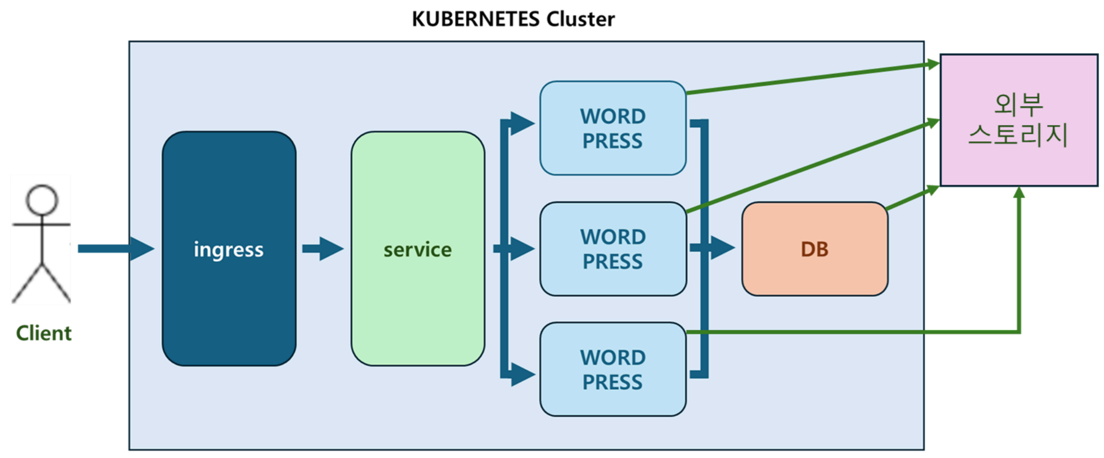
kubernetes를 활용한 wordpress 구성

구성 아키텍쳐



기본 구성

- image
  web - word press
  db   - mysql or mariadb
- volume
  외부 스토리지 - nfs
  control-plan 을 nfs 서버로 사용 가능
- pod
  컨트롤러를 이용한 생성
  사용 컨트롤러는 자유

추가 구성 사항

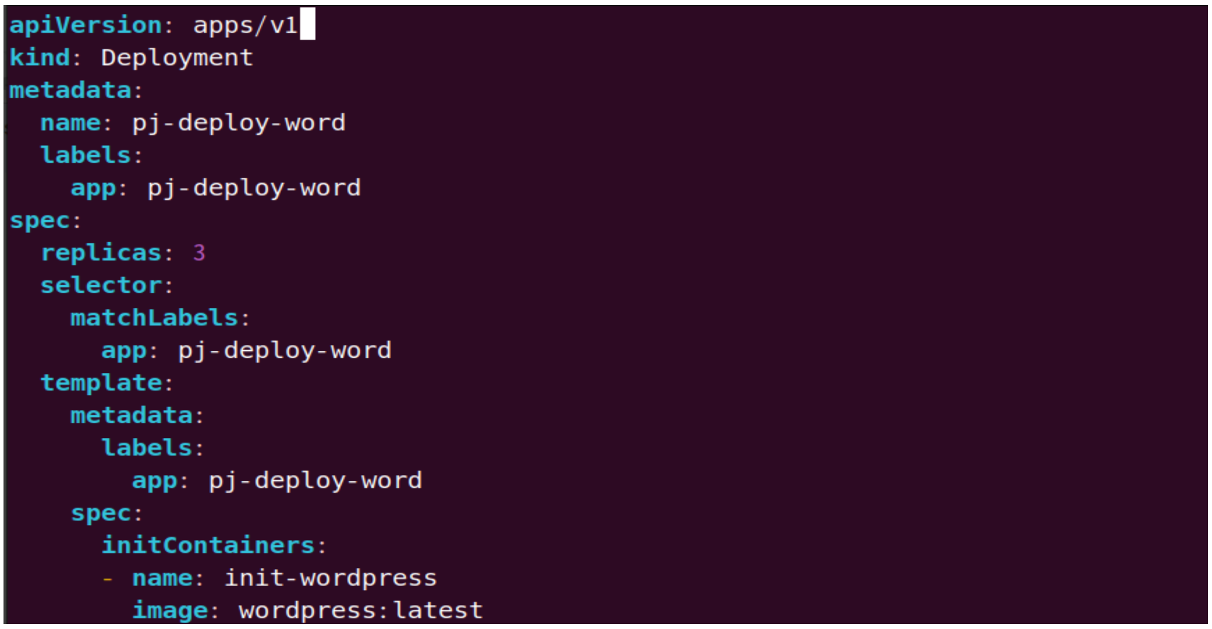
- wordpress image 제작
  Dockerfile 활용

본 프로젝트에서는 쿠버네티스를 활용해 고가용성 웹 어플리케이션을 배포/관리하였습니다.

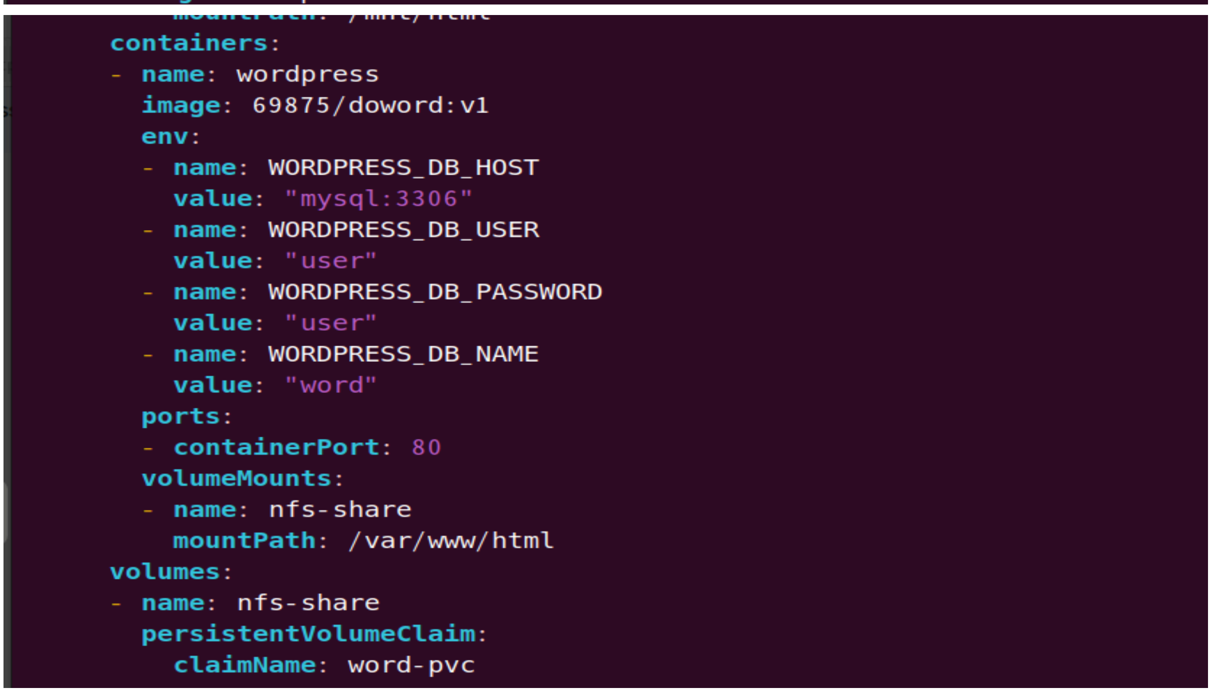
1.1 Image 구성

web과 db를 구성하였습니다.
web은 wordpress를 사용했으며 db는 mysql:5.7 을 사용하였으며
deployment를 사용하여 yaml 파일을 작성 하였습니다.

pj-deploy-word.yml

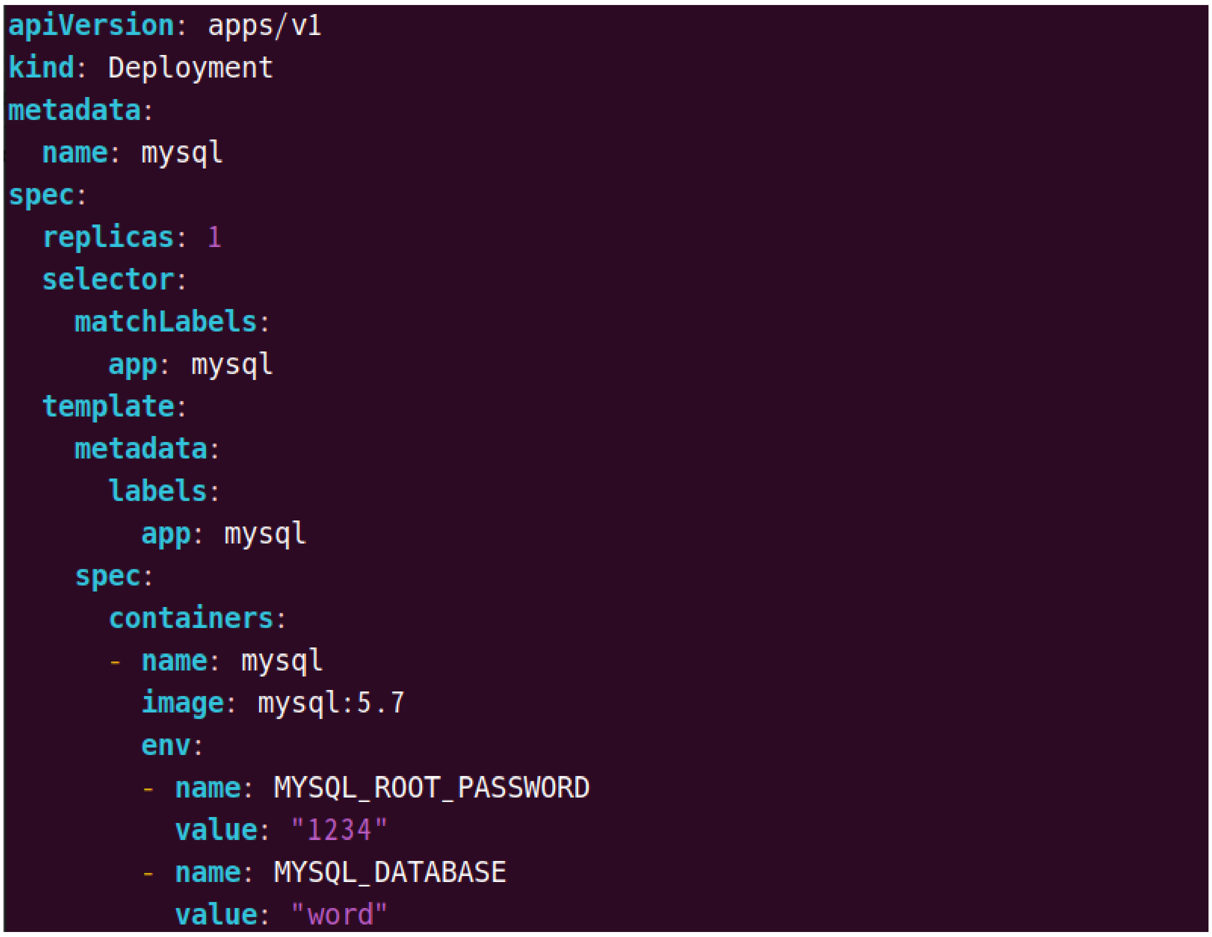


```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: pj-deploy-word
  labels:
    app: pj-deploy-word
spec:
  replicas: 3
  selector:
    matchLabels:
      app: pj-deploy-word
  template:
    metadata:
      labels:
        app: pj-deploy-word
    spec:
      initContainers:
      - name: init-wordpress
        image: wordpress:latest
```

