



# CI/CD pipeline

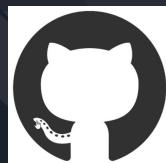
## Terraform-EKS-Jenkins

OLEG MANDRYCHENKO  
EPAM DevOps course winter 2020-21

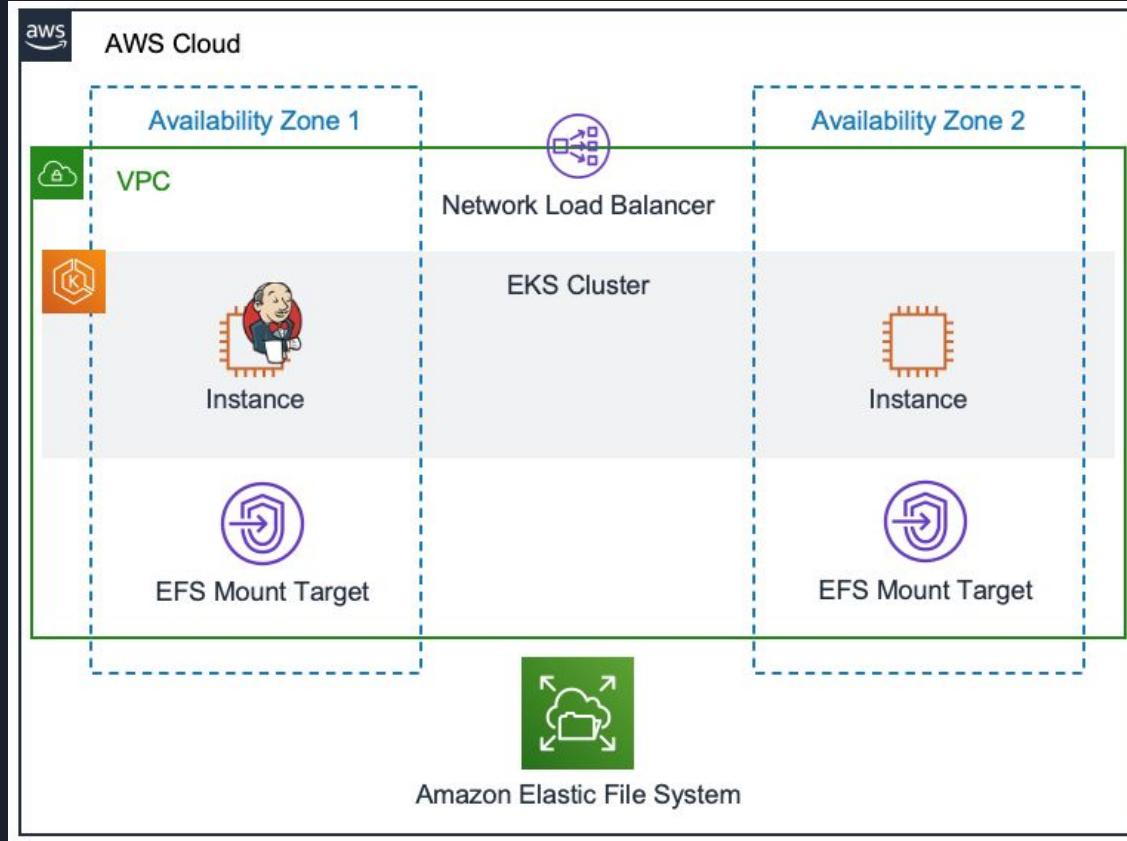


# Used Technologies and Tools

Git  
GitHub  
Docker  
Docker Hub  
Terraform  
Kubernetes  
AWS EKS  
Jenkins



# Scheme of task.





# Common Steps of Task

1. Provisioning EKS cluster by Terraform.
2. Setup EFS storage and mount it.
3. Setup storage for Jenkins and Deploy it.
4. Apply service for Jenkins.
5. Jenkins initial setup and configure.
6. Make a pipeline.
7. Add domain to EKS ELB through Route 53.

