Enter a new world of web development where everything is serverless

What's possible and how will infrastructure be shaped in the future

WAQ 18
Bastian Widmer - @dasrecht | @amazeeio

Bonjour WAQ18!

Bonjour WAQ18!
Pardon my Français - or the absence of it.

We will talk about: serverless, containers, infrastructure and modern software architecture

Overview

- What do you do?
- Past
- Your Goal
- Serverlessness
- Our way into the new world
- To the future and beyond!
- How to get started

$> whoami bastian

- System Engineer at amazee.io
- Containers in Production 👻🤖
- Zurich, Switzerland
- English, German, Swiss-German and a bit of French
- @dasrecht
- Too many side projects!*
    - TEDxBern
    - DevOpsDays Zurich
    - CommunityRack.org
    - Running TOR nodes for fun
    - Working with real containers

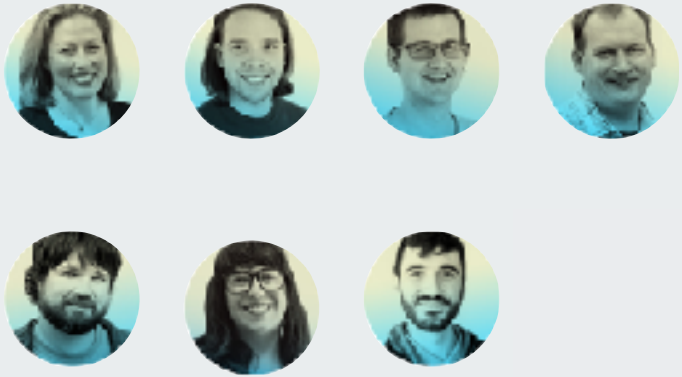
* this list is not complete!

$>


Caldeira
1633
Toggenburger

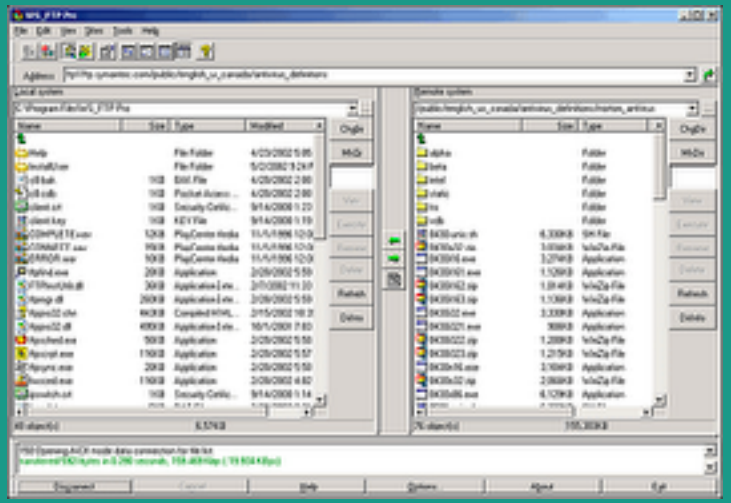
amazee.io?

amazee.io



- Fully Open Sourced Hosting Platform for ~~Drupal~~ Web Projects
- Hosting since 8 years
- We're a remote team of 7
  - Zurich, Switzerland
  - Barcelona, Spain
  - Austin, TX
  - Portland, OR
  - Melbourne
- Hosting in 16 different countries

Back in the day™




Microsoft
Internet
Explorer 5

What is your goal?

Write code
Deploy code
~~Manage Infrastructure~~
Enjoy that things just work!
Learn something along the way.

Wouldn't it be cool to define our infrastructure directly in our project repository?

— one of my colleagues, 2012

2015

- The bleeding edge gets dull after a while
- Full rebuild of the hosting infrastructure
- Configuration management
- Per-Commit Deployments
- Local Dev Environment built with the same tools based on vagrant

2016

- It’s not flexible enough
- VMs use a lot of space and updating is a pain!

