**DATA FETCHING**

**Client-side Data Fetching with SWR**

[SWR is a React Hooks library for data fetching](https://github.com/vercel/swr)[1](https://github.com/vercel/swr). [The name “SWR” is derived from stale-while-revalidate, a cache invalidation strategy](https://github.com/vercel/swr)[1](https://github.com/vercel/swr). [SWR first returns the data from cache (stale), then sends the fetch request (revalidate), and finally comes with the up-to-date data again](https://github.com/vercel/swr)[1](https://github.com/vercel/swr).

Here’s a simple example of how to use SWR:

import useSWR from 'swr'

function Profile() {

const { data, error } = useSWR('/api/user', fetch)

if (error) return <div>Failed to load</div>

if (!data) return <div>Loading...</div>

return <div>Hello {data.name}!</div>

}

**Integrating External APIs**

[API integration is the process of using APIs to connect two or more software systems in order to facilitate the seamless transfer of data](https://github.com/vercel/swr)[2](https://www.postman.com/api-platform/api-integration/). [When you learn how to use an API, you’re able to use services that would otherwise take you a long time to code yourself](https://github.com/vercel/swr)[3](https://snipcart.com/blog/integrating-apis-introduction).

Here’s a simple example of how to integrate an external API using fetch:

fetch('https://api.example.com/data')

.then(response => response.json())

.then(data => console.log(data))

.catch(error => console.error('Error:', error));

**Handling Data with React Query**

[React Query is a powerful tool for handling data fetching, caching, and state management in React applications](https://github.com/vercel/swr)[4](https://refine.dev/blog/react-query-guide/). [It provides a way of handling server states using the useQuery hook](https://github.com/vercel/swr)[4](https://refine.dev/blog/react-query-guide/).

Here’s a simple example of how to use React Query:

import { useQuery } from 'react-query';

function App() {

const { isLoading, error, data } = useQuery('repoData', () =>

fetch('https://api.github.com/repos/tannerlinsley/react-query').then(res =>

res.json()

)

);

if (isLoading) return 'Loading...';

if (error) return 'An error has occurred: ' + error.message;

return (

<div>

<h1>{data.name}</h1>

<p>{data.description}</p>

</div>

);

}

**Using the Latest Next.js Version**

[Next.js is a React framework for building full-stack web applications](https://github.com/vercel/swr)[5](https://nextjs.org/docs). [To update to the latest Next.js version (version 13 as of the time of writing), you can run the following command using your preferred package manager6](https://nextjs.org/docs/pages/building-your-application/upgrading/version-13):

npm i next@latest react@latest react-dom@latest

[Please note that you should also upgrade @types/react and @types/react-dom to their latest versions if you are using TypeScript](https://github.com/vercel/swr)[6](https://nextjs.org/docs/pages/building-your-application/upgrading/version-13).

I hope this helps! Let me know if you have any other questions.