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 from pages/components without needing to create an in-between API layer
  - Simpler alternative to using client-side fetch and API routes for data mutations
  - Reduce 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 server-action function to a client-side component to directly invoke it

Example - Handle Form Submission

```
async function onSubmit(formData) {
  "use server";
  const cat = {
    name: formData.get("title"),
    imageUrl: formData.get("imageUrl"),
    breed: formData.get("breed"),
  };
  await updateCat(catId, cat);
  // Revalidate to re-render the UI
  revalidatePath(`/cats/${params.id}/edit`);
}

return (
  <div>
    <form action={onSubmit}>
      <input name="id" type="hidden" defaultValue={cat?.id} />
```

Re-rendering after Data Mutation

- After data mutation (e.g., handling the form submission), you can re-render the UI to ensure the correct data is displayed on the client using:
  - **`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-side component

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

```
"use client";
import { useState } from "react";

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