- Let’s take look at Docker
- Building tooling around Docker - pygmy

- **Why using Containers just locally?**

Pygmy: https://github.com/amazeeio/pygmy

2017
2018

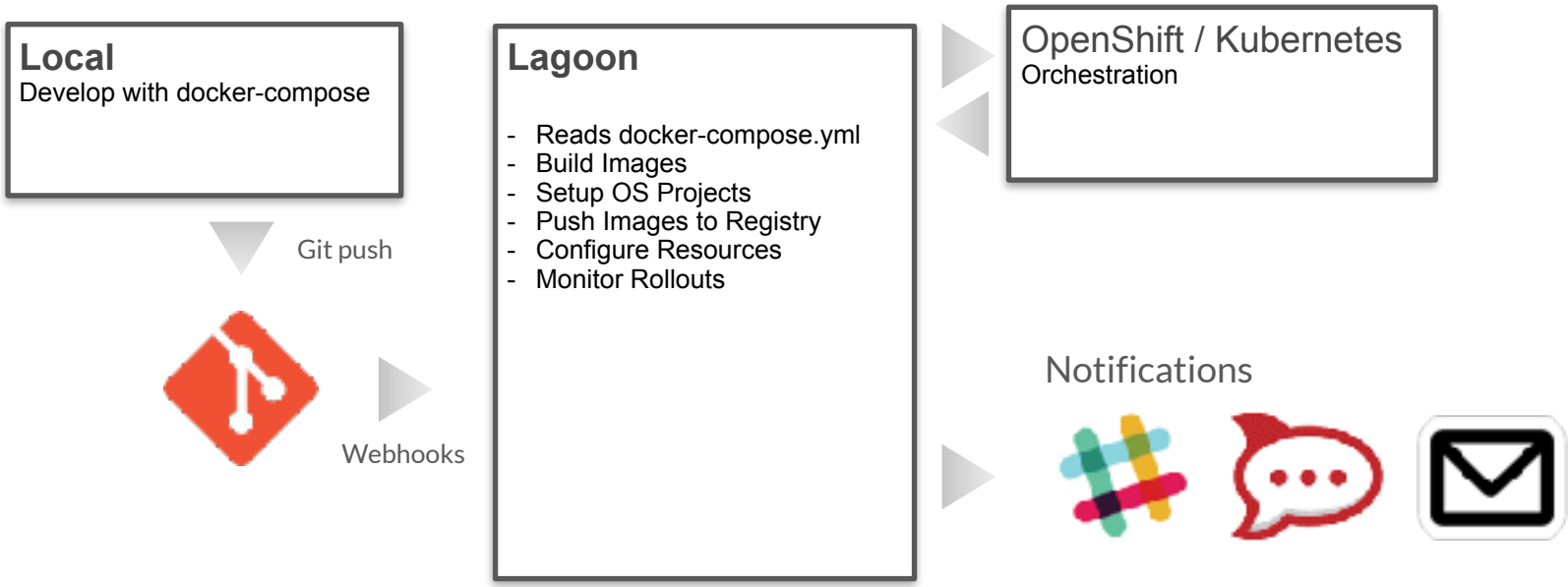
- 2017
  - First Website running on Docker!
  - Eureka! This actually works!
  - Pull-Request Environments
  - Open Source?
  - Open Source! - Lagoon
- 2018
  - Working towards V1.0.0 Release

Lagoon?

