

DevOps 2TIN

Chapter 6

DTAP & Environments on demand



**DE HOGESCHOOL
MET HET NETWERK**

Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be



DTAP

Wat is het
Configuratie management
Mutable infrastructuur
Pets Vs. Cattle
Docker

Wat is een Pipeline?



Definitie “deployment pipeline”:

(first defined by Jez Humble and David Farley in their book *Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation*)

It ensures that all code checked in to version control is automatically built and tested in a production-like environment.

Sleutelwoorden:

- Alle Code
- Versiebeheer
- Automatisch gebouwd
- Automatisch getest
- **Productie-waardige omgeving**

Wat is een DTAP?



Rectal
Swab Test



Urine Test



Vaginal
Swab Test

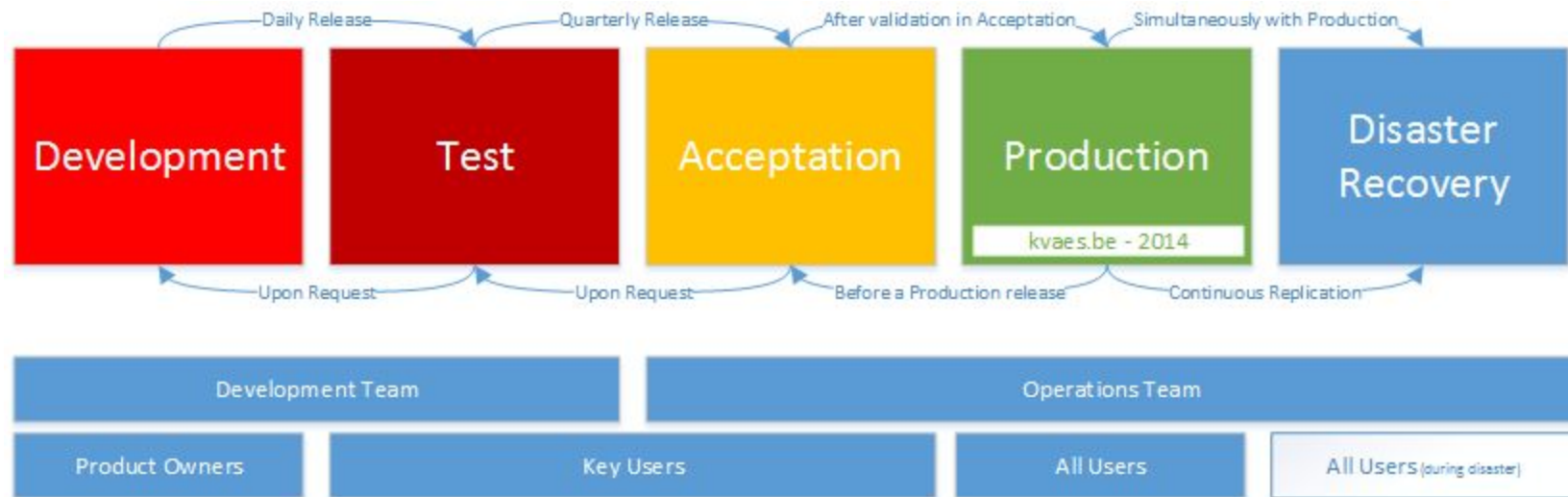


Throat
Swab Test



Self-STD Testing Clinic

Wat is DTAP?



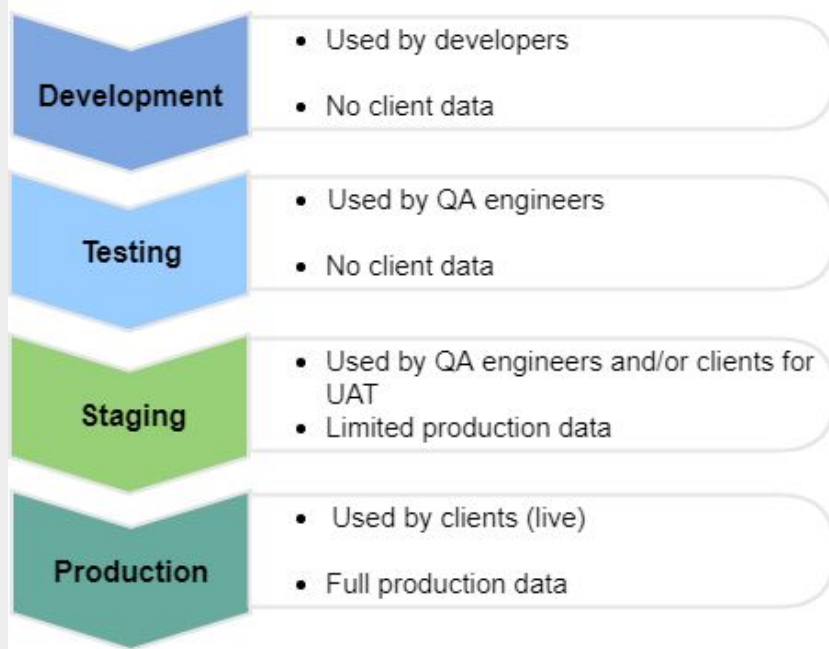
Source: <https://kvaes.wordpress.com/2014/10/26/the-dtap-street-a-phased-approach-to-a-development-deployment-cycle/>