# Provisioning steps.

The screenshot shows a Visual Studio Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** README.md - project101-tf-eks-jenkins - Visual Studio Code [Administrator].
- Explorer:** Shows a tree view of files and folders:
  - OPEN EDITORS
  - PROJECT101-TF-EKS-JENKINS
    - jenkins
      - amazon-eks
      - jenkins.pv.yaml
      - jenkins.pvc.yaml
    - dockerfiles
    - jenkins.deployment...
    - jenkins.rbac.yaml
    - jenkins.service.yaml
    - kubernetes
    - configmaps
    - deployment
    - secrets
    - services
    - python
    - src
    - Dockerfile
    - .gitignore
    - data
    - main.tf
    - outputs.tf
    - variables.tf
  - README.md
- Terminal:** Shows the command `sh-4.2# yum install -y jq gzip nano tar git curl wget`.
- Code Editor:** Displays the content of the README.md file, which includes provisioning steps for an AWS EKS cluster using Jenkins. The code is highlighted in blue and black.
- Bottom Status Bar:** Includes icons for main, outline, timeline, and a status message: Ln 19, Col 1 (45 selected) Spaces: 4 CRLF Markdown Go Live.

# Provisioning steps.

The screenshot shows a Visual Studio Code interface with a dark theme. The left sidebar displays a file tree with files like `jenkins.pv.yaml`, `jenkins.pvc.yaml`, `jenkins.deployment.yaml`, `jenkins.rbac.yaml`, `jenkins.service.yaml`, `configmaps.yaml`, `deployments.yaml`, `secrets.yaml`, `services.yaml`, `Dockerfile`, `.gitignore`, `data`, `main.tf`, `outputs.tf`, `variables.tf`, and `README.md`. The `README.md` file is open in the main editor area, showing provisioning steps for an AWS EKS cluster using Terraform and AWS CLI. A large rectangular selection box highlights the provisioning steps for installing `kubectl`, `eksctl`, and `terraform`. The bottom right corner of the code editor shows a terminal window with the output of the provisioning commands, including the installation of `openssl` and `terraform`.

```
17
18 docker run -it --rm -v ${PWD}:/work -w /work --entrypoint /bin/sh amazon/aws-cli:latest
19 yum install -y jq gzip nano tar git curl wget
20
21
22 ### Install kubectl, eksctl, terraform
23
24
25 # Install kubectl
26 curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/
27 kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl
28 chmod +x ./kubectl && mv ./kubectl /usr/local/bin/kubectl
29
30 # Install eksctl
31 curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.
32 gz" | tar xz -C /tmp
33 mv /tmp/eksctl /usr/local/bin
34
35 # Install Terraform
36 yum install -y yum-utils
37 yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
38 yum -y install terraform
39
40 ### Configure AWS credentials and provisioning EKS cluster
41
42 # Configure your credentials from before created IAM account with all needed permissions
43 aws configure
44
45 # Init and Deploy Terraform infrastructure
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Verifying : 1:openssl-1.0.2k-19.amzn2.0.6.x86\_64

Installed:  
  terraform.x86\_64 0:0.14.8-1

Dependency Installed:  
  make.x86\_64 1:3.82-24.amzn2

Complete!  
sh-4.2#

1: docker

3/3

openssl.x86\_64 1:1.0.2k-19.amzn2.0.6

Ln 36, Col 1 (24 selected) Spaces: 4 UTF-8 CRLF Markdown Go Live

# Provisioning steps.

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure named "PROJECT101-TF-EKS-JENKINS". It includes subfolders like ".terraform", "jenkins" (containing "amazon-eks" with files "jenkins.pv.yaml", "jenkins.pvc.yaml"), "dockefiles", "jenkins.deployment...", "jenkins.rbac.yaml", "jenkins.service.yaml", "kubernetes" (containing "configmaps" with "configmapyaml", "deployments" with "deployment.yaml", "secrets" with "secret.yaml", "services" with "service.yaml"), "python", "src", "Dockerfile", ".gitignore", ".terraform.lock.hcl", "data", "kubecfg\_final-task...", "main.tf", "outputs.tf", "README.md", "terraform.tfstate", and "variables.tf".
- Terminal:** The terminal window displays the following provisioning steps:

```
 README.md - project101-tf-eks-jenkins - Visual Studio Code [Administrator]

# project101-tf-eks-jenkins > ## Final task EPAM DevOps course winter 2020-2021 > ### Setup our Cloud Storage
30 yum -y install terraform
31
32
33
34
35
36
37
38
39     ## Configure AWS credentials and provisioning EKS cluster
40
41
42 # Configure your credentials from before created IAM account with all needed permissions
43 aws configure
44
45 # Init and Deploy Terraform infrastructure
46 terraform init
47 terraform apply
48
49 # Update kube config after EKS created.
50 aws eks update-kubeconfig --name CLUSTER_NAME --region eu-central-1
51 cp ~/.kube/config .
52

- context:
  cluster: eks_final-task-eks
  user: eks_final-task-eks
  name: eks_final-task-eks

current-context: eks_final-task-eks

users:
- name: eks_final-task-eks
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1alpha1
      command: aws-iam-authenticator
      args:
        - "token"
        - "-i"
        - "final-task-eks"

EOT
region = "eu-central-1"
sh-4.2# aws eks update-kubeconfig --name final-task-eks --region eu-central-1
Added new context arn:aws:eks:eu-central-1:595753745217:cluster/final-task-eks to /root/.kube/config
sh-4.2# [ ]
```
- Status Bar:** Shows "Ln 54, Col 29" and other standard status bar information.

# Setup cloud storage.

The screenshot shows a Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Explorer:** Shows a file tree for a project named "PROJECT101-TF-EKS-JENKINS". The tree includes .terraform, jenkins (with subfolders amazon-eks, dockerfiles, jenkins.deployment..., kubernetes, configmaps, deployments, secrets, services, python, Dockerfile, .gitignore), and .terraform.lock.hcl. Below these are data, kubecfg\_final-task..., main.tf, outputs.tf, README.md (which is the active editor), terraform.tfstate, and variables.tf.
- Terminal:** The terminal window displays Jenkins pipeline code for setting up EFS storage. The code includes commands for deploying the EFS storage driver, getting VPC\_ID and CIDR range, creating a security group, creating storage, and mounting it. A portion of the JSON output from the aws efs describe-file-systems command is also visible at the bottom of the terminal.
- Status Bar:** ShowsLn 77, Col 99, Spaces: 4, UTF-8, CRLF, Markdown, Go Live, and a file icon.

```
## Setup our Cloud Storage
```
# Deploy EFS storage driver
kubectl apply -k "github.com/kubernetes-sigs/aws-efs-csi-driver/deploy/kubernetes/overlays/stable/?ref=master"

# Get VPC_ID (and save it to txt file)
aws eks describe-cluster --name CLUSTER_NAME --query "cluster.resourcesVpcConfig.vpcId" --output text

# Get CIDR range (and save it to txt file)
aws ec2 describe-vpcs --vpc-ids VPC_ID --query "Vpcs[].[CidrBlock" --output text

# Security for our instances to access file storage (and save sg-xxxxxx to txt file)
aws ec2 create-security-group --description efs-test-sg --group-name efs-sg --vpc-id VPC_ID
aws ec2 authorize-security-group-ingress --group-id sg-xxxxxx --protocol tcp --port 2049 --cidr VPC_CIDR

# Create storage
aws efs create-file-system --creation-token eks-efs

# Grab our volume handle. Save FileSystemId to txt file and update our jenkins.pv.yaml
aws efs describe-file-systems --query "FileSystems[*].FileSystemId" --output text

# Create mount point (look subnet_id in created instance in EC2)
aws efs create-mount-target --file-system-id FS_ID --subnet-id SUBNET_ID --security-group GROUP_ID
```
{
    "MountTargetId": "fsmt-61565038",
    "FileSystemId": "fs-5c0ff607",
    "SubnetId": "subnet-0575ca6a0f5bd8890",
    "LifeCycleState": "creating",
    "IpAddress": "10.0.6.118",
    "NetworkInterfaceId": "eni-00ef31eda99b7d442",
    "AvailabilityZoneId": "euc1-az1",
    "AvailabilityZoneName": "eu-central-1c",
    "VpcId": "vpc-0604d1d4fb47ef18e"
}
```

# Setup storage for jenkins.

The screenshot shows a Visual Studio Code interface with a dark theme. The left sidebar contains an 'EXPLORER' view with a tree structure of files and folders related to a Jenkins setup. The main editor area displays a script titled 'README.md' with several sections of code highlighted by a large rectangular selection box. The terminal at the bottom shows the execution of commands to set up storage for Jenkins on an AWS EKS cluster.

File Edit Selection View Go Run Terminal Help

• README.md - project101-tf-eks-jenkins - Visual Studio Code [Administrator]

OPEN EDITORS 2 UNSAVED

PROJECT101-TF-EKS-JENKIN

- .terraform
- jenkins
- amazon-eks
  - jenkins.pv.yaml
  - jenkins.pvc.yaml
- dockerfiles
- jenkins.deployment....
  - jenkins.rbac.yaml
  - jenkins.service.yaml
- kubernetes
  - configmaps
    - configmap.yaml
  - deployments
    - deployment.yaml
  - secrets
    - secret.yaml
  - services
    - service.yaml
- python
  - src
  - Dockerfile
- .gitignore
- .terraform.lock.hcl

data

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: docker