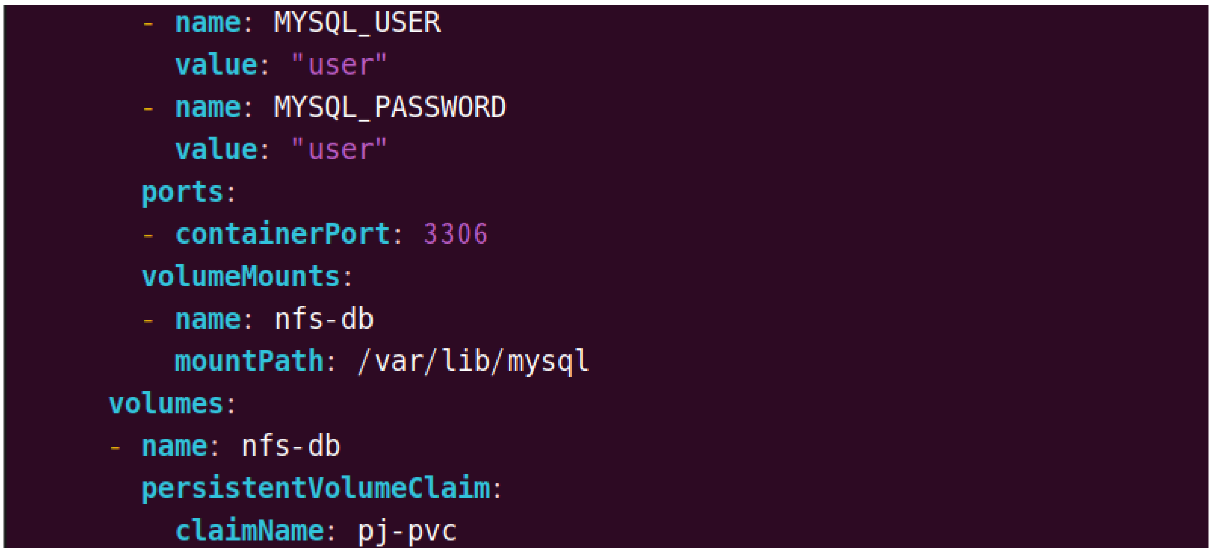


```
      containers:
      - name: wordpress
        image: 69875/doword:v1
        env:
        - name: WORDPRESS_DB_HOST
          value: "mysql:3306"
        - name: WORDPRESS_DB_USER
          value: "user"
        - name: WORDPRESS_DB_PASSWORD
          value: "user"
        - name: WORDPRESS_DB_NAME
          value: "word"
        ports:
        - containerPort: 80
        volumeMounts:
        - name: nfs-share
          mountPath: /var/www/html
      volumes:
      - name: nfs-share
        persistentVolumeClaim:
          claimName: word-pvc
```

pj-db.yml

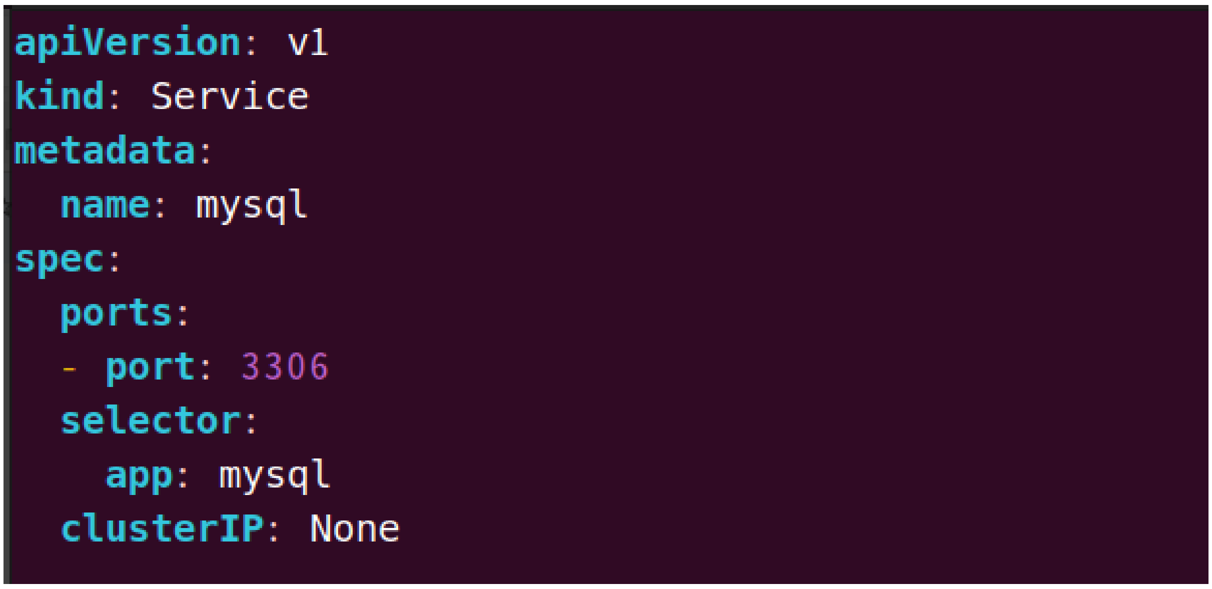


```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - name: mysql
        image: mysql:5.7
        env:
        - name: MYSQL_ROOT_PASSWORD
          value: "1234"
        - name: MYSQL_DATABASE
          value: "word"
```



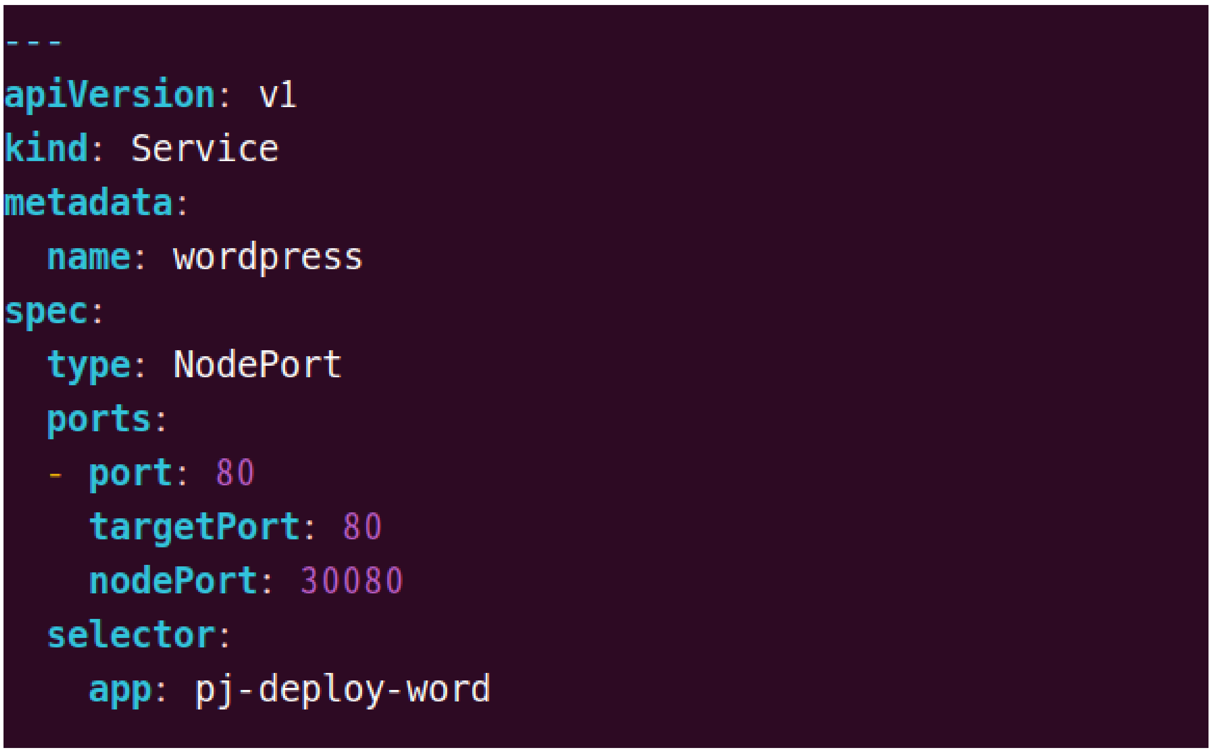
```yaml
        - name: MYSQL_USER
          value: "user"
        - name: MYSQL_PASSWORD
          value: "user"
        ports:
        - containerPort: 3306
        volumeMounts:
        - name: nfs-db
          mountPath: /var/lib/mysql
      volumes:
      - name: nfs-db
        persistentVolumeClaim:
          claimName: pj-pvc
```

pj-db-db.yml



```yaml
apiVersion: v1
kind: Service
metadata:
  name: mysql
spec:
  ports:
  - port: 3306
  selector:
    app: mysql
  clusterIP: None
```

