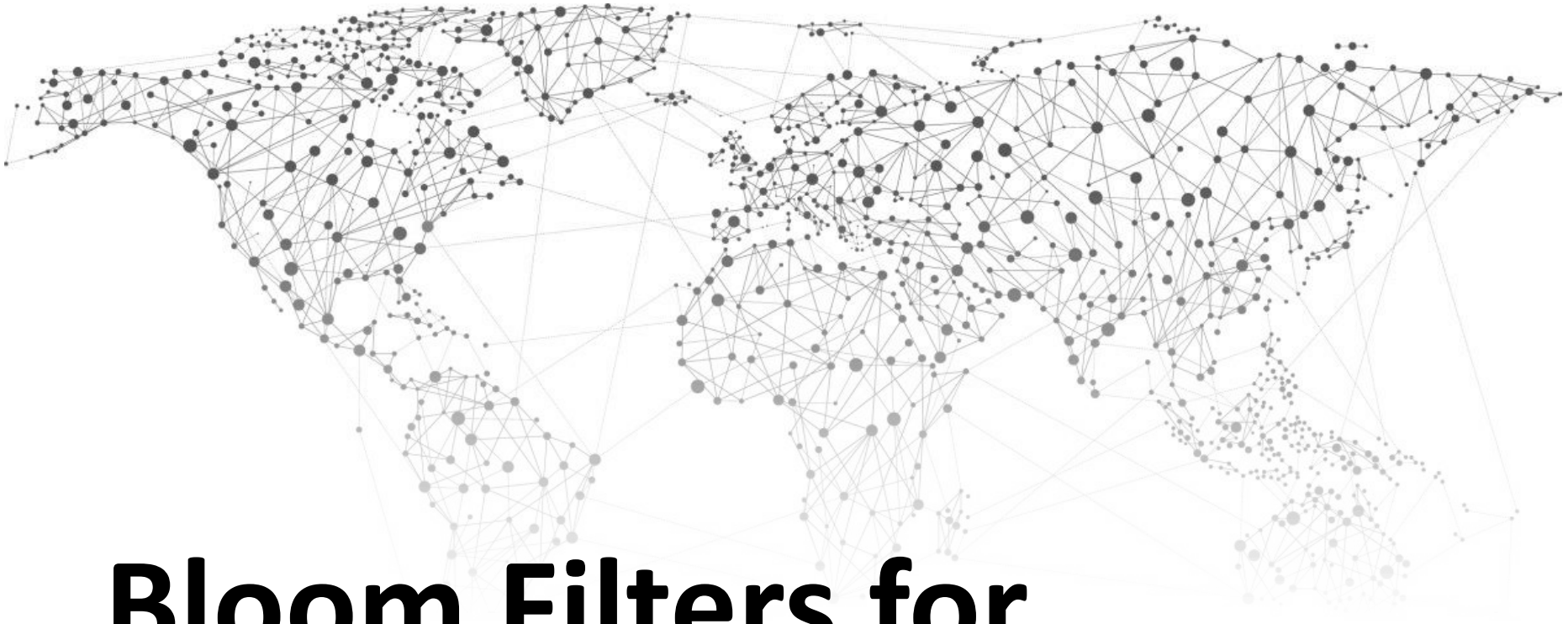


Bloom Filters for Web Caching

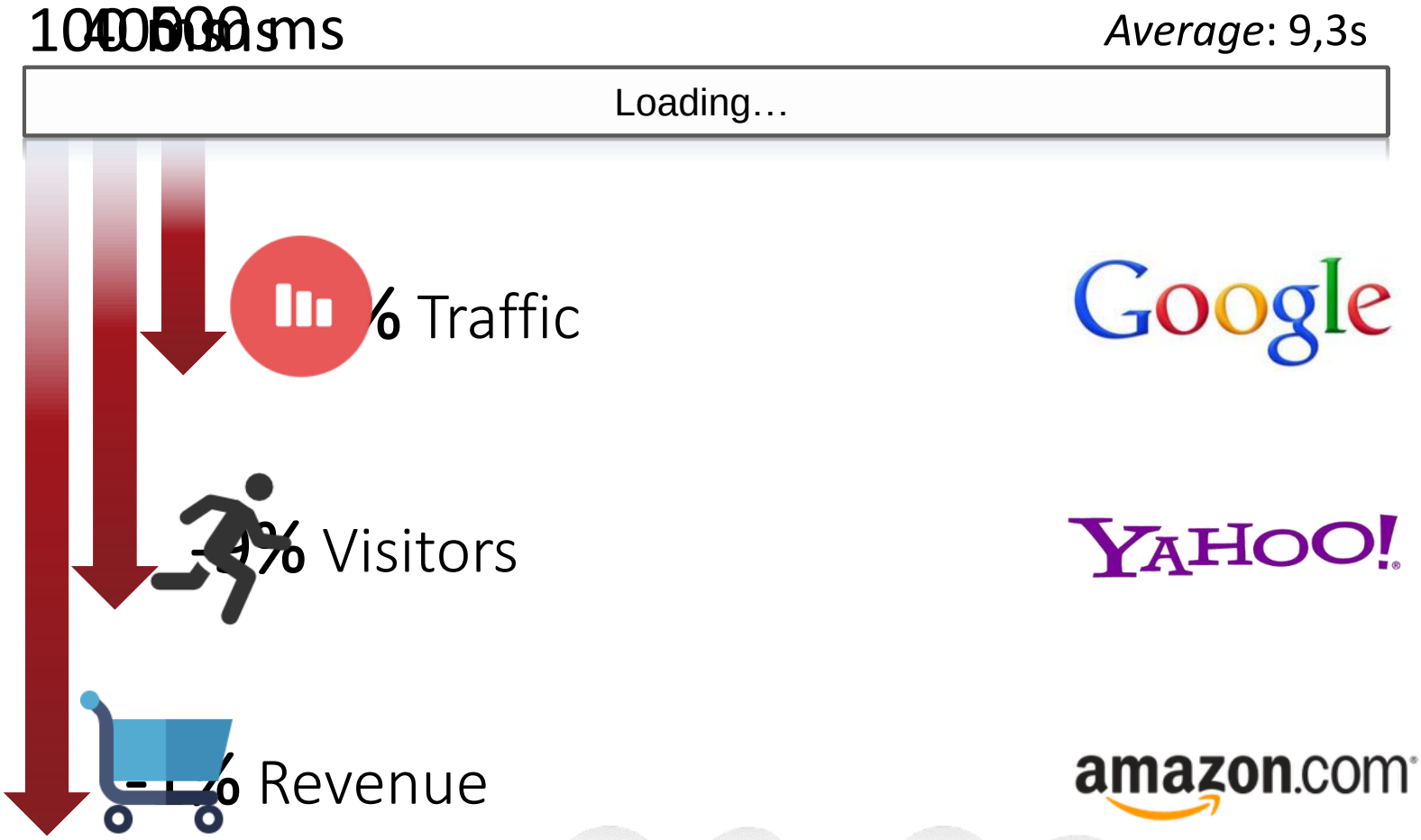
Felix Gessert

