# UX/Frontend

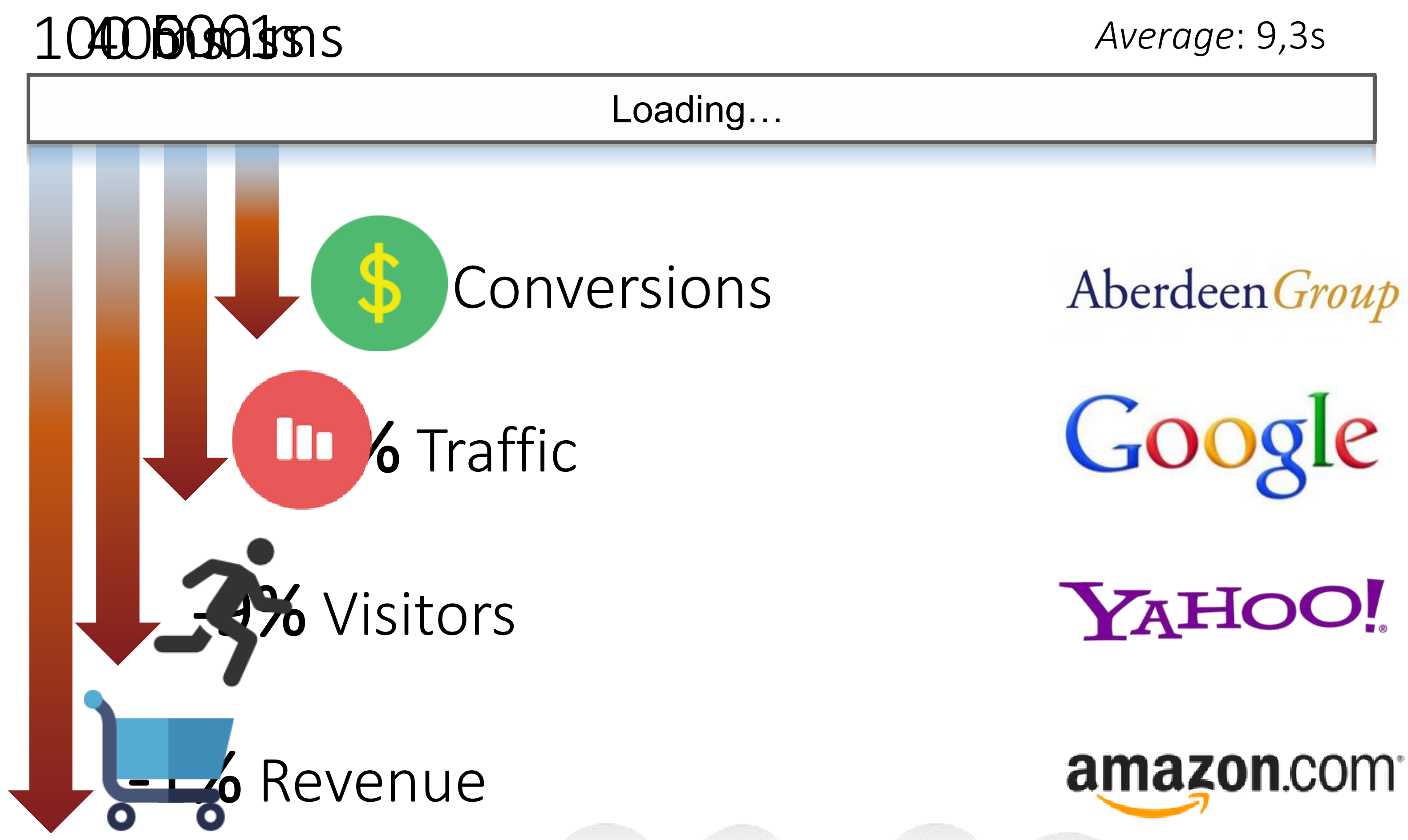
Erik Witt

# Web Performance

## Die effektivsten Techniken aus der Praxis

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

# Why performance matters



# What should be the goal?

| Delay | User Perception |
|---|---|
| 0 – 100 ms | Instant |
| 100 – 300 ms | Small perceptible delay |
| 300 – 1000 ms | Machine is working |
| 1+ s | Mental context switch |
| 10+ s | Taks is abandoned |

Stay under 1000 ms to keep users attention

I. Grigorik, High performance browser networking. O'Reilly Media, 2013.

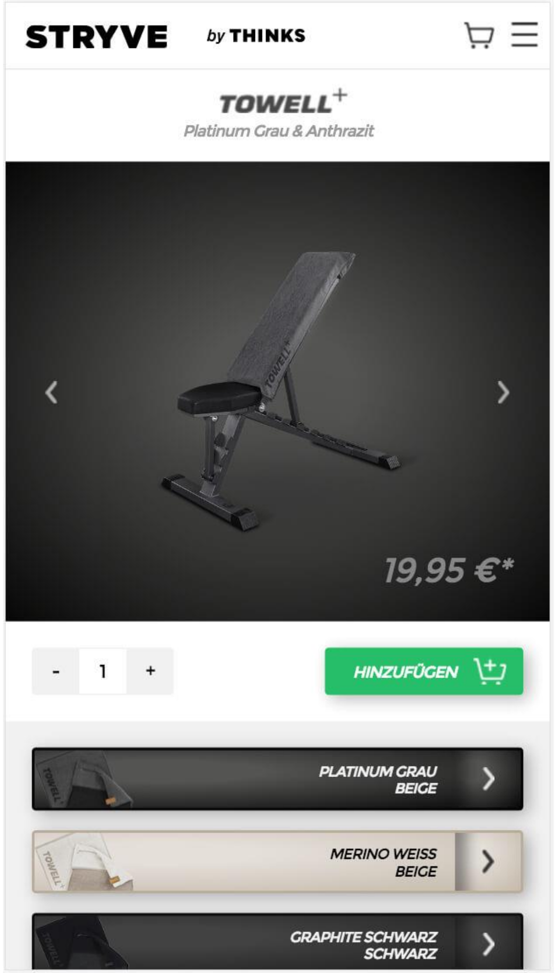
# Concrete Example

## A scalable webshop

THINKS

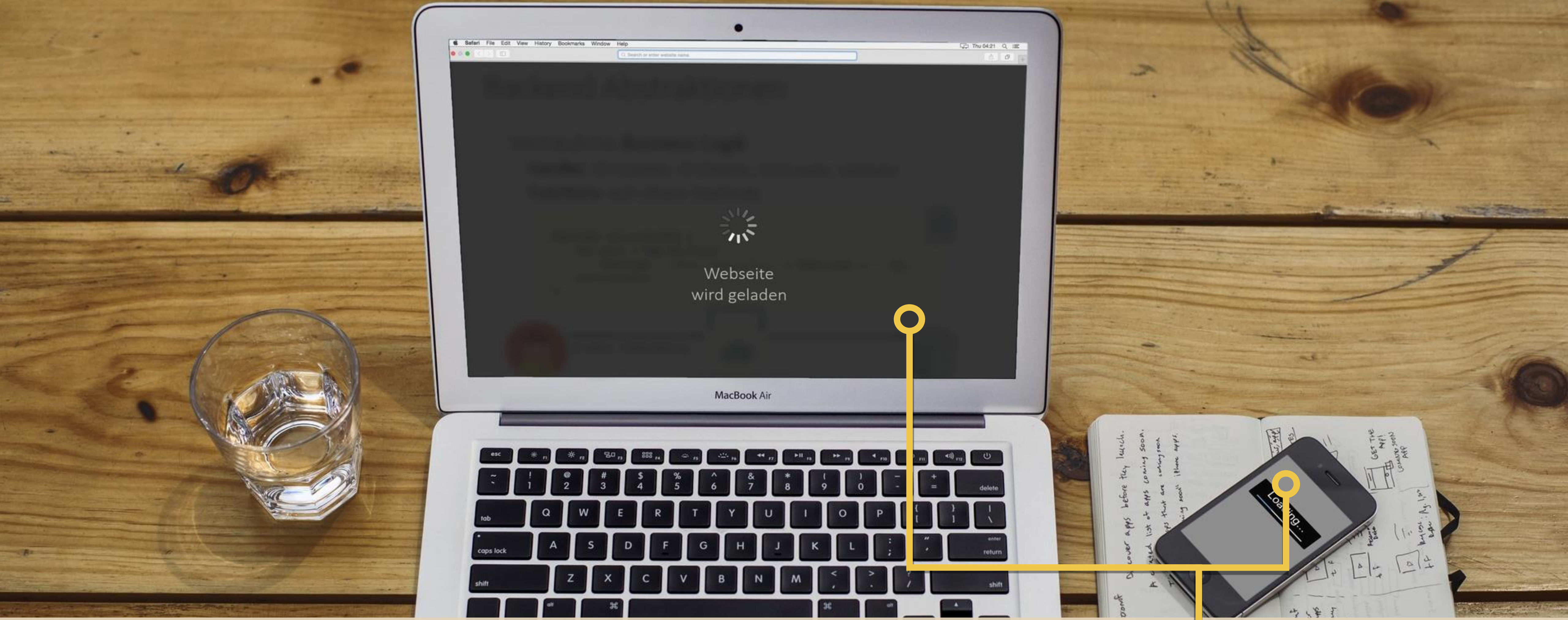
at

DIE HÖHLE DER LÖWEN


Expectations:

- 2.7 Million Viewers
- 300.000 Visitors in 30 minutes
- 20.000 Requests per second



