



# Docker Changes the Way You Develop and Release Your Scalable Solutions

Tugberk Ugurlu
Redgate Software
@tourismgeek
http://tugberkugurlu.com

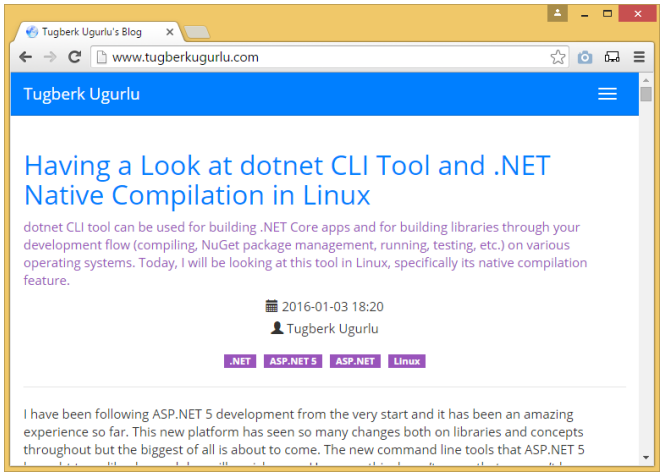
#itakeunconf

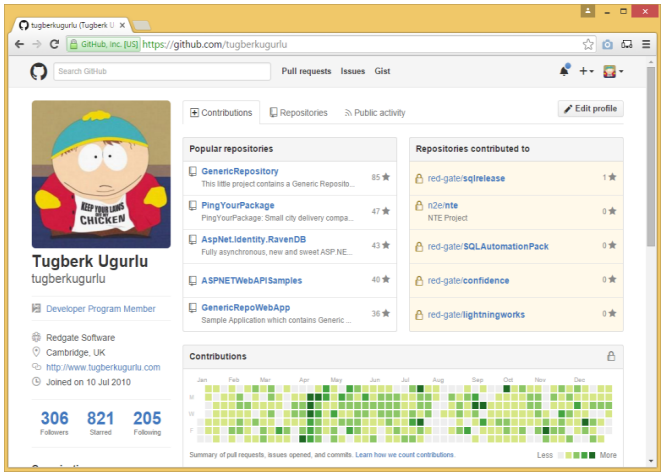

[@tourismgeek](#)

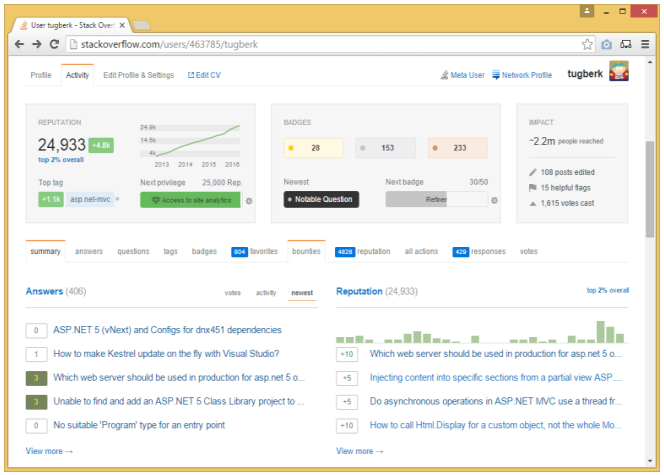
[tugberkugurlu.com](#)

[github.com/tugberkugurlu](#)

[stackoverflow.com/users/463785/tugberk](#)




TONI'S ICES
of Fortune 100 compa
use Red Gate's softv
RED GATE
DAY OUT 2014
mocca
COMIC SANS

[@redgate]

# The Problem

# Inconsistent Environments

# Result

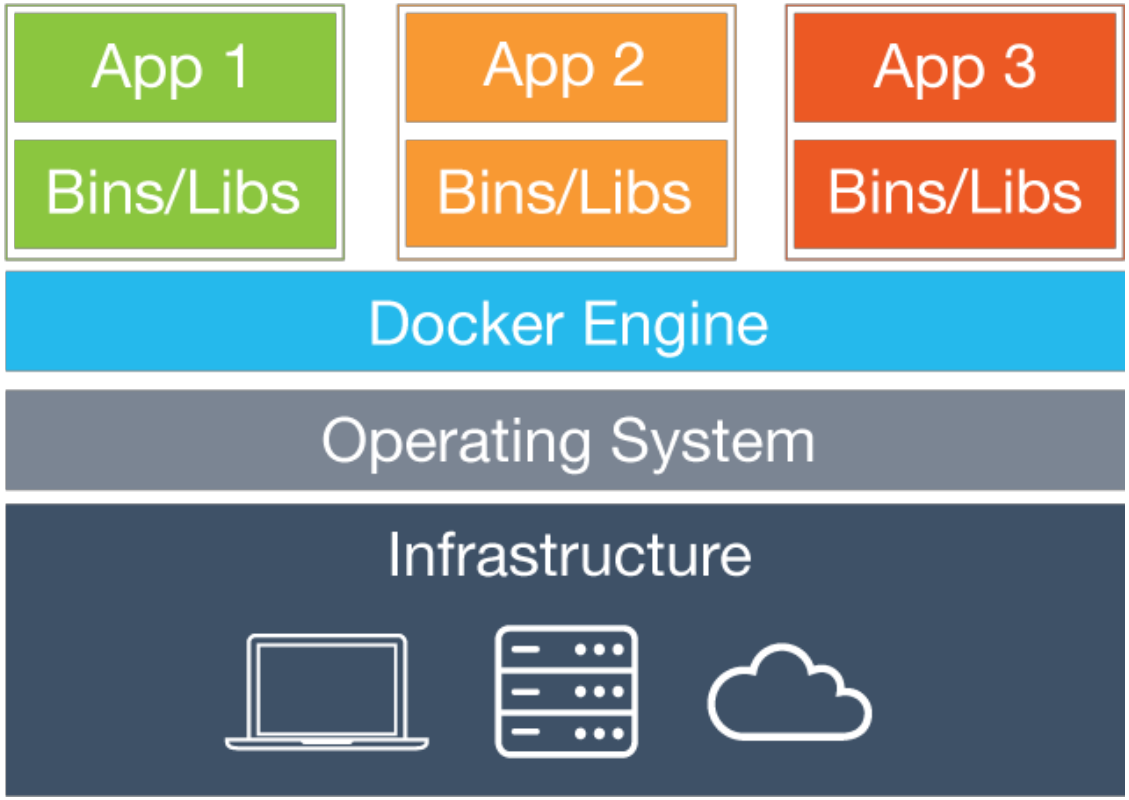
- Hard Onboarding
- Hard to Set up Different Environments
- Different Deployment Needs for Each Tech
- Hard to Reproduce Issues