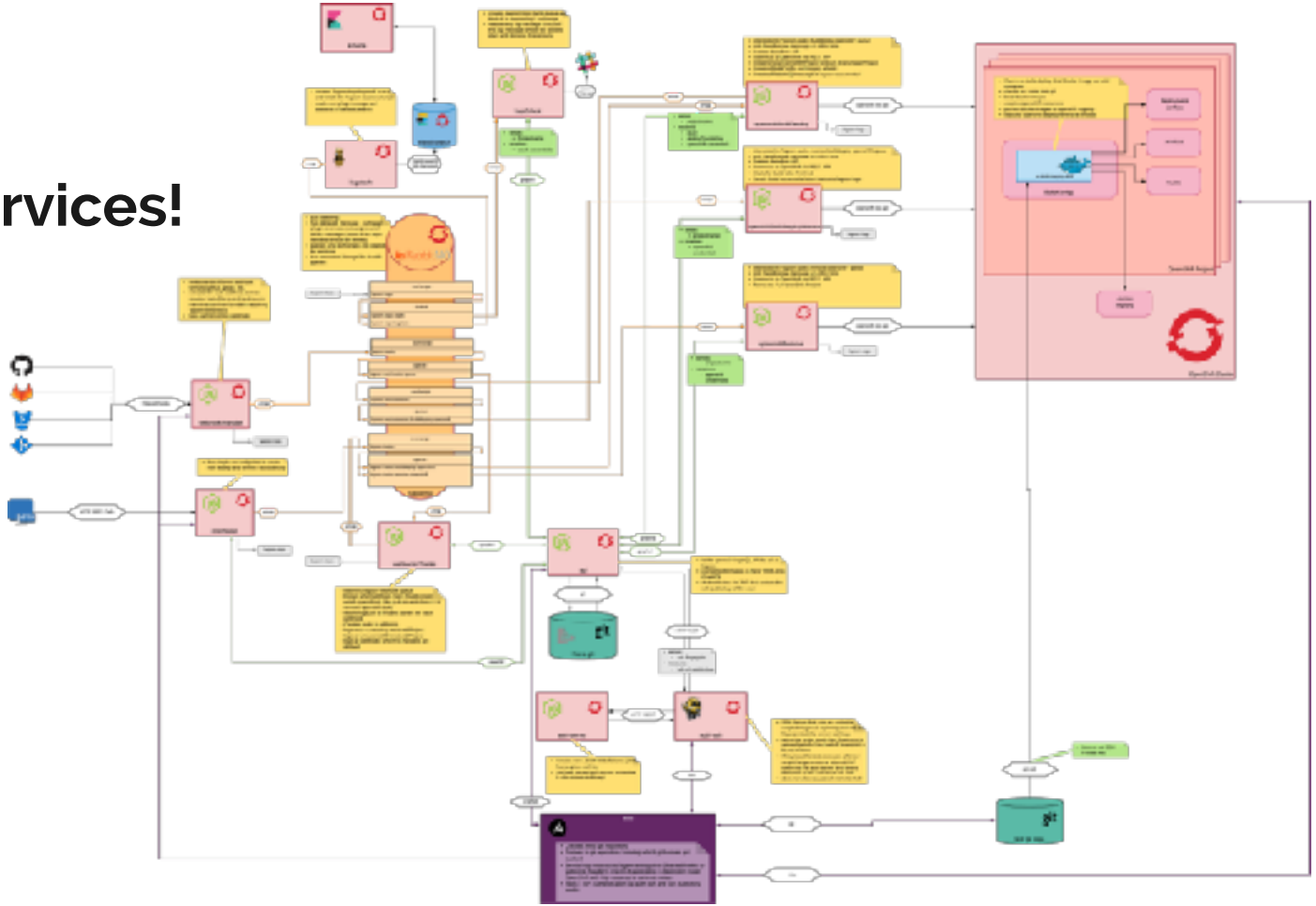
- 4. Iteration of our Hosting Stack
- Microservices
- Deployment Pipeline for ~~Drupal~~ Web Projects
- Local Development Environment
- Infrastructure/Services Defined in Code