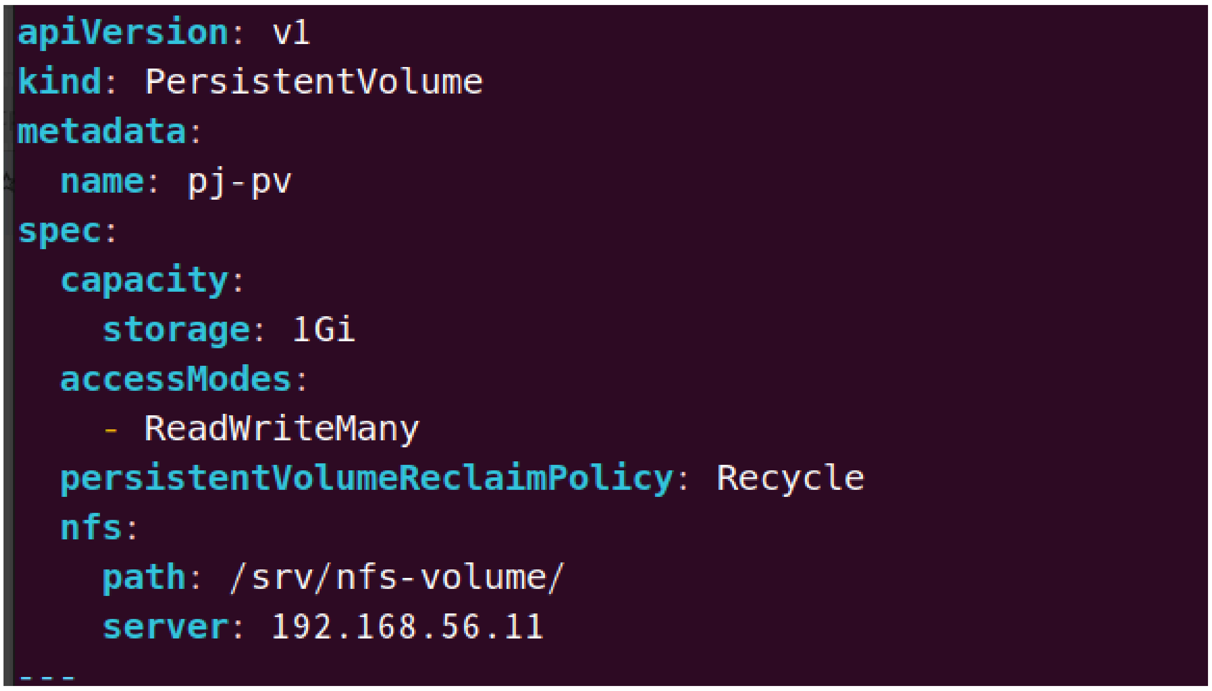
pj-svc-word.yml



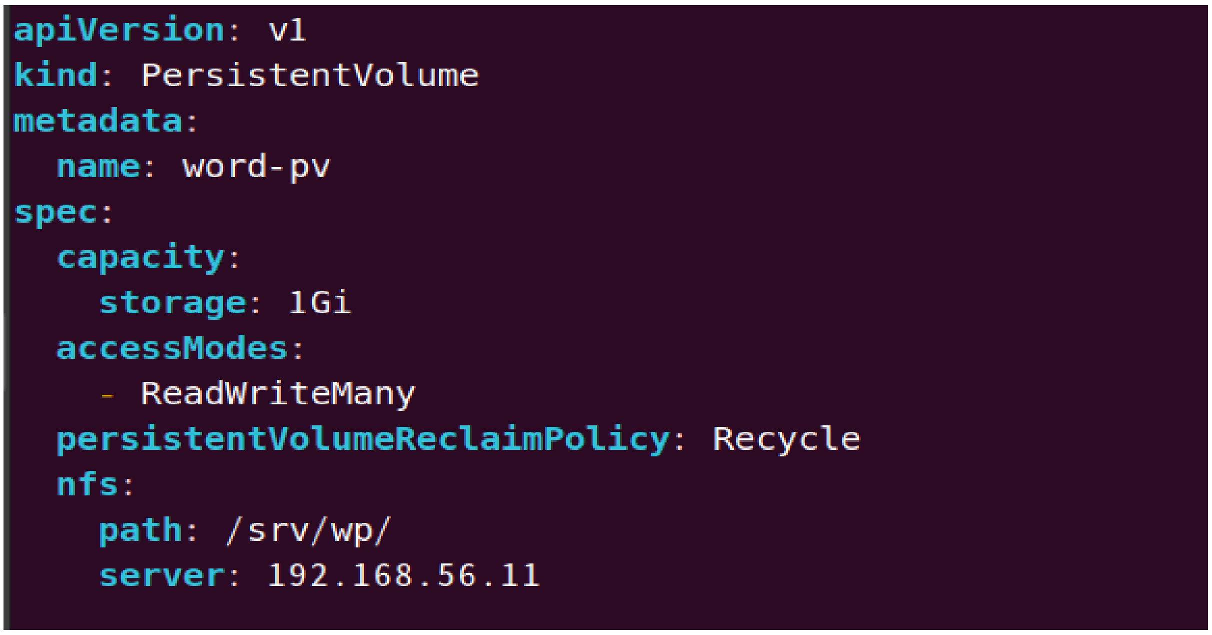
```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: wordpress
spec:
  type: NodePort
  ports:
  - port: 80
    targetPort: 80
    nodePort: 30080
  selector:
    app: pj-deploy-word
```

1.2 Volume 구성

외부 스토리지 구성을 위해 nfs 방식을 사용했습니다.
정적 프로비저닝을 사용하여 연결하였습니다.
두개의 pv 파일을 만들어 db는 /srv/nfs-volume, wordpress는 /srv/wp에 위치합니다.

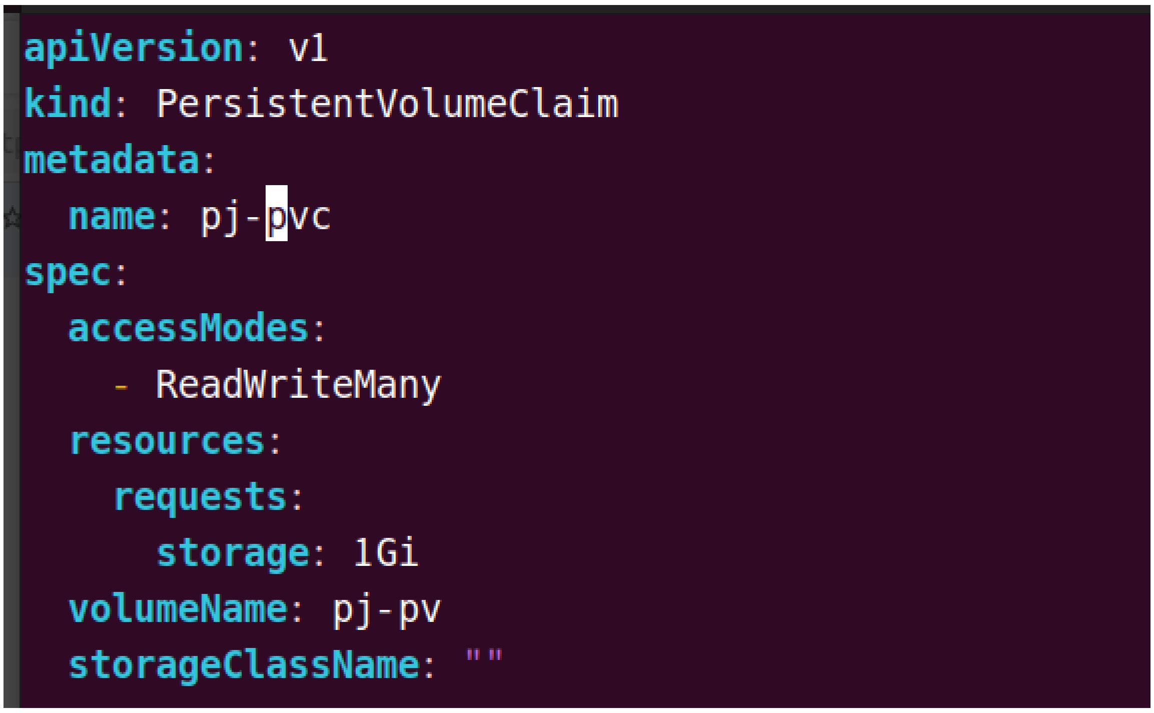


```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pj-pv
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Recycle
  nfs:
    path: /srv/nfs-volume/
    server: 192.168.56.11
---
```

