zühlke
empowering ideas

# Introduction to Cloud Computing

(v20191211)

# Definitions

# The NIST Definition of Cloud Computing

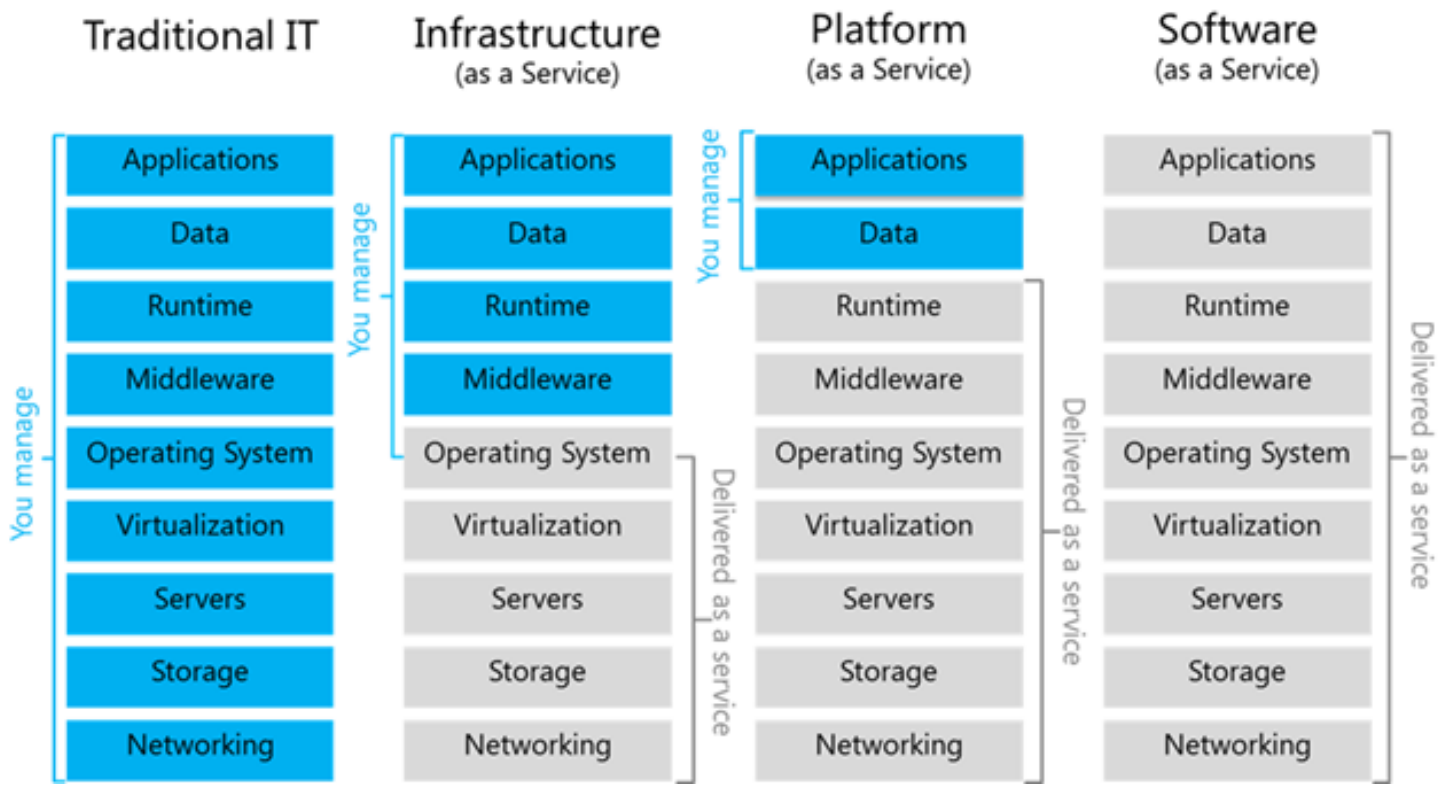

## Essential Characteristics

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

# The NIST Definition of Cloud Computing

## Service Models

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)



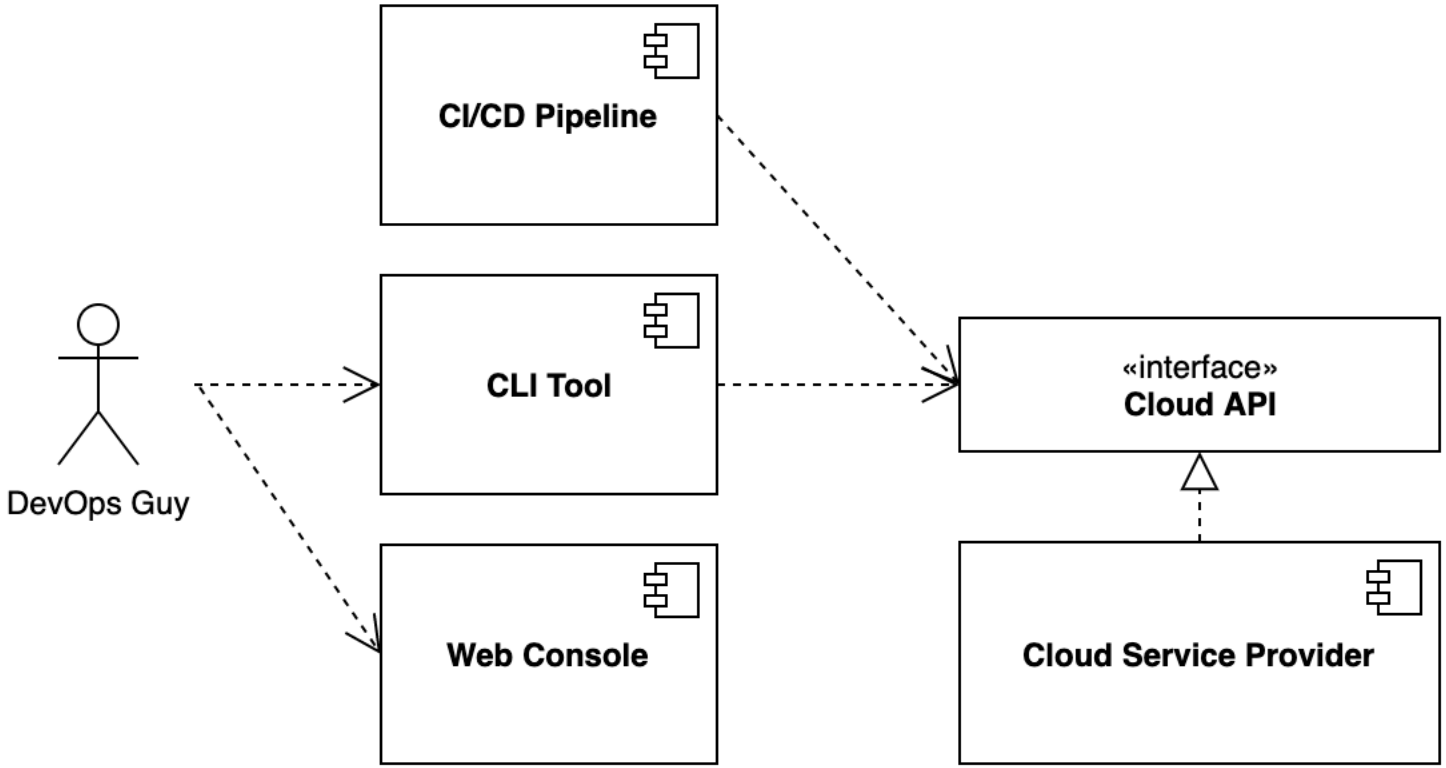
# The NIST Definition of Cloud Computing

## Deployment Models

- Private cloud
- Community cloud
- Public cloud
- Hybrid cloud

# Cloud Computing

## Cloud Service Provider APIs



# Relevant Players (Public Cloud)

**Figure 1. Magic Quadrant for Cloud Infrastructure as a Service, Worldwide**


Source: Gartner (July 2019)

# Cloud Computing Benefits

# Infrastructure-as-Code

Software engineering principles applied to infrastructure


## Stone Age

- Manual infrastructure provisioning (e.g. ordering new servers, putting them in a 19” rack)
- Slow infrastructure provisioning (e.g. weeks or even months)
- Non-reproducible and diverging infrastructure environments (e.g. snowflake servers)

## Today

- Fully automated infrastructure provisioning
- Fast infrastructure provisioning (e.g. minutes or hours, not days)
- Infrastructure versioning (e.g. infrastructure code versioned in git)
- CI/CD pipelines not only for application workloads, but also for infrastructure

# Shared responsibility model


More focus – less “undifferentiated heavy-lifting”

## IaaS (fallback)

- Focus on VM workloads – let the cloud provider manage datacenters, physical security, power, cooling, severs, network infrastructure, storage infrastructure, etc.

## PaaS (preferred)

- Use managed services and focus on application DevOps – let the cloud provider manage operating systems, databases, load balancers, firewalls, container orchestrators, etc. (installation, configuration, hardening, patching, monitoring)

# Public Cloud Providers

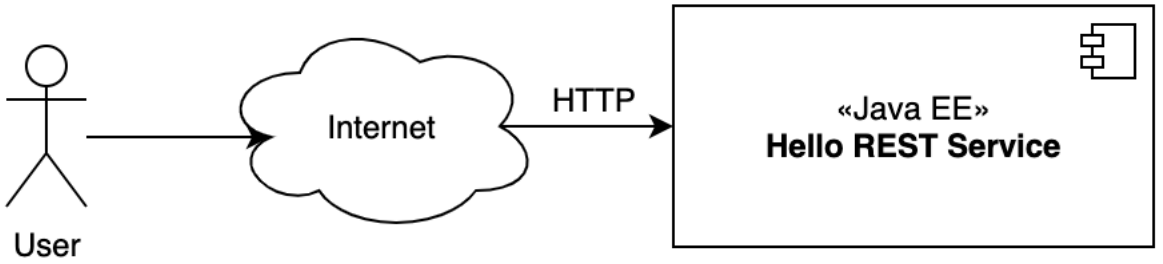
## AWS, GCP, Azure


- Pay-per-use model (CAPEX ⇨ OPEX)
- Rapid elasticity
- Huge and growing service portfolio
- Rapid pace of innovation
- Global footprint (e.g. regions across the globe, including Europe)
- Datacenters in Switzerland (GCP, Azure)
- De-facto unlimited resources
- …

# Let's bring a Java EE application into production at AWS...

# Hello World

## Stateless Java EE REST service (JAX-RS)




```
import java.util.logging.Logger;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Response;

@ApplicationPath("/")
public class CatalogRestApplication extends Application {}

@Path("/")
public class CatalogEndpoint {
   private final static Logger log = Logger.getLogger(CatalogEndpoint.class.getName());

   @GET
   @Produces("application/json")
   public Response doGet() {
         log.info("Service invoked");

         return Response.ok("Hello World").build();
   }
}
```

# Java Application Servers in the Cloud?

Deployment Artifact ⇨ Runnable JAR

## Classic

Enterprise Application deployed on Application Server:


## New

Enterprise Application bundled with tailored Application Server:

# Runnable Java EE JAR

## Maven pom.xml snippet


```
...

<plugin>
   <groupId>io.thorntail</groupId>
   <artifactId>thorntail-maven-plugin</artifactId>
   <version>${version.thorntail}</version>
   <executions>
         <execution>
                  <goals>
                           <goal>package</goal>
                  </goals>
         </execution>
   </executions>
</plugin>

...

<dependency>
   <groupId>io.thorntail</groupId>
   <artifactId>jaxrs</artifactId>
</dependency>

...
```

