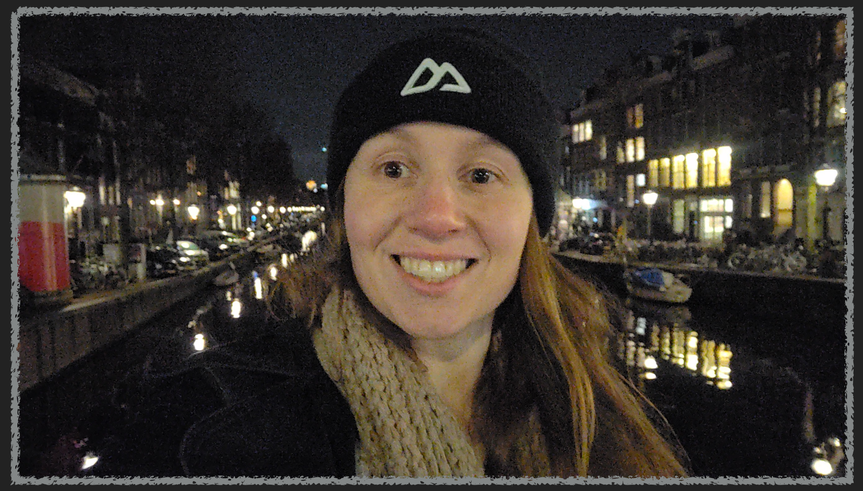
VUEJS AMSTERDAM 2024

EFFICIENT SERVER STATE MANAGEMENT

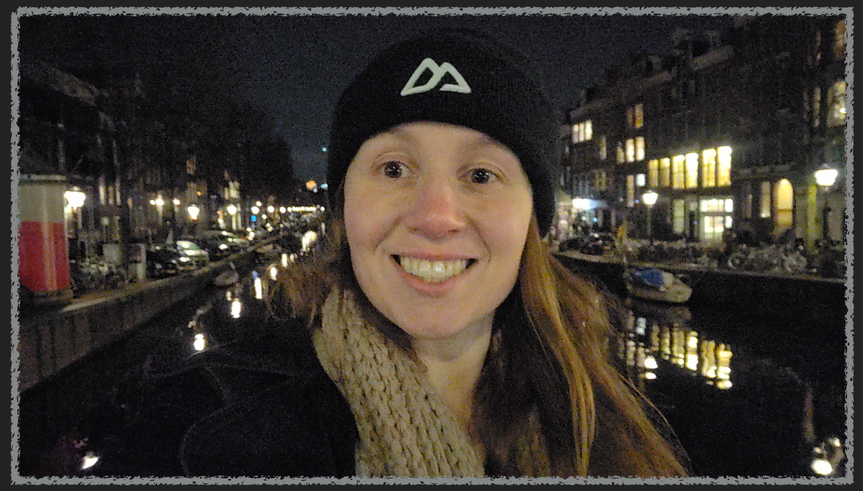
WHO AM I?

- My name is Elise Patrikainen
- Live in Paris
- Senior front-end freelance developer
- Working at Accor
- Co-organiser of the meetup Vue.js Paris



WHO AM I?

- My name is Elise Patrikainen
- Live in Paris
- Senior front-end freelance developer
- Working at Accor
- Co-organiser of the meetup Vue.js Paris
  => looking for speakers



WHAT IS SERVER STATE ?

Spoiler: this is not about SSR.

WHAT IS STATE ?

State is made of:

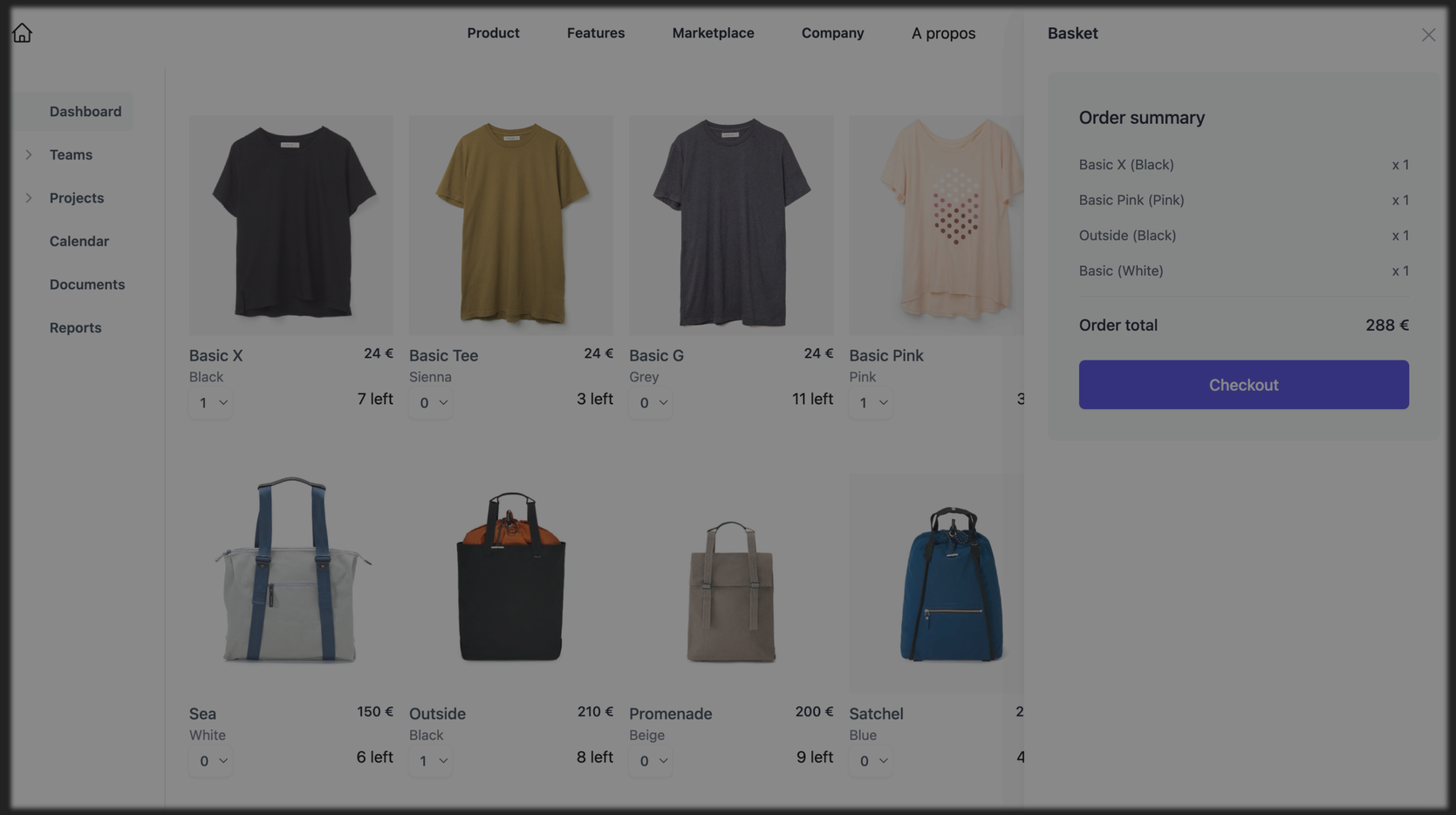
- data fetch from the server
- data owned by our application

WHAT IS STATE ?

State is made of:

- data fetch from the server => server state
- data owned by our application => client state

EXAMPLE


Product
Features
Marketplace
Company
A propos
Basket
Dashboard
Teams
Projects
Calendar
Documents
Reports
Basic X
Black
24 €
7 left
Basic Tee
Sienna
24 €
3 left
Basic G
Grey
24 €
11 left
Basic Pink
Pink
Sea
White
150 €
6 left
Outside
Black
210 €
8 left
Promenade
Beige
200 €
9 left
Satchel
Blue
Order summary
Basic X (Black) x 1
Basic Pink (Pink) x 1
Outside (Black) x 1
Basic (White) x 1
Order total 288 €
Checkout

