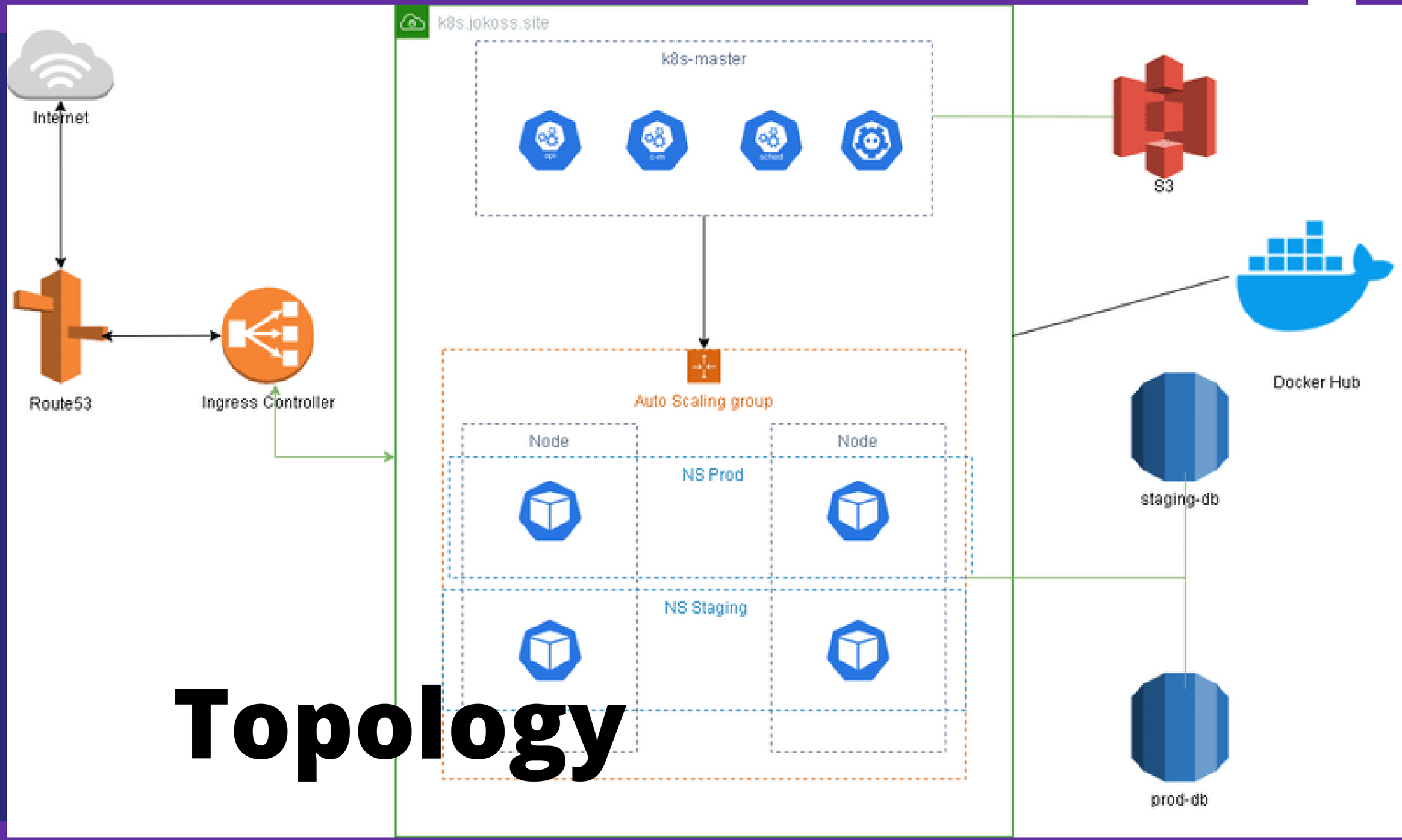
Macross

Membangun Infrastruktur Berbasis Container Orchestration

Joko Sarjono Slameto

Problem

1. Migrasi layanan menjadi container base dan integrasi ke layanan AWS.
2. Memiliki 2 server yaitu staging dan production dengan tiga aplikasi yaitu sosial media pesbuk, website company, dan blog documentation berbasis wordpress.
3. Memindahkan semua service ke kubernetes.


