Florian Bücklers

Wie man ein Backend-as-a-Service entwickelt: Lessons Learned

Architecture

Who is talking today?

Florian Bücklers

CTO & Co-Founder
Baqend Platform
Speed Kit Plugin

BaQend

Backend-as-a-Service

Overview

Backend-as-a-Service

Firebase

kinvey

Microsoft Azure

Parse

BaQend

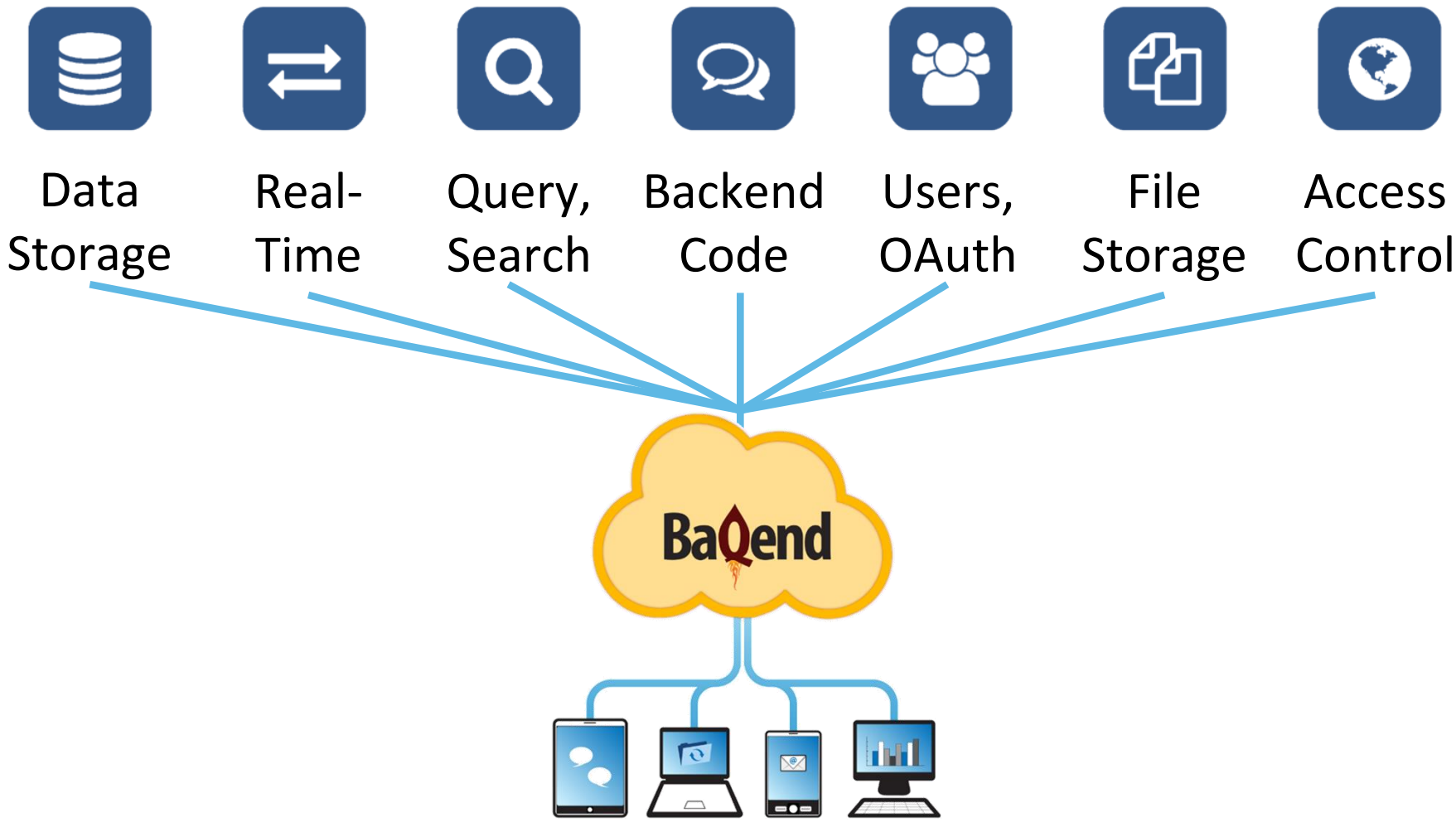
Backend-as-a-Service

Feature Sets

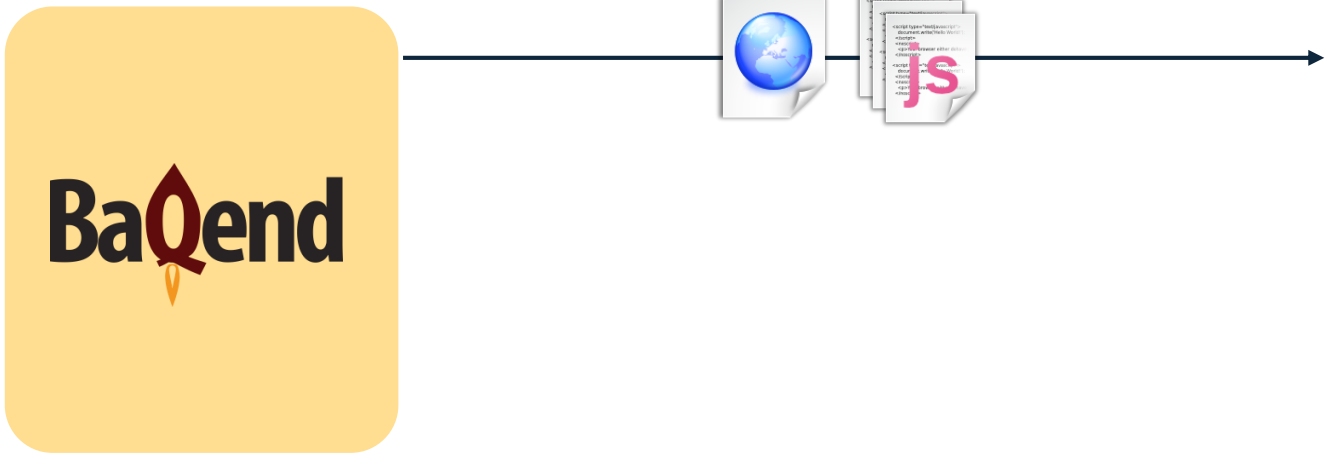
API for application features

Hosting and **Delivery**

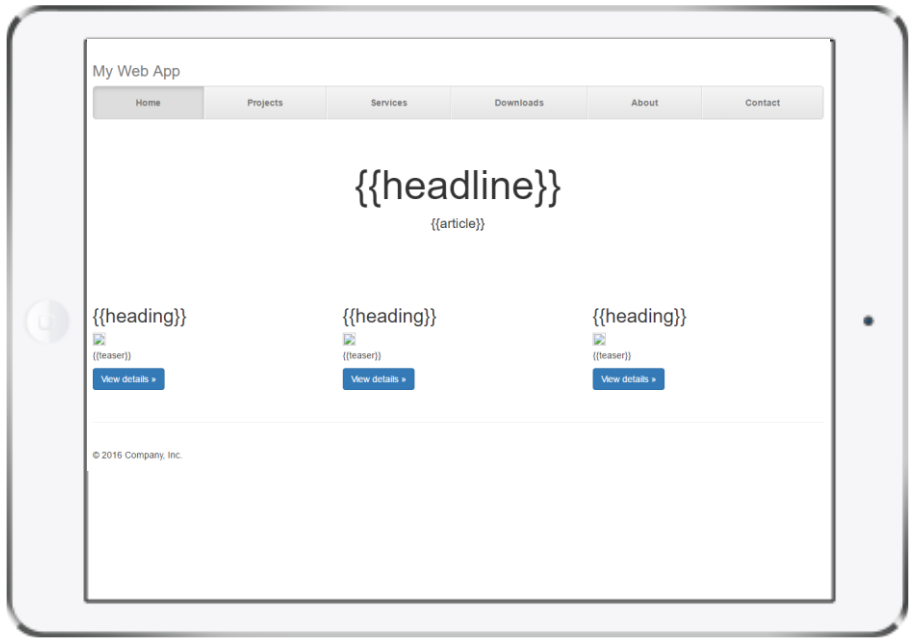
REST API and **JS SDK**



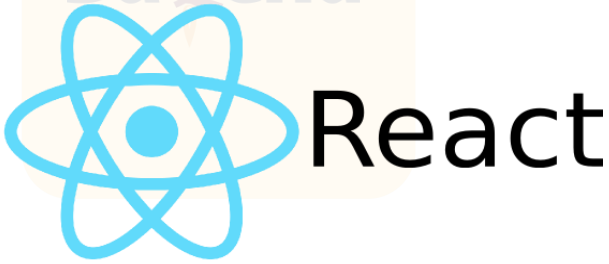
Frontend



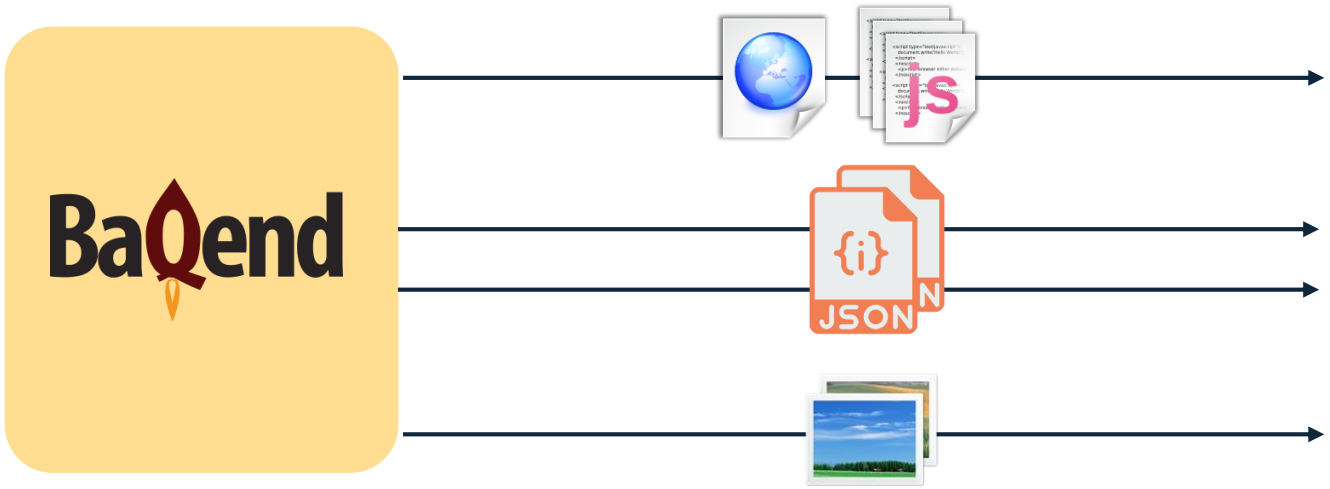
GET /app.html
GET /js/main-34da93.js
GET /css/main-9ad7ca3.css



