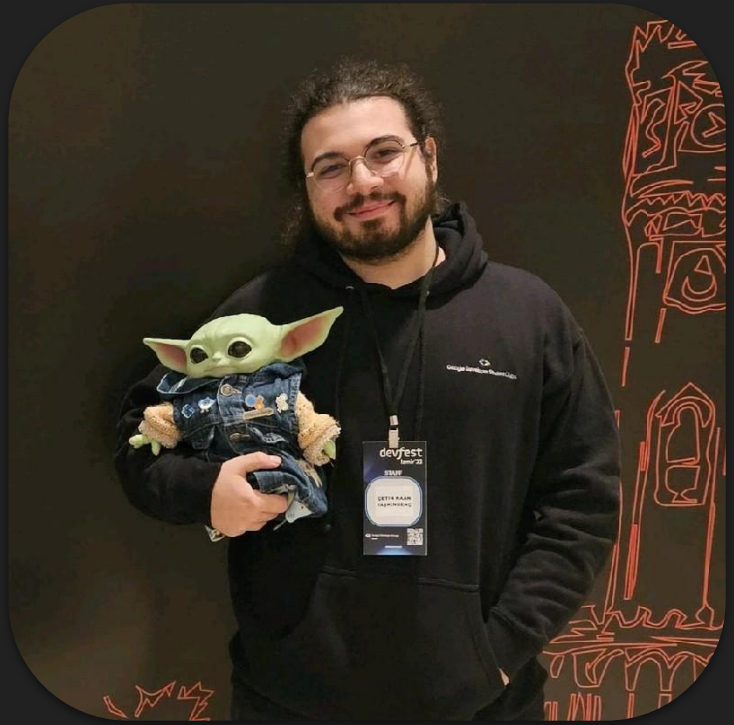
Data Fetching Like a Pro using Tanstack Query

@Çetin Kaan Taşkıngenç

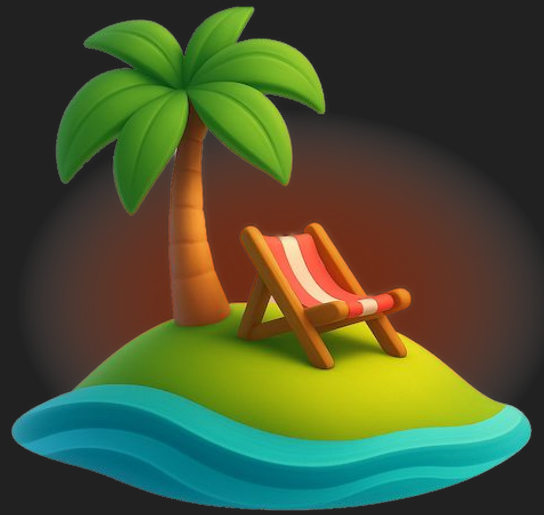
Selamlar 👋

- React Developer @MagiClick
- Security enthusiast
- Ex Gold Microsoft Gold Ambassador & TR Program Lead
- Ex GDG Member & GDSC Lead



Ajanda

- Tanstack Query Nedir
- Neden Tanstack Query
- TanStack Query Setup
- useQuery
- data caching, loading, query params
- Conditional Queries
- Reusable Queries
- Typescript Support
- Multiple Queries



TanStack Query Nedir

- Tanner Linsley tarafından geliştirilmiştir
- En popüler veri yönetme kütüphanesinden biridir
- 6 React uygulamasından birisi kullanıyor
- Otomatik cacheleme özelliğine sahip
- Performans odaklıdır
- State yönetimini çok kolaylaştırır
- Eski adı React Query

Neden Tanstack Query

useState
create a value that is preserved across renders and triggeres a re-render when it changes

useEffect
synchronize a component with some external system

useRef
create a value that is preserved across renders, but won't trigger a re-render when it changes

useContext
get access to what was passed to a Context's Provider

useReducer
create a value that is preserved across renders and triggers a re-render when it changes, using the reducer pattern

useMemo
cache the result of a calculation between renders

useCallback
cache a function between renders

useLayoutEffect
synchronize a component with some external system before the browser paints the screen

useSyncExternalStore
subscribe to an external store

useEffectEvent
encapsulate a side effect that synchronizes your component with some outside system

```
import * as React from "react"
import PokemonCard from "./PokemonCard"
import ButtonGroup from "./ButtonGroup"

export default function App () {
  const [id, setId] = React.useState(1)
  const [pokemon, setPokemon] = React.useState(null)

  React.useEffect(() => {
    const handleFetchPokemon = async () => {
      setPokemon(null)

      const res = await fetch(`https://pokeapi.co/api/v2/pokemon/${id}`)
      const json = await res.json()
      setPokemon(json)
    }

    handleFetchPokemon()
  }, [id])

  return (
    <>
      <PokemonCard data={pokemon}/>
      <ButtonGroup handleSetId={setId} />
    </>
  )
}
```

```jsx
export default function App () {
  const [id, setId] = React.useState(1)
  const [pokemon, setPokemon] = React.useState(null)
  const [isLoading, setIsLoading] = React.useState(true)
  const [error, setError] = React.useState(null)

  React.useEffect(() => {
    const handleFetchPokemon = async () => {
      setPokemon(null)
      setIsLoading(true)
      setError(null)

      try {
        const res = await fetch(`https://pokeapi.co/api/v2/pokemon/${id}`)

        if (res.ok === false) {
          throw new Error(`Error fetching pokemon #${id}`)
        }

        const json = await res.json()

        setPokemon(json)
        setIsLoading(false)
      } catch (e) {
        setError(e.message)
        setIsLoading(false)
      }
    }

    handleFetchPokemon()
  }, [id])

  return (
    <>
      <PokemonCard
        isLoading={isLoading}
        data={pokemon}
        error={error}
      />
      <ButtonGroup handleSetId={setId} />
    </>
  )
```

```
const [error, setError] = React.useState(null)

React.useEffect(() => {
  const handleFetchPokemon = async () => {
    setPokemon(null)
    setIsLoading(true)
    setError(null)

    try {
      const res = await fetch(`https://pokeapi.co/api/v2/pokemon/${id}`)

      if (res.ok === false) {
        throw new Error(`Error fetching pokemon #${id}`)
      }

      const json = await res.json()
```

```jsx
import * as React from "react"
import PokemonCard from "./PokemonCard"
import ButtonGroup from "./ButtonGroup"

export default function App () {
  const [id, setId] = React.useState(1)
  const [pokemon, setPokemon] = React.useState(null)
  const [isLoading, setIsLoading] = React.useState(true)
  const [error, setError] = React.useState(null)

  React.useEffect(() => {
    let ignore = false

    const handleFetchPokemon = async () => {
      setPokemon(null)
      setIsLoading(true)
      setError(null)

      try {
        const res = await fetch(`https://pokeapi.co/api/v2/pokemon/${id}`

        if (ignore) {
          return
        }

        if (res.ok === false) {
          throw new Error(`Error fetching pokemon #${id}`)
        }

        const json = await res.json()

        setPokemon(json)
        setIsLoading(false)
      } catch (e) {
        setError(e.message)
        setIsLoading(false)
      }
    }

    handleFetchPokemon()

    return () => {
      ignore = true
    }
  }, [id])

  return (
    <>
      <PokemonCard
        isLoading={isLoading}
        data={pokemon}
        error={error}
      />
      <ButtonGroup handleSetId={setId} />
    </>
  )
}
```