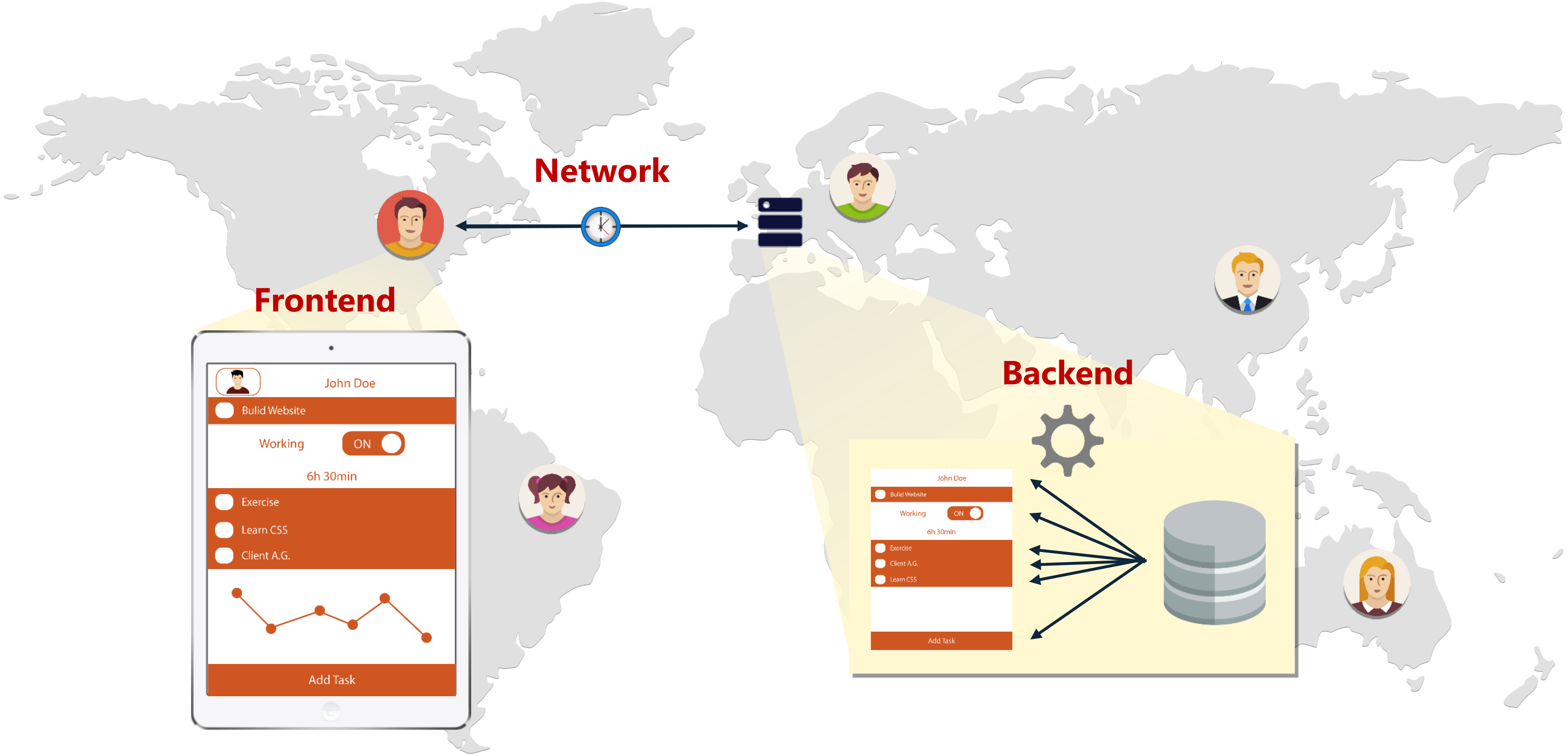

If performance is so important for

**Business Success**

...what causes slow page load times?

# State of the Art

## The tree bottlenecks

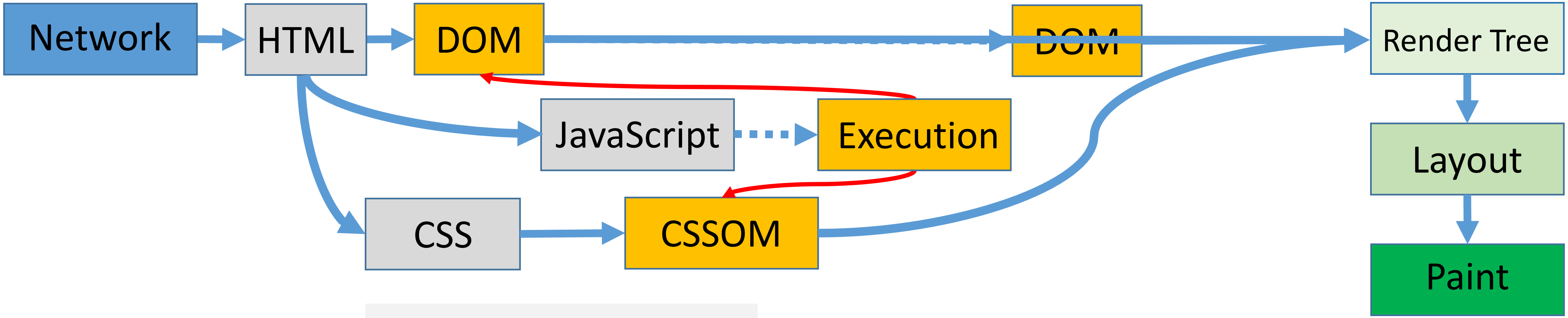


# Frontend Performance

## The critical rendering path

```html
<!doctype html>
<title>Code Talks</title>
<link href=all.css rel=stylesheet />
<script src=app.css ></script>
<div>
  <h1>Web Performance</h1>
</div>
```

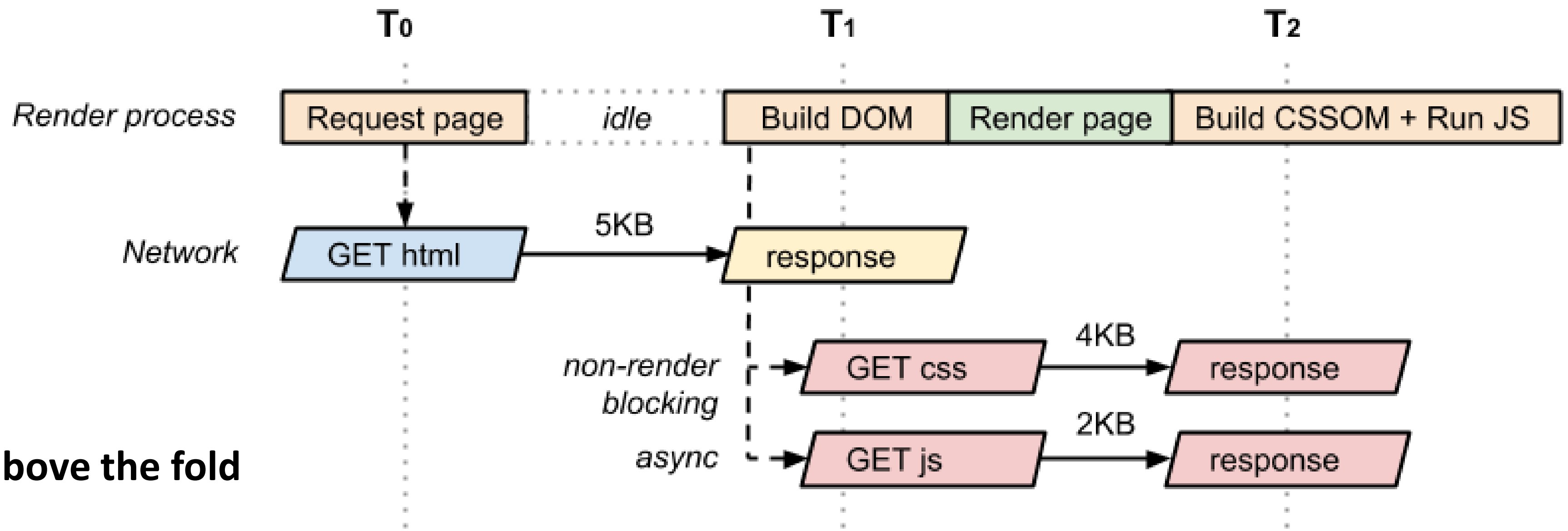
```html
<script>

  elem.style.width = "50px";
  document.write("JS is awesome!");

</script>
```




```css
div   { color: green; }
h1    { padding: 10px; }
```

# Frontend Performance

## Render and parser blocking




What to do:

- **Inline** critical CSS and JS **above the fold**
- CSS at the **top**, JS at the **bottom**
- Load non-critical CSS and JS **asynchronously**
- Rendering the page **progressively**
- **Minify, concatenate, compress** and **cache** CSS, JS and images

Google Developers, Web Fundamentals
https://developers.google.com/web/fundamentals/performance/critical-rendering-path/analyzing-crp

# Frontend Performance

## Tools to improve your page load

### Profiling

GTmetrix

WEBPAGETEST

PageSpeed Insights

### Inlining & Optimization

Critical

PostCSS

processhtml

### Minification & Compression

Google Closure

tiny png

UglifyJs & cssmin

# Frontend Performance

## Applied in the example

handlebars

Critical

PostCSS

processhtml

tiny png

Caching

THINKS Towell+ Sporthandtuch – TV: DHDL | STRYVE

| Request | Status | Domain | Size | Time |
|---|---|---|---|---|
| GET www.thinks.com | 200 OK | thinks.com | 7.1 KB | 26ms |
| GET 01.png | 200 OK | thinks.com | 86.4 KB | 20ms |
| GET founders.png | 200 OK | thinks.com | 93.3 KB | 31ms |
| GET jquery.min.js | 200 OK | ajax.googleap | 28.6 KB | 145ms |
| GET app.js | 200 OK | thinks.com | 86.6 KB | 33ms |
| GET app.css | 200 OK | thinks.com | 25.9 KB | 24ms |
| GET sprite.png | 200 OK | thinks.com | 34.3 KB | 30ms |
| GET bg.jpg | 200 OK | thinks.com | 103.6 K | 48ms |
| GET bg.jpg | 200 OK | thinks.com | 31.3 KB | 29ms |
| GET zhcz-_WihjSQC0oHJ9T | 200 OK | fonts.gstatic.c | 13.2 KB | 276ms |
| GET IQHow_FEYlDC4Gzy_m | 200 OK | fonts.gstatic.c | 13.2 KB | 281ms |
| GET connect | 200 OK | thinks.com | 3.6 KB | 166ms |
| GET analytics.js | 200 OK | google-analyti | 11.3 KB | 199ms |
| GET logo-stryve-inv.svg | 200 OK | thinks.com | 939 B | 7ms |
| GET logo-thinks-inv.svg | 200 OK | thinks.com | 1.2 KB | 8ms |
| GET 10.png | 200 OK | thinks.com | 21.4 KB | 7ms |
| GET alert_message | 200 OK | thinks.com | 281 B | 7ms |
| GET grau_anthrazit | 200 OK | thinks.com | 511 B | 7ms |
| GET merino_taupe | 200 OK | thinks.com | 514 B | 6ms |
| GET graphit_anthrazit | 200 OK | thinks.com | 534 B | 8ms |
| GET platinum_black | 200 OK | thinks.com | 572 B | 6ms |
| GET neonblue_black | 200 OK | thinks.com | 504 B | 7ms |
| GET neonred_black | 200 OK | thinks.com | 563 B | 6ms |
| GET neongreen_black | 200 OK | thinks.com | 630 B | 14ms |
| 24 Requests | | | 565.9 KB | 693ms (onload: 767ms) |

Load time:

**767 ms**

Size:

**565,9 KB**

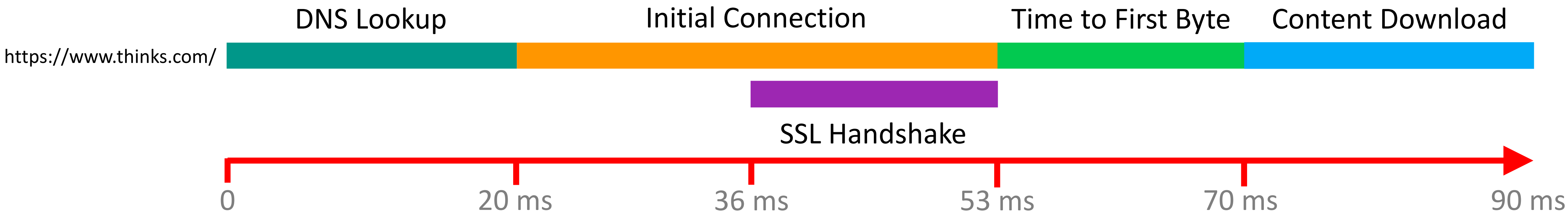
Requests:

**24**

# Network Performance

## Break down of a single resource load

**Maximum** 6 parallel connections




**DNS Lookup**

- Every domain has its own DNS lookup

**Initial connection**

- TCP makes a three way handshake → 2 roundtrips (1 with the new TCP Fast Open)
- SSL connections have a more complex handshake → +2 roundtrips (only 1 with TLS False Start or Session Resumption)