TL;DR

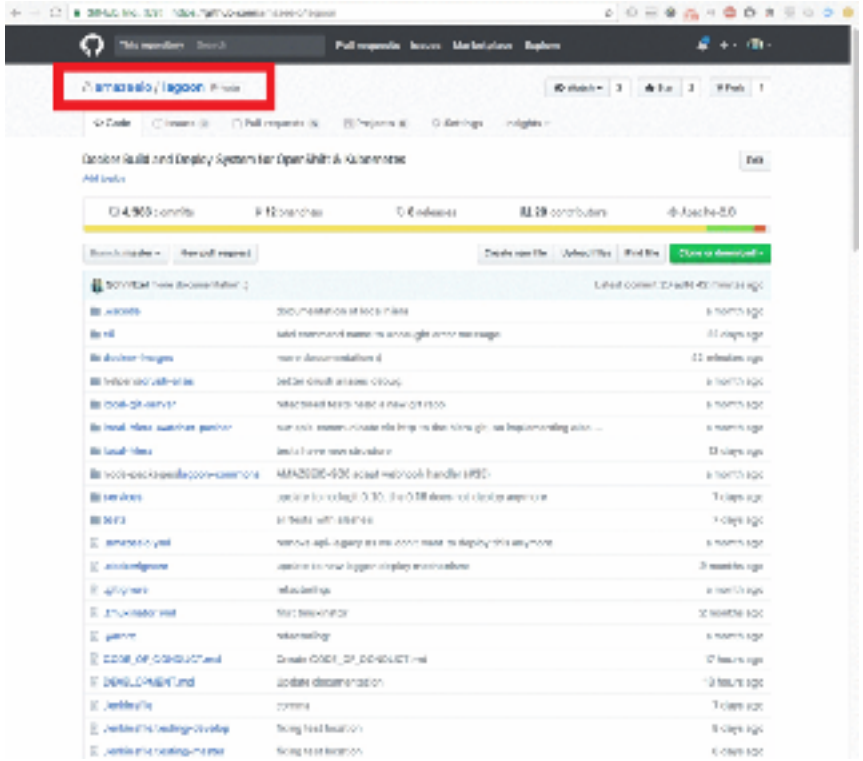
More details



Microservices!



And yes! It's all opensource!



COMPLEXITY

Complexity of orchestrated services

- If you don’t master traditional infrastructure - You will have a hard time
- Orchestrated micro-services are a living thing
- You still need skilled people that know how to engineer on top of cloud services
- (and you don’t get rid of servers - you just treat them differently)

Serverless?

Spoiler alert: Serverless still involves servers!

Serverless

A Serverless solution is one that costs you nothing to run if nobody is using it (excluding data storage)

Paul Johnston : https://medium.com/@PaulDJohnston/a-simple-definition-of-serverless-8492adfb175a

Functions as a Service
vs.
Serverless

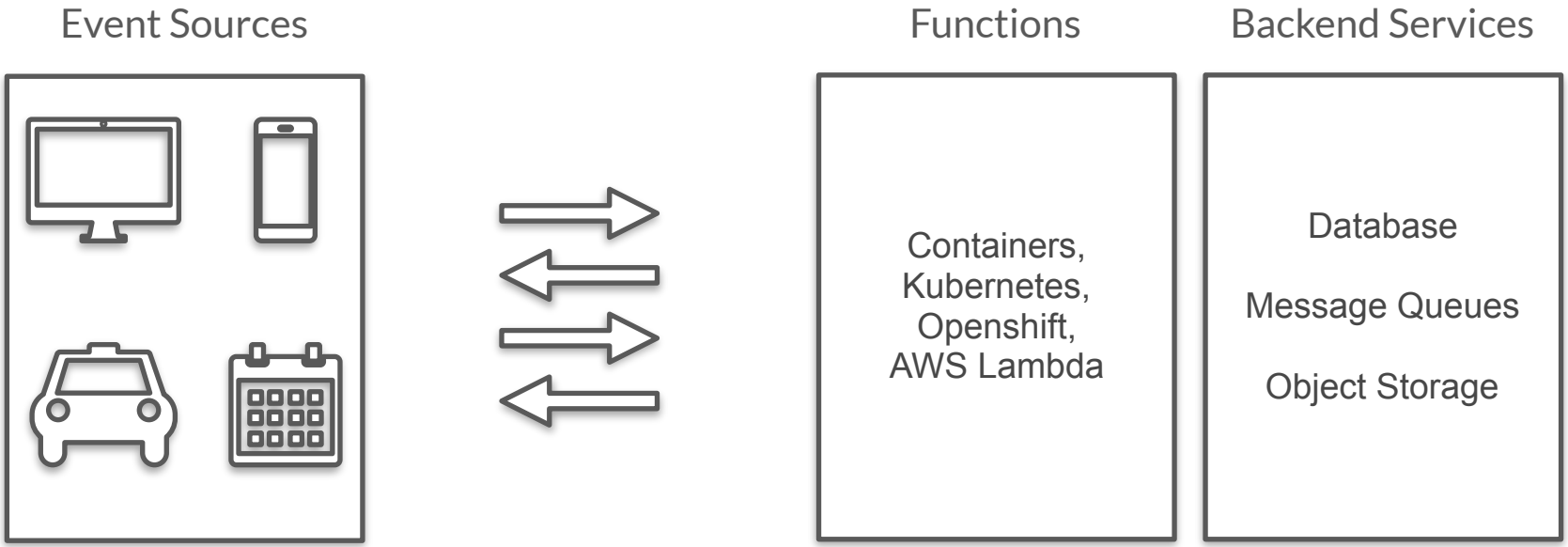
Functions as a Service (FaaS)

