49. Custom Hook - useIsAuth()

#reactjs #hook #graphql #query #frontend #authentication #routing #url

Implement useIsAuth()

- We implement a custom hook to check if the user is logged in or not on the frontend
- This simple hook routes the user to login page is he's not logged in
- Note that we wait for fetching to end before we check if there's a user that's logged in or not
- We also use the next query in the URL to tell the login page where it should send the user after succesfully logging in

/hooks/useIsAuth.ts

```
import { useRouter } from "next/router";
import { useMeQuery } from "../generated/graphql";
import { useEffect } from "react";

export const useIsAuth = () => {
  const [{ data, fetching }] = useMeQuery();
  const router = useRouter();
  useEffect(() => {
    // if user is not logged in, re-route to login page
    if (!fetching && !data?.me) {
      router.replace("/login?next=" + router.pathname);
    }
  }, [fetching, data, router]);
};
```

Update create-post page to use useIsAuth() hook

- Simply insert the hook at the top of the page so it gets executed first

/pages/create=page.tsx

```
const CreatePost: React.FC<{}> = ({}) => {
  const router = useRouter();
  useIsAuth(); // reroute user to login if not logged in
  const [, createPost] = useCreatePostMutation();
```

Update login page to direct to “next” query in the URL

- If the URL includes a next query, we direct the user to that page

/pages/login.tsx

```
onSubmit={async (values, { setErrors }) => {
  const response = await login(values);
  if (response.data?.login.errors) {
    setErrors(toErrorMap(response.data.login.errors));
  } else if (response.data?.login.user) {
    if (typeof router.query.next === "string") {
      router.push(router.query.next);
    } else {
      router.push("/");
    }
  }
}}
```