Compatible with:



Frontend

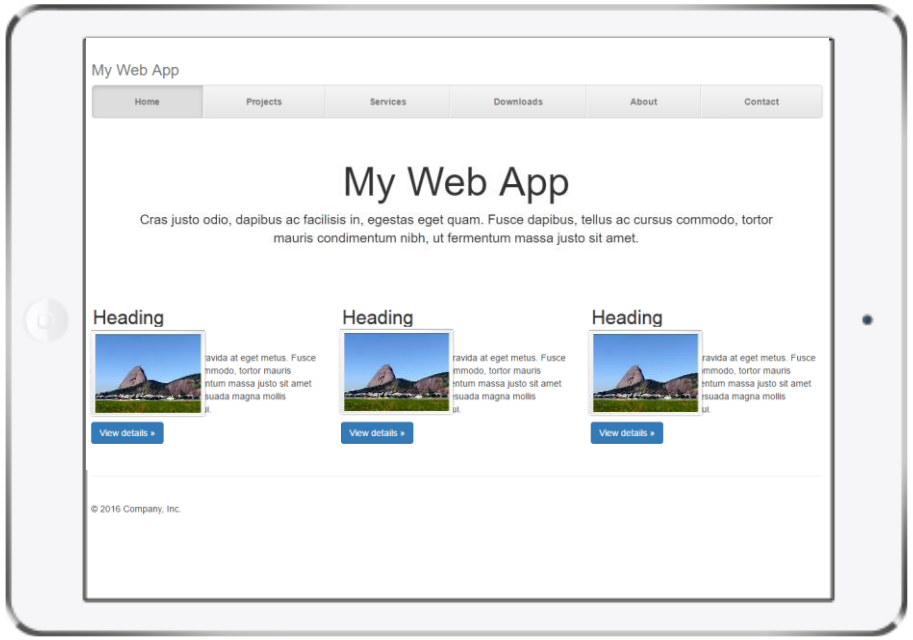


GET /app.html
GET /js/main-34da93.js
GET /css/main-9ad7ca3.css

```
db.Page.load('main')
  .done(...);
```

```
db.Page.find()
  .descending('published')
  .limit(3)
  .resultList(...);
```

GET /img/pic005.jpg
GET /img/pic017.jpg
GET /img/pic022.jpg

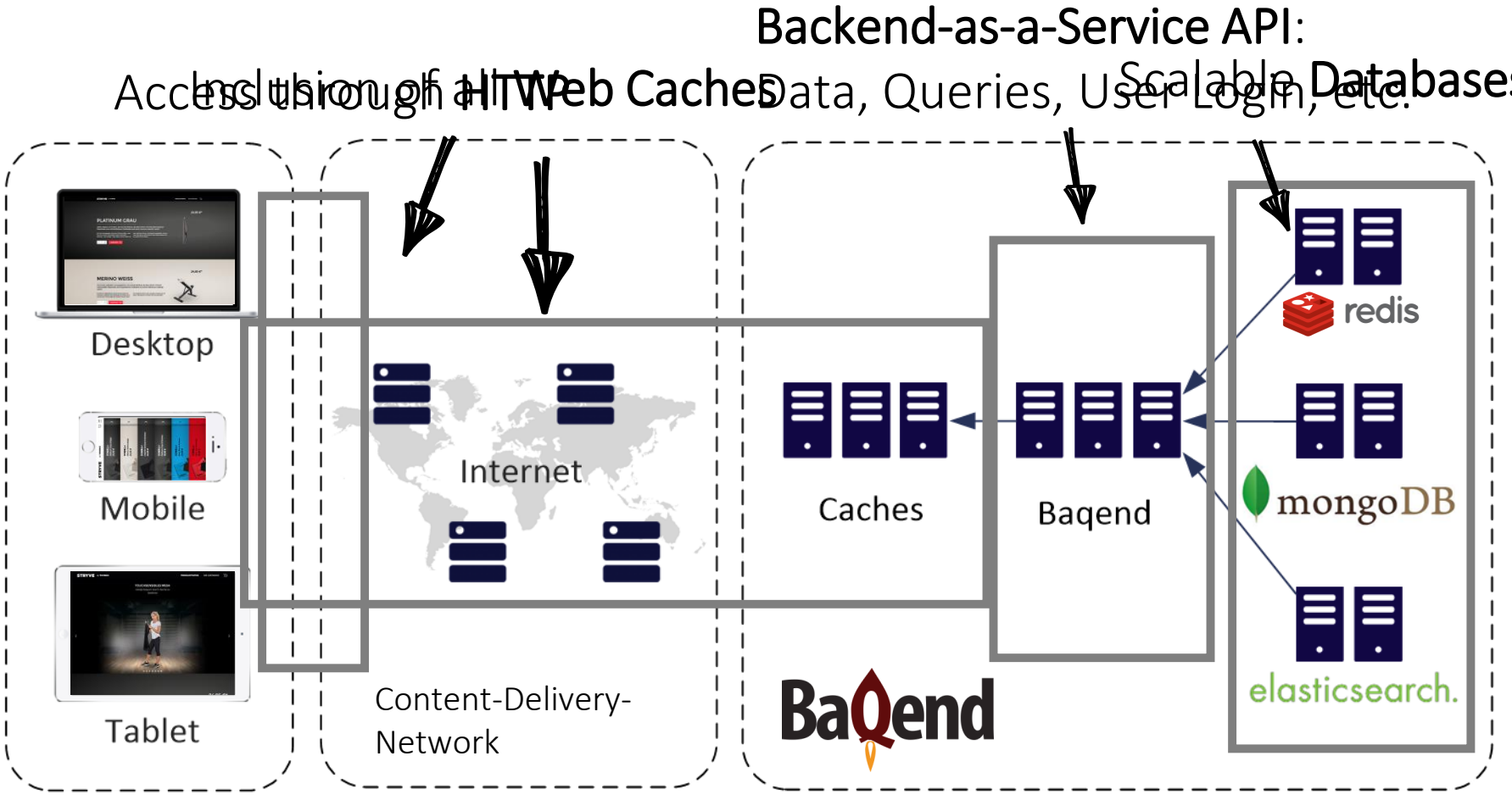


Challenges

- Multi-Tenancy
- Load-Balancing and Horizontal Scaling
- High Availability Deployments
- Rolling Updates without Downtime
- Session Handling and Authorization