# What’s Docker?

https://www.docker.com/what-docker

# What's Docker?

- Containerization
- Provides Good Isolation
- Consistent Environments



# Docker Family

- Docker
- Docker Compose
- Docker Machine
- Docker Swarm
- Docker Registry

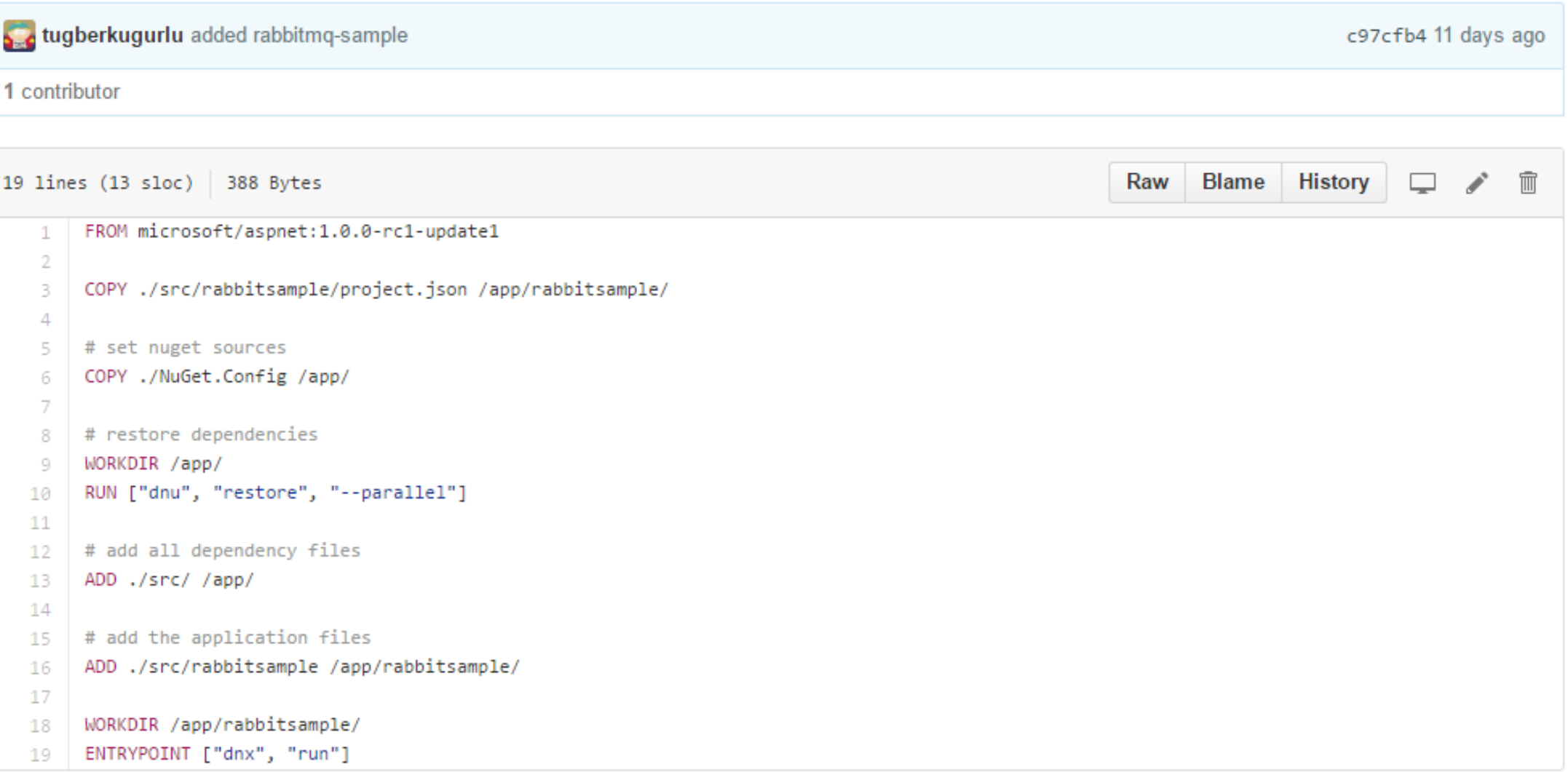
# docker run ubuntu /bin/echo 'Hello world'

https://docs.docker.com/engine/userguide/containers/dockerizing/

# Packaging Your Application

https://github.com/tugberkugurlu/AspNetCoreSamples/blob/master/rabbitmq-sample/rabbitsample.dockerfile

# $ docker build -t example-app -f rabbitsample.dockerfile .



tugberkugurlu added rabbitmq-sample c97cfb4 11 days ago

1 contributor

19 lines (13 sloc) | 388 Bytes Raw Blame History

```
1  FROM microsoft/aspnet:1.0.0-rc1-update1
2
3  COPY ./src/rabbitsample/project.json /app/rabbitsample/
4
5  # set nuget sources
6  COPY ./NuGet.Config /app/
7
8  # restore dependencies
9  WORKDIR /app/
10 RUN ["dnu", "restore", "--parallel"]
11
12 # add all dependency files
13 ADD ./src/ /app/
14
15 # add the application files
16 ADD ./src/rabbitsample /app/rabbitsample/
17
18 WORKDIR /app/rabbitsample/
19 ENTRYPOINT ["dnx", "run"]
```