k8s.jokoss.site
k8s-master
Internet
Route53
Ingress Controller
Auto Scaling group
Node
Node
NS Prod
NS Staging
S3
Docker Hub
staging-db
prod-db

Topology

Budgeting (Minimum Load)

Service	Monthly	Configuration summary
Amazon RDS for MySQL (Production)	317.48	Storage for each RDS instance (General Purpose SSD (gp2)), Storage amount (50 GB), Quantity (1), Instance type (db.t3.large), Deployment option (Multi-AZ), Pricing strategy (OnDemand)
Amazon RDS for MySQL (Staging)	160.12	Storage for each RDS instance (General Purpose SSD (gp2)), Storage amount (30 GB), Quantity (1), Instance type (db.t3.medium), Deployment option (Multi-AZ), Pricing strategy (OnDemand)
Classic Load Balancer (Ingress)	26.28	Number of Classic Load Balancers (1), Processed bytes per CLB (1 GB per hour)
Amazon EC2 (Master)	72.51	Operating system (Linux), Quantity (1), Pricing strategy (On-Demand Instances), Storage amount (30 GB), Instance type (t3a.large)
Amazon EC2 (Worker)	76.11	Operating system (Linux), Quantity (2), Pricing strategy (On-Demand Instances), Storage amount (30 GB), Instance type (t3a.medium)
Amazon EC2 (Bastion)	24.92	Operating system (Linux), Quantity (1), Pricing strategy (On-Demand Instances), Storage amount (30 GB), Instance type (t2.small)
S3 Standard	0.03	S3 Standard storage (1 GB per month)
Total	677.45	

Total Cost (3 months) : 3 x 677.45 = 2032.35

Budgeting (Maksimum Load)

Service	Monthly	Configuration summary
Amazon RDS for MySQL (Production)	317.48	Storage for each RDS instance (General Purpose SSD (gp2)), Storage amount (50 GB), Quantity (1), Instance type (db.t3.large), Deployment option (Multi-AZ), Pricing strategy (OnDemand)
Amazon RDS for MySQL (Staging)	160.12	Storage for each RDS instance (General Purpose SSD (gp2)), Storage amount (30 GB), Quantity (1), Instance type (db.t3.medium), Deployment option (Multi-AZ), Pricing strategy (OnDemand)
Classic Load Balancer (Ingress)	26.28	Number of Classic Load Balancers (1), Processed bytes per CLB (1 GB per hour)
Amazon EC2 (Master)	72.51	Operating system (Linux), Quantity (1), Pricing strategy (On-Demand Instances), Storage amount (30 GB), Instance type (t3a.large)
Amazon EC2 (Worker)	190.28	Operating system (Linux), Quantity (5), Pricing strategy (On-Demand Instances), Storage amount (30 GB), Instance type (t3a.medium)
Amazon EC2 (Bastion)	24.92	Operating system (Linux), Quantity (1), Pricing strategy (On-Demand Instances), Storage amount (30 GB), Instance type (t2.small)
S3 Standard	0.03	S3 Standard storage (1 GB per month)
Total	791.62	

