ALL DAY DEVOPS

NOVEMBER 6, 2019

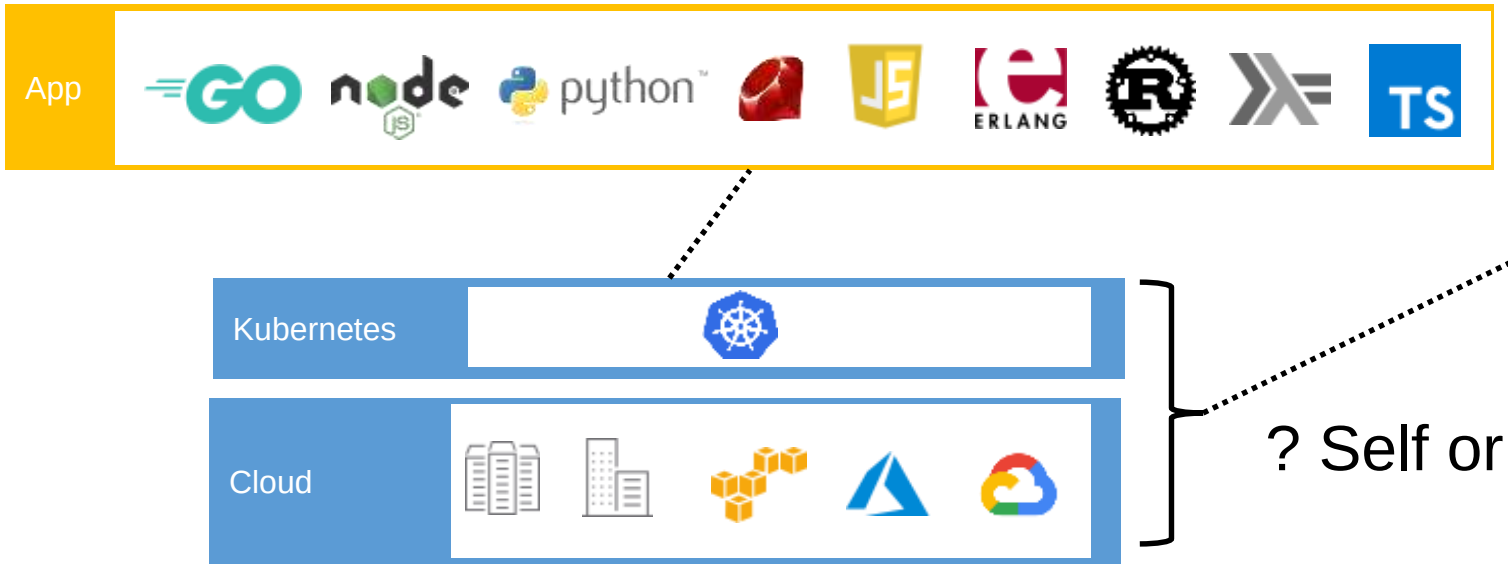
Creating a stateful application with K8S and AWS DB Services

Bahubali Shetti
@Shetti
Director of Cloud Developer Advocacy VMware

Maximizing portability with Kubernetes – BUT what about the Data?