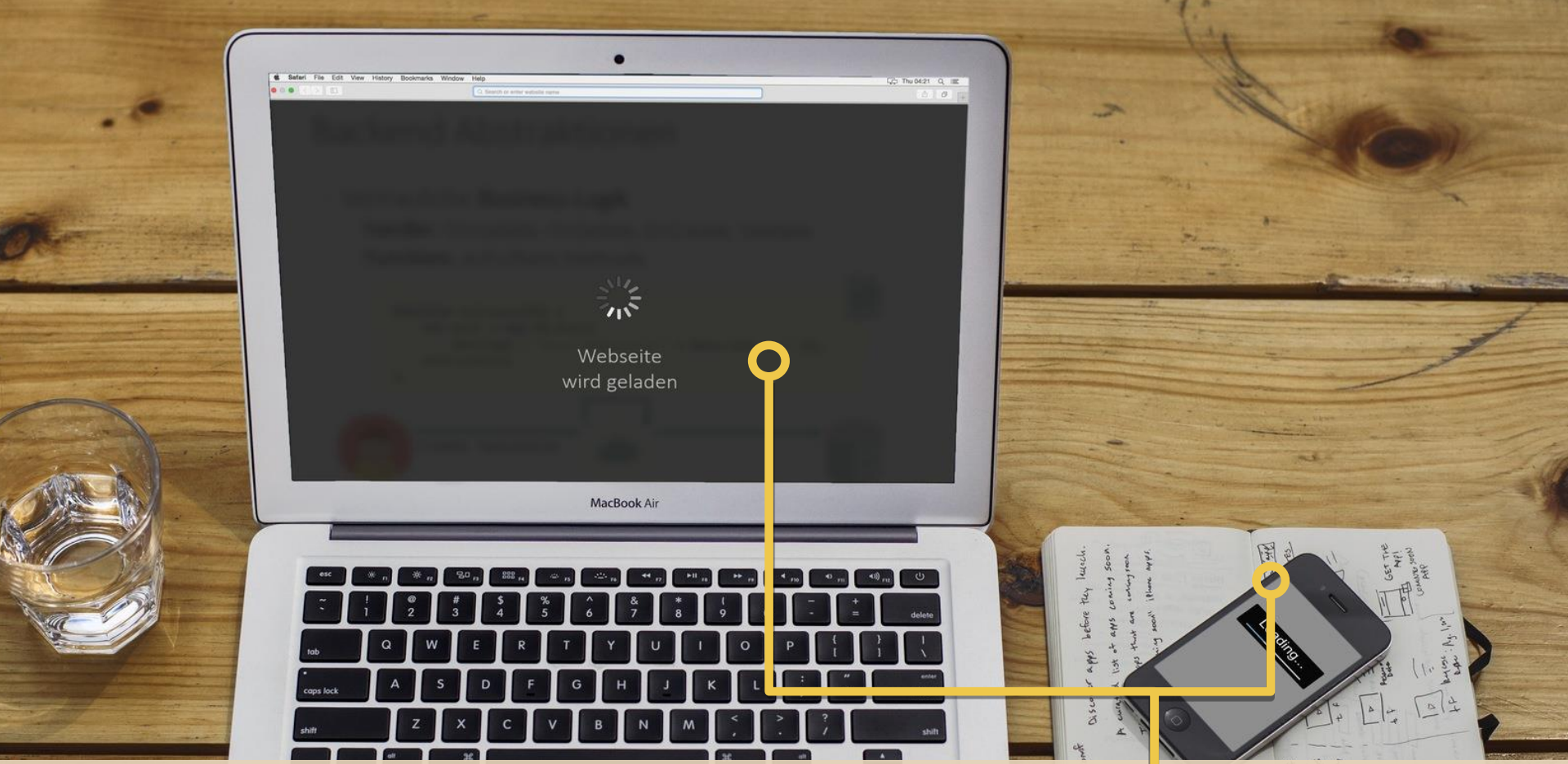
Felix.gessert@baqend.com

@baqendcom

Presentation
is loading

The Latency Problem



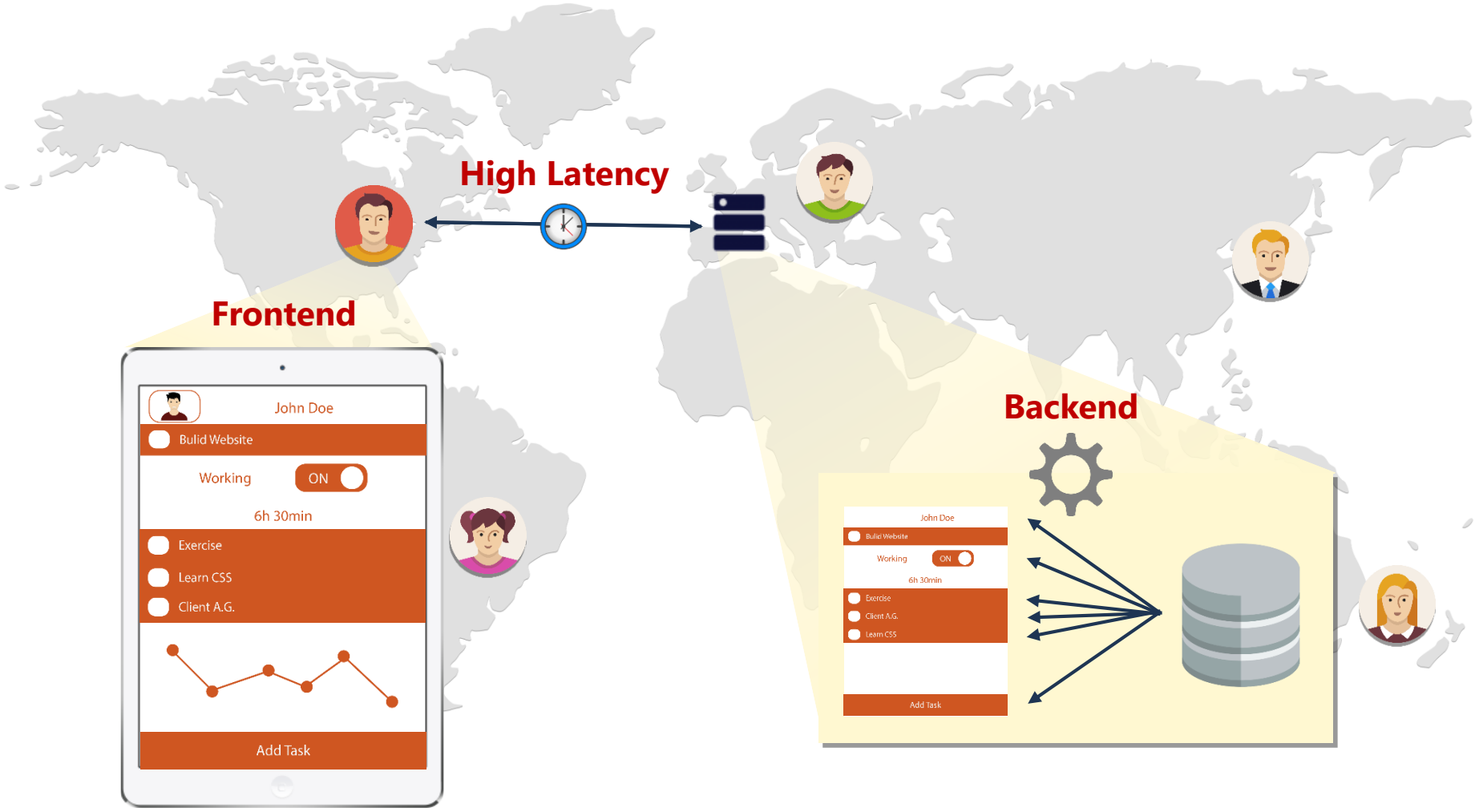