# Java Applications in the Cloud?

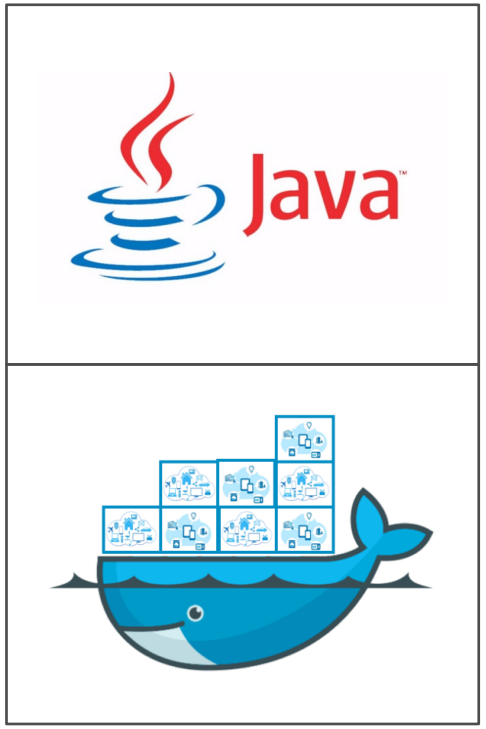
IaaS ⇨ PaaS

## IaaS

Running java workloads on virtual machines (e.g. EC2):


## PaaS

Running java workloads on **managed** container infrastructure (e.g. ECS, EKS):



# Dockerized Java Application

## Leverage existing docker image with pre-installed JRE

zühlke
empowering ideas

## Dockerfile

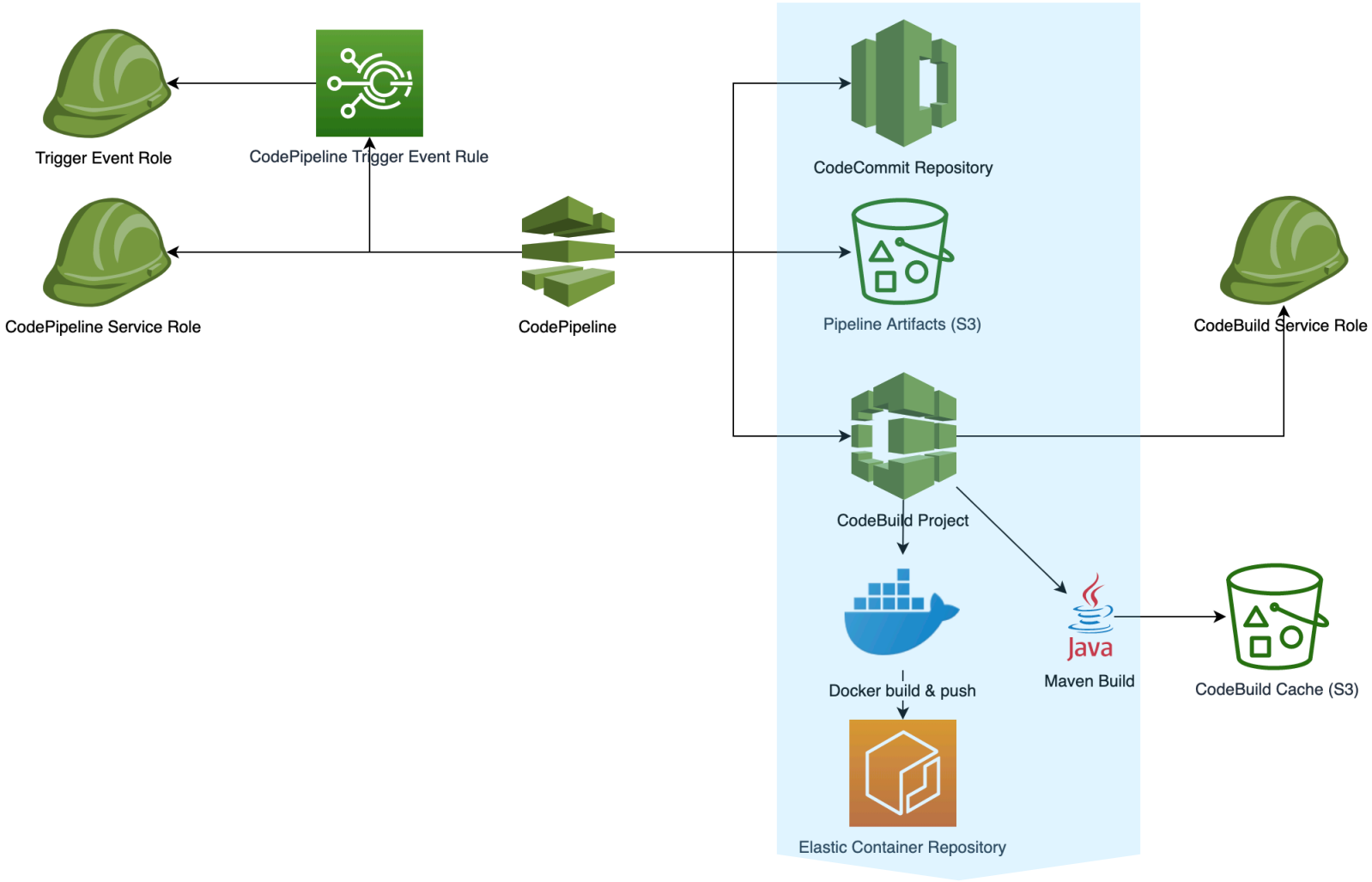
```
FROM openjdk:jre-alpine
ADD target/app-thorntail.jar /opt/thorntail.jar
EXPOSE 8080
ENTRYPOINT ["java", "-jar", "/opt/thorntail.jar", "-Djava.net.preferIPv4Stack=true"]
```

## pom.xml snippet

```
<plugin>
   <groupId>io.fabric8</groupId>
   <artifactId>docker-maven-plugin</artifactId>
   <version>0.28.0</version>
   <configuration>
         <images><image>
                  <name>${dockerimage.name}</name>
                  <build><dockerFileDir>${project.basedir}</dockerFileDir></build>
         </image></images>
   </configuration>
</plugin>
```

# Build Pipeline

## Java Sources ⇨ Docker Image



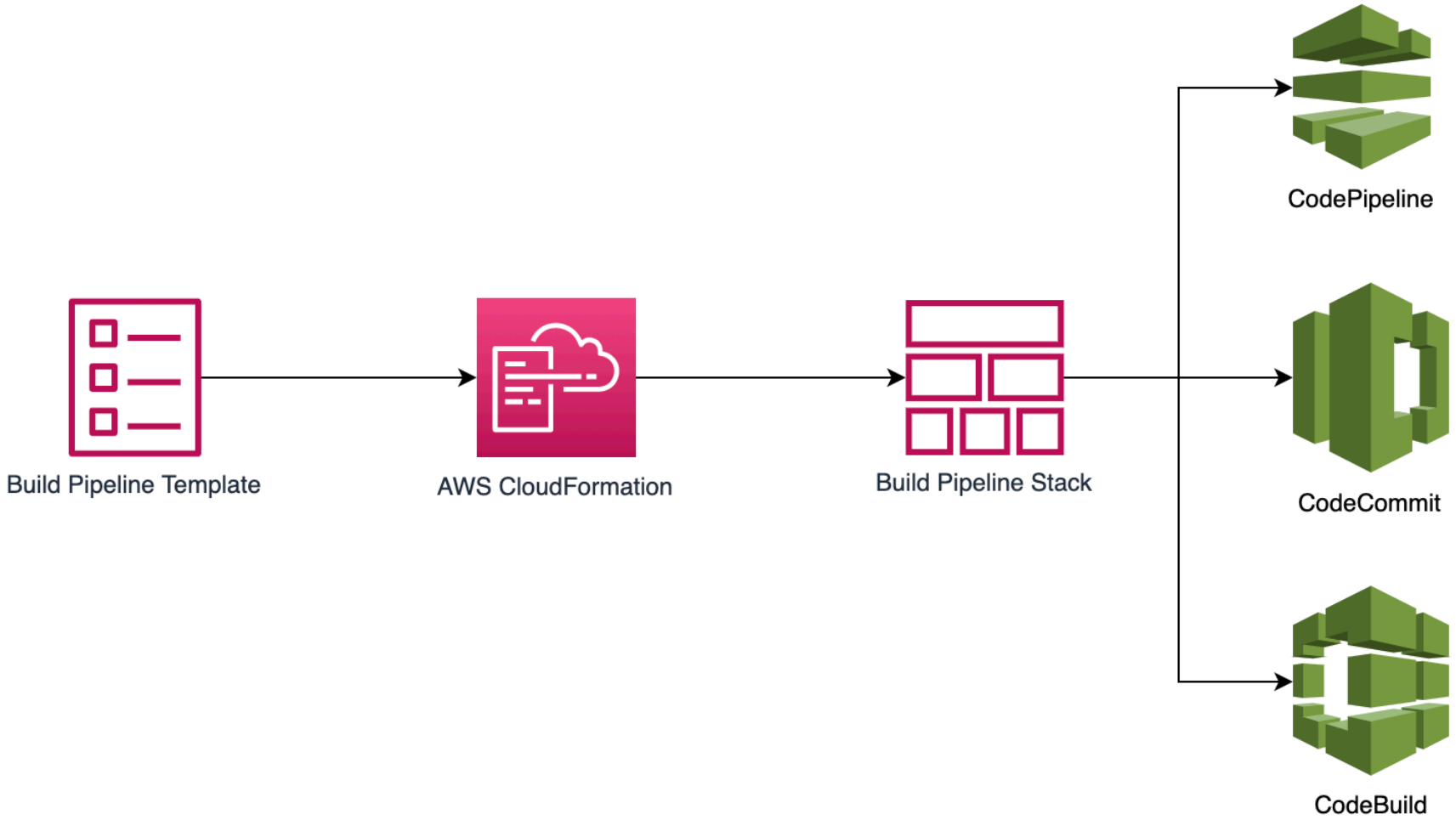
# Build Pipeline

## AWS CodeBuild sample buildspec.yaml

```
version: 0.2

phases:
  install:
    commands:
      - $(aws ecr get-login --region $AWS_DEFAULT_REGION --no-include-email)
  build:
    commands:
      - mvn -B clean install docker:build
      - docker tag $DOCKER_IMAGE_NAME:latest $REPOSITORY_URI:latest
  post_build:
    commands:
      - docker push $REPOSITORY_URI:latest
```

# Infrastructure-as-Code

## Automated creation of the build pipeline



# Build Pipeline

## CloudFormation template snippet (details omitted)

```
CodePipeline:
    Type: "AWS::CodePipeline::Pipeline"
    Properties:
      Stages:
        - Actions:
            - ActionTypeId:
                Provider: CodeCommit
              Configuration:
                RepositoryName: !Ref CodeCommitRepositoryName
          Name: Source
          OutputArtifacts:
                - Name: SourceArtifact
        - Actions:
            - ActionTypeId:
                Provider: CodeBuild
              Configuration:
                ProjectName: !Ref CodeBuildProjectName
              InputArtifacts:
                - Name: SourceArtifact
          Name: Build
          …
```