**Time to First Byte**

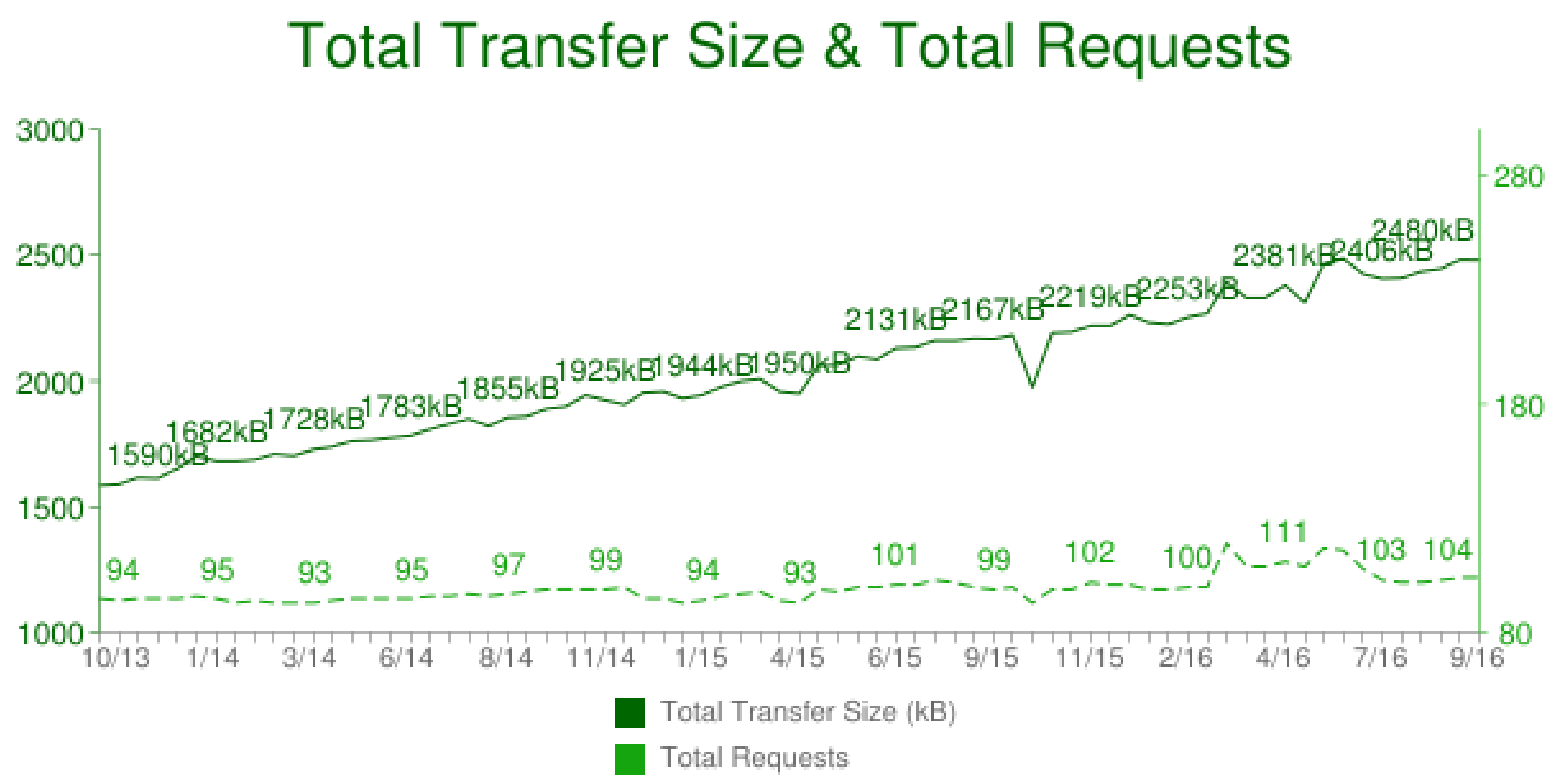
- Depends heavily on the distance between client and the backend
- Includes the time the backend needs to render the page
  - ➢ Session lookups, Database Queries, Template rendering ...

**Content Download**

- Files have a high transfer time on new connections, since the initial congestion window is small → many roundtrips

# Network Performance

## The average website



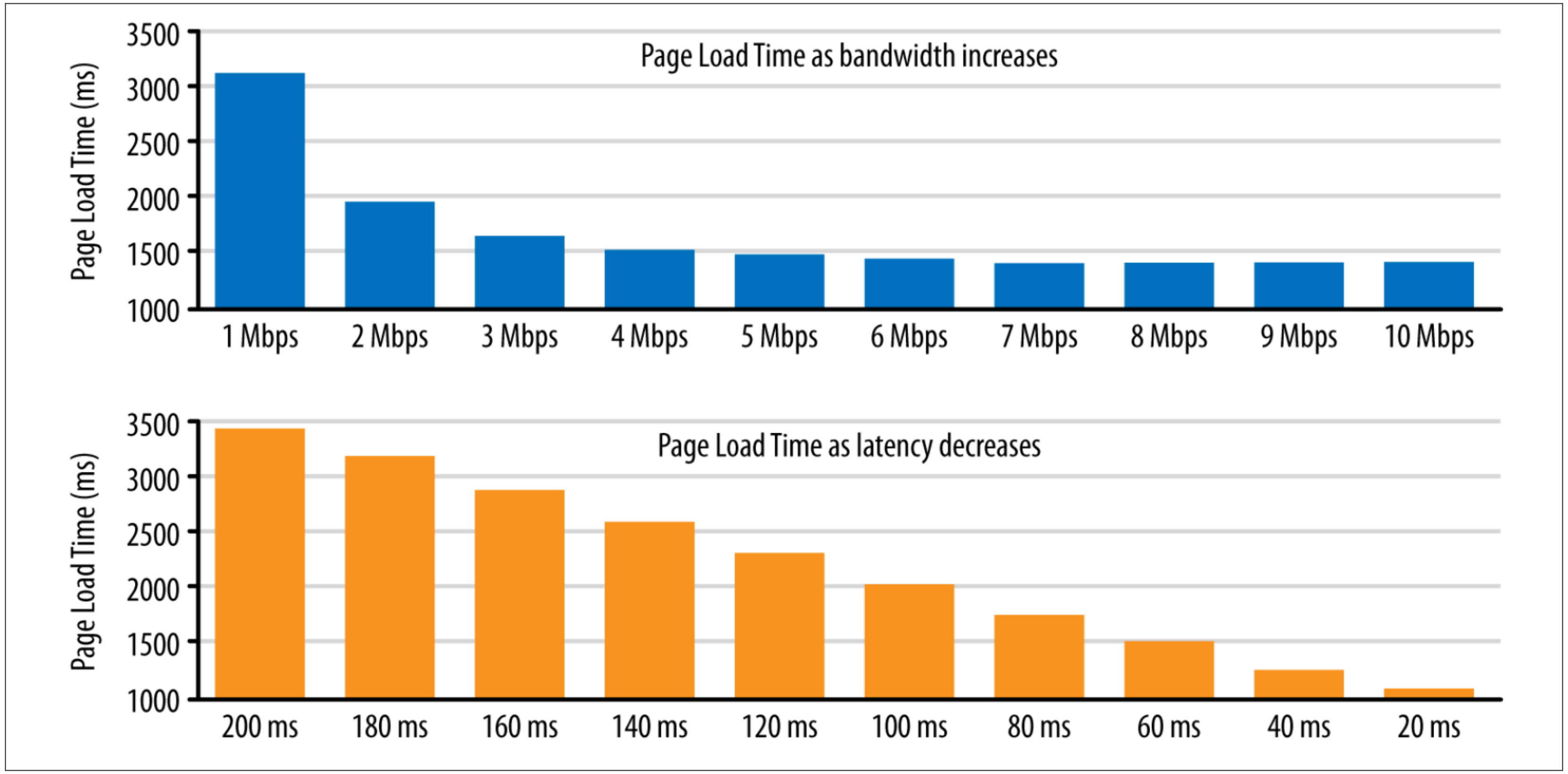

Transfer Size: **2480 KB**

Total Requests: **104**

http://httparchive.org/

# Network Performance

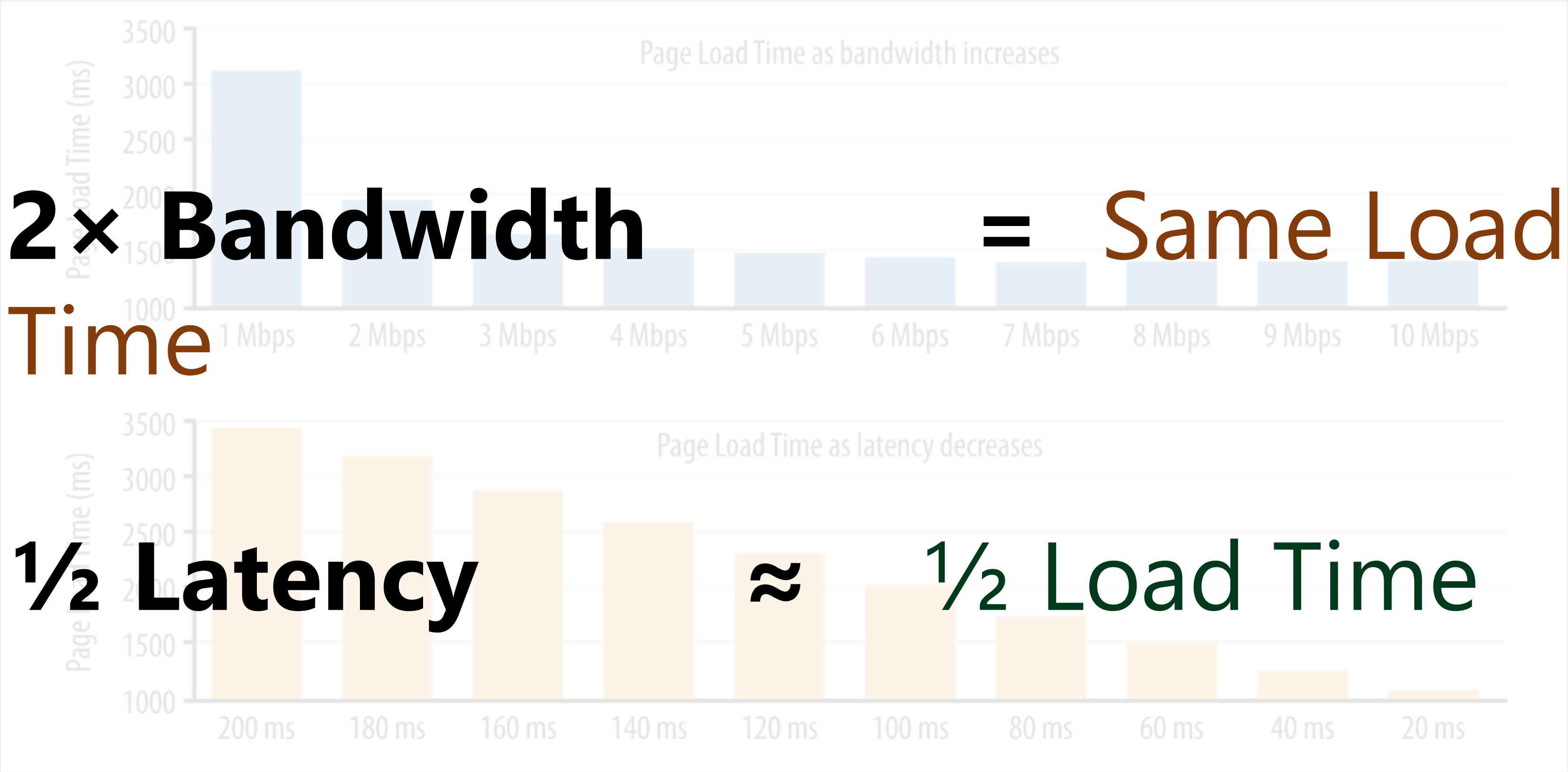
## Network latency impact




I. Grigorik, High performance browser networking. O'Reilly Media, 2013.

# Network Performance

## Network latency impact




**2× Bandwidth** = Same Load Time

**½ Latency** ≈ ½ Load Time

I. Grigorik, High performance browser networking. O'Reilly Media, 2013.

# Network Performance
## Common Tuning Knobs

- **Persistent** connections, if possible **HTTP/2**
- Avoid **redirects**
- Explicit **caching headers** (no heuristic caching)
- **C**ontent **D**elivery **N**etworks
  - To reduce the distance between client and server
  - To cache images, CSS, JS
  - To terminate SSL early and optimized
- **S**ingle **P**age **A**pps:
  - Small initial page that loads additional parts asynchronously
  - Cacheable HTML templates + load dynamic data
  - Only update sections of the page during navigation

