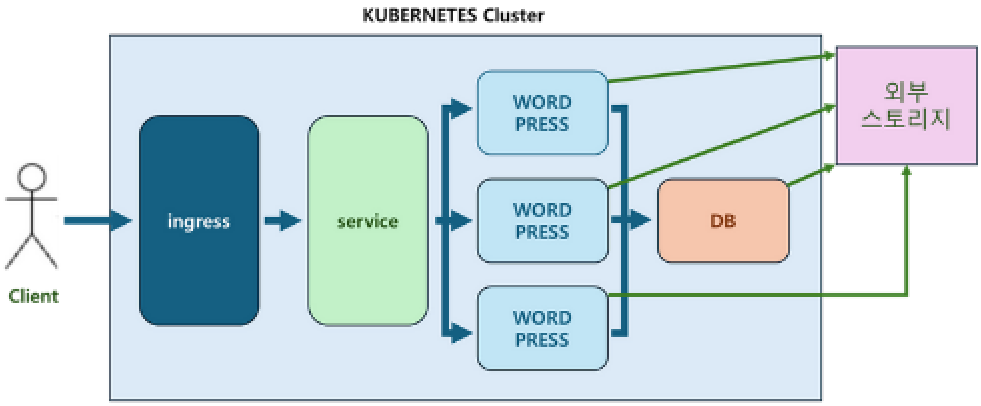
쿠버네티스 미니프로젝트

장민우

개요

kubernetes 를 활용한 wordpress 구성

구성 아키텍쳐



기본 구성

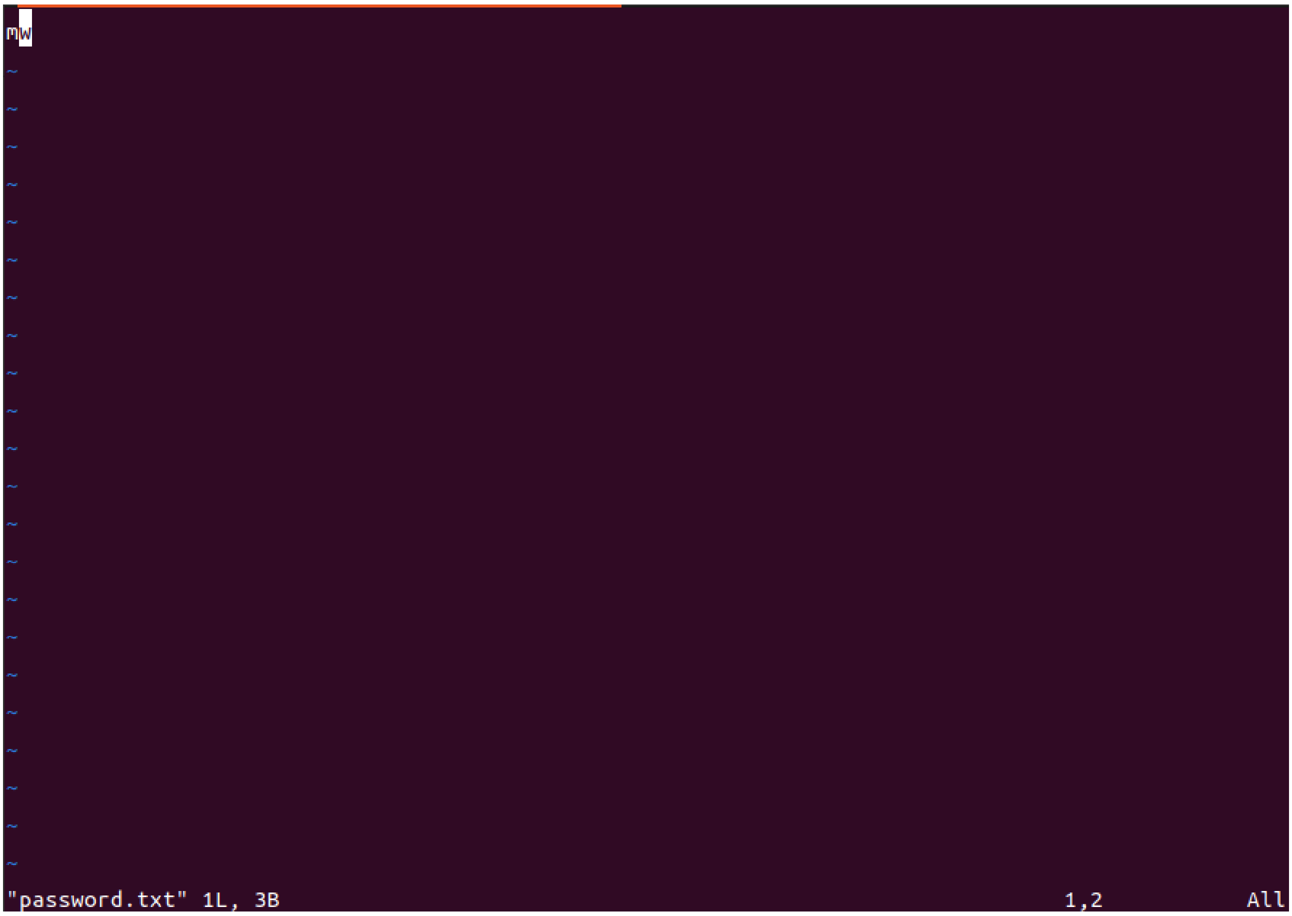
- image
  - web - word press
  - db - mysql or mariadb
- volume
  - 외부 스토리지 - nfs
  - control-plan 을 nfs 서버로 사용 가능
- pod
  - 컨트롤러를 이용한 생성
  - 사용 컨트롤러는 자유