If perceived speed is such an import factor for **business**

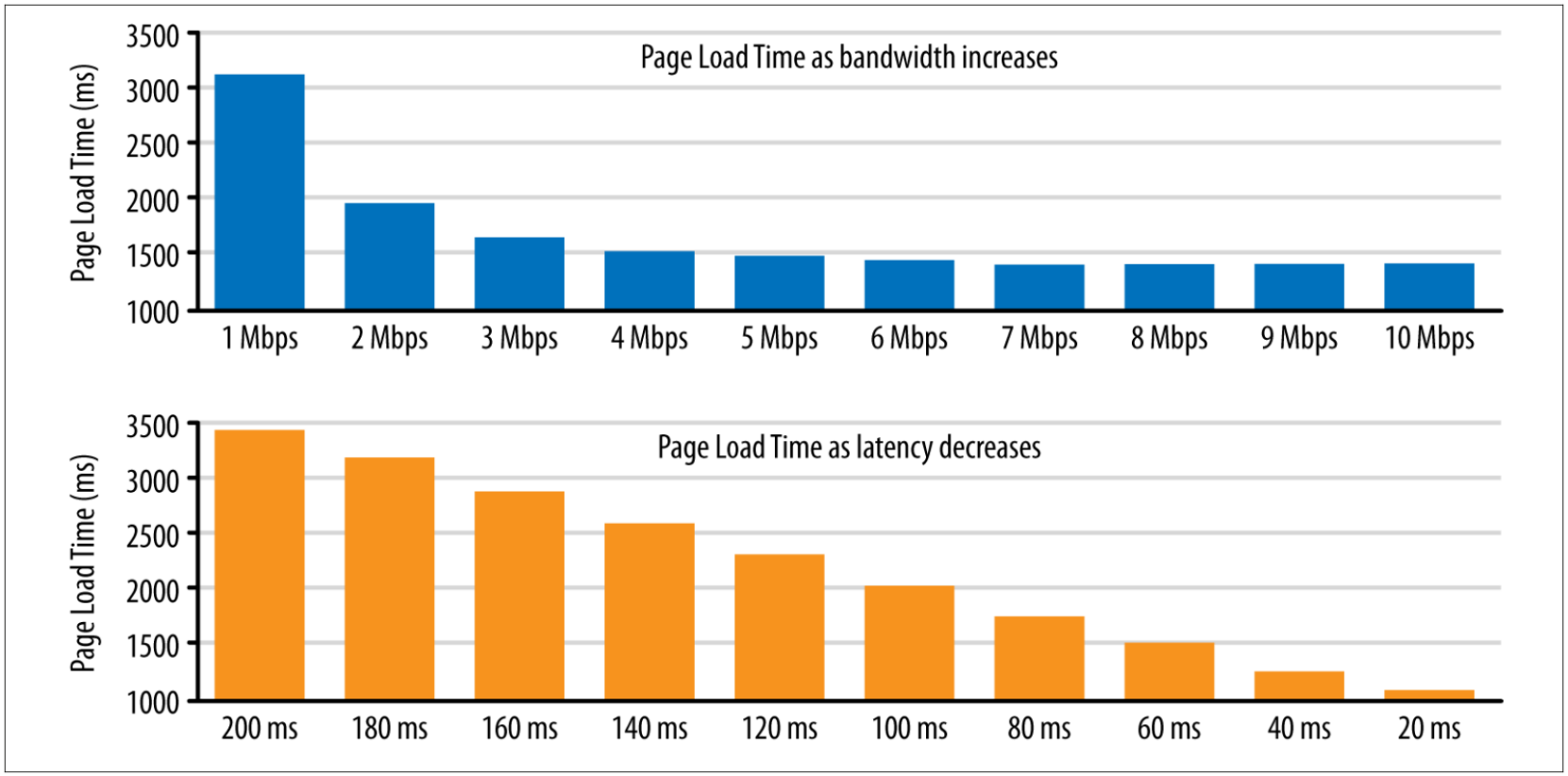
...what causes slow page load times?