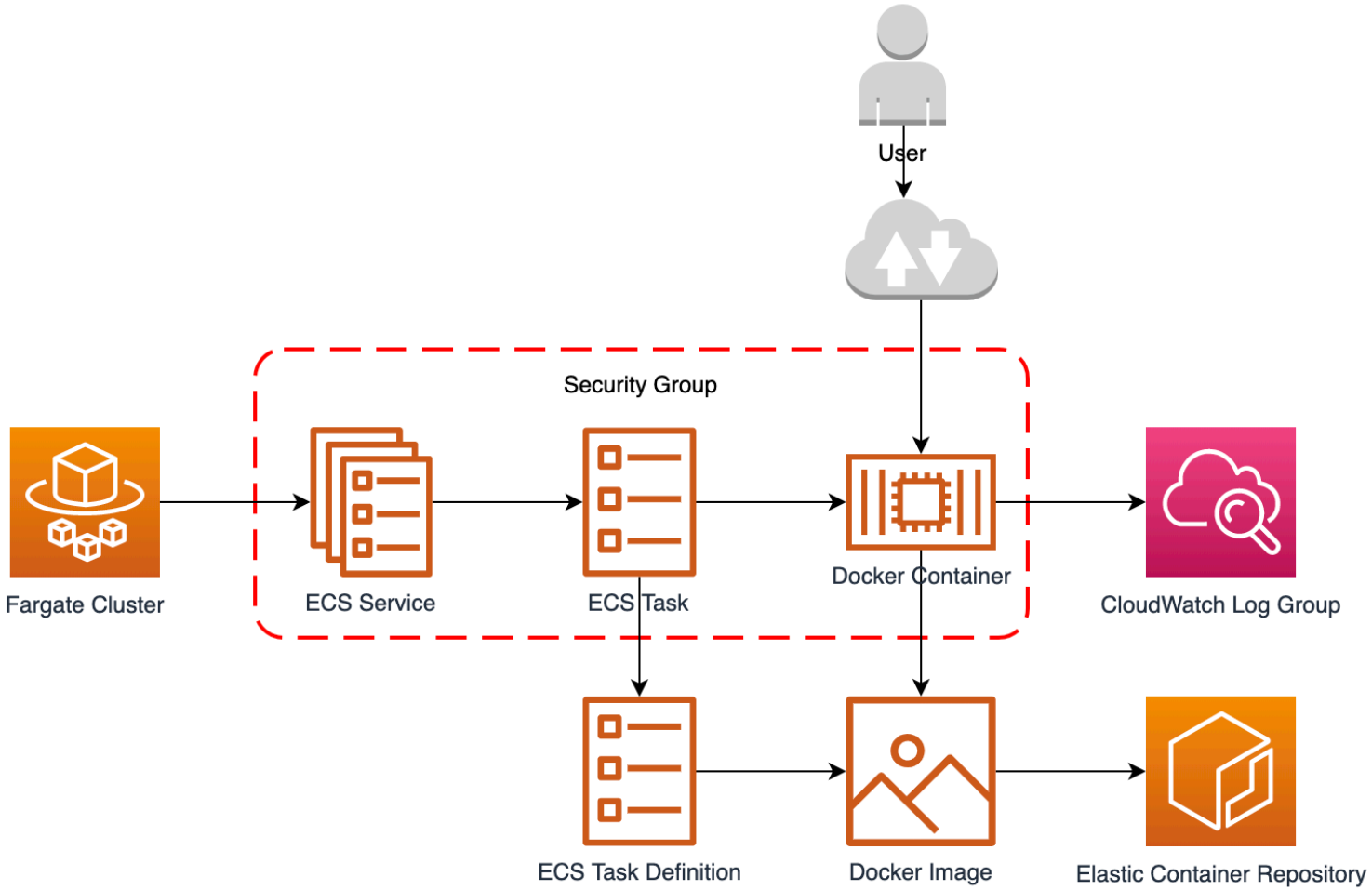
# Build Pipeline

## Let's create the build pipeline stack and trigger the first build


```
aws cloudformation create-stack --stack-name pipeline \
    --capabilities CAPABILITY_IAM \
    --template-body file://./cloudformation/pipeline.yaml \
    --parameters    ParameterKey=CodeCommitRepositoryName,ParameterValue=helloworld \
                    ParameterKey=CodeBuildProjectName,ParameterValue=helloworld \
                    ParameterKey=CodePipelineName,ParameterValue=helloworld \
                    ParameterKey=ECRRepositoryName,ParameterValue=helloworld

aws cloudformation wait stack-create-complete --stack-name pipeline

GIT_CLONE_URL=$(aws cloudformation describe-stacks --stack-name pipeline \
    --query 'Stacks[].Outputs[?OutputKey==`CodeCommitRepositoryCloneURL`].OutputValue' --output text)

git clone $GIT_CLONE_URL helloworld
tar xvfz helloworld.tgz
cd helloworld
git add .
git commit -a -m "initial"
git push
```

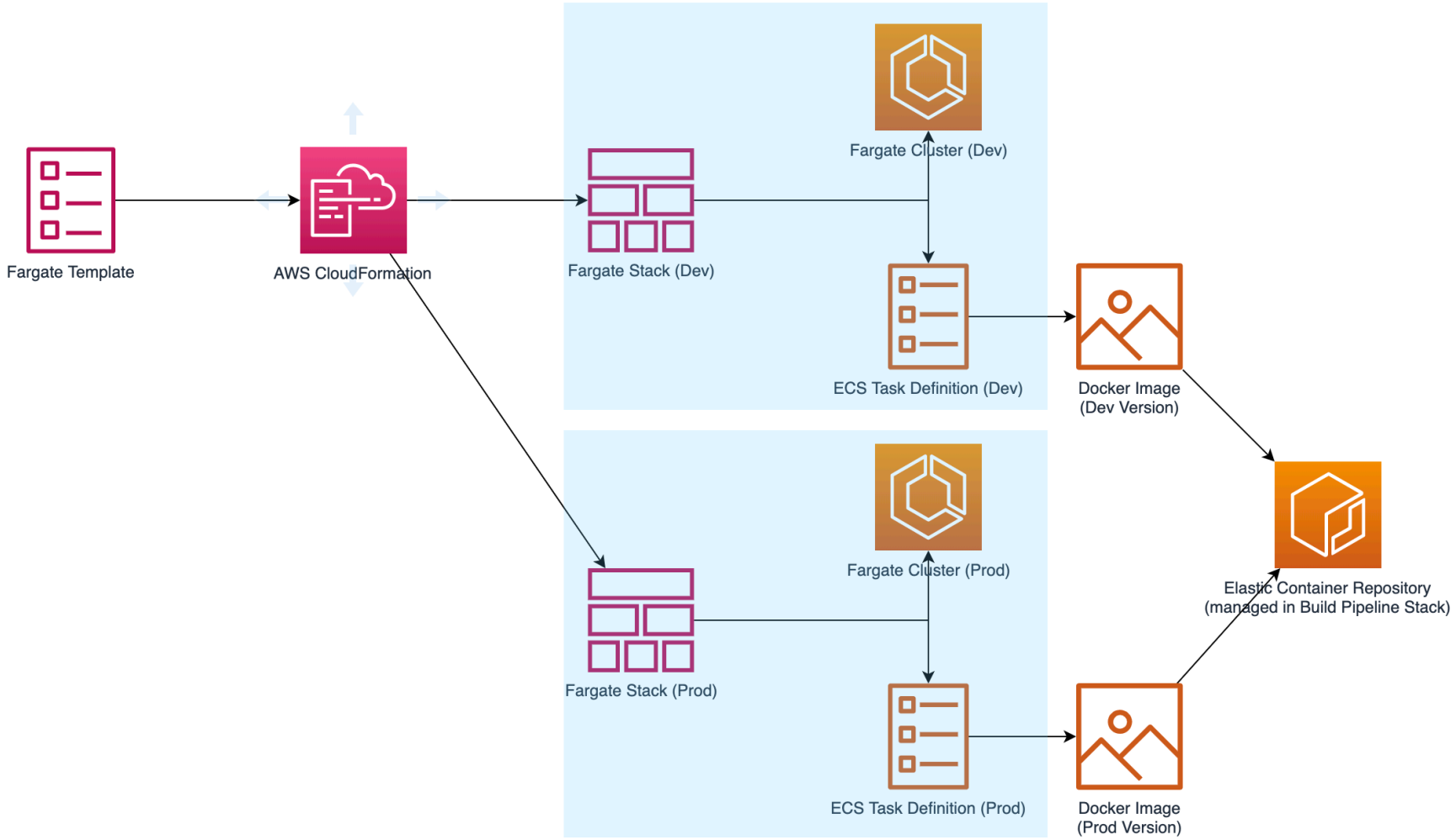
# Managed container runtime environment

## Run dockerized stateless apps (e.g. Java, .NET Core, nodejs) in Fargate



# Infrastructure-as-Code

## Automated creation of runtime environments



# Fargate runtime environment

## CloudFormation template snippet (details omitted)

```
TaskDefinition:
    Type: "AWS::ECS::TaskDefinition"
    Properties:
      ContainerDefinitions:
        - Cpu: "1024"
          Image: !Ref: DockerImageURL
          LogConfiguration:
            LogDriver: awslogs
            Options:
              awslogs-group: !Ref TaskLogGroup
              awslogs-region: !Ref "AWS::Region”
              awslogs-stream-prefix: ecs
          Memory: "2048"
          Name: !Ref FargateServiceName
          PortMappings:
            - ContainerPort: 8080
              Protocol: tcp
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      …
```

# Fargate runtime environment

## Let's create a Fargate dev environment (stack)


```
aws cloudformation create-stack --stack-name fargate \
    --template-body file://./cloudformation/fargate.yaml \
    --capabilities CAPABILITY_IAM \
    --parameters    ParameterKey=Subnets,ParameterValue=\"$SUBNET_IDS\" \
                    ParameterKey=VPC,ParameterValue=$DEFAULT_VPC_ID \
                    ParameterKey=FargateServiceName,ParameterValue=helloworld \
                    ParameterKey=ECRRepositoryName,ParameterValue=helloworld

aws cloudformation wait stack-create-complete --stack-name fargate
```

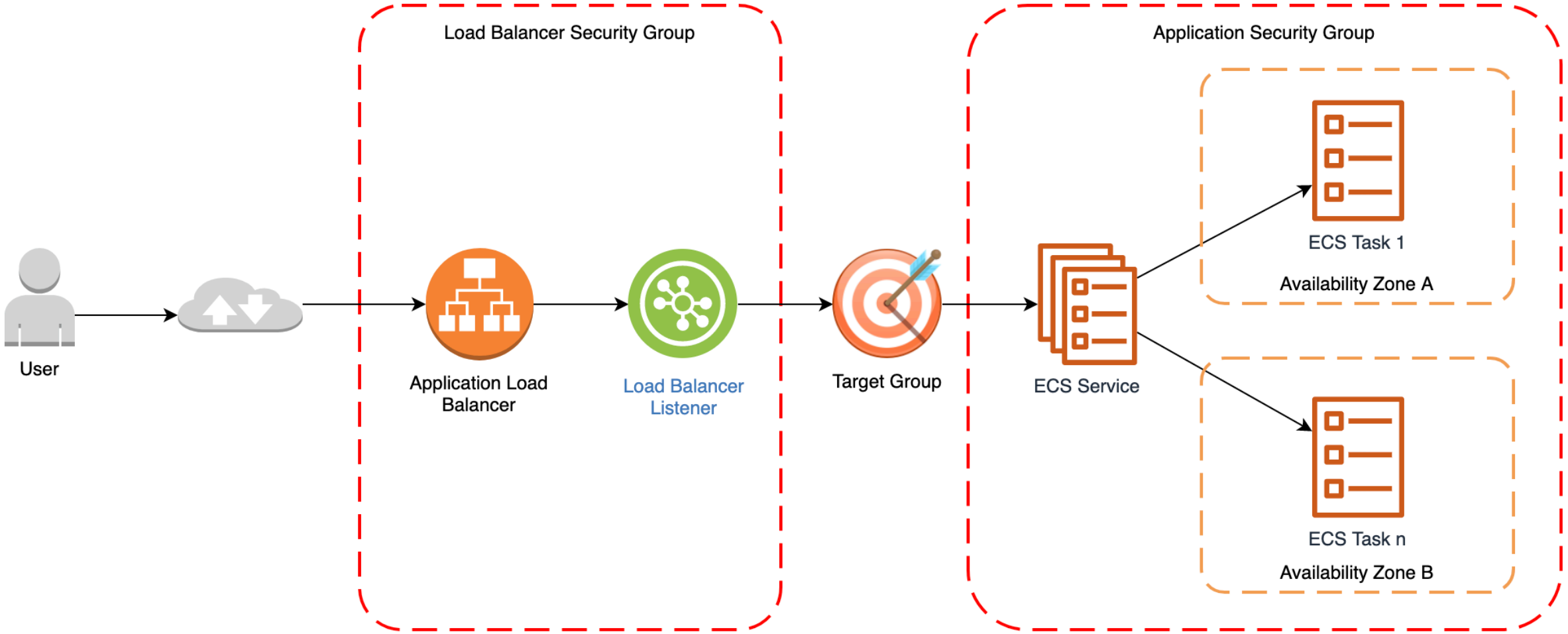
# Your app is up and running

## Let's test


```
TASK_ID=$(aws ecs list-tasks --service helloworld --cluster helloworld --output text --query "taskArns")

ENI_ID=$(aws ecs describe-tasks --cluster helloworld --tasks $TASK_ID --query
'tasks[].attachments[].details[?name==`networkInterfaceId`].value' --output text)

PUBLIC_DNS=$(aws ec2 describe-network-interfaces --query
"NetworkInterfaces[?NetworkInterfaceId==\`$ENI_ID\`].Association.PublicIp" --output text)

curl http://$PUBLIC_DNS:8080
```