추가 구성 사항

- wordpress image 제작
  - Dockerfile 활용
- DB 이중화
  - statefulset 활용

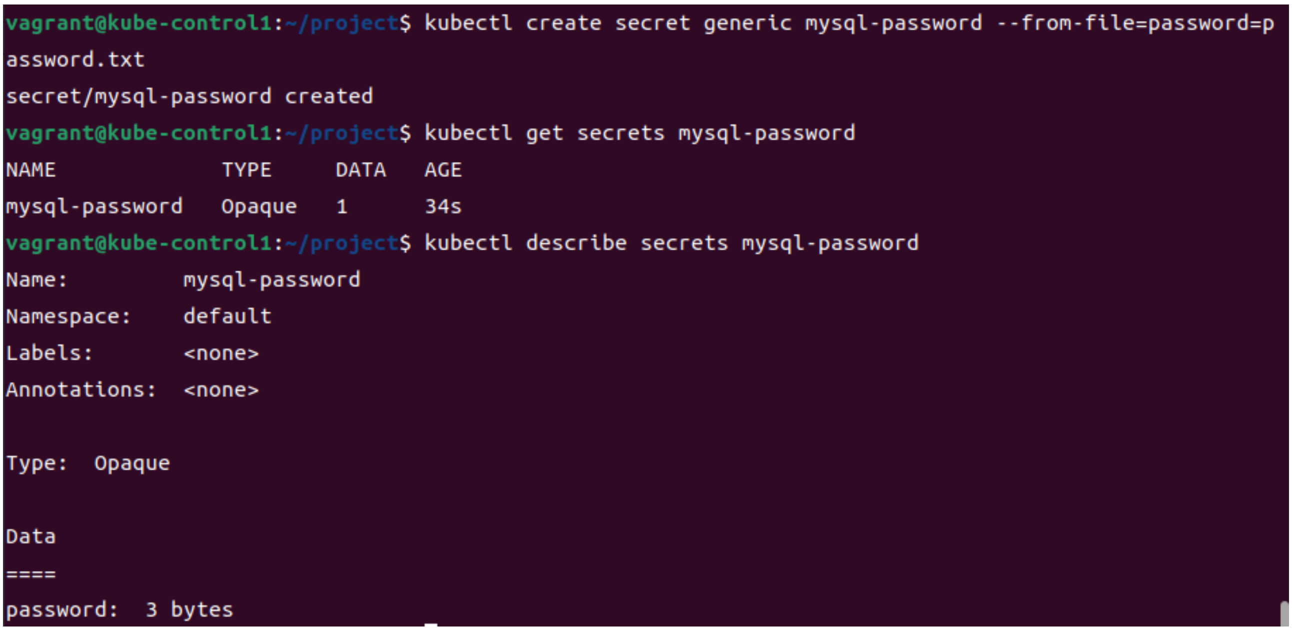
1. MYSQL ROOT 비밀번호로 쓰일 비밀번호 파일 생성.

```
#vim password.txt
```



```
mw
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"password.txt" 1L, 3B                                                    1,2           All
```

2. MYSQL 비밀번호 생성.

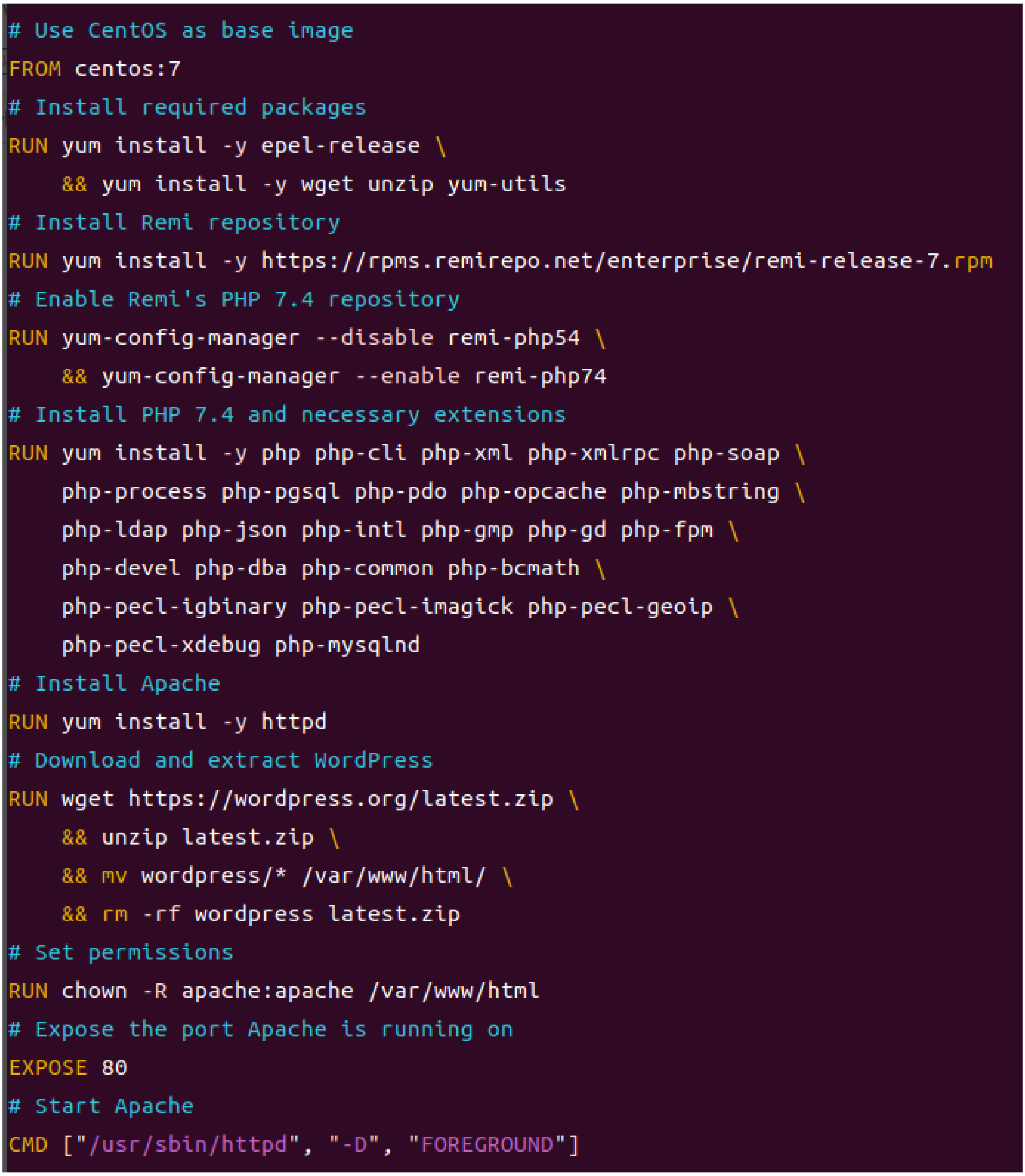


```
vagrant@kube-control1:~/project$ kubectl create secret generic mysql-password --from-file=password=p
assword.txt
secret/mysql-password created
vagrant@kube-control1:~/project$ kubectl get secrets mysql-password
NAME             TYPE     DATA   AGE
mysql-password   Opaque   1      34s
vagrant@kube-control1:~/project$ kubectl describe secrets mysql-password
Name:         mysql-password
Namespace:    default
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
password:  3 bytes
```

