

# Big Project

Migrasi Infrastruktur On  
Premises ke Cloud AWS

*By. Heri Sulistiyo*



The number '01.' is rendered in a white, elegant script font. It is positioned on the left side of the slide, with a horizontal, textured brown brushstroke behind it. The background is a solid dark navy blue. There are abstract organic shapes in the corners: a light cream shape with dark dots in the bottom-left, a large orange shape with a grey textured area and white dashed lines in the top-right, and a brown shape with dark dots in the bottom-left.

*Problem and Solution*



## *Problem*

Production & staging dalam satu server on premises (tidak mendukung HA).

Maintenance Datacenter yang mahal.

Manual Deployment dan Manual Integration yang mengakibatkan resiko failure yang tinggi dan waktu yang cukup lama dalam pengerjaan.




## *solution*

Menggunakan teknologi containerization dengan cloud server (Mendukung HA).

Biaya disesuaikan dengan penggunaan.

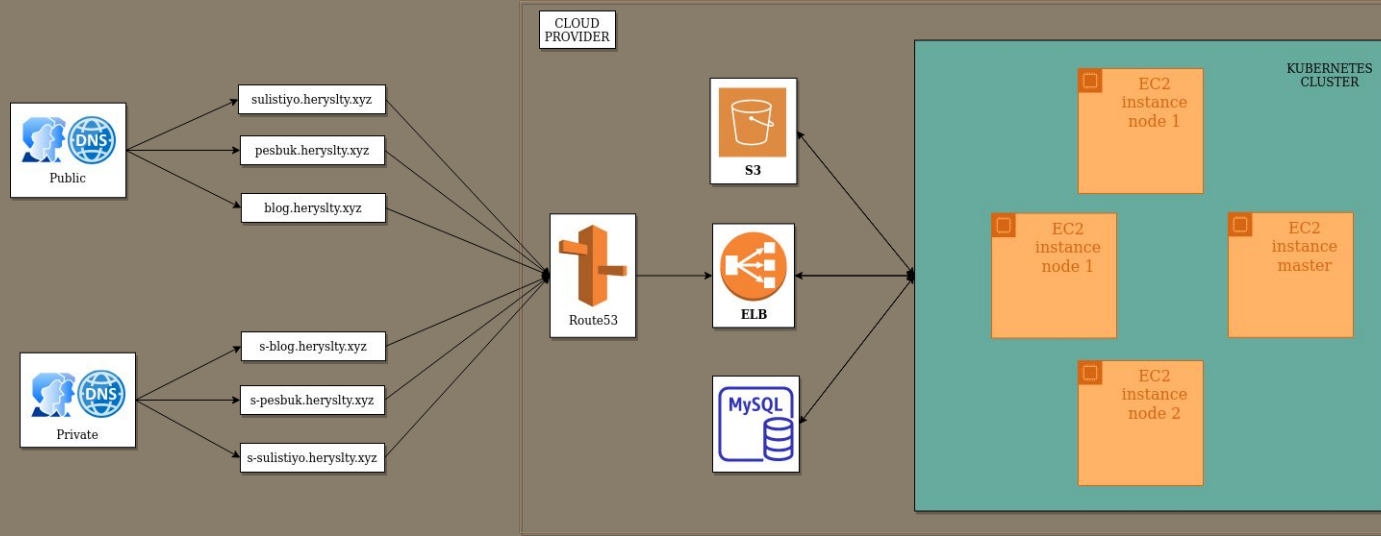
CI/CD akan memberikan resiko failure yang lebih rendah dan waktu pengerjaan yang lebih cepat.



02.

