

Düsseldorf, 09.02.2021

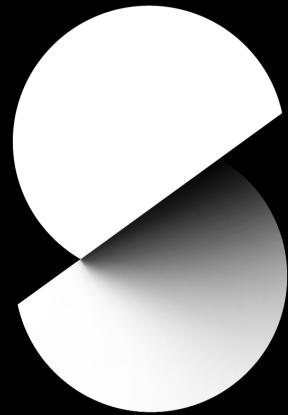


# Pufferspeicher und andere Geschwindigkeitsoptimierungen



Ben

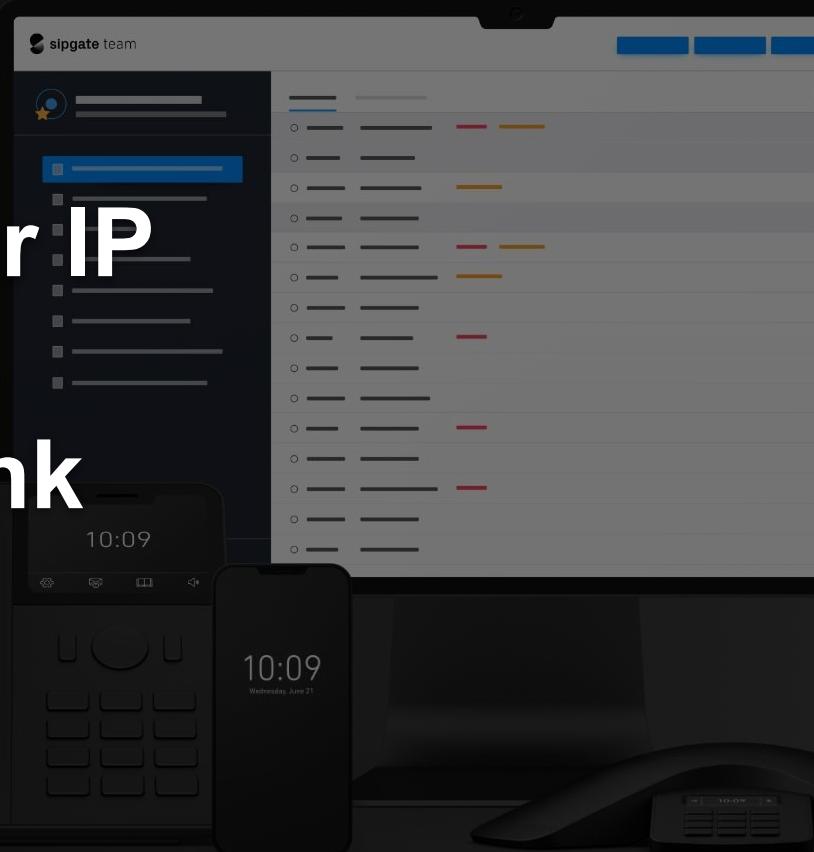
Peter



sipgate



# Voice over IP & Mobilfunk



The image captures a panoramic night view of the Düsseldorf skyline. In the center-right stands the iconic Rheinturm (Rhine Tower), its illuminated spire reaching towards the dark blue sky. To the left, the bridge of the Media Harbour (Medienhafen) is visible, its cables and lights reflected in the calm water of the Rhine River. The city's modern architecture is evident in the large glass-fronted buildings of the Media Harbour, which are brightly lit from within. In the background, the historic buildings of the old town are visible across the river. The overall atmosphere is one of a vibrant, cosmopolitan city at night.

# Düsseldorf

An aerial photograph of a city skyline during the day. In the foreground, there's a large, green park with many trees showing autumn foliage. Beyond the park, several modern buildings of varying heights are visible against a clear blue sky. One prominent building on the left has a grid-like facade, while others are more traditional skyscrapers.

**> 1.000.000 aktive Accounts**

A large group photograph of about 250 people, mostly men, gathered together. They are all looking upwards and slightly towards the camera, with their hands resting near their chins in a thoughtful or posed manner. The background is a dark, modern interior space.

250 Leute



# Pufferspeicher und andere Geschwindigkeitsoptimierungen

# Agenda

- 01** Der Auftrag  
“Bau mir eine App” 
- 02** Herausforderungen  

- 03** Frontend  

- 04** Backend  
Ja..
- 05** Caching  
TOO
- 06** Was noch geht...  
und was wir gern machen

# Disclaimer

Code

Michelle

»Ich brauche eine  
**Bierbörsen-App** und  
das Ding muss  
**performen wie Sau!**  
Wir werden definitiv  
reich damit!«