The Problem

Three Bottlenecks: Latency, Backend & Frontend

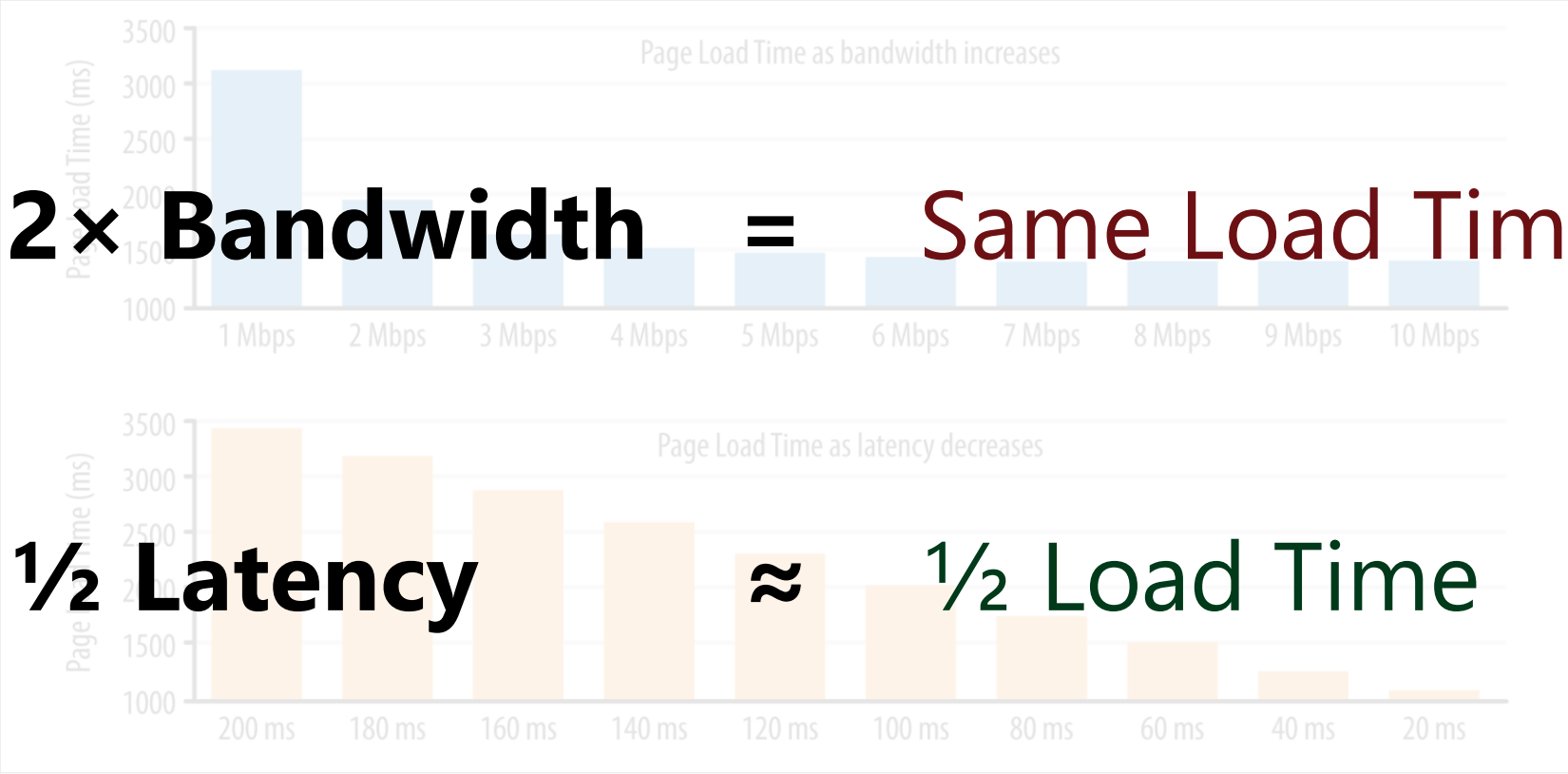


Network Latency: Impact



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

Network Latency: Impact