Total Cost (3 months) : 3 x 791.62 = 2374.86

```
### Install kops ###
curl -LO https://github.com/kubernetes/kops/releases/download/$(curl -s https://api.github.com/repos/kubernetes/kops/releases/latest | grep tag_name | cut -d '"' -f 4)/kops-linux-amd64
chmod +x kops-linux-amd64
sudo mv kops-linux-amd64 /usr/local/bin/kops

### Install kubectl ###
curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt`/bin/linux/amd64/kubectl
chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin/kubectl
kubectl version --client

### Export variabel yang nanti akan dibutuhkan  ###
export bucket_name=k8s-jokoss-site
export KOPS_CLUSTER_NAME=k8s.jokoss.site
export KOPS_STATE_STORE=s3://${bucket_name}

### Create cluster ###
kops create cluster --zones=ap-southeast-1a \
--node-count=3 \
--master-count=1 \
--node-size=t2.micro \
--master-size=t2.medium \
--name=${KOPS_CLUSTER_NAME} \
--ssh-public-key=~/.ssh/id_rsa.pub

kops update cluster --name ${KOPS_CLUSTER_NAME} --yes --admin
kops validate cluster --wait 10m
kubectl get nodes --show-labels
```

```
### Install Ingress ###
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/nginx-0.27.1/deploy/static/mandatory.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/nginx-0.27.1/deploy/static/provider/aws/service-l4.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/nginx-0.27.1/deploy/static/provider/aws/patch-configmap-l4.yaml

git clone https://github.com/kubernetes/ingress-nginx
cd ingress-nginx/deploy/static/provider/aws/
kubectl apply -f deploy.yaml

### Create namespace ###
kubectl create namespace staging
kubectl create namespace production

### Install Certificate Authority ###
snap install helm --classic
ln -s /snap/bin/helm /usr/local/bin/helm
kubectl apply --validate=false -f https://github.com/jetstack/cert-manager/releases/download/v0.14.3/cert-manager.crds.yaml
kubectl create namespace cert-manager
helm repo add jetstack https://charts.jetstack.io
helm repo update
helm --version
helm install cert-manager --namespace cert-manager --version v0.14.3 jetstack/cert-manager
kubectl get pods --namespace cert-manager
```

```
### Setting node autoscaler ###
kops edit instancegroups nodes-ap-southeast-1a
# Change this value
  maxSize: 5
  minSize: 1
kops edit cluster
# Add this value on spec
spec:
  additionalPolicies:
    node: |
      [
        {
          "Effect": "Allow",
          "Action": [
            "autoscaling:DescribeAutoScalingGroups",
            "autoscaling:DescribeAutoScalingInstances",
            "autoscaling:SetDesiredCapacity",
            "autoscaling:DescribeLaunchConfigurations",
            "autoscaling:DescribeTags",
            "autoscaling:TerminateInstanceInAutoScalingGroup"
          ],
          "Resource": ["*"]
        }
      ]
```

```
wget https://raw.githubusercontent.com/kubernetes/autoscaler/master/cluster-autoscaler/cloudprovider/aws/examples/cluster-autoscaler-autodiscover.yaml
nano cluster-autoscaler-autodiscover.yaml
# Change this value
    spec:
      serviceAccountName: cluster-autoscaler
      containers:
        - image: asia.gcr.io/k8s-artifacts-prod/autoscaling/cluster-autoscaler:v1.15.6
          name: cluster-autoscaler
          resources:
            limits:
              cpu: 100m
              memory: 300Mi
            requests:
              cpu: 100m
              memory: 300Mi
          command:
            - ./cluster-autoscaler
            - --v=4
            - --stderrthreshold=info