\$\$\$



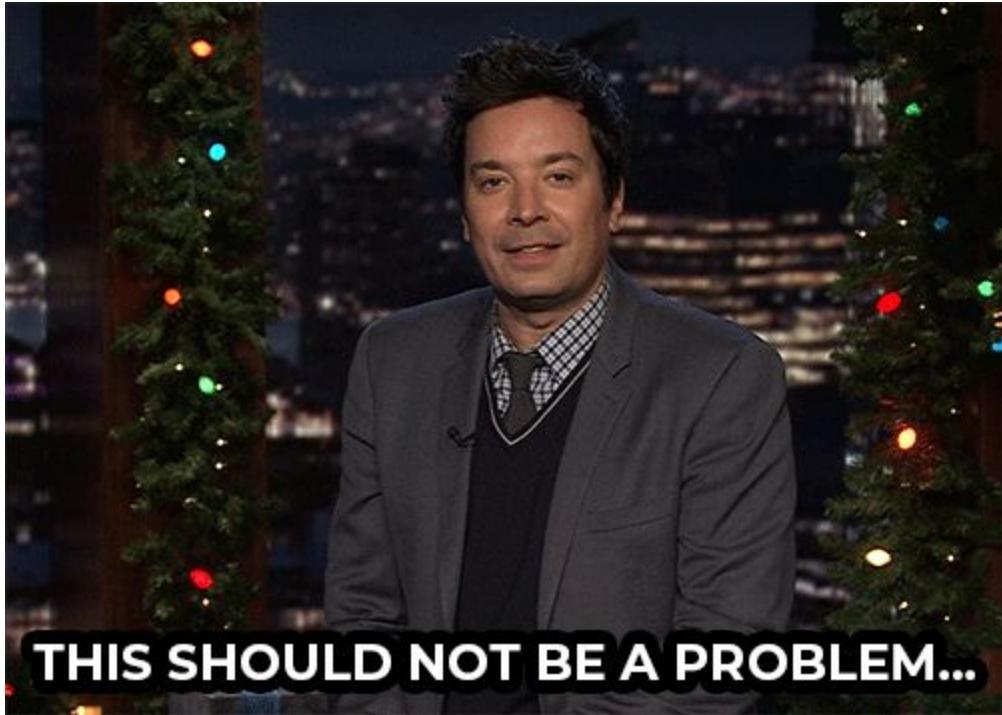
Michelle

»Fang doch einfach schon mal an. Ich besorge in der Zwischenzeit die **API Zugangsdaten** der Lieferanten.«



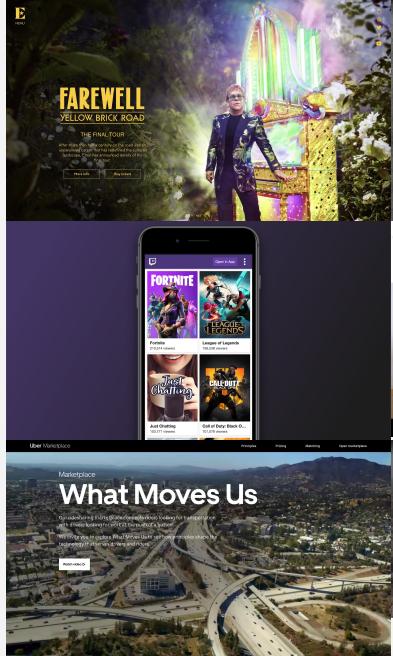
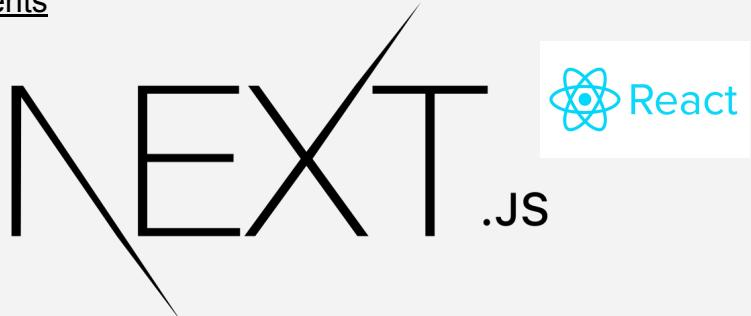
## Herausforderungen

- Viele viele viele Requests (~10000/s)
- Viele Viele Biersorten (TAUSENDE!)
- Daten sollen möglichst aktuell sein
- Begrenzte Serverkapazitäten
- Klein starten



**THIS SHOULD NOT BE A PROBLEM...**

Los gehts



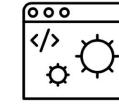
npx create-next-app

# Server Side vs Client Side Rendering



## Server Side Rendering (SSR)

- Server rendert die Seite
- Fast nur HTML & CSS
- Kleinere Bundle Size
- Mehr Batterie für den User
- Cachebar** und kürzere Ladezeiten
- Webserver nötig



## Client Side Rendering (CSR)

- Client muss selber JS ausführen
- Single Page Applications / Web-Apps
- Einfaches Hosting per **CDN**
- JS Framework wird mit ausgeliefert
- Einfacher Code
- Browser regelt rendering

NICHT: Static Site Generation (SSG)

Was ist besser?