3. 도커가 설치된 환경에서 도커파일 생성.

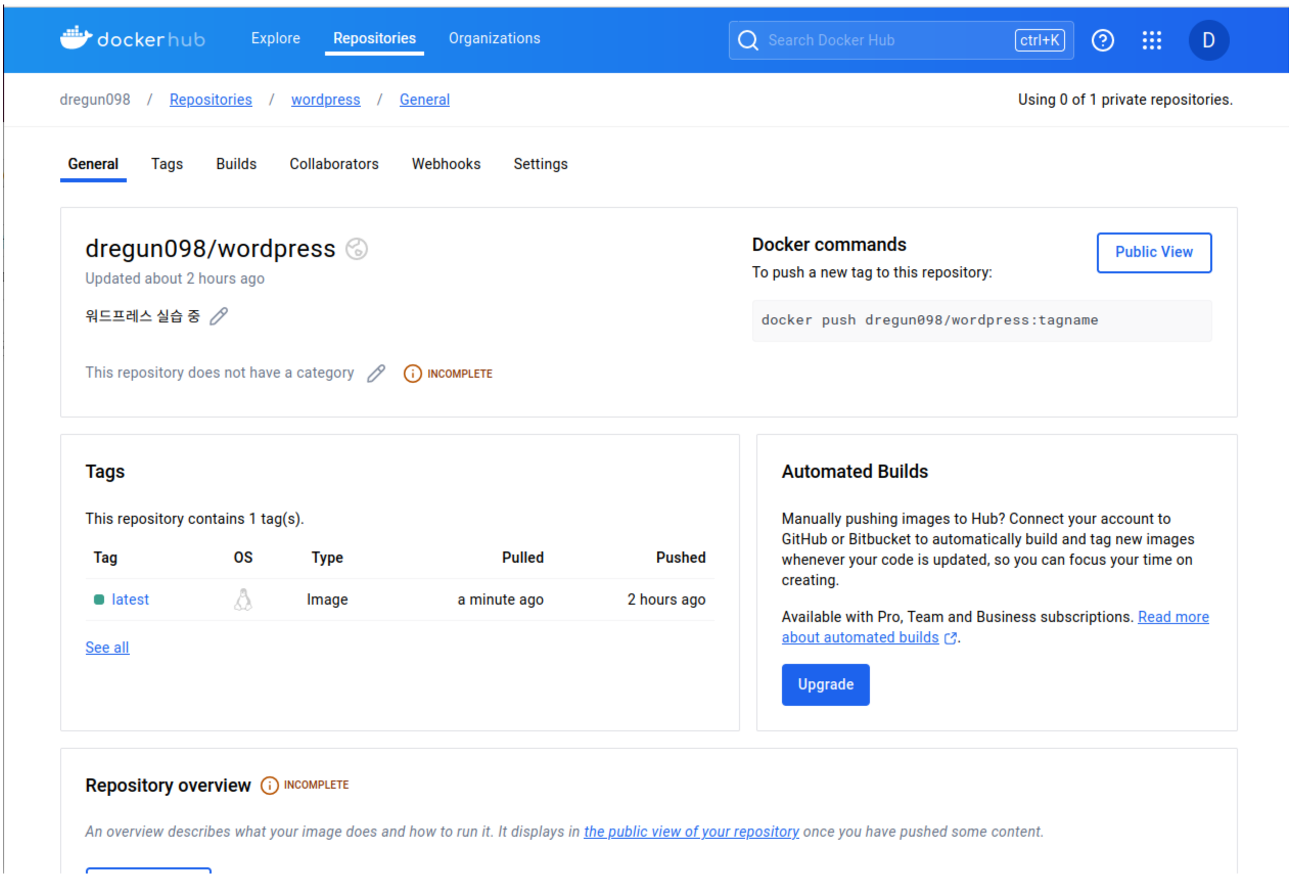


```
# Use CentOS as base image
FROM centos:7
# Install required packages
RUN yum install -y epel-release \
    && yum install -y wget unzip yum-utils
# Install Remi repository
RUN yum install -y https://rpms.remirepo.net/enterprise/remi-release-7.rpm
# Enable Remi's PHP 7.4 repository
RUN yum-config-manager --disable remi-php54 \
    && yum-config-manager --enable remi-php74
# Install PHP 7.4 and necessary extensions
RUN yum install -y php php-cli php-xml php-xmlrpc php-soap \
    php-process php-pgsql php-pdo php-opcache php-mbstring \
    php-ldap php-json php-intl php-gmp php-gd php-fpm \
    php-devel php-dba php-common php-bcmath \
    php-pecl-igbinary php-pecl-imagick php-pecl-geoip \
    php-pecl-xdebug php-mysqlnd
# Install Apache
RUN yum install -y httpd
# Download and extract WordPress
RUN wget https://wordpress.org/latest.zip \
    && unzip latest.zip \
    && mv wordpress/* /var/www/html/ \
    && rm -rf wordpress latest.zip
# Set permissions
RUN chown -R apache:apache /var/www/html
# Expose the port Apache is running on
EXPOSE 80
# Start Apache
CMD ["/usr/sbin/httpd", "-D", "FOREGROUND"]
```