- Concept
- You write the code
- Event-based execution
- Stateless
- Short-lived (Milliseconds — Minutes)
- Pay per use
- Uses a compute service to run the code (no servers)

Functions as a Service (FaaS)

- Deployscript triggers a function saving the logs to an Object Storage,
- Someone calling our Emergency phone - Triggers a function and forwards the Call

Functions as a Service (FaaS)

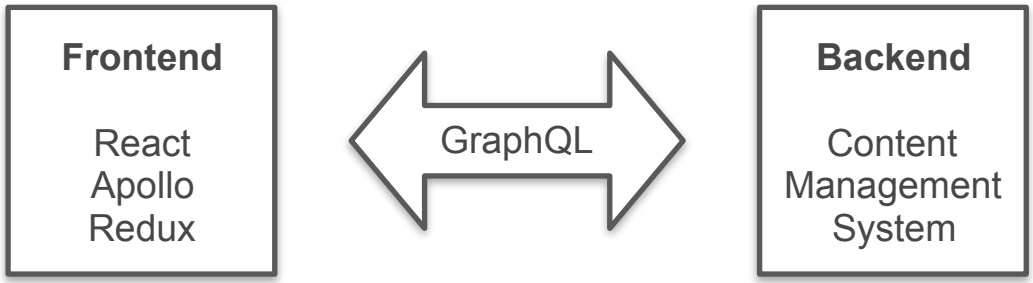


Serverless

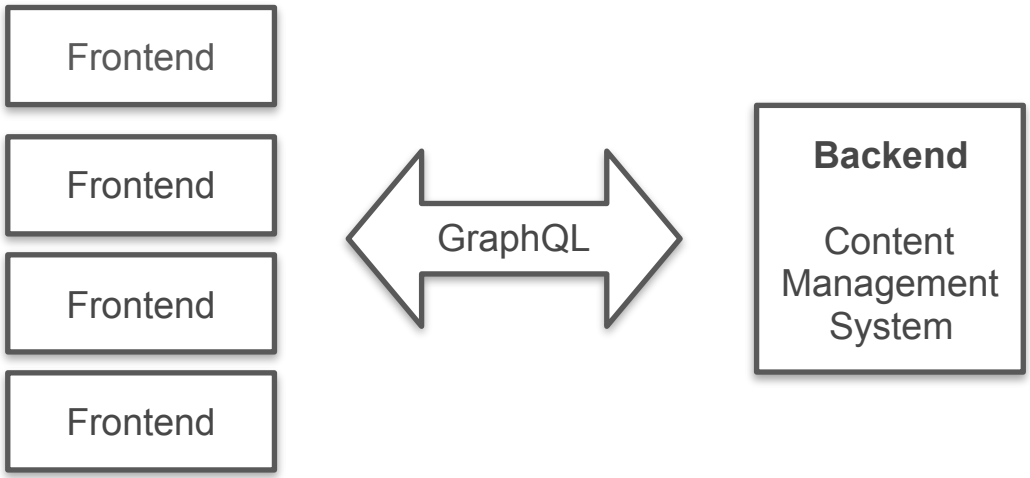
FaaS but enriched by:

- Auto-scaling based on demand
- Scaling down to zero running instances when it’s not used
- You don’t need to worry about memory or cpu usage
- Embrace third-party services
- Loosely coupled

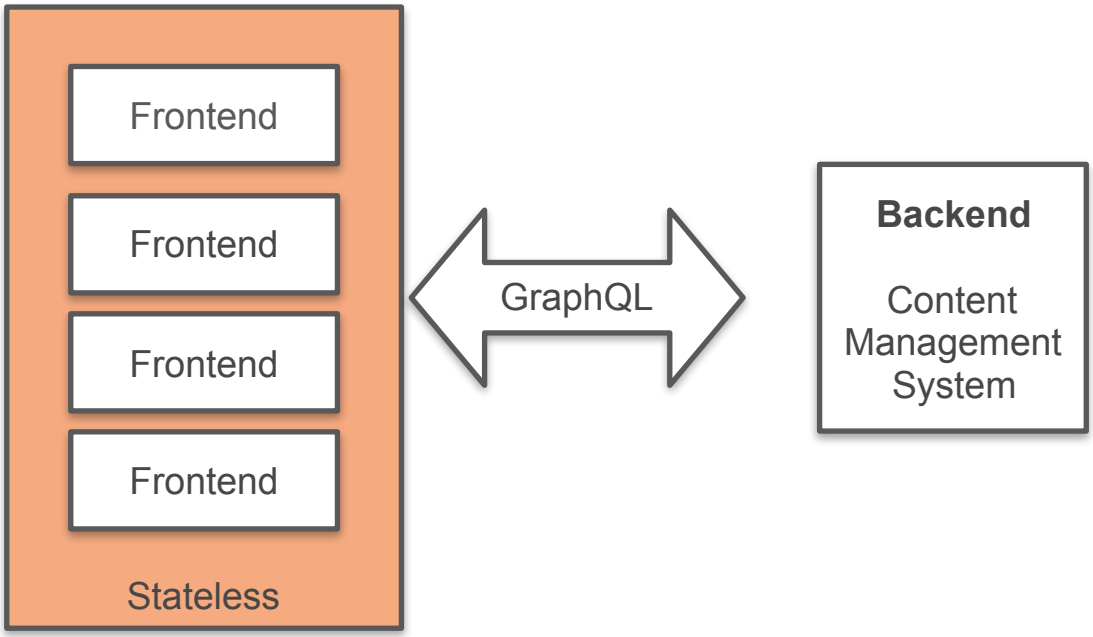
Loose Coupling / Decoupling



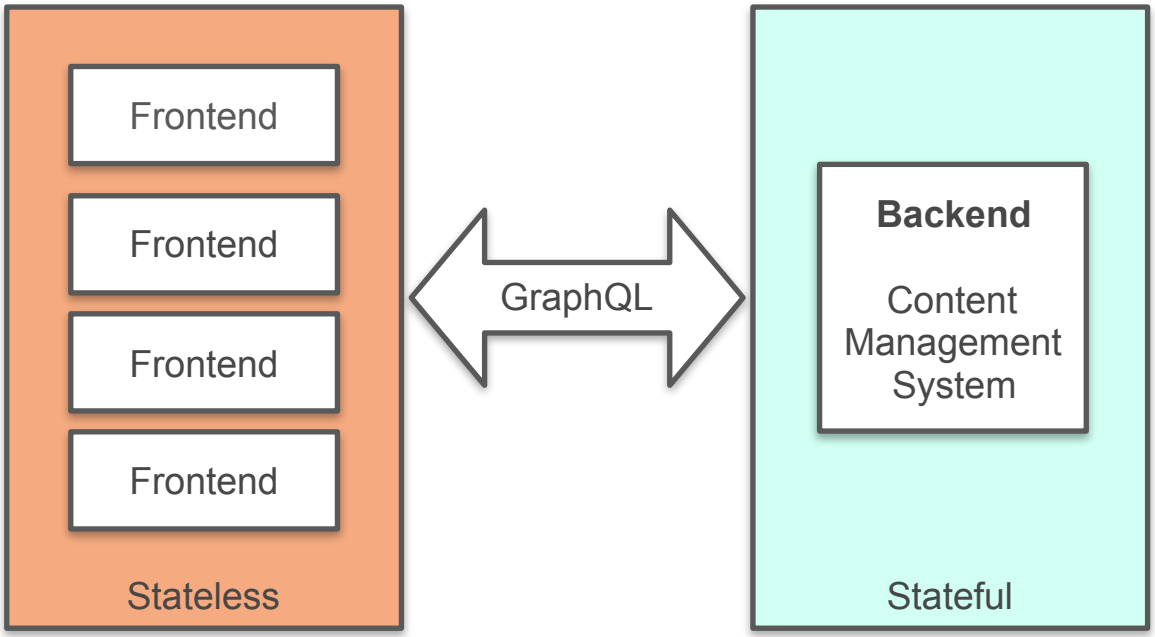
Loose Coupling / Decoupling



Loose Coupling / Decoupling



Loose Coupling / Decoupling



Serverless-ish

As soon as you have traditional stateful applications you will not have a serverless application.

We host websites with databases. So much for serverless?

Let’s call it Serverless-*ish*

Serverless-ish

But, there are no running costs beside the storage if you don’t use the application. We remove the containers.

And spin them back up if there’s demand for it.

We are on our way. It’s a journey after all.

FUTURE

In the cloud, traditional concepts don't hold up anymore.

Don’t get attached to your infrastructure!

Don't get attached to your infrastructure!
Don't give your servers names!

Don't get attached to your infrastructure!
Don't give your servers names!
Never!

Don't get attached to your infrastructure!
Don't give your servers names!
Never!

Seriously...Don't!

But why?

Pets vs. Cattle Metaphor

Pets

- Sometimes manually built
- Have names „webserver“, „billing“
- Are managed with care
- If they fail people are sad
- Think about it like your office coffee machine

Cattle/Herd

- Built via automation
- web01, web02, web03, web04
- Managed automatically
- Self-healing / orchestrated
- If they fail, they get replaced

Monitoring?
Uptime?

Monitoring