- Next.js macht es einfach für uns alle Möglichkeiten zu nutzen.
- Ihr bekommt automatisch eine hybride App
- Alle statischen Seiten sind vorgerendert
- Dynamische Seiten werden im Client gebaut

Datenstand: 8 seconds

0.001€/l	0.004€/l	0.004€/l	0.006€/l	0.006€/l	0.009€/l
Dogfish Head Raison D'Etre	Flying Dog Old Scratch Amber Lager	Leffe Bruin	Marston's India Export Pale Ale	New River Pale Ale	Weihenstephaner Vitus
<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>

0.010€/l	0.010€/l	0.011€/l	0.013€/l	0.014€/l	0.015€/l
Riggwelter Yorkshire Ale	Sleeman Original Draught	Rickard's Dark	James Squire Australian Best Ale	Miller High Life Light	Ridgeway Lump of Coal Dark Holiday Stout
<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>

0.017€/l	0.017€/l	0.019€/l	0.023€/l	0.024€/l	0.025€/l
Shepherd Neame India Pale Ale	Spaten Pils	Tripel Karmeliet	Harp Lager	Santa Fe Pale Ale	Capital Oktoberfest
<a href="#">Jetzt kaufen!</a>					

0.031€/l	0.032€/l	0.033€/l	0.034€/l	0.034€/l	
Old Milwaukee Light	Edelweiss (Weissbier-Hefetrub)	Redhook Sunrye Ale	Bass Country Chase Ale	Chamby Noire	
<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	<a href="#">Jetzt kaufen!</a>	

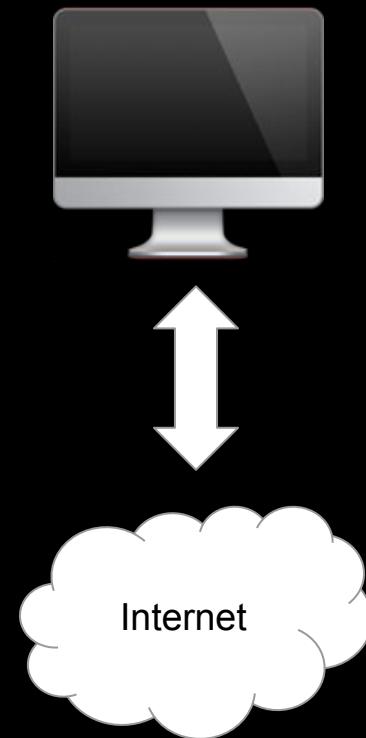
0.035€/l	0.035€/l	0.035€/l	0.036€/l	0.040€/l	
He'Brew Messiah Bold	Molson Golden	Victory Golden Monkey	Catamount Pale Ale	Independence Pale Ale	Amstel 1870



»Gute Nachrichten!  
Die **API Credentials**  
sind am Start.  
Eigentlich können  
wir dann ja **morgen**  
**launchen**, oder?«