# Less hello- and more real-world

## Scalability and Reliability



# Application Load Balancer

## CloudFormation template snippet (details omitted)


```
…

LoadBalancer:
    Type: "AWS::ElasticLoadBalancingV2::LoadBalancer”
    Properties:
      Scheme: internet-facing
      SecurityGroups:
        - !Ref LoadBalancerSecurityGroup
      Subnets: !Ref Subnets

Listener:
    Type: "AWS::ElasticLoadBalancingV2::Listener”
    Properties:
      DefaultActions:
        - TargetGroupArn: !Ref TargetGroup
          Type: forward
      LoadBalancerArn: !Ref LoadBalancer
      Port: 80
      Protocol: HTTP

TargetGroup:
    Type: "AWS::ElasticLoadBalancingV2::TargetGroup"
    Properties: …
```

# Security Group Ingress Rules

## CloudFormation template snippet (details omitted)


```
LoadBalancerSecurityGroup:
    Type: "AWS::EC2::SecurityGroup"
    Properties:
      GroupDescription: HTTP 80
      SecurityGroupIngress:
        - CidrIp: "0.0.0.0/0"
          FromPort: 80
          IpProtocol: tcp
          ToPort: 80
      VpcId: !Ref VPC

FargateSecurityGroup:
   Type: "AWS::EC2::SecurityGroup"
    Properties:
      GroupDescription: HTTP 8080
      SecurityGroupIngress:
        - FromPort: 8080
          IpProtocol: tcp
          SourceSecurityGroupId: !GetAtt LoadBalancerSecurityGroup.GroupId
          ToPort: 8080
      VpcId: !Ref VPC
```

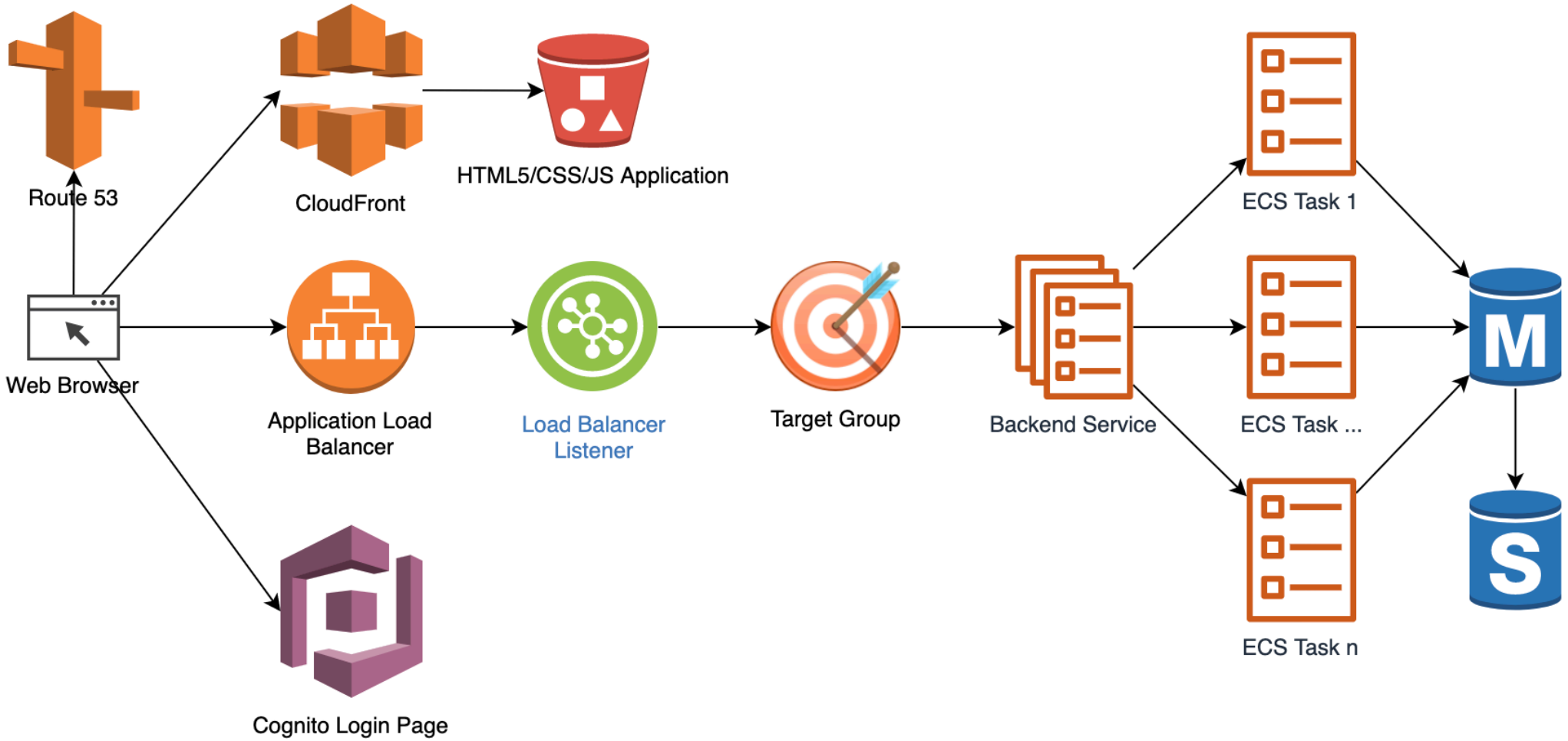
# Iterative infrastructure development

## Let's add a Load Balancer to our existing Fargate (stack)


```
vi ./cloudformation/fargate.yaml
…

aws cloudformation update-stack --stack-name fargate \
    --template-body file://./cloudformation/fargate.yaml \
    --capabilities CAPABILITY_IAM \
    --parameters    ParameterKey=Subnets, UsePreviousValue=true \
                    ParameterKey=VPC,UsePreviousValue=true \
                    ParameterKey=FargateServiceName,UsePreviousValue=true \
                    ParameterKey=ECRRepositoryName,UsePreviousValue=true

aws cloudformation wait stack-update-complete --stack-name fargate
```

# Welcome to the Cloud

## Tons of managed services at your disposal!



# Tl;dr

https://github.com/chtzuehlke/hslu-helloworld (not for production!)

zühlke
empowering ideas

# Thanks for listening!

# Questions?

https://twitter.com/cht_z

btw., Zühlke is hiring Software Engineers ;-)


zühlke
empowering ideas

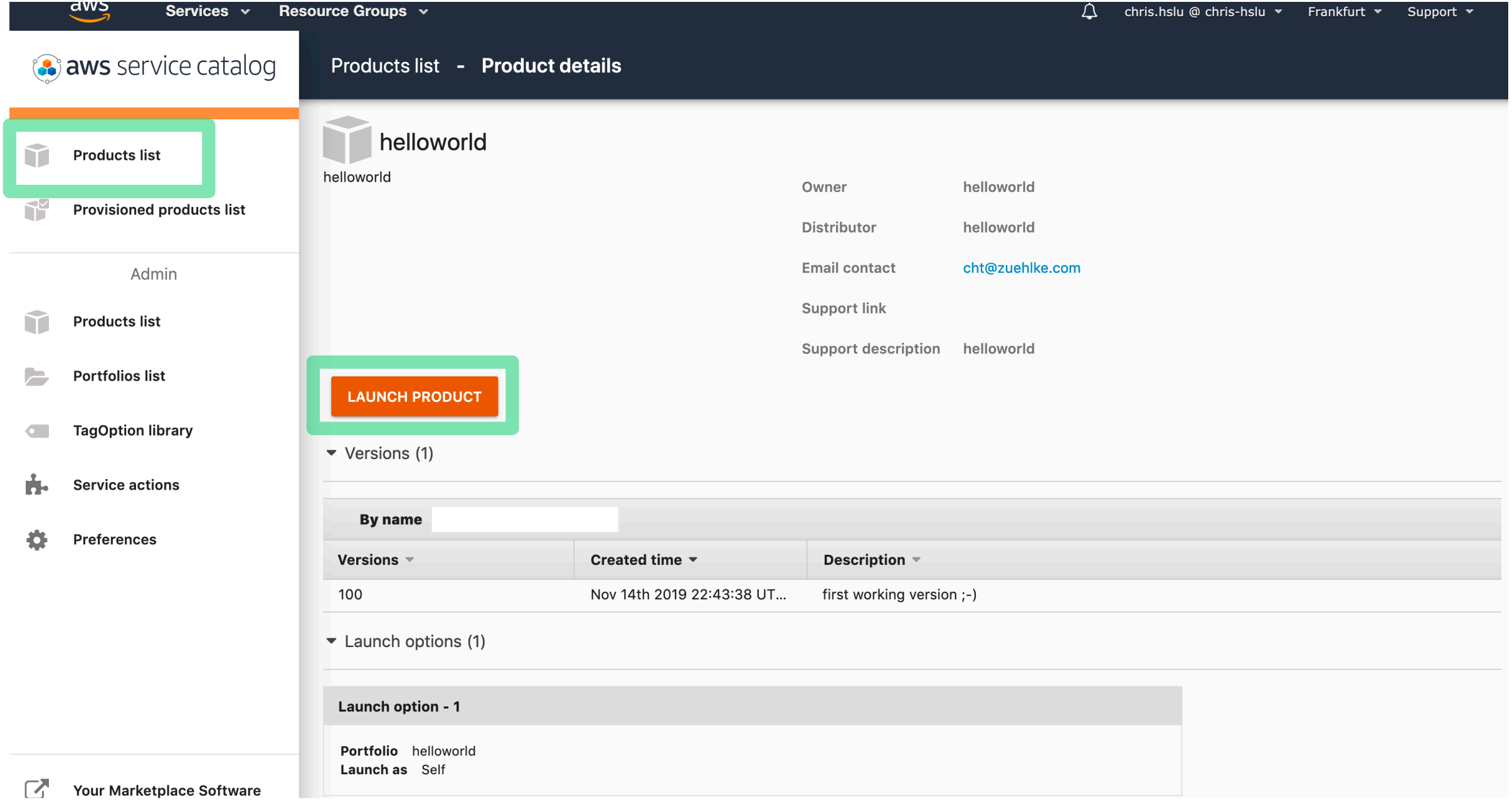
**Christian Tschenett**
Principal Consultant

