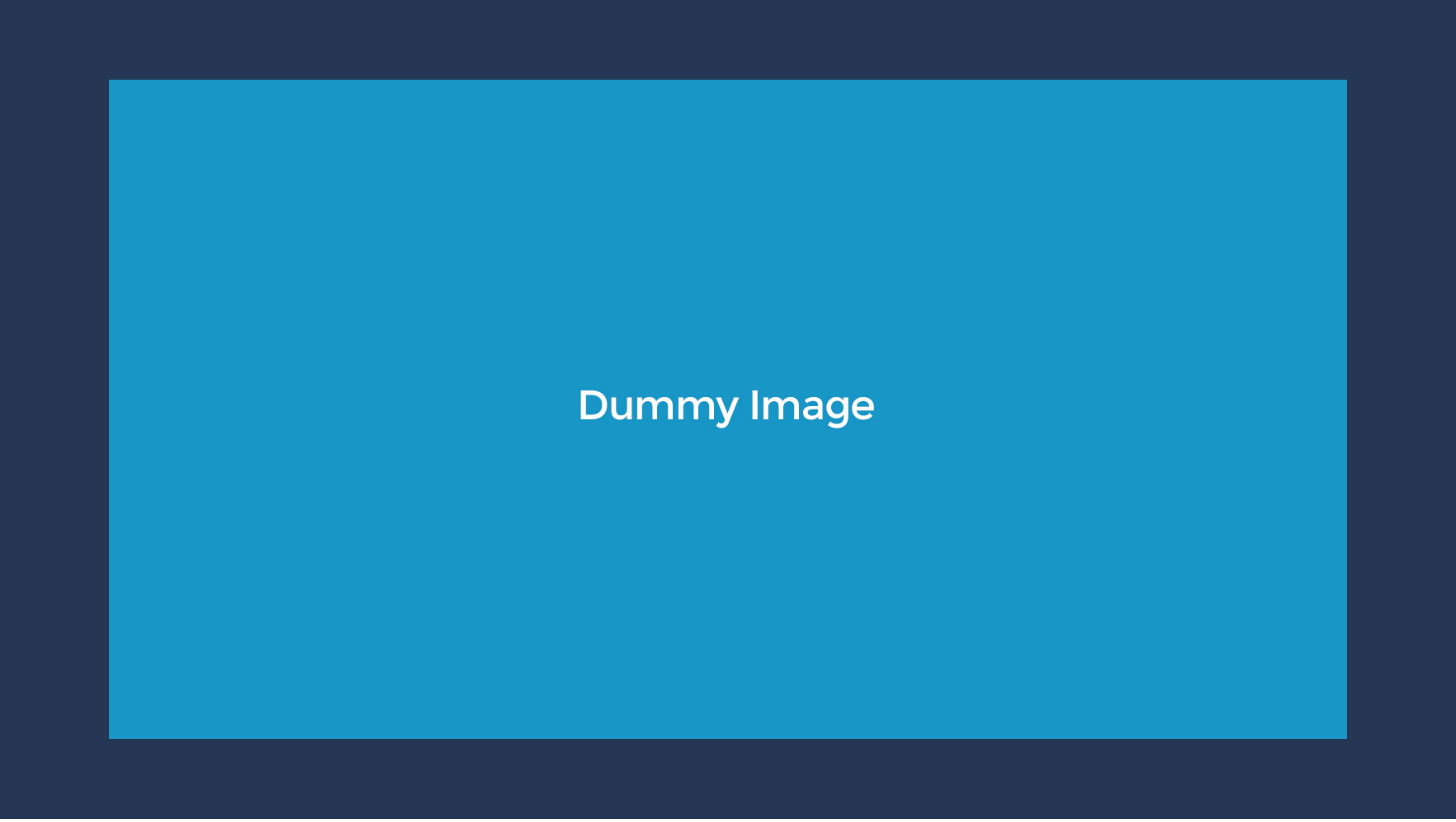
Project Presentation Stage 1

Stefanie Ziltener, Marc Heimgartner, Benjamin Bürgisser, Simon Tännler

Advanced Software Engineering FS 2017, University of Zürich



High-level Overview: Client-Server

^ we will go through each of these boxes

Containerized Microservices deployed through Kubernetes

- Components embedded in (Docker) Containers
- Containers have (ideally) one single responsibility

Containers as the instantiation of a Microservice
^ We've seen the benefits of Containers in the lecture