# APIs direkt anbinden

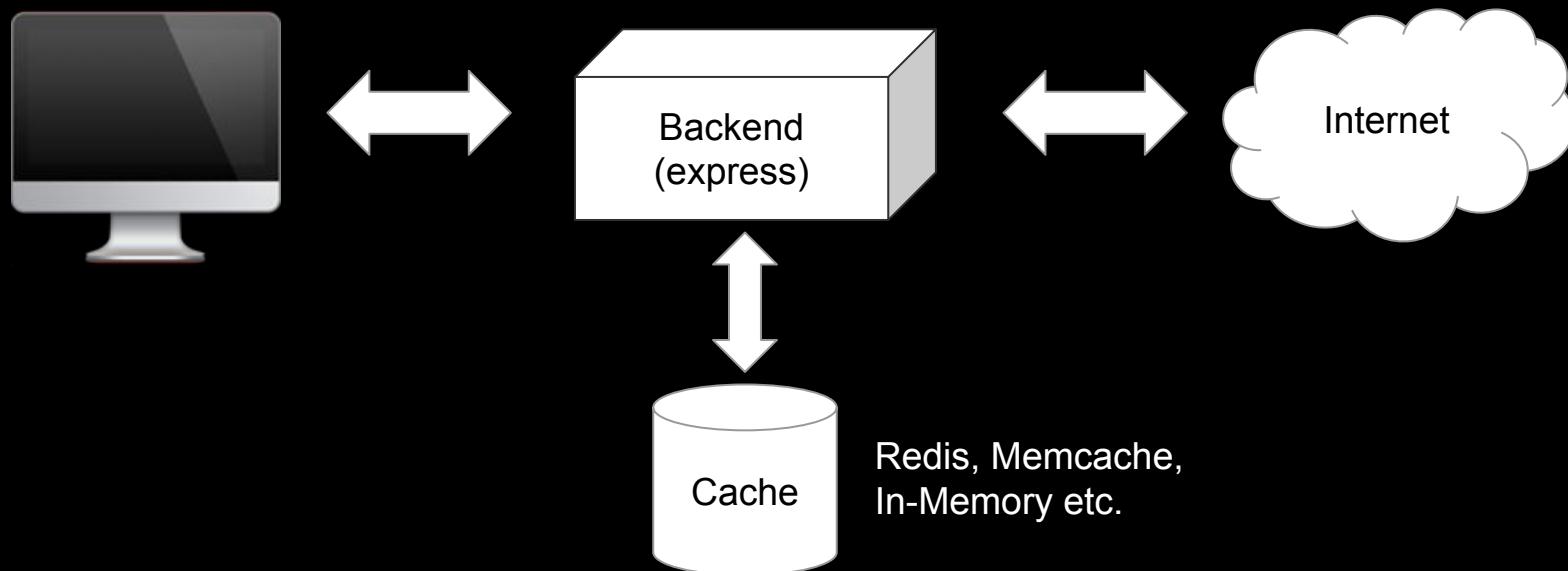




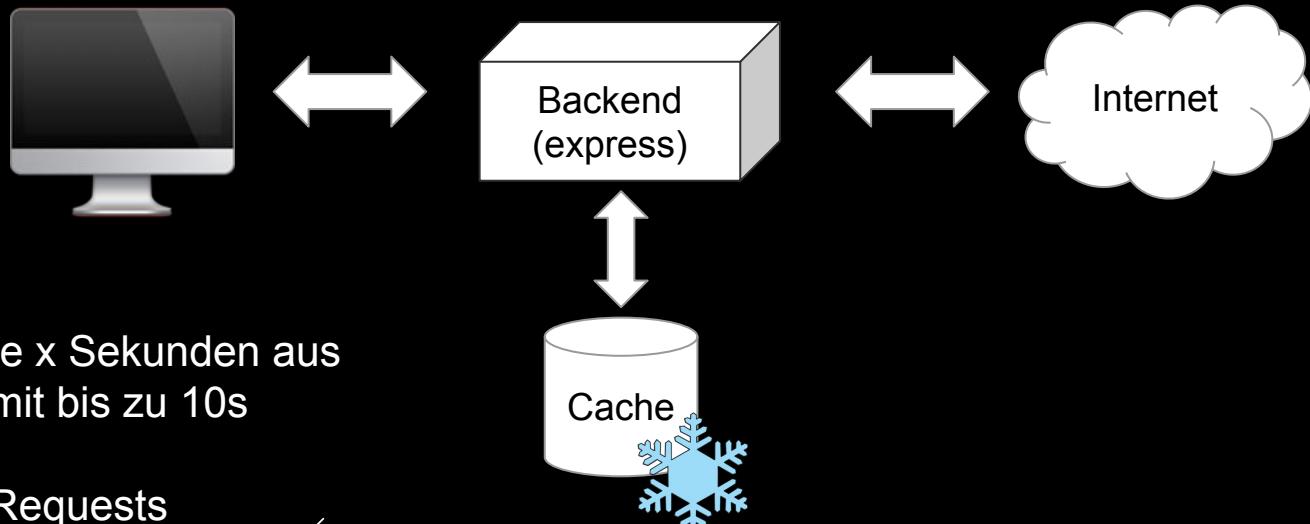
- APIs sind laaaaangsam (Bis zu 10s)
- Kein Anzeichen von SLI / SLA / SLO
- Netzwerk können wir nicht beeinflussen
- UX ist besch\*\*en



# Backend mit Cache

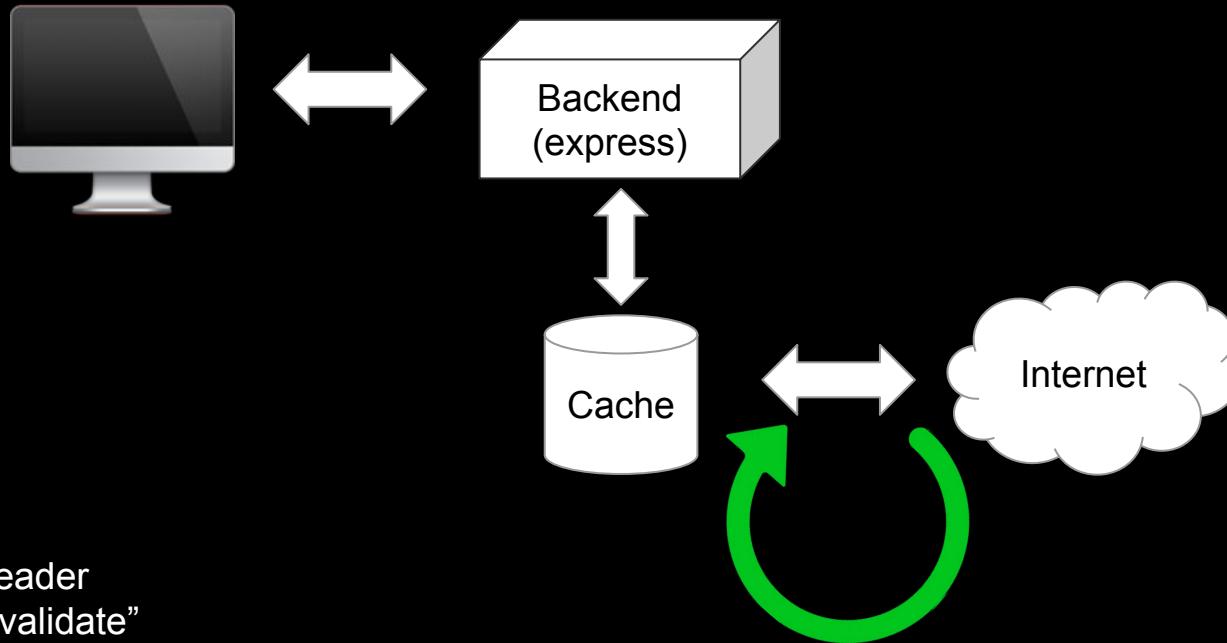


# Problem: Cold Cache



- Cache läuft alle x Sekunden aus
- Kalter Cache mit bis zu 10s Ladezeit
- Bis zu 10000 Requests
- **Backend wird explodieren** 😱