            - --cloud-provider=aws
            - --skip-nodes-with-local-storage=false
            - --expander=least-waste
            - --nodes=1:5:nodes-ap-southeast-1a.k8s.jokoss.site
            - --node-group-auto-discovery=asg:tag=k8s.io/cluster-autoscaler/enabled,k8s.io/cluster-autoscaler/k8s.jokoss.site
          env:
            - name: AWS_REGION
              value: ap-southeast-1
          volumeMounts:
            - name: ssl-certs
              mountPath: /etc/ssl/certs/ca-bundle.crt #/etc/ssl/certs/ca-bundle.crt for Amazon Linux Worker Nodes
              readOnly: true
          imagePullPolicy: "Always"
      volumes:
        - name: ssl-certs
          hostPath:
            path: "/etc/ssl/certs/ca-bundle.crt"
```

```
### Deploy landing page ###
kubectl apply -f landing-cert.yml
kubectl apply -f landing-page.yml

### Deploy pesbuk ###
kubectl apply -f pesbuk-cert.yml
kubectl apply -f pesbuk-kube.yml
```

```
### Landing page Dockerfile ###
FROM nginx

COPY . /usr/share/nginx/html
```

```yaml
### Landing page cert ###
apiVersion: cert-manager.io/v1alpha2
kind: Issuer
metadata:
  name: landing-prod
  namespace: production
spec:
  acme:
    # The ACME server URL
    server: https://acme-v02.api.letsencrypt.org/directory
    # Email address used for ACME registration
    email: johno.smakaduta@gmail.com
    # Name of a secret used to store the ACME account private key
    privateKeySecretRef:
      name: landing-prod
    # Enable the HTTP-01 challenge provider
    solvers:
    - selector: {}
      http01:
        ingress:
          class:  nginx
```

```yaml
### Landing page deployment ###
apiVersion: apps/v1
kind: Deployment
metadata:
  name: landing-page-deployment
  namespace: production
  labels:
    name: landing-page
spec:
  replicas: 1
  selector:
    matchLabels:
      name: landing-page
  template:
    metadata:
      labels:
        name: landing-page
    spec:
      containers:
      - name: landing-page
        image: johnojss/landing-page:v1
        ports:
        - containerPort: 80
```

```yaml
---
### Landing page service ###
apiVersion: v1
kind: Service
metadata:
  name: landing-page-service
  namespace: production
spec:
  ports:
    - port: 80
  selector:
    name: landing-page
```

```yaml
---
### Landing page ingress ###
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: landing-page-ingress
  namespace: production
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
    kubernetes.io/ingress.class: "nginx"
    cert-manager.io/issuer: "landing-prod"
spec:
  rules:
  - host: profile.jokoss.site
    http:
      paths:
      - backend:
          serviceName: landing-page-service
          servicePort: 80

  tls:
  - hosts:
    - profile.jokoss.site
    secretName: landing-tls
```

```
### Pesbuk Dockerfile ###
FROM ubuntu:bionic
RUN apt-get update
RUN apt-get install -y nginx php-fpm php-mysql curl mysql-client
RUN echo "\ndaemon off;" >> /etc/nginx/nginx.conf
RUN sed -i -e "s/;\?daemonize\s*=\s*yes/daemonize = no/g" /etc/php/7.2/fpm/php-fpm.conf
#COPY FILE
COPY . /var/www/html/
COPY ./www.conf /etc/php/7.2/fpm/pool.d/
# Nginx config
RUN rm /etc/nginx/sites-enabled/default
ADD ./pesbuk.conf /etc/nginx/sites-available/
RUN ln -s /etc/nginx/sites-available/pesbuk.conf /etc/nginx/sites-enabled/pesbuk.conf
# Expose ports.
EXPOSE 80
# RUN PHP and NGINX
CMD service php7.2-fpm start && nginx
```

```
### Pesbuk cert ###
apiVersion: cert-manager.io/v1alpha2
kind: Issuer
metadata:
  name: pesbuk-prod