*Architecture*

# Application Accessibility



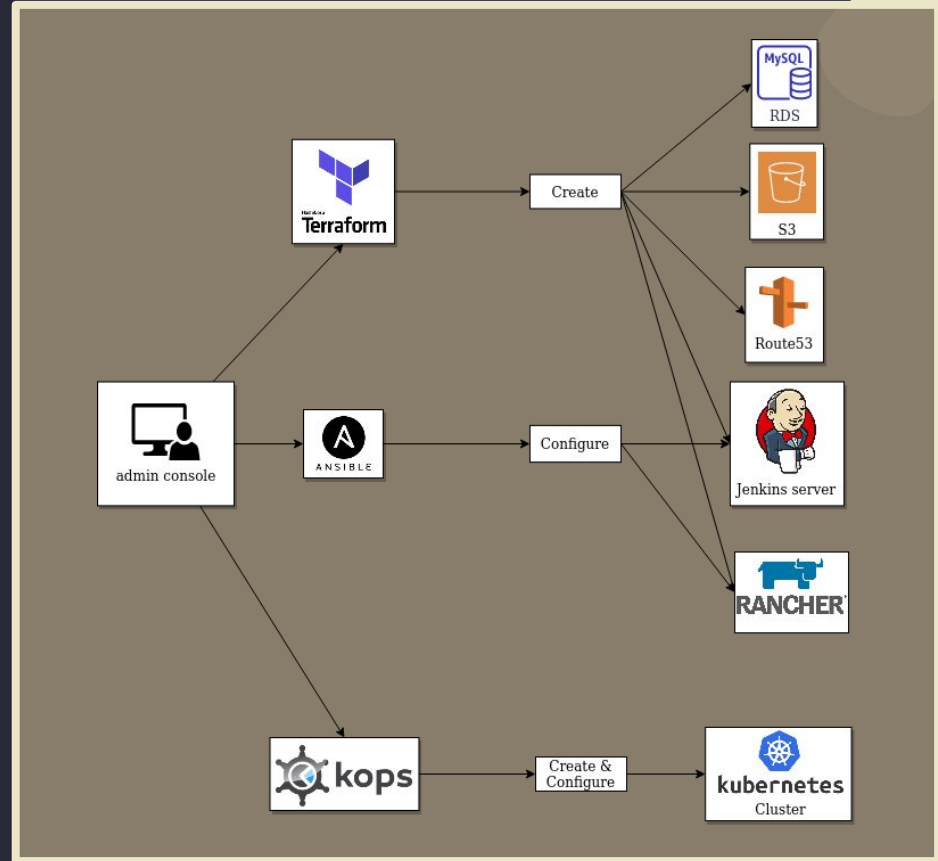


## Create & Configuration

Terraform – dengan menggunakan tools ini kita dapat menerapkan infrastructure as a code sehingga memudahkan apabila akan membuat infrastruktur yang sama.

Ansible – dengan menggunakan ansible, kita dapat mengkonfigurasi server dengan mudah dan mengurangi resiko kesalahan akibat perulangan

Kops – dengan kops kita dapat dengan mudah membuat cluster kubernetes pada cloud provider (AWS)

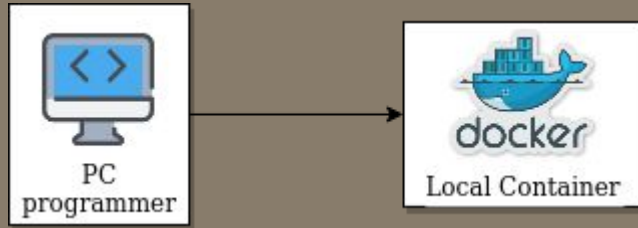




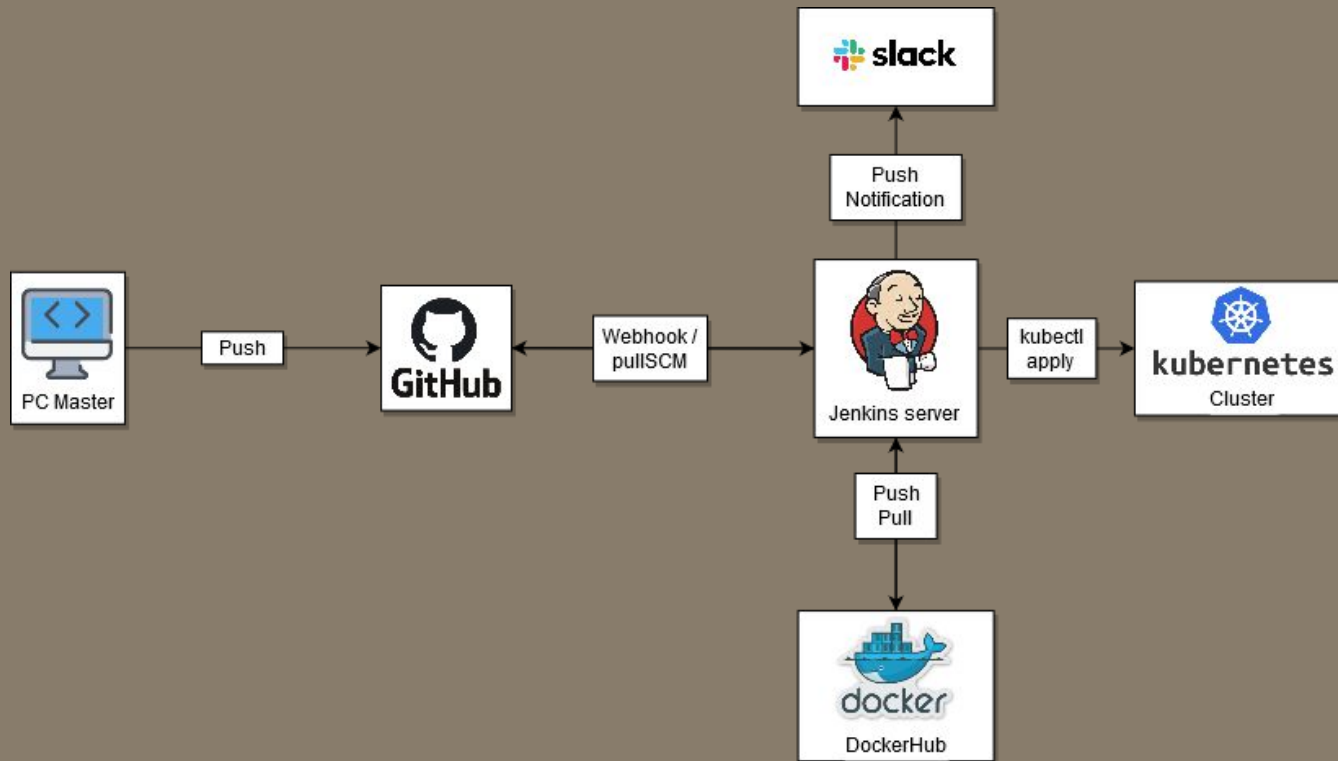
## Local Development

Dengan menggunakan teknik local container yang menerapkan volume bind, programmer dapat langsung merunning aplikasi yang dibuat pada local laptop masing-masing.

