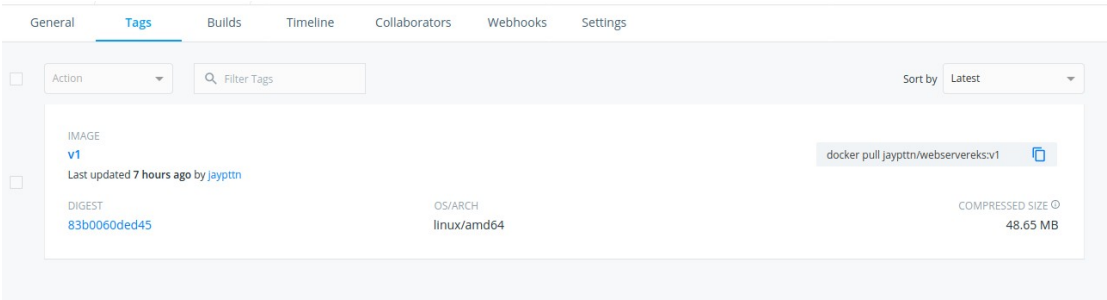
- Project 1
  - Create an ansible playbook to setup WordPress stack on a remote machine
  - Run playbook from local
  - Create user sam and mike on your local Linux system and sync dynamically their ssh keys to the target instance.

- Project 2
  - Create a docker image to run nginx

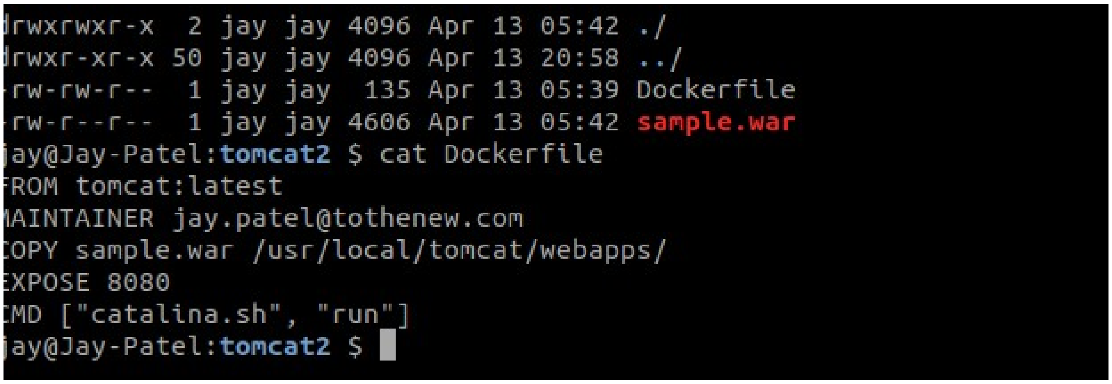
```
FROM nginx:latest
MAINTAINER jay.patel@tothenew.com
RUN rm -v /etc/nginx/conf.d/default.conf
COPY xyz /etc/nginx/conf.d/default.conf
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
~
~
~
~
~
~
~
```

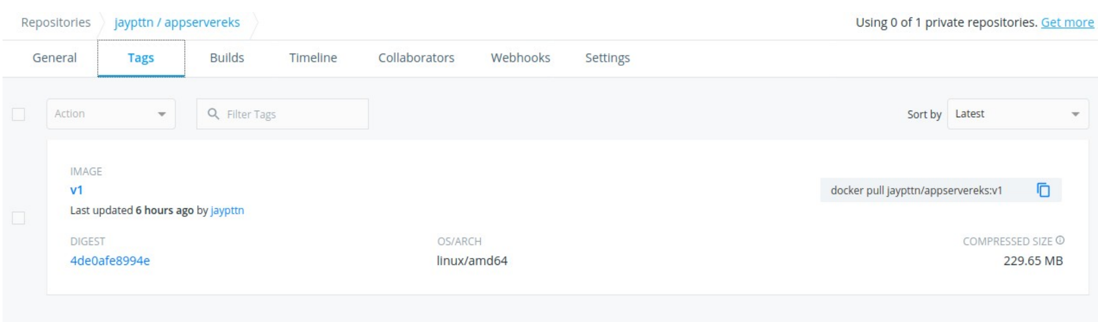
```
jay@Jay-Patel:docker2 $ cat xyz
server {
            listen 80;
            server_name localhost;

            access_log /var/log/nginx/xyz.access.log;
            error_log /var/log/nginx/xyz.error.log;

            error_page 404 /404.html;

            location / {
                proxy_pass http://tomcat:8080/sample/;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
                proxy_set_header Host $http_host;
                proxy_set_header X-NginX-Proxy true;
                proxy_redirect off;
            }

}

jay@Jay-Patel:docker2 $
```

```
jay@Jay-Patel:docker2 $ sudo docker build -t nginx:v2 .