# Introducing: Stale-While-Revalidate



Disclaimer:

Nicht HTTP Header  
“stale-while-revalidate”  
(RFC 5861)



Title

```
1 var cache = {}
2
3 while(true){
4     cache = fetch("http://some-api.com/data");
5     sleep(1000);
6 }
```



- Wir bestimmen die Anzahl der Requests an die APIs
- Unsere Daten sind immer “aktuell”
- Wir haben Caching im Griff
- Unsere Kunden kriegen die Daten schneller



- “App” wird von einem CDN bereitgestellt
- Backend Cache reduziert Ladezeit von 10s auf 1s



- “App” wird von einem CDN bereitgestellt
- Backend Cache reduziert Ladezeit von 10s auf 1s



The screenshot shows the Network tab of the Chrome DevTools interface. At the top, there are tabs for Elements, Console, Sources, Network (which is selected and underlined in blue), Performance, Memory, and Application. Below the tabs, there are several controls: a magnifying glass icon, a dropdown arrow, a search bar, a checkbox for 'Preserve log', and a checked checkbox for 'Disable cache'. There is also a checkbox for 'Hide data URLs' and a 'All' button next to it. The main area displays network requests. One request has a duration of '40 ms'. To the right of this request, there is a context menu with the following options: 'Disabled', 'Online', 'Presets', 'Fast 3G', 'Slow 3G', 'Offline' (which is highlighted with a blue background and has a small orange arrow pointing to it), 'Custom...', and 'Add...'. The menu is titled 'Online'.

- Disabled
- Online
- Presets
- Fast 3G
- Slow 3G
- Offline**
- Custom...
- Add...

Page	Size	First Load JS
/	30.4 kB	93.7 kB
css/de22f3a88674af6dbabb.css	526 B	</div>
_app	0 B	<div> 63.3 kB
index.tsx	2.77 kB	Date 66.1 kB
/404	63.3 kB	</div>
+ First Load JS shared by all	3.19 kB	<div className="
chunks/commons.a94f12.js	10.6 kB	{beerPriceDa
chunks/f6078781a05fe1bcb0902d23dbbb2662c8d200b3.ce80ca.js	41.8 kB	.map((beer
chunks/framework.ae602c.js	6.44 kB	-> <Beer ke
chunks/main.dc76f5.js	529 B	) )>
chunks/pages/_app.0b7960.js	751 B	
chunks/webpack.50bee0.js	202 B	
css/2f26bb9842d84a608fa3.css		

HI 00141 00050



## No internet

Try:

- Checking the network cables, modem, and router
- Reconnecting to Wi-Fi
- [Running Windows Network Diagnostics](#)

ERR\_INTERNET\_DISCONNECTED

# → Service Worker

- ◆ Offline-Caching der ganzen Seite
- ◆ Request Proxy



o

## next-offline

Use [Workbox](#) with Next.js and easily enable offline functionality in your application!

downloads 1.8M/year next-offline v5.0.3  
repo size 2.33 MB

## Installation

```
$ npm install --save next-offline
```

```
$ yarn add next-offline
```

## Usage

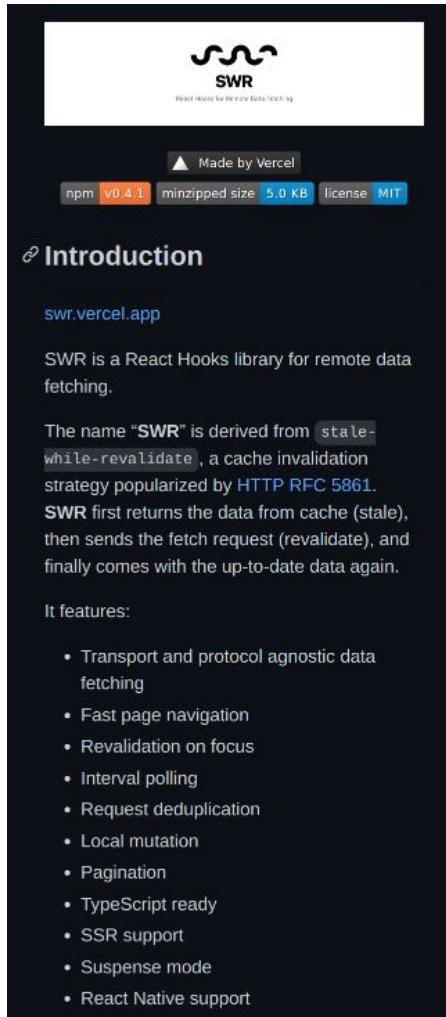
There are two important things to set up, first we need `next-offline` to wrap your next config.