```
## Final task EPAM DevOps course winter 2020-2021 ## Setup our storage for Jenkins
aws efs describe-file-systems --query 'FileSystems[*].FilesystemId' --output text
# Create mount point (look subnet_id in created instance in EC2)
aws efs create-mount-target --file-system-id FS_ID --subnet-id SUBNET_ID --security-group GROUP_ID
...
### Setup our storage for Jenkins
...
# Create namespace check storage class
kubectl create ns jenkins
kubectl get storageclass
# Create volume (copy fs-xxxxxx to file jenkins.pv.yaml) and check it
kubectl apply -f ./jenkins/amazon-eks/jenkins.pv.yaml
kubectl get pv
# Create volume claim and check it
kubectl apply -n jenkins -f ./jenkins/amazon-eks/jenkins.pvc.yaml
kubectl -n jenkins get pvc
...
### Deploy Jenkins
...
# Install RBAC and Deploy Jenkins check it
kubectl apply -n jenkins -f ./jenkins/jenkins.rbac.yaml
kubectl apply -n jenkins -f ./jenkins/jenkins.deployment.yaml
```

NAME	PROVISIONER	RECLAIMPOLICY	VOLUMEBINDINGMODE	ALLOWVOLUMEEXPANSION	AGE
gp2 (default)	kubernetes.io/aws-ebs	Delete	WaitForFirstConsumer	false	13m

```
sh-4.2# kubectl get storageclass
NAME      PROVISIONER      RECLAIMPOLICY  VOLUMEBINDINGMODE  ALLOWVOLUMEEXPANSION  AGE
gp2 (default)  kubernetes.io/aws-ebs  Delete        WaitForFirstConsumer  false            13m
```

```
sh-4.2# kubectl apply -f ./jenkins/amazon-eks/jenkins.pv.yaml
persistentvolume/jenkins created
sh-4.2# kubectl get pv
NAME      CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS    CLAIM      STORAGECLASS  REASON  AGE
jenkins  5Gi       RWX          Retain        Available   jenkins   efs-sc      10s
sh-4.2# kubectl apply -n jenkins -f ./jenkins/amazon-eks/jenkins.pvc.yaml
persistentvolumeclaim/jenkins-claim created
sh-4.2# kubectl -n jenkins get pvc
NAME      STATUS  VOLUME      CAPACITY  ACCESS MODES  STORAGECLASS  AGE
jenkins-claim  Bound  jenkins  5Gi       RWX          efs-sc      6s
```

sh-4.2#

OUTLINE

TIMELINE

Ln 95, Col 1 Spaces: 4 UTF-8 CRLF Markdown Go Live

# Deploy Jenkins.

The screenshot shows a Visual Studio Code interface with a dark theme. On the left is the Explorer sidebar showing project files like `README.md`, `main.tf`, `outputs.tf`, etc. The main editor area displays a `README.md` file containing a Jenkins deployment script. The script includes sections for deploying Jenkins, creating services for agents, and performing initial setup. A portion of the script is highlighted with a white box. Below the editor is a terminal window titled "1: docker" showing the execution of the script using `kubectl`. The terminal output shows the creation of a Jenkins pod and service, and the retrieval of the initial admin password. A status table at the bottom provides details about the Jenkins service.