4. 도커파일을 이용해 도커 이미지 올리기.

```
#sudo docker build -t wordpress .
#sudo docker image tag wordpress dregun098/wordpress
#sudo docker push dregun098/wordpress
```

5. 생성된 도커이미지 확인.



dregun098 / Repositories / wordpress / General

Using 0 of 1 private repositories.

dregun098/wordpress

Updated about 2 hours ago

워드프레스 실습 중

This repository does not have a category INCOMPLETE

Docker commands

To push a new tag to this repository:

```
docker push dregun098/wordpress:tagname
```

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	a minute ago	2 hours ago

See all

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

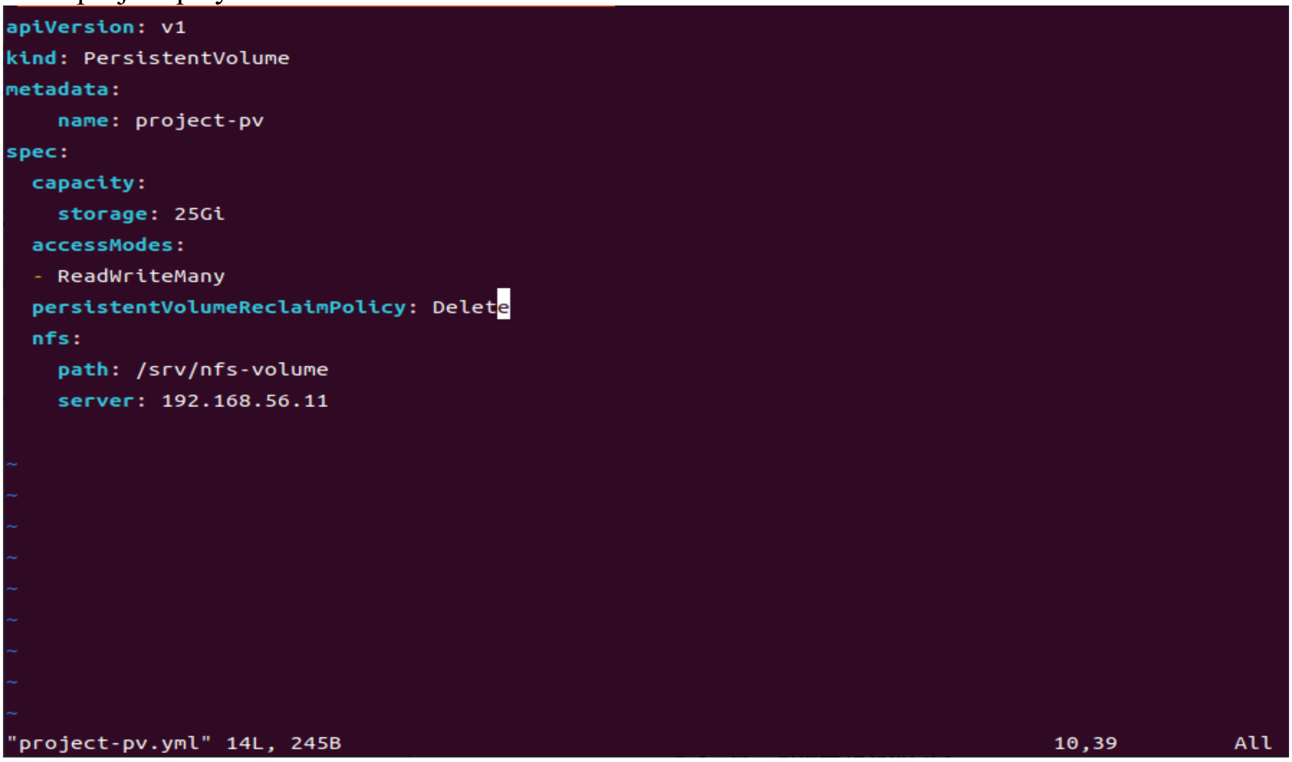
Available with Pro, Team and Business subscriptions. Read more about automated builds.

Repository overview INCOMPLETE

An overview describes what your image does and how to run it. It displays in the public view of your repository once you have pushed some content.