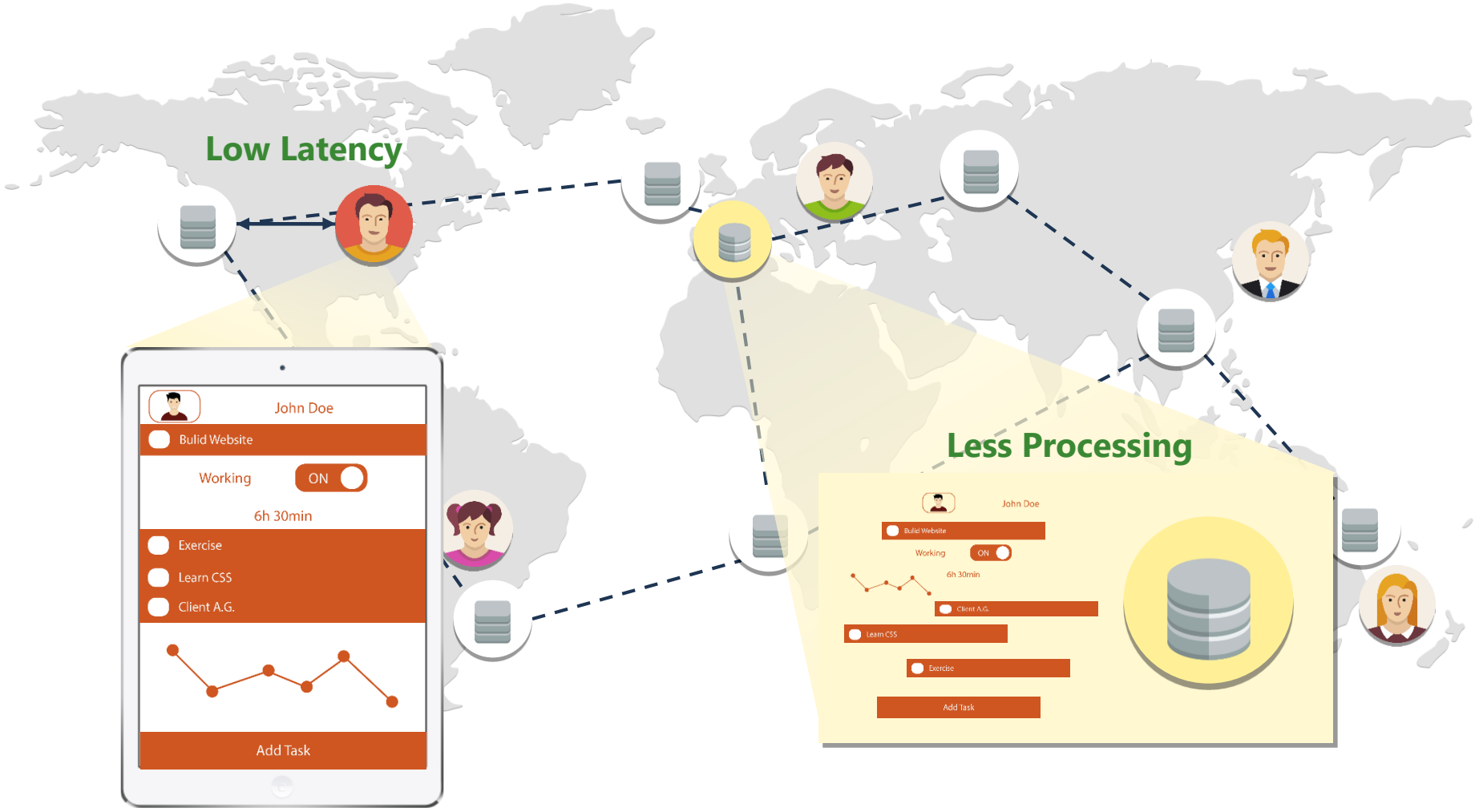
2× Bandwidth = Same Load Time

½ Latency ≈ ½ Load Time

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

Goal: Low-Latency for Dynamic Content

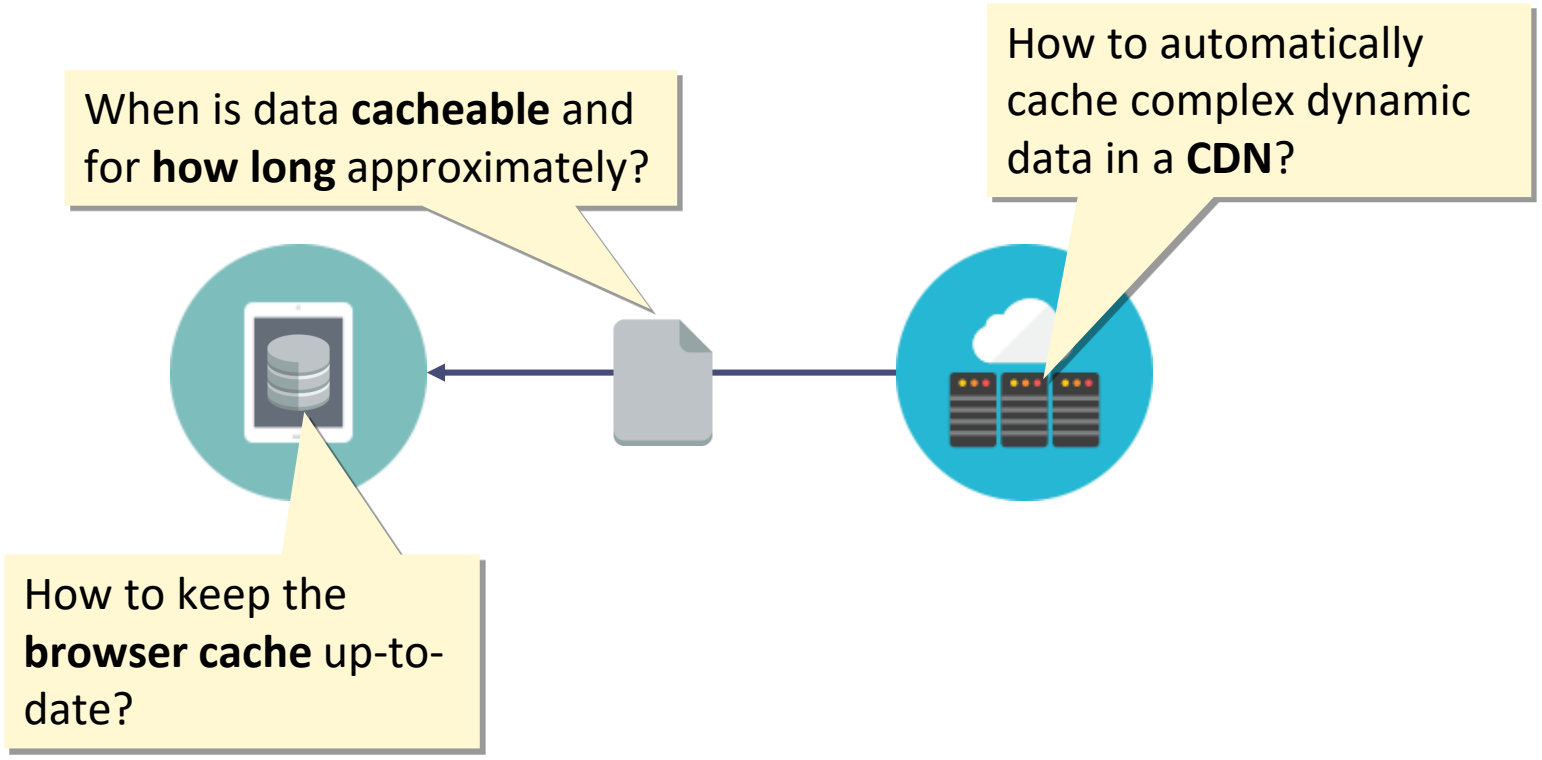