Iedereen een omgeving - Recap



Automatiseren van de deploy

- Omgeving in orde maken
 - Dependencies
 - omgeving specifieke parameters, credentials, secrets, ...



Code	Dependencies	build	(unit)test	archive artifact	Deploy to QA	Deploy to production
CI					CD	

Deployment Pipeline



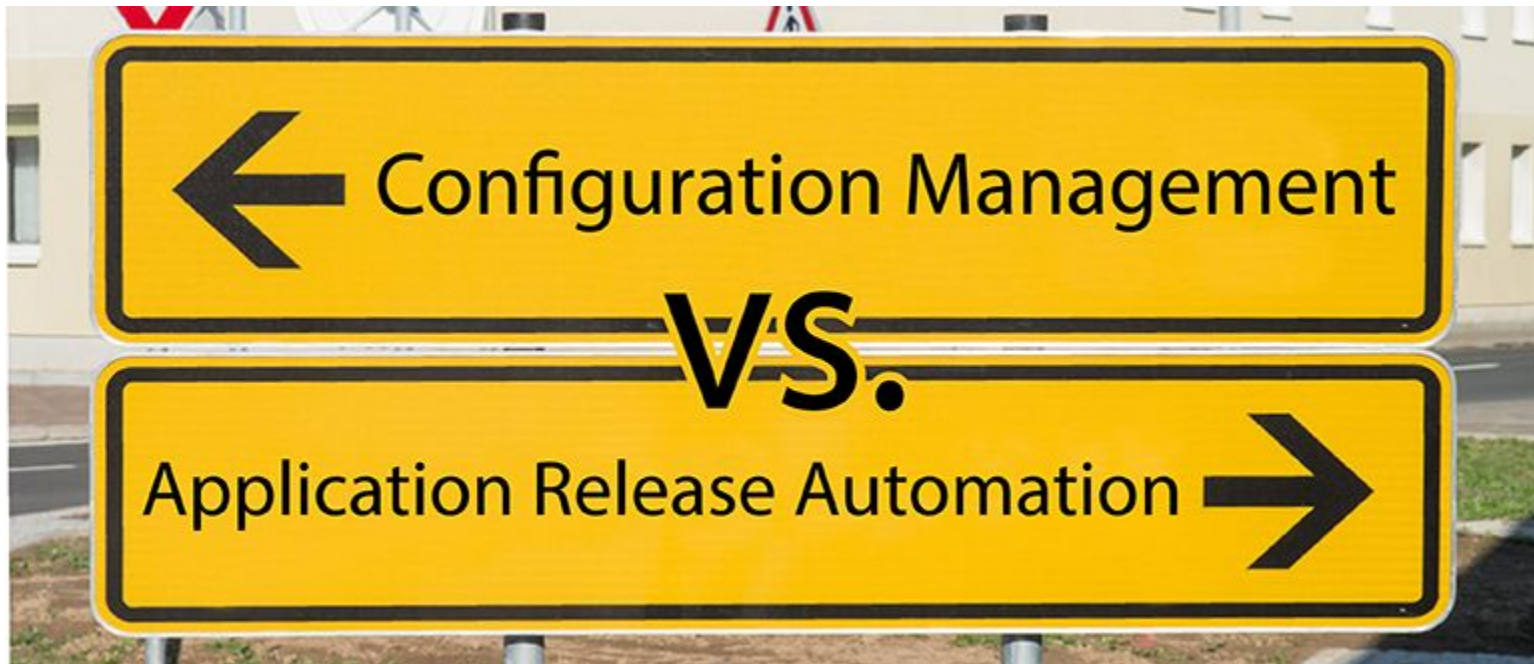
Omgevingen zijn onderdeel van de Continuous Deployment stappen van de Pipeline:

- Definitie:

Continuous deployment (CD) is a [software engineering approach](#) in which software functionalities are delivered frequently through automated [deployments](#) (-Wikipedia)

Elke omgeving wordt opgezet met specifieke doelen, vroeger manueel, nu als het kan automatisch, om de applicatie te testen en valideren alvorens deze naar de volgende omgeving gaat, met als doel Productie te bereiken.