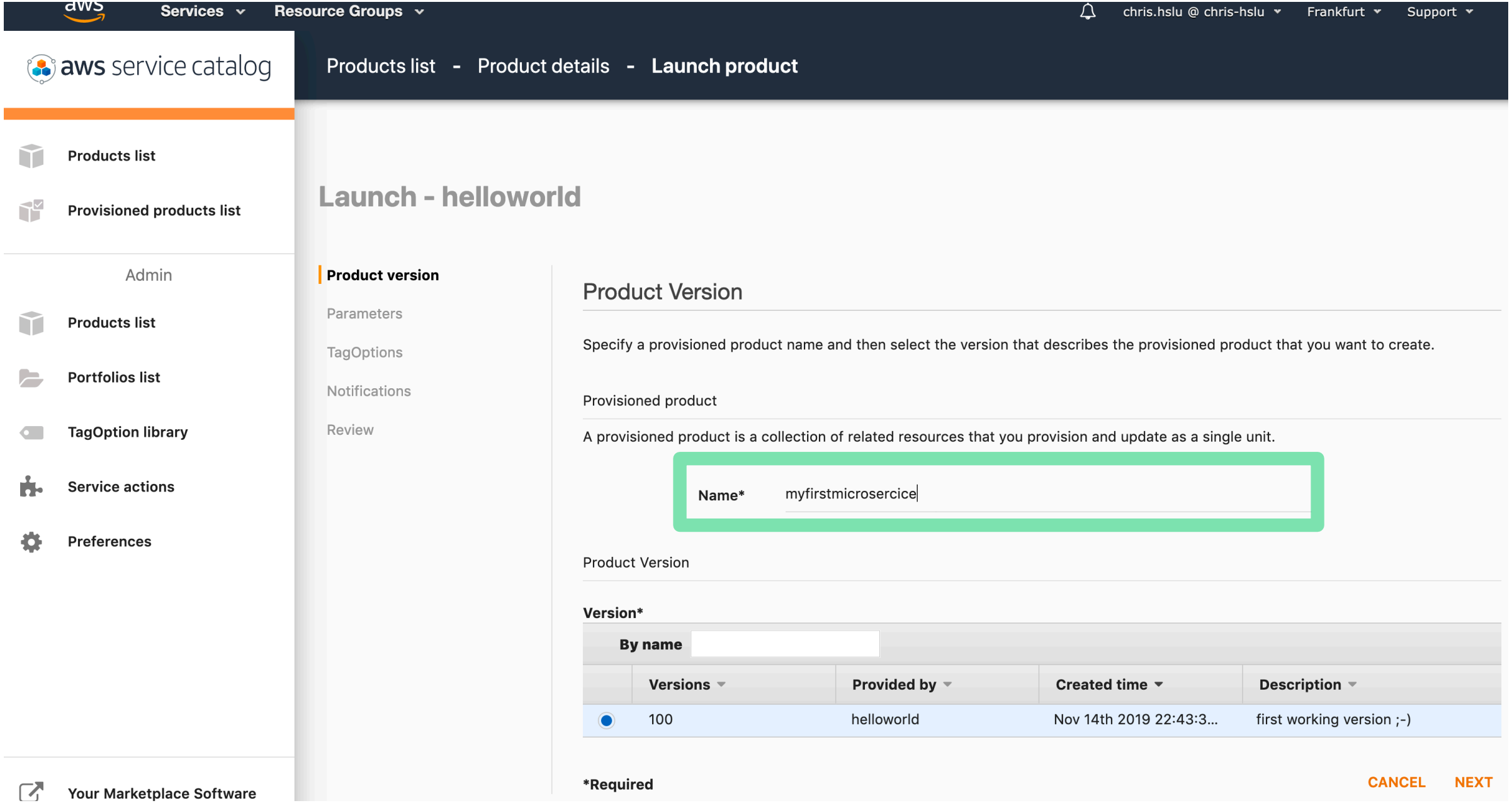
Zühlke Engineering AG
Wiesenstrasse 10a
8952 Schlieren (Zürich)
Schweiz

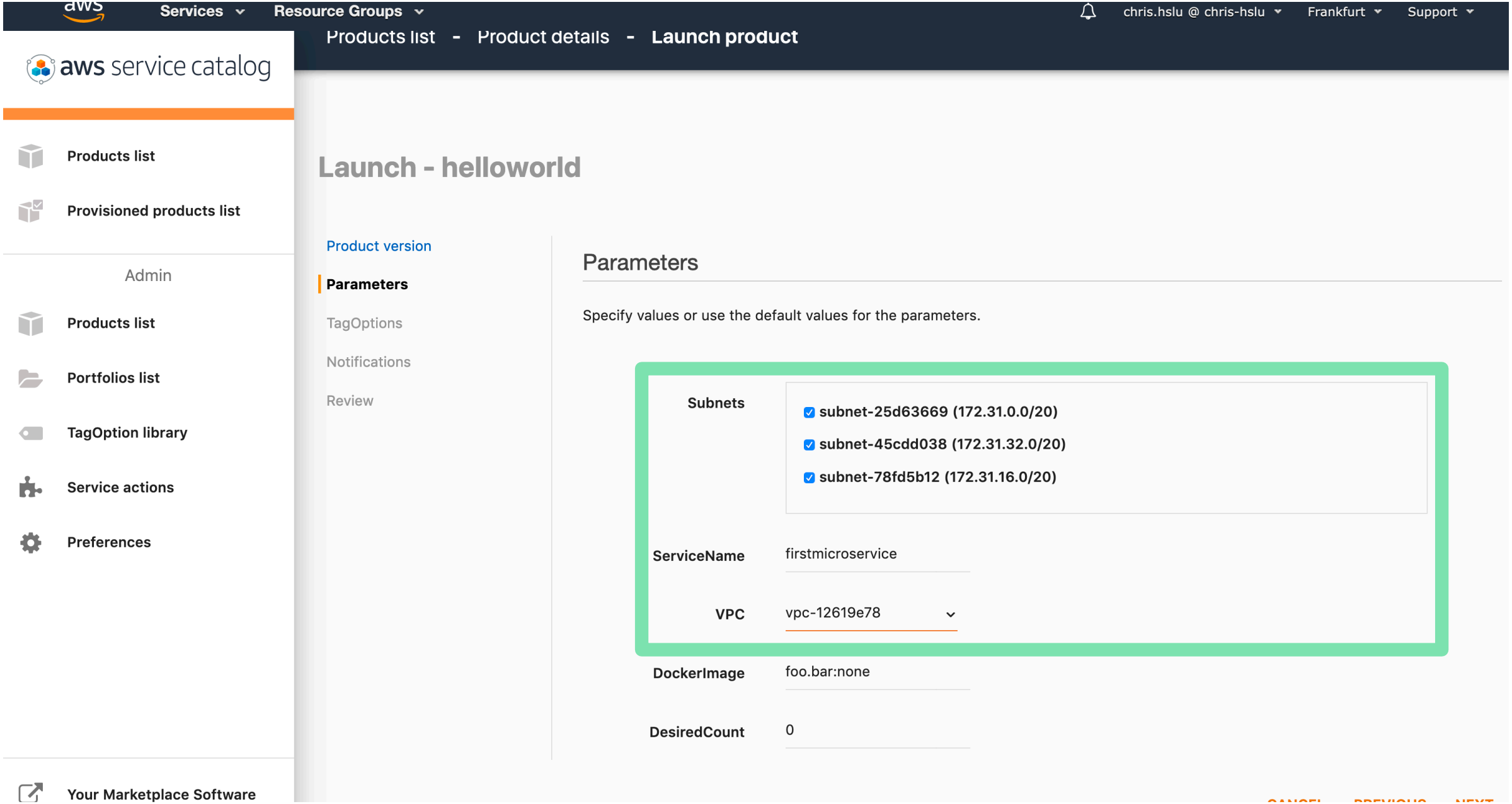
Telefon +41 43 216 6611
Direkt +41 43 216 6680
christian.tschenett@zuehlke.com

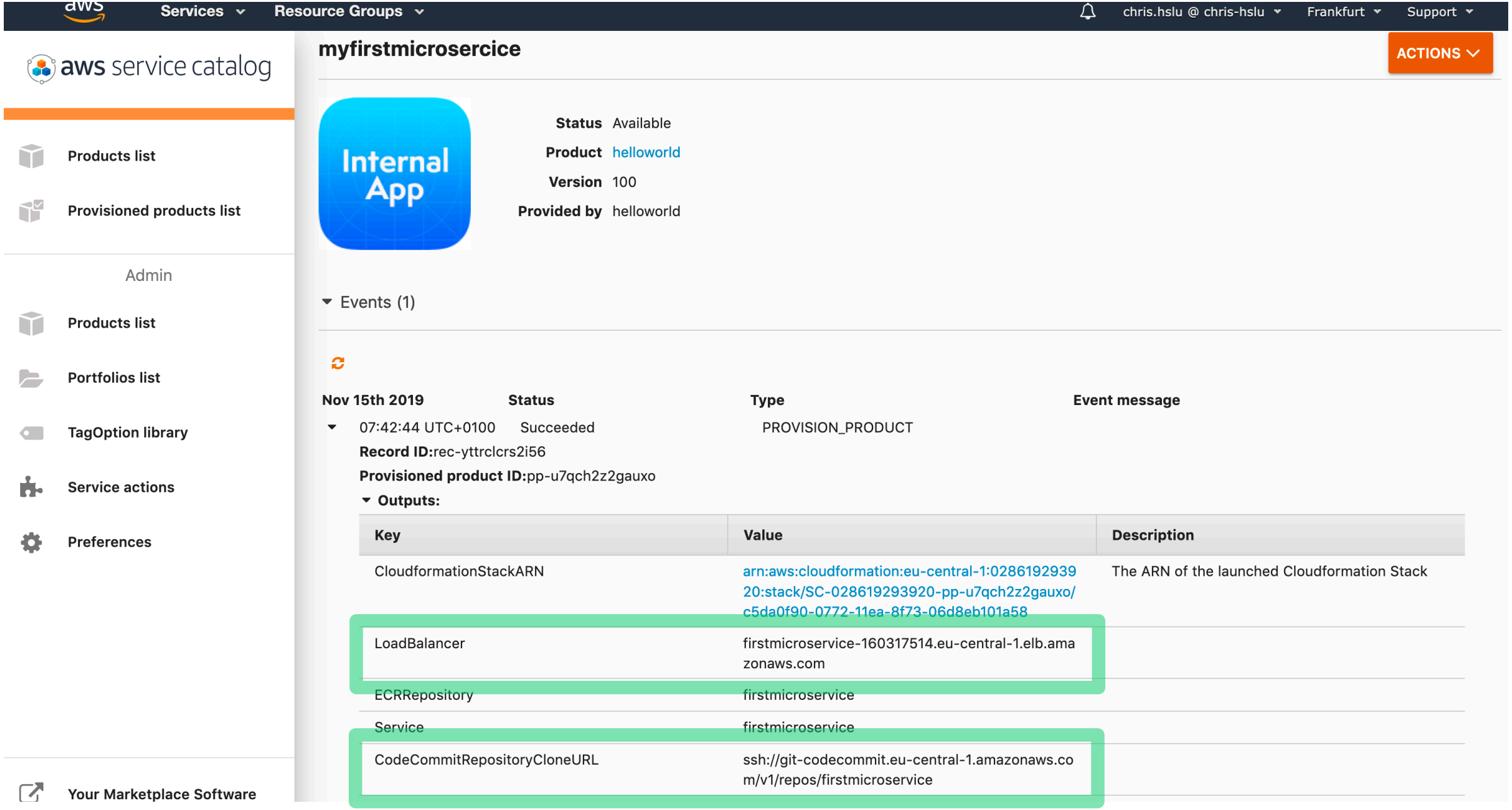
zuehlke.com

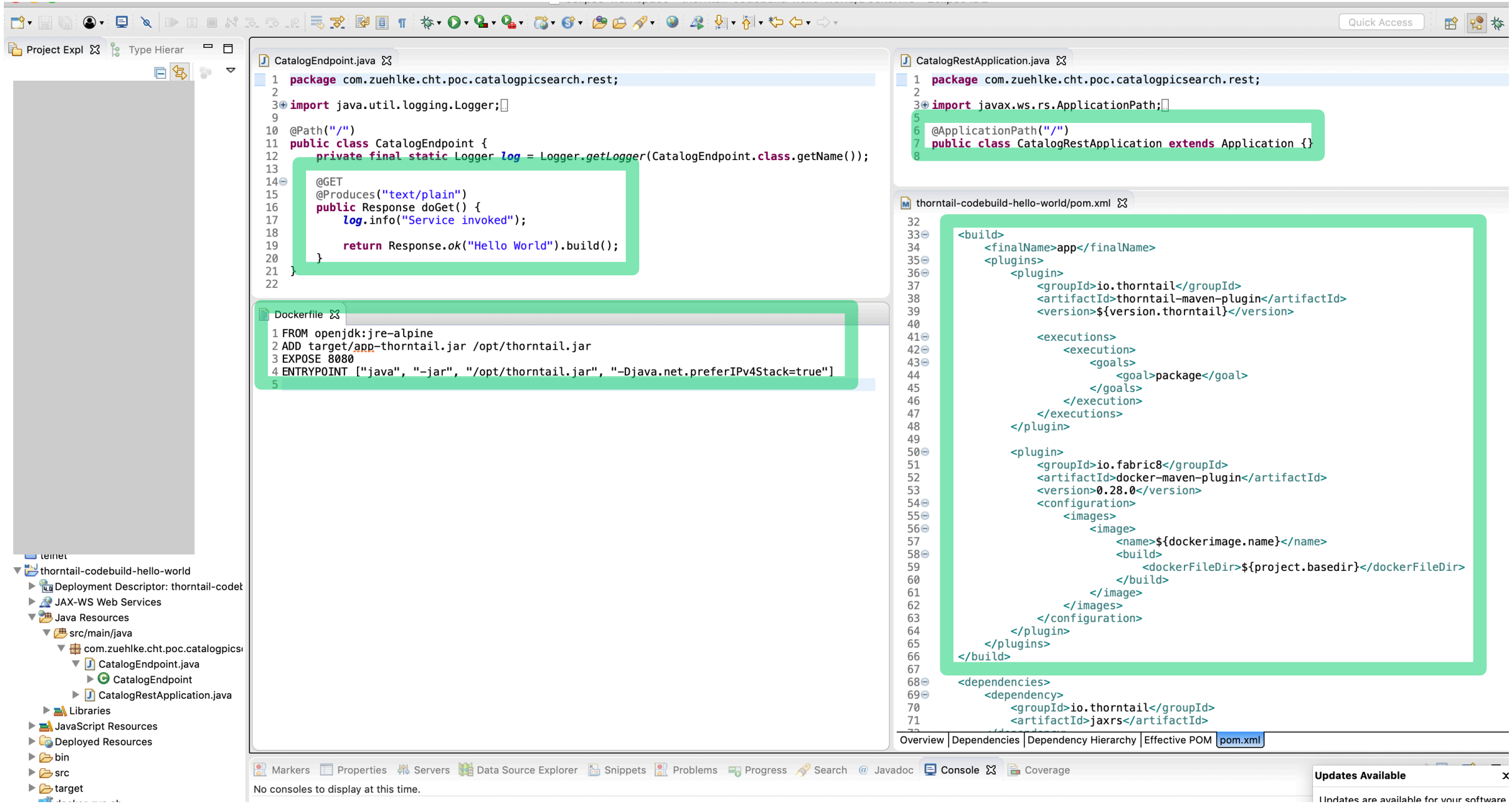
# Appendix A
# AWS Service Catalog – from zero to prod in 20’