Sending build context to Docker daemon  3.584kB
Step 1/6 : FROM nginx:latest
 ---> ed21b7a8aee9
Step 2/6 : MAINTAINER jay.patel@tothenew.com
 ---> Using cache
 ---> feba487dd4bd
Step 3/6 : RUN rm -v /etc/nginx/conf.d/default.conf
 ---> Using cache
 ---> b6a3cde4a186
Step 4/6 : COPY xyz /etc/nginx/conf.d/default.conf
 ---> Using cache
 ---> a9187b61bc6a
Step 5/6 : EXPOSE 80
 ---> Using cache
 ---> 8426280b3f85
Step 6/6 : CMD ["nginx", "-g", "daemon off;"]
 ---> Using cache
 ---> c2e4cde11308
Successfully built c2e4cde11308
Successfully tagged nginx:v2
jay@Jay-Patel:docker2 $
```

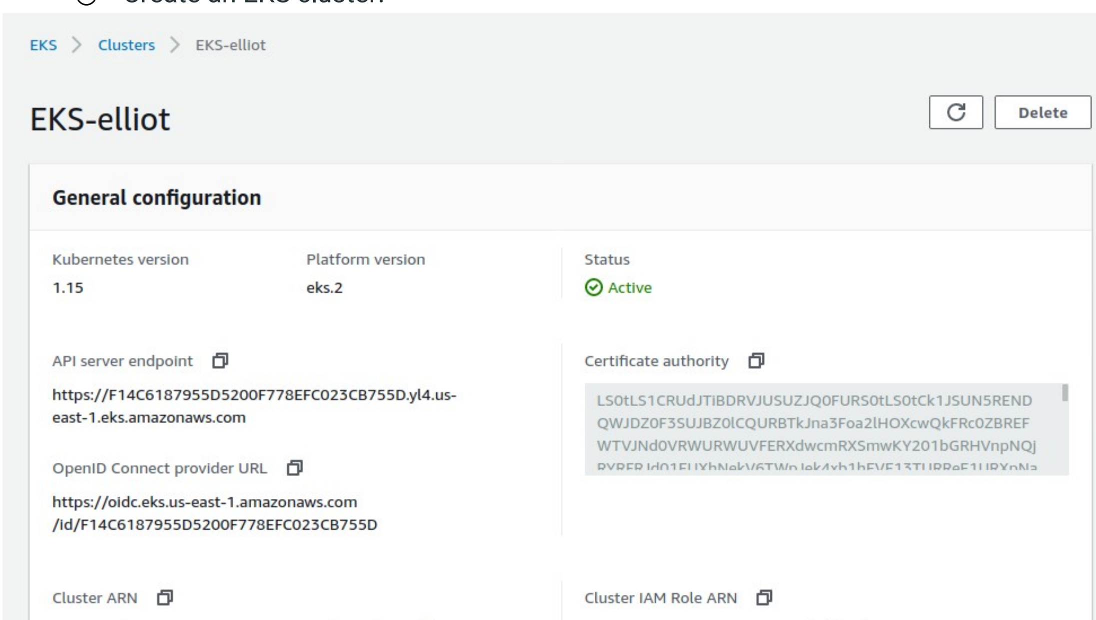


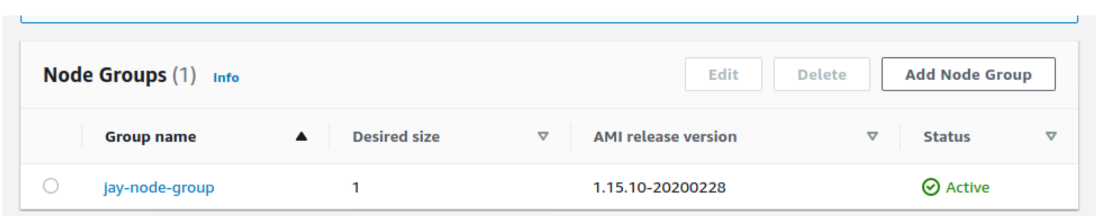
- Create an image of tomcat with a sample war file.