```
## Deploy Jenkins
...
### Create a service for agents
...
### Jenkins Initial Setup
...
# Use type LoadBalancer in service and go to LoadBalancer DNS to configure Jenkins
kubectl -n jenkins get svc
...
### Second option is create service with type ClusterIP and make port forwarding
### kubectl port-forward -n jenkins POD_NAME 8080
...
# Setup user and recommended basic plugins
# Update jenkins after setup
```

NAME	READY	STATUS	RESTARTS	AGE
jenkins-799cc894cf-gzbfm	0/1	ContainerCreating	0	6s
sh-4.2#	kubectl apply -n jenkins -f ./jenkins/jenkins.service.yaml			

service/jenkins created

```
sh-4.2# kubectl -n jenkins exec -it jenkins-799cc894cf-gzbfm cat /var/jenkins_home/secrets/initialAdminPassword
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use kubectl exec [POD] -- [COMMAND] instead.
552c98a9e3dc4c16a9f285be7ab6e92a
sh-4.2# kubectl -n jenkins get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
jenkins	LoadBalancer	172.20.172.201	aa54ce87b0f0941c2a877fbba41f67d23-1297553464.eu-central-1.elb.amazonaws.com	8080:31671/TCP,50000:31017/TCP,80:3122
		53/TCP	69s	

Ln 123, Col 29 Spaces: 4 UTF-8 CRLF Markdown Go Live

# Setup Jenkins and Kubernetes plugin.

aa54ce87fb0f941c2a877fb41f67d23-1297553464.eu-central-1.elb.amazonaws.com

## Getting Started

Getting Started

Folders	OWASP Markup Formatter	Build Timeout	Credentials Binding
✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding
✓ Timestamper	✗ Workspace Cleanup	✗ Ant	✗ Gradle
✗ Pipeline	✗ GitHub Branch Source	✗ Pipeline: Groovy Libraries	✗ Pipeline: Stage View
✗ Git	✗ Subversion	✗ SSH Build Agents	✗ Matrix Authorization Strategy
✗ PAM Authentication	✗ LDAP	✗ Email Extension	✗ Mailer

## Containers

Container Template

Name: jnlp

Docker image: aimvector/jenkins-slave

Always pull image

Working directory: /home/jenkins/agent

Command to run:

Arguments to pass to the command:

[Go back to the top page](#)  
(you can start using the installed plugins right away)

Restart Jenkins when installation is complete and no jobs are running

## Configure Clouds

Kubernetes

Name: kubernetes

Kubernetes URL: https://kubernetes.default:443

Use Jenkins Proxy

Kubernetes server certificate key:

Disable https certificate check

Kubernetes Namespace: jenkins

Credentials: - none -

Connected to Kubernetes v1.17.12-eks-7684af

## Volumes

### Host Path Volume

Host path: /var/run/docker.sock

Mount path: /var/run/docker.sock

### DOCKER\_USERNAME

### DOCKER\_PASSWORD

## Jenkins URL

http://jenkins

## Jenkins tunnel

jenkins:50000

## Connection Timeout

5

## Read Timeout

15

## Concurrency Limit

10

## Pod Templates

Pod Template

Name: jenkins-slave

## Namespace

jenkins

## Labels

jenkins-slave

## Usage

Only build jobs with label expressions matching the following:

Pod template to inherit from:

## Run As User ID

1001

## Run As Group ID

1950

## Request CPU

Secret text

## DOCKER\_USERNAME

## DOCKER\_PASSWORD

Secret text

# Make a pipeline.

Git

Repositories

Repository URL  
<https://github.com/OLG-MAN/project101-tf-eks-jenkins.git>

Credentials  
- none -

Branches to build

Branch Specifier (blank for 'any')  
\*/main

Repository browser  
(Auto)