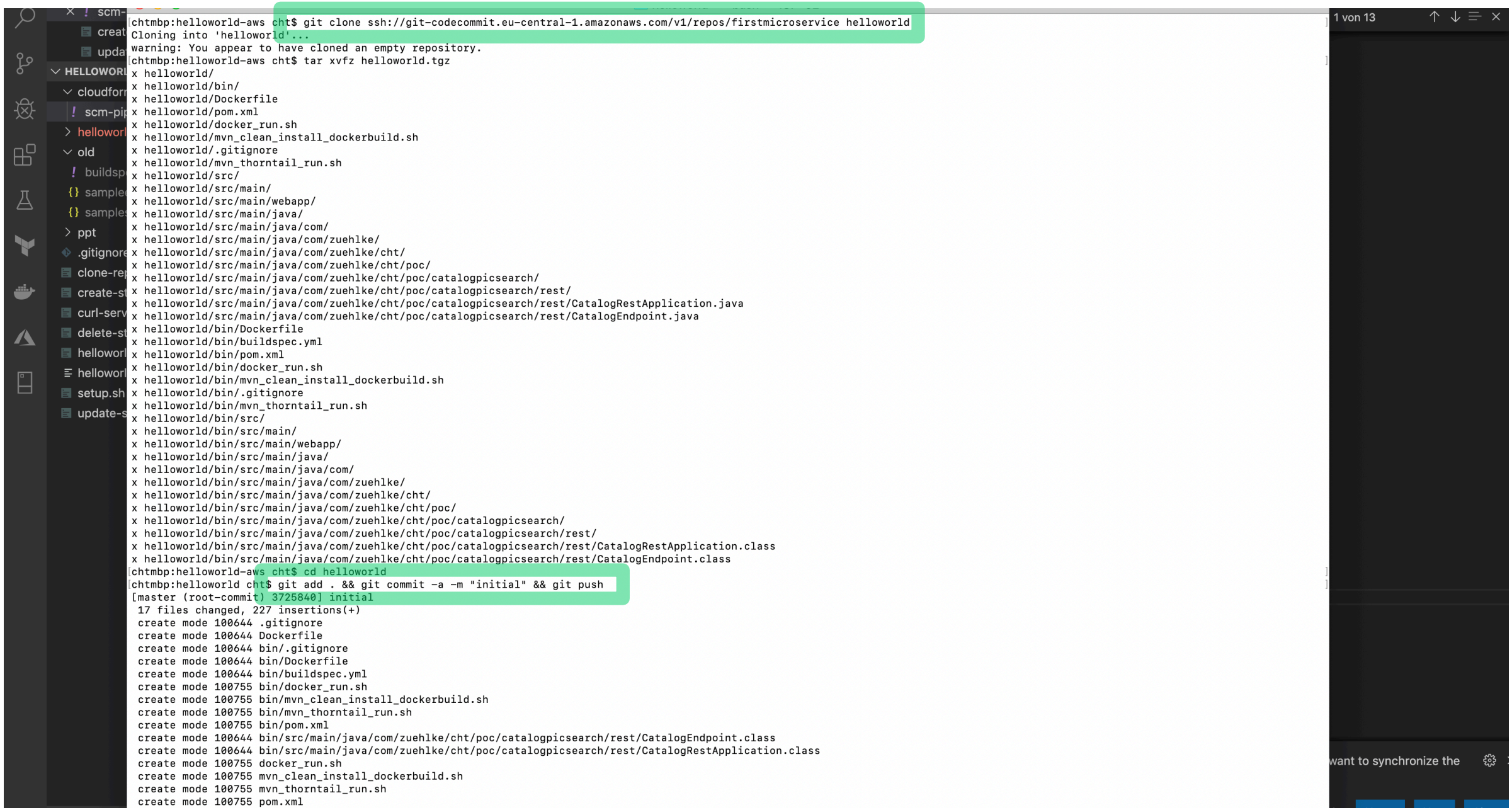

Services
Resource Groups
chris.hslu @ chris-hslu
Frankfurt
Support
aws service catalog
Products list - Product details
Products list
Provisioned products list
Admin
Products list
Portfolios list
TagOption library
Service actions
Preferences
Your Marketplace Software
helloworld
helloworld
Owner helloworld
Distributor helloworld
Email contact cht@zuehlke.com
Support link
Support description helloworld
LAUNCH PRODUCT
Versions (1)
By name
Versions
Created time
Description
100
Nov 14th 2019 22:43:38 UT...
first working version ;-)
Launch options (1)
Launch option - 1
Portfolio helloworld
Launch as Self

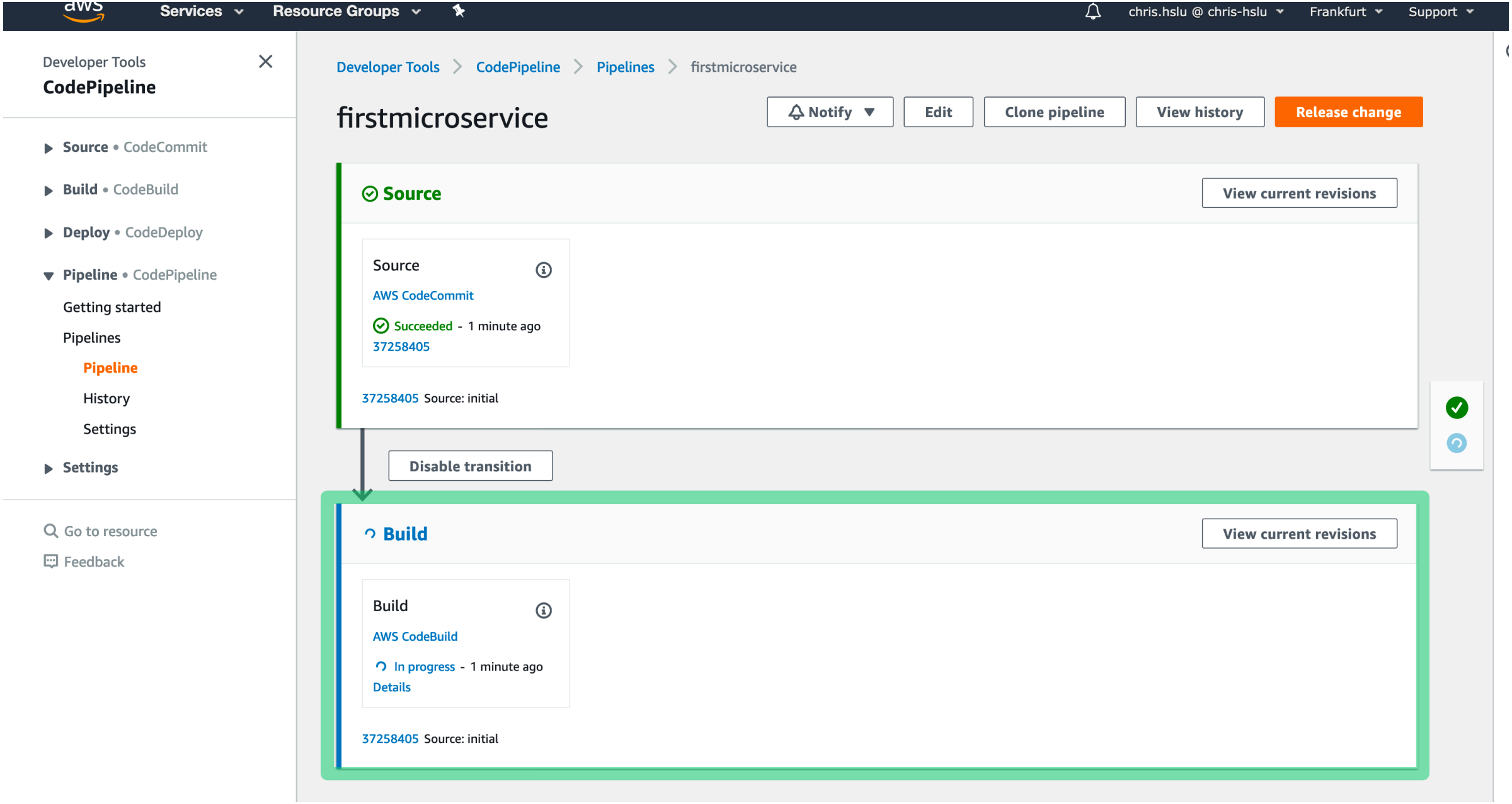

Services
Resource Groups
chris.hslu @ chris-hslu
Frankfurt
Support
aws service catalog
Products list - Product details - Launch product
Products list
Provisioned products list
Admin
Products list
Portfolios list
TagOption library
Service actions
Preferences
Your Marketplace Software
Launch - helloworld
Product version
Parameters
TagOptions
Notifications
Review
Product Version
Specify a provisioned product name and then select the version that describes the provisioned product that you want to create.
Provisioned product
A provisioned product is a collection of related resources that you provision and update as a single unit.
Name*
myfirstmicrosercice
Product Version
Version*
By name
Versions
Provided by
Created time
Description
100
helloworld
Nov 14th 2019 22:43:3...
first working version ;-)
*Required
CANCEL
NEXT