- Create an EKS cluster.





- Serve the application in tomcat using nginx and EKS service

```
jay@Jay-Patel:EKS_file $ cat deployment.yaml
apiVersion: v1
kind: Service
metadata:
 name: nginx
 labels:
   app: nginx
spec:
 selector:
   app: nginx
 ports:
   - protocol: TCP
     port: 80
     targetPort: 80
     nodePort: 32001
 type: NodePort
---
apiVersion: apps/v1
kind: Deployment
metadata:
 name: nginx
spec:
 replicas: 2
 selector:
   matchLabels:
     app: nginx
 template:
   metadata:
     labels:
       app: nginx
   spec:
     containers:
      - name: nginx
        image: jaypttn/webservereks:v1
        ports:
        - containerPort: 80
        resources:
          requests:
```

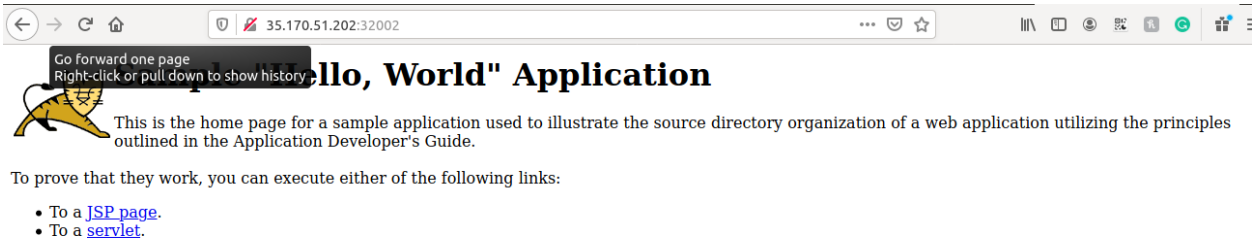
```
      app: nginx
  spec:
    containers:
     - name: nginx
       image: jaypttn/webservereks:v1
       ports:
       - containerPort: 80
       resources:
         requests:
           memory: "200Mi"
           cpu: "350m"
     - name: tomcat
       image: jaypttn/appservereks:v1
       ports:
       - containerPort: 8080
       resources:
         requests:
           memory: "200Mi"
           cpu: "350m"

jay@Jay-Patel:EKS_file $
```

```
jay@Jay-Patel:EKS_file $ aws eks --region us-east-1 update-kubeconfig --name EKS-elliot
Updated context arn:aws:eks:us-east-1:187632318301:cluster/EKS-elliot in /home/jay/.kube/config
jay@Jay-Patel:EKS_file $
```

```
jay@Jay-Patel:EKS_file $ kubectl get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   172.20.0.1   <none>        443/TCP   132m
jay@Jay-Patel:EKS_file $ kubectl get nodes
NAME                           STATUS   ROLES    AGE    VERSION
ip-10-0-2-201.ec2.internal   Ready    <none>   124m   v1.15.10-eks-bac369
jay@Jay-Patel:EKS_file $ kubectl get npods
^C
jay@Jay-Patel:EKS_file $ kubectl get pods
No resources found.
```

```
jay@Jay-Patel:EKS_file $ kubectl apply -f deployment.yaml
service/nginx created
deployment.apps/nginx unchanged
jay@Jay-Patel:EKS_file $ kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
nginx-6bf4954df7-tz6x2   2/2     Running   0          42s
nginx-6bf4954df7-wjs7f   2/2     Running   0          42s
jay@Jay-Patel:EKS_file $ kubectl get svc
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   172.20.0.1       <none>        443/TCP        134m
nginx        NodePort    172.20.228.162   <none>        80:32002/TCP   14s
jay@Jay-Patel:EKS_file $ kubectl get node -o wide
NAME                         STATUS   ROLES    AGE    VERSION              INTERNAL-IP   EXTERNAL-IP     OS-IMAGE         KERNEL-VERSION
          CONTAINER-RUNTIME
ip-10-0-2-201.ec2.internal   Ready    <none>   126m   v1.15.10-eks-bac369   10.0.2.201    35.170.51.202   Amazon Linux 2   4.14.165-133.209.amzn
2.x86_64   docker://18.9.9
jay@Jay-Patel:EKS_file $
```

```
[ec2-user@ip-10-0-2-201 ~]$ curl 10.0.2.97
<html>
<head>
<title>Sample "Hello, World" Application</title>
</head>
<body bgcolor=white>

<table border="0">
<tr>
<td>
<img src="images/tomcat.gif">
</td>
<td>
<h1>Sample "Hello, World" Application</h1>
<p>This is the home page for a sample application used to illustrate th
e
source directory organization of a web application utilizing the princi
ples
outlined in the Application Developer's Guide.