Frontent and API

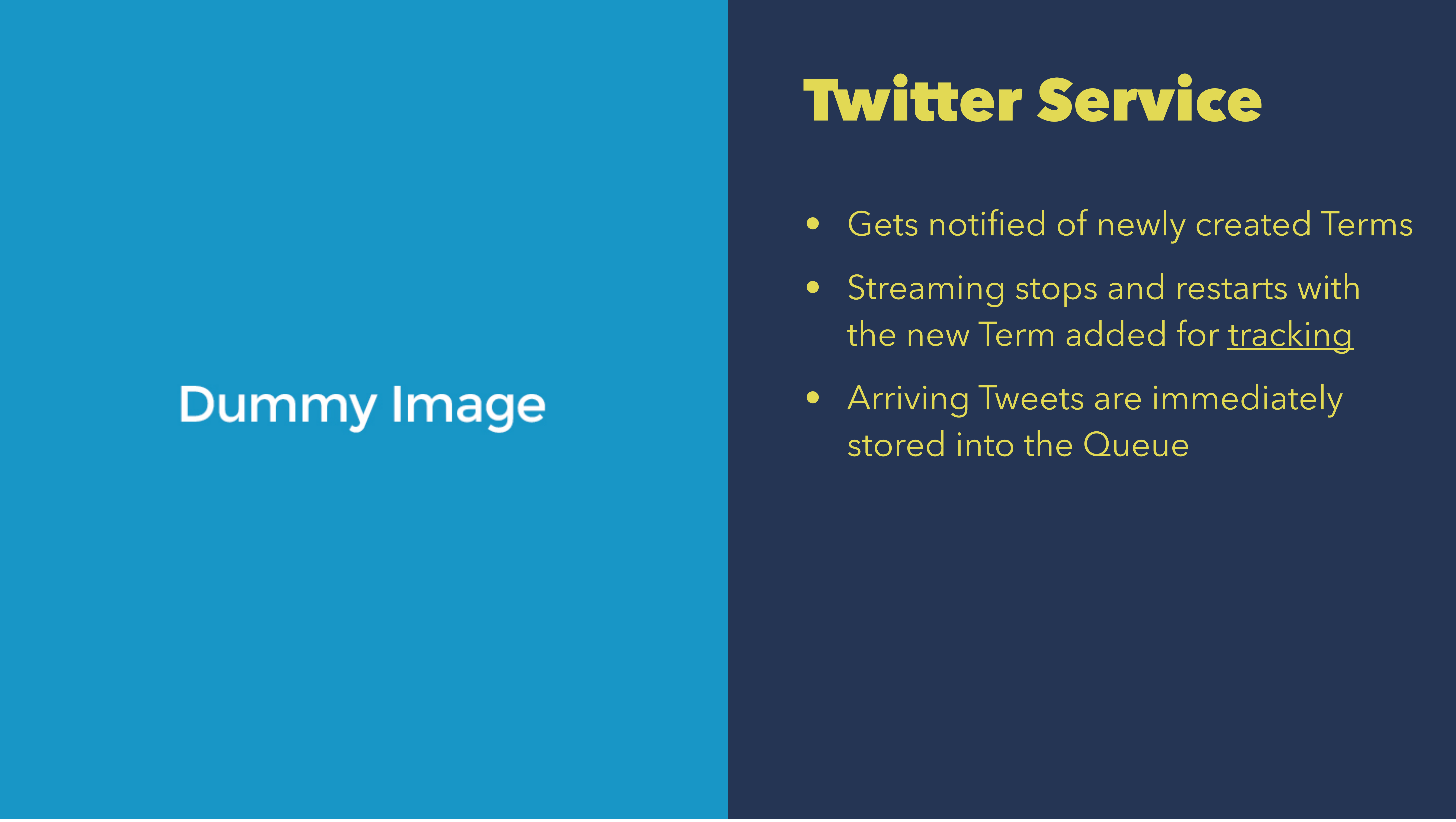
- User inputs term
- Click on register
- Send request to API

Explain the Architecture by
^ following the Story of a Request through
our Architecture

Timeseries: MongoDB (I)

- Request Handler stores Term in MongoDB
- Persistency guaranteed by GCE Persistence Disk

TODO: Img of newly created Term in JSON



Twitter Service

- Gets notified of newly created Terms
- Streaming stops and restarts with the new Term added for tracking
- Arriving Tweets are immediately stored into the Queue

Restarts are due to Streaming API limitations
^ We're trying to keep this a light as possible
^ b/c we have some limitation which we'll talk about later

Worker Queue

- Redis: in-memory data structure store
- A FIFO queue of Strings (Tweets)
- Load generator
    - API endpoint to add Strings to Queue directly

Compute Workers

- Running Workers process the Queue:
  - Assign Tweet to Term (filtering)
  - Calculate Sentiment
- Length of the Queue defined the number of Workers
  - Scaled through Kubernetes.