Sebagai contoh, working directory web pesbuk pada container yaitu :  
`/var/www/html/sosial-media/` maka directory pada container tersebut disinkronisasi dengan directory local programmer sehingga programmer dapat langsung melakukan perubahan tanpa perlu build ulang container.



# Deployment Architecture

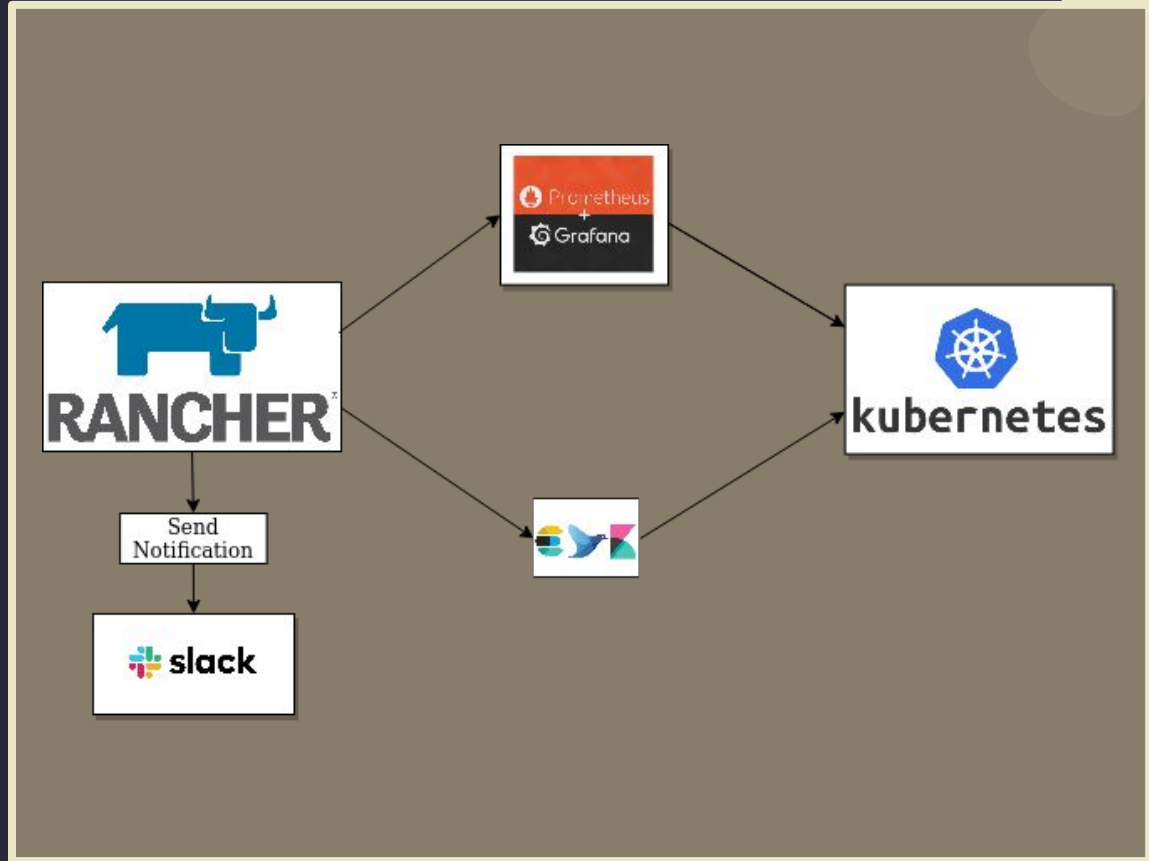




# Logging & Monitoring



Rancher menjadi logging & monitoring management, yang terhubung ke prometheus & grafana untuk mendukung fitur monitoring, serta menggunakan EFK untuk menjalankan fitur logging. Selain itu, rancher juga dapat mengirimkan informasi kondisi cluster ke aplikasi slack



03

# Workflow Process

01

## *Development*

Proses dilakukan di laptop masing-masing programmer

02

## *Staging*

Merupakan kondisi dimana aplikasi akan dipantau pada server yang merupakan replikasi production

03

## *Production*

Merupakan kondisi dimana aplikasi telah dapat digunakan oleh user

04

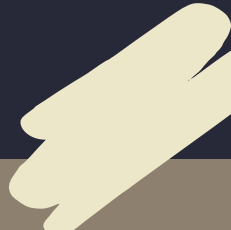
## *Monitoring*

Monitoring dan Logging digunakan untuk memantau semua environment berjalan dengan baik

00

## *ci/cd*

Continuous Integration (CI), Continuous Delivery (CD) berada pada setiap proses tersebut





04

*Budgeting*



# Cost



NAME	Service Type	Components Information	Component Price	quantity	Service Price
Domain	Domain	1 years	1,2	1	1,2
Node Server	EC2 Service (Asia Pacific (Singapore))				
		Compute: t2.medium(2 cpu, 4Gb RAM)	42,75	3	128,25
		EBS Volumes: SSD 128GB/instance	15,36	3	46,08
Master Server	EC2 Service (Asia Pacific (Singapore))				
		Compute: m3.medium(2 cpu, 3,75Gb RAM)	71,74	1	71,74
		EBS Volumes: SSD 104GB/instance	12,48	1	12,48