</td>
</tr>
</table>

<p>To prove that they work, you can execute either of the following lin
ks:
<ul>
<li>To a <a href="hello.jsp">JSP page</a>.
<li>To a <a href="hello">servlet</a>.
</ul>

</body>
</html>
```



- Create service and hpa on basis of CPU util for this deployment.

```
jay@Jay-Patel:EKS_file $ git clone https://github.com/kubernetes-sigs/metrics-server.git
Cloning into 'metrics-server'...
        remote: Enumerating objects: 52, done.
remote: Counting objects: 100% (52/52), done.
remote: Compressing objects: 100% (48/48), done.
remote: Total 11859 (delta 22), reused 20 (delta 4), pack-reused 11807
Receiving objects: 100% (11859/11859), 12.36 MiB | 24.00 KiB/s, done.
Resolving deltas: 100% (6176/6176), done.
```

```
jay@Jay-Patel:EKS_file $ ll
total 36
drwxrwxr-x  3 jay jay 4096 Apr 13 21:20  ./
drwxr-xr-x 50 jay jay 4096 Apr 13 21:19  ../
-rw-rw-r--  1 jay jay  801 Apr 13 21:07  deployment.yaml
drwxrwxr-x 10 jay jay 4096 Apr 13 21:29  metrics-server/
-rw-rw-r--  1 jay jay  209 Apr 13 19:52  nginsvc.yaml
-rw-rw-r--  1 jay jay  371 Apr 13 19:53  nginxdev.yaml
-rw-rw-r--  1 jay jay  425 Apr 13 19:38  tomcatdev.yaml
-rw-rw-r--  1 jay jay  249 Apr 13 19:22  tomcatsvc.yaml
-rw-rw-r--  1 jay jay  534 Apr 13 17:44 'ubuntu@3.208.90.207'
jay@Jay-Patel:EKS_file $ cd metrics-server/
jay@Jay-Patel:metrics-server (master)$ kubectl create -f deploy/kubernetes/
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
serviceaccount/metrics-server created
deployment.apps/metrics-server created
service/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
jay@Jay-Patel:metrics-server (master)$ kubectl get ns
NAME              STATUS   AGE
default           Active   156m
kube-node-lease   Active   156m
kube-public       Active   156m
kube-system       Active   156m
jay@Jay-Patel:metrics-server (master)$ kubectl get pods --all-namespaces
NAMESPACE     NAME                              READY   STATUS    RESTARTS   AGE
default       nginx-6bf4954df7-tz6x2            2/2     Running   0          23m
default       nginx-6bf4954df7-wjs7f            2/2     Running   0          23m
kube-system   aws-node-5g7mn                    1/1     Running   0          148m
kube-system   coredns-59dfd6b59f-jxd56          1/1     Running   0          156m
kube-system   coredns-59dfd6b59f-xjtpv          1/1     Running   0          156m
kube-system   kube-proxy-sz4nc                  1/1     Running   0          148m
kube-system   metrics-server-7668599459-s87nh    1/1     Running   0          26s
jay@Jay-Patel:metrics-server (master)$
```

```
jay@Jay-Patel:EKS_file $ kubectl apply -f hpa.yml
E0413 21:33:57.774798   22767 round_trippers.go:174] CancelRequest not implemented by *exec.roundTripper
E0413 21:33:57.775161   22767 request.go:853] Unexpected error when reading response body: &http.httpError{err:"context deadline exceeded (Clien
t.Timeout exceeded while reading body)", timeout:true}
horizontalpodautoscaler.autoscaling/hpa-deployment created
jay@Jay-Patel:EKS_file $
```

```
jay@Jay-Patel:EKS_file $ kubectl apply -f hpa.yml
E0413 21:33:57.774798   22767 round_trippers.go:174] CancelRequest not implemented by *exec.roundTripper
E0413 21:33:57.775161   22767 request.go:853] Unexpected error when reading response body: &http.httpError{err:"context deadline exceeded (Clien
t.Timeout exceeded while reading body)", timeout:true}
horizontalpodautoscaler.autoscaling/hpa-deployment created
jay@Jay-Patel:EKS_file $ kubectl get hpa