# Packaging Dev Tools

https://github.com/BenHall/docker-awscli/blob/master/Dockerfile

**BenHall** Initial Dockerfile `37c40f2` on Feb 6

**1** contributor

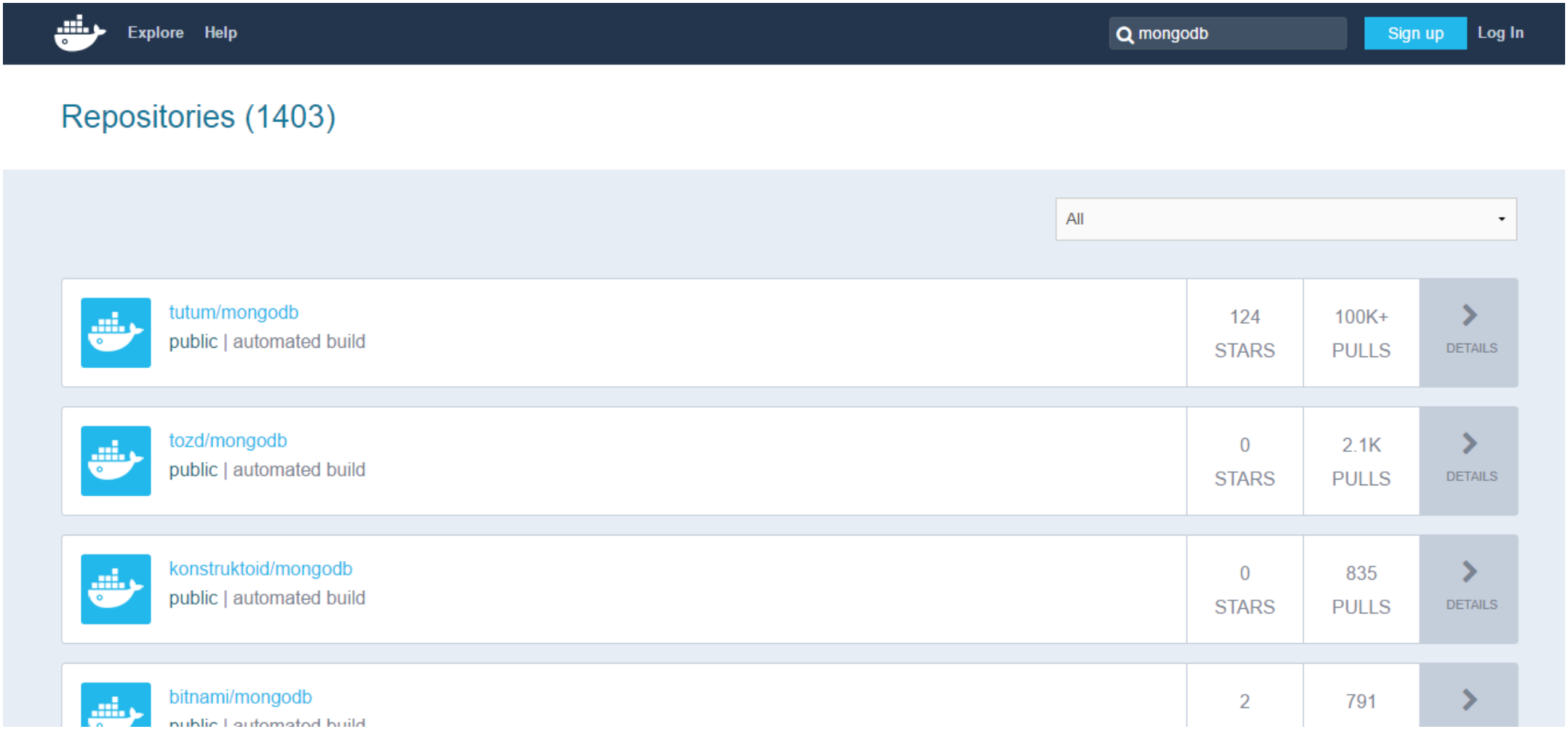
5 lines (3 sloc) | 140 Bytes

Raw | Blame | History

```
1  FROM ubuntu:14.04
2
3  RUN apt-get update -q && DEBIAN_FRONTEND=noninteractive apt-get install -qy python-pip groff-base
4  RUN pip install awscli
```

# Docker Registry

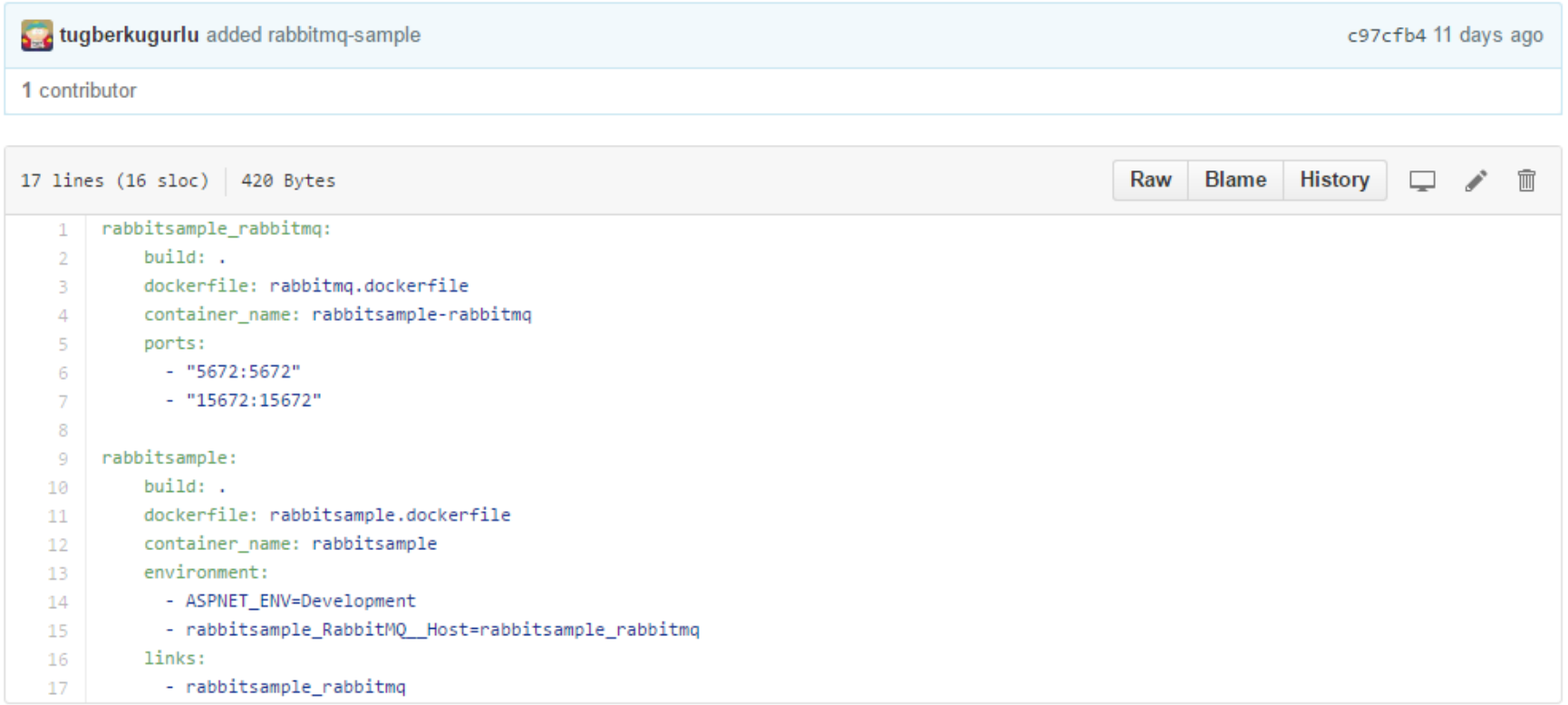
https://hub.docker.com


Explore
Help
mongodb
Sign up
Log In
Repositories (1403)
All
tutum/mongodb
public | automated build
124
STARS
100K+
PULLS
DETAILS
tozd/mongodb
public | automated build
0
STARS
2.1K
PULLS
DETAILS
konstruktoid/mongodb
public | automated build
0
STARS
835
PULLS
DETAILS
bitnami/mongodb
2
791