Twitter API does not tell you which term the tweet matched on

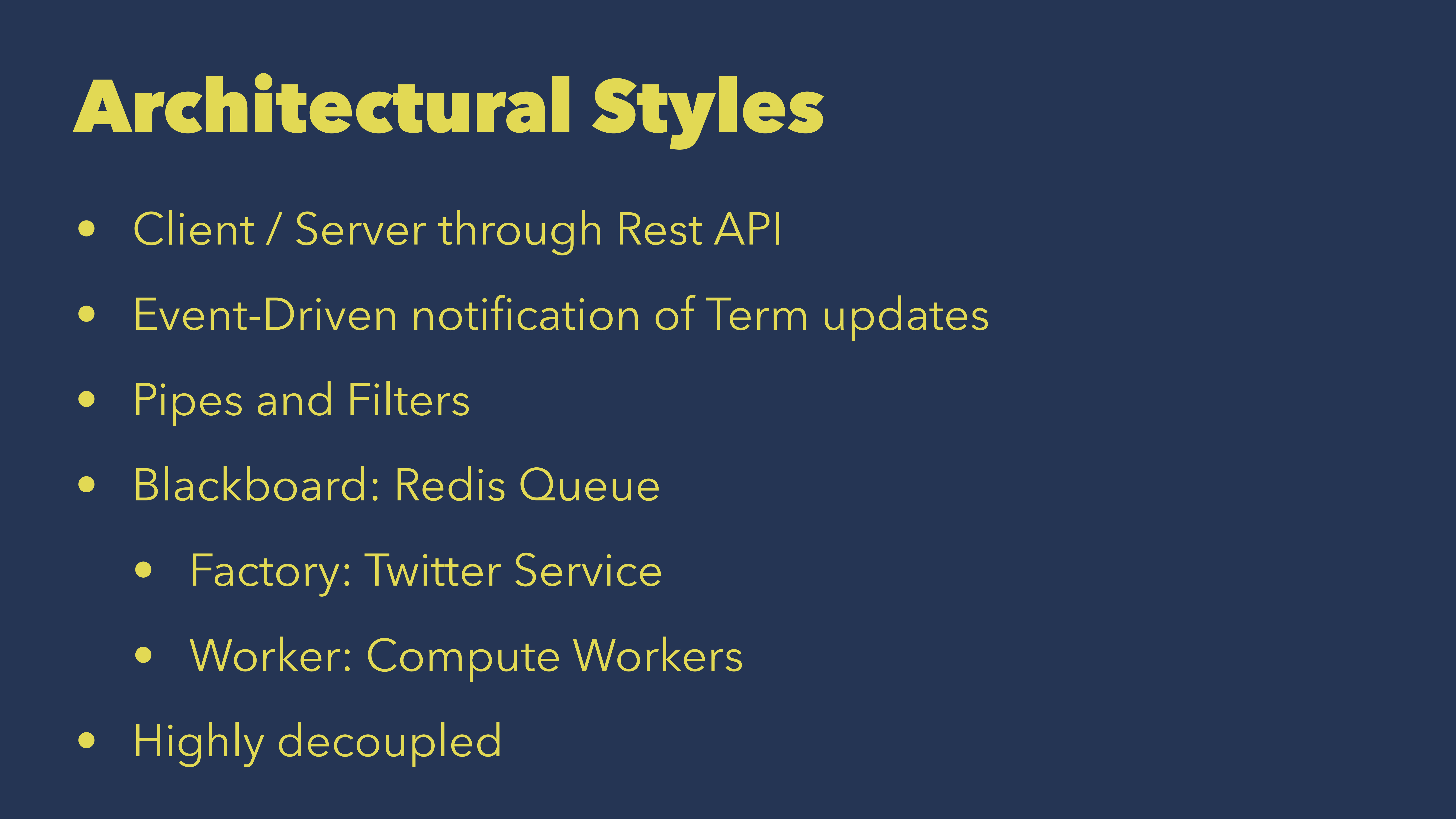
Timeseries: MongoDB (II)

- Workers store the calculated Sentiment into MongoDB.