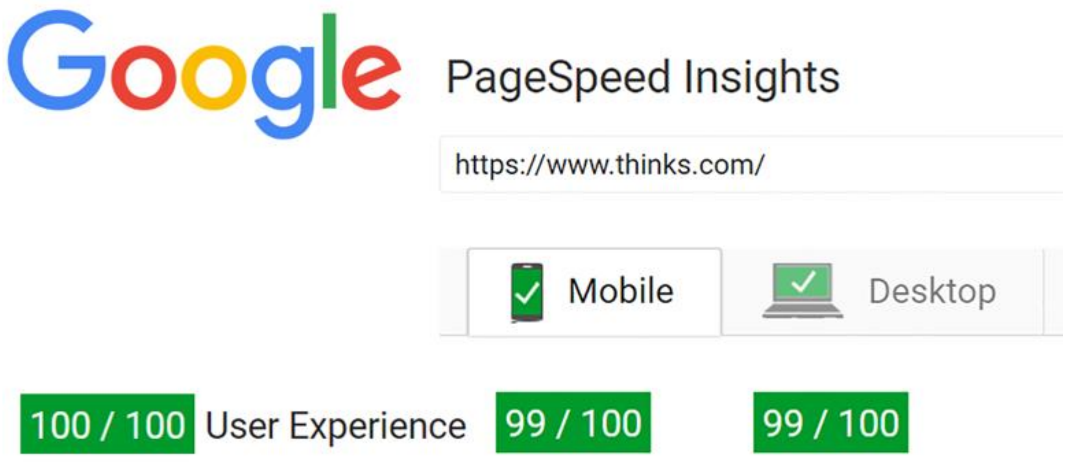
HTTP/2 Performance Advantages:

- Server **Push**
- **Header Compression**
- Request **Pipelining**
- **Multiplexing** over 1 TCP connection (no head-of-line blocking)

# Network Performance
## Applied in the example

What we do:

- Heavy browser and CDN Caching
- Avoid Redirects (+ serve from CDN)
- Single Page App functionality
- Persistent Backend Connections
- Gzip Compression
- IP Anycasting to nearest POP

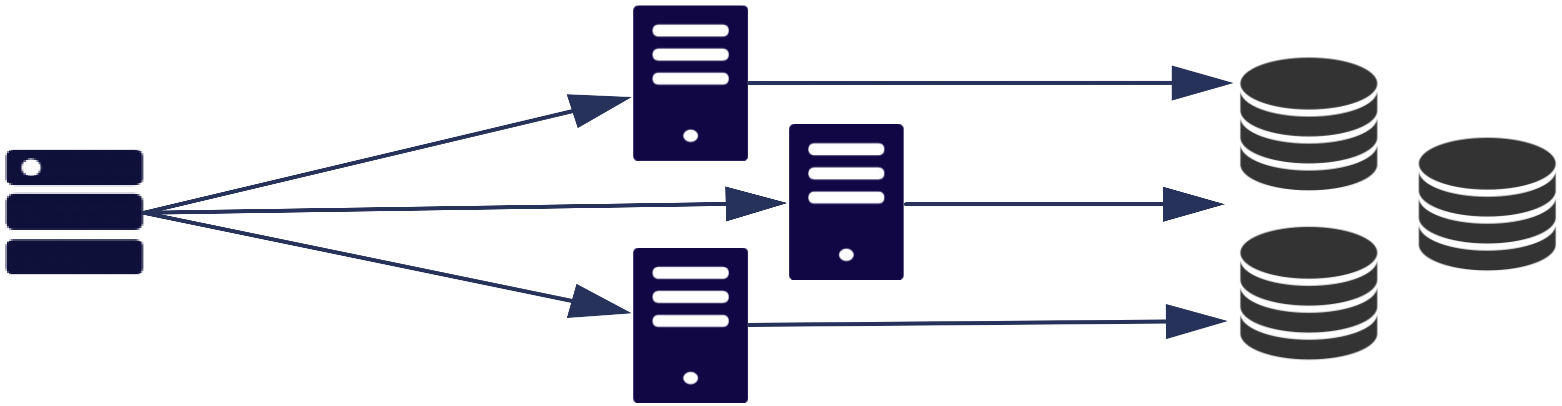



| Name | Size | Time |
|---|---|---|
| www.thinks.com | 332 B | 25 ms |
| 01.png | (from memory cache) | 0 ms |
| founders.png | (from memory cache) | 0 ms |
| jquery.min.js | (from disk cache) | 10 ms |
| app.js | (from disk cache) | 17 ms |
| app.css | (from disk cache) | 8 ms |
| sprite.png | (from memory cache) | 0 ms |
| bg.jpg | (from memory cache) | 0 ms |
| zhcz-_WihjSQC0oHJ9TCYC3USBnSvpkopQaUR-... | (from disk cache) | 6 ms |
| bg.jpg | (from memory cache) | 0 ms |
| IQHow_FEYIDC4Gzy_m8fcvEr6Hm6RMS0v1dtXs... | (from disk cache) | 3 ms |
| app.css | (from disk cache) | 6 ms |
| connect | (from disk cache) | 3 ms |
| analytics.js | (from disk cache) | 3 ms |
| logo-thinks-inv.svg | (from disk cache) | 5 ms |
| logo-stryve-inv.svg | (from disk cache) | 8 ms |
| collect?v=1&_v=j46&aip=1&a=2145881643&t=pa... | 72 B | 28 ms |
| collect?v=1&_v=j46&aip=1&a=2145881643&t=ti... | 66 B | 26 ms |
| alert_message | (from disk cache) | 3 ms |
| grau_anthrazit | (from disk cache) | 4 ms |
| merino_taupe | (from disk cache) | 6 ms |
| graphit_anthrazit | (from disk cache) | 10 ms |
| platinum_black | (from disk cache) | 10 ms |
| neonblue_black | (from disk cache) | 9 ms |
| neonred_black | (from disk cache) | 9 ms |
| neongreen_black | (from disk cache) | 9 ms |
| 10.png | (from memory cache) | 1 ms |
| large_1.png | (from memory cache) | 0 ms |
| large_1.png | (from memory cache) | 0 ms |
| large_1.png | (from memory cache) | 0 ms |
| large_1.png | (from memory cache) | 0 ms |
| large_1.png | (from memory cache) | 0 ms |
| large_1.png | (from memory cache) | 0 ms |
| large_1.png | (from memory cache) | 0 ms |
| 04.png | (from memory cache) | 0 ms |
| 09.png | (from memory cache) | 0 ms |
| 03.png | (from memory cache) | 0 ms |
| 05.png | (from memory cache) | 0 ms |
| 07.png | (from memory cache) | 0 ms |

# Backend Performance

## Overview

Load Balancer | Application Server | Database



- Load Balancing
- Auto-scaling
- Failover

- Stateless session handling
- Minimize shared state
- Efficient Code & IO

- Horizontally scalable databases (e.g. “NoSQL”)
  - Replication
  - Sharding
  - Failover

# Backend Performance

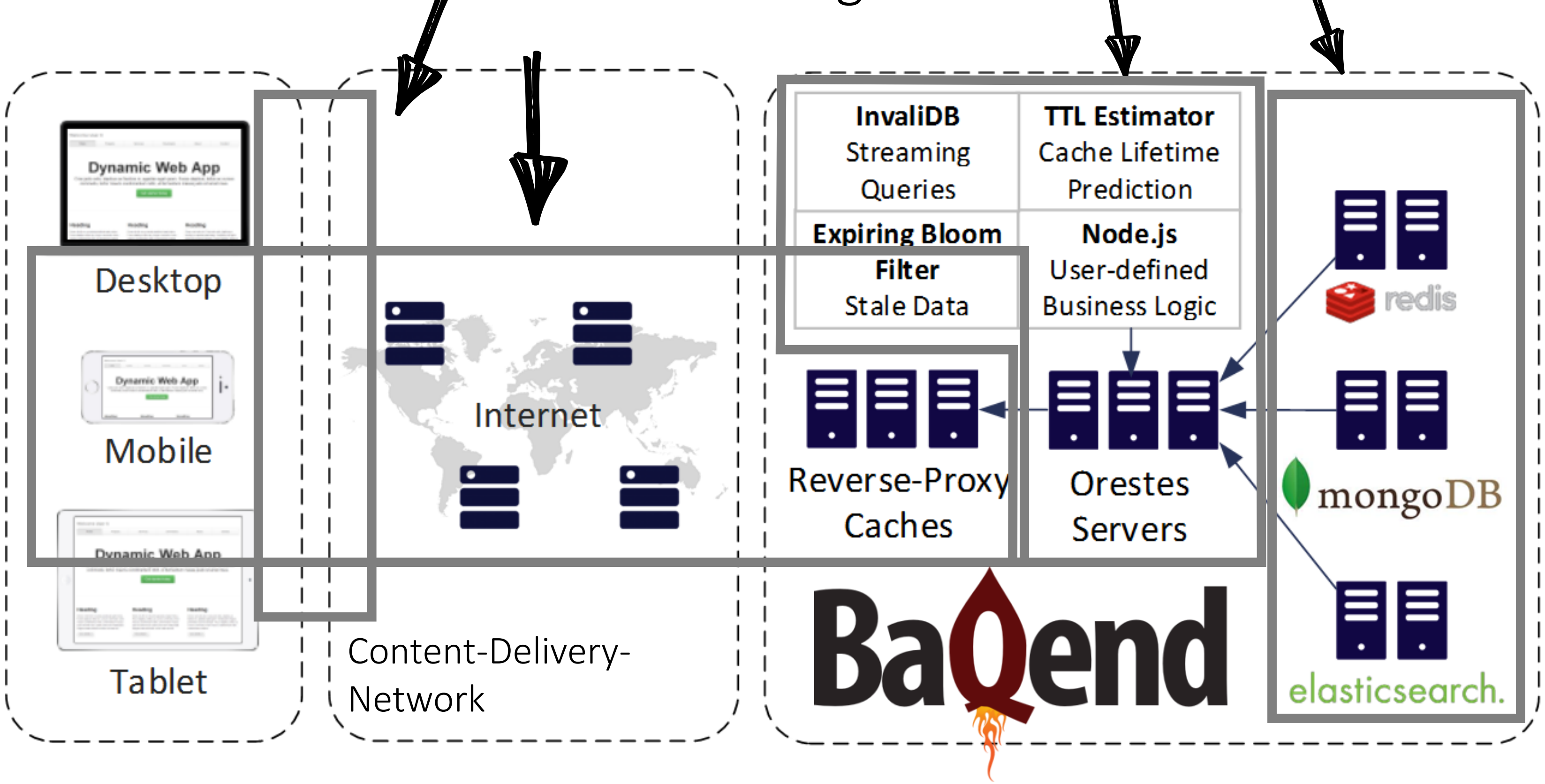
## Applied in the example

Backend-as-a-Service Middleware:
Caching, Transactions, Schemas,
Invalidation Detection, …