Not easy to move

Stateless



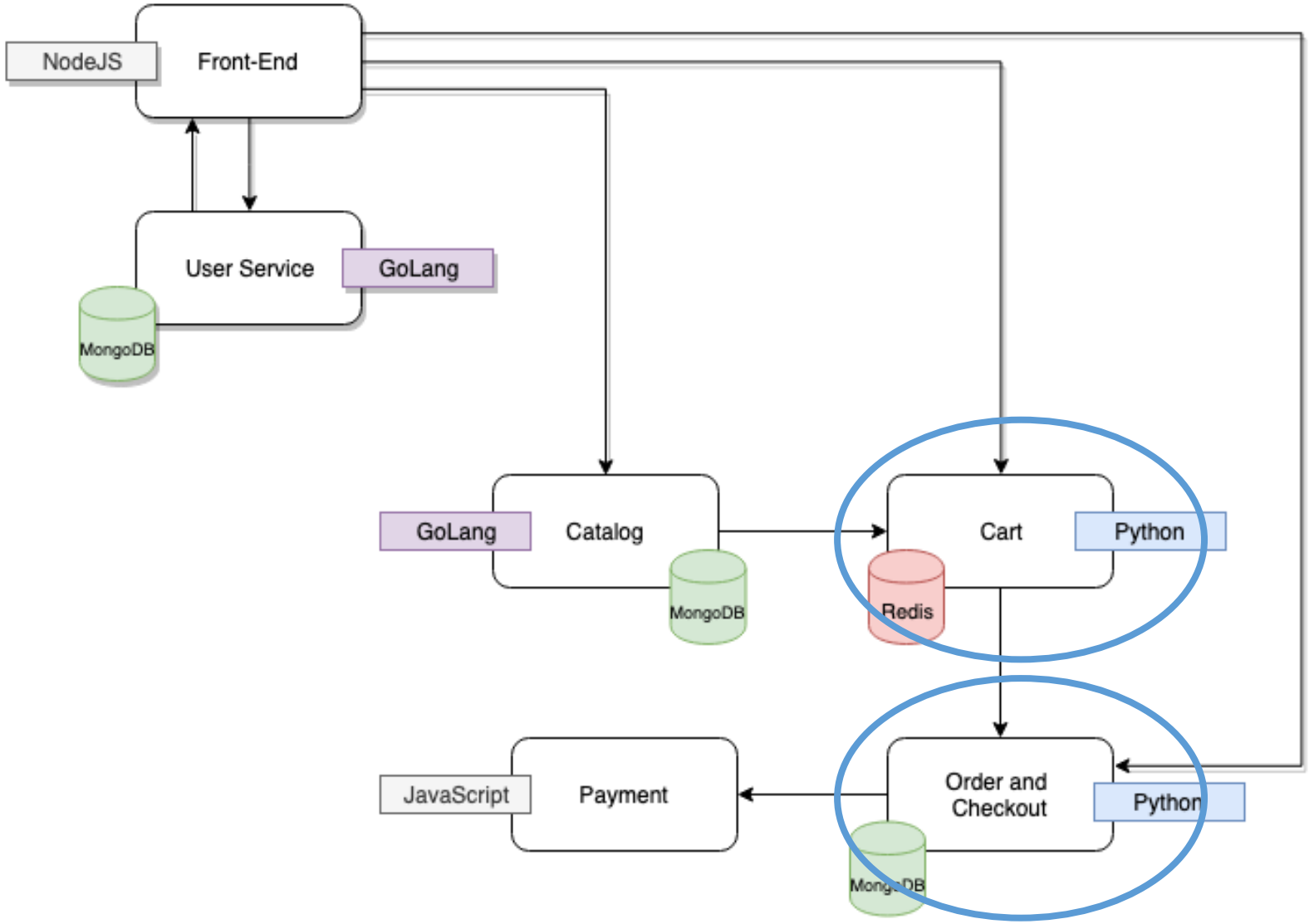
Application Data
(Application State)

? Self or Fully Manage

Self Managed vs Fully Managed….

Typical Microservice App - AcmeShop

Find it at cloudjourney.io (look for the github site)



Code Instrumentation for App-DB connectivity

Parameterizing the database end points in K8S

K8S Yaml

```
...
…
 spec:
     volumes:
     - name: acmefit-order-data
       emptyDir: {}
     containers:
     - image: order:latest
       name: order
       env:
       - name: ORDER_DB_HOST
         value: 'order-mongo'
       - name: ORDER_DB_PASSWORD
         valueFrom:
           secretKeyRef:
             name: order-mongo-pass
             key: password
       - name: ORDER_DB_PORT
         value: '27017’
…
…
```