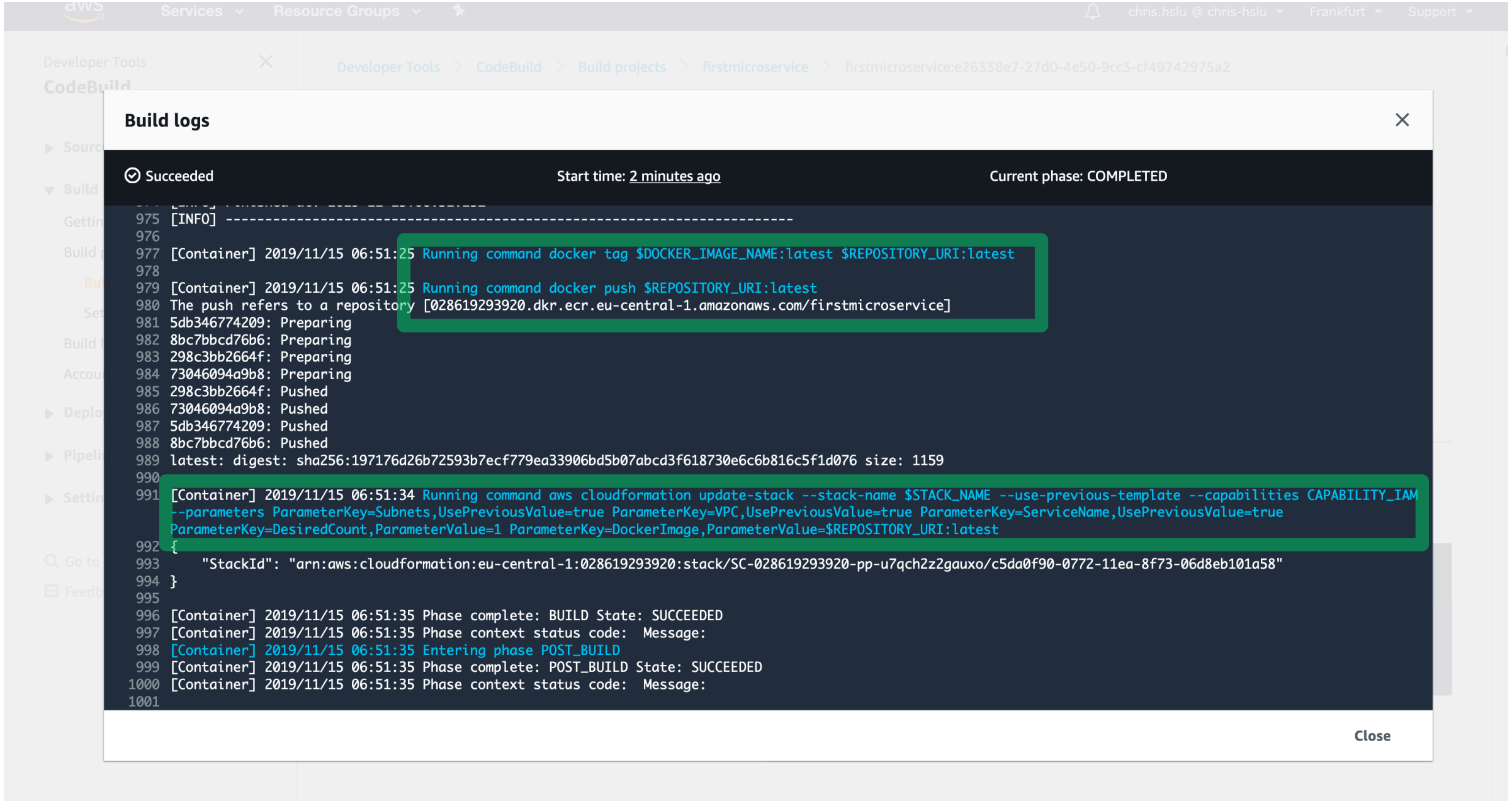

Services
Resource Groups
chris.hslu @ chris-hslu
Frankfurt
Support
aws service catalog
Products list - Product details - Launch product
Products list
Provisioned products list
Admin
Products list
Portfolios list
TagOption library
Service actions
Preferences
Your Marketplace Software
Launch - helloworld
Product version
Parameters
TagOptions
Notifications
Review
Parameters
Specify values or use the default values for the parameters.
Subnets
subnet-25d63669 (172.31.0.0/20)
subnet-45cdd038 (172.31.32.0/20)
subnet-78fd5b12 (172.31.16.0/20)
ServiceName
firstmicroservice
VPC
vpc-12619e78
DockerImage
foo.bar:none
DesiredCount
0

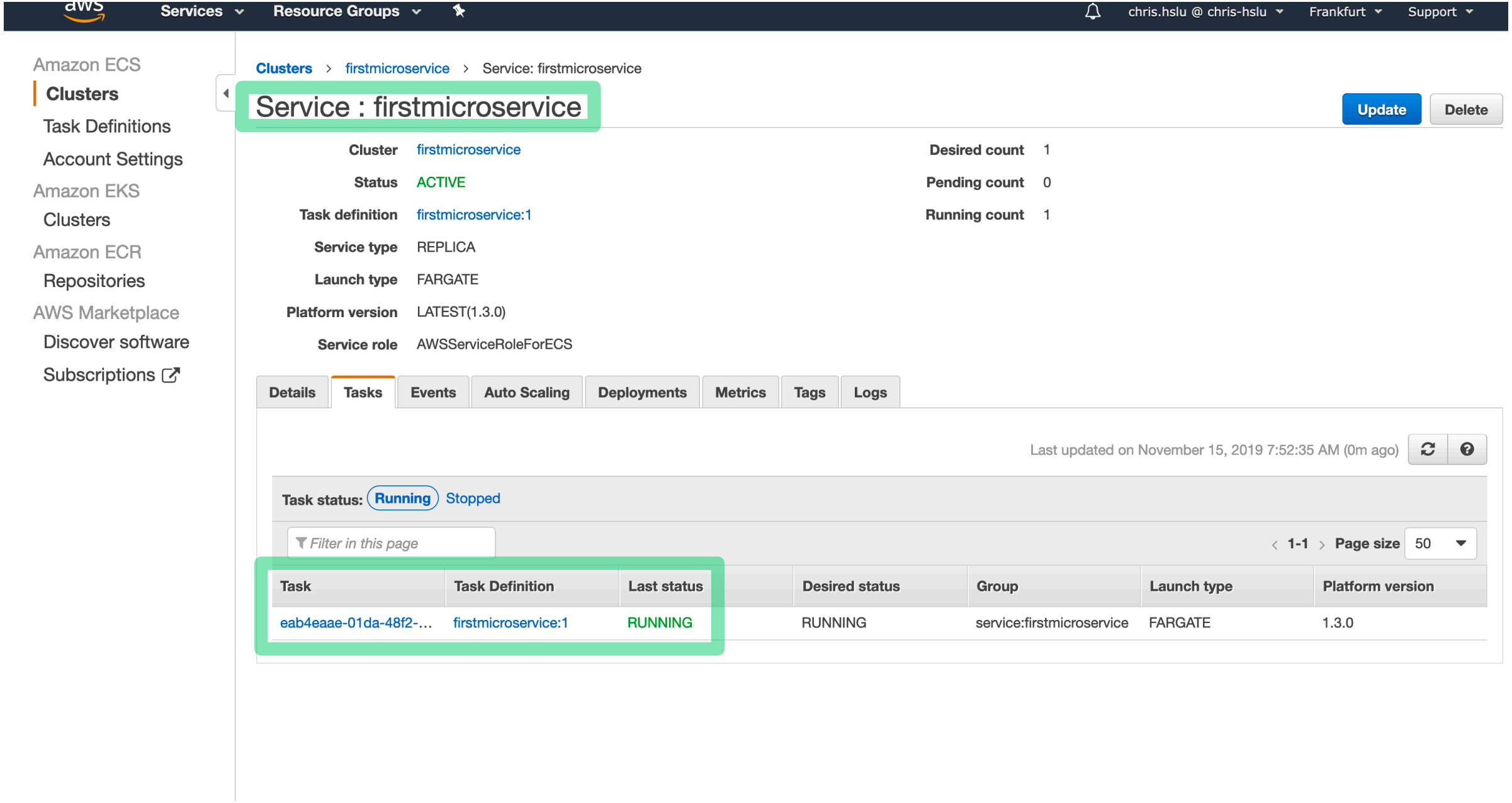

Services
Resource Groups
chris.hslu @ chris-hslu
Frankfurt
Support
aws service catalog
Products list
Provisioned products list
Admin
Products list
Portfolios list
TagOption library
Service actions
Preferences
Your Marketplace Software
myfirstmicrosercice
ACTIONS
Internal App
Status Available
Product helloworld
Version 100
Provided by helloworld
Events (1)
Nov 15th 2019
Status
Type
Event message
07:42:44 UTC+0100
Succeeded
PROVISION_PRODUCT
Record ID:rec-yttrclcrs2i56
Provisioned product ID:pp-u7qch2z2gauxo
Outputs:
Key
Value
Description
CloudformationStackARN
arn:aws:cloudformation:eu-central-1:028619293920:stack/SC-028619293920-pp-u7qch2z2gauxo/c5da0f90-0772-11ea-8f73-06d8eb101a58
The ARN of the launched Cloudformation Stack
LoadBalancer
firstmicroservice-160317514.eu-central-1.elb.amazonaws.com
ECRRepository
firstmicroservice
Service
firstmicroservice
CodeCommitRepositoryCloneURL
ssh://git-codecommit.eu-central-1.amazonaws.com/v1/repos/firstmicroservice

