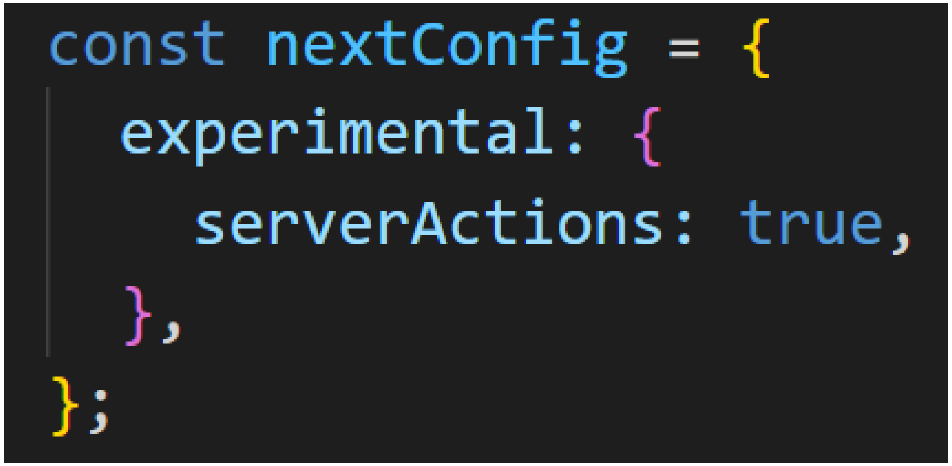
NEXT.JS

Server Actions
&
Rendering Strategies

Server Actions

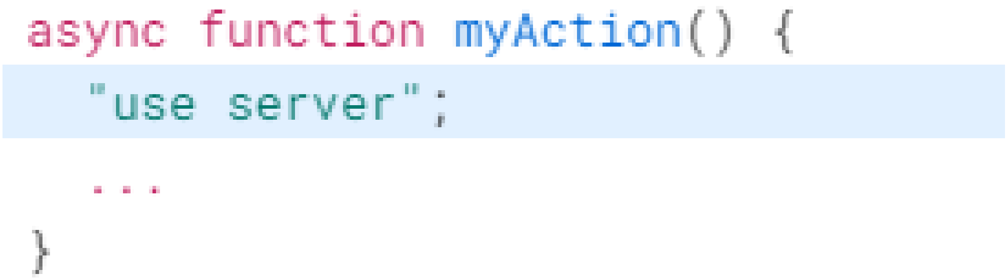
- Server Actions allow us to create functions that run on the server and can be called directly without needing to create an in-between API layer
  - They can often be a simpler alternative to API routes for data mutations
  - reduced client-side JavaScript
- Server Actions are not fully-stable yet, so you must opt-in via the **`next.config.js`** file



```
const nextConfig = {
  experimental: {
    serverActions: true,
  },
};
```

Server Actions

- Create a Server Action by defining an asynchronous function with the `"use server"` directive at the top of the function body



```
async function myAction() {
  "use server";
  ...
}
```

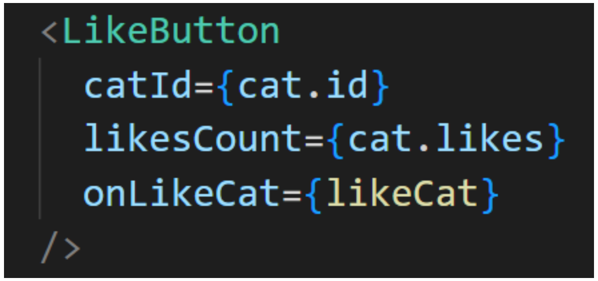
- You can invoke Server Actions using the following methods:
  - Using form action to allows invoking a server action on a <form> element
  - Passing the form action to a client-side component to directly invoke it