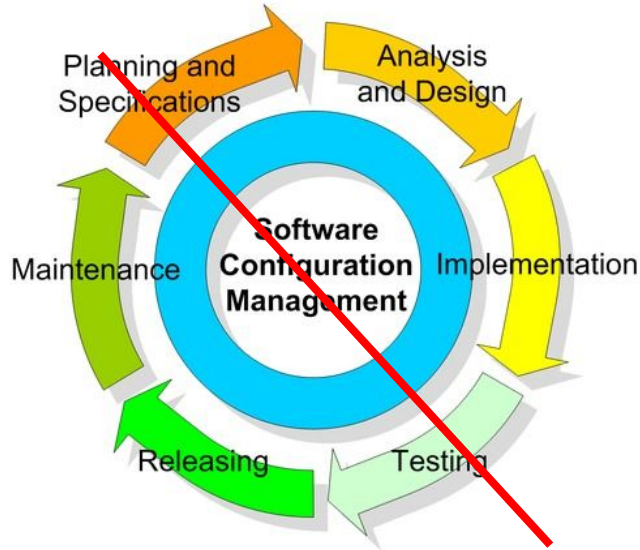
Differentiëren per omgeving



Configuration management



Geen ITIL



Wel configuratie parameters zoals :

- IP adressen
- API endpoints
- Credentials
- (test)data

Configuration management



Voorlopig Manueel / semi-geautomatiseerd (best practices):

- Aparte repository voor configuration data
 - Single source of truth
- Georganiseerd per omgeving (Test, Acceptatie, Productie)
- In-line toevoegen aan deploy in Jenkins.

Later automatisch met Configuration Management Tools, nu semi automatisch via
bv. ssh / scp / docker



SSH & sshagent plugin in jenkins



- SSH authenticatie naar verschillende servers
 - private & public keys (ssh-keygen)
 - Zie opdracht security essentials 1TIN
- sshagent
 - plugin voor jenkins
 - gebruik van keys bij ssh commando's als authenticatie
 - integratie met credential manager

Global (Jenkins, nodes, items, all child items, etc.)

ID ?

prodserver

Description ?

private key for the production server

Username

student

☐ Treat username as secret ?