Dockerfile

```
FROM bitnami/python:3.7
MAINTAINER Bill Shetti
"billshetti@gmail.com"
ENV ORDER_DB_HOST="localhost"
ENV ORDER_DB_PORT="27017"
ENV ORDER_DB_PASSWORD=""
ENV ORDER_DB_USERNAME=""
ENV PAYMENT_HOST="localhost"
ENV PAYMENT_PORT="9000"

# needed for mongo client
RUN install_packages mongodb-
clients

COPY ./requirements.txt
/app/requirements.txt
RUN pip3 install -r requirements.txt

…
…
```

Python order.py code

```
…
from os import environ

if environ.get('ORDER_DB_USERNAME') is not
None:
   if os.environ['ORDER_DB_USERNAME'] !=
"":

mongouser=os.environ['ORDER_DB_USERNA
ME']
   else:
      mongouser=''
else:
   mongouser=''

if environ.get('ORDER_DB_HOST') is not None:
   if os.environ['ORDER_DB_HOST'] != "":

mongohost=os.environ['ORDER_DB_HOST']
   else:
      mongohost='localhost'
else:
   mongohost='localhost’
…
…
```

Code Instrumentation for DB

Libraries and connecting to the DB

import pymongo

```
from pymongo import MongoClient
from pymongo import errors as mongoerrors

client=MongoClient(mongouri)
#uri=username:password@host:port
Or
client=MongoClient(host=mongohost, port=int(mongoport),
username=mongouser, password=mongopassword)
```

import redis

```
rConn=redis.StrictRedis(host=redishost, port=redisport,
password=redispassword, db=0)
```

Lots of standard libraries (go, python, etc) with significant support

Setting up your own containerized DB

Installation and management – Several options

Simple K8S Create

Initialization

kubectl apply -f config-map.yaml

Secrets

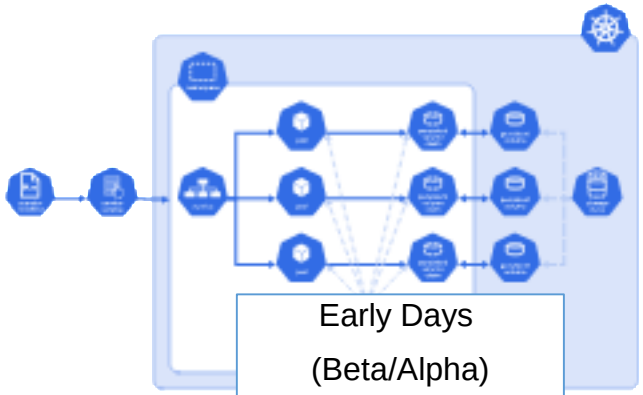
kubectl create secret generic order-mongo-pass --from-literal=password=<value>

Create

kubectl apply -f order-db-total.yaml

(ISH)

Operators



Early Days

(Beta/Alpha)

(still adding features)