GLOBAL STATE = SERVER STATE + CLIENT STATE

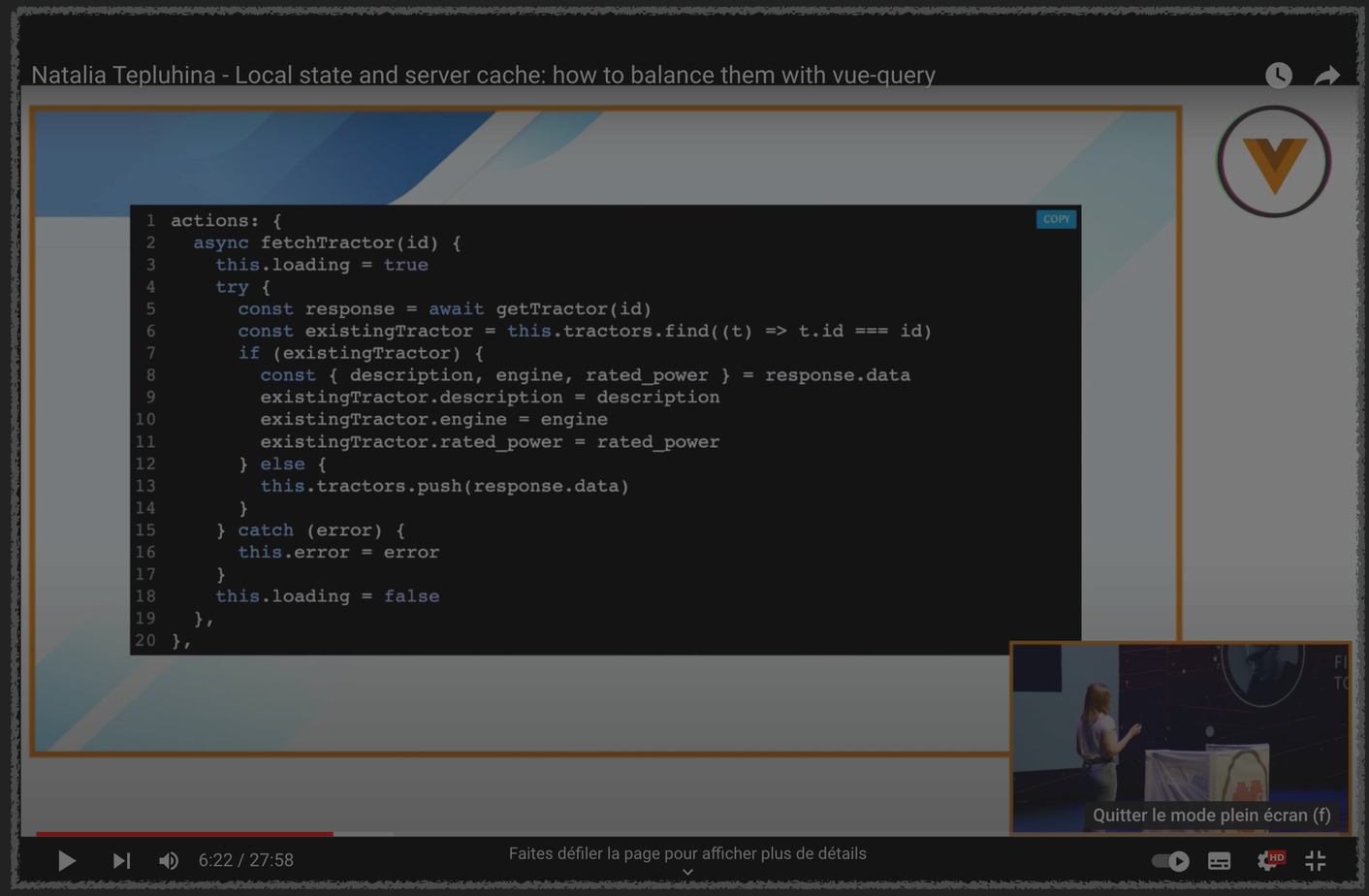
Fetch from the server

Belongs to the client

SERVER STATE	CLIENT STATE
Asynchroneous => fetch (loading / error) => cache	Synchroneous
Shared ownership => refresh	Single ownership

- A concrete example of how quickly server state can become complex :

- A concrete example of how quickly server state can become complex :



WHY IS SERVER STATE MANAGEMENT SO HARD ?

- Handling states related to the fetch (error, loading)?
- Request deduplications?
- How to cache the data?
- When to refresh the data? On component mount? Polling? On window focus? On reconnection?
- Garbage collecting the inactive cache?

2 PATTERNS:

- server state in component scope
  => no cache

- server state in a store
  => no refresh

So what can we do ?

SERVER STATE MANAGEMENT

We can use a server state management library 👀

- One of them belongs to this place:

- One of them belongs to this place:

TANSTACK

TANSTACK

- A set of open source libraries (by Tanner Linsley)
- Framework agnostic

TANSTACK

- A set of open source libraries (by Tanner Linsley)
- Framework agnostic

TANSTACK QUERY

SHORT HISTORY

- ReactQuery
  => Vue Query (wrapper)
- Migrated to TanStack Query
  - JS core
  - frameworks adaptations

TABLE OF CONTENTS:

OVERVIEW OF TANSTACK QUERY

3 NOTIONS:

- Queries: fetch the data 📡
- Mutations : mutate the data 📡
- QueryClient: queries and mutations state ⚙️