Private Key

☒ Enter directly

Key

Concealed for Confidentiality Replace

SSH & SSH agent plugin in jenkins



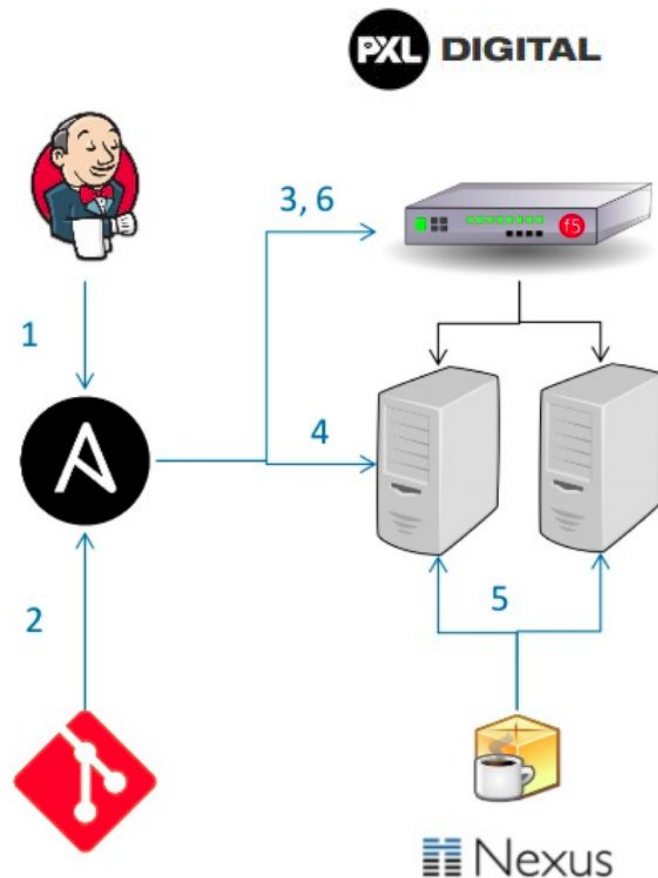
```
stage ('connect to production'){
  steps{
    sshagent(['prodserver']) {
      // Commands in deze blok gebruiken de ssh key 'prodserver'
      sh "ssh -o StrictHostKeyChecking=no student@172.25.49.178 ls -alh"
    }
  }
}
```

```
[ssh-agent] Using credentials student (private key for the production server)
[ssh-agent] Looking for ssh-agent implementation...
[ssh-agent] Exec ssh-agent (binary ssh-agent on a remote machine)
$ ssh-agent
SSH_AUTH_SOCK=/tmp/ssh-P7jVGRKtf0nk/agent.4580
SSH_AGENT_PID=4582
Running ssh-add (command line suppressed)
Identity added: /var/lib/jenkins/workspace/testpipeline@tmp/private_key_18000783756353641703.key (student@student-Virtual-Machine)
[ssh-agent] Started.
[Pipeline] {
[Pipeline] sh
+ ssh -o StrictHostKeyChecking=no student@172.25.49.178 ls -alh
Warning: Permanently added '172.25.49.178' (ECDSA) to the list of known hosts.
total 88K
drwxr-xr-x 18 student student 4,0K Nov 17 12:24 .
drwxr-xr-x  3 root    root    4,0K Nov 17 07:49 ..
-rw-r--r--  1 student student  18 Nov 17 12:22 .bash_history
-rw-r--r--  1 student student 220 Nov 17 07:49 .bash_logout
-rw-r--r--  1 student student 3,7K Nov 17 07:49 .bashrc
drwxr-xr-x 10 student student 4,0K Nov 17 12:22 .cache
```

Configuration management

Later automated:

1. triggers deployment run
 2. playbook checkout
 3. remove from loadbalancer
 4. deployment tasks
 5. download software artefact
 6. put back in loadbalancer
- same playbook in every \$ENV



Application Release Automation



=

“we nemen het build artifact, kopiëren en unpacken het op de server, toch eh?”

ARA Tools = Hype Enterprise tooling terminologie

MAAR definitie kan ons helpen:

Application release automation provides a number of benefits, including:

- Streamlining processes
- Reducing timelines and decreasing manual tasks
- Increasing departmental agility and flexibility
- Improving productivity and collaboration while controlling risk

Application Release Automation made easier

Laat ons beginnen bij het begin:



Build once, deploy anywhere - maar niet elke omgeving is gelijk, toch?



Applicatie dependencies

Calculator-App: NodeJS, libraries, ... ?

Oplossing voor elke deploy omgeving: Manueel dependencies in orde maken

OF:



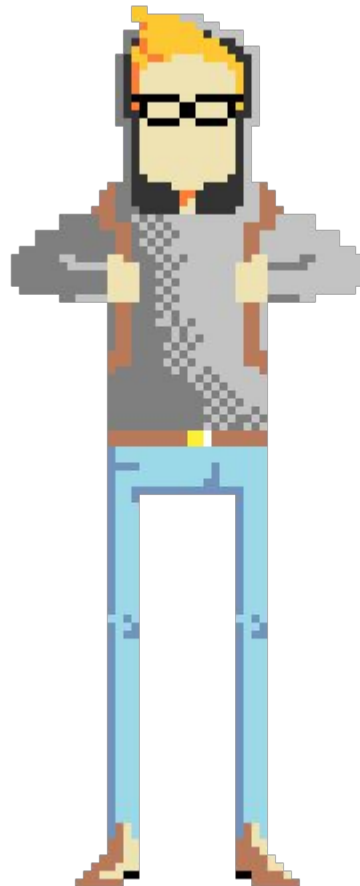
Build once, pack correctly, deploy anywhere

Voordelen van applicatie in docker te bouwen:

- Runs anywhere*
- Dependencies included
- Promote immutable infrastructure

Immutable wat?