  namespace: production
spec:
  acme:
    # The ACME server URL
    server: https://acme-v02.api.letsencrypt.org/directory
    # Email address used for ACME registration
    email: johno.smakaduta@gmail.com
    # Name of a secret used to store the ACME account private key
    privateKeySecretRef:
      name: pesbuk-prod
    # Enable the HTTP-01 challenge provider
    solvers:
    - selector: {}
      http01:
        ingress:
          class:  nginx
```

```yaml
### Pesbuk Secret ###
apiVersion: v1
kind: Secret
metadata:
  name: pesbuk-secret
  namespace: production
type: Opaque
data:
  DB_USER: YWRtaW4=
  DB_PASS: REJiMDB0QzFsc3kj
```

```yaml
---
### Pesbuk Deployment ###
apiVersion: apps/v1
kind: Deployment
metadata:
  name: pesbuk-deployment
  namespace: production
spec:
  replicas: 3
  selector:
    matchLabels:
      name: pesbuk
  template:
    metadata:
      labels:
        name: pesbuk
    spec:
      containers:
      - name: pesbuk
        image: johnojss/pesbuk:v3
        ports:
        - containerPort: 8080
        env:
          - name: DB_HOST
            value: "databasecilsy.cqtgouliw6ug.ap-southeast-1.rds.amazonaws.com"
          - name: DB_USER
            valueFrom:
              secretKeyRef:
                name: pesbuk-secret
                key: DB_USER
          - name: DB_PASS
            valueFrom:
              secretKeyRef:
                name: pesbuk-secret
                key: DB_PASS
```

```yaml
---
### Pesbuk Service ###
apiVersion: v1
kind: Service
metadata:
  name: pesbuk-service
  namespace: production
spec:
  ports:
    - port: 8080
      protocol: TCP
      targetPort: 80
  selector:
    name: pesbuk
```

```yaml
---
### Pesbuk Ingress ###
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: pesbuk-ingress
  namespace: production
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
    kubernetes.io/ingress.class: "nginx"
    cert-manager.io/issuer: "pesbuk-prod"
spec:
  rules:
  - host: pesbuk.jokoss.site
    http:
      paths:
      - backend:
          serviceName: pesbuk-service
          servicePort: 8080
  tls:
  - hosts:
    - pesbuk.jokoss.site
    secretName: pesbuk-tls
```

```yaml
### Wordpress cert ###
apiVersion: cert-manager.io/v1alpha2
kind: ClusterIssuer
metadata:
  name: wordpress-prod
  namespace: production
  labels:
    name: wordpress-prod
spec:
  acme:
    email: johno.smakaduta@gmail.com
    privateKeySecretRef:
      name: letsencrypt-prod
    server: https://acme-v02.api.letsencrypt.org/directory
    solvers:
    - http01:
        ingress:
          class: nginx
```

```
### Deploy wordpress using helm ###
kubectl apply -f wordpress-cert.yaml
helm repo add bitnami https://charts.bitnami.com/bitnami
helm install wordpress-deployment bitnami/wordpress \
--namespace=production \
--set mariadb.enabled=false \
--set externalDatabase.host="databasecilsy.cqtgouliw6ug.ap-southeast-1.rds.amazonaws.com" \
--set externalDatabase.user=admin \
--set externalDatabase.password=DBb00tCilsy# \
--set externalDatabase.database=wordpress --set externalDatabase.port=3306 \
--set service.type=ClusterIP \
--set ingress.enabled=true \
--set ingress.certManager=true \
--set ingress.annotations."kubernetes\.io/ingress\.class"=nginx --set ingress.annotations."cert-manager\.io/cluster-issuer"=wordpress-prod
--set ingress.hostname=blog.jokoss.site \
--set ingress.extraTls[0].hosts[0]=blog.jokoss.site \
--set ingress.extraTls[0].secretName=wordpress.local-tls
```