# Docker Compose

https://docs.docker.com/compose

# docker-compose.yml



tugberkugurlu added rabbitmq-sample c97cfb4 11 days ago

1 contributor

17 lines (16 sloc) | 420 Bytes Raw Blame History

```
rabbitsample_rabbitmq:
    build: .
    dockerfile: rabbitmq.dockerfile
    container_name: rabbitsample-rabbitmq
    ports:
      - "5672:5672"
      - "15672:15672"

rabbitsample:
    build: .
    dockerfile: rabbitsample.dockerfile
    container_name: rabbitsample
    environment:
      - ASPNET_ENV=Development
      - rabbitsample_RabbitMQ__Host=rabbitsample_rabbitmq
    links:
      - rabbitsample_rabbitmq
```

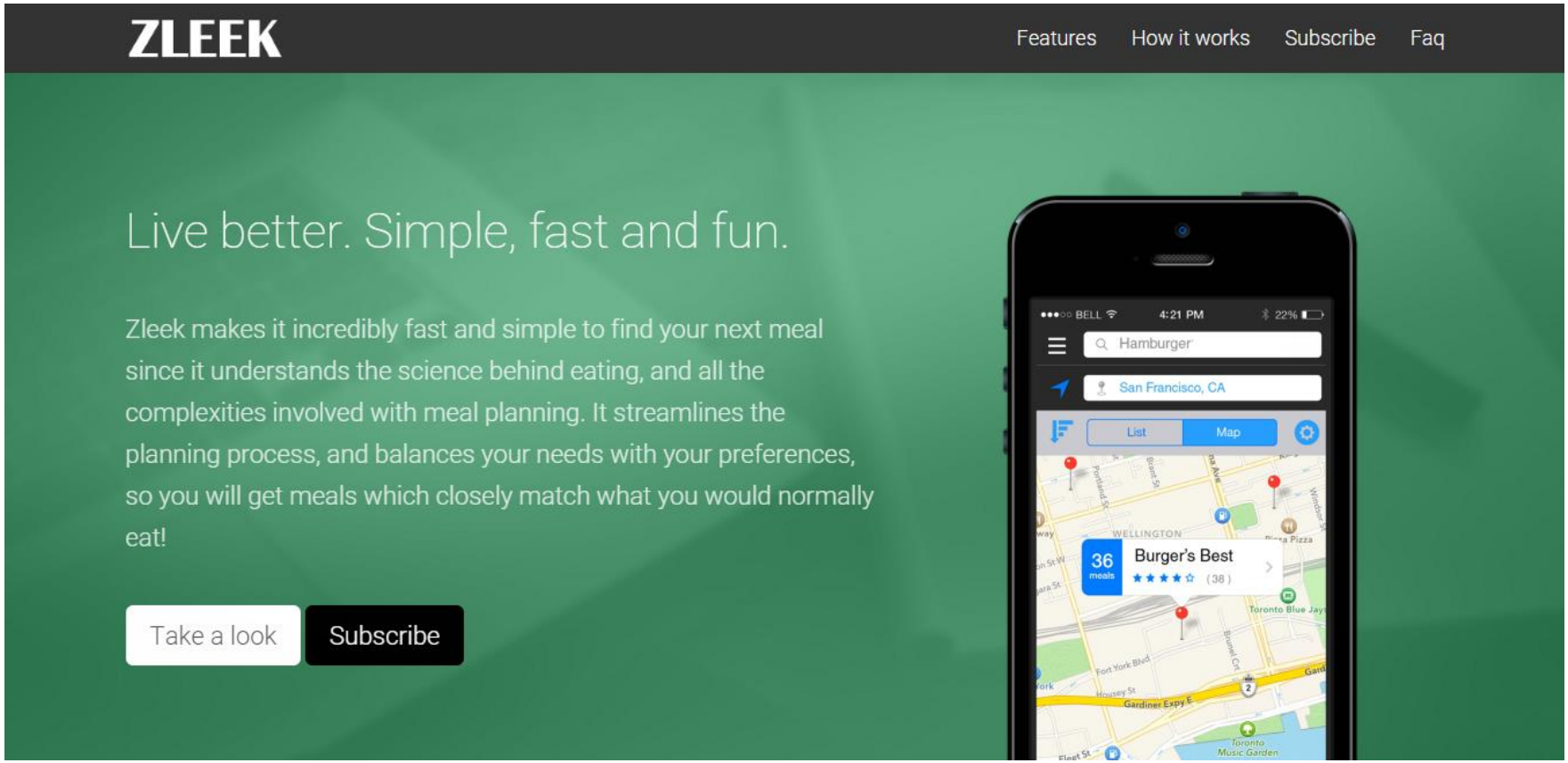
# Demo

docker-compose up

https://github.com/tugberkugurlu/AspNetCoreSamples/tree/itakeu16/rabbitmq-sample

# Complete Example

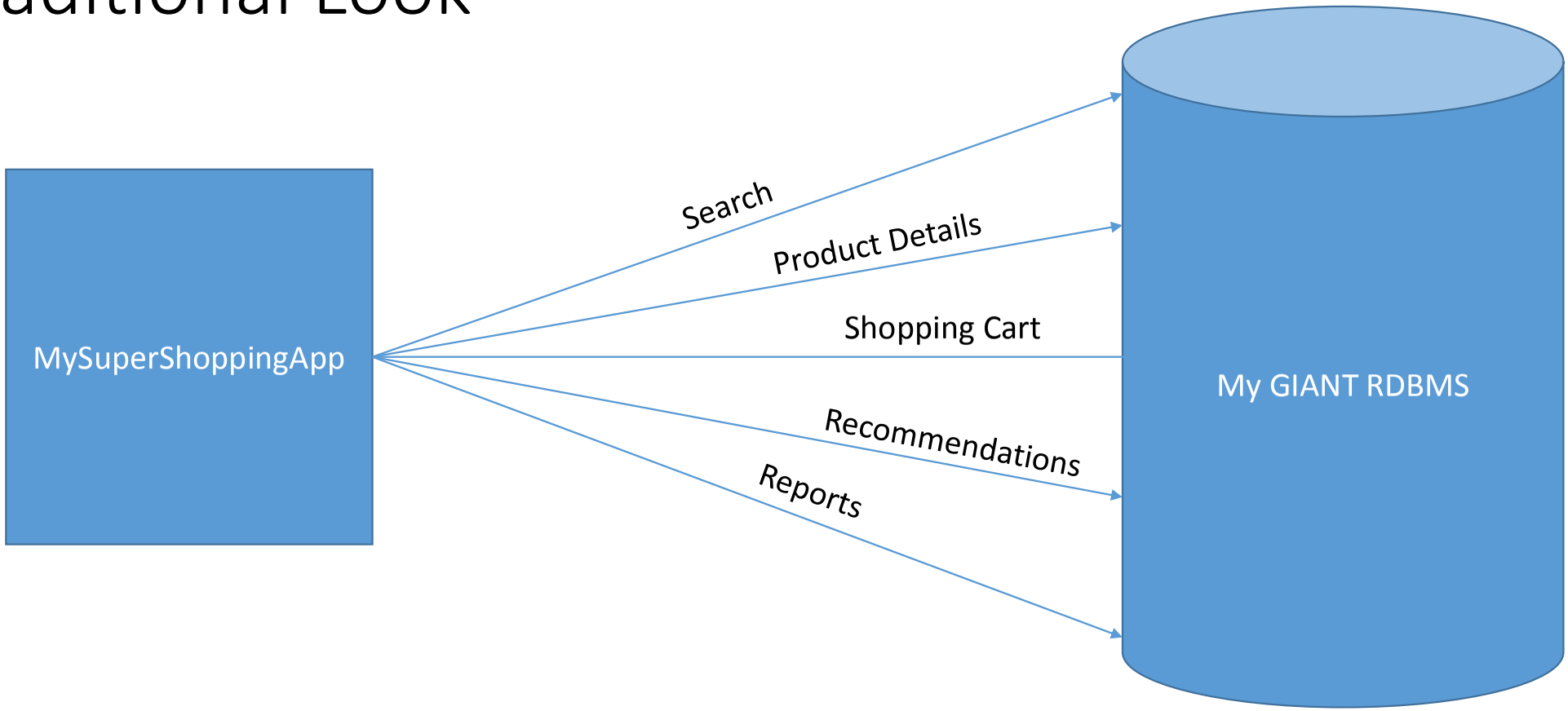
# zleek.com


ZLEEK
Features
How it works
Subscribe
Faq
Live better. Simple, fast and fun.
Zleek makes it incredibly fast and simple to find your next meal since it understands the science behind eating, and all the complexities involved with meal planning. It streamlines the planning process, and balances your needs with your preferences, so you will get meals which closely match what you would normally eat!
Take a look
Subscribe
BELL
4:21 PM
22%
Hamburger
San Francisco, CA
List
Map
36
meals
Burger's Best
(38)