etc

Automates c ngle node or replica sets for mongodb

Enables set up of alerting, monitoring

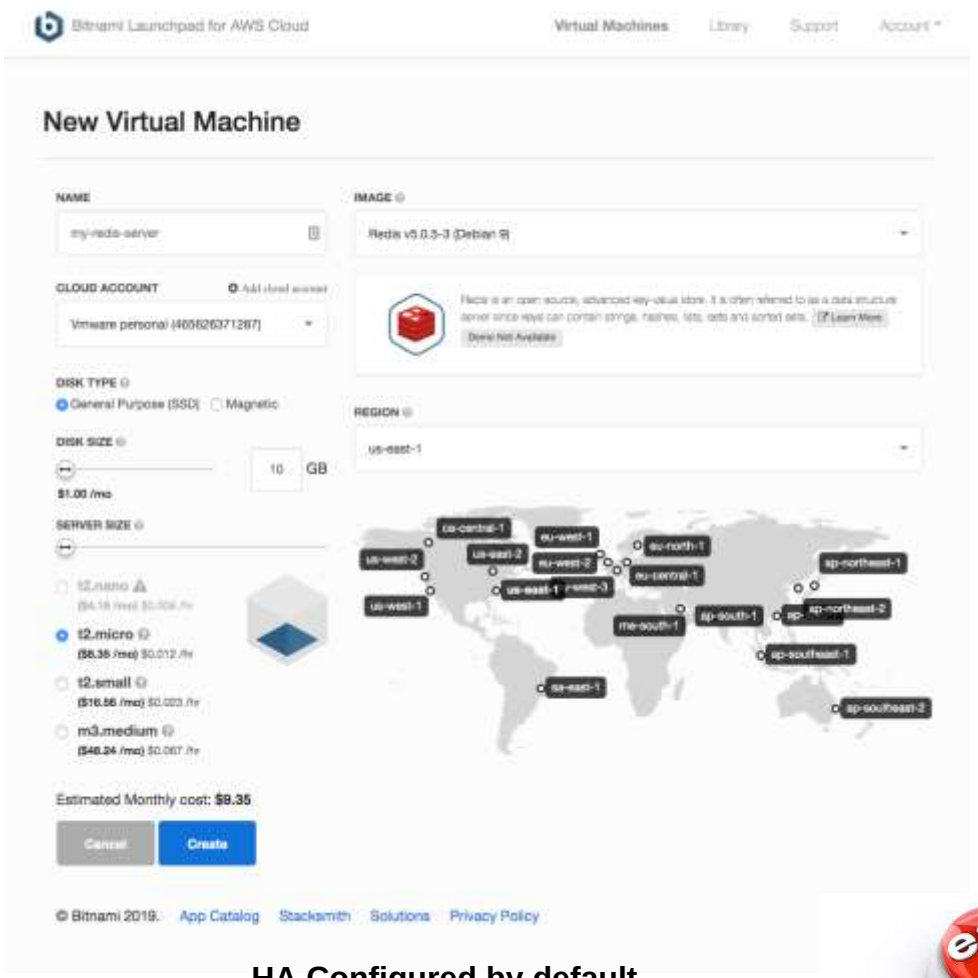
Optional persistence and storage configuration

Easy scale

(ISH)

Setting up your own containerized DB

Installation and management – Several options



HA Configured by default

bitnami

aws

Microsoft Azure

Google Cloud

Setting up your own containerized DB

So what's hard? – Keeping state persistent

Storage Provisioning and Management

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
        name: fast
provisioner: kubernetes.io/aws-ebs
parameters:
        type: io1
```

```
containers:
  - name: mongo
    image: mongo
    ports:
      - containerPort: 27017
    volumeMounts:
      - name: mongo-persistent-storage
        mountPath: /data/db
volumeClaimTemplates:
- metadata:
name: mongo-persistent-storage
annotations:
  volume.beta.kubernetes.io/storage-class: "fast"
spec:
accessModes: ["ReadWriteOnce"]
resources:
  requests:
    storage: 100Gi
```

What's important to you?

Infra and DB expertise or App and business focus?

Infra Skills

App Skills

ADDO
ALL DAY DEVOPS

Fully Managed Databases

AWS Options

Database type	Use cases	AWS service
Relational	Traditional applications, ERP, CRM, e-commerce	Amazon Aurora, Amazon RDS, Amazon Redshift
Key-value	High-traffic web apps, e-commerce systems, gaming applications	Amazon DynamoDB
In-memory	Caching, session management, gaming leaderboards, geospatial applications	Amazon ElastiCache for Memcached, Amazon ElastiCache for Redis
Document	Content management, catalogs, user profiles	Amazon DocumentDB
Graph	Fraud detection, social networking, recommendation engines	Amazon Neptune
Time series	IoT applications, DevOps, industrial telemetry	Amazon Timestream
Ledger	Systems of record, supply chain, registrations, banking transactions	Amazon QLDB

Managed Databases

Redis and AWS Elasticache

Simple to implement

Creating HA arch – operational overhead

No support for sharding

No encryption

Operationally Expensive

K8S Operator

VS

Simple to implement

Built in HA with read replicas, multiple primaries, failovers, etc

Easly scalable

Sharding support

Encryption at rest/intransit

√

Managed Databases

DocumentDB and MongoDB Atlas

Built in sharding & replica sets for easy scaling
Still have to manually add nodes
Manage backup manually or with tools (OpsManager, CloudManager etc)
Manage Upgrades etc

K8S Operator

VS

Easy setup (only compatible with MongoDB 3.6)
Managed sharding, replicas
Managed scale (up to 64TB)
Easy backups - AWS
Handles 100ks reads/writes/sec

√

Easy setup – deploys on AWS/Azure/GCP
Managed sharding, replicas
Managed scale
Easy backups – AWS/Azure/GCP
Pure Mongo experience with latest and greatest features

√

Using a managedDB

Keeping state persistent?

Storage Provisi… …d Management

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
        name: fast
provisioner: kubernetes.io/aws-ebs
parameters:
        type: io1
```

```
cont…
  - name: …
    image: mo…
    ports:
      - containerPo… …017
    volumeMounts:
      - name: mongo-p… …stent-storage
        mountPath: /dat…
volumeClaimTemplate…
- metadata:
name: mongo-persiste… …torage
  annotations:
    …me.beta.kuber… …s.io/storage-class: "fast"
sp…
acce… …des: ["Re… …riteOnce"]
resourc…
  requests:…
    storage: 1…
```