6. PV 생성.

#vim project-pv.yml

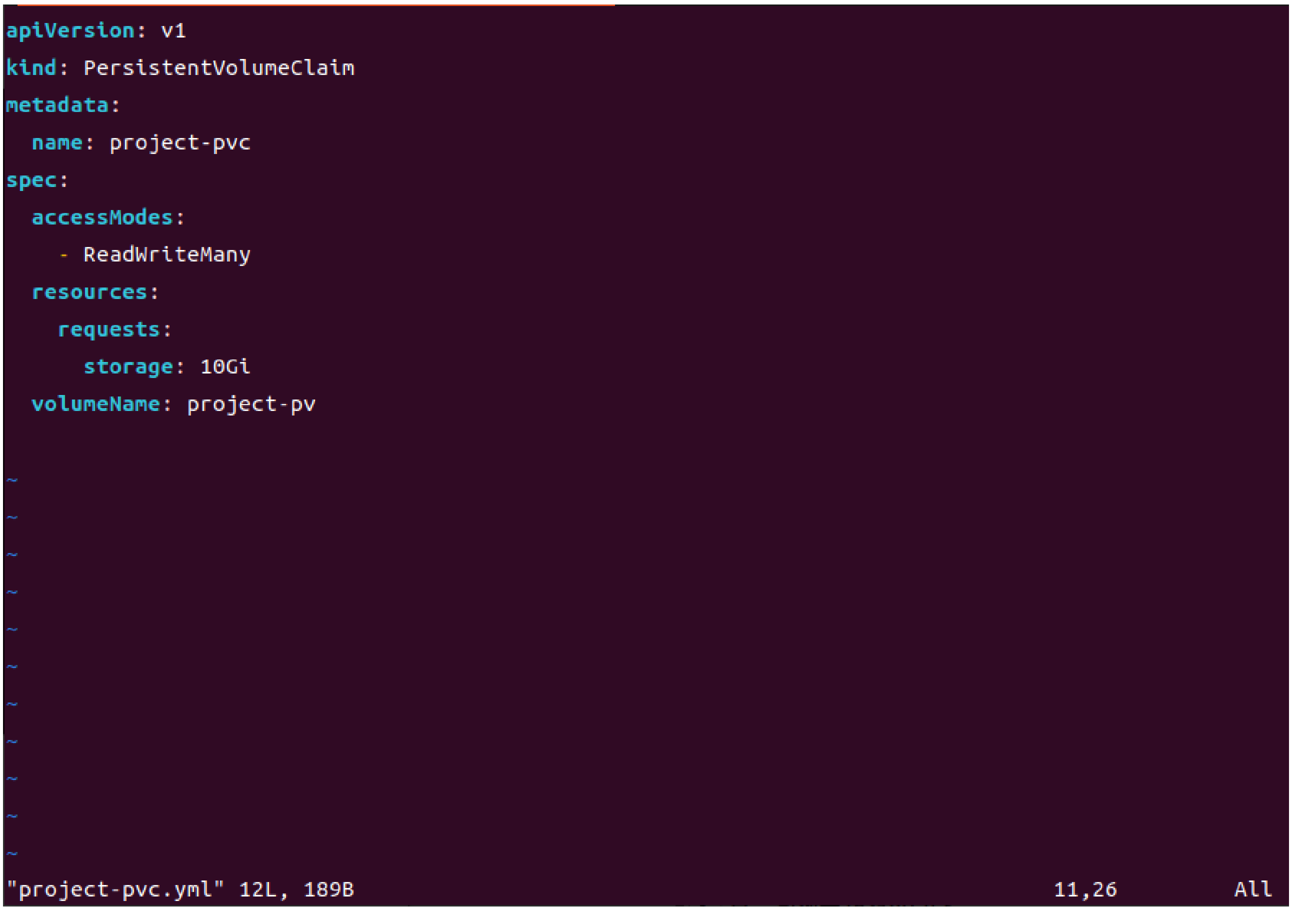


```
apiVersion: v1
kind: PersistentVolume
metadata:
    name: project-pv
spec:
  capacity:
    storage: 25Gi
  accessModes:
  - ReadWriteMany
  persistentVolumeReclaimPolicy: Delete
  nfs:
    path: /srv/nfs-volume
    server: 192.168.56.11

~
~
~
~
~
~
~
~
~
"project-pv.yml" 14L, 245B                                      10,39         All
```