Polyglot Storage

Unified REST API

Standard HTTP Caching



# Baqend Architecture

## Infrastructure

# Performance: State of the Art

## Summarized

### Frontend

- Doable with the right set of best practices
- Good support through build tools

### Latency

- Caching and CDNs help, but a considerable effort and only for static content

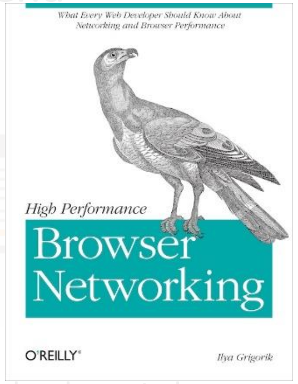
### Backend

- Many frameworks and platforms
- Horizontal scalability is very difficult

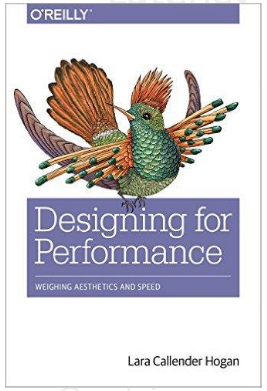
# Performance: State of the Art

## Summarized

### Good Resources:




chimera.labs.oreilly.com/books/1230000000545



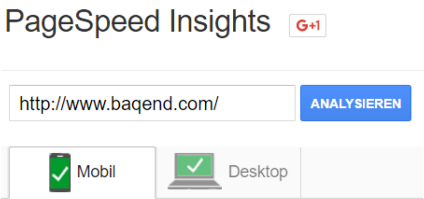

shop.oreilly.com/product/0636920033578.do

Google Developers Performance

https://developers.google.com/web/fundamentals/performance/?hl=en


https://www.udacity.com/course/website-performance-optimization--ud884

### Good Tools:




https://developers.google.com/speed/pagespeed/

GTmetrix

https://gtmetrix.com

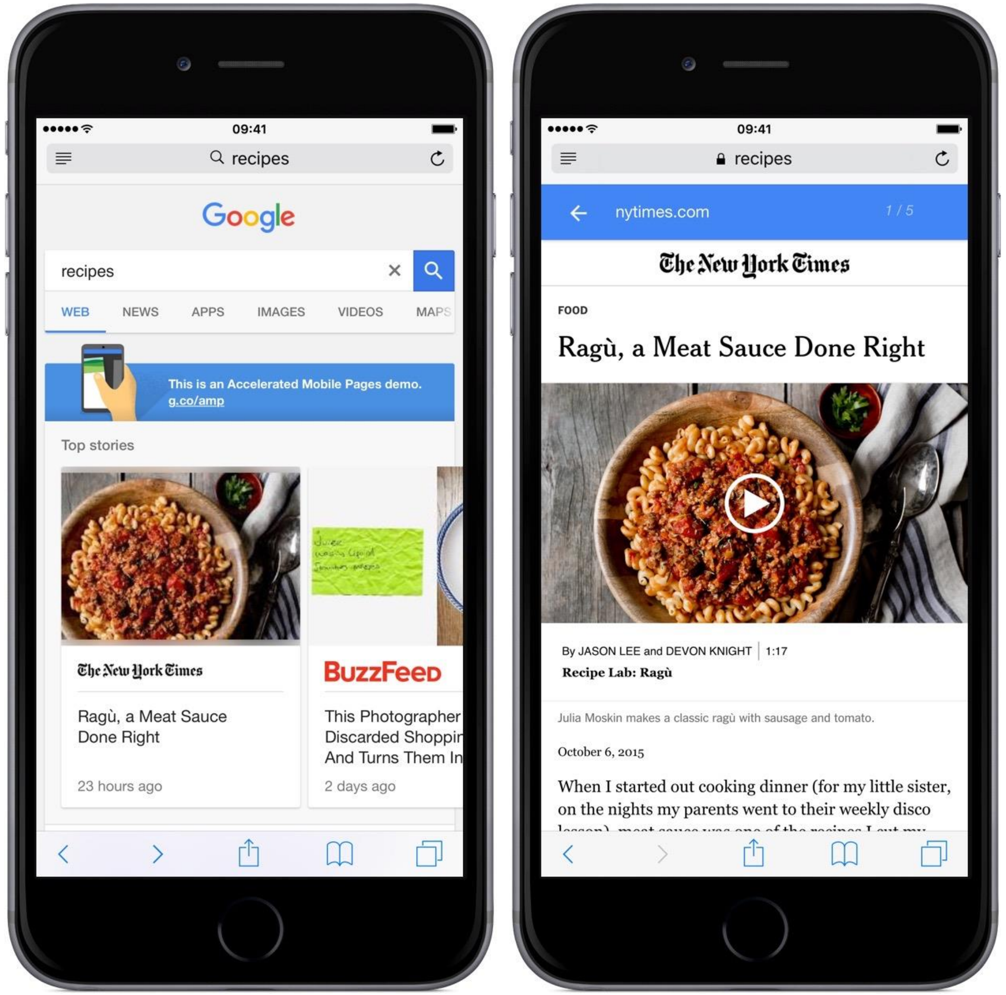
WEBPAGETEST

http://www.webpagetest.org/

https://www.ampproject.org
/docs/reference/spec.html

# Accelerated Mobile Pages

## Google's Approach for a Faster Web



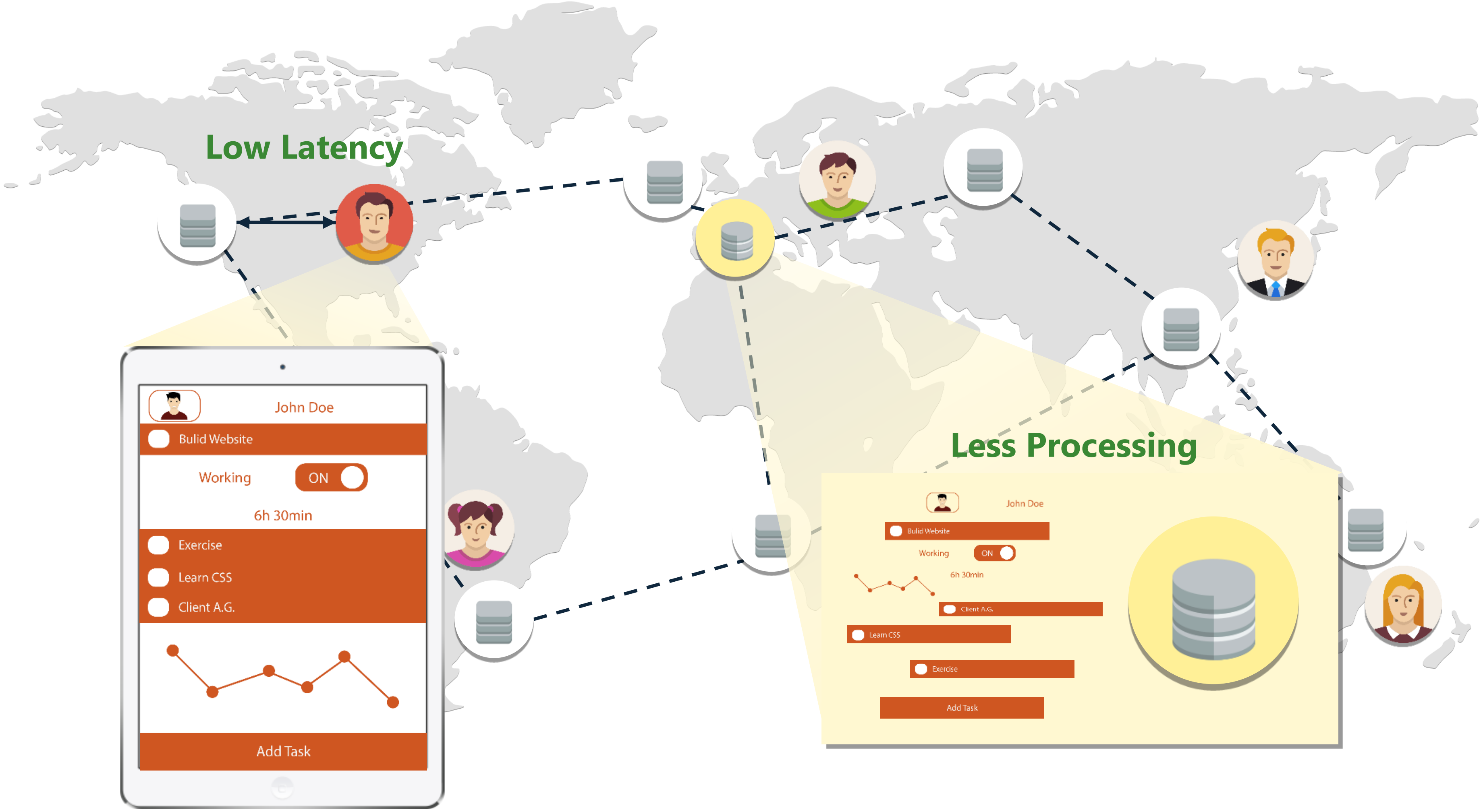

How AMP works:

- Stripped down HTML + AMP tags
  → rendered asynchronously by AMP runtime
- All CSS must be inlined
- All JS must be async
- Static sizes (e.g. iframes) → no repaints
- Cached in Google CDN, as long as it is crawled the next time (no invalidation)
  → only suited for static media, e.g. news

How to apply the same techniques for *any* website?