By Serving Data from Ubiquitous Web Caches



Our Approach

Caching Dynamic Data with Bloom filters

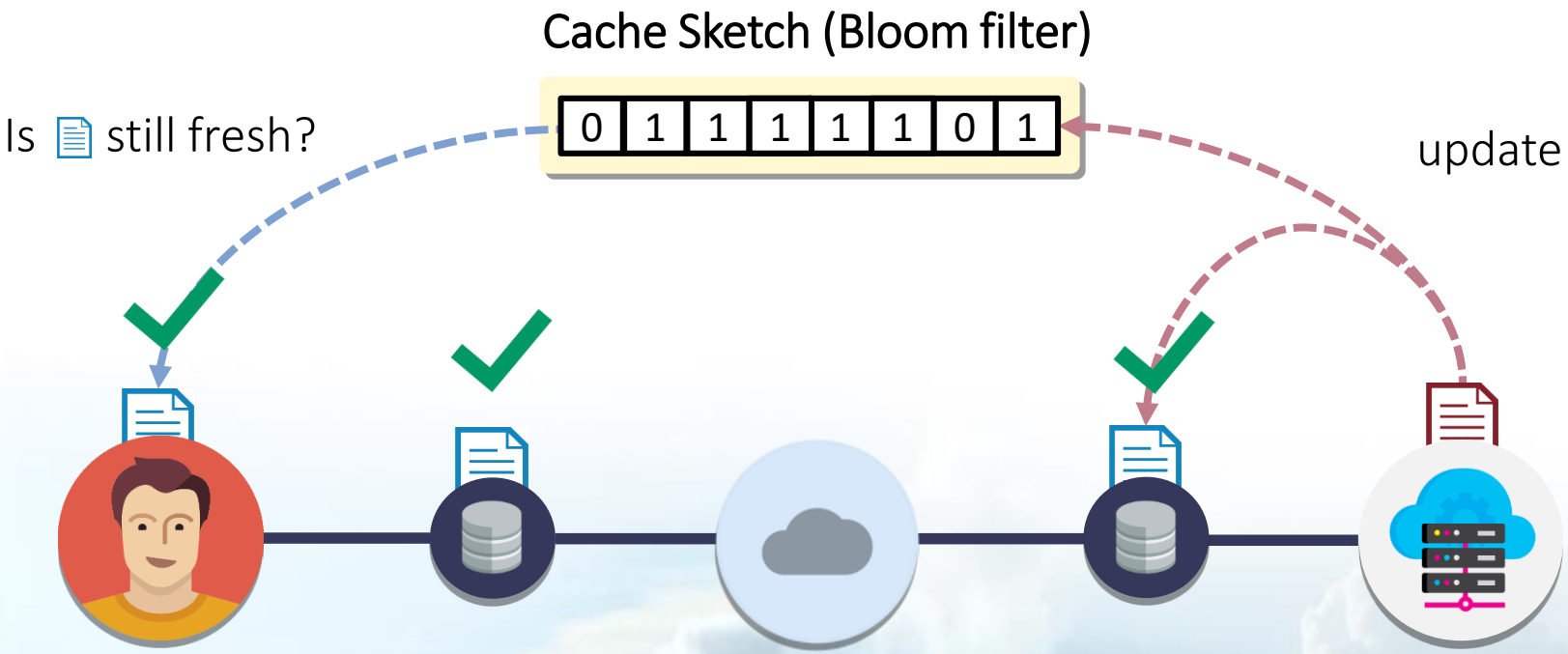
Idea: use *HTTP Caching* for **dynamic data** (e.g. queries)