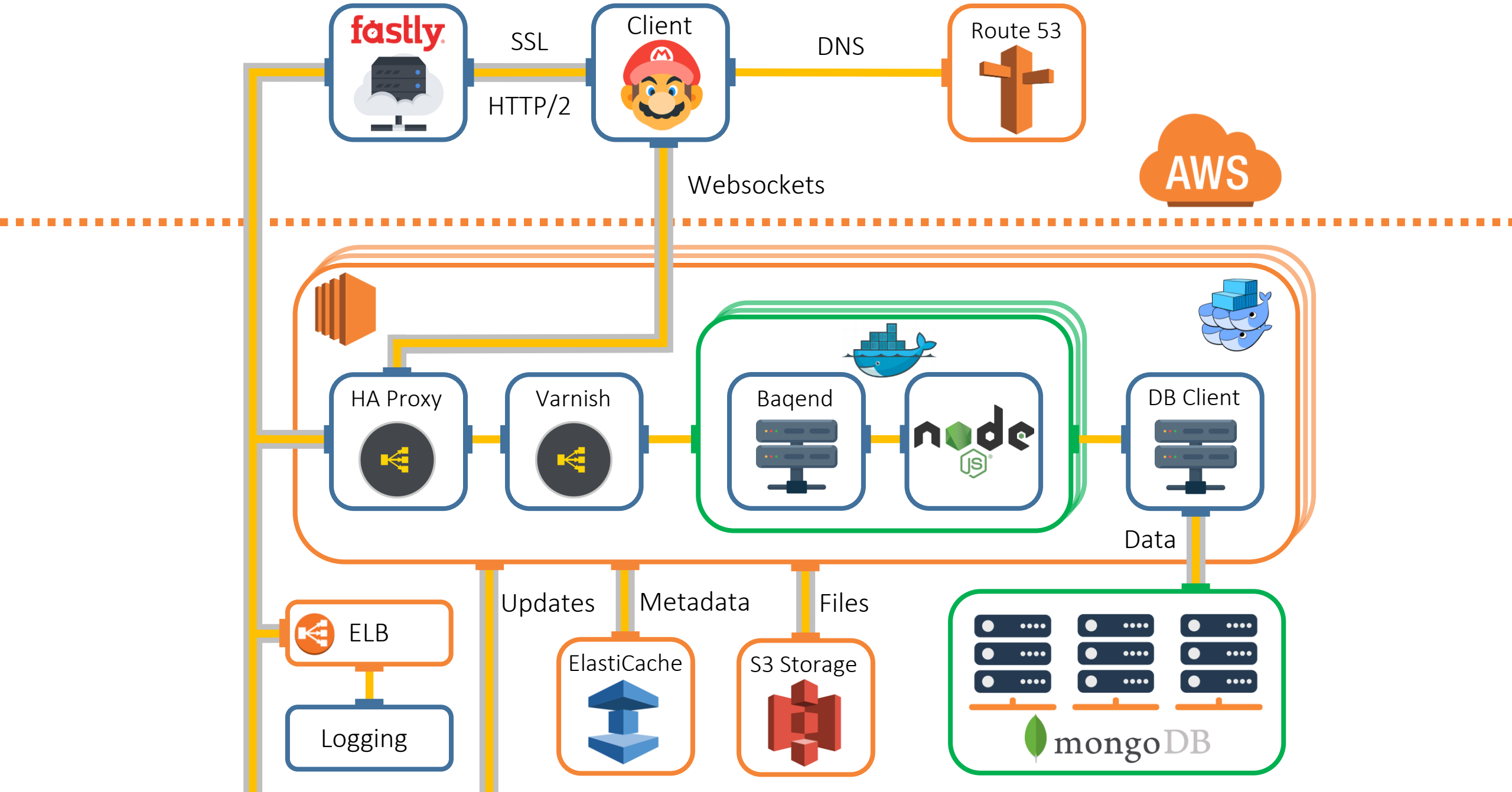
Backend Architecture

Baqend Cloud



Backend Architecture

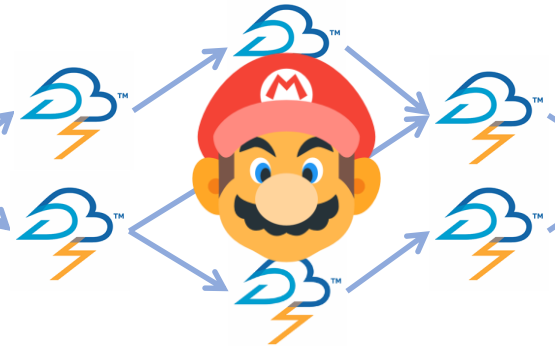
Baqend Cloud


fastly
SSL
HTTP/2
Client
DNS
Route 53
AWS
Websockets
HA Proxy
Varnish
Baqend
node
DB Client
Data
ELB
Logging
Updates
Metadata
Files
ElastiCache
S3 Storage
mongoDB

Updates

Updated Query

Storm Query Matching



Pub/Sub

Pub/Sub

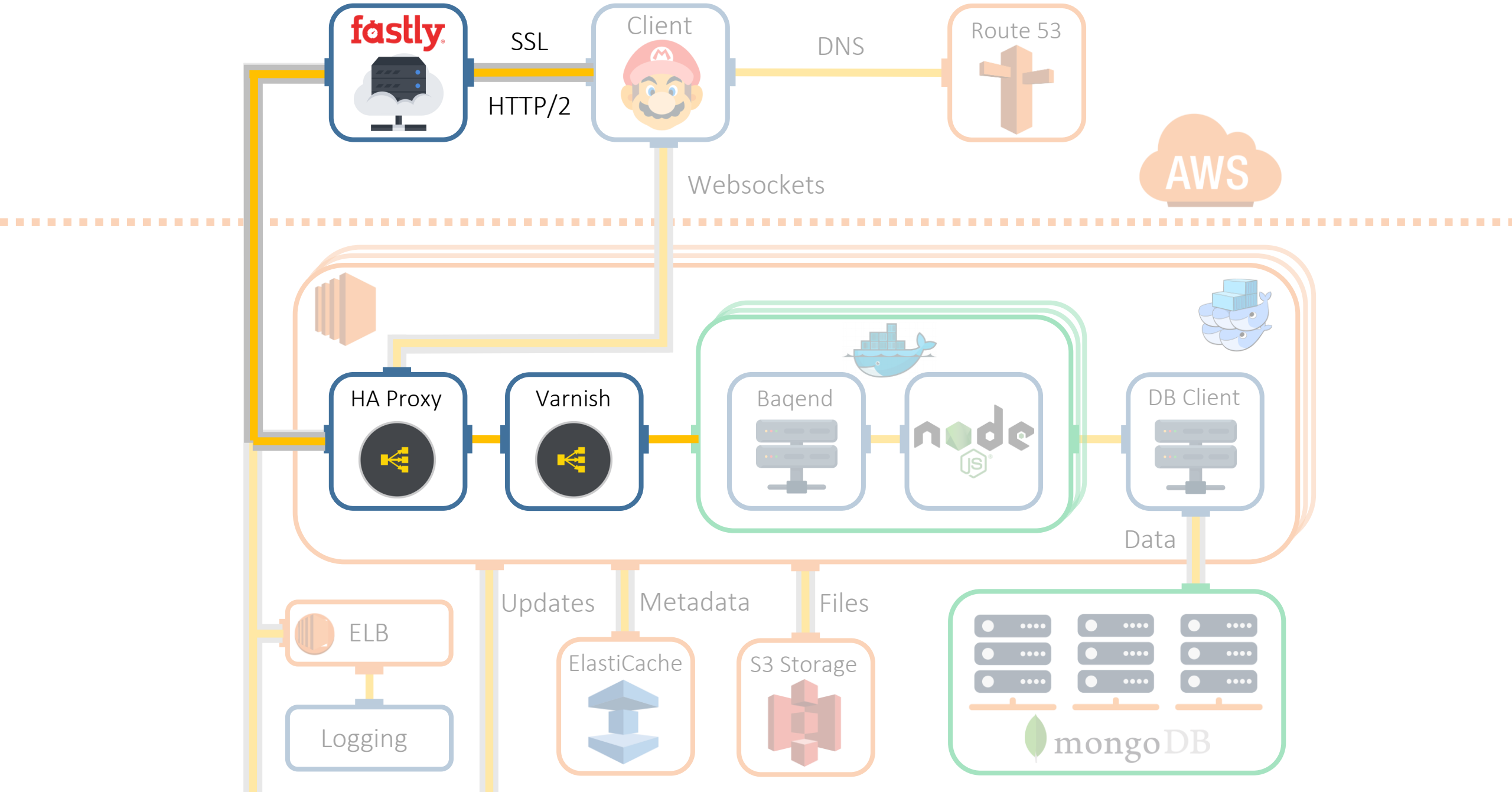
Management and Orchestration Server

Manager

Swarm

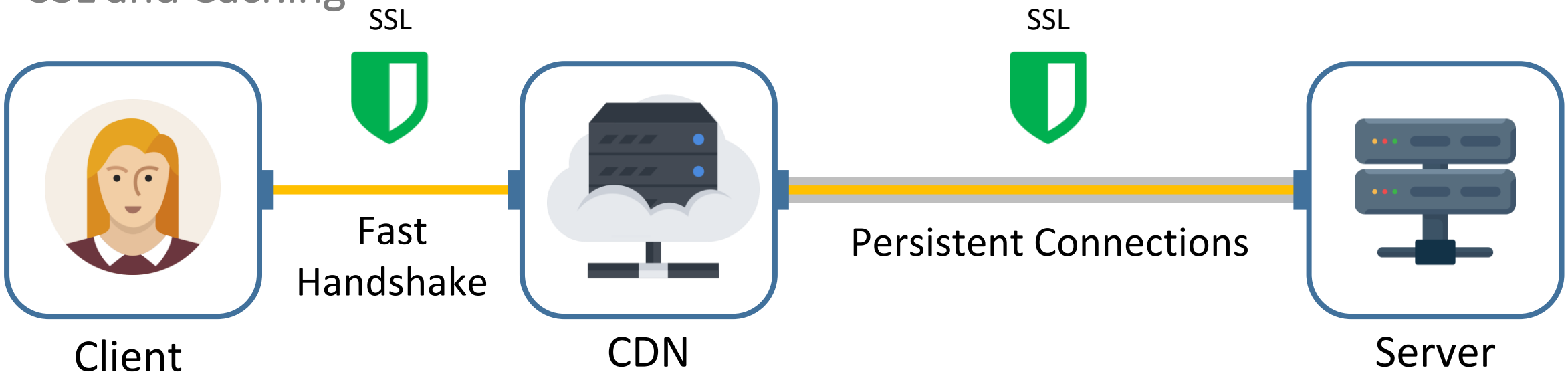
Nimbus

Zookeeper


fastly
SSL
HTTP/2
Client
DNS
Route 53
AWS
Websockets
HA Proxy
Varnish
Baqend
DB Client
Data
ELB
Logging
Updates
Metadata
Files
ElastiCache
S3 Storage
mongoDB

Content Delivery Network (CDN)