If you haven't yet, create a `next.config.js` in your project.

```
// next.config.js
const withOffline = require('next-offline')

const nextConfig = {
  ...
}

module.exports = withOffline(nextConfig)
```



The screenshot shows the SWR library page on Vercel. At the top, there's a logo with the letters "SWR" and a small "Made by Vercel" badge. Below that, it says "npm v0.4.1 minzipped size 5.0 KB license MIT". The main content starts with a section titled "Introduction" which defines SWR as a React Hooks library for remote data fetching. It explains that the name "SWR" is derived from `stale-while-revalidate`, a cache invalidation strategy. The text also mentions that SWR first returns data from cache (stale), then sends a fetch request (revalidate), and finally provides up-to-date data again. A "It features:" section lists various capabilities: Transport and protocol agnostic data fetching, Fast page navigation, Revalidation on focus, Interval polling, Request deduplication, Local mutation, Pagination, TypeScript ready, SSR support, Suspense mode, and React Native support.

# Libraries wie



- next-offline
- swr (vercel)
- react-query
- ...
- Google Workbox

## → **Query caching**

- ◆ <http://web.dev?q=pale%20ale>



## → **Vorsicht was ihr Cached**

- ◆ Keine Nutzerdaten cachen
- ◆ Input immer validieren (cache poisoning)

## → HTTP-Header

- ◆ max-age
  - “Offline” Caching für Ressourcen
- ◆ cache-control
- ◆ stale-while-revalidate
- ◆ ...

## → Libraries wie



- ◆ apicache
- ◆ express-cache-middleware

Anyway

- Langsame APIs zu schnellen gemacht
- API kann viele viele Requests (min 5000/s)
- Webseite im CDN
- Optimiert für langsames Internet
- Offline Webseite
- Offline API
- Hosting Kosten sind gering (<20€/Monat)