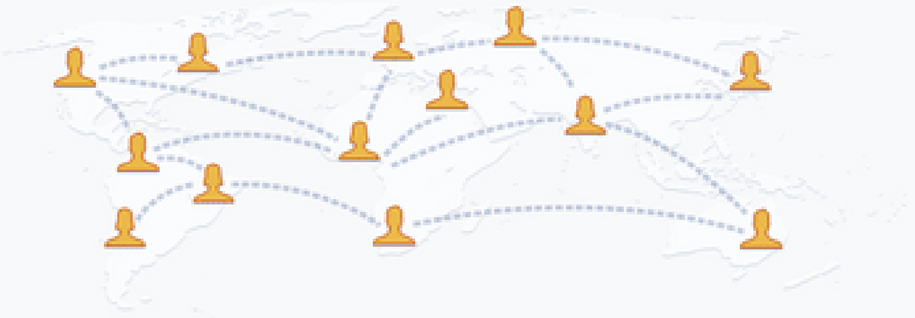
Record name	Type	Routing policy	Differentiator	Value/Route traffic to
jokoss.site	A	Simple	-	54.251.28.56
jokoss.site	NS	Simple	-	ns-964.awsdns-56.net. ns-1996.awsdns-57.co.uk. ns-92.awsdns-11.com. ns-1493.awsdns-58.org.
jokoss.site	SOA	Simple	-	ns-964.awsdns-56.net. awsdns-hostmaster.amazon.com. 1 7200 900 1209600 86400
blog.jokoss.site	CNAME	Simple	-	a797819a74d91421a9071591c7bed808-5801c6c97d660986.elb.ap-southeast-1.amazonaws.com
cilsy-api.jokoss.site	CNAME	Simple	-	ac159b7e717e045f3bf10ca6d776cdc8-fd85acf7666e3753.elb.ap-southeast-1.amazonaws.com
api.k8s.jokoss.site	A	Simple	-	13.212.26.81
api.internal.k8s.jokoss.site	A	Simple	-	172.20.41.183
kops-controller.internal.k8s.jokoss.site	A	Simple	-	172.20.41.183
pesbuk.jokoss.site	CNAME	Simple	-	a797819a74d91421a9071591c7bed808-5801c6c97d660986.elb.ap-southeast-1.amazonaws.com
profile.jokoss.site	CNAME	Simple	-	a797819a74d91421a9071591c7bed808-5801c6c97d660986.elb.ap-southeast-1.amazonaws.com