# Cost

NAME	Service Type	Components Information	Component Price	quantity	Service Price
<b>Rancher server</b>	EC2 Service (Asia Pacific (Singapore))				
		Compute: t2.medium(2 cpu, 4Gb RAM)	42,75	1	42,75
		EBS Volumes: SSD 40GB/instance	4,8	1	4,8
<b>Route53</b>	Amazon Route 53 Service				
		Hosted Zones: 1 Hosted Zone	0,5	1	0,5
<b>RDS Mysql</b>	Amazon RDS Service (Asia Pacific (Singapore))				
		DB instances: *mysql *db.t2.micro	19,04	1	19,04
		Storage: SSD 20GB	2,76	1	2,76



# Cost



NAME	Service Type	Components Information	Component Price	quantity	Service Price
ELB	Amazon Elastic Load Balancing (Asia Pacific (Singapore))				
		Classic LBs: 1 ELB	20,5	1	20,5
S3	Amazon S3 Service (Asia Pacific (Singapore))				
		S3 Standard Storage 100GB storage	2,5	1	2,5
jenkins server	EC2 Service (Asia Pacific (Singapore))				
		Compute: t2.medium(2 cpu, 4Gb RAM)	42,75	1	42,75
		EBS Volumes: SSD 40GB/instance	4,8	1	4,8
Total Monthly Payment:					400,15

# • *Total Cost estimation* •

Dapat dilihat dari hasil perhitungan diatas maka pembayaran yang akan dilakukan perbulan yaitu kurang lebih sebesar \$400,15 atau setara dengan Rp6.227.575. Nilai tersebut merupakan estimasi nilai rata -rata yang akan dibayarkan, hal ini dikarenakan nilai yang dibayarkan akan bergantung dengan trafik yang ada sehingga nilainya dapat berubah tiap bulannya. sehingga dalam 6 bulan estimasi pembayaran yaitu kurang lebih sebesar \$2400,9 atau setara dengan Rp37.365.450.



# *How to reduce this Cost?*

Dengan mengurangi fitur monitoring dan logging maka kita dapat mengurangi biaya yang dikeluarkan menjadi \$251,75 perbulan atau setara dengan Rp23.508.060 /6bulan. Hal ini terjadi karena kedua fitur tersebut membutuhkan resource yang cukup besar, sehingga dengan mengurangi kedua fitur tersebut kita dapat mengurangi resourcenya dari yang awalnya menggunakan 3 buah node ec2 t2.medium menjadi 2buah ec2 t2.small, dan mengurangi resouce untuk server rancher serta mengurangi penggunaan hardisk sebesar 168Gb.