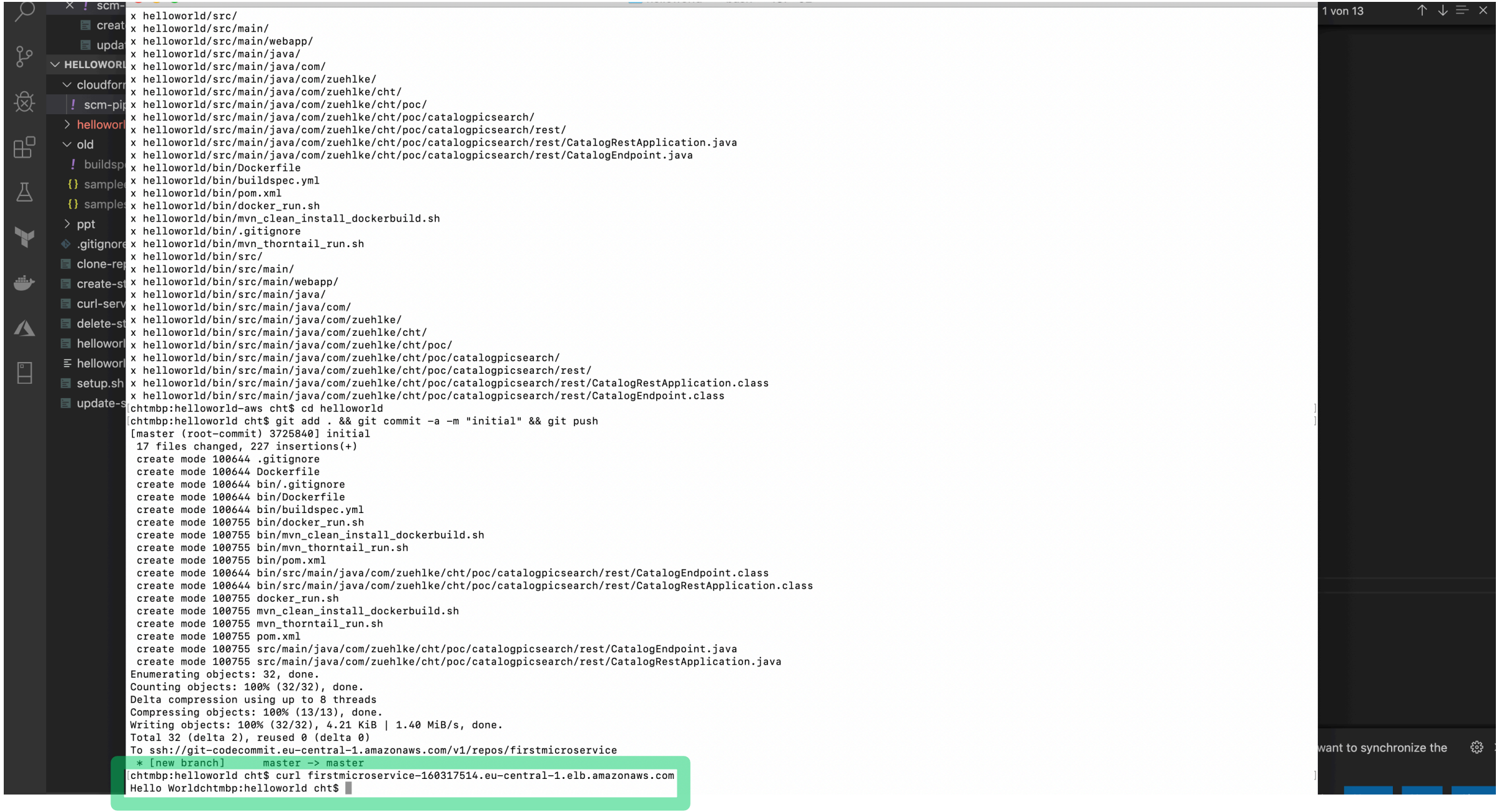

Project Expl
Type Hierar
Quick Access
CatalogEndpoint.java
package com.zuehlke.cht.poc.catalogpicsearch.rest;
import java.util.logging.Logger;
@Path("/")
public class CatalogEndpoint {
private final static Logger log = Logger.getLogger(CatalogEndpoint.class.getName());
@GET
@Produces("text/plain")
public Response doGet() {
log.info("Service invoked");
return Response.ok("Hello World").build();
}
}
CatalogRestApplication.java
package com.zuehlke.cht.poc.catalogpicsearch.rest;
import javax.ws.rs.ApplicationPath;
@ApplicationPath("/")
public class CatalogRestApplication extends Application {}
Dockerfile
FROM openjdk:jre-alpine
ADD target/app-thorntail.jar /opt/thorntail.jar
EXPOSE 8080
ENTRYPOINT ["java", "-jar", "/opt/thorntail.jar", "-Djava.net.preferIPv4Stack=true"]
thorntail-codebuild-hello-world/pom.xml
<build>
<finalName>app</finalName>
<plugins>
<plugin>
<groupId>io.thorntail</groupId>
<artifactId>thorntail-maven-plugin</artifactId>
<version>${version.thorntail}</version>
<executions>
<execution>
<goals>
<goal>package</goal>
</goals>
</execution>
</executions>
</plugin>
<plugin>
<groupId>io.fabric8</groupId>
<artifactId>docker-maven-plugin</artifactId>
<version>0.28.0</version>
<configuration>
<images>
<image>
<name>${dockerimage.name}</name>
<build>
<dockerFileDir>${project.basedir}</dockerFileDir>
</build>
</image>
</images>
</configuration>
</plugin>
</plugins>
</build>
<dependencies>
<dependency>
<groupId>io.thorntail</groupId>
<artifactId>jaxrs</artifactId>
Overview
Dependencies
Dependency Hierarchy
Effective POM
pom.xml
thorntail-codebuild-hello-world
Deployment Descriptor: thorntail-codel
JAX-WS Web Services
Java Resources
src/main/java
com.zuehlke.cht.poc.catalogpics
CatalogEndpoint.java
CatalogEndpoint
CatalogRestApplication.java
Libraries
JavaScript Resources
Deployed Resources
bin
src
target
Markers
Properties
Servers
Data Source Explorer
Snippets
Problems
Progress
Search
Javadoc
Console
Coverage
No consoles to display at this time.
Updates Available


chtmbp:helloworld-aws cht$ git clone ssh://git-codecommit.eu-central-1.amazonaws.com/v1/repos/firstmicroservice helloworld
Cloning into 'helloworld'...
warning: You appear to have cloned an empty repository.
chtmbp:helloworld-aws cht$ tar xvfz helloworld.tgz
x helloworld/
x helloworld/bin/
x helloworld/Dockerfile
x helloworld/pom.xml
x helloworld/docker_run.sh
x helloworld/mvn_clean_install_dockerbuild.sh
x helloworld/.gitignore
x helloworld/mvn_thorntail_run.sh
x helloworld/src/
x helloworld/src/main/
x helloworld/src/main/webapp/
x helloworld/src/main/java/
x helloworld/src/main/java/com/
x helloworld/src/main/java/com/zuehlke/
x helloworld/src/main/java/com/zuehlke/cht/
x helloworld/src/main/java/com/zuehlke/cht/poc/
x helloworld/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/
x helloworld/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/
x helloworld/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogRestApplication.java
x helloworld/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogEndpoint.java
x helloworld/bin/Dockerfile
x helloworld/bin/buildspec.yml
x helloworld/bin/pom.xml
x helloworld/bin/docker_run.sh
x helloworld/bin/mvn_clean_install_dockerbuild.sh
x helloworld/bin/.gitignore
x helloworld/bin/mvn_thorntail_run.sh
x helloworld/bin/src/
x helloworld/bin/src/main/
x helloworld/bin/src/main/webapp/
x helloworld/bin/src/main/java/
x helloworld/bin/src/main/java/com/
x helloworld/bin/src/main/java/com/zuehlke/
x helloworld/bin/src/main/java/com/zuehlke/cht/
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogRestApplication.class
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogEndpoint.class
chtmbp:helloworld-aws cht$ cd helloworld
chtmbp:helloworld cht$ git add . && git commit -a -m "initial" && git push
[master (root-commit) 3725840] initial
17 files changed, 227 insertions(+)
create mode 100644 .gitignore
create mode 100644 Dockerfile
create mode 100644 bin/.gitignore
create mode 100644 bin/Dockerfile
create mode 100644 bin/buildspec.yml
create mode 100755 bin/docker_run.sh
create mode 100755 bin/mvn_clean_install_dockerbuild.sh
create mode 100755 bin/mvn_thorntail_run.sh
create mode 100755 bin/pom.xml
create mode 100644 bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogEndpoint.class
create mode 100644 bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogRestApplication.class
create mode 100755 docker_run.sh
create mode 100755 mvn_clean_install_dockerbuild.sh
create mode 100755 mvn_thorntail_run.sh
create mode 100755 pom.xml
1 von 13
want to synchronize the


Services
Resource Groups
chris.hslu @ chris-hslu
Frankfurt
Support
Developer Tools
CodePipeline
Source • CodeCommit
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Go to resource
Feedback
Developer Tools > CodePipeline > Pipelines > firstmicroservice
firstmicroservice
Notify
Edit
Clone pipeline
View history
Release change
Source
View current revisions
Source
AWS CodeCommit
Succeeded - 1 minute ago
37258405
37258405 Source: initial
Disable transition
Build
View current revisions
Build
AWS CodeBuild
In progress - 1 minute ago
Details
37258405 Source: initial


Services
Resource Groups
chris.hslu @ chris-hslu
Frankfurt
Support
Developer Tools
CodeBuild
Developer Tools > CodeBuild > Build projects > firstmicroservice > firstmicroservice:e26338e7-27d0-4e50-9cc3-cf49742975a2
Build logs
Succeeded
Start time: 2 minutes ago
Current phase: COMPLETED
975 [INFO] ------------------------------------------------------------------------
976
977 [Container] 2019/11/15 06:51:25 Running command docker tag $DOCKER_IMAGE_NAME:latest $REPOSITORY_URI:latest
978
979 [Container] 2019/11/15 06:51:25 Running command docker push $REPOSITORY_URI:latest
980 The push refers to a repository [028619293920.dkr.ecr.eu-central-1.amazonaws.com/firstmicroservice]
981 5db346774209: Preparing
982 8bc7bbcd76b6: Preparing
983 298c3bb2664f: Preparing
984 73046094a9b8: Preparing
985 298c3bb2664f: Pushed
986 73046094a9b8: Pushed
987 5db346774209: Pushed
988 8bc7bbcd76b6: Pushed
989 latest: digest: sha256:197176d26b72593b7ecf779ea33906bd5b07abcd3f618730e6c6b816c5f1d076 size: 1159
990
991 [Container] 2019/11/15 06:51:34 Running command aws cloudformation update-stack --stack-name $STACK_NAME --use-previous-template --capabilities CAPABILITY_IAM
--parameters ParameterKey=Subnets,UsePreviousValue=true ParameterKey=VPC,UsePreviousValue=true ParameterKey=ServiceName,UsePreviousValue=true
ParameterKey=DesiredCount,ParameterValue=1 ParameterKey=DockerImage,ParameterValue=$REPOSITORY_URI:latest
992 {
993 "StackId": "arn:aws:cloudformation:eu-central-1:028619293920:stack/SC-028619293920-pp-u7qch2z2gauxo/c5da0f90-0772-11ea-8f73-06d8eb101a58"
994 }
995
996 [Container] 2019/11/15 06:51:35 Phase complete: BUILD State: SUCCEEDED
997 [Container] 2019/11/15 06:51:35 Phase context status code: Message:
998 [Container] 2019/11/15 06:51:35 Entering phase POST_BUILD
999 [Container] 2019/11/15 06:51:35 Phase complete: POST_BUILD State: SUCCEEDED
1000 [Container] 2019/11/15 06:51:35 Phase context status code: Message:
1001
Close


Services
Resource Groups
chris.hslu @ chris-hslu
Frankfurt
Support
Amazon ECS
Clusters
Task Definitions
Account Settings
Amazon EKS
Clusters
Amazon ECR
Repositories
AWS Marketplace
Discover software
Subscriptions
Clusters > firstmicroservice > Service: firstmicroservice
Service : firstmicroservice
Update
Delete
Cluster firstmicroservice
Status ACTIVE
Task definition firstmicroservice:1
Service type REPLICA
Launch type FARGATE
Platform version LATEST(1.3.0)
Service role AWSServiceRoleForECS
Desired count 1
Pending count 0
Running count 1
Details
Tasks
Events
Auto Scaling
Deployments
Metrics
Tags
Logs
Last updated on November 15, 2019 7:52:35 AM (0m ago)
Task status: Running Stopped
Filter in this page
1-1
Page size 50
Task
Task Definition
Last status
Desired status
Group
Launch type
Platform version
eab4eaae-01da-48f2-...
firstmicroservice:1
RUNNING
RUNNING
service:firstmicroservice
FARGATE
1.3.0


x helloworld/src/
x helloworld/src/main/
x helloworld/src/main/webapp/
x helloworld/src/main/java/
x helloworld/src/main/java/com/
x helloworld/src/main/java/com/zuehlke/
x helloworld/src/main/java/com/zuehlke/cht/
x helloworld/src/main/java/com/zuehlke/cht/poc/
x helloworld/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/
x helloworld/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/
x helloworld/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogRestApplication.java
x helloworld/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogEndpoint.java
x helloworld/bin/Dockerfile
x helloworld/bin/buildspec.yml
x helloworld/bin/pom.xml
x helloworld/bin/docker_run.sh
x helloworld/bin/mvn_clean_install_dockerbuild.sh
x helloworld/bin/.gitignore
x helloworld/bin/mvn_thorntail_run.sh
x helloworld/bin/src/
x helloworld/bin/src/main/
x helloworld/bin/src/main/webapp/
x helloworld/bin/src/main/java/
x helloworld/bin/src/main/java/com/
x helloworld/bin/src/main/java/com/zuehlke/
x helloworld/bin/src/main/java/com/zuehlke/cht/
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogRestApplication.class
x helloworld/bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogEndpoint.class
chtmbp:helloworld-aws cht$ cd helloworld
chtmbp:helloworld cht$ git add . && git commit -a -m "initial" && git push
[master (root-commit) 3725840] initial
17 files changed, 227 insertions(+)
create mode 100644 .gitignore
create mode 100644 Dockerfile
create mode 100644 bin/.gitignore
create mode 100644 bin/Dockerfile
create mode 100644 bin/buildspec.yml
create mode 100755 bin/docker_run.sh
create mode 100755 bin/mvn_clean_install_dockerbuild.sh
create mode 100755 bin/mvn_thorntail_run.sh
create mode 100755 bin/pom.xml
create mode 100644 bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogEndpoint.class
create mode 100644 bin/src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogRestApplication.class
create mode 100755 docker_run.sh
create mode 100755 mvn_clean_install_dockerbuild.sh
create mode 100755 mvn_thorntail_run.sh
create mode 100755 pom.xml
create mode 100755 src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogEndpoint.java
create mode 100755 src/main/java/com/zuehlke/cht/poc/catalogpicsearch/rest/CatalogRestApplication.java
Enumerating objects: 32, done.
Counting objects: 100% (32/32), done.
Delta compression using up to 8 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (32/32), 4.21 KiB | 1.40 MiB/s, done.
Total 32 (delta 2), reused 0 (delta 0)
To ssh://git-codecommit.eu-central-1.amazonaws.com/v1/repos/firstmicroservice
* [new branch] master -> master
chtmbp:helloworld cht$ curl firstmicroservice-160317514.eu-central-1.elb.amazonaws.com
Hello Worldchtmbp:helloworld cht$
1 von 13
want to synchronize the