({
  children,
}: Readonly<{
  children: React.ReactNode;
}>) {
  return (
    <html lang="en">
      <body>
        <header style={{ backgroundColor: "lightblue", padding: "2rem" }}>
          <h1>Header</h1>
        </header>
        {children}
        <footer style={{ backgroundColor: "lightgreen", padding: "2rem" }}>
          <h1>Footer</h1>
        </footer>
      </body>
    </html>
  );
}
```

npm run dev

https://github.com/oguzhandilek

Scenario 2 (Nested Layout)






```tsx
import React from "react";

export default function ProductDetailLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  return (
    <>
      {children}
      <h2 style={{ backgroundColor: "grey", padding: "2rem" }}>
        Nested layouts
      </h2>
    </>
  );
}
```

npm run dev

Bonus Information!

Oğuzhan Dilek

https://github.com/oguzhandilek

- Thank you for following me and making it this far.
- I hope I've been able to add value to you!
- Now, I'm going to share an advanced-level piece of information with you!

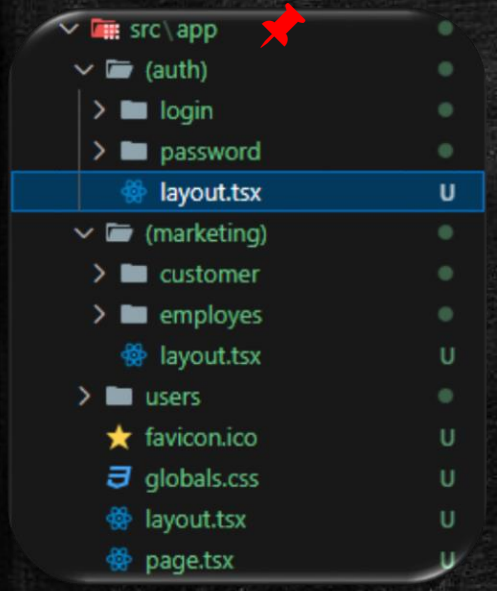
Let's keep moving forward.

--▶

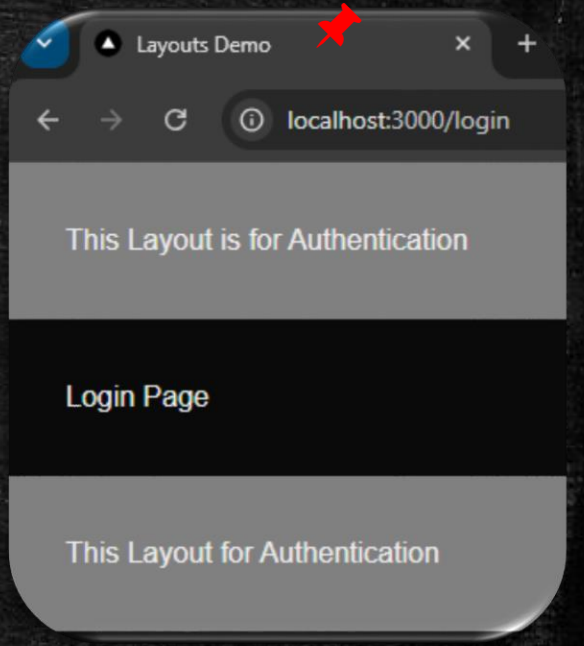
Multiple Root Layouts and Route Groups

- Organize our project structure without affecting the URLs
- Apply layouts selectively to specific parts of our app
- This approach (Route Groups) enhances modularity, making code maintenance more efficient, especially in larger projects.
- To create a route group, it's enough to wrap the main folder in parentheses — for example, (auth) or (marketing).

Scenario Bonus (Multiple Layout Routing and Route Groups)

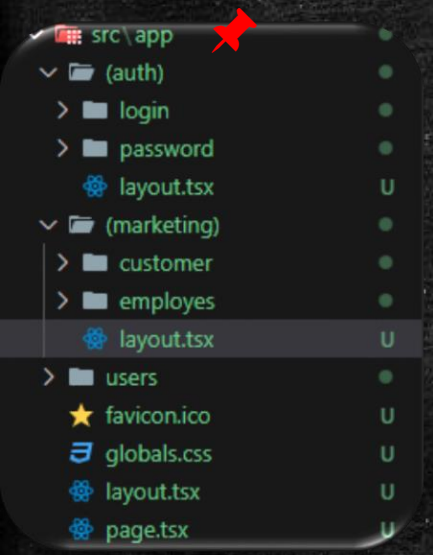



```tsx
import React from "react";
export default function RootLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  return (
    <>
      <html lang="en">
        <body>
          <header style={{ backgroundColor: "grey", padding: "2rem" }}>
            {" "}
            This Layout is for Authentication
          </header>
          {children}
          <footer style={{ backgroundColor: "grey", padding: "2rem" }}>
            This Layout for Authentication
          </footer>
        </body>
      </html>
    </>
  );
}
```

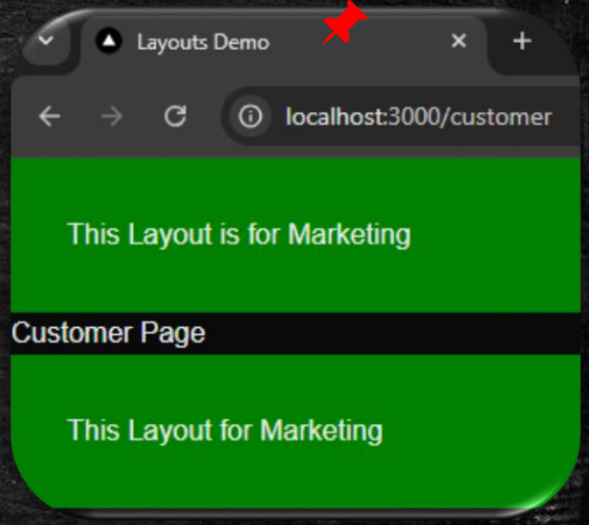


npm run dev

Scenario Bonus (Multiple Layout Routing and Route Groups)




```tsx
export default function RootLayout({children}:{
    children:React.ReactNode;
}) {
    return <>
     <html lang="en">
        <body>
          <header style={{ backgroundColor: "green", padding: "2rem" }}>
            This Layout is for Marketing
          </header>
          {children}
          <footer style={{ backgroundColor: "green", padding: "2rem" }}>
            This Layout for Marketing
          </footer>
        </body>
      </html>
    </>
}
```



You can visit my GitHub profile to see the complete code.

https://github.com/oguzhandilek/Next.js-Fundamentals

npm run dev

Next.js Fundamentals

Oğuzhan Dilek
https://github.com/oguzhandilek

```
export default function AlbeyazimSoft() {
  return (
    <>
      <h1>The series will continue...</h1>
      <h1>Don't forget to like and share!</h1>
    </>
  );
}
```