SSL and Caching

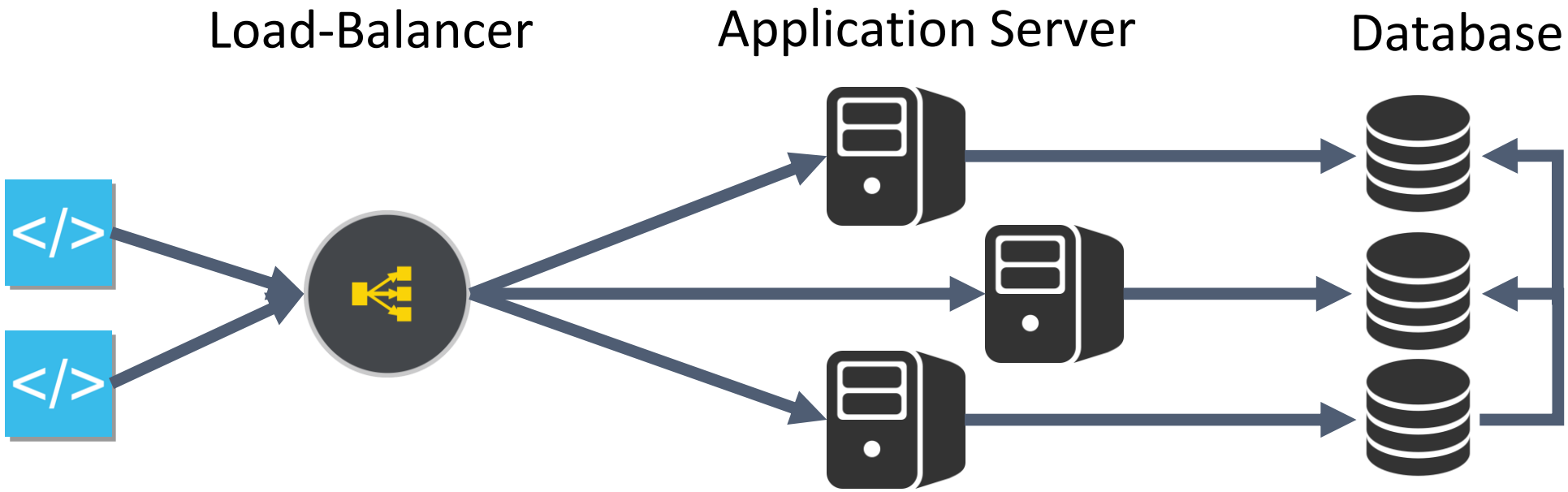


- Caching at **the Edge**
- **Low** Read **Latency**
- Early **SSL Termination**

- Warm **Backend Connections**
- Backend **Failover**
- Stale-on-error

Load Balancing

Best Practices



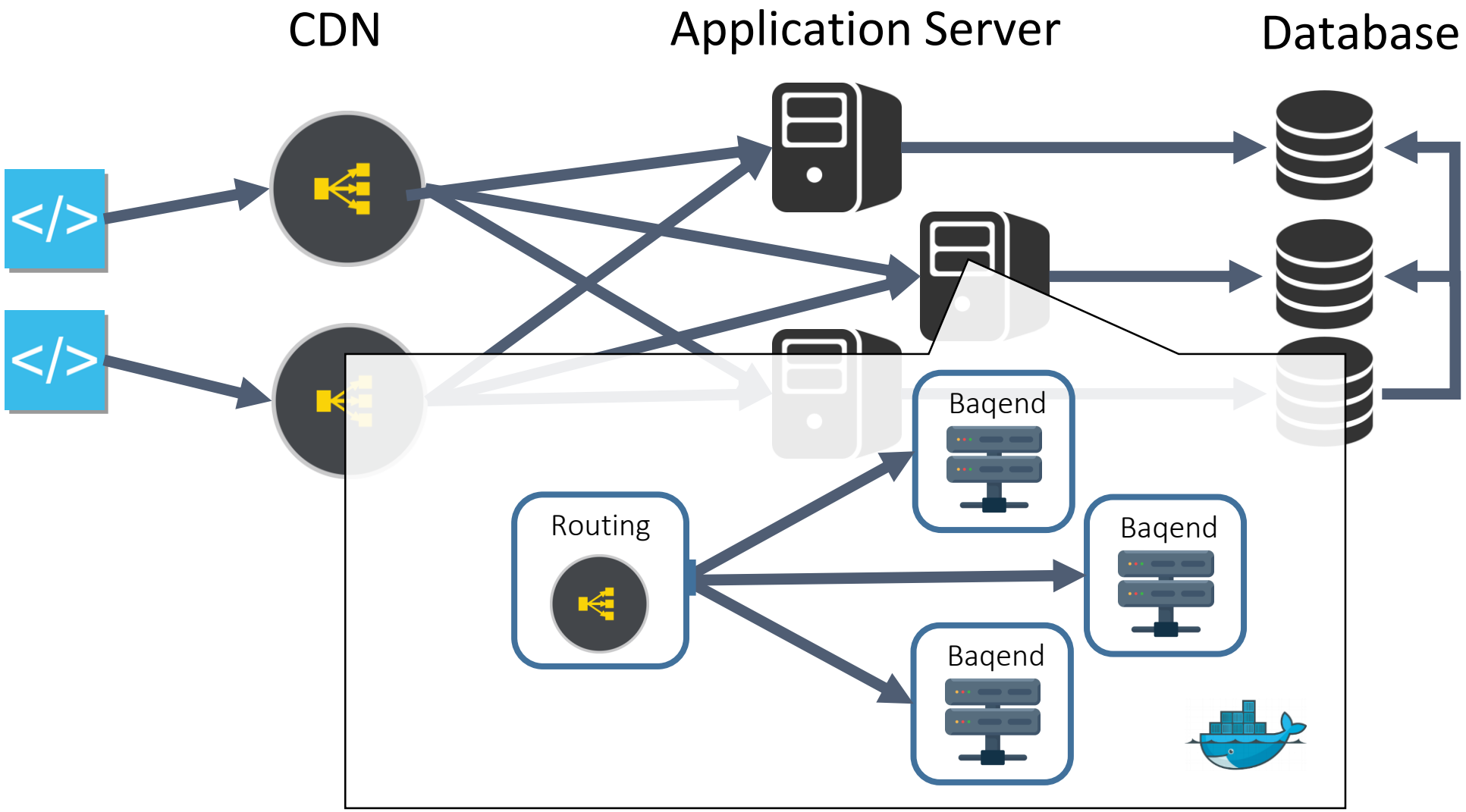
- Load-Balancing
- Auto-scaling
- Failover