Additional Behaviours  
  Subversion

All

S	W	Name	Last Success
		Pipeline	51 sec - #3
		pre-build	1 min 1 sec - #3

Icon: S M L

Discard old builds  
 GitHub project  
Project url  
<https://github.com/OLG-MAN/project101-tf-eks-jenkins>

This build requires lockable resources  
 This project is parameterized  
 Throttle builds  
 Disable this project  
 Execute concurrent builds if necessary  
 Restrict where this project can be run  
Label Expression

Post-build Actions

Build other projects  
Projects to build  
Pipeline.

Trigger only if build is stable  
 Trigger even if the build is unstable  
 Trigger even if the build fails

Payload URL \*  
<http://aa54ce87b0f0941c2a877fb41f67d23-1297553464.eu-central-1.amazonaws.com:443/>

Content type  
application/x-www-form-urlencoded

Secret

Which events would you like to trigger this webhook?

Just the push event.  
 Send me everything.  
 Let me select individual events.

Active  
We will deliver event details when this hook is triggered.

- Build Triggers
- Trigger builds remotely (e.g., from scripts)  
 Build after other projects are built  
 Build periodically  
 GitHub hook trigger for GITScm polling  
 Poll SCM

# Make a pipeline.

File Edit Selection View Go Run Terminal Help

README.md - project101-tf-eks-jenkins - Visual Studio Code [Administrator]

EXPLORER OPEN EDITORS PROJECT101-TF-EKS-JENKINSL README.md README.md # project101-tf-eks-jenkins > ## Final task EPAM DevOps course winter 2020-2021 > ### Install kubectl, eksctl, terraform

```
200 pipeline {
201   agent {
202     kubernetes{
203       label 'jenkins-slave'
204     }
205   }
206   environment{
207     DOCKER_USERNAME = credentials('DOCKER_USERNAME')
208     DOCKER_PASSWORD = credentials('DOCKER_PASSWORD')
209   }
210   stages {
211     stage('docker login') {
212       steps{
213         sh('docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD')
214       }
215     }
216
217     stage('git clone') {
218       steps{
219         sh(script: """
220           git clone https://github.com/OLG-MAN/project101-tf-eks-jenkins.git
221           """, returnStdout: true)
222       }
223     }
224
225     stage('docker build') {
226       steps{
227         sh script: """
228           #!/bin/bash
229           cd $WORKSPACE/project101-tf-eks-jenkins/python
230           docker build . --network host -t olegan/testapp:${BUILD_NUMBER}
231           """
232       }
233     }
234
235     stage('docker push') {
236       steps{
237         sh(script: """
238           docker push olegan/testapp:${BUILD_NUMBER}
239           """
240       }
241     }
242
243     stage('deploy') {
244       steps{
245         sh script: """
246           #!/bin/bash
247           cd $WORKSPACE/project101-tf-eks-jenkins/
248           #get kubectl for this demo
249           curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt`/bin/linux/amd64/kubectl
250           chmod +x ./kubectl
251           ./kubectl apply -f ./kubernetes/configmaps/configmap.yaml
252           ./kubectl apply -f ./kubernetes/secrets/secret.yaml
253           cat ./Kubernetes/deployments/deployment.yaml | sed s/10/${BUILD_NUMBER}/g | ./kubectl apply -f -
254           ./kubectl apply -f ./kubernetes/services/service.yaml
255           """
256       }
257     }
258   }
259 }
```

OUTLINE TIMELINE

In 26 Col 78 Spaces: 4 UTF-8 CRLF Markdown Go Live

File Edit Selection View Go Run Terminal Help • README.md - project101-tf-eks-jenkins - Visual Studio Code [Administrator]

EXPLORER OPEN EDITORS 2 UNSAVED PROJECT101-TF-EKS-JENKIN... src app \_pycache\_ ajax auth cart general \_pycache\_ templates\gene... base.html index.html search\_results.... \_init\_.py general.py products static \_init\_.py dbschema.py models-dev.py models.py site.db app.py requirements.txt Dockerfile .gitignore .terraform.lock.hcl data kubeconfig\_final-task... main.tf outputs.tf README.md terraform.tfstate variables.tf OUTLINE TIMELINE