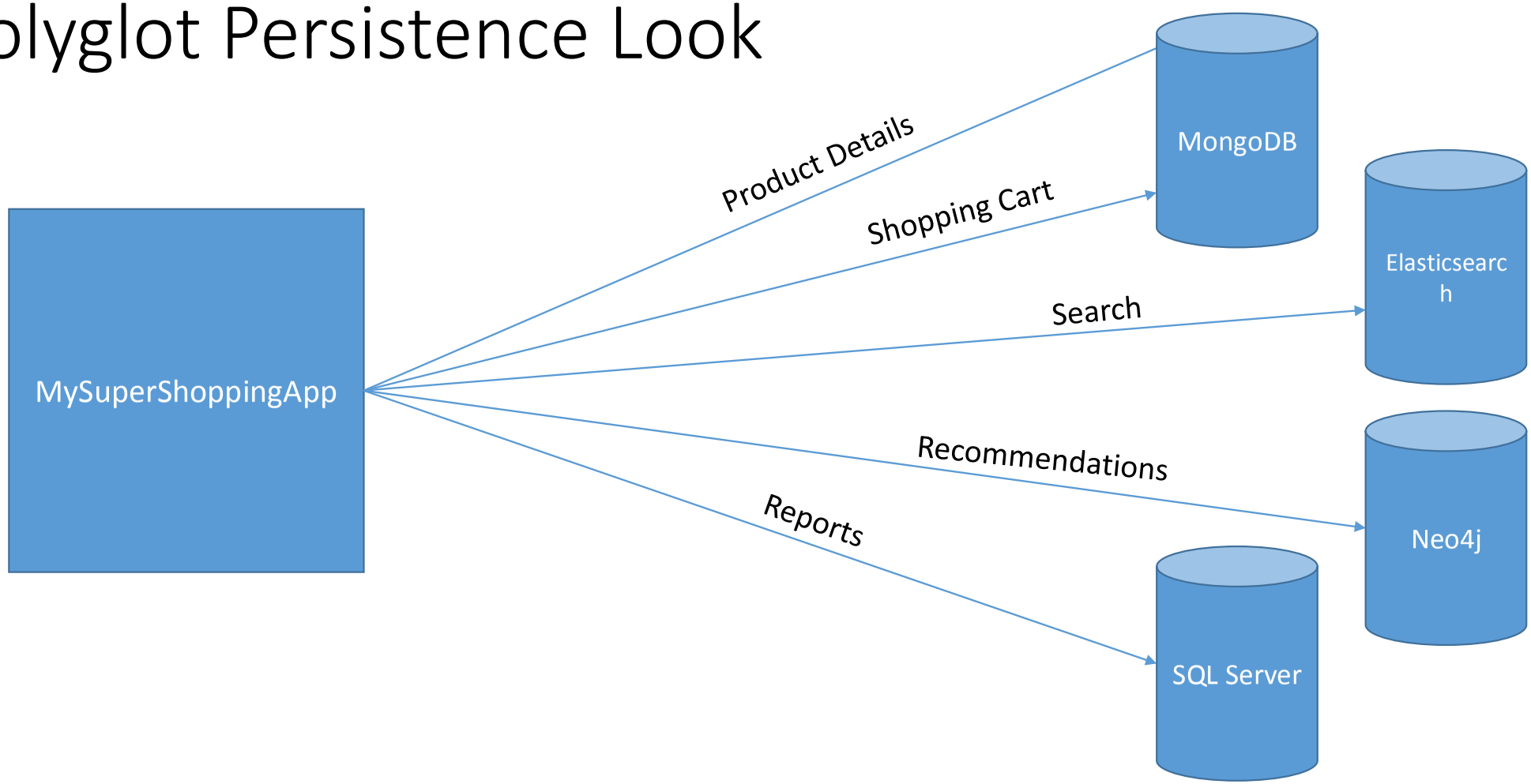
# Embrace Polyglot Persistence

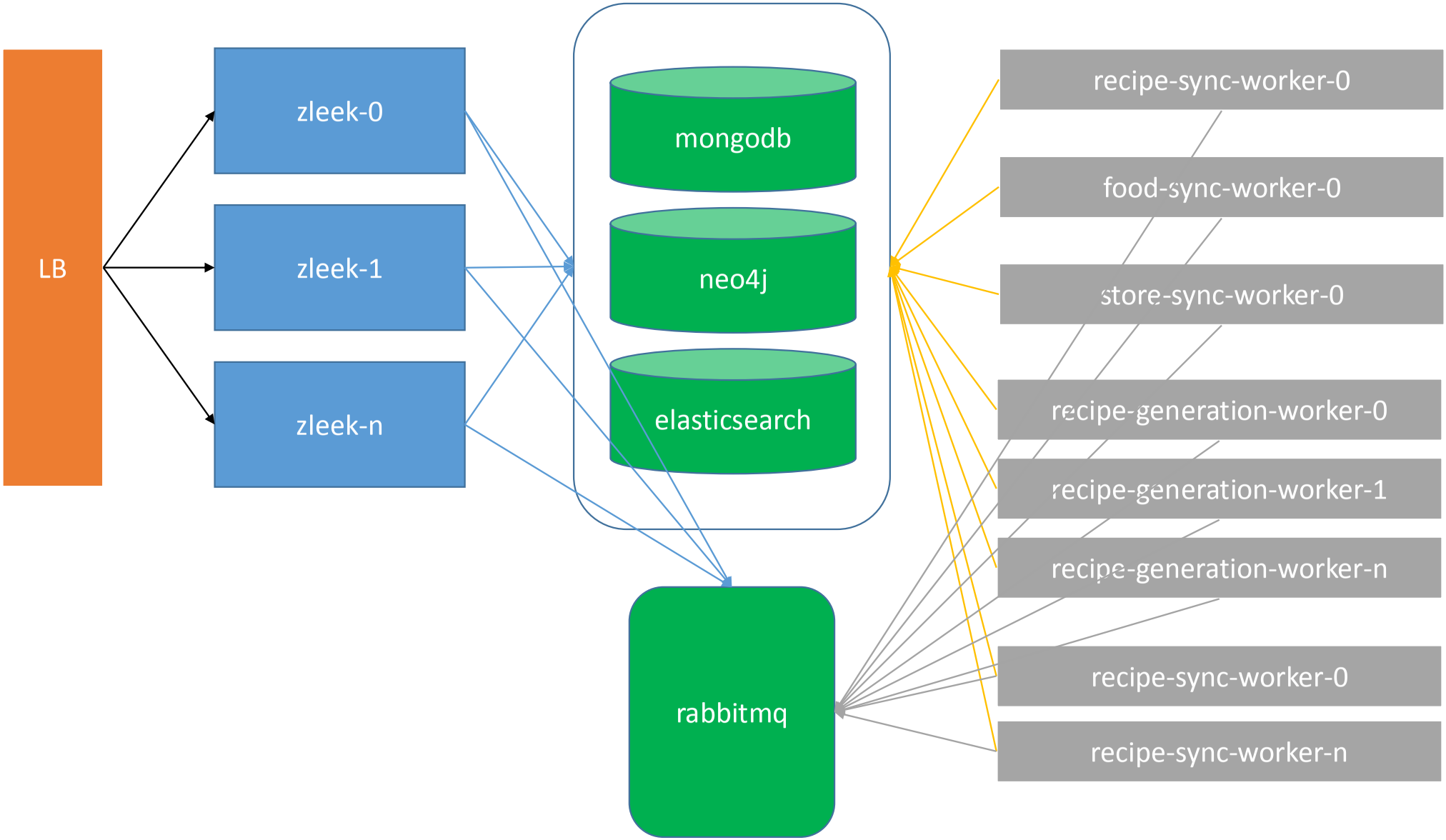
http://martinfowler.com/bliki/PolyglotPersistence.html

# Traditional Look



# Polyglot Persistence Look




LB
zleek-0
zleek-1
zleek-n
mongodb
neo4j
elasticsearch
rabbitmq
recipe-sync-worker-0
food-sync-worker-0
store-sync-worker-0
recipe-generation-worker-0
recipe-generation-worker-1
recipe-generation-worker-n
recipe-sync-worker-0
recipe-sync-worker-n

README.md

# NTE Project build passing

Contains the Core HTTP API, Authorization Server, Sync Workers and Initial Data Migration codebase of NTE Application.

> NTE Project has not been on production yet and it's still an ongoing project.

## Software Requirements

Before getting started with the project, you need to meet the following software requirements:

- Windows 8.1 (The project may work in any other Windows operating system but it has been never tested under any other OS than Windows 8.1)
- Visual Studio 2015
- MongoDB (version 2.6.0 or higher)
- Robomongo - UI tool for MongoDB
- Elasticsearch (version 1.3.2 of higher)
- Marvel plugin for elasticsearch
- Git for Windows (msysgit)
- Install "2007 Office System Driver: Data Connectivity Components" from http://www.microsoft.com/download/en/details.aspx?id=23734 (this is needed for the data migration project).

## Getting Started With the Project

Cloning the repository is the first step to get started with the project. Later, you can run the **build.cmd** file to build the project:

```
build.cmd
```

You need a few more steps to complete before you can actually run the project with Visual Studio and run the actual application:

1. You need to configure FQDN. How you can and the reason why you need to do this are explained inside the Wiki page.
2. Install and set up MongoDB.
3. Install and set up Elasticsearch.

## MongoDB Setup

MongoDB is the primary data storage system used for this project. During development, you need to have a MongoDB instance running and listening on port **27017** without needing any auth.