- Minimize shared state

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

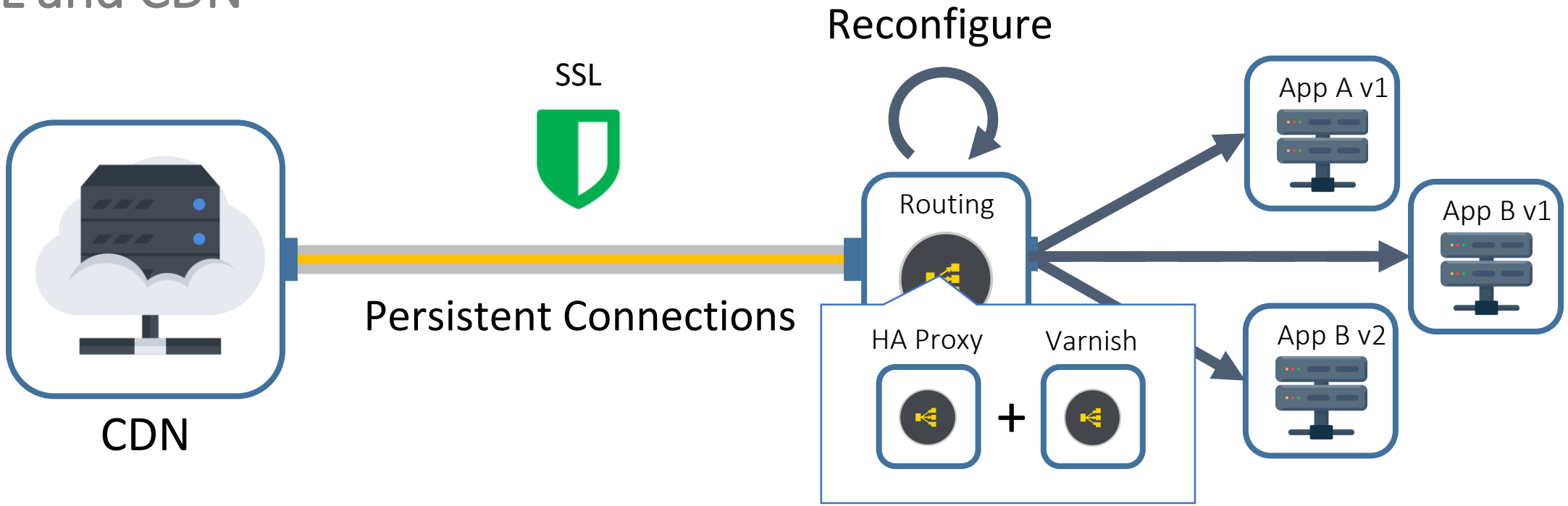
Load-Balancing

at CDN Edge Nodes



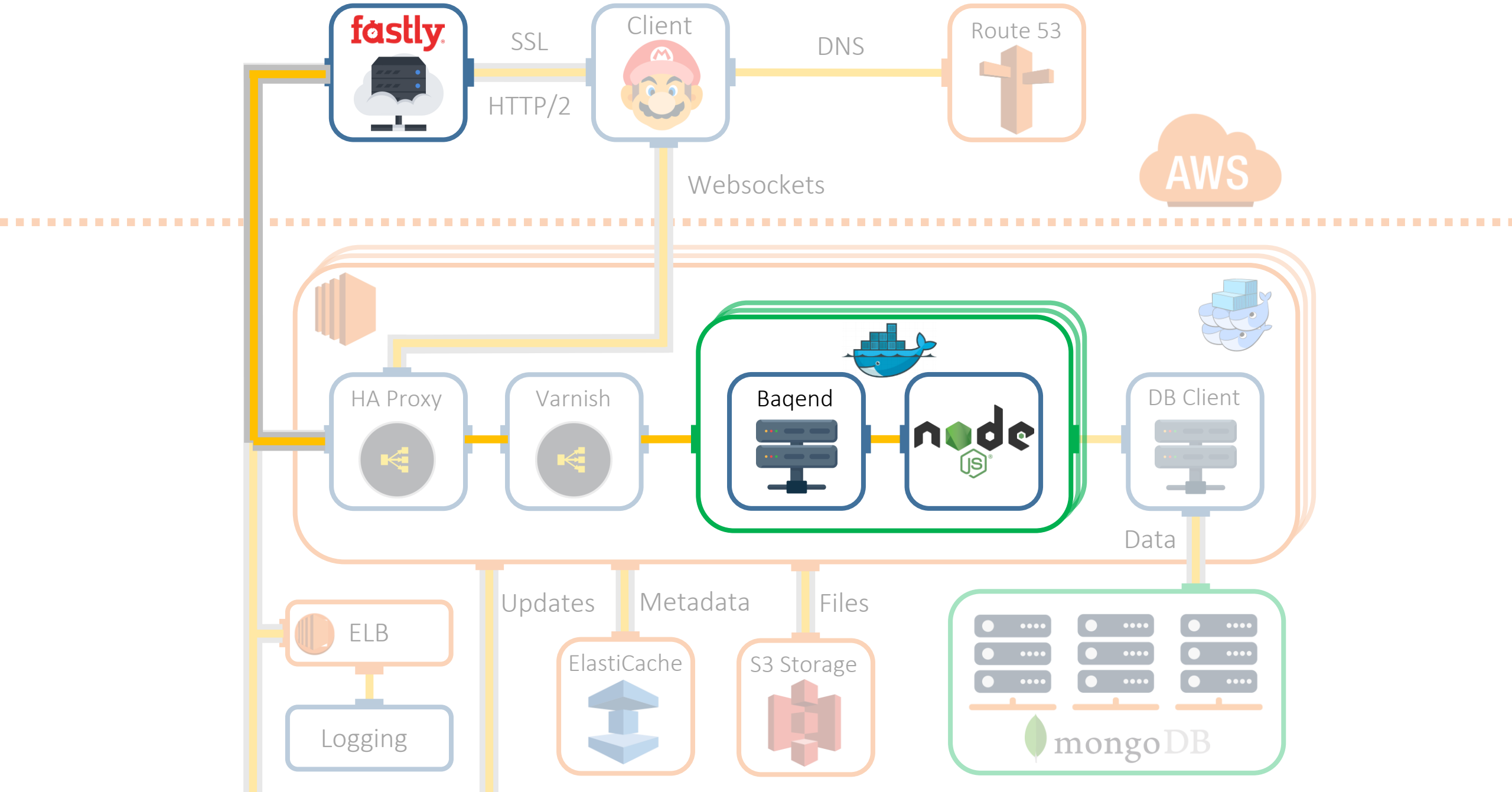
Persistent Connections and Containerization

SSL and CDN



App containers often moving / updated

- New app servers will be registered
- Most load-balancers need a reload (Nginx/HAProxy)
- Old connections kept alive and served with old configuration
  →Requests will be sent to removed app servers

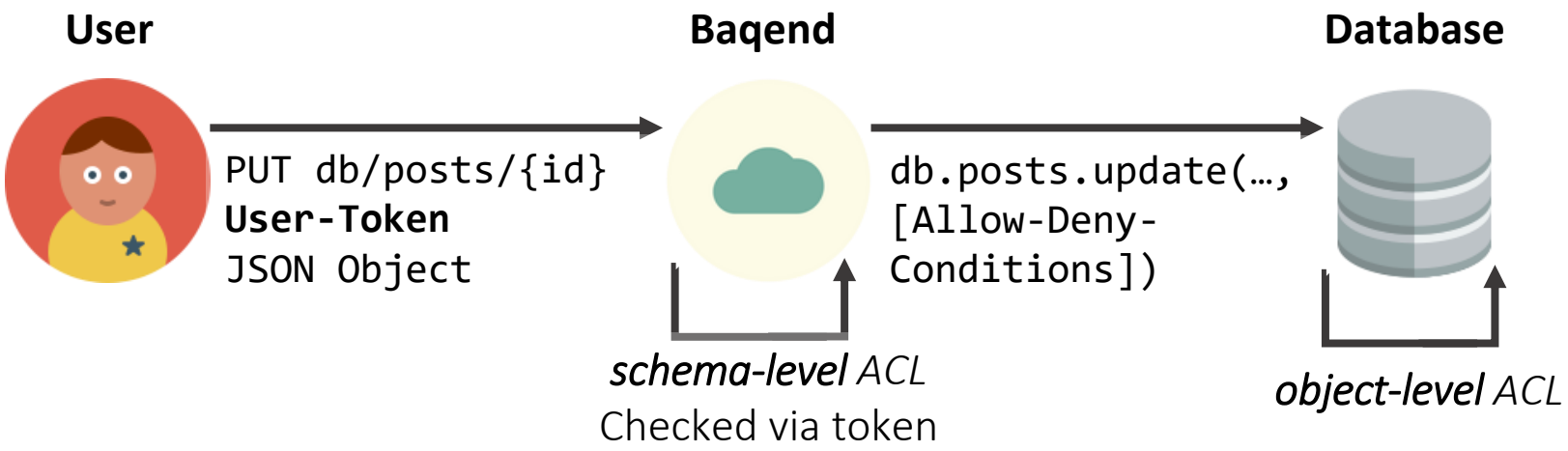

fastly
SSL
HTTP/2
Client
DNS
Route 53
AWS
Websockets
HA Proxy
Varnish
Baqend
node
DB Client
Data
ELB
Logging
Updates
Metadata
Files
ElastiCache
S3 Storage
mongoDB

Access Control

Authorization

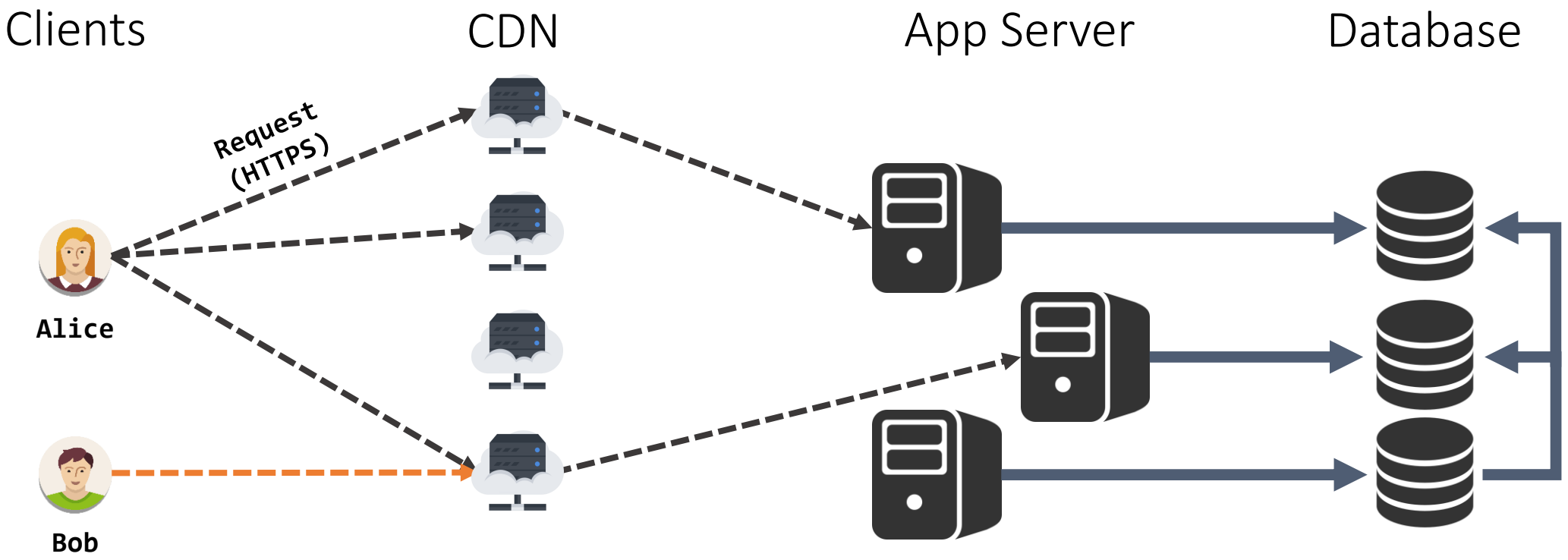
	Load	Insert	Update	Delete	Query
Public	✔	✖	✖	✖	✔
admin (/db/Role/1)	✔				✔
node (/db/Role/2)	✔	✔	✔	✔	✔
+ felix-					