7. PVC 생성.

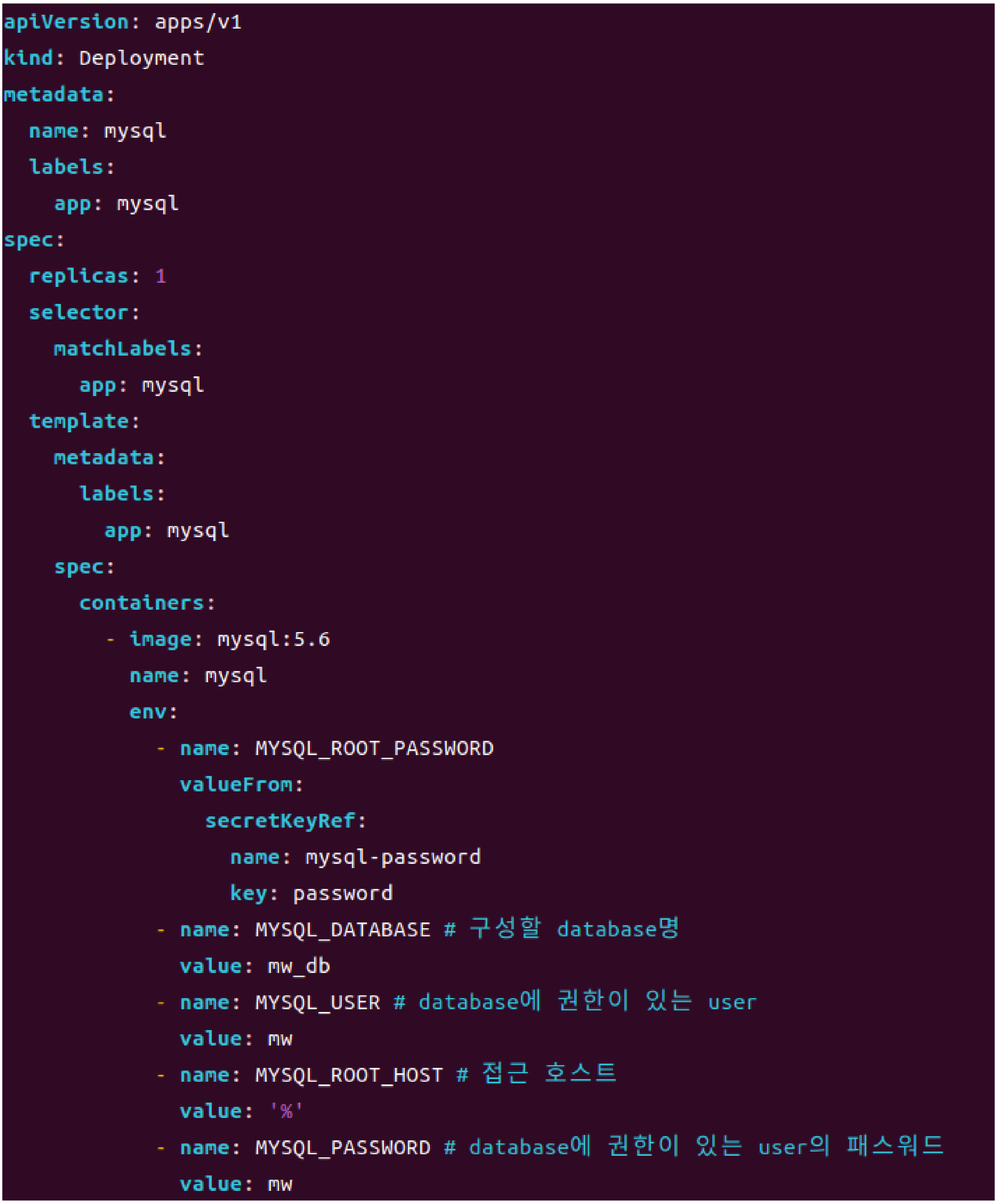
#vim project-pvc.yml



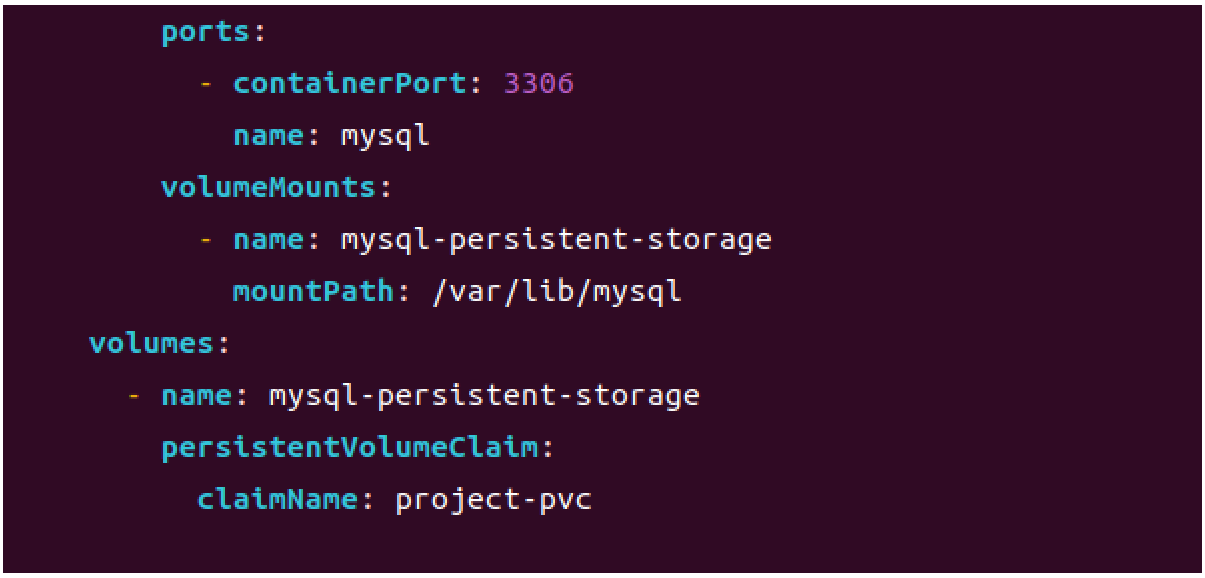
```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: project-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
  volumeName: project-pv

~
~
~
~
~
~
~
~
~
~
~
"project-pvc.yml" 12L, 189B                                             11,26         All
```

8. mysql Deployment 생성.
#vim mysql.yml



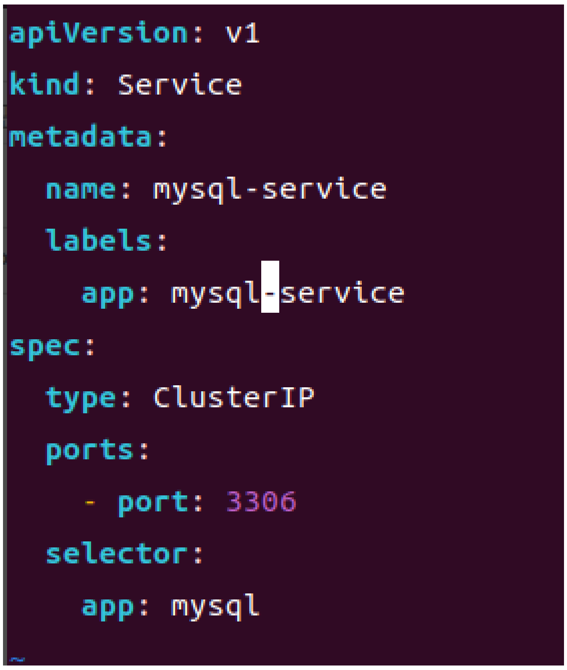
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql
  labels:
    app: mysql
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - image: mysql:5.6
          name: mysql
          env:
            - name: MYSQL_ROOT_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: mysql-password
                  key: password
            - name: MYSQL_DATABASE # 구성할 database명
              value: mw_db
            - name: MYSQL_USER # database에 권한이 있는 user
              value: mw
            - name: MYSQL_ROOT_HOST # 접근 호스트
              value: '%'
            - name: MYSQL_PASSWORD # database에 권한이 있는 user의 패스워드
              value: mw
```