Preferably, unzip the MongoDB distribution under **c:\mongo** folder and have the following folder structure:

```
├──bin
├──config
├──data
│   ├──db-nte
│   └──logs
```

Under the config folder, have the following **mongod-nte.conf** file with the following content:

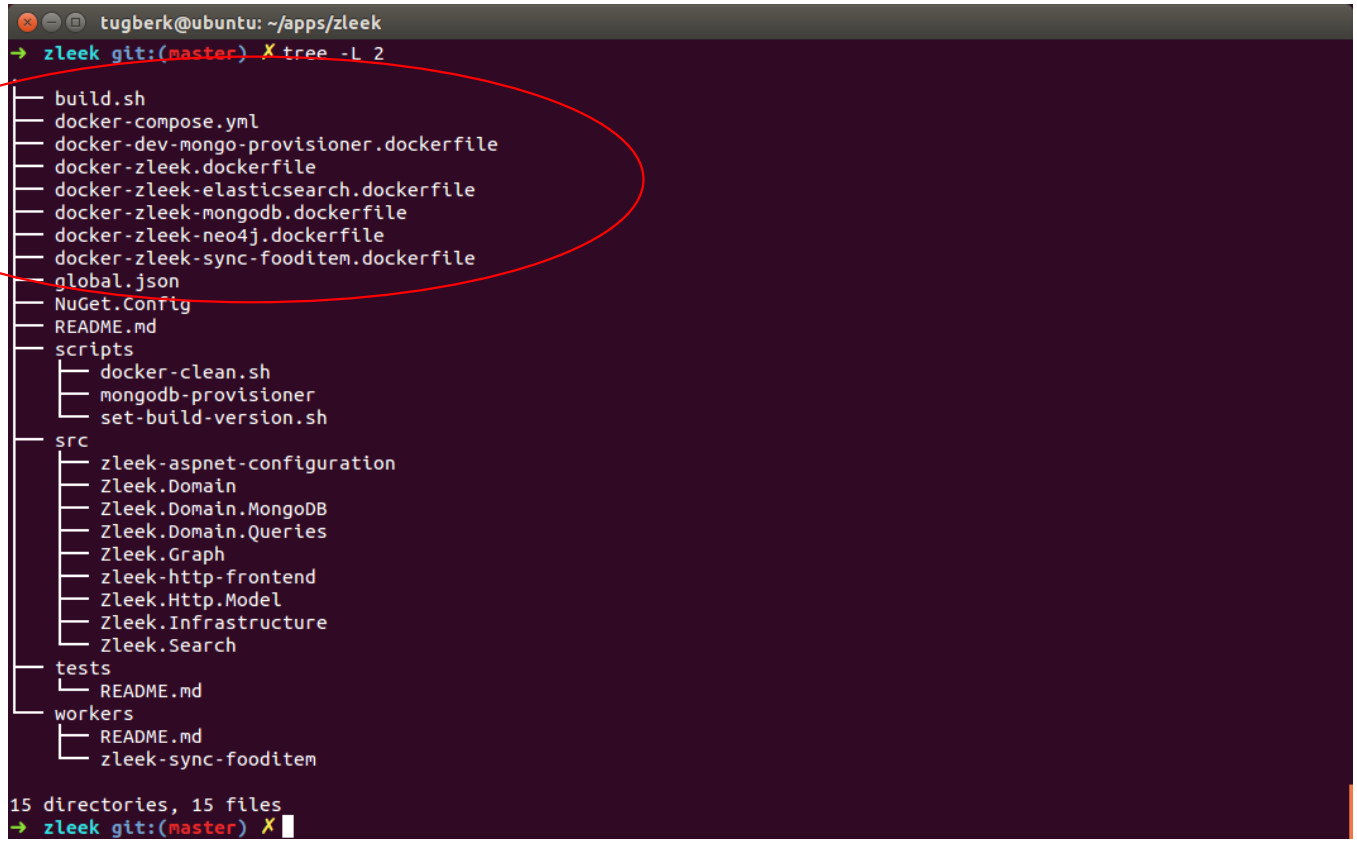
```
dbpath = c:\mongo\data\db-nte
port = 27017
logpath = c:\mongo\data\logs\mongo-nte.log
```

You can now start your mongodb instance by running the following command:

```
mongod.exe --config ..\config\mongod-nte.conf
```

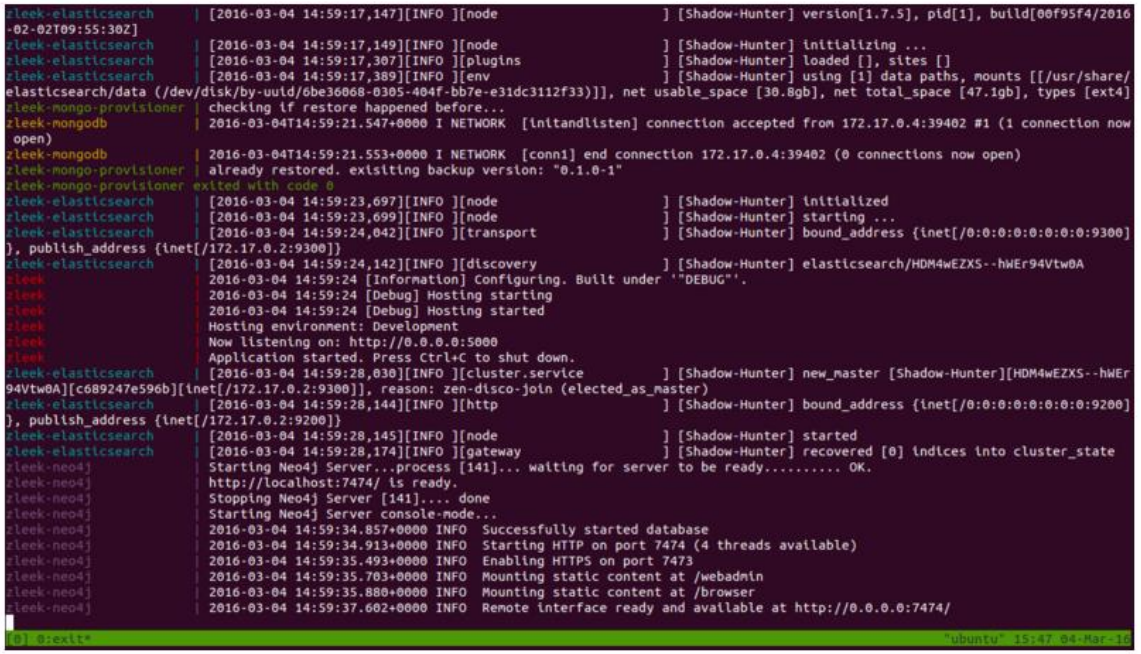
## Elasticsearch Setup

Preferably, unzip the Elasticsearch distribution under **c:\elasticsearch** folder and have the default folder structure. You can now start the elasticsearch instance by running **elasticsearch.bat** under **bin** folder.


