BY ABDULLAH RADWAN

SERVER ACTIONS IN NEXT.JS

معلومات مهمه و نصائح !

app/page.tsx TypeScript

```tsx
export default async function Page() {
  const publishVersion = await getLatestVersion();

  async function publish() {
    "use server";
    if (publishVersion !== await getLatestVersion()) {
      throw new Error('The version has changed since pressing publish');
    }
    ...
  }

  return (
    <form>
      <button formAction={publish}>Publish</button>
    </form>
  );
}
```

CLOSURE

- **Server Actions** بتستخدم مفهوم الـ **Closure** لما تستدعي متغيرات معرفة في نفس الـ **Server File**
- ال**closure** هو ان ال **function** يكون قادر يشوف **variables** ال **Scope** الي فوقيه
- هنا **function** ال **public** قدرت انها تشوف ال**constant** **publishVersion** مع انها مش في ال**Scope** بتاعها, هو ده ال**Closure**

SECURITY

- احرص دائما انك تعمل **validation** علي ال **data** الي رجعالك من ال**inputs**, ممكن تستخدم مكتبه زي **ZOD**
- و اضافه **authentication** يبقا افضل

app/actions.ts — **VALIDATION** — TypeScript

```ts
'use server'

import { z } from 'zod'

const schema = z.object({
  email: z.string({
    invalid_type_error: 'Invalid Email',
  }),
})

export default async function createUser(formData: FormData) {
  const validatedFields = schema.safeParse({
    email: formData.get('email'),
  })

  // Return early if the form data is invalid
  if (!validatedFields.success) {
    return {
      errors: validatedFields.error.flatten().fieldErrors,
    }
  }

  // Mutate data
}
```

app/actions.ts — **AUTHENTICATION**

```ts
'use server'

import { auth } from './lib'

export function addItem() {
  const { user } = auth()
  if (!user) {
    throw new Error('You must be signed in to perform this action')
  }

  // ...
}
```

app/client-component.tsx TypeSc

```tsx
'use client'

import { updateUser } from './actions'

export function UserProfile({ userId }: { userId: string }) {
  const updateUserWithId = updateUser.bind(null, userId)

  return (
    <form action={updateUserWithId}>
      <input type="text" name="name" />
      <button type="submit">Update User Name</button>
    </form>
  )
}
```

BIND method

- لو عايز تبعت **parameter** لل **server action** زي مثلا **userId**
- ف لازم تستخدم **bind** عشان لو بعت **paramerter** من غير **bind** بالشكل ده :

❌ **const user = updateUser(null,userId)**

ف انت بت**invoke** ال**function** علي طول

- ف وظيفه **bind** انها تنشئ نسخه من ال**function** وتنفزها وقت استدعائها
- ال**null** هنا قيمتها **this** و مش مهمه في ال **Server action**

RevalidatePath

- لو عايز تعيد تحميل صفحه بعد اي **server action** تقدر تستخدم **revalidatepath** و تحط **path** الصفحه في ال **param**

app/actions.ts — TypeScript

```ts
'use server'

import { revalidatePath } from 'next/cache'

export async function createPost() {
  try {
    // ...
  } catch (error) {
    // ...
  }

  revalidatePath('/posts')
}
```

RevalidateTag

- اما لو عايز تحدث **requests** معينه بعد اي **server action,** ف تقدر تبعت **next param** مع ال **fetch** و جوه ال **object** تبعت ال **tags** الي انت حددتها في ال **Server action**

app/actions.ts — TypeScript

```ts
'use server'

import { revalidateTag } from 'next/cache'

export async function createPost() {
  try {
    // ...
  } catch (error) {
    // ...
  }

  revalidateTag('posts')
}
```

```ts
const posts = await fetch('https://api.example.com/posts', {
  next: { tags: ['posts'] }, // وضع وسم 'posts' على هذه البيانات
});
```

useActionState

- تقدر تستخدم **hook** زي **useActionState** بتديك ال **server action , pending state** و **state** انت بتعرفها في ال **action**

1

```ts
'use server'

import { redirect } from 'next/navigation'

export async function createUser(prevState: any, formData: FormData) {
  const res = await fetch('https://...')
  const json = await res.json()

  if (!res.ok) {
    return { message: 'Please enter a valid email' }
  }

  redirect('/dashboard')
}
```

ال **state** الي راجعه

app/ui/signup.tsx

2

TypeScript

```ts
'use client'

import { useActionState } from 'react'
import { createUser } from '@/app/actions'

const initialState = {
  message: '',
}

export function Signup() {
  const [state, formAction, pending] = useActionState(createUser, initialState)

  return (
    <form action={formAction}>
      <label htmlFor="email">Email</label>
      <input type="text" id="email" name="email" required />
      {/* ... */}
      <p aria-live="polite">{state?.message}</p>
      <button disabled={pending}>Sign up</button>
    </form>
  )
}
```

اللهم انصر اخواتنا المستضعفين في فلسطين وفي جميع بلاد المسلمين