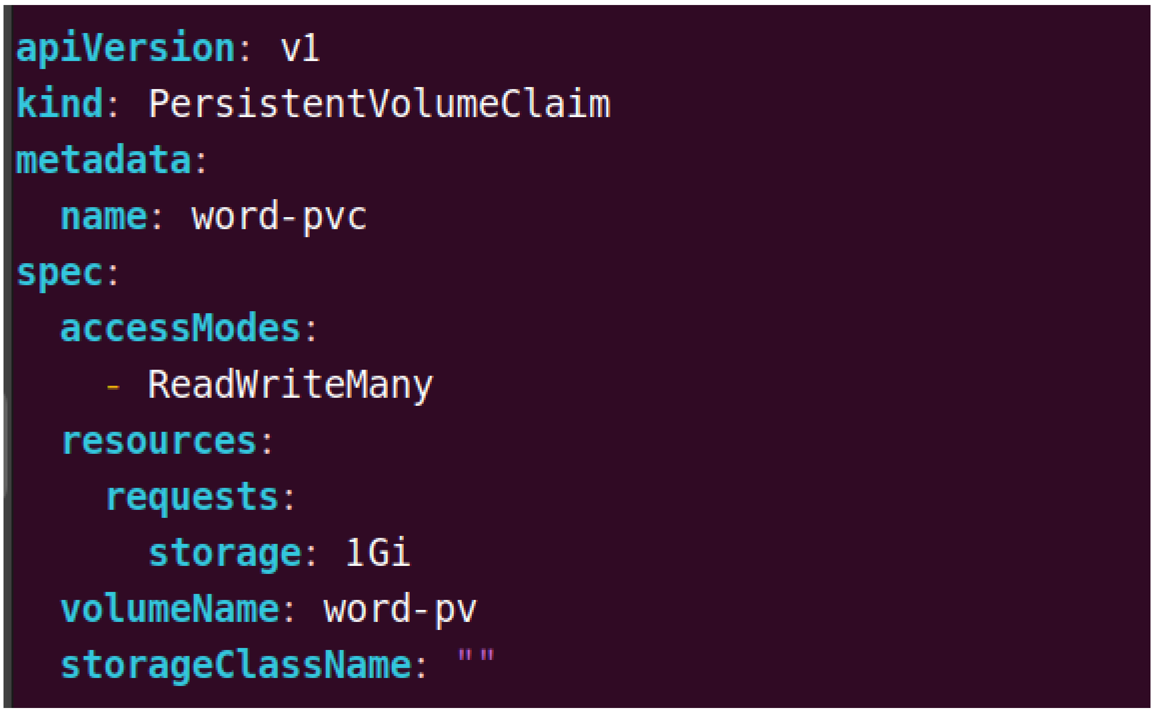


```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: word-pv
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Recycle
  nfs:
    path: /srv/wp/
    server: 192.168.56.11
```

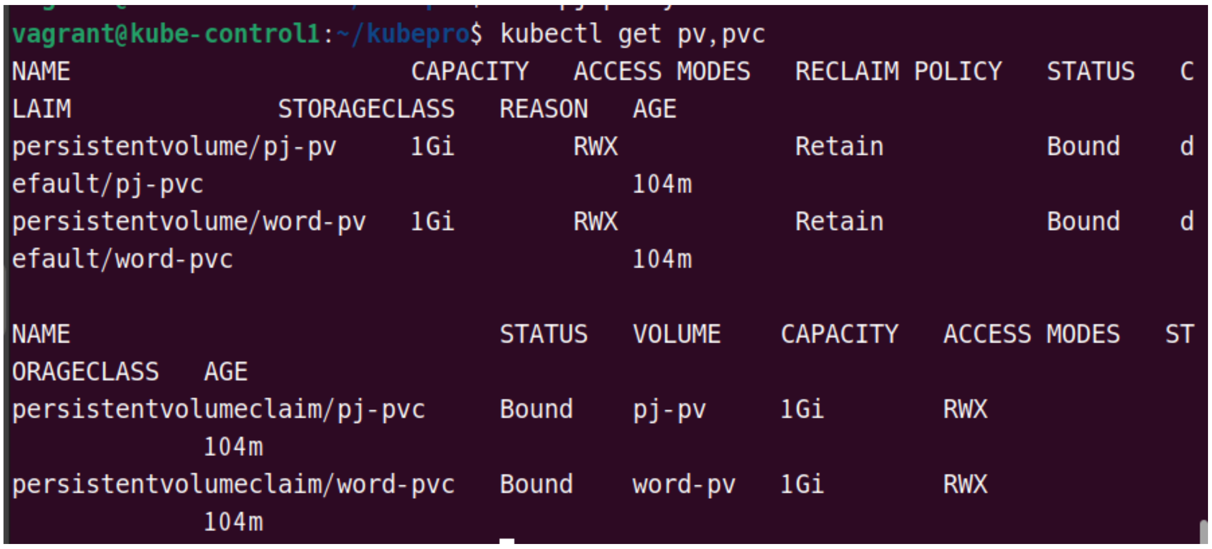
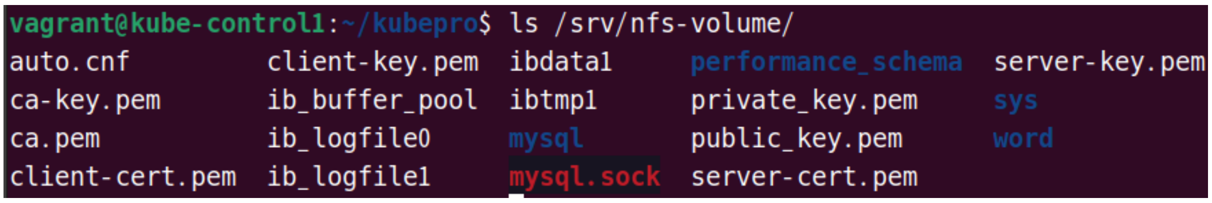
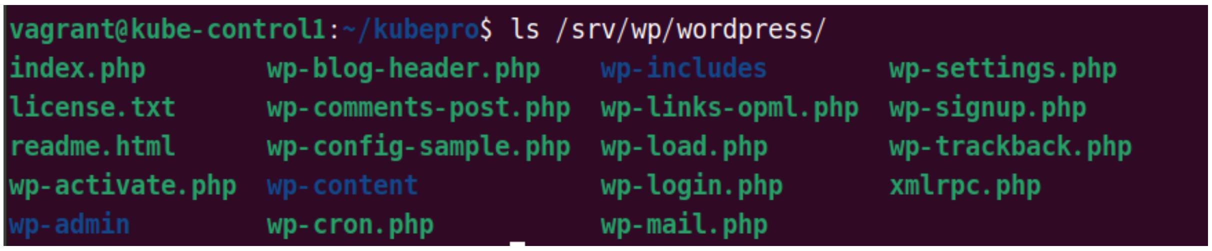
마찬가지로 두개의 pvc 파일을 생성하여 pv와 연결하였습니다.



```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pj-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  volumeName: pj-pv
  storageClassName: ""
```

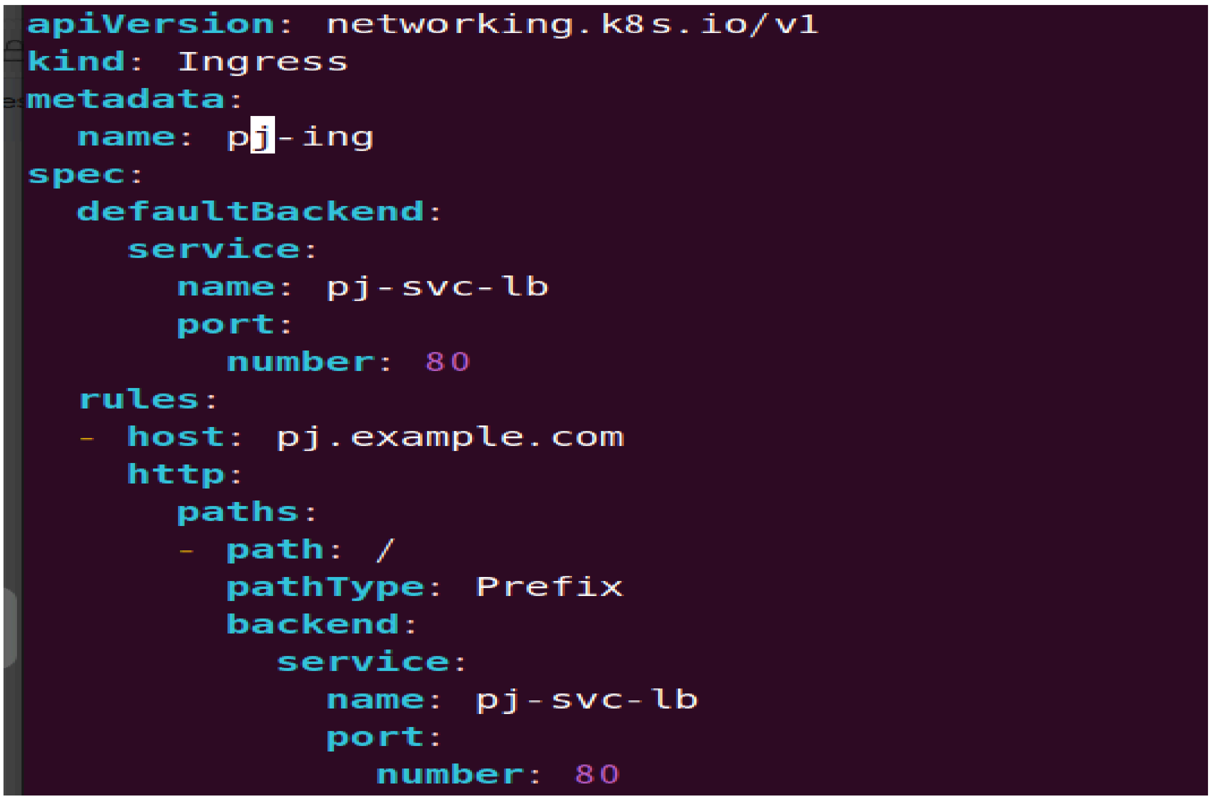


```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: word-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  volumeName: word-pv
  storageClassName: ""
```