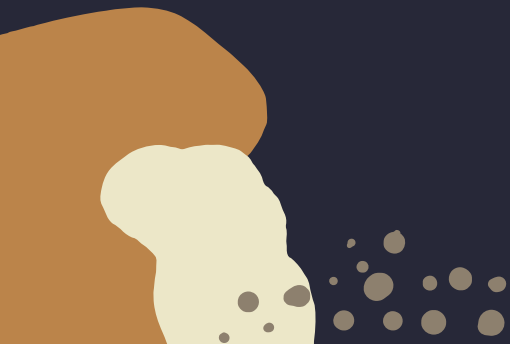


05



# *Source Code*

You could enter a subtitle here if you need it



# Ansible

main.yml - s-bigProject - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

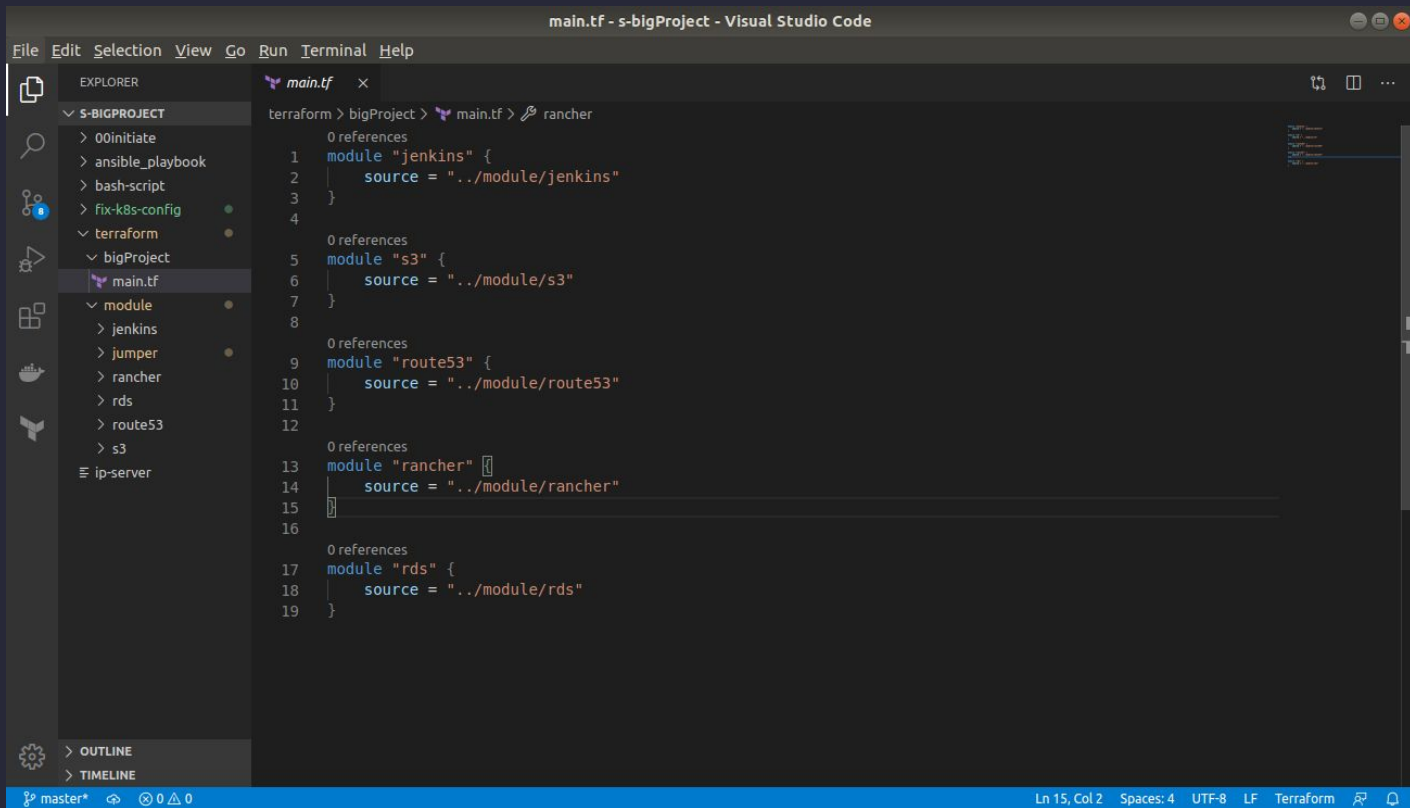
- ▼ S-BIGPROJECT
  - > 00initiate
  - ▼ ansible\_playbook
    - ▼ roles
    - > docker
    - ▼ jenkins-nginx-ssl
      - > handlers
      - ▼ tasks
        - ! main.yml
        - > templates
        - > kops
        - > kubectl
        - > ssh-key
        - ! bigproject.yml
      - ≡ hosts
    - ▼ bash-script
      - docker.sh
      - kops-configure.sh
      - kops-create-cluster.sh
      - kops.sh
      - kubectl.sh
      - nginx-ssl.sh
      - nginx.sh
    - > fix-k8s-config
    - > terraform
    - ≡ ip-server

! main.yml

```
1 ---
2 - name: Run script instal jenkins nginx with config ssl
3   script: /home/hery/workspace/bigProject/bash_script/nginx-ssl.sh
4
5
6 - name: config sites-available nginx
7   template:
8     src: default
9     dest: /etc/nginx/sites-available/default
10  become: true
11  notify: Reload service NGINX
12
```

Ln 12, Col 1 Spaces: 2 UTF-8 LF YAML

# Terraform



main.tf - s-bigProject - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

- main.tf
- S-BIGPROJECT
  - 00initiate
  - ansible\_playbook
  - bash-script
  - fix-k8s-config
  - terraform
    - bigProject
      - main.tf
      - module
        - jenkins
        - jumper
        - rancher
        - rds
        - route53
        - s3
  - ip-server