NAME             REFERENCE           TARGETS         MINPODS   MAXPODS   REPLICAS   AGE
hpa-deployment   Deployment/nginx    <unknown>/55%   1         5         0          15s
jay@Jay-Patel:EKS_file $ kubectl get hpa -o wide
NAME             REFERENCE           TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
hpa-deployment   Deployment/nginx    0%/55%    1         5         2          27s
jay@Jay-Patel:EKS_file $
```

- Project 3 : Deploy Wordpress on this cluster and it should connect to a MySQL installed on Kubernetes. Using PVC

```
jay@Jay-Patel:wordpress_mysql $ cat mysql.yml
apiVersion: v1
kind: ConfigMap
metadata:
  name: configmysql
data:
  MYSQL_DATABASE: wordpress
  MYSQL_ROOT_PASSWORD: password
  MYSQL_PASSWORD: password
```

```
jay@Jay-Patel:wordpress_mysql $ cat wordpress.yml
apiVersion: v1
kind: ConfigMap
metadata:
        name: configwordpress
data:
        MYSQL_ROOT_PASSWORD: password
        MYSQL_USER: root
        MYSQL_PASSWORD: password
        WORDPRESS_DB_HOST: wordpress-mysql:3306
        WORDPRESS_DB_USER: root
        WORDPRESS_DB_PASSWORD: password
        WORDPRESS_DB_NAME: wordpress

jay@Jay-Patel:wordpress_mysql $
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  ports:
    - port: 3306
  selector:
    app: wordpress
  clusterIP: None
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pv-claim
  labels:
    app: wordpress
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 20Gi
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  selector:
    matchLabels:
      app: wordpress
```

```
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 20Gi
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  selector:
    matchLabels:
      app: wordpress
  template:
    metadata:
      labels:
        app: wordpress
    spec:
      containers:
      - name: dmysql
        image: mysql:5.6
        ports:
          - containerPort: 3306
        envFrom:
        - configMapRef:
           name: configmysql
        volumeMounts:
        - name: mysql-persistent-storage
          mountPath: /var/lib/mysql
      volumes:
      - name: mysql-persistent-storage
        persistentVolumeClaim:
          claimName: mysql-pv-claim
```

```
jay@Jay-Patel:wordpress_mysql $ cat conf

server {
        listen 80;
        server_name localhost;
        index index.php index.html index.htm;
        root /var/www/html;
        location / {
                try_files $uri $uri/ /index.php$is_args$args;
        }
        location ~ \.php$ {
                try_files $uri =404;
                fastcgi_split_path_info ^(.+\.php)(/.+)$;
                fastcgi_pass localhost:9000;
                fastcgi_index index.php;
                include fastcgi_params;
                fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name
                fastcgi_param PATH_INFO $fastcgi_path_info;
        }
        location ~ /\.ht {
                deny all;
        }
        location = /favicon.ico {
                log_not_found off; access_log off;
        }
        location = /robots.txt {
                log_not_found off; access_log off; allow all;
        }
        location ~* \.(css|gif|ico|jpeg|jpg|js|png)$ {
                expires max;
                log_not_found off;
        }
}
```

```
jay@Jay-Patel:wordpress_mysql $ cat Dockerfile
FROM nginx
COPY conf /etc/nginx/conf.d/default.conf
RUN mkdir -p /var/www/html
EXPOSE 80
```

```
jay@Jay-Patel:wordpress_mysql $ sudo docker build -t mysql-wordpress .