- Is my service reachable?
- Is the usage pattern within set boundaries (response time, response codes)
- Anomalies? Act on out-of-band errors
- Negative Uptime - Is the container/function running for too long?

Cloud Native - Adopting the new mindset

Cloud Native - Built for leveraging Cloud Services

- Need Block Storage - API CALL: I need 500GB SSD storage with 5000 IOPS
- Saving Objects - API CALL: store my files
- Too much load - API CALL: I need 2 Compute nodes with 96GB of RAM and 16 CPUs
- Maintenance? - Create a new compute node, schedule the containers/functions there, toss away the old machine (cattle vs pets)

To sum it up: How do you get started?

How do you get started?

- Automate, Experiment, Learn
- Embrace third-party services
- Don’t get attached to your infrastructure (Cloud-Computing isn’t a one time thing)
- Make small steps (e.g. break up your application in several services)
- Start with the parts that don’t hurt you if they fail
- You will fail and learn - that’s normal
- Build on opensource and if you can, also opensource

Thank you for your attention!

Bastian Widmer - @dasrecht | @amazeeio

Resources

- WS_FTP Pro Screenshot - https://www.cnet.com/products/ws-ftp-pro-7-5/review/
- IE Screenshot - https://guidebookgallery.org/splashes/internetexplorer
- Serverless https://martinfowler.com/articles/serverless.html
- Cattles and Pets - https://twitter.com/noggin143/status/354666097691205633
- Cattles and Pets - http://cloudscaling.com/blog/cloud-computing/the-history-of-pets-vs-cattle/
- The Power of Serverless - https://thepowerofserverless.info/