pesbuk.jokoss.site

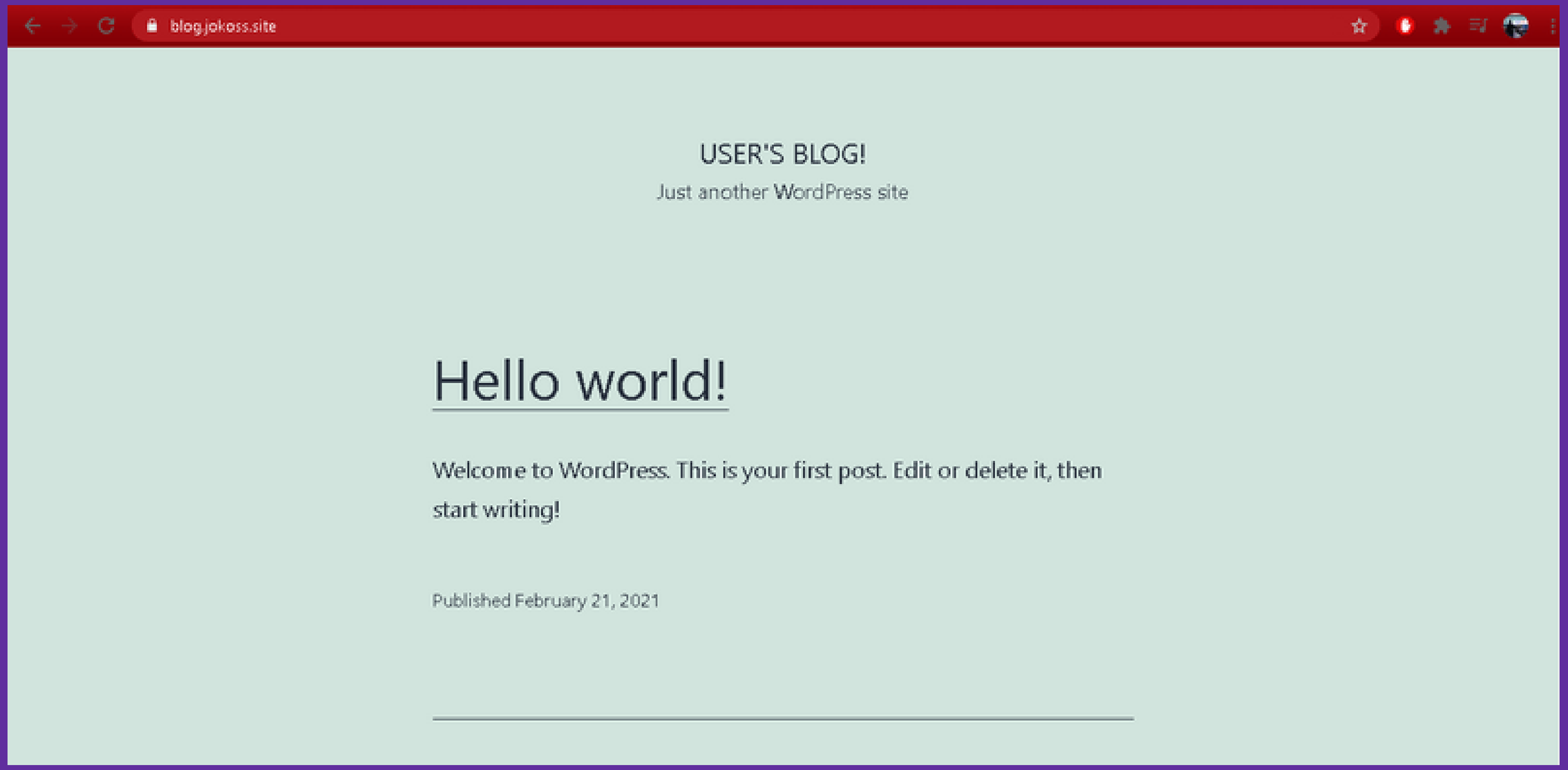
Selamat datang di Pesbuk

Bergabunglah bersama jutaan orang lainnya...

Masuk Daftar




profile.jakass.site
TEKKADAN
ABOUT
SERVICES
PORTFOLIO
CONTACT
BUAT APLIKASI IMPIANMU
BERSAMA KAMI
Tekkadan merupakan sebuah startup software developent yang menyediakan
pembuatan website beserta sistem yang terintegrasi dengan Cloud Platfrom AWS
TEMUKAN LEBIH BANYAK


blog.jokoss.site
USER'S BLOG!
Just another WordPress site
Hello world!
Welcome to WordPress. This is your first post. Edit or delete it, then start writing!
Published February 21, 2021

```
### Upscale deployment ###
kubectl scale deployment pesbuk-deployment -n production --replicas=20
kubectl scale deployment landing-page-deployment -n production --replicas=20
kubectl scale deployment wordpress-deployment -n production --replicas=20
```

```
root@ip-172-31-38-169:~# kubectl get nodes
NAME                                              STATUS   ROLES    AGE    VERSION
ip-172-20-41-183.ap-southeast-1.compute.internal   Ready    master   21h    v1.19.7
ip-172-20-45-97.ap-southeast-1.compute.internal    Ready    node     118s   v1.19.7
ip-172-20-46-85.ap-southeast-1.compute.internal    Ready    node     119s   v1.19.7
ip-172-20-50-54.ap-southeast-1.compute.internal    Ready    node     117s   v1.19.7
ip-172-20-62-250.ap-southeast-1.compute.internal   Ready    node     21h    v1.19.7
ip-172-20-63-144.ap-southeast-1.compute.internal   Ready    node     21h    v1.19.7
ip-172-20-63-155.ap-southeast-1.compute.internal   Ready    node     21h    v1.19.7
root@ip-172-31-38-169:~#
```

```
root@ip-172-31-38-169:~/pesbuk# kubectl get pods -n production
NAME                                          READY   STATUS    RESTARTS   AGE
landing-page-deployment-f4976f4c4-2586x       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-6hchx       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-bvmcj       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-drfh7       1/1     Running   0          20h
landing-page-deployment-f4976f4c4-h295p       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-h8wv4       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-h9mvm       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-k7784       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-kghb9       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-kt28c       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-lf7b9       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-m6r9c       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-n8zct       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-q6246       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-sbskg       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-smt9v       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-stkc6       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-v5gch       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-xwfk9       1/1     Running   0          5m8s
landing-page-deployment-f4976f4c4-zfp62       1/1     Running   0          5m8s
pesbuk-deployment-6f8855cf7c-2kjc2            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-4kxtx            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-5ttcr            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-5vzk2            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-6v5lt            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-7j7kc            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-7zdd8            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-8frhc            1/1     Running   0          18h
pesbuk-deployment-6f8855cf7c-hpdvh            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-hwttr            1/1     Running   0          18h
pesbuk-deployment-6f8855cf7c-jhbl9            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-kzmnz            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-lh8k5            1/1     Running   0          5m43s
pesbuk-deployment-6f8855cf7c-m9f2w            1/1     Running   0          5m43s
```

Terima Kasih