Michelle

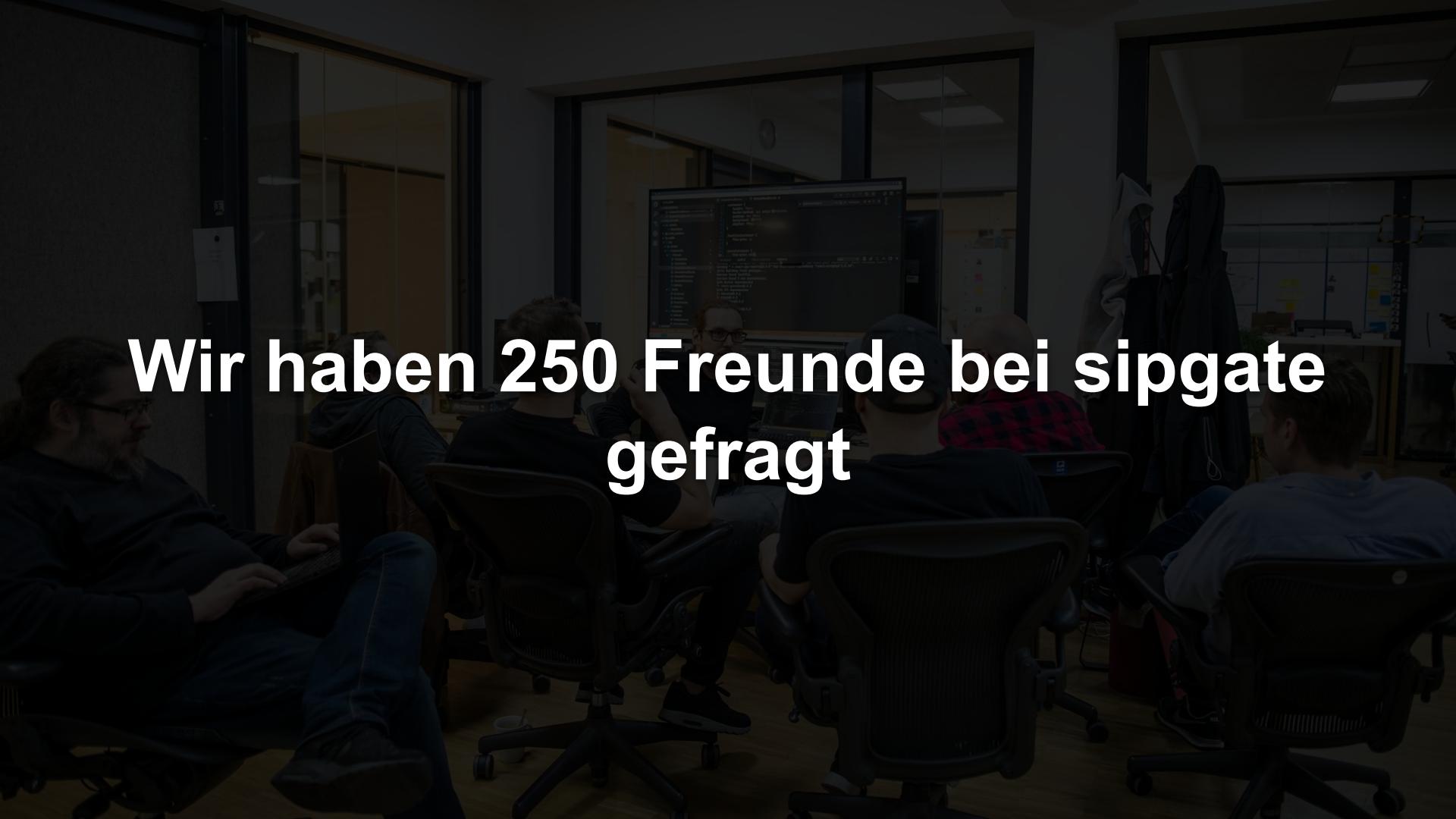
»Jawoll!«



Michelle

»Hier ist dein  
Lamborghini!«





Wir haben 250 Freunde bei sipgate gefragt



googelt

dabei

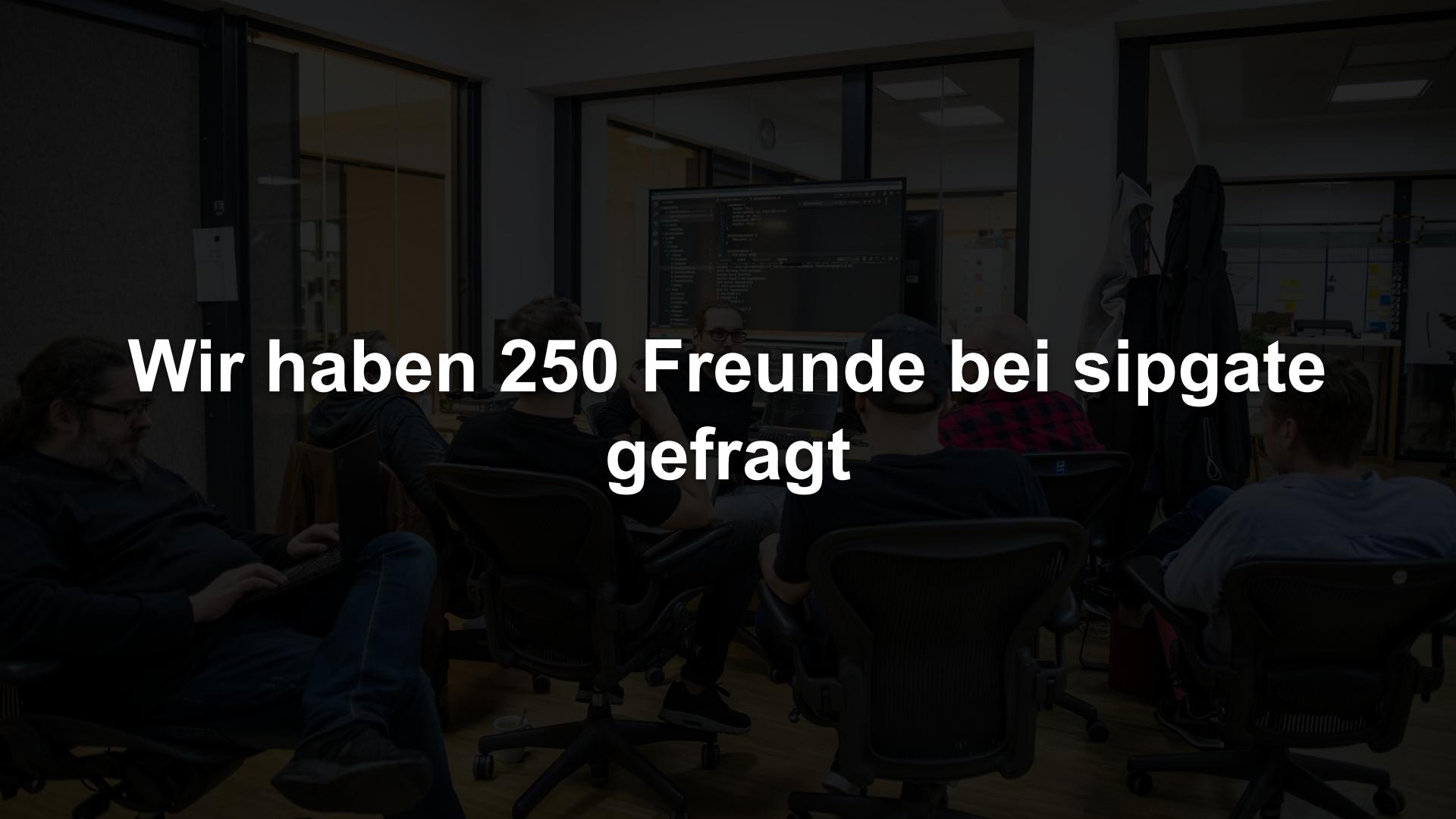
ratlos

hat eine  
Idee

dabei

dabei

hat es  
schon  
fertig



Wir haben 250 Freunde bei sipgate gefragt

»Nicht vergessen: Wenn ihr mit **Pure-Functions** arbeitet, könnt ihr die einfach **memoizen!**«