*waar docker op werkt

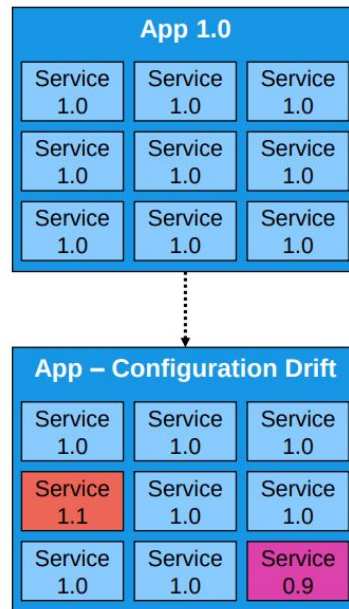


Mutable & immutable infrastructuur



Mutable Infrastructure

- Infrastructure will be **continually updated**, patched and tuned to meet the ongoing needs of the purpose it serves.
- Over time, as you apply more and more updates, each server builds up a **unique history of changes**.
- As a result, each service becomes slightly different than all the others, leading to **configuration drift** and can result to bugs which can be difficult to diagnose and reproduce.

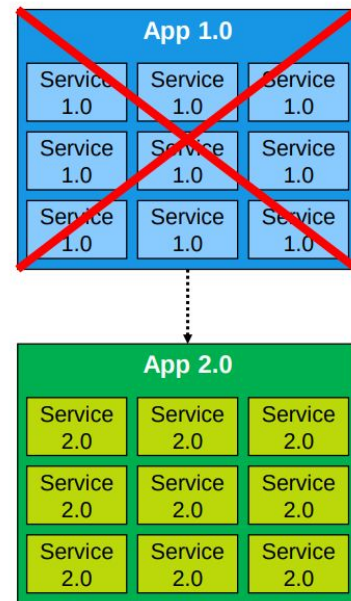


Mutable & immutable infrastructuur



Immutable Infrastructure

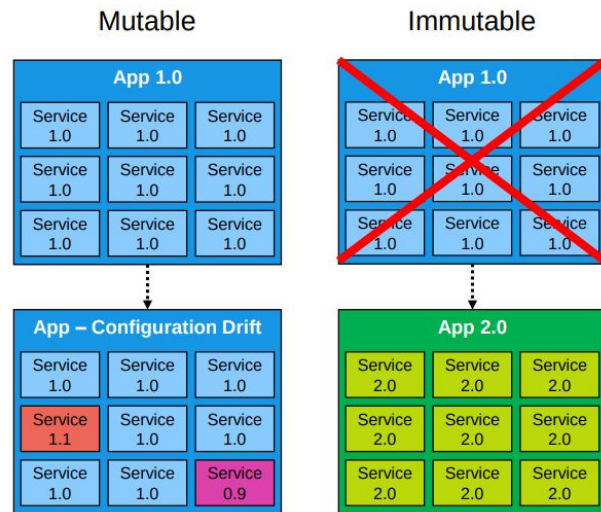
- Using **Infrastructure-As-Code** to deploy machine images created by Docker, result “changes” to be deployments of a completely new app version.
- Reduces the likelihood of **configuration drift** bugs, makes it easier to know exactly what software is running on each server.
- **Automated testing** are more effective, as an immutable image that passes your tests in the test environment is likely to behave exactly the same way in the production.



Mutable & immutable infrastructuur



- The **Pets and Cattle** debate.
- One approach is not necessary better then the other, it depends on your use-case.
- With the mutable approach, the team needs to be aware of the infrastructure “history”.
- Generally speaking, the immutable approach is better for stateless applications.
- Immutable drives no deviation and no changes. It is what it is.



Pets Vs. Cattle



Pets

Legacy Infrastructure

Pets are given names like grumpycat.petstore.com
They are unique, lovingly hand raised and cared for
When they get ill, you nurse them back to health

Infrastructure is a permanent fixture in the data center

Infrastructure takes days to create, are serviced weekly,
maintained for years, and requires migration projects to move

Infrastructure is modified during maintenance hours and
generally requires special privileges such as root access

Infrastructure requires several different teams to coordinate and
provision the full environment

Infrastructure is static, requiring excess capacity to be dominant
for use during peak periods of demands

Infrastructure is an capital expenditure that charges a fix amount
regardless of usage patterns



Cattle

Cloud-Friendly Infrastructure

Cattle are given numbers like 10200713.cattlerancher.com
They are almost identical to other cattle
When they get ill, you replace them and get another

Infrastructure is stateless, ephemeral, and transient

Infrastructure is instantiated, modified, destroyed and recreated
in minutes from scratch using automated scripts

Infrastructure uses version-controlled scripts to modify any
service without requiring root access or privileged logins