# Goal: Low-Latency for Dynamic Content

## By Serving Data from Ubiquitous Web Caches



# Innovation

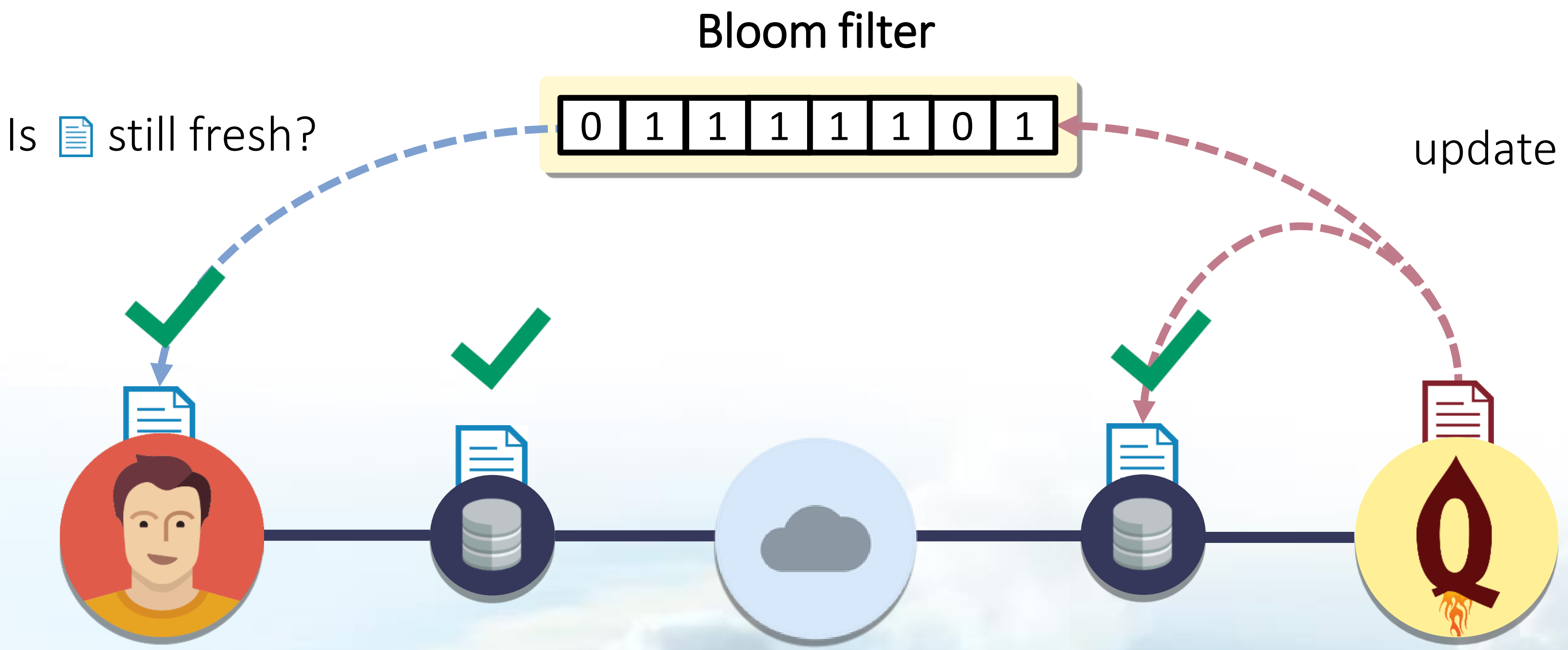
## Problem:

# Innovation

## Solution: Baqend proactively revalidates data

**6 Years**
Research & Development

**New Algorithms**
Solve Consistency Problem



# Innovation

## Solution: Baqend proactively revalidates data

6 Years

New Algorithms


- F. Gessert, F. Bücklers, und N. Ritter, „ORESTES: a Scalable Database-as-a-Service Architecture for Low Latency“, in *CloudDB 2014*, 2014.
- F. Gessert und F. Bücklers, „ORESTES: ein System für horizontal skalierbaren Zugriff auf Cloud-Datenbanken“, in Informatiktage 2013, 2013.
- F. Gessert und F. Bücklers, *Performanz- und Reaktivitätssteigerung von OODBMS vermittels der Web-Caching-Hierarchie*. Bachelorarbeit, 2010.
- M. Schaarschmidt, F. Gessert, und N. Ritter, „Towards Automated Polyglot Persistence“, in BTW 2015.
- S. Friedrich, W. Wingerath, F. Gessert, und N. Ritter, „NoSQL OLTP Benchmarking: A Survey“, in *44. Jahrestagung der Gesellschaft für Informatik*, 2014, Bd. 232, S. 693–704.
- W. Wingerath, F. Gessert, S. Friedrich, N. Ritter „Real-time stream processing for Big Data“, *Big Data Analytics it - Information Technology,* 2016
- F. Gessert, W. Wingerath, S. Friedrich, N. Ritter “NoSQL Database Systems: A Survey and Decision Guidance”, *Computer Science - Research and Development,* 2016
- F. Gessert, S. Friedrich, W. Wingerath, M. Schaarschmidt, und N. Ritter, „Towards a Scalable and Unified REST API for Cloud Data Stores“, in *44. Jahrestagung der GI*, Bd. 232, S. 723–734.
- F. Gessert, M. Schaarschmidt, W. Wingerath, S. Friedrich, und N. Ritter, „The Cache Sketch: Revisiting Expiration-based Caching in the Age of Cloud Data Management“, in BTW 2015.
- F. Gessert und F. Bücklers, *Kohärentes Web-Caching von Datenbankobjekten im Cloud Computing*. Masterarbeit 2012.
- W. Wingerath, S. Friedrich, und F. Gessert, „Who Watches the Watchmen? On the Lack of Validation in NoSQL Benchmarking“, in BTW 2015.
- F. Gessert, „Skalierbare NoSQL- und Cloud-Datenbanken in Forschung und Praxis“, BTW 2015
- F. Gessert, N. Ritter „Scalable Data Management: NoSQL Data Stores in Research and Practice“, *32nd IEEE International Conference on Data Engineering, ICDE,* 2016
- F. Gessert, N. Ritter „Polyglot Persistence“, *Datenbank Spektrum*, 2016.

# 1 Faster Page Loads

- Clients load the Cache Sketch at connection
- Every non-stale cached record can be reused without degraded consistency

l)

# 2 Continuous Query Matching

## Generalizing the Cache Sketch to query results

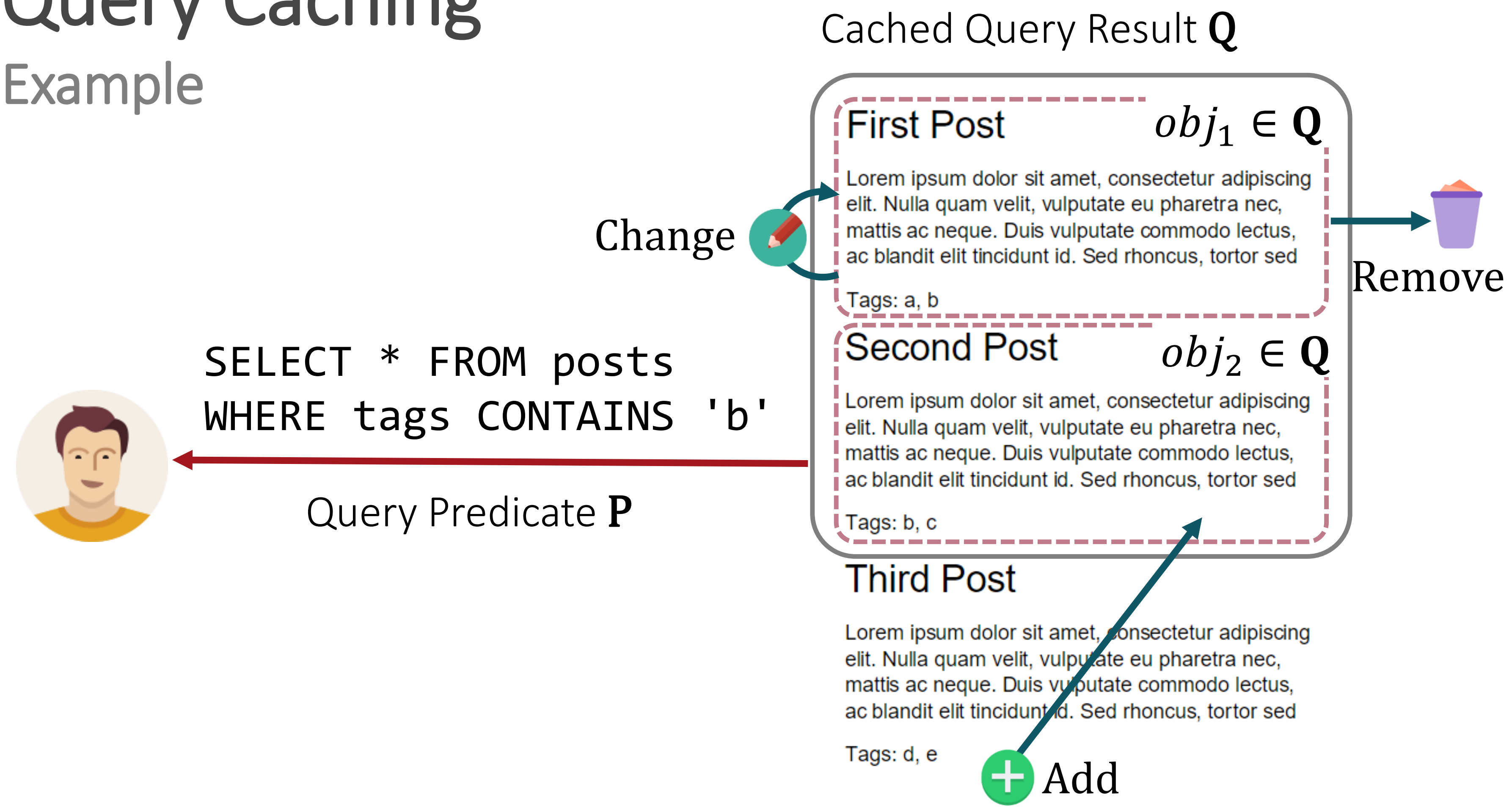
**Main challenge**: when to invalidate?

- **Objects**: for every update and delete
- **Queries**: as soon as the query result changes

How to detect query **result changes in real-time**?

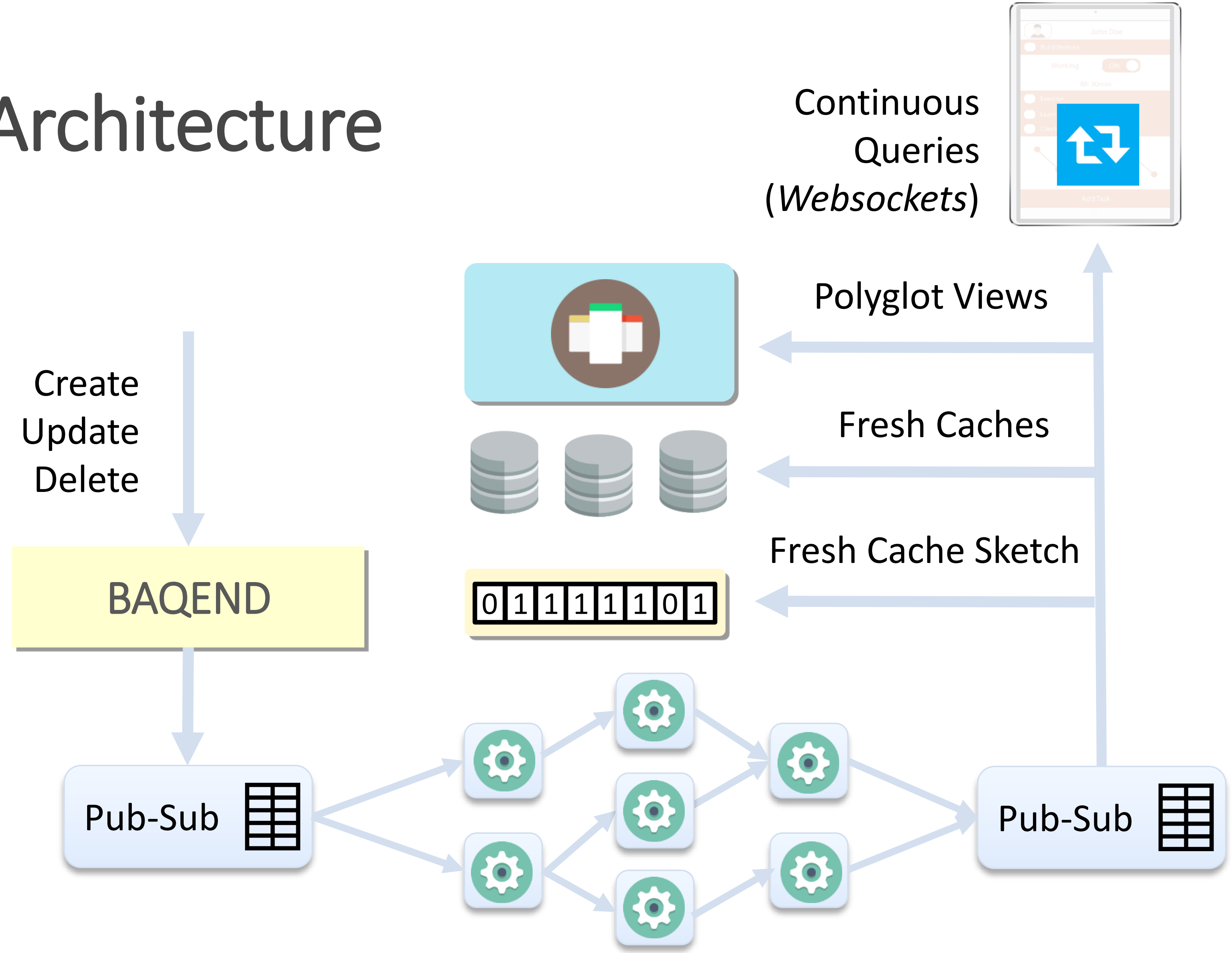
# 2 Query Caching

## Example




- **Add, Change, Remove** all entail an invalidation and addition to the cache sketch