tugberk@ubuntu: ~/apps/zleek
→ zleek git:(master) ✗ tree -L 2
├── build.sh
├── docker-compose.yml
├── docker-dev-mongo-provisioner.dockerfile
├── docker-zleek.dockerfile
├── docker-zleek-elasticsearch.dockerfile
├── docker-zleek-mongodb.dockerfile
├── docker-zleek-neo4j.dockerfile
├── docker-zleek-sync-fooditem.dockerfile
├── global.json
├── NuGet.Config
├── README.md
├── scripts
│ ├── docker-clean.sh
│ ├── mongodb-provisioner
│ └── set-build-version.sh
├── src
│ ├── zleek-aspnet-configuration
│ ├── Zleek.Domain
│ ├── Zleek.Domain.MongoDB
│ ├── Zleek.Domain.Queries
│ ├── Zleek.Graph
│ ├── zleek-http-frontend
│ ├── Zleek.Http.Model
│ ├── Zleek.Infrastructure
│ └── Zleek.Search
├── tests
│ └── README.md
└── workers
├── README.md
└── zleek-sync-fooditem
15 directories, 15 files
→ zleek git:(master) ✗

```
mkdir ~/apps && cd ~/apps
git clone git@github.com:n2e/zleek.git && cd zleek
docker-compose build
docker-compose up
```

You shoud end up with a terminal filled with docker-compose output:



# Demo

# Deployments

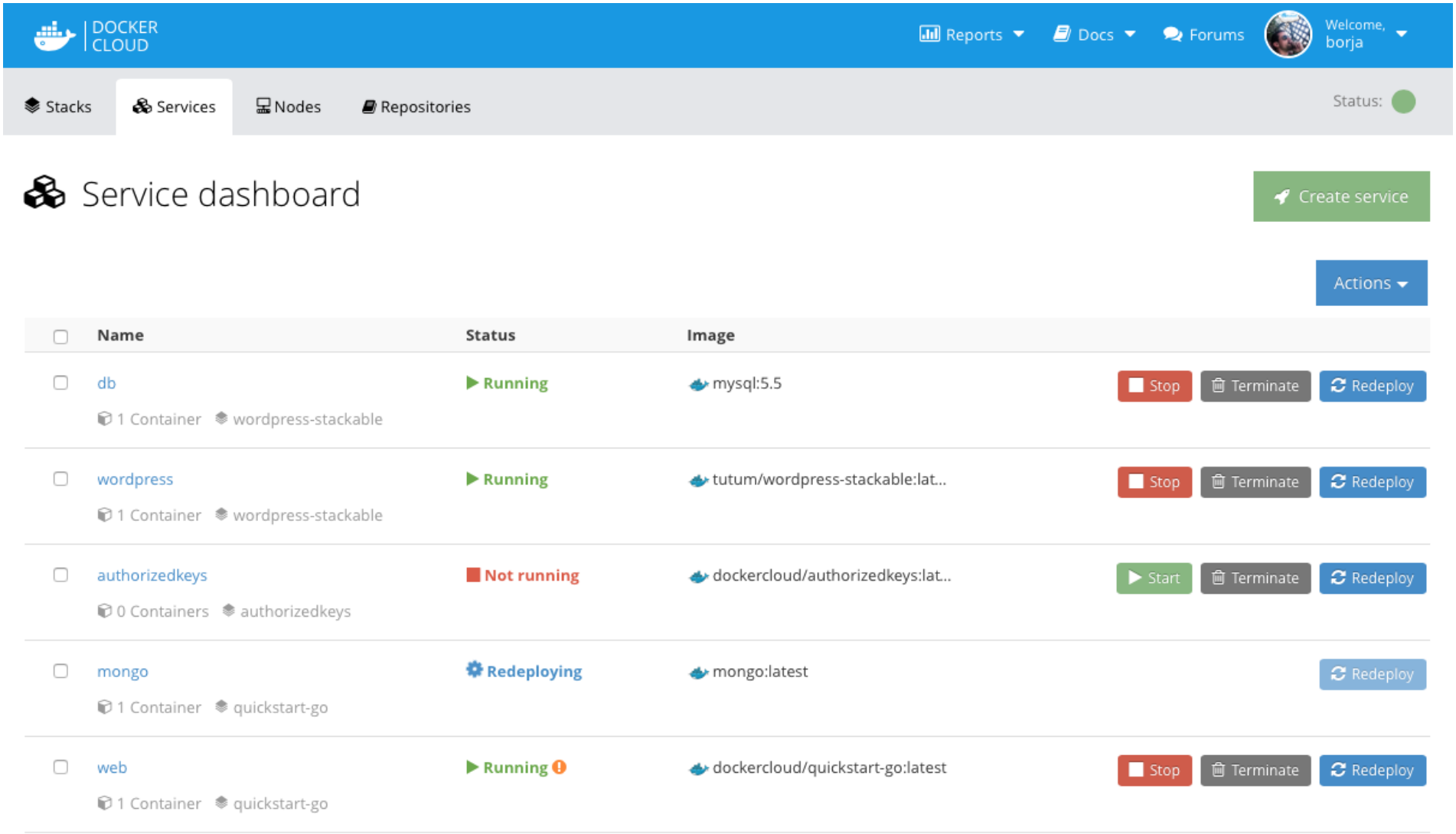
```
declare -A repositoryMapping=( ["docker-nginx.dockerfile"]="zleek-web-client-nginx" ["docker-landing.dockerfile"]="zleek-landing" ["docker-
scriptsDir=$( cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd )
rootDir=$(dirname $scriptsDir)
dockerFiles=${rootDir%%/}/*.dockerfile

echo "configuring AWS CLI"
aws configure set aws_access_key_id $AWS_ACCESS_KEY_ID || exit 1
aws configure set aws_secret_access_key $AWS_SECRET_ACCESS_KEY || exit 1
aws configure set default.region $AWS_REGION || exit 1

echo "Docker login for aws registry"
$(aws ecr get-login --region $AWS_REGION)

#build, tag and push
for dockerFilePath in $dockerFiles
do
    echo "processing for '$dockerFilePath'"

    dockerFileName=$(basename $dockerFilePath)
    repositoryName=${repositoryMapping[$dockerFileName]}
    if [ ! -z "$repositoryName" ];
    then
        docker build -t $repositoryName -f $dockerFilePath $rootDir || exit 1
        docker tag $(echo "$repositoryName:latest") $(echo "719773605397.dkr.ecr.us-east-1.amazonaws.com/$repositoryName:latest") || exit 1
        docker tag $(echo "$repositoryName:latest") $(echo "719773605397.dkr.ecr.us-east-1.amazonaws.com/$repositoryName:$VERSION") || exit
        docker push $(echo "719773605397.dkr.ecr.us-east-1.amazonaws.com/$repositoryName:latest") || exit 1
        docker push $(echo "719773605397.dkr.ecr.us-east-1.amazonaws.com/$repositoryName:$VERSION") || exit 1
    else
        echo "'$dockerFileName' docker file ('$dockerFilePath') doesn't have a mapping for repository name. Skipping."
    fi
done
```




http://www.infoworld.com/article/3040475/application-virtualization/4-no-bull-takeaways-about-docker-cloud.html