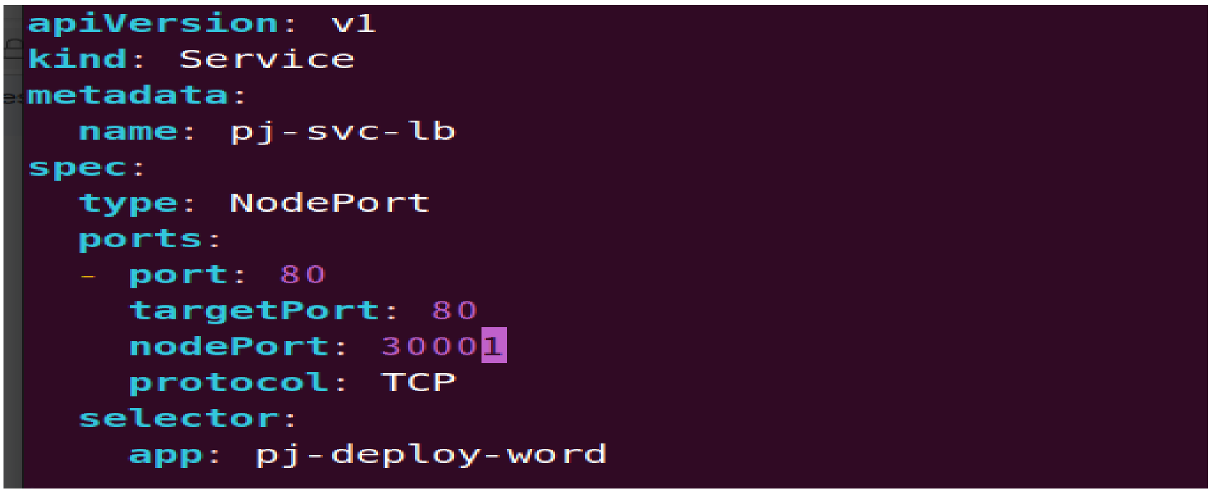

```
vagrant@kube-control1:~/kubepro$ kubectl get pv,pvc
NAME                          CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   C
LAIM                 STORAGECLASS   REASON   AGE
persistentvolume/pj-pv      1Gi        RWX            Retain           Bound    d
efault/pj-pvc                                   104m
persistentvolume/word-pv    1Gi        RWX            Retain           Bound    d
efault/word-pvc                                 104m

NAME                              STATUS   VOLUME    CAPACITY   ACCESS MODES   ST
ORAGECLASS   AGE
persistentvolumeclaim/pj-pvc      Bound    pj-pv     1Gi        RWX
             104m
persistentvolumeclaim/word-pvc    Bound    word-pv   1Gi        RWX
             104m
```

```
vagrant@kube-control1:~/kubepro$ ls /srv/nfs-volume/
auto.cnf         client-key.pem  ibdata1     performance_schema  server-key.pem
ca-key.pem       ib_buffer_pool  ibtmp1      private_key.pem     sys
ca.pem           ib_logfile0     mysql       public_key.pem      word
client-cert.pem  ib_logfile1     mysql.sock  server-cert.pem
```

```
vagrant@kube-control1:~/kubepro$ ls /srv/wp/wordpress/
index.php        wp-blog-header.php    wp-includes        wp-settings.php
license.txt      wp-comments-post.php  wp-links-opml.php  wp-signup.php
readme.html      wp-config-sample.php  wp-load.php        wp-trackback.php
wp-activate.php  wp-content            wp-login.php       xmlrpc.php
wp-admin         wp-cron.php           wp-mail.php
```
db와 wordpress가 연결된것을 볼수있습니다.

1.3 Pod 구성

마지막으로 service와 ingress를 구성하여 아키텍처를 구성하였습니다.

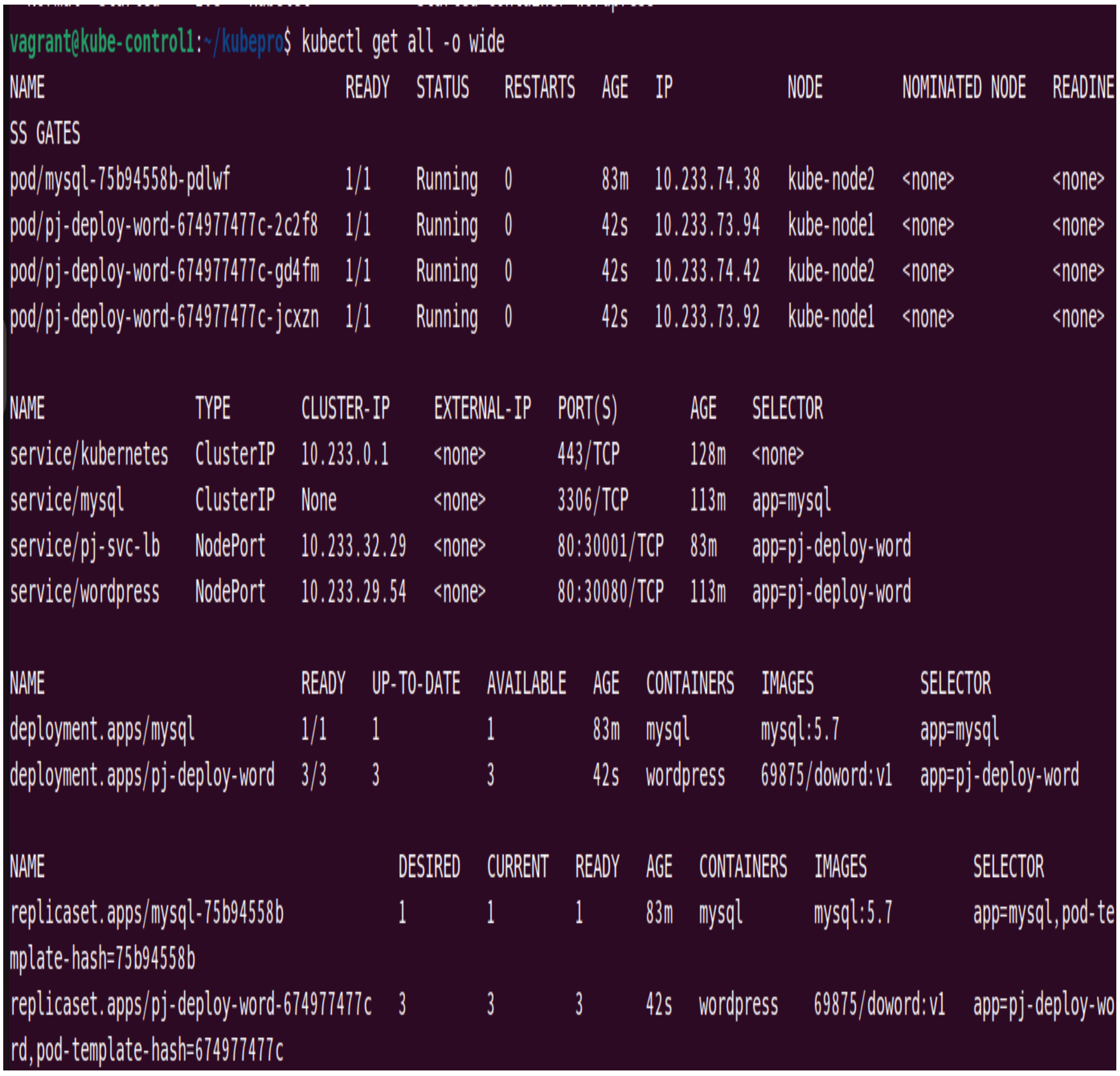


```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: pj-ing
spec:
  defaultBackend:
    service:
      name: pj-svc-lb
      port:
        number: 80
  rules:
  - host: pj.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: pj-svc-lb
            port:
              number: 80
```

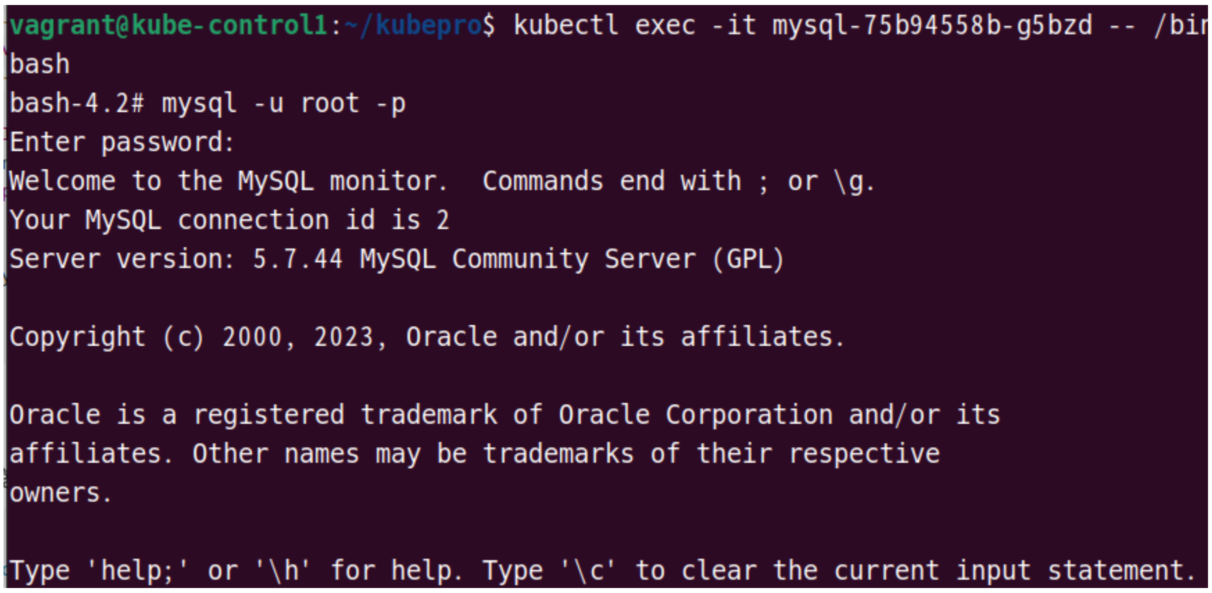


```yaml
apiVersion: v1
kind: Service
metadata:
  name: pj-svc-lb
spec:
  type: NodePort
  ports:
  - port: 80
    targetPort: 80
    nodePort: 30001
    protocol: TCP
  selector:
    app: pj-deploy-word
```