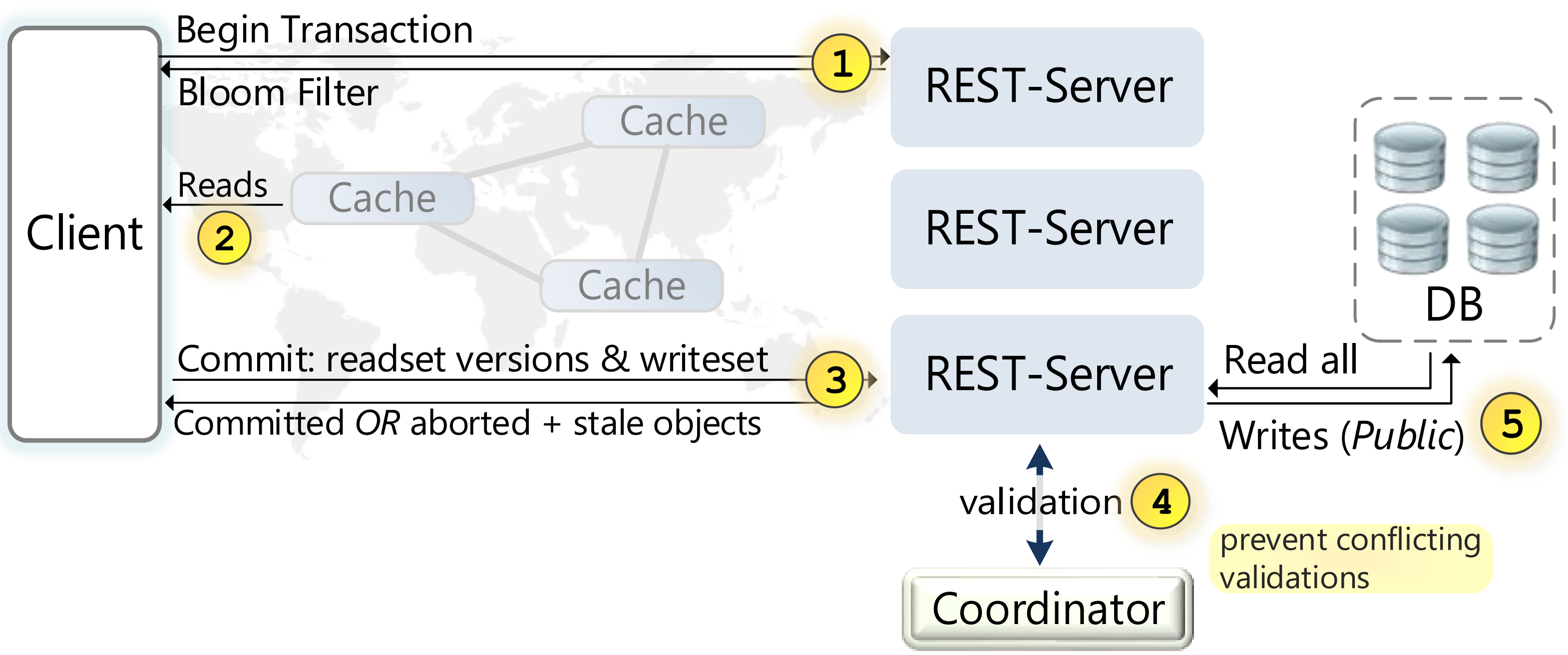
# 2 Architecture




Felix Gessert, Michael Schaarschmidt, Wolfram Wingerath, Steffen Friedrich, Norbert Ritter: *Quaestor: Scalable and Fresh Query Caching on the Web's Infrastructure*. Under Submission.

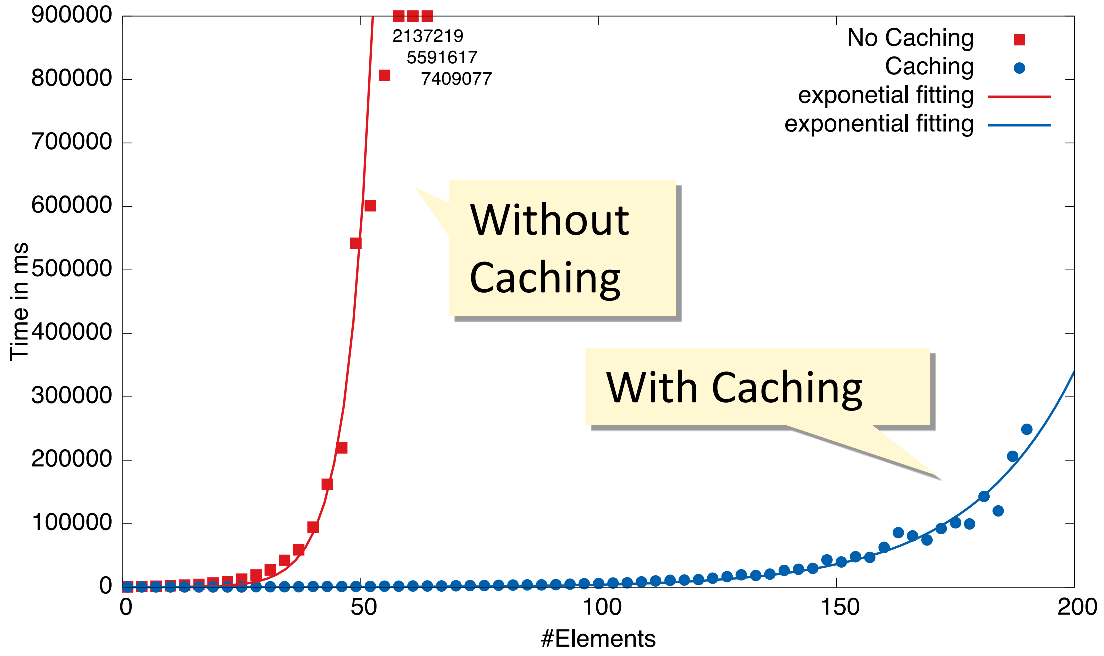
# 3 Scalable ACID Transcations

- Solution: **Conflict-Avoidant Optimistic Transactions**
  - Cache Sketch fetched with transaction begin
  - **Cached reads → Shorter transaction duration → less aborts**



# 3 Scalable ACID Transcations

- **Novelty:** ACID transactions on sharded DBs like MongoDB




- Current Work: **DESY** and **dCache** building a scalable namespace for their file system on this www.baqend.com