Handle Form Submission

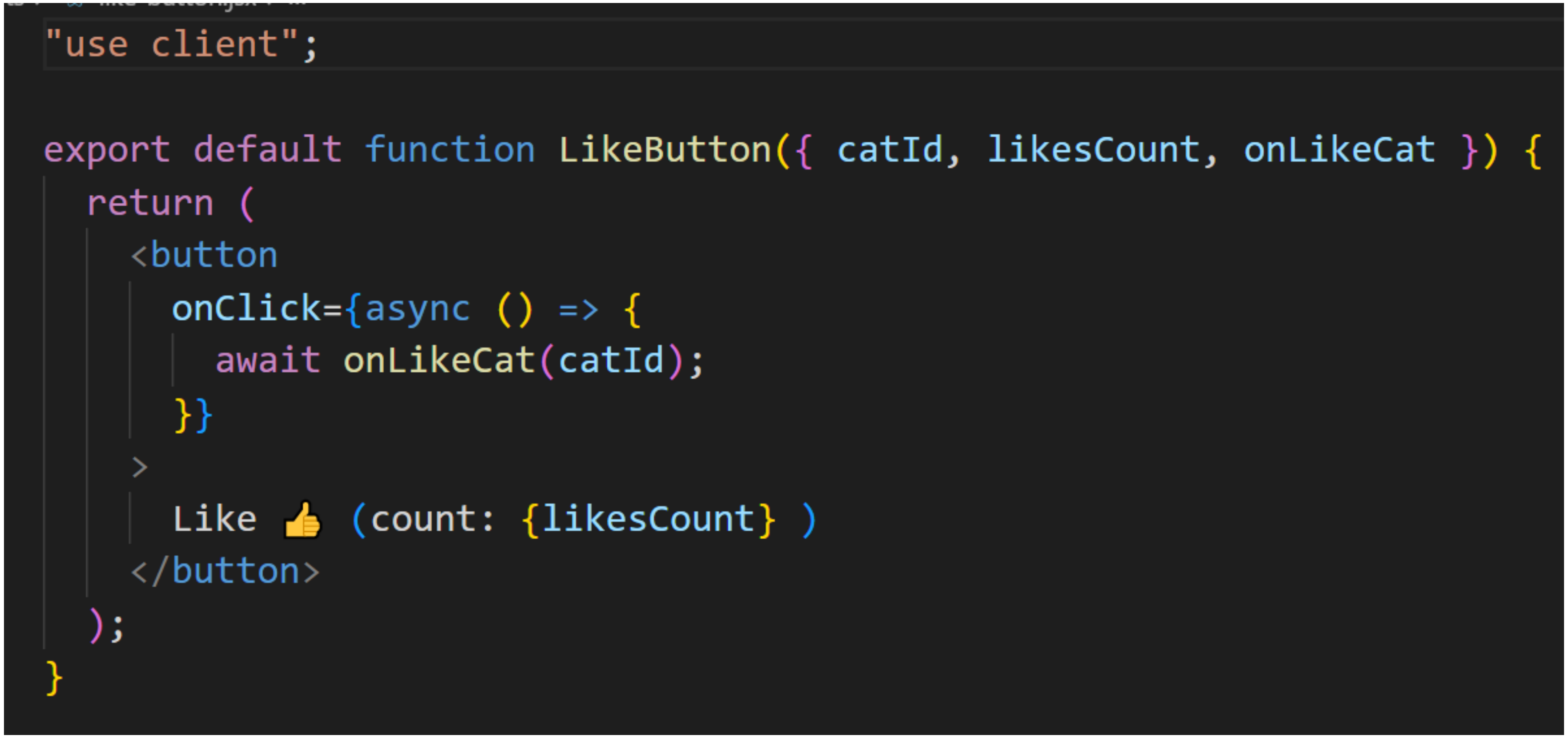
- After handling the form submission, you can re-render the UI to `ensure` the correct data is displayed on the client after mutation
  - **`revalidatePath`** allows re-rendering the current page
  - **`redirect`** allows redirecting to another page

Calling Server Action function from a client-side component

- Server action function can be passed and called from a client component



```jsx
<LikeButton
  catId={cat.id}
  likesCount={cat.likes}
  onLikeCat={likeCat}
/>
```



```jsx
"use client";

export default function LikeButton({ catId, likesCount, onLikeCat }) {
  return (
    <button
      onClick={async () => {
        await onLikeCat(catId);
      }}
    >
      Like 👍 (count: {likesCount} )
    </button>
  );
}
```