felix-test



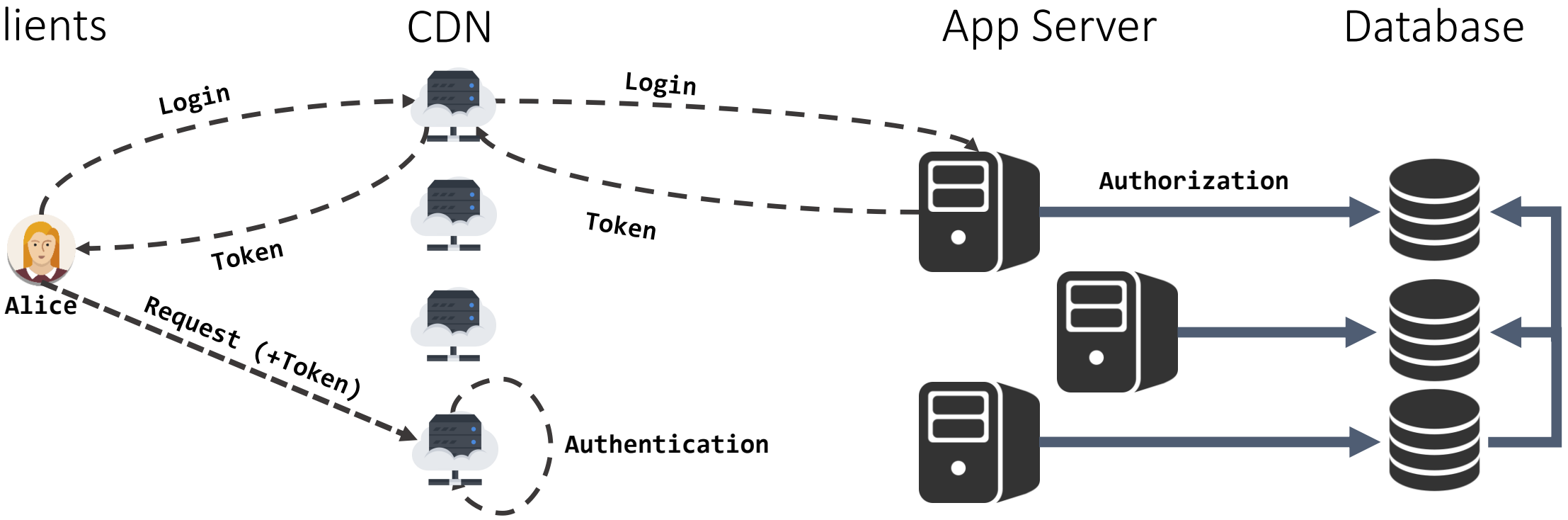
Problem: Sessions ↔ Stateless

Authorization and Authentication in Distributed Systems



Solution: Signed Tokens

Stateless Sessions



Token in Detail

Creation Date

590c965e590c978aAAAAABgAAAABgAAAADda0f7ebbf0ba76b66c97432a886d26895ad669ee

1493997150

05.05.17 17:12:30

Token in Detail

Renew Time

590c965e590c978aAAAAABgAAAABgAAAADda0f7ebbf0ba76b66c97432a886d26895ad669ee

1493997450

05.05.17 17:17:30

Token in Detail

User ID and Role IDs

590c965e590c978aAAAAABgAAAACgAAAADda0f7ebbf0ba76b66c97432a886d26895ad669ee

User ID: 1

Role ID: 2

Role ID: 3

Token in Detail

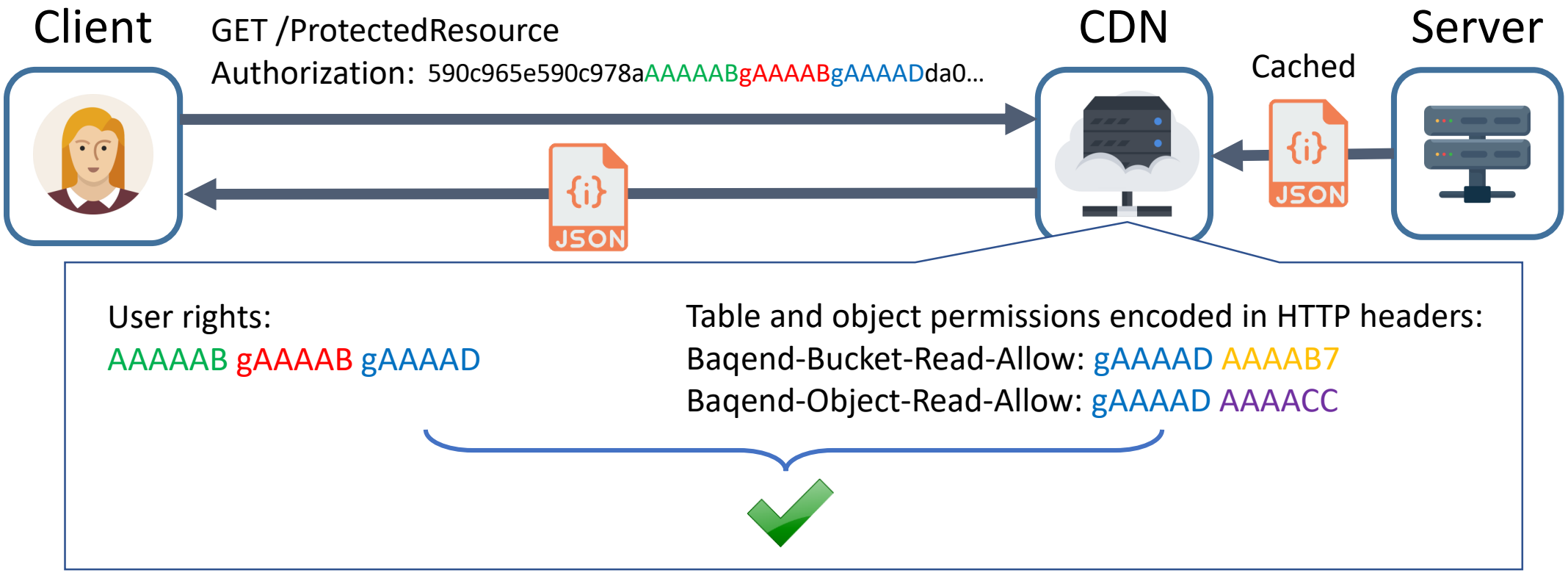
Signature

590c965e590c978aAAAAABgAAAABgAAAADda0f7ebbf0ba76b66c97432a886d26895ad669ee

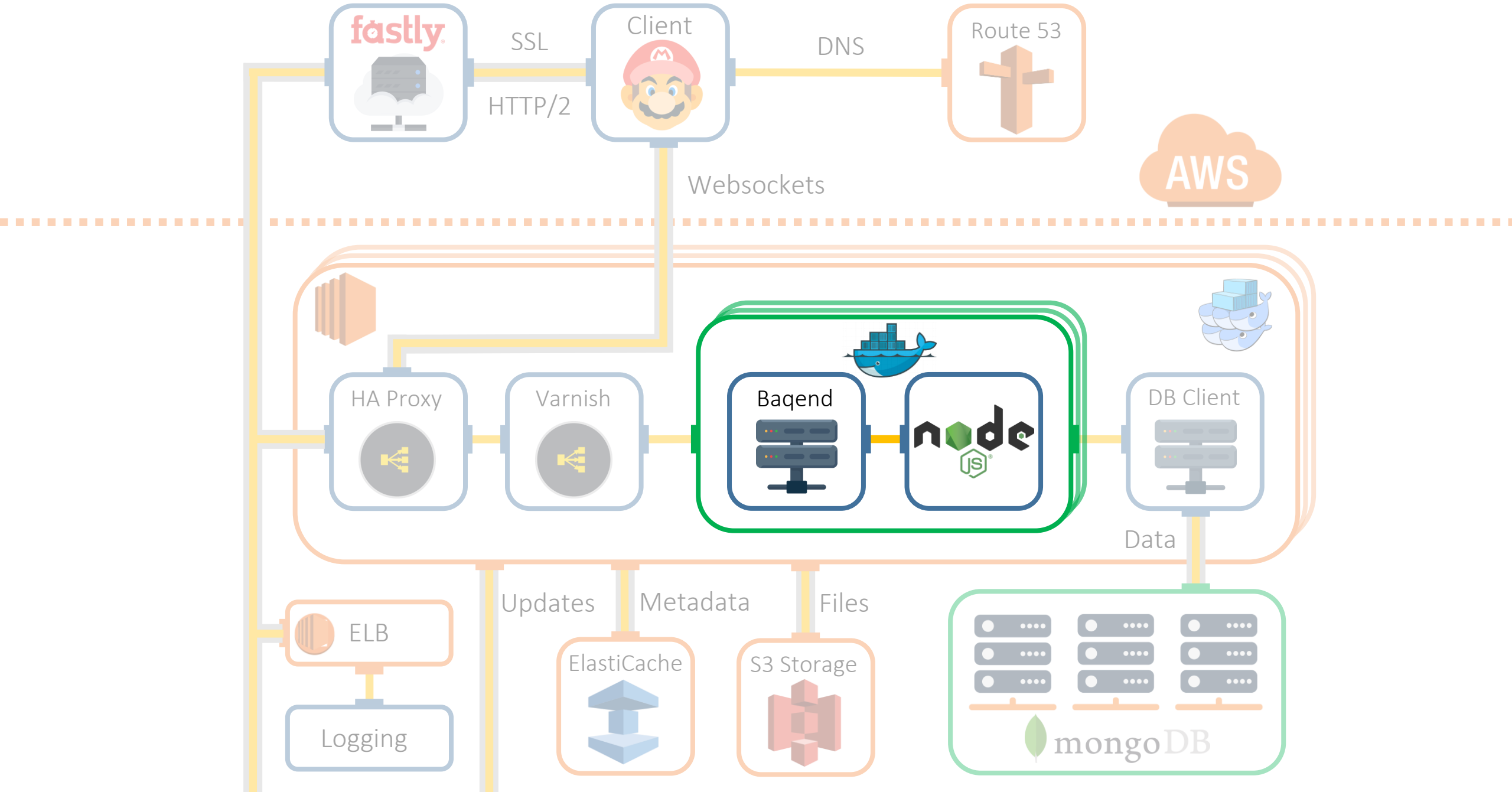
HMAC Signature

Access Control in the CDN

Permissions at the Edge



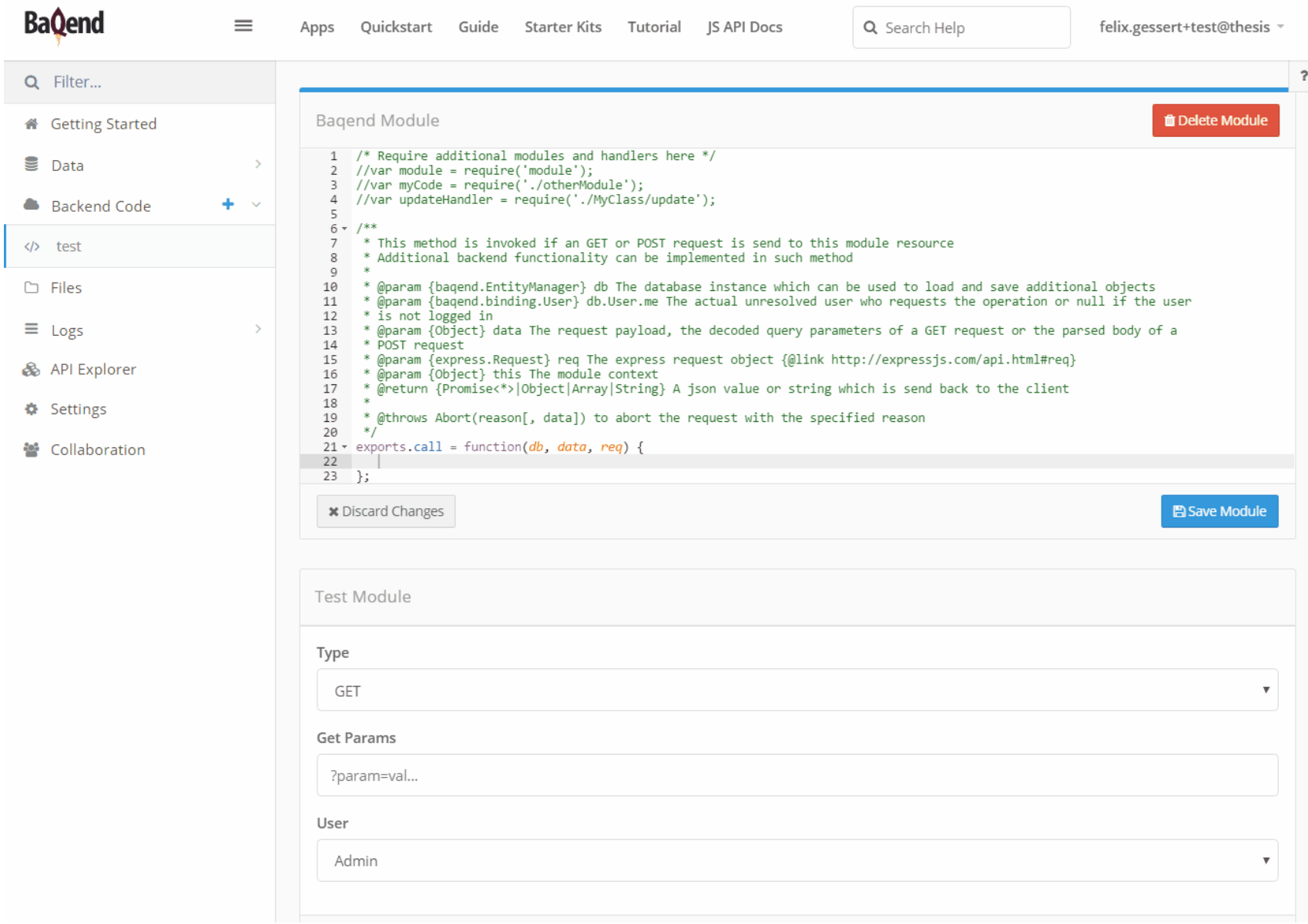
- Required permissions cached at the edge
- Can be checked in VCL (Varnish Configuration Language)
- No backend communication at all


fastly
SSL
HTTP/2
Client
DNS
Route 53
AWS
Websockets
HA Proxy
Varnish
Baqend
node
DB Client
Data
ELB
Logging
Updates
Metadata
Files
ElastiCache
S3 Storage
mongoDB

Backend Code

Trusted Business Logic

- Callable methods
- For **microservices** and custom **web APIs**

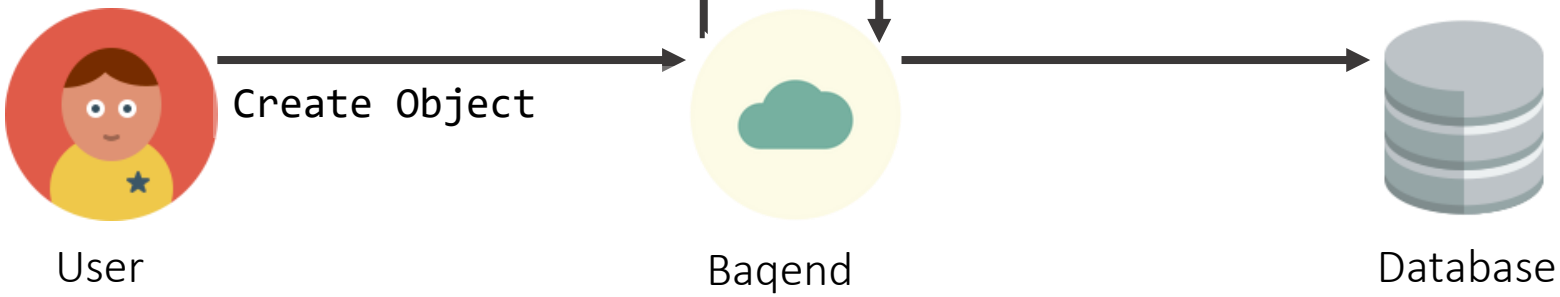


Backend Handler

Hooking Into Updates

- **Handler**: OnUpdate, OnDelete, OnInsert, OnValidate
- Node.js code, same SDK

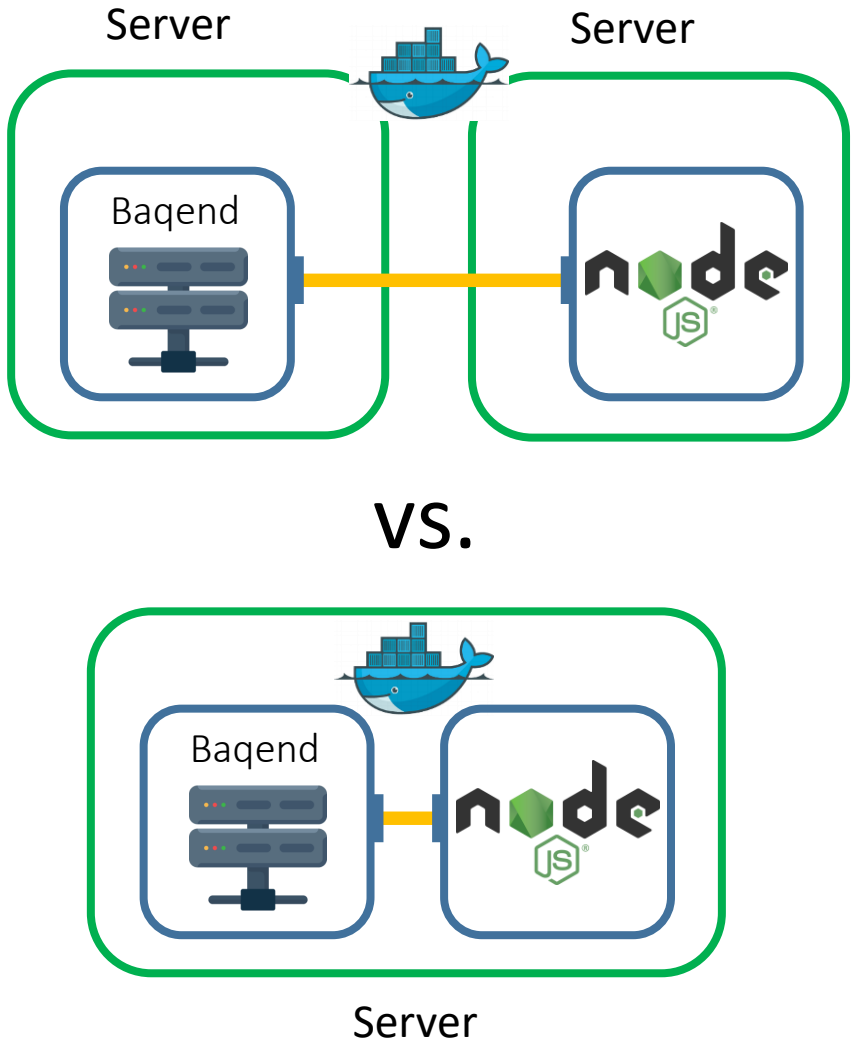
```
function onInsert(DB, obj) {
    var post = new DB.Post(
        {message : "New Data: " + this.name + "."});