Using a managed DB

Installation and management –

AWS CLI with automation

aws --region us-east-2 elasticache create-cache-cluster --cache-cluster-id my-cluster --cache-node-type cache.r4.large --engine redis --engine-version 3.2.4 --num-cache-nodes 1 --cache-parameter-group default.redis3.2

aws --region us-east-2 docdb create-db-cluster --db-cluster-identifier mongoeq --engine docdb --master-username bill --master-user-password password1

OR

AWS service operator

https://github.com/awslabs/aws-service-operator

(ISH)

Code Instrumentation for DB

Parameterizing the database end points in K8S

K8S Yaml

```
...
…
 spec:
     volumes:
     - name: acmefit-order-data
       emptyDir: {}
     containers:
     - image: order:latest
       name: order
       env:
       - name: ORDER_DB_HOST
         value: 'order-mongo'
       - name: ORDER_DB_PASSWORD
         valueFrom:
           secretKeyRef:
             name: order-mongo-pass
             key: password
       - name: ORDER_DB_PORT
         value: '27017'
…
…
```

Dockerfile

```
FROM bitnami/python:3.7
MAINTAINER Bill Shetti
"billshetti@gmail.com"
ENV ORDER_DB_HOST="localhost"
ENV ORDER_DB_PORT="27017"
ENV ORDER_DB_PASSWORD=""
ENV ORDER_DB_USERNAME=""
ENV PAYMENT_HOST="localhost"
ENV PAYMENT_PORT="9000"

# needed for mongo client
RUN install_packages mongodb-
clients

COPY ./requirements.txt
/app/requirements.txt
RUN pip3 install -r requirements.txt

…
…
```

Python order.py code

```
…
from os import environ

if environ.get('ORDER_DB_USERNAME') is not
None:
   if os.environ['ORDER_DB_USERNAME'] !=
"":

mongouser=os.environ['ORDER_DB_USERNA
ME']
   else:
      mongouser=''
else:
   mongouser=''

if environ.get('ORDER_DB_HOST') is not None:
   if os.environ['ORDER_DB_HOST'] != "":

mongohost=os.environ['ORDER_DB_HOST']
   else:
      mongohost='localhost'
else:
   mongohost='localhost’
…
…
```

Insert Document DB URL INFO HERE

Using a Managed DB

Steps and hurdles – code instrumentization

import pymongo

```
from pymongo import MongoClient
from pymongo import errors as mongoerrors

client=MongoClient(mongouri)
#uri=username:password@host:port
Or
client=MongoClient(host=mongohost, port=int(mongoport),
username=mongouser, password=mongopassword)
```

import redis

```
rConn=redis.StrictRedis(host=redishost, port=redisport,
password=redispassword, db=0)
```

URL FROM AWS Services

ADDO
ALL DAY DEVOPS

Self Managed vs Fully Managed….

?

www.cloudjourney.io
@cloudjourneyio

Bahubali Shetti - @Shetti

ADDO
ALL DAY DEVOPS