```
        ports:
          - containerPort: 3306
            name: mysql
        volumeMounts:
          - name: mysql-persistent-storage
            mountPath: /var/lib/mysql
    volumes:
      - name: mysql-persistent-storage
        persistentVolumeClaim:
          claimName: project-pvc
```

9. mysql 서비스 생성.

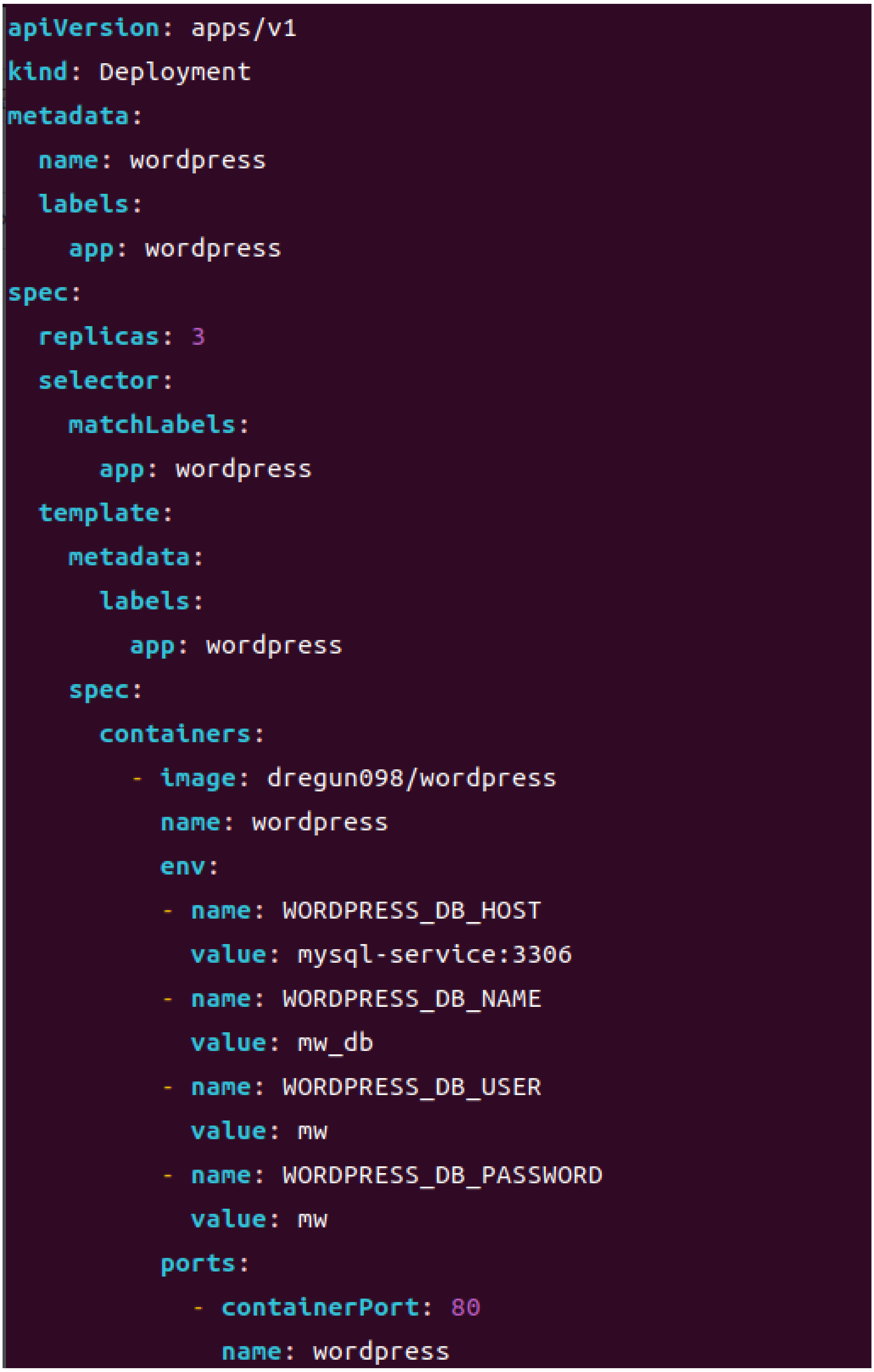
#vim mysql-service.yml



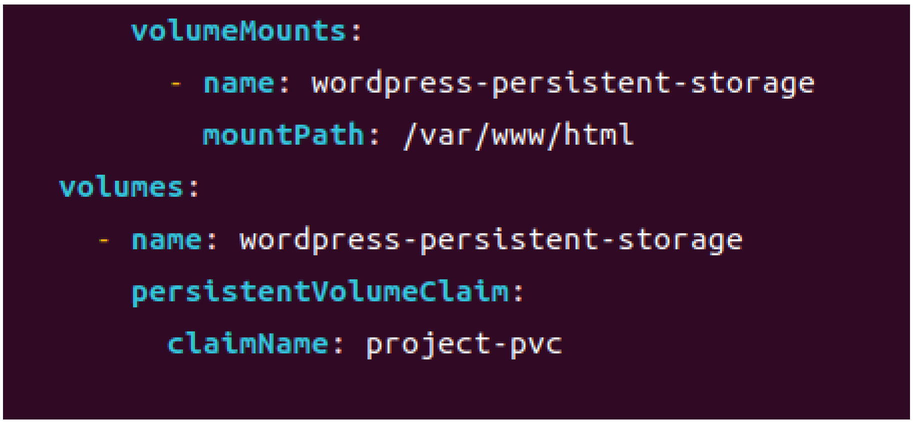
```
apiVersion: v1
kind: Service
metadata:
  name: mysql-service
  labels:
    app: mysql-service
spec:
  type: ClusterIP
  ports:
    - port: 3306
  selector:
    app: mysql
```

10. wordpress Deployment 생성.

#vim wordpress.yml



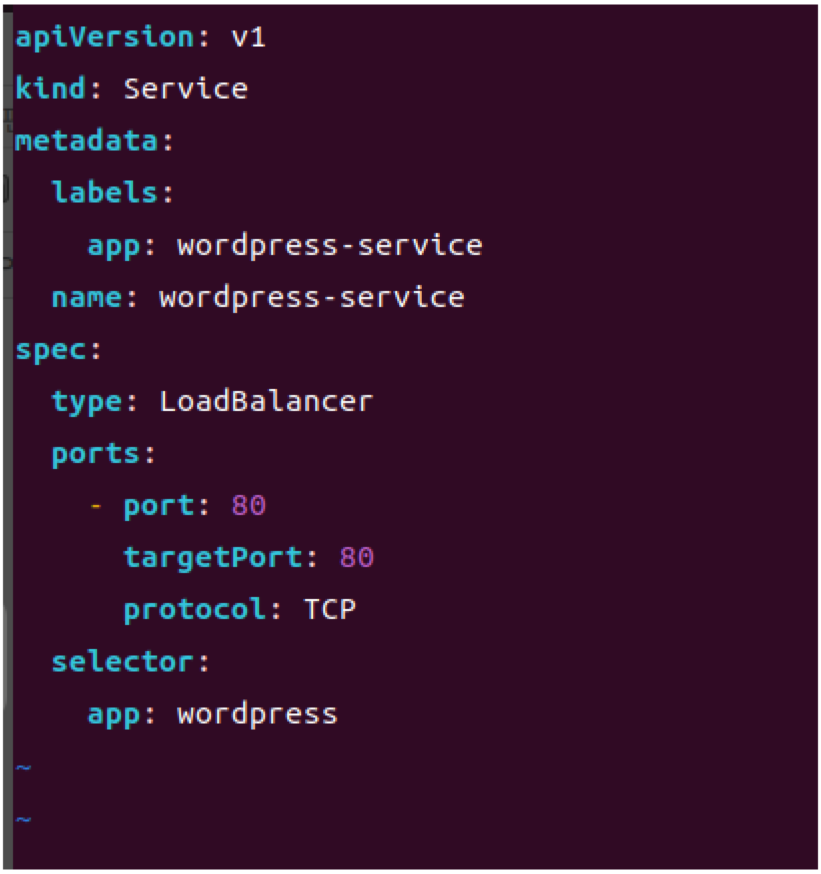
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wordpress
  labels:
    app: wordpress
spec:
  replicas: 3
  selector:
    matchLabels:
      app: wordpress
  template:
    metadata:
      labels:
        app: wordpress
    spec:
      containers:
        - image: dregun098/wordpress
          name: wordpress
          env:
          - name: WORDPRESS_DB_HOST
            value: mysql-service:3306
          - name: WORDPRESS_DB_NAME
            value: mw_db
          - name: WORDPRESS_DB_USER
            value: mw
          - name: WORDPRESS_DB_PASSWORD
            value: mw
          ports:
            - containerPort: 80
              name: wordpress
```



```
      volumeMounts:
        - name: wordpress-persistent-storage
          mountPath: /var/www/html
  volumes:
    - name: wordpress-persistent-storage
      persistentVolumeClaim:
        claimName: project-pvc
```