The amount of data is actually very small
^ TODO: Img of newly created Term in JSON

Displaying Results

- Rest API gets request
- Collect relevant data from MongoDB
- Browser renders data
- Socket gets opened for continues pushes

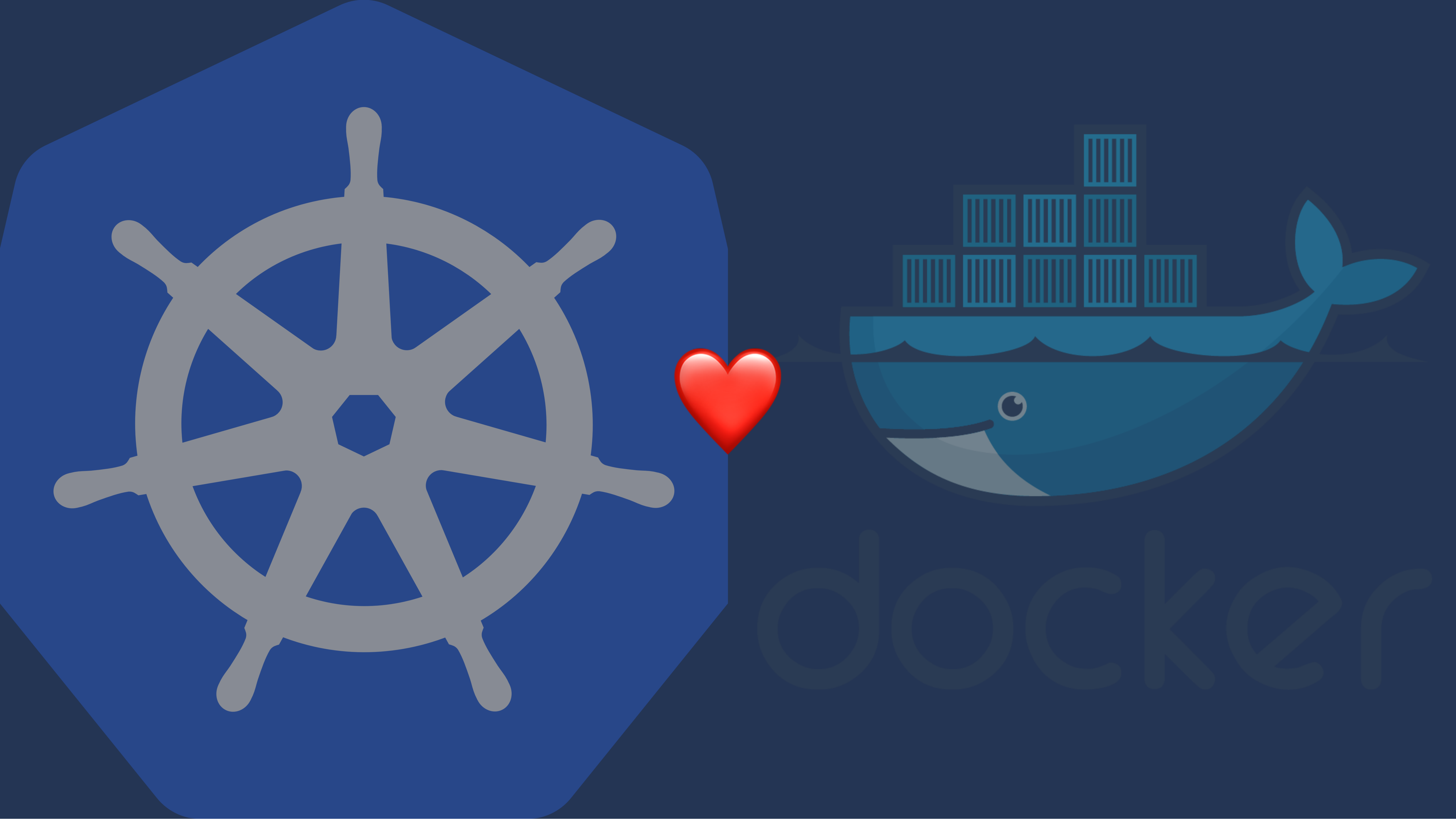


Architectural Styles

- Client / Server through Rest API
- Event-Driven notification of Term updates
- Pipes and Filters
- Blackboard: Redis Queue
  - Factory: Twitter Service
  - Worker: Compute Workers
- Highly decoupled

TODO: this feels like it needs more work

Do you even scale?