Infrastructure is self-service with the ability to provision
computing, network and storage services with a single click

Infrastructure is elastic and scales automatically, expanding
and contracting on-demand to service peak usage periods

Infrastructure is an operating expenditure that charges only for
services when they are consumed

Pets Vs. Cattle

Service Model



- Pets are given names like pussinboots.cern.ch
- They are unique, lovingly hand raised and cared for
- When they get ill, you nurse them back to health



- Cattle are given numbers like vm0042.cern.ch
- They are almost identical to other cattle
- When they get ill, you get another one

- Future application architectures should use Cattle but Pets with strong configuration management are viable and still needed

Maar dus...Docker:



Dockerfile:

```
8
9 FROM php:7.4-apache
10 COPY . /var/www/html/
11 EXPOSE 80
12
13 RUN apt-get update && apt-get install -y zip
14 RUN curl -sS https://getcomposer.org/installer | php -- --install-dir=/usr/local/bin --
    filename=composer
15
16 WORKDIR /var/www/html
17 RUN composer install
```


Docker files moeten idempotent zijn.



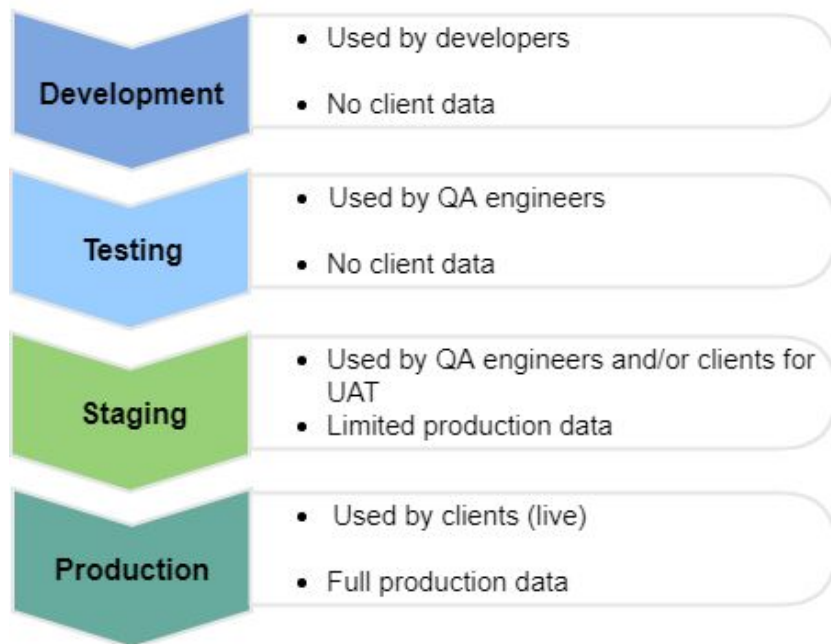
Dockerfiles moet dus NIET:



- Fixed versie nummers bevatten (in dit geval puppet)
- ervan uitgaan dat het toegang heeft tot bepaalde tools (puppet in dit geval)
- zomaar dingen naar het internet wegschrijven

```
1 FROM alpine:3.4
2 RUN apk add --no-cache \
3     ca-certificates \
4     pciutils \
5     ruby \
6     ruby-irb \
7     ruby-rdoc \
8     && \
9     echo http://dl-4.alpinelinux.org/alpine/edge/community/ && /etc/apk/repositories && \
10    apk add --no-cache shadow && \
11    gem install puppet:"5.5.1" facter:"2.5.1" && \
12    /usr/bin/puppet module install puppetlabs-apk
13 # Push docs to S3
14 FROM containerlabs/aws-sdk AS push-docs
15 ARG push-docs=false
16 COPY --from=build docs docs
17 RUN [ "$push-docs" == true ] && aws s3 cp -r docs s3://my-docs-bucket/
18 # Install Java application
19 RUN /usr/bin/puppet agent --onetime --no-daemonize
20 ENTRYPOINT ["java", "-jar", "/app/spring-boot-application.jar"]
21
```

Recap





PLURALSIGHT



Title: [Building Reactive Microservices](#)

how to build good infrastructure, features importance of Idempotent operations .
[1h26mins]

Title: [Getting Started with Ansible](#)

Introduction to this IaC tool that focuses heavily on idempotency [4h45mins]

Title: [Importance of DTAP environments - WordCampSG 2017](#)

Conference talk about the importance of the different environments. [17mins]



Assignments