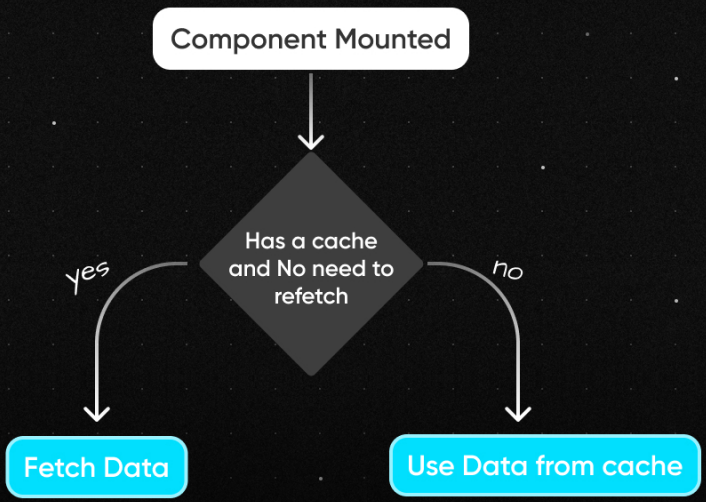
RTK Query

in React



codewithsloba.com

Swipe →

Simplified Data Fetching

RTK Query eliminates the need to write boilerplate code for fetching data from APIs.

```
import { createApi, fetchBaseQuery } from
'@reduxjs/toolkit/query/react';

const booksApi = createApi({
  reducerPath: 'booksApi',
  baseQuery: fetchBaseQuery({ baseUrl:
'https://api.example.com/books' }),
  endpoints: (builder) => ({
    getBooks: builder.query({
      query: () => '/all',
    }),
  }),
});
```

It provides a declarative approach where you define endpoints for various data operations, making your code cleaner and more concise.

React Component Integration

RTK Query generates hooks like `useGetBooksQuery` based on your defined endpoints.

```
import { useGetBooksQuery } from './booksApi';

function BooksList() {
  const { data, isLoading, error } = useGetBooksQuery();

  // ... render UI based on data, isLoading, and error
}
```

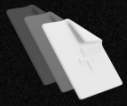
These hooks allow you to interact with the data fetching logic directly within your React components, abstracting away the complexities of asynchronous operations and caching.

Cache Invalidation

RTK Query also handles cache invalidation, ensuring your application always works with the latest data.

```
getBooks: builder.query({
  query: () => '/all',
  providesTags: (result) =>
    result.map((book) => ({ type: 'Book', id: book.id }))
}),
```

It automatically updates the cache whenever there are changes on the server-side.



Automatic Caching

It automatically caches the fetched data, **reducing unnecessary network requests** and improving the overall performance of your application.

Scalability and Maintainability

The modular design of RTK Query promotes code maintainability. It **separates concerns** for data fetching logic and UI components, making it easier to scale your application as it grows.

Efficient Data Fetching and Normalization

RTK Query offers **built-in support** for efficient data fetching techniques like **pagination** and **optimistic updates**. Additionally, it helps with data normalization, which simplifies how you **manage related data entities** within your application.

Swipe →

Subscribe on YouTube to learn more