TANSTACK QUERY – OVERVIEW

- Queries:

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:

```
const { isPending, isError, data } = useQuery({
    queryKey: ['items'],
    queryFn: () => ofetch<Item[]>(`https://.../products`),
    staleTime: 60 * 1000
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:

```
const { isPending, isError, data } = useQuery({
    queryKey: ['items'],
    queryFn: () => ofetch<Item[]>(`https://.../products`),
    staleTime: 60 * 1000
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:

```
const { isPending, isError, data } = useQuery({
    queryKey: ['items'],
    queryFn: () => ofetch<Item[]>(`https://.../products`),
    staleTime: 60 * 1000
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):

```
const { isPending, isError, data } = useQuery({
    queryKey: ['items'],
    queryFn: () => ofetch<Item[]>(`https://.../products`),
    staleTime: 60 * 1000
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):

```
const itemsQuery = queryOptions({



})

const { isPending, isError, data } = useQuery({
     queryKey: ['items'],
     queryFn: () => ofetch<Item[]>(`https://.../products`),
     staleTime: 60 * 1000
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):

```
const itemsQuery = queryOptions({
    queryKey: ['items'],
    queryFn: () => ofetch<Item[]>(`https://.../products`),
    staleTime: 60 * 1000
})

const { isPending, isError, data } = useQuery({



})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):

```
const itemsQuery = queryOptions({
     queryKey: ['items'],
     queryFn: () => ofetch<Item[]>(`https://.../products`),
     staleTime: 60 * 1000
})
```

```
const { isPending, isError, data } = useQuery(itemsQuery)
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):

```
const itemsQuery = queryOptions({
     queryKey: ['items'],
     queryFn: () => ofetch<Item[]>(`https://.../products`),
     staleTime: 60 * 1000
})
```

```
const { isPending, isError, data } = useQuery(itemsQuery)
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):

```
const itemsQuery = queryOptions({
     queryKey: ['items'],
     queryFn: () => ofetch<Item[]>(`https://.../products`),
     staleTime: 60 * 1000
})

const { isPending, isError, data } = useQuery(itemsQuery)
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
    - → Increase queries reusability ♻️

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
    - Increase queries reusability ♻️
    - Improves type safety ✨

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:

```
const { isPending, isSuccess, mutate } = useMutation({
  mutationFn: (body: any) => ofetch('/comment', { method: "POST", body }),
  onError() {
    displayErrorSnackbar()
  }
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:

```
const { isPending, isSuccess, mutate } = useMutation({
  mutationFn: (body: any) => ofetch('/comment', { method: "POST", body }),
  onError() {
    displayErrorSnackbar()
  }
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:

```
const { isPending, isSuccess, mutate } = useMutation({
  mutationFn: (body: any) => ofetch('/comment', { method: "POST", body }),
  onError() {
    displayErrorSnackbar()
  }
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:

DECLARATIVE

IMPERATIVE

```
const { isPending, isSuccess, mutate } = useMutation({
  mutationFn: (body: any) => ofetch('/comment', { method: "POST", body }),
  onError() {
    displayErrorSnackbar()
  }
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:

```
const { isPending, isSuccess, mutate } = useMutation({
  mutationFn: (body: any) => ofetch('/comment', { method: "POST", body }),
  onError() {
    displayErrorSnackbar()
  }
})
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:

```
const { isPending, isSuccess, mutate } = useMutation({
  mutationFn: (body: any) => ofetch('/comment', { method: "POST", body }),
  onError() {
    displayErrorSnackbar()
  }
})

function addComment(comment: string) {
  mutate({ content: comment })
}
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:

```
const { isPending, isSuccess, mutate } = useMutation({
  mutationFn: (body: any) => ofetch('/comment', { method: "POST", body }),
  onError() {
    displayErrorSnackbar()
  }
})

function addComment(comment: string) {
  mutate({ content: comment })
}
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:
- QueryClient:

TANSTACK QUERY - OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:
- QueryClient:

```
const queryClient = useQueryClient()
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:
- QueryClient:

```
const queryClient = useQueryClient()
queryClient.prefetchQuery(itemsQuery)
queryClient.setQueryData(itemsQuery.queryKey, [...oldItems, newItem])
queryClient.invalidateQueries(itemsQuery)
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:
- QueryClient:

```
const queryClient = useQueryClient()
queryClient.prefetchQuery(itemsQuery)
queryClient.setQueryData(itemsQuery.queryKey, [...oldItems, newItem])
queryClient.invalidateQueries(itemsQuery)
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:
- QueryClient:

```
const queryClient = useQueryClient()
queryClient.prefetchQuery(itemsQuery)
queryClient.setQueryData(itemsQuery.queryKey, [...oldItems, newItem])
queryClient.invalidateQueries(itemsQuery)
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:
- QueryClient:

```
const queryClient = useQueryClient()
queryClient.prefetchQuery(itemsQuery)
queryClient.setQueryData(itemsQuery.queryKey, [...oldItems, newItem])
queryClient.invalidateQueries(itemsQuery)
```

TANSTACK QUERY – OVERVIEW

- Queries:
  - Declaring a query:
  - OptionQuery API (v5):
- Mutations:
- QueryClient:

```
const queryClient = useQueryClient()
queryClient.prefetchQuery(itemsQuery)
queryClient.setQueryData(itemsQuery.queryKey, [...oldItems, newItem])
queryClient.invalidateQueries(itemsQuery)
```

DEMO

TANSTACK QUERY ALTERNATIVES

VUE APOLLO

TANSTACK QUERY ALTERNATIVES

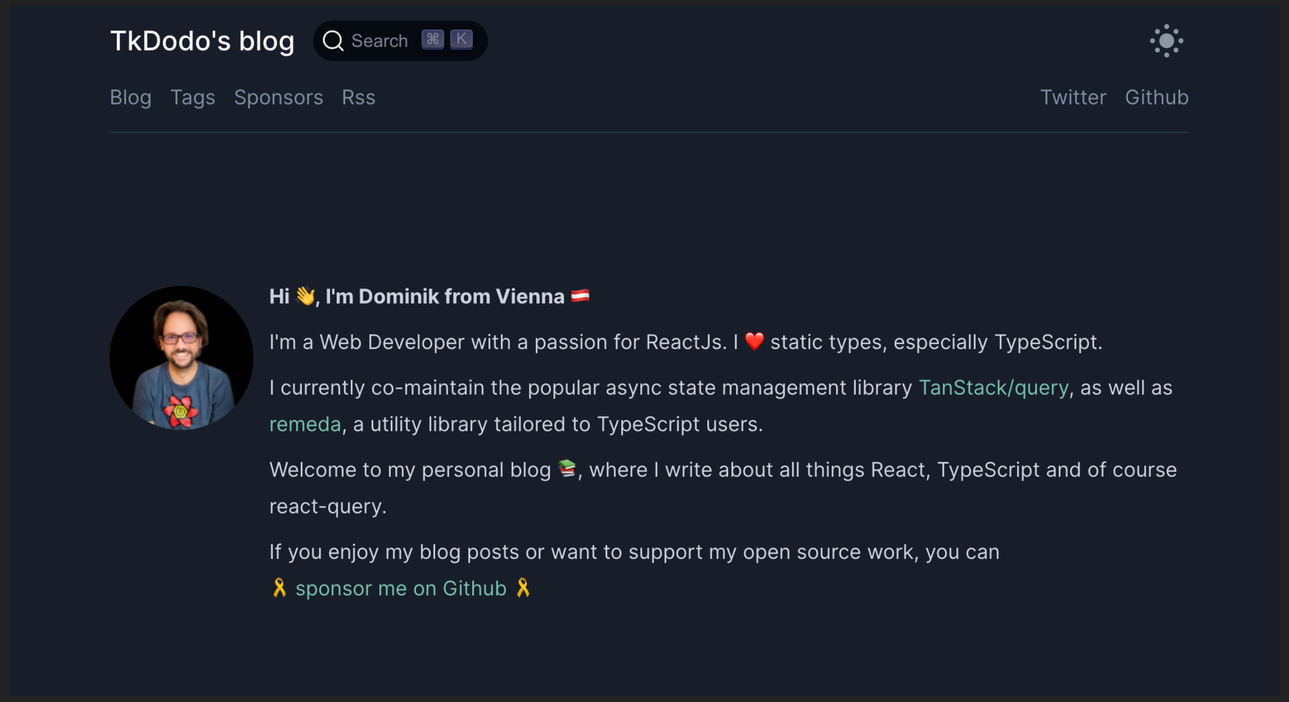
VUE APOLLO

PINIA COLADA (WIP)

RESOURCES

TK DODO'S BLOG

- https://tkdodo.eu/blog/



DEMO

SLIDES

- Demo: https://github.com/ElisePatrikainen/demo-vuejs-amsterdam-2024
- Slides: https://github.com/ElisePatrikainen/slides-vuejs-amsterdam-2024

THANK YOU