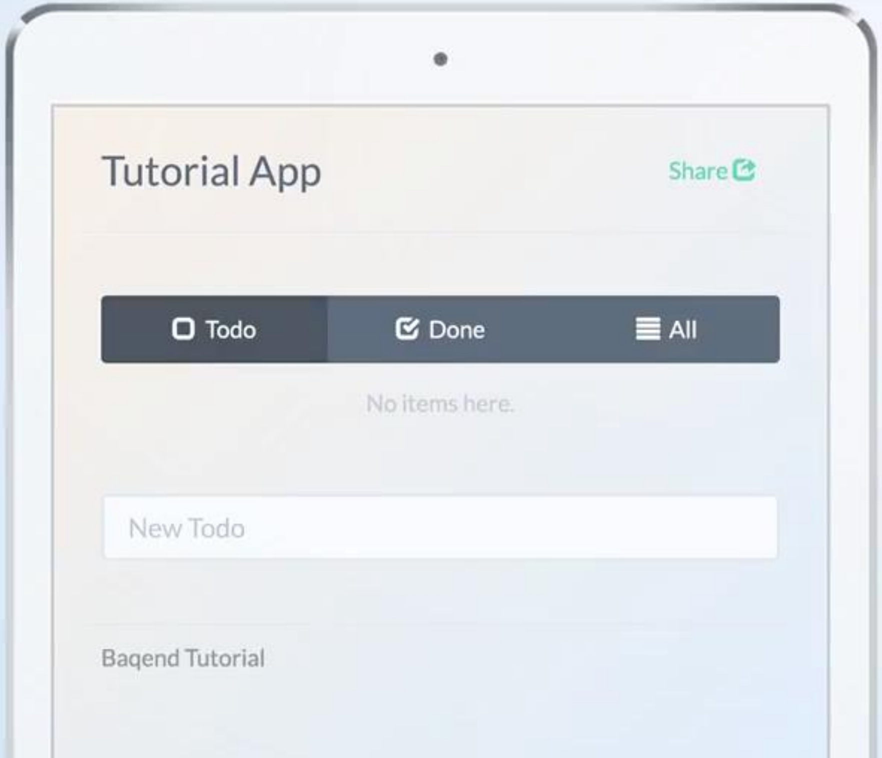
BaQend

Product Developer About us Blog Press

Log in or Sign Up Free

# The World's Fastest Backend

**Build websites and apps that load instantly.**




THE BAQEND PLATFORM

## Sky-rocket your Development

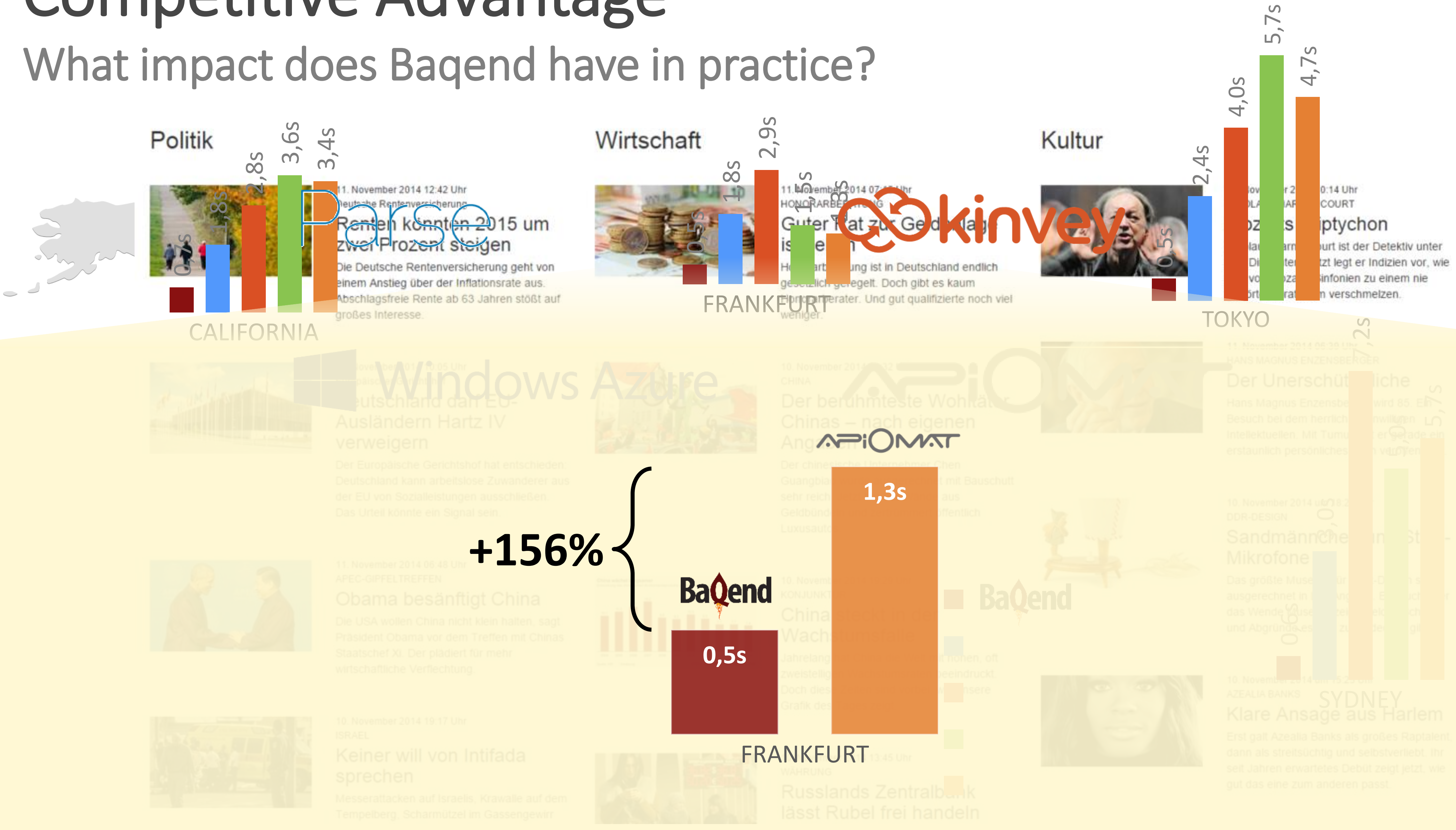
**Start building now. Baqend Cloud is free and easy to get started with.**

TUTORIAL TRY BAQEND


Parse
Baqend
Kinvey
2.6 s
0.3 s
3.9 s

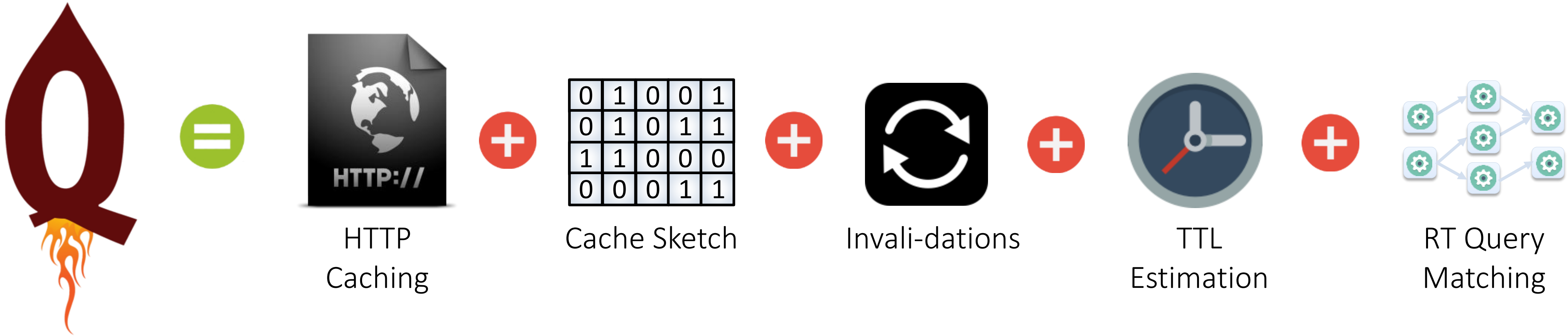
# Competitive Advantage

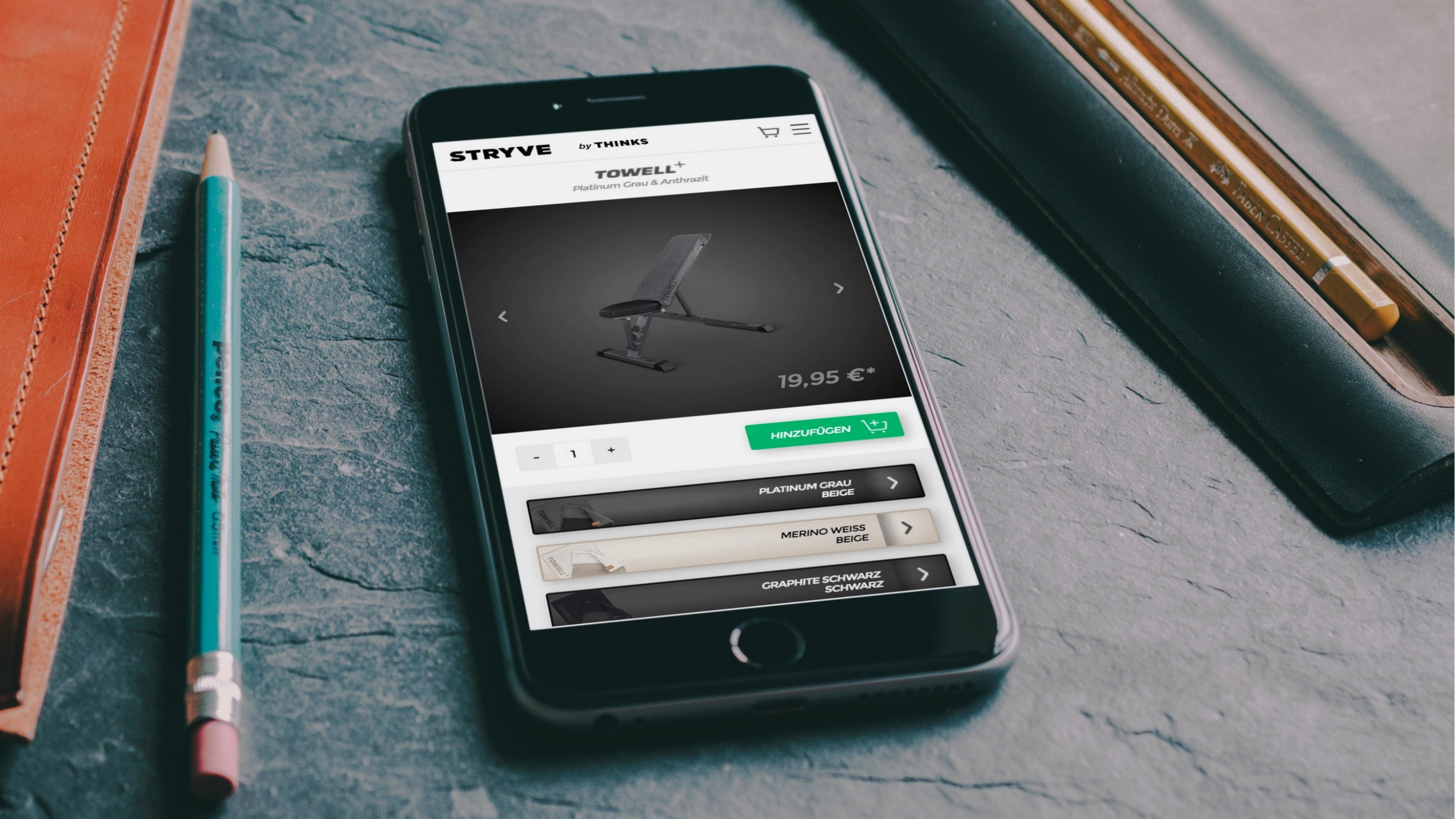
## What impact does Baqend have in practice?



# Summary


Frontend
HTML
CSS
JS


HTTP://
0 1 0 0 1
0 1 0 1 1
1 1 0 0 0
0 0 0 1 1
HTTP Caching
Cache Sketch
Invali-dations
TTL Estimation
RT Query Matching


STRYVE
by THINKS
TOWELL+
Platinum Grau & Anthrazit
19,95 €*
HINZUFÜGEN
- 1 +
PLATINUM GRAU
BEIGE
MERINO WEISS
BEIGE
GRAPHITE SCHWARZ
SCHWARZ

BaQend

STRYVE by THINKS Shop

Aktuell
46557
aktive Nutzer auf der Website

MOBILGERÄT TABLET DESKTOP
77% 15% 9%

< 1 second
Page Load Time

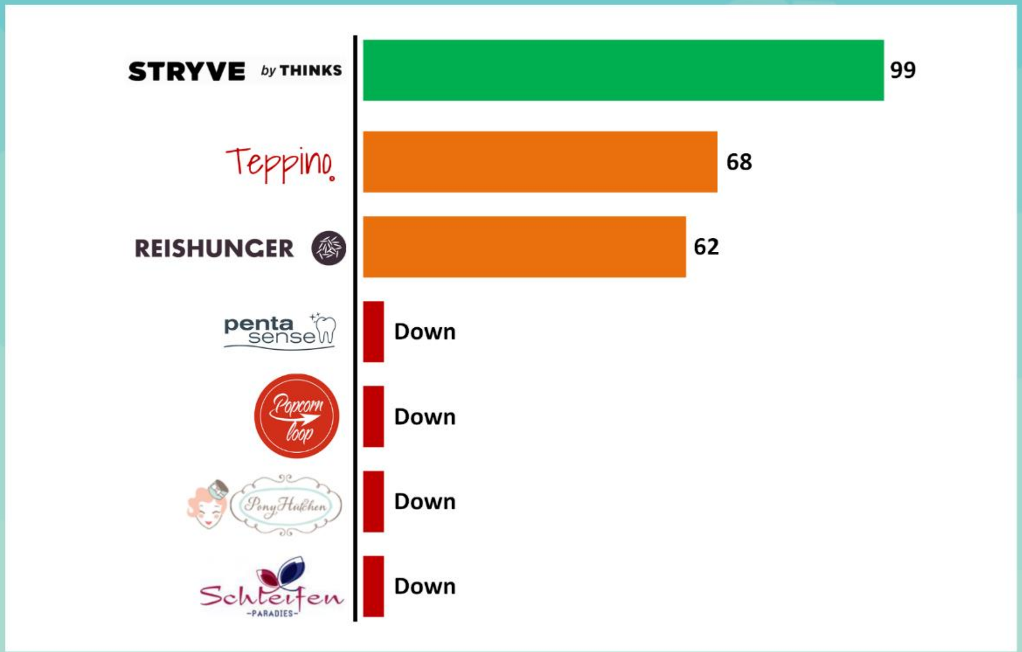
7.8%
Conversion Rate

Simultaneous Users

3%
Server Usage

# Shops in "Die Höhle der Löwen"

The Google Page Speed Scores
for Season 3, 09/06/2016



Our Vision for Baqend:
„A web without load times“

www.baqend.com
@Baqendcom