



Rendering_Modes


◆ SSR (Server-Side Rendering)

- **Default in Next.js 14 App Router** (Server Components render on server).
- Data fetched fresh **on every request**.
-  Always up-to-date, SEO friendly.
-  Slower than static, more server load.
- **How to use (Axios):**

```
import axios from "axios";
```

```
export default async function Page() {  
  const res = await axios.get("https://api.example.com/data");  
  return <div>{res.data.message}</div>;  
}
```

◆ SSG (Static Site Generation)

- Page is built **once at build time**.
- Any content which does not have network calls is a static page id default.
- Super fast, but data becomes **stale until redeploy**.
-  Best for blogs, docs, portfolios.
- **How to use (Axios + cache):**

```
import axios from "axios";
```



```
import { cache } from "react";
```

```
const getData = cache(async () => {  
  const res = await axios.get("https://api.example.com/data");
```

```
    return res.data;
  });
```

```
default async function Page() {
  const data = await getData();
  return <div>{data.message}</div>;
}
```

◆ ISR (Incremental Static Regeneration)

- Like SSG but can **auto-update after a set time**.
-  Combines speed of static with fresh content.
-  Not real-time (slightly stale).

- **How to use (Axios + API Route with revalidate):**

- /app/api/data/route.ts

```
export async function GET() {
  const res = await fetch("https://api.example.com/data", {
    next: { revalidate: 60 }, // regenerate every 60s
  });
  const data = await res.json();
  return Response.json(data);
}
```



- /app/isr/page.tsx

```
import axios from "axios";
```

```
export default async function Page() {
  const res = await axios.get("http://localhost:3000/api/data");
}
```

```
    return <div>{res.data.message}</div>;  
  }  
}
```

◆ CSR (Client-Side Rendering)

- Rendering happens in the **browser after load**.
-  Always fresh, interactive.
-  Poor SEO, slower first paint.
- **How to use (Axios + client component):**

```
"use client";
```

```
import { useEffect, useState } from "react";
```

```
import axios from "axios";
```

```
export default function Page() {
```

```
  const [data, setData] = useState<any>(null);
```

```
  useEffect(() => {
```

```
    axios.get("https://api.example.com/data").then((res) => setData(res.data));
```

```
  }, []);
```

```
  return <div>{data ? data.message : "Loading..."}</div>;
```

```
}
```

Default in Next.js 14: SSR (Server Components run on server). (Axios used:)

- Use "use client" for CSR.
- Use `cache()` for SSG.
- Use `fetch(..., { revalidate })` in API route for ISR.

Data Fetching in Next.js 13+ (App Router)

- **fetch in a Server Component** → runs on the server, data is fetched before sending HTML.
- You can also control caching:
 - `fetch(url, { cache: 'no-store' })` → always fetch fresh data (**SSR-like**).
 - `fetch(url, { next: { revalidate: 60 } })` → ISR (revalidate every 60 seconds).
 - Default fetch → cached and static (**SSG**).