README.md # project101-tf-eks-jenkins ## Final task EPAM DevOps course winter 2020-2021 ## Check all components of k8s cluster

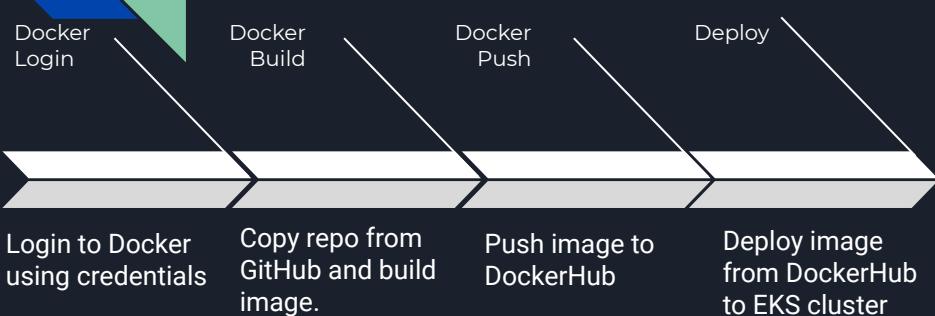
```
259 }
260 }
261 ...
262
263 ### Check all components of k8s cluster
264 ...
265
266 #Check our deployment
267 kubectl -n jenkins get deploy -owide
268 #Check our pods
269 kubectl -n jenkins get pods
270 #Check services
271 kubectl -n jenkins get svc
272 ...
273
274 ### Copy DNS from "example-service" and paste to browser. Check working web app.
275
276 ### Add domain name to our EKS cluster
277 * Register free domain like web-a.pp.ua
278 * Create hosted zone in Route 53 with same address name
279 * Change NS in domain registrar to Route53 Hosted Zone NS
280 * Create record - address alias to Cluster LoadBalancer
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: docker

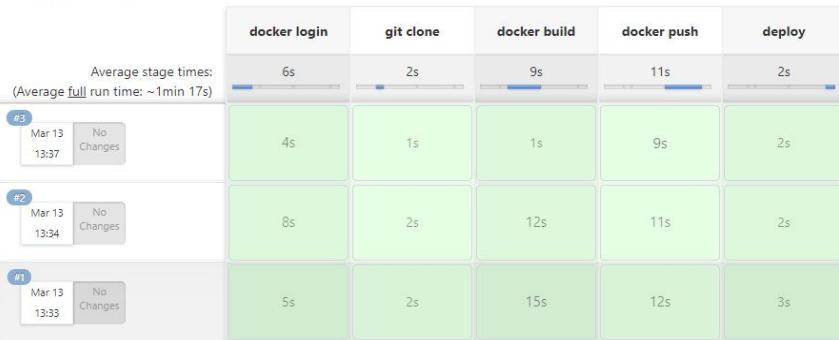
```
jenkins LoadBalancer 172.20.172.201 aa54ce87b0f0941c2a877fb41f67d23-1297553464.eu-central-1.elb.amazonaws.com 8080:31671/TCP,50000:31017/TCP,80:3125/TCP 69s
sh-4.2# kubectl -n jenkins get deploy -owide
NAME READY UP-TO-DATE AVAILABLE AGE CONTAINERS IMAGES SELECTOR
example-deploy 2/2 2 2 3m49s example-app olegan/testapp:3 app-example-app
jenkins 1/1 1 1 48m jenkins jenkins/jenkins:2.235.1-lts-alpine app=jenkins
sh-4.2# kubectl -n jenkins get pods
NAME READY STATUS RESTARTS AGE
example-deploy-54b4446487-9wk9z 1/1 Running 0 65s
example-deploy-54b4446487-dptkn 1/1 Running 0 62s
jenkins-799cc894cf-gzbfm 1/1 Running 0 48m
sh-4.2# kubectl -n jenkins get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S)
AGE
example-service LoadBalancer 172.20.151.18 a1930bbe350fb43fb05310a320c38fa-1431468998.eu-central-1.elb.amazonaws.com 80:31451/TCP
4m3s