Sending build context to Docker daemon  9.216kB
Step 1/4 : FROM nginx
 ---> ed21b7a8aee9
Step 2/4 : COPY conf /etc/nginx/conf.d/default.conf
 ---> 21a2d4873867
Step 3/4 : RUN mkdir -p /var/www/html
 ---> Running in bef7180de23f
Removing intermediate container bef7180de23f
 ---> 6a603073b85b
Step 4/4 : EXPOSE 80
 ---> Running in 650a37e83986
Removing intermediate container 650a37e83986
 ---> e3bd7fcf4eab
Successfully built e3bd7fcf4eab
Successfully tagged mysql-wordpress:latest
jay@Jay-Patel:wordpress_mysql $
```

```
jay@Jay-Patel:wordpress_mysql $ sudo docker tag mysql-wordpress jaypttn/mysql_wordpress_pvc:v1

jay@Jay-Patel:wordpress_mysql $
jay@Jay-Patel:wordpress_mysql $ sudo docker push jaypttn/mysql_wordpress_pvc:v1
The push refers to repository [docker.io/jaypttn/mysql_wordpress_pvc]
b67037eebfe8: Pushed
52cc92902e91: Pushed
d37eecb5b769: Mounted from jaypttn/webservereks
99134ec7f247: Mounted from jaypttn/webservereks
c3a984abe8a8: Mounted from jaypttn/webservereks
v1: digest: sha256:083896b1ca0a9fa6e6a00469dec748e955a7cb141c46e8c581b32490bd45754d size: 1362
jay@Jay-Patel:wordpress_mysql $
```

```yaml
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: wp-pv-claim
  labels:
    app: wordpress
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 20Gi
---

apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
######## PHP Container #########
      - name: dphp
        image: wordpress:5.1.1-fpm-alpine
        ports:
          - containerPort: 9000
```

```
####### PHP Container #########
      - name: dphp
        image: wordpress:5.1.1-fpm-alpine
        ports:
          - containerPort: 9000
        envFrom:
        - configMapRef:
           name: configwordpress
        volumeMounts:
          - name: wordpress-persistent-storage
            mountPath: /var/www/html
####### Nginx Container ############
      - name: jnginx
        image: jaypttn/mysql_wordpress_pvc:v1
        volumeMounts:
          - name: wordpress-persistent-storage
            mountPath: /var/www/html
      volumes:
      - name: wordpress-persistent-storage
        persistentVolumeClaim:
          claimName: wp-pv-claim

---
apiVersion: v1
kind: Service
metadata:
        name: my-service
spec:
        type: NodePort
        selector:
                app: nginx
        ports:
                - protocol: TCP
                  port: 80
                  targetPort: 80
```

```
jay@Jay-Patel:wordpress_mysql $ kubectl apply -f mysql.yml
configmap/configmysql created
```

```
jay@Jay-Patel:wordpress_mysql $ kubectl apply -f mysql_deploy.yml
service/wordpress-mysql created
persistentvolumeclaim/mysql-pv-claim created
deployment.apps/wordpress-mysql created
```

```
jay@Jay-Patel:wordpress_mysql $ kubectl apply -f wordpress.yml
E0413 21:47:38.321913   25859 round_trippers.go:174] CancelRequest not implemented by *exec.roundTripper
E0413 21:47:38.322185   25859 request.go:853] Unexpected error when reading response body: &http.httpError{err:"context deadline exceeded (C
t.Timeout exceeded while reading body)", timeout:true}
configmap/configwordpress created
```

```
jay@Jay-Patel:wordpress_mysql $ kubectl apply -f wp_deploy.yml
persistentvolumeclaim/wp-pv-claim created
deployment.apps/nginx-deployment created
service/my-service created
jay@Jay-Patel:wordpress_mysql $
```

```
jay@Jay-Patel:wordpress_mysql $ kubectl get nodes -o wide
NAME                       STATUS   ROLES    AGE    VERSION              INTERNAL-IP   EXTERNAL-IP     OS-IMAGE         KERNEL-VERSION
          CONTAINER-RUNTIME
ip-10-0-2-201.ec2.internal   Ready    <none>   170m   v1.15.10-eks-bac369   10.0.2.201    35.170.51.202   Amazon Linux 2   4.14.165-133.209.amzn
2.x86_64   docker://18.9.9
jay@Jay-Patel:wordpress_mysql $ kubectl get svc
NAME              TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes        ClusterIP   172.20.0.1       <none>        443/TCP        179m
my-service        NodePort    172.20.96.55     <none>        80:31809/TCP   71s
nginx             NodePort    172.20.228.162   <none>        80:32002/TCP   45m