docker

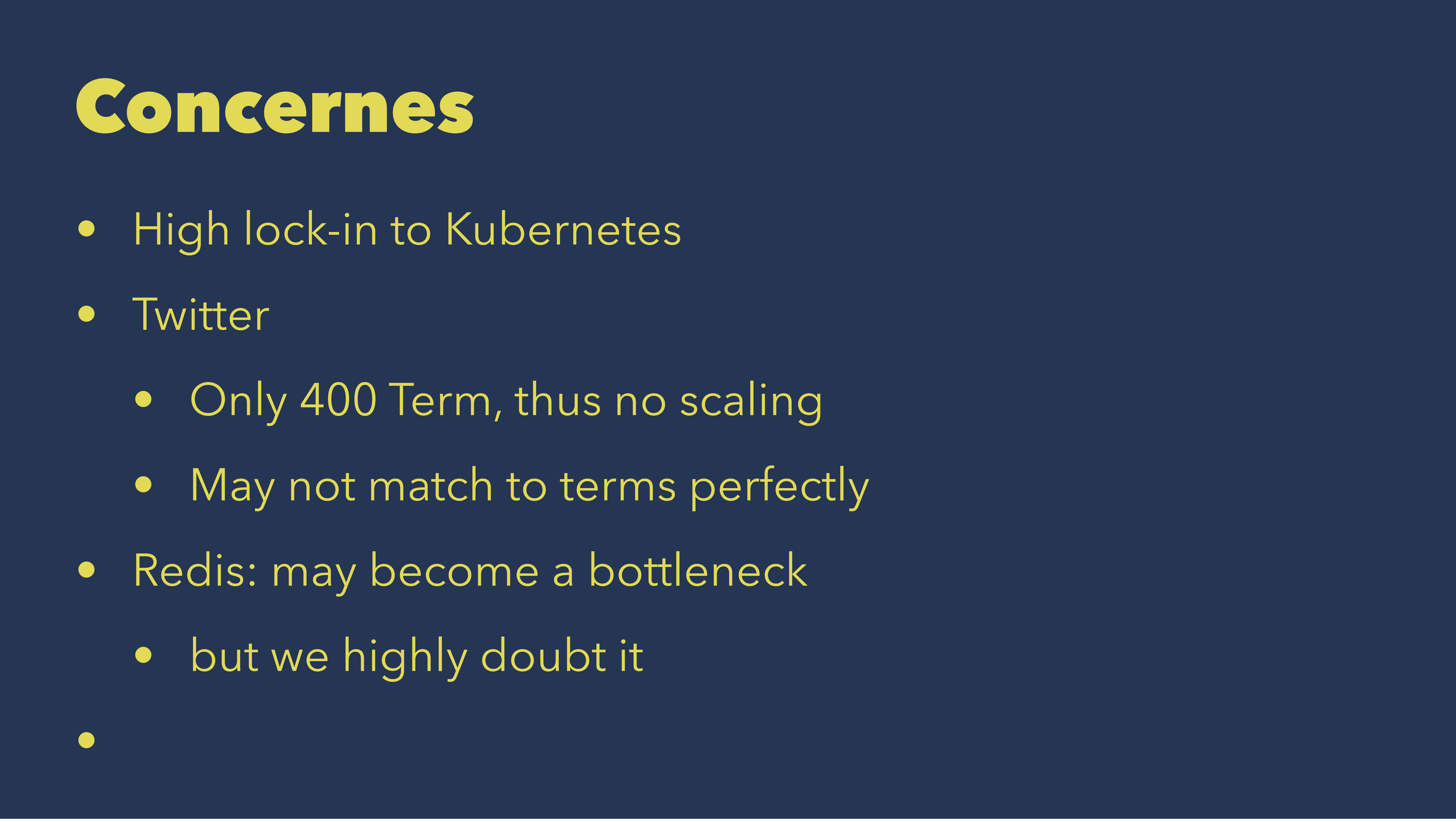
Kubernetes in one slide

@marc chasch ächt du das no mache?

Containers embedded in Pods

What does Kubernetes do for us

- Every component is potentially scalable through Kubernetes
  - Even MongoDB!
- Fault Tolerance:
  - Container recovery through Kubernetes
  - Decoupled design and Microservice
- Elasticity
  - Container scaling through Kubernetes



Concernes

- High lock-in to Kubernetes
- Twitter
  - Only 400 Term, thus no scaling
  - May not match to terms perfectly
- Redis: may become a bottleneck
  - but we highly doubt it
-

TODO: this feels like it needs more work

Technology Zoo - Platform

- Cloud Platform: Google Container Engine (GCE)
  - Easy support of Kubernetes
- Containerization: Docker
  - Popular Container engine
- Container orchestration: Kubernetes
  - Popular Container orchestration

Technology Zoo - Backend

- Programming Language: Google Go
  - New Programming language 🎉
  - Uniquely suited for Web development
  - Have I mentioned it's fast?
- Terms Storage: MongoDB
  - Easy data schema
- Queue Storage: Redis

Technology Zoo - Frontend

- Frontend: ViewerJS
  - Similar to Angular
- Webserver: nginx
  - Battle-proven Webserver

Dev Environment

Docker containers using Docker Compose

Demo

Questions?