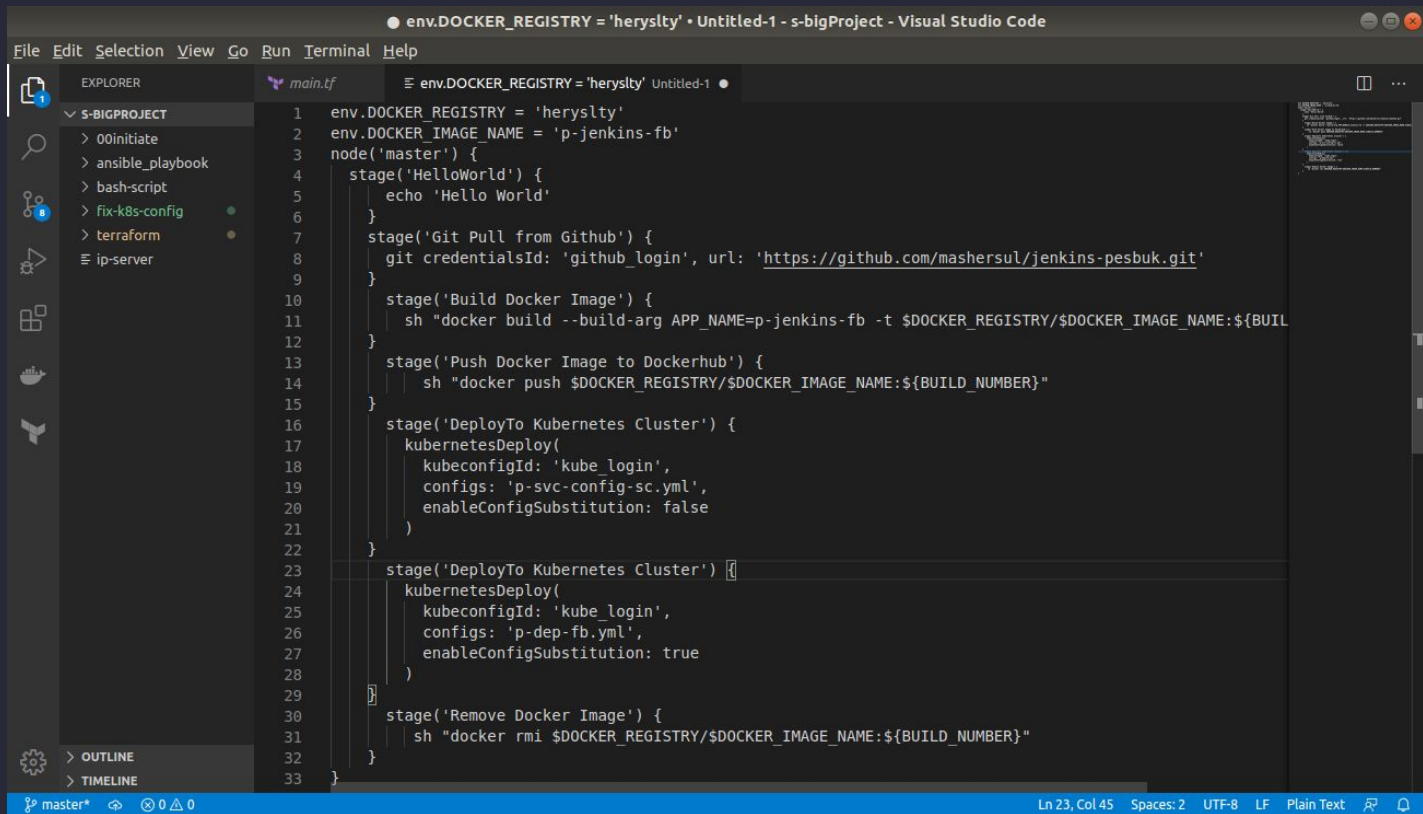
OUTLINE

TIMELINE

```
terraform > bigProject > main.tf > rancher
0 references
1 module "jenkins" {
2   | source = "../module/jenkins"
3 }
4
5 0 references
6 module "s3" {
7   | source = "../module/s3"
8 }
9
10 0 references
11 module "route53" {
12   | source = "../module/route53"
13 }
14
15 0 references
16 module "rancher" {
17   | source = "../module/rancher"
18 }
19
20 0 references
21 module "rds" {
22   | source = "../module/rds"
23 }
```