    post.save();
}
```



How to Schedule App & Node Server

User Code Isolation & Placement

- Internal access (DB/Storm/Redis/S3)
- Isolation
- Public-only access for user code
- Strong coupling between Node and App server
  - High throughput
  - Low latency



How to Schedule App & Node Server

Container Schedulers

AWS ECS 

+ Automated Server and Service scheduling
- No Docker network isolation
- Only Spread and Binpack as scheduling strategies
- Service definitions can’t be templated

Kubernetes 

+ Automated Service scheduling
+ Complex scheduling strategies
- No network isolation within Pods

Swarm (standalone) 

- No Services
+ Custom scheduling strategies
+ All Docker features
  - Networking
  - Isolation
  - Resource constraints

Docker

```
ubuntu@ip-10-0-0-105:~$ docker ps -a
CONTAINER ID        CREATED             STATUS

d53e07a0f144        33 minutes ago      Exited (128) 292 years ago

0f8e468a1845        47 minutes ago      Up 47 minutes

c6cec3578628        50 minutes ago      Up 50 minutes
0.255:34512->8443/tcp
83135b3529cd        50 minutes ago      Up 50 minutes

3a0589022e7d        52 minutes ago      Up 52 minutes
0.255:34510->8443/tcp
d866ce3f6aca        53 minutes ago      Up 52 minutes

765d35dfe29c        55 minutes ago      Up 55 minutes
0.255:34508->8443/tcp
3691b6b77c46        55 minutes ago      Up 55 minutes
```

Many Docker Containers on one Host

We are starting 183 Containers

→All works as expected

Starting the 184th container …

oci runtime error: could not synchronise with container process: could not create session key: disk quota exceeded

Nope, the disk is not full!