Google:

- *React.useMemo*
- *lodash.memoize*



»Schmeißt mal die **PNGs** aus eurer App und sagt allen, dass sie euch **SVGs** schicken sollen.«

*Google:*

- *WebP*
- *WebM*



»Habt ihr **GZIP**  
vergessen oder warum  
schickt ihr so viel JSON  
über den Draht?«



Google:

- *Express compress*
- *NGINX “gzip on;”*

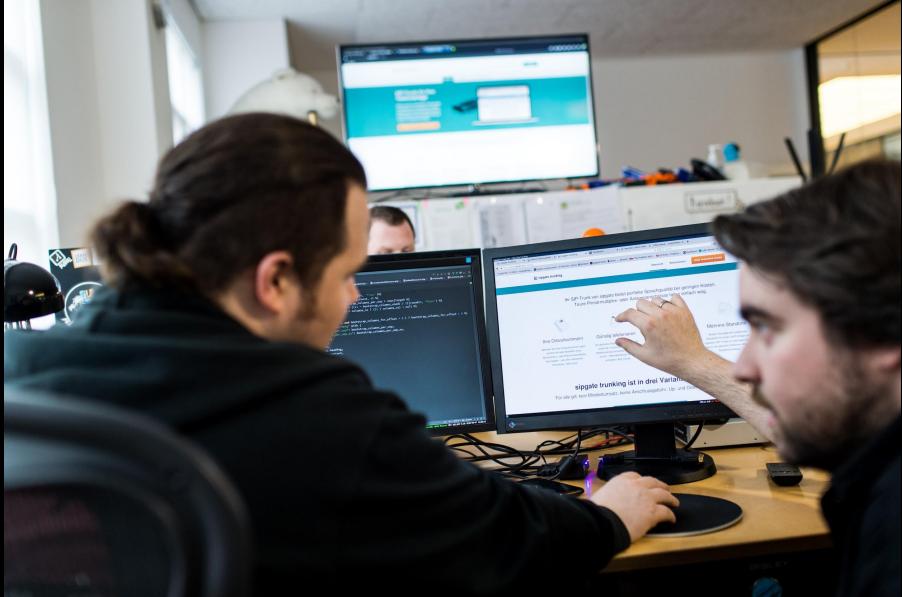
»Diese PDF Library,  
die du da mit auslieferst,  
wird nur von 5% der  
Nutzer gebraucht.«



Google:

- *Code splitting*
- *Dynamic imports*

»Du musst doch nicht die ganze Lib mit in dein **Bundle** packen, wenn du nur zwei Funktionen davon benutzt.«



*Google:*

- *webpack*
- *purgecss*

»Wann hast du zuletzt die **Chrome-Dev-Tools** auf gehabt? Die hätten dir das auch **vorher** alles gesagt.«

Google:

- *Lighthouse*
- *Network panel*



»Warum lädst du überhaupt den ganzen Kram, wenn der User eventuell gar nicht nach unten **scrollt?**«



Google:

- *Infinite scroll*
- *GraphQL*

»Zeig' dem User einfach schon mal an, dass das geklappt hat.  
Entschuldigen kannst du dich immer noch.«

Google:

- *Saga pattern*
- *SLI / SLA / SLO*



»Kann es sein, dass  
deine **Batterie** leer ist  
und du jedes Byte direkt  
auf die Platte schreibst,  
weil der **Cache aus** ist?

«

Google:

- *BBU Monitoring*



»Zeig' dem User deine **Lieblingsbiere** an, wenn der Request nach fünf Sekunden noch immer läuft!«

Google:

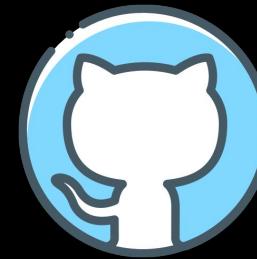
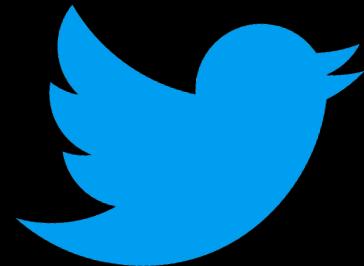
- *Resilience patterns*
- *Netflix*





**Und jetzt kommt ihr...**  
schreibt eure Tricks und Fragen in  
den Chat und kommt ins Talque

Hände  
waschen!



Maske auf!



Bleibt gesund!

# Danke fürs Zuhören

[sipgate.de](http://sipgate.de)

*Ben:* [fuglu.net](http://fuglu.net)

*Peter:* [codedrift.net](http://codedrift.net)

Code auf <http://sipg.at/ct-webdev-2021>



# Shortcomings

- Seite kommt zwar ausm bucket aber nur schnell in einer region
- -> multi region multicast (aka wir launchen international  )
- Single point of failure backend
- Load balancing -> prevent request delay on new deploy