Ln 15, Col 2 Spaces: 4 UTF-8 LF Terraform

# • Pipeline Jenkinsfile •

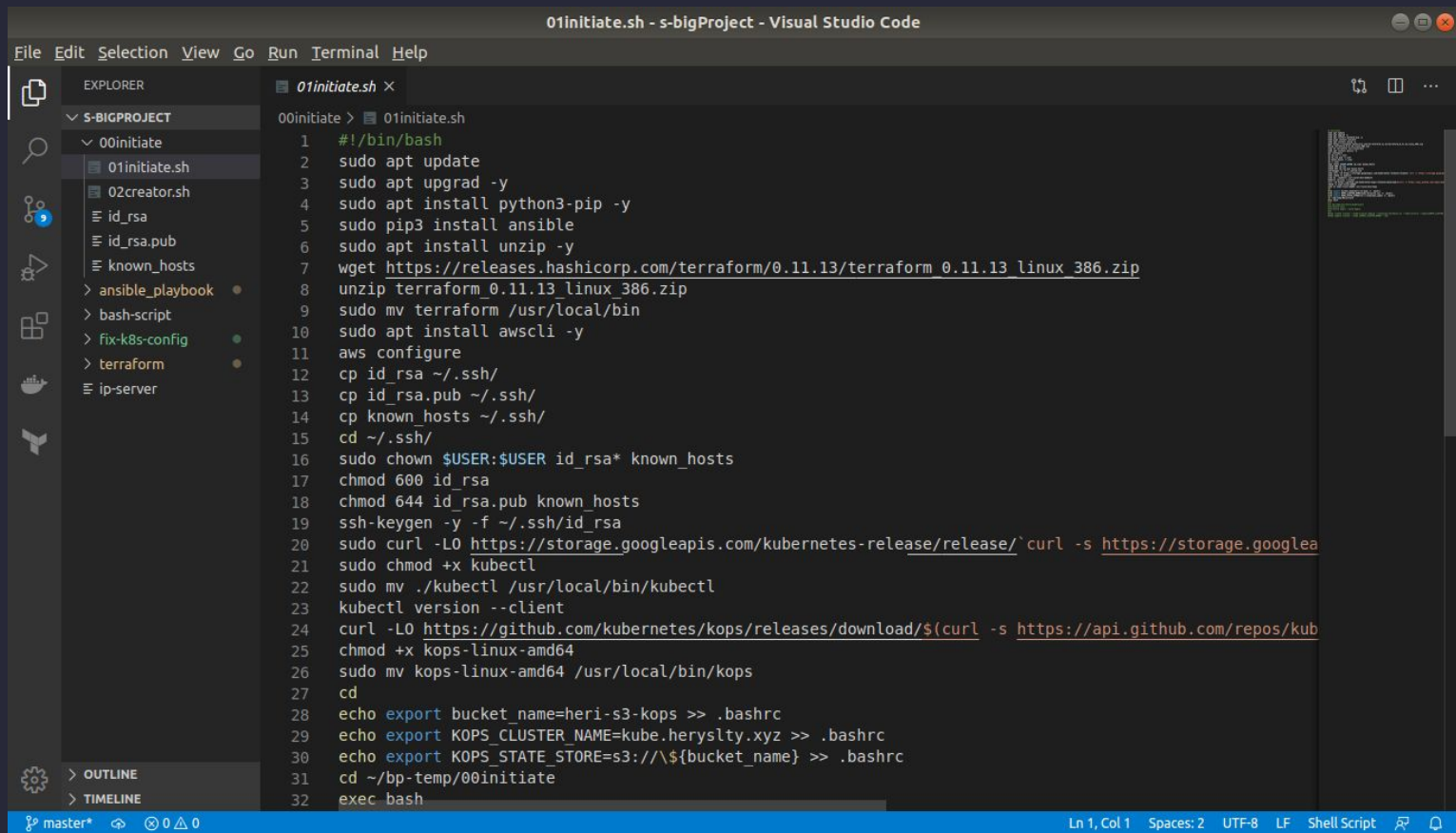


```
env.DOCKER_REGISTRY = 'heryslty' • Untitled-1 - s-bigProject - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
main.tf
env.DOCKER_REGISTRY = 'heryslty' Untitled-1 •
s-BIGPROJECT
  > 00initiate
  > ansible_playbook
  > bash-script
  > fix-k8s-config
  > terraform
  > ip-server
  > OUTLINE
  > TIMELINE

1 env.DOCKER_REGISTRY = 'heryslty'
2 env.DOCKER_IMAGE_NAME = 'p-jenkins-fb'
3 node('master') {
4   stage('HelloWorld') {
5     echo 'Hello World'
6   }
7   stage('Git Pull from Github') {
8     git credentialsId: 'github_login', url: 'https://github.com/mashersul/jenkins-pesbuk.git'
9   }
10  stage('Build Docker Image') {
11    sh "docker build --build-arg APP_NAME=p-jenkins-fb -t $DOCKER_REGISTRY/$DOCKER_IMAGE_NAME:${BUILD_NUMBER}"
12  }
13  stage('Push Docker Image to Dockerhub') {
14    sh "docker push $DOCKER_REGISTRY/$DOCKER_IMAGE_NAME:${BUILD_NUMBER}"
15  }
16  stage('DeployTo Kubernetes Cluster') {
17    kubernetesDeploy(
18      kubeconfigId: 'kube_login',
19      configs: 'p-svc-config-sc.yml',
20      enableConfigSubstitution: false
21    )
22  }
23  stage('DeployTo Kubernetes Cluster') {
24    kubernetesDeploy(
25      kubeconfigId: 'kube_login',
26      configs: 'p-dep-fb.yml',
27      enableConfigSubstitution: true
28    )
29  }
30  stage('Remove Docker Image') {
31    sh "docker rmi $DOCKER_REGISTRY/$DOCKER_IMAGE_NAME:${BUILD_NUMBER}"
32  }
33 }
```

Ln 23, Col 45 Spaces: 2 UTF-8 LF Plain Text

# Initiate Bash



01initiate.sh - s-bigProject - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

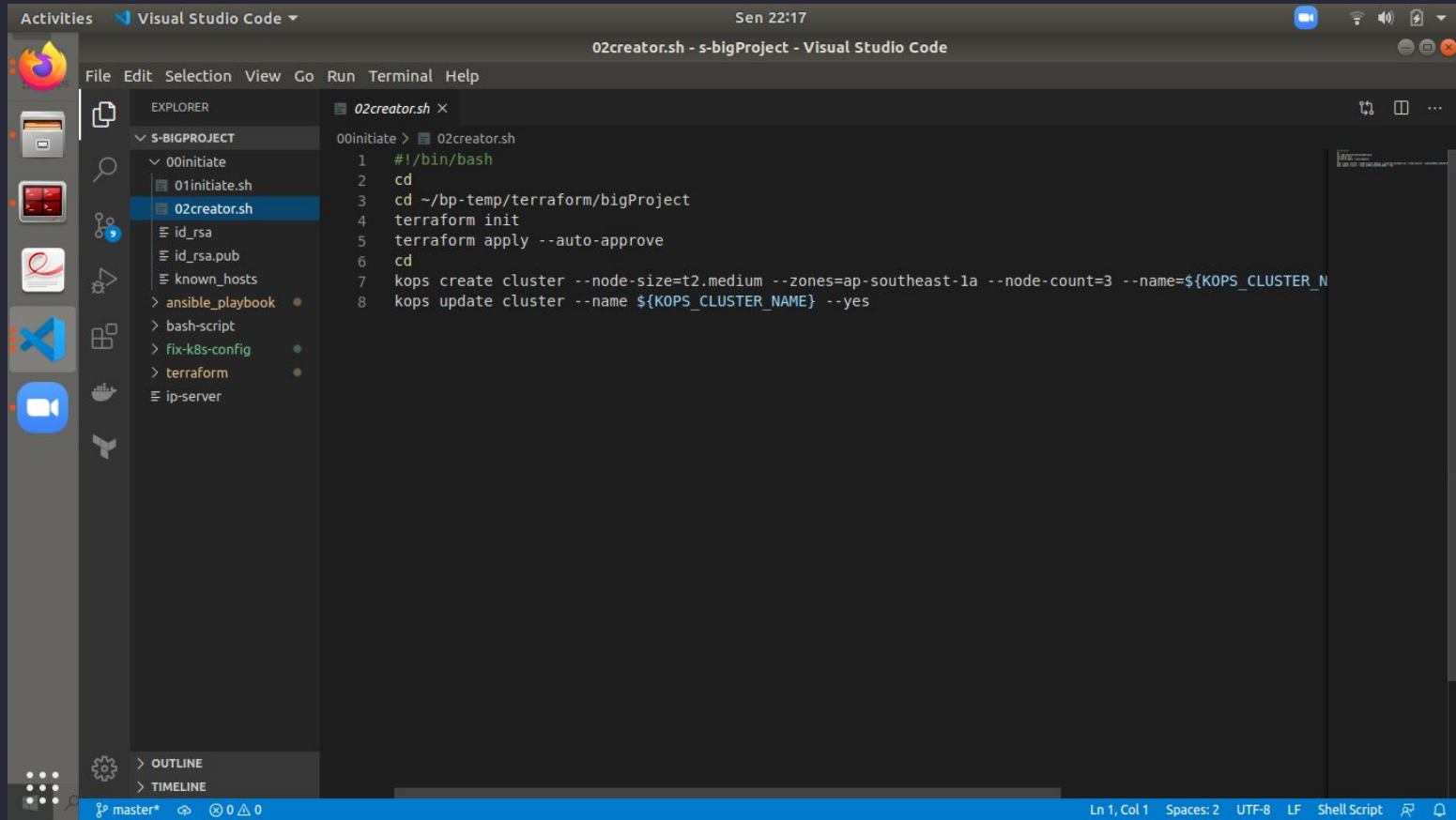
- S-BIGPROJECT
  - 00initiate
    - 01initiate.sh
    - 02creator.sh
    - id\_rsa
    - id\_rsa.pub
    - known\_hosts
    - ansible\_playbook
    - bash-script
    - fix-k8s-config
    - terraform
    - ip-server

