81. Page - edit

#reactjs #nextjs #chakraui #frontend #page #error #server #backend #urql

Implement edit Page

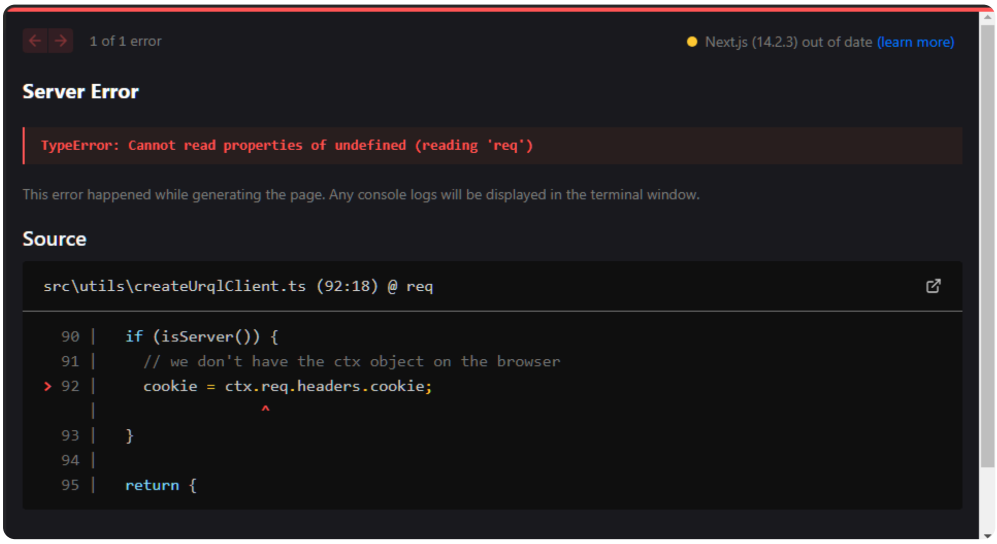
- We need a variable (postId) in the URL of this page, so that the post with that id is shown to edit
  i.e. **localhost:3000/edit/123**
- In NextJS the convention for such pages is to create a folder with the page name post and place the file inside this folder
- The file name should be `[variableName].tsx` - in our case it will be [id].tsx
- **Note that** this page is almost 100% the same as the post page. We could create a component that *could* be used in both pages to reduce code repetition (but we won't do it here)

/pages/post/edit/[id].tsx

```
import { withUrqlClient } from "next-urql";
import { createUrqlClient } from "../../../utils/createUrqlClient";
import { Box, Button } from "@chakra-ui/react";
import { Formik, Form } from "formik";
import { InputField } from "../../../components/InputField";
import { Layout } from "../../../components/Layout";
import { TextAreaField } from "../../../components/TextAreaField";
import { useGetPostFromUrl } from "../../../utils/useGetPostFromUrl";
import { useUpdatePostMutation } from "../../../generated/graphql";
import { useRouter } from "next/router";

const EditPost = ({}) => {
  const router = useRouter();
  const [{ data, fetching, error, postId }] = useGetPostFromUrl();
  const [, updatePost] = useUpdatePostMutation();

  if (fetching) {
    return (
      <Layout>
        <Box>Loading...</Box>
      </Layout>
    );
  }
  if (error) {
    return <div>error</div>;
  }

  if (!data?.post) {
    return (
      <Layout>
        <Box>Could not find post...</Box>
      </Layout>
    );
  }

  return (
    <Layout variant="small">
      <Formik
        initialValues={{ title: data.post.title, text: data.post.text }}
        onSubmit={async (values) => {
          await updatePost({ id: postId, ...values });
          router.back(); // takes back to the page that routed here
        }}
      >
        {({ isSubmitting }) => (
          <Form>
            <InputField
              name="title"
              label="Title"
              placeholder="Title of the post"
            />
            <Box mt={4}>
              <TextAreaField
                name="text"
                label="Body"
                placeholder="Enter your text"
              />
            </Box>
            <Button type="submit" isLoading={isSubmitting} mt={4} color="teal">
              Update Post
            </Button>
          </Form>
        )}
      </Formik>
    </Layout>
  );
};

export default withUrqlClient(createUrqlClient, { ssr: true })(EditPost);
```

Fix Server Error

- A server error occurs if we refresh the edit page:



Server Error

```
TypeError: Cannot read properties of undefined (reading 'req')
```

This error happened while generating the page. Any console logs will be displayed in the terminal window.

Source

```
src\utils\createUrqlClient.ts (92:18) @ req

  90 |   if (isServer()) {
  91 |     // we don't have the ctx object on the browser
> 92 |     cookie = ctx.req.headers.cookie;
     |                  ^
  93 |   }
  94 |
  95 |   return {
```

- We can fix this simplt by using an optional chain as follows:

/utils/createUrqlClient.ts

```
if (isServer()) {
  // we don't have the ctx object on the browser
  cookie = ctx?.req?.headers?.cookie;
  }
```