1.4 기본구성 결과

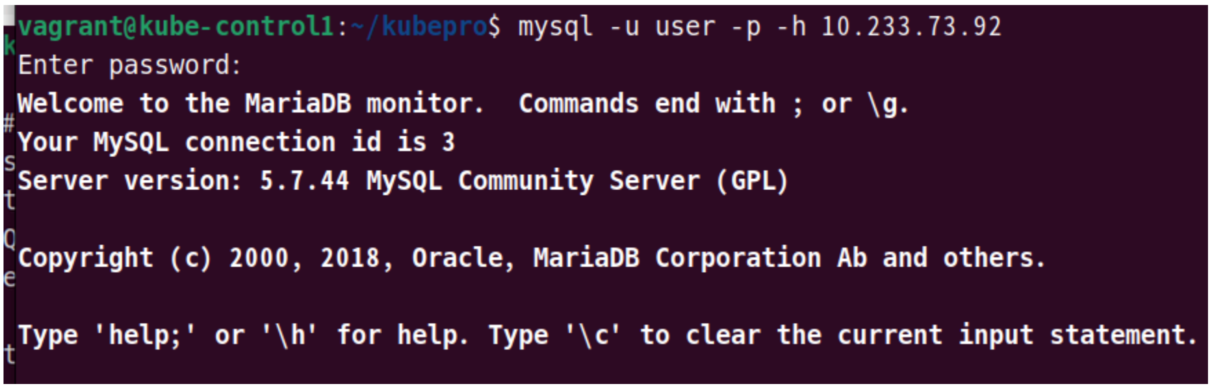


```
vagrant@kube-control1:~/kubepro$ kubectl get all -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP             NODE         NOMINATED NODE   READINE
SS GATES
pod/mysql-75b94558b-pdlwf           1/1     Running   0          83m   10.233.74.38   kube-node2   <none>           <none>
pod/pj-deploy-word-674977477c-2c2f8 1/1     Running   0          42s   10.233.73.94   kube-node1   <none>           <none>
pod/pj-deploy-word-674977477c-gd4fm 1/1     Running   0          42s   10.233.74.42   kube-node2   <none>           <none>
pod/pj-deploy-word-674977477c-jcxzn 1/1     Running   0          42s   10.233.73.92   kube-node1   <none>           <none>

NAME                 TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE    SELECTOR
service/kubernetes   ClusterIP   10.233.0.1     <none>        443/TCP        128m   <none>
service/mysql        ClusterIP   None           <none>        3306/TCP       113m   app=mysql
service/pj-svc-lb    NodePort    10.233.32.29   <none>        80:30001/TCP   83m    app=pj-deploy-word
service/wordpress    NodePort    10.233.29.54   <none>        80:30080/TCP   113m   app=pj-deploy-word

NAME                             READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS   IMAGES            SELECTOR
deployment.apps/mysql            1/1     1            1           83m   mysql        mysql:5.7         app=mysql
deployment.apps/pj-deploy-word   3/3     3            3           42s   wordpress    69875/doword:v1   app=pj-deploy-word

NAME                                        DESIRED   CURRENT   READY   AGE   CONTAINERS   IMAGES            SELECTOR
replicaset.apps/mysql-75b94558b             1         1         1       83m   mysql        mysql:5.7         app=mysql,pod-te
mplate-hash=75b94558b
replicaset.apps/pj-deploy-word-674977477c   3         3         3       42s   wordpress    69875/doword:v1   app=pj-deploy-wo
rd,pod-template-hash=674977477c
```



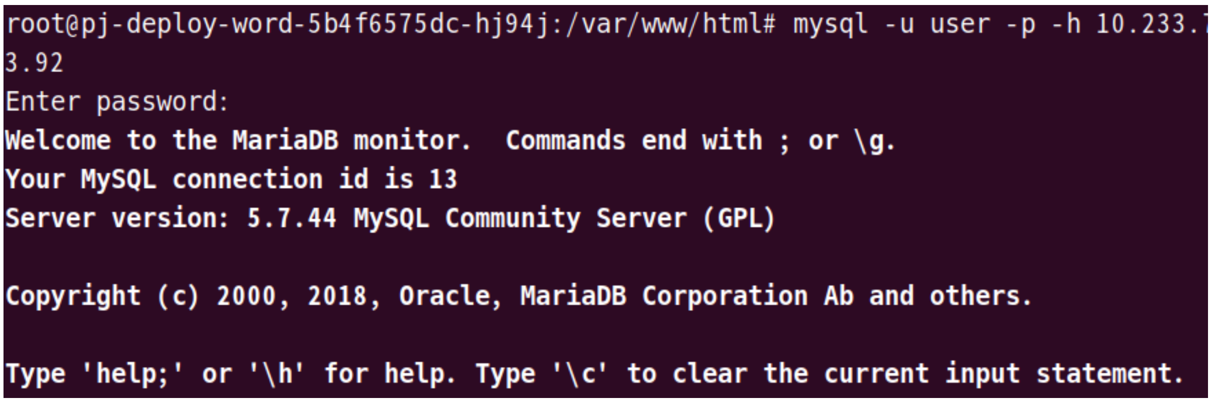
```
vagrant@kube-control1:~/kubepro$ kubectl exec -it mysql-75b94558b-g5bzd -- /bi
bash
bash-4.2# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

db 컨테이너에서 접속가능

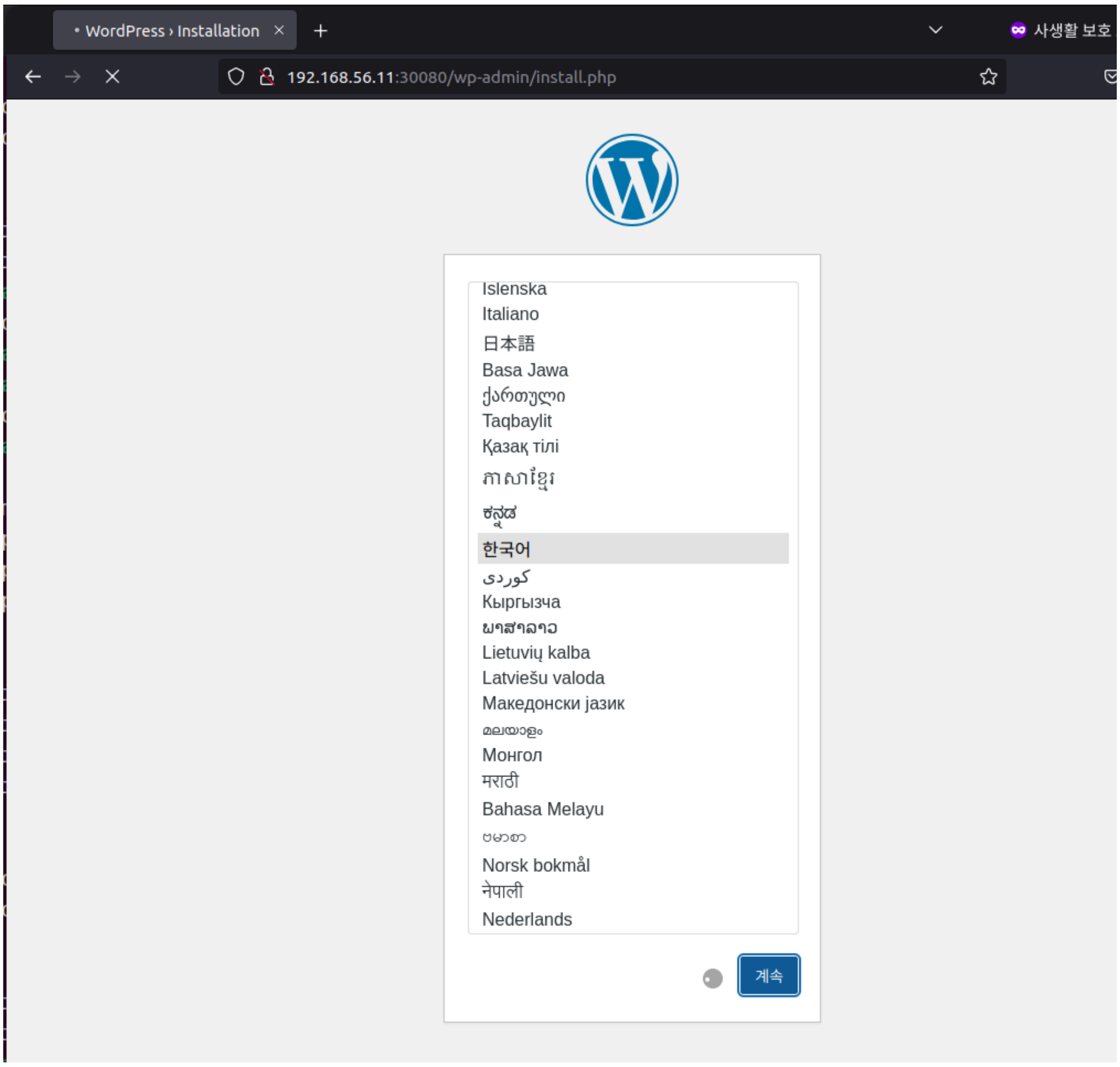


```
vagrant@kube-control1:~/kubepro$ mysql -u user -p -h 10.233.73.92
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