Many Docker Containers on one Host

- Tune the kernel parameters!

```
#https://github.com/docker/docker/issues/22865
sysctl -w kernel/keys/root_maxkeys=1000000
sysctl -w kernel/keys/root_maxbytes=25000000
sysctl -w kernel/keys/maxkeys=1000000
sysctl -w kernel/keys/maxbytes=25000000
```

- Used by docker for container session handling
- Known issue and fixed in newer distributions/kernels

Docker Swarm & AWS

Node Drops

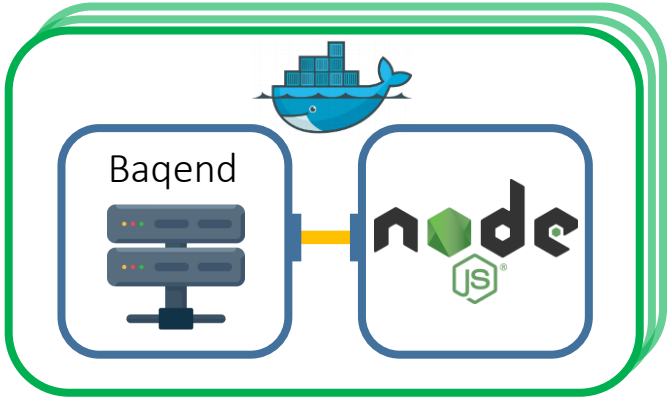
- AWS introduces random network errors occasionally

```
time="2017-09-23T12:33:48Z" level=info msg="Removed Engine ip-10-0-15-21“
time="2017-09-23T12:36:00Z" level=info msg="Registered Engine ip-10-0-15-21 at 10.0.15.21:2375"
```

- Swarm drops a node
  - →Swarm drops all containers
  - →Status checks will fail, since running containers are missing
- You must live with this behavior, if you run a swarm cluster on AWS!

Docker Networks

- Containers can be isolated in docker networks
- Docker creates two iptable entries for each pair of containers
- This results in **$O(n^2)$** iptable entries



→ Adding the 10th container → 100 rules → wait <1s

→ Adding the 100th container → 10 000 rules → wait 30s

→ …

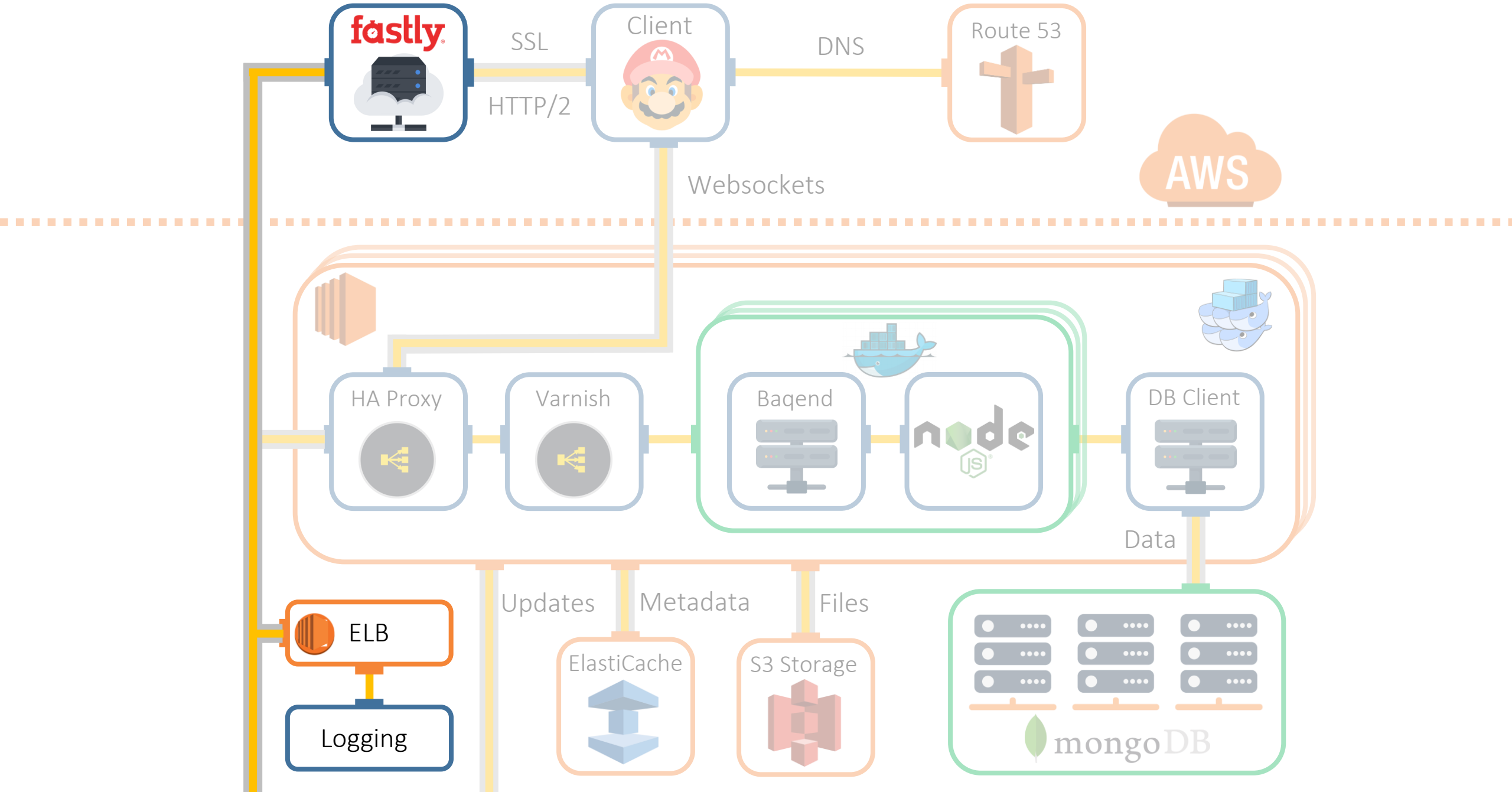
Docker & EBS Volumes

Docker Maintenance vs. EBS Rate-Limiting

- Docker stores all its content in /var/lib/docker
  - Images
  - Volumes
  - Containers
  - Networks
- Docker writes on EBS Volume per default
- Container creation and removal → many IOPS

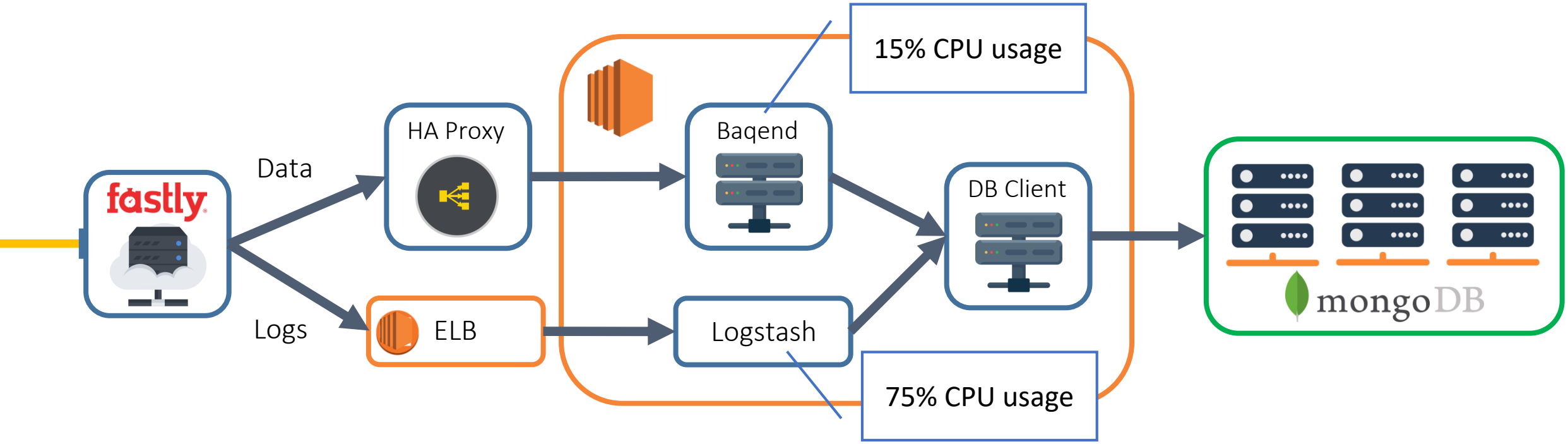
EBS: Network, rate-limited!

→ Causes many creation and removal issues


fastly
SSL
HTTP/2
Client
DNS
Route 53
AWS
Websockets
HA Proxy
Varnish
Baqend
DB Client
Data
ELB
Logging
Updates
Metadata
Files
ElastiCache
S3 Storage
mongoDB

Logstash “Perfomance”

Streaming Logs



- Solution: custom Node.js server → 5% CPU usage!

Wrap-Up

- Persistent connections and zero-downtime deployments are tricky
- Sessions should be stateless, to be horizontally scalable
- Docker has powerful tools to isolate untrusted components in a secured infrastructure
- Many Docker issues are caused by wrong defaults
- Test performance of each component of your infrastructure

We are hiring.

Frontend Developers
Mobile Developers
Java Developers
Web Performance Engineers

Contact us.

BaQend

Florian Bücklers · fb@baqend.com · www.baqend.com

Questions?

Our other talks:

Th. 16:00 Real-Time Databases Explained: Why Meteor, RethinkDB, Parse and Firebase Don't Scale

Fr. 10:00 Real-Time Anwendungen mit React und React Native entwickeln

Fr. 14:00 AMP, PWAs, HTTP/2 and Service Workers: A new Era of Web Performance?

BaQend Florian Bücklers · fb@baqend.com · www.baqend.com