In a nutshell

In a nutshell

Solution: Proactively Revalidate Data



Innovation

Solution: Proactively Revalidate Data

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.

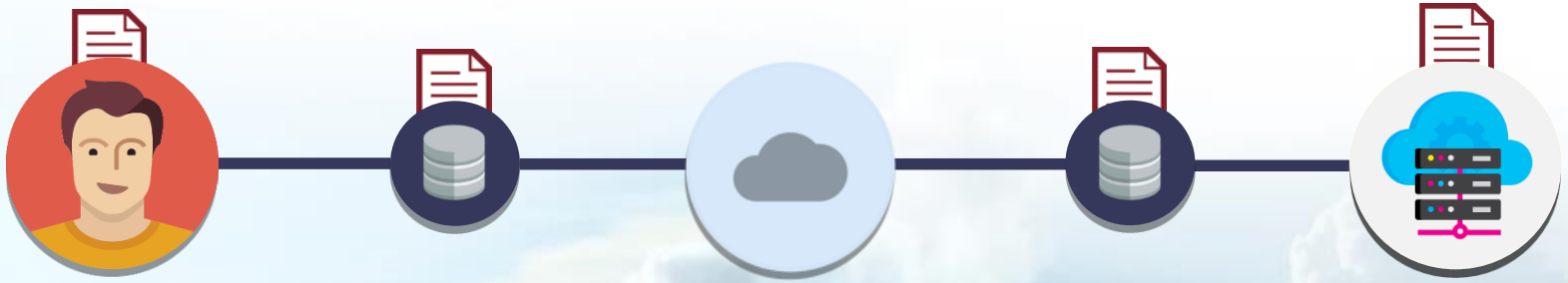
F. Gessert, S. Friedrich, W. Wingerath, M. Schaarschmidt, und N. Ritter, „Towards a Scalable and Unified REST API for Cloud Data Stores“, in *44. Jahrestagung der GI*, Bd. 232, S. 723–734.

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.

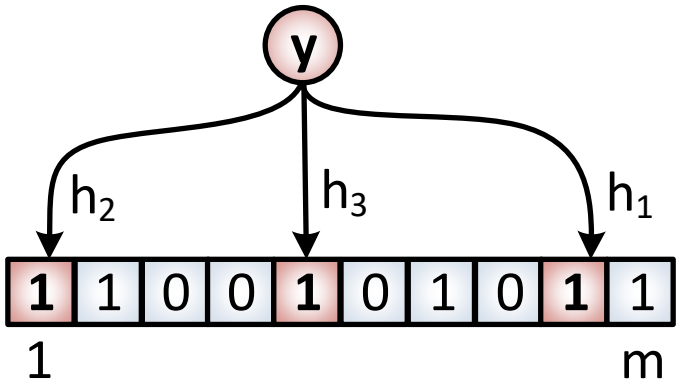
F. Gessert, „Skalierbare NoSQL- und Cloud-Datenbanken in Forschung und Praxis“, BTW 2015



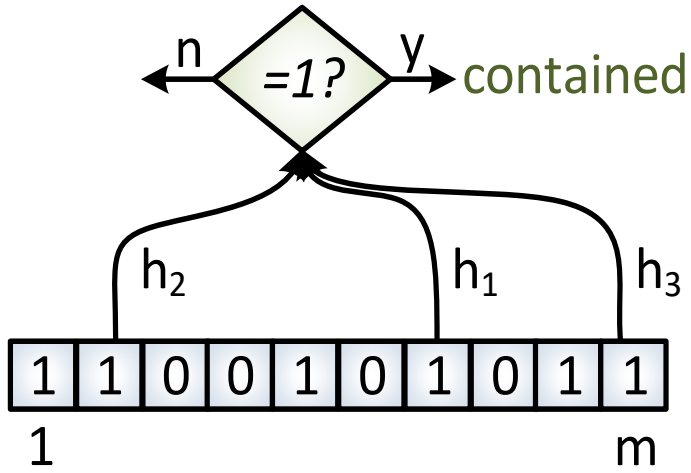
What is a Bloom filter?