컨트롤 노드에서도 접속가능



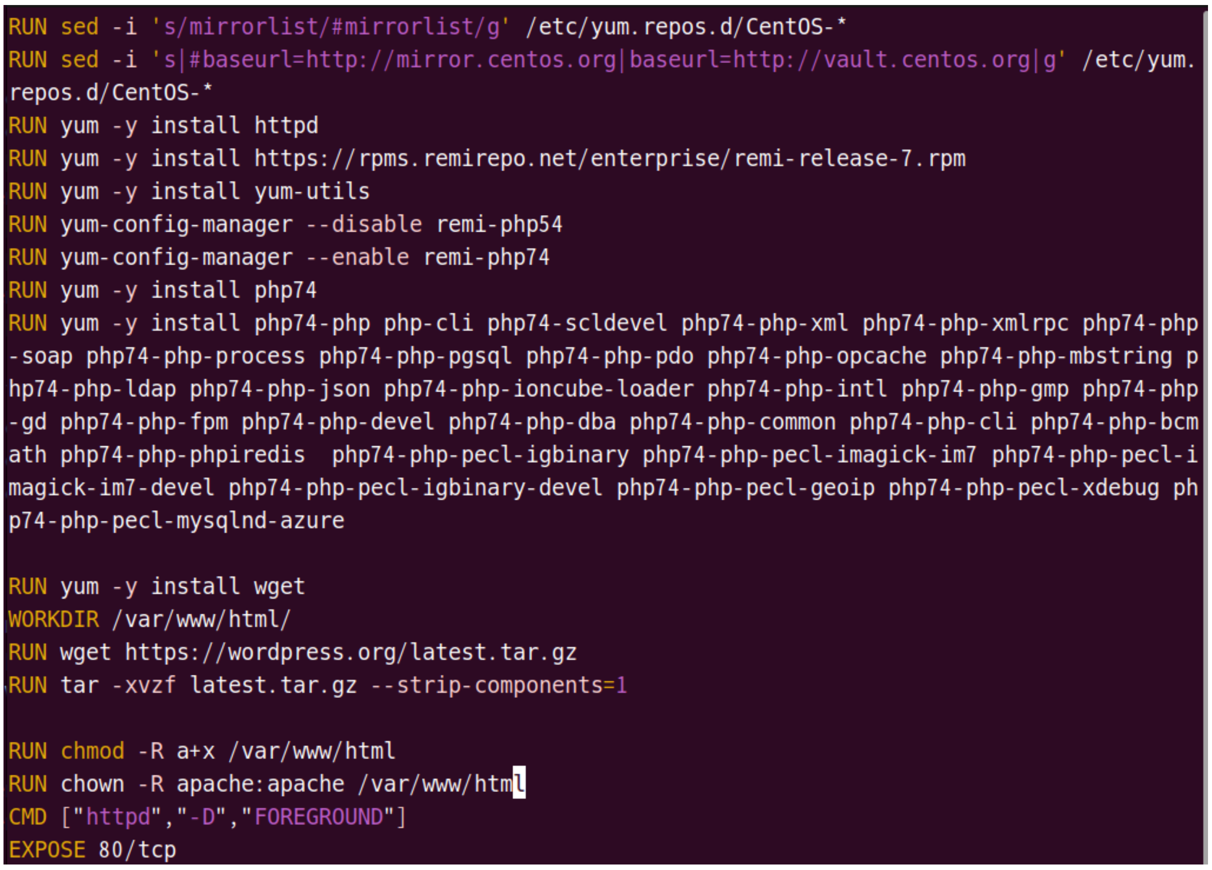
```
root@pj-deploy-word-5b4f6575dc-hj94j:/var/www/html# mysql -u user -p -h 10.233.
3.92
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```


• WordPress › Installation
사생활 보호
192.168.56.11:30080/wp-admin/install.php
Íslenska
Italiano
日本語
Basa Jawa
ქართული
Taqbaylit
Қазақ тілі
ភាសាខ្មែរ
ಕನ್ನಡ
한국어
كوردی
Кыргызча
ພາສາລາວ
Lietuvių kalba
Latviešu valoda
Македонски јазик
മലയാളം
Монгол
मराठी
Bahasa Melayu
ဗမာစာ
Norsk bokmål
नेपाली
Nederlands
계속

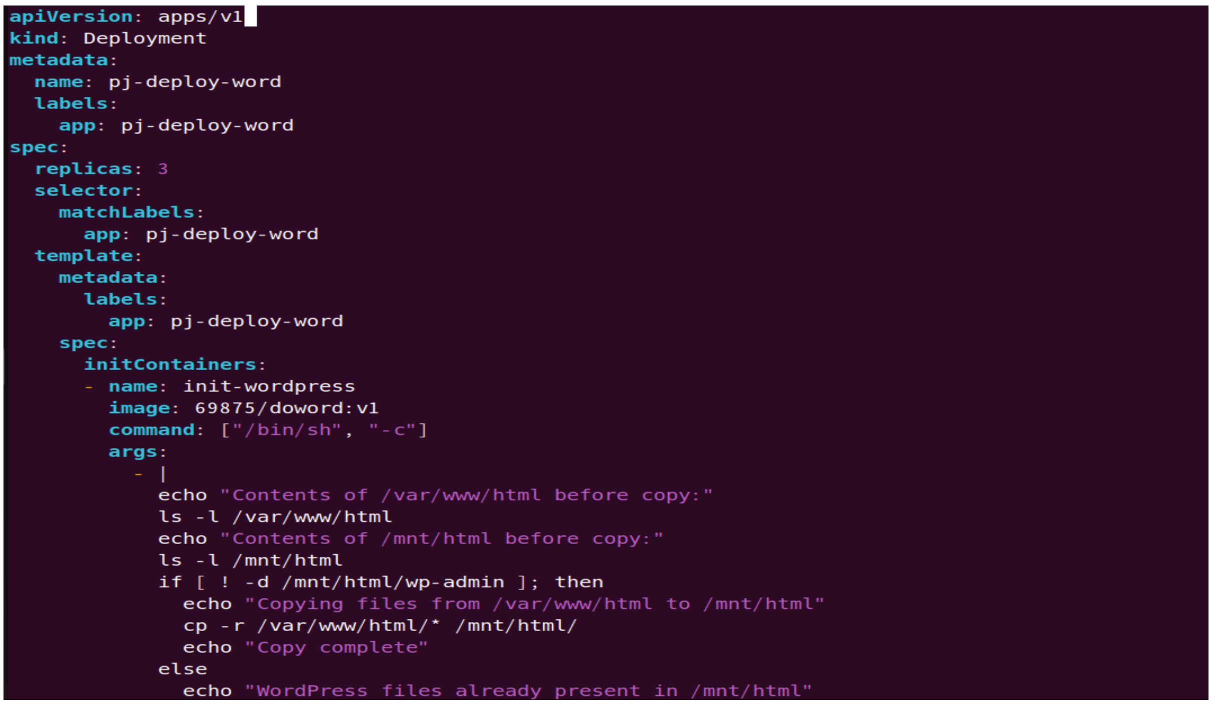
2.1 Wordpress image

wordpress image 를 직접 구성하기 위해 도커파일을 작성 후 빌드

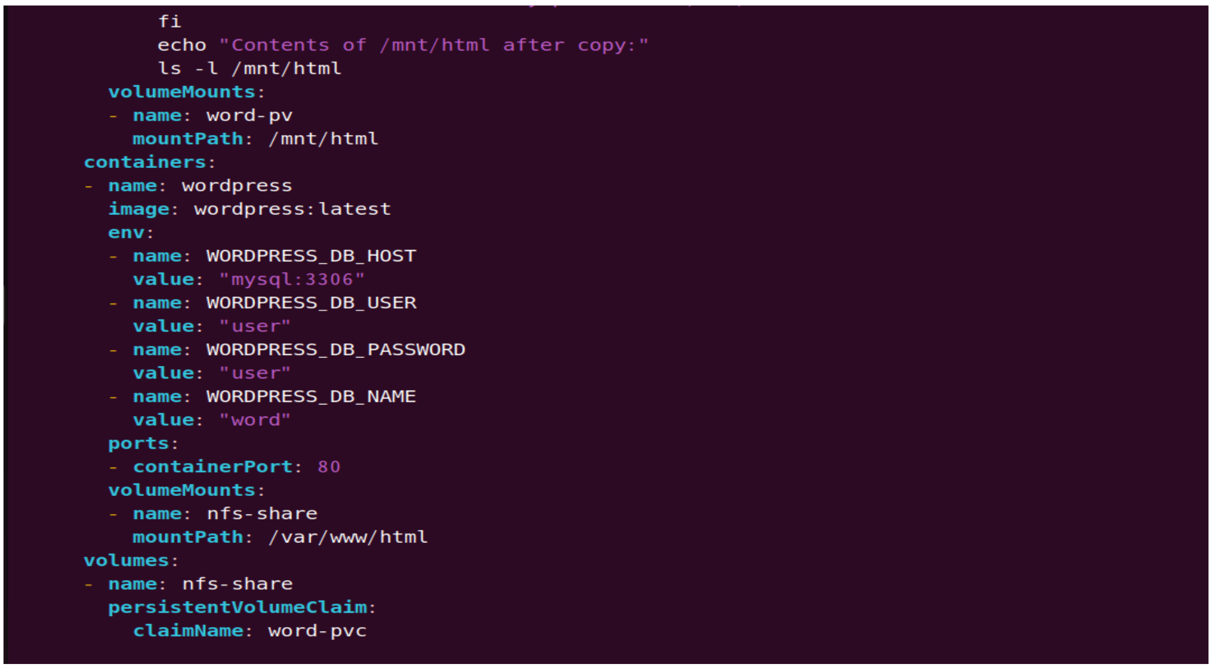
도커허브에 업로드 하였습니다.