wordpress-mysql   ClusterIP   None             <none>        3306/TCP       2m1s
jay@Jay-Patel:wordpress_mysql $ ^C
jay@Jay-Patel:wordpress_mysql $ kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
nginx-6bf4954df7-tz6x2             2/2     Running   0          47m
nginx-deployment-697b75894d-c6fnv   2/2     Running   1          3m4s
nginx-deployment-697b75894d-rrqgg   2/2     Running   0          3m4s
wordpress-mysql-6dfcfcc75f-2cjjn    1/1     Running   0          3m50s
jay@Jay-Patel:wordpress_mysql $ kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
nginx-6bf4954df7-tz6x2             2/2     Running   0          54m
nginx-deployment-697b75894d-c6fnv   2/2     Running   1          9m52s
nginx-deployment-697b75894d-rrqgg   2/2     Running   0          9m52s
wordpress-mysql-6dfcfcc75f-2cjjn    1/1     Running   0          10m
jay@Jay-Patel:wordpress_mysql $ kubectl get pods -o wide
NAME                               READY   STATUS    RESTARTS   AGE   IP           NODE                          NOMINATED NODE   READINESS GATE
S
nginx-6bf4954df7-tz6x2             2/2     Running   0          54m   10.0.2.97    ip-10-0-2-201.ec2.internal   <none>           <none>
nginx-deployment-697b75894d-c6fnv   2/2     Running   1          10m   10.0.2.20    ip-10-0-2-201.ec2.internal   <none>           <none>
nginx-deployment-697b75894d-rrqgg   2/2     Running   0          10m   10.0.2.74    ip-10-0-2-201.ec2.internal   <none>           <none>
wordpress-mysql-6dfcfcc75f-2cjjn    1/1     Running   0          10m   10.0.2.251   ip-10-0-2-201.ec2.internal   <none>           <none>
jay@Jay-Patel:wordpress_mysql $ D
```

- Project 4: Create a bash script which lists all the security group rules and delete all the rules in which public access is allowed except 80 and 443 ports.

```
#!/bin/bash
aws ec2 describe-security-groups | tee output.txt
gid=$(jq '.SecurityGroups[].GroupId' output.txt | wc -l)
for (( i=0; i<$gid; i++ ))
do
        count=$(jq '.SecurityGroups['$i'].IpPermissions[].FromPort' output.txt | wc -l)
        for (( j=0; j<$count; j++ ))
        do
                port=$(jq '.SecurityGroups['$i'].IpPermissions['$j'].FromPort' output.txt)
                if (( $port!=80 ))
                then
                        if (( $port!=443 ))
                        then
                                ipv4=$(jq '.SecurityGroups['$i'].IpPermissions['$j'].IpRanges[].CidrIp' output.txt)
                                ipv6=$(jq '.SecurityGroups['$i'].IpPermissions['$j'].Ipv6Ranges[].CidrIpv6' output.txt)
                                if [[ $ipv4 == *"0.0.0.0/0"* || $ipv6 == *"::/0"* ]]
                                then
                                        echo "port : "$port
                                        grpid=$(jq '.SecurityGroups['$i'].GroupId' output.txt | sed 's/"//g' )
                                        aws ec2 revoke-security-group-ingress --group-id "$grpid" --protocol tcp --port $port --cidr 0.0.0.0/0
                                        echo "Successfully deleted rule with port $port ";

                                fi

                        fi
                fi

        done
done
~
~
```

```
        ],
        {
            "Description": "default VPC security group",
            "GroupName": "default",
            "IpPermissions": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [],
                    "Ipv6Ranges": [],
                    "PrefixListIds": [],
                    "UserIdGroupPairs": [
                        {
                            "GroupId": "sg-0ec57657b30ede284",
                            "UserId": "555492600276"
                        }
                    ]
                }
            ],
            "OwnerId": "555492600276",
            "GroupId": "sg-0ec57657b30ede284",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": [],
                    "UserIdGroupPairs": []
                }
            ],
            "VpcId": "vpc-0ce90014d4d6d891c"
        }
    ]
}
port : 22
Successfully deleted rule with port 22
```