Compact Probabilistic Sets

- Bit array of length **m** and **k** independent hash functions
- **insert(obj)**: add to set
- **contains(obj)**: might give a false positive



Insert y

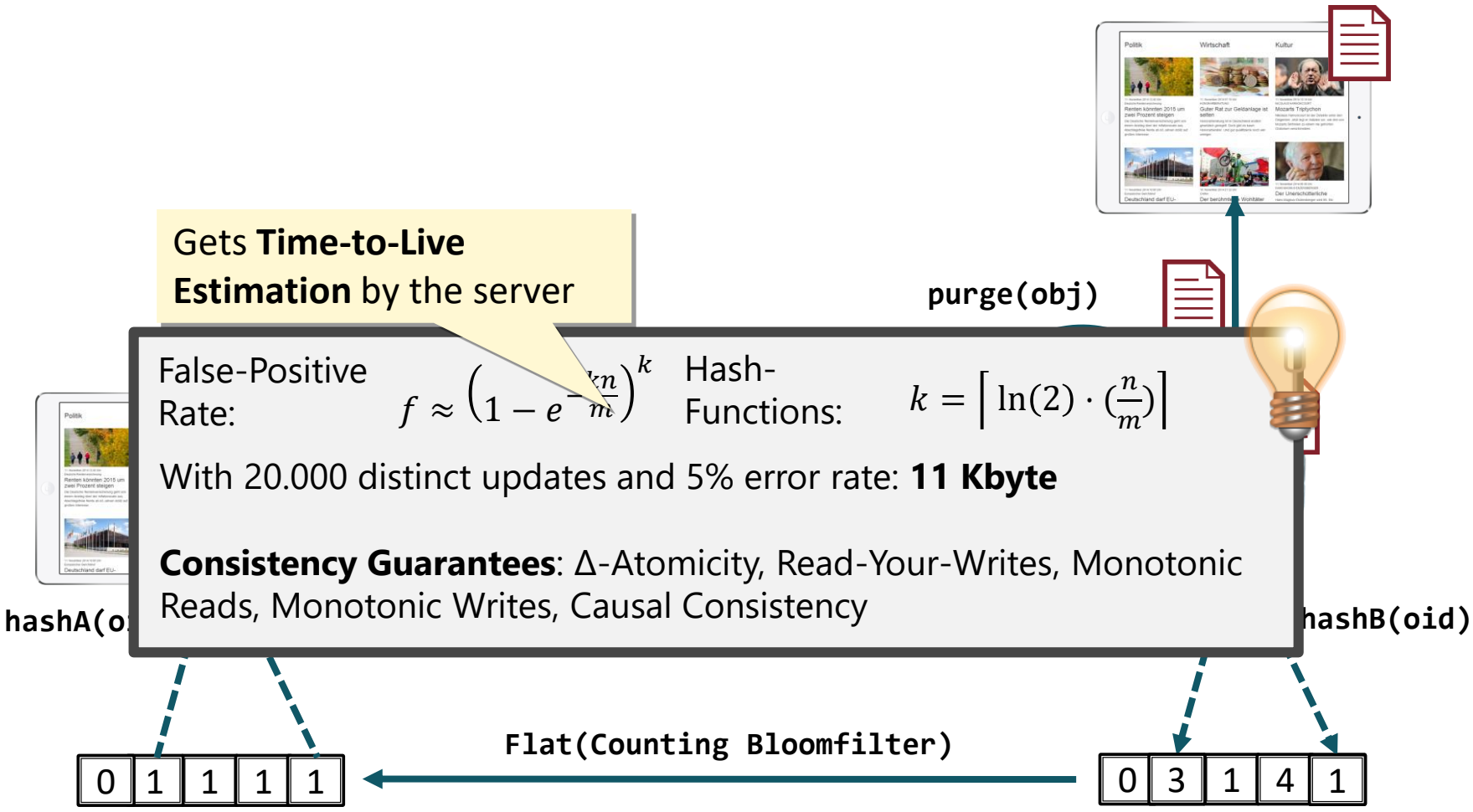


Query x

https://github.com/Baqend/Orestes-Bloomfilter

Bloom filters for Caching

End-to-End Example



Show me some Code!

End-to-End Example

BaQend Product About us Blog Whitepaper Docs Log in

Querying Data

1 2 3 4 5 6 7 8 9

If we do not know the ids, we use queries to specify which objects we are looking for. The following query for instance gets us all the Todos in a specific list that are not marked as done:

```
DB.Todo.find()
    .equal("listId", "myList")
    .notEqual("done", true)
    .resultList(myCallback); //Returns a list of matching objects
```

I did a query ›

`resultList` returns all matching objects, whereas `singleResult` only returns the first match.

Baqend supports many query operators, which allow very sophisticated lookups, including ordering, geospatial queries, text search and boolean combinations of query predicates.

HTML CSS JS Result EDIT ON CODEPEN

```
var TodoService = (function() {
  return {
    //load unfinished Todos
    unfinished: function(listId) {
      return DB.Todo.find()
        .equal("listId", listId)
        .notEqual("done", true)
        .resultList();
    }
}})();

//Load data
function onReady() {
  insertData().then(function() {
    return TodoService.unfinished("my-list"); //query
  }).then(function(todos) {
    printItems("Unfinished Todos:", todos);
  });
}

//insert some data
function insertData() {
  var todo1 = new DB.Todo();
```

LIVE

Unfinished Todos:

- My Todo [id:/db/Todo/my-todo] from list my-list
- My other Todo [id:/db/Todo/bfc54e9e-e558-4031-aad2-4e4d7ba4cd86] from list my-list
- My other Todo [id:/db/Todo/my-todo-2] from list my-list
- My Todo [id:/db/Todo/7c8bb7f9-0992-4fc6-9d3d-1f6ec40c6ea5] from list my-list
- My Todo [id:/db/Todo/e6dc959b-5b6b-4f22-90c6-8bcdfd41ef7f] from list my-list
- My Todo [id:/db/Todo/6e687190-5c6f-4aaf-acc9-ccef27051380] from list my-list

Try this: www.baqend.com#tutorial

Our Mission at Baqend:

„Kick load times in the ass using Bloom filters“

www.baqend.com
@Baqendcom