01initiate.sh

```
00initiate > 01initiate.sh
1  #!/bin/bash
2  sudo apt update
3  sudo apt upgrad -y
4  sudo apt install python3-pip -y
5  sudo pip3 install ansible
6  sudo apt install unzip -y
7  wget https://releases.hashicorp.com/terraform/0.11.13/terraform_0.11.13_linux_386.zip
8  unzip terraform_0.11.13_linux_386.zip
9  sudo mv terraform /usr/local/bin
10 sudo apt install awscli -y
11 aws configure
12 cp id_rsa ~/.ssh/
13 cp id_rsa.pub ~/.ssh/
14 cp known_hosts ~/.ssh/
15 cd ~/.ssh/
16 sudo chown $USER:$USER id_rsa* known_hosts
17 chmod 600 id_rsa
18 chmod 644 id_rsa.pub known_hosts
19 ssh-keygen -y -f ~/.ssh/id_rsa
20 sudo curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.googlea
21 sudo chmod +x kubectl
22 sudo mv ./kubectl /usr/local/bin/kubectl
23 kubectl version --client
24 curl -LO https://github.com/kubernetes/kops/releases/download/$(curl -s https://api.github.com/repos/kub
25 chmod +x kops-linux-amd64
26 sudo mv kops-linux-amd64 /usr/local/bin/kops
27 cd
28 echo export bucket_name=heri-s3-kops >> .bashrc
29 echo export KOPS_CLUSTER_NAME=kube.heryslty.xyz >> .bashrc
30 echo export KOPS_STATE_STORE=s3://\${bucket_name} >> .bashrc
31 cd ~/bp-temp/00initiate
32 exec bash
```

master\* 0 0 0 Ln 1, Col 1 Spaces: 2 UTF-8 LF Shell Script

# Initiate Bash 02



The screenshot shows the Visual Studio Code interface with the file explorer on the left, the editor in the center, and the terminal on the right. The file explorer shows a project structure with a file named `02creator.sh` selected. The editor displays the contents of `02creator.sh`, which is a bash script. The terminal on the right shows the output of the script, which is a series of commands being executed in a shell.

```
02creator.sh ×
00initiate > 02creator.sh
1  #!/bin/bash
2  cd
3  cd ~/bp-temp/terraform/bigProject
4  terraform init
5  terraform apply --auto-approve
6  cd
7  kops create cluster --node-size=t2.medium --zones=ap-southeast-1a --node-count=3 --name=${KOPS_CLUSTER_NAME}
8  kops update cluster --name ${KOPS_CLUSTER_NAME} --yes
```

Visual Studio Code interface details:

- Activities: Visual Studio Code
- Sen 22:17
- 02creator.sh - s-bigProject - Visual Studio Code
- File Edit Selection View Go Run Terminal Help
- EXPLORER
  - 5-BIGPROJECT
    - 00initiate
      - 01initiate.sh
      - 02creator.sh
      - id\_rsa
      - id\_rsa.pub
      - known\_hosts
      - ansible\_playbook
      - bash-script
      - fix-k8s-config
      - terraform
      - ip-server
- OUTLINE
- TIMELINE
- master\*
- Ln 1, Col 1 Spaces: 2 UTF-8 LF Shell Script



DEMO



*Terimakasih*