jenkins LoadBalancer 172.20.172.201 aa54ce87b0f0941c2a877fb41f67d23-1297553464.eu-central-1.elb.amazonaws.com 8080:31671/TCP,50000:31017/TCP
P:80:31253/TCP 47m
sh-4.2#
```

main □ 0 ▲ 0 Signing in to github.com... Ln 271, Col 1 (26 selected) Spaces: 4 UTF-8 CRLF Markdown Go Live

# Pipeline CI/CD



## Stage View



## Add domain to Route53

### Add domain name to our EKS cluster

- Register free domain like web-a.pp.ua
- Create hosted zone in Route 53 with same address name
- Change NS in domain registrar to Route53 Hosted Zone NS
- Create record - address alias to Cluster LoadBalancer
- After 15-20 min registered address start working. (Can check status in <https://www.whatsmydns.net/>)

Route 53 > Hosted zones > finaltask-a.pp.ua

finaltask-a.pp.ua [Info](#)

#### Hosted zone details

[Records \(3\)](#) DNSSEC signing Hosted zone tags (0)

#### Records (3) [Info](#)

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

<input type="checkbox"/> Record name	Type	Routin...	Differ...	Value/Route traffic to
<input type="checkbox"/> finaltask-a.pp.ua	A	Simple	-	dualstack.a1930bbe350fb43fb05310a320c38fa-1431468998.eu-central-1.elb.amazonaws.com.
<input type="checkbox"/> finaltask-a.pp.ua	NS	Simple	-	ns-97.awsdns-12.com. ns-917.awsdns-50.net. ns-1674.awsdns-17.co.uk. ns-1215.awsdns-23.org.
<input type="checkbox"/> finaltask-a.pp.ua	SOA	Simple	-	ns-97.awsdns-12.com. awsdns-hostmaster.amazon.com. 17200 900 1209600 86400

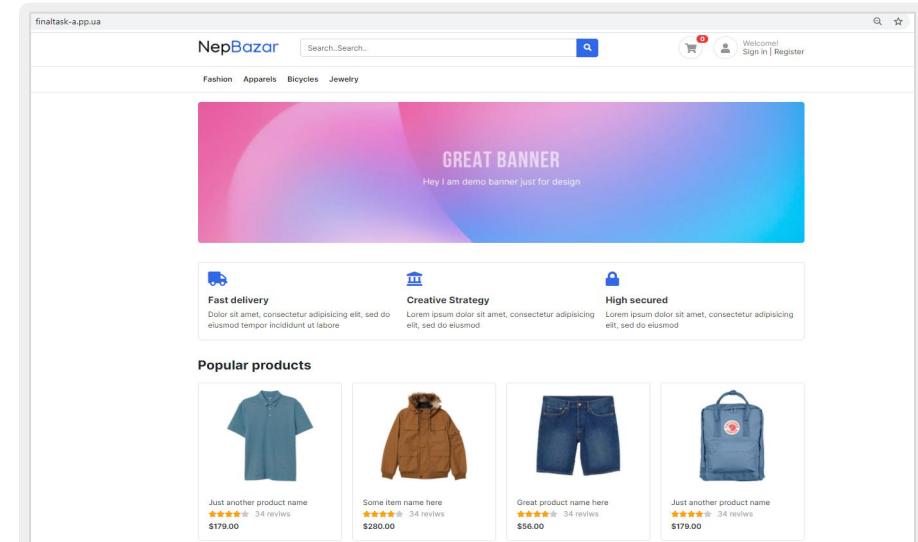


# Final result.

We can see our application by going to the address  
<http://finaltask-a.pp.ua>

Also we can go to GitHub repo with source code and detailed readme.

<https://github.com/OLG-MAN/project101-tf-eks-jenkins>



Thanks for watching.

email: mandrik89@gmail.com

cv: <https://olg-man.github.io/cv2021/>