```
RUN sed -i 's/mirrorlist/#mirrorlist/g' /etc/yum.repos.d/CentOS-*
RUN sed -i 's|#baseurl=http://mirror.centos.org|baseurl=http://vault.centos.org|g' /etc/yum.
repos.d/CentOS-*
RUN yum -y install httpd
RUN yum -y install https://rpms.remirepo.net/enterprise/remi-release-7.rpm
RUN yum -y install yum-utils
RUN yum-config-manager --disable remi-php54
RUN yum-config-manager --enable remi-php74
RUN yum -y install php74
RUN yum -y install php74-php php-cli php74-scldevel php74-php-xml php74-php-xmlrpc php74-php
-soap php74-php-process php74-php-pgsql php74-php-pdo php74-php-opcache php74-php-mbstring p
hp74-php-ldap php74-php-json php74-php-ioncube-loader php74-php-intl php74-php-gmp php74-php
-gd php74-php-fpm php74-php-devel php74-php-dba php74-php-common php74-php-cli php74-php-bcm
ath php74-php-phpiredis  php74-php-pecl-igbinary php74-php-pecl-imagick-im7 php74-php-pecl-i
magick-im7-devel php74-php-pecl-igbinary-devel php74-php-pecl-geoip php74-php-pecl-xdebug ph
p74-php-pecl-mysqlnd-azure

RUN yum -y install wget
WORKDIR /var/www/html/
RUN wget https://wordpress.org/latest.tar.gz
RUN tar -xvzf latest.tar.gz --strip-components=1

RUN chmod -R a+x /var/www/html
RUN chown -R apache:apache /var/www/html
CMD ["httpd","-D","FOREGROUND"]
EXPOSE 80/tcp
```



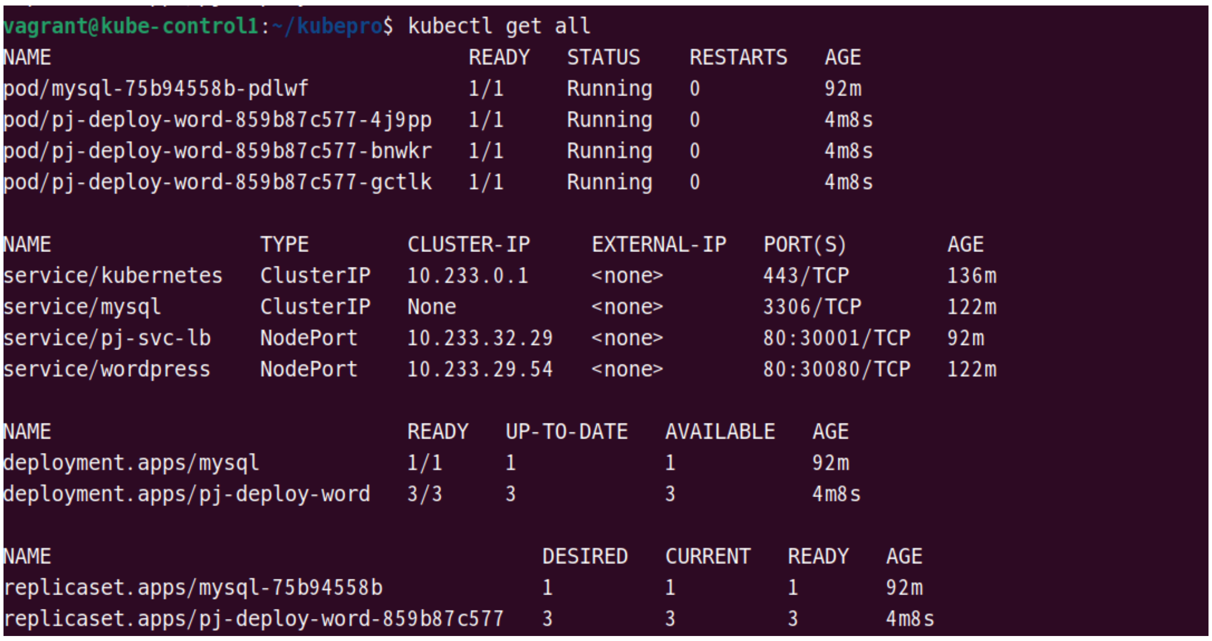
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: pj-deploy-word
  labels:
    app: pj-deploy-word
spec:
  replicas: 3
  selector:
    matchLabels:
      app: pj-deploy-word
  template:
    metadata:
      labels:
        app: pj-deploy-word
    spec:
      initContainers:
      - name: init-wordpress
        image: 69875/doword:v1
        command: ["/bin/sh", "-c"]
        args:
          - |
            echo "Contents of /var/www/html before copy:"
            ls -l /var/www/html
            echo "Contents of /mnt/html before copy:"
            ls -l /mnt/html
            if [ ! -d /mnt/html/wp-admin ]; then
              echo "Copying files from /var/www/html to /mnt/html"
              cp -r /var/www/html/* /mnt/html/
              echo "Copy complete"
            else
              echo "WordPress files already present in /mnt/html"
```



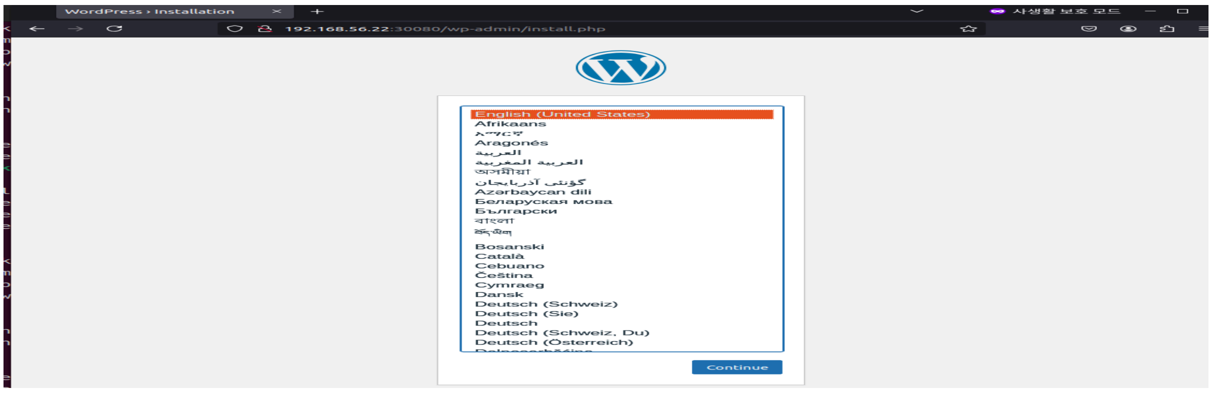
```
            fi
            echo "Contents of /mnt/html after copy:"
            ls -l /mnt/html
        volumeMounts:
        - name: word-pv
          mountPath: /mnt/html
      containers:
      - name: wordpress
        image: wordpress:latest
        env:
        - name: WORDPRESS_DB_HOST
          value: "mysql:3306"
        - name: WORDPRESS_DB_USER
          value: "user"
        - name: WORDPRESS_DB_PASSWORD
          value: "user"
        - name: WORDPRESS_DB_NAME
          value: "word"
        ports:
        - containerPort: 80
        volumeMounts:
        - name: nfs-share
          mountPath: /var/www/html
      volumes:
      - name: nfs-share
        persistentVolumeClaim:
          claimName: word-pvc
```

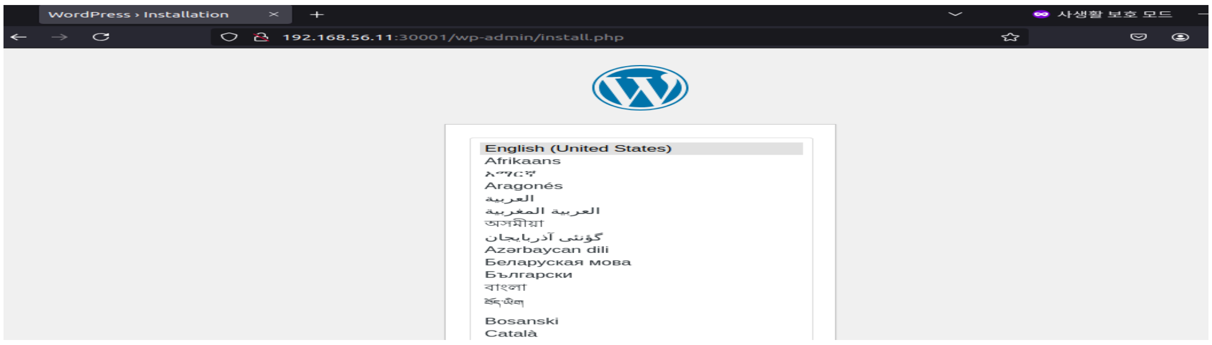
wordpress 파일을 만들어진 이미지를 사용해 동작하였습니다.

2.2 결과



```
vagrant@kube-control1:~/kubepro$ kubectl get all
NAME                                  READY   STATUS    RESTARTS   AGE
pod/mysql-75b94558b-pdlwf             1/1     Running   0          92m
pod/pj-deploy-word-859b87c577-4j9pp   1/1     Running   0          4m8s
pod/pj-deploy-word-859b87c577-bnwkr   1/1     Running   0          4m8s
pod/pj-deploy-word-859b87c577-gctlk   1/1     Running   0          4m8s

NAME                   TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
service/kubernetes     ClusterIP   10.233.0.1     <none>        443/TCP        136m
service/mysql          ClusterIP   None           <none>        3306/TCP       122m
service/pj-svc-lb      NodePort    10.233.32.29   <none>        80:30001/TCP   92m
service/wordpress      NodePort    10.233.29.54   <none>        80:30080/TCP   122m

NAME                             READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mysql            1/1     1            1           92m
deployment.apps/pj-deploy-word   3/3     3            3           4m8s

NAME                                        DESIRED   CURRENT   READY   AGE
replicaset.apps/mysql-75b94558b             1         1         1       92m
replicaset.apps/pj-deploy-word-859b87c577   3         3         3       4m8s
```


WordPress › Installation
사생활 보호 모드
192.168.56.22:30080/wp-admin/install.php
English (United States)
Afrikaans
አማርኛ
Aragonés
العربية
العربية المغربية
অসমীয়া
گؤنئی آذربایجان
Azərbaycan dili
Беларуская мова
Български
বাংলা
བོད་ཡིག
Bosanski
Català
Cebuano
Čeština
Cymraeg
Dansk
Deutsch (Schweiz)
Deutsch (Sie)
Deutsch
Deutsch (Schweiz, Du)
Deutsch (Österreich)
Continue


WordPress › Installation
사생활 보호 모드
192.168.56.11:30001/wp-admin/install.php
English (United States)
Afrikaans
አማርኛ
Aragonés
العربية
العربية المغربية
অসমীয়া
گؤنئی آذربایجان
Azərbaycan dili
Беларуская мова
Български
বাংলা
བོད་ཡིག
Bosanski
Català