11. wordpress-service 생성.

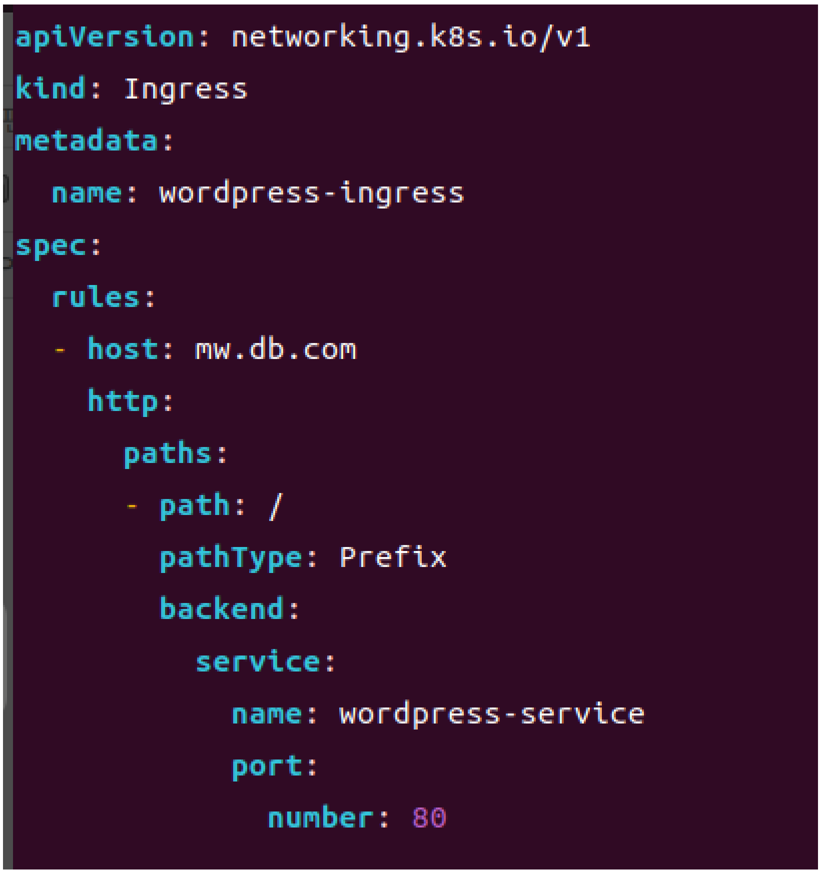
#vim wordpress-service.yml



```
apiVersion: v1
kind: Service
metadata:
  labels:
    app: wordpress-service
  name: wordpress-service
spec:
  type: LoadBalancer
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  selector:
    app: wordpress
~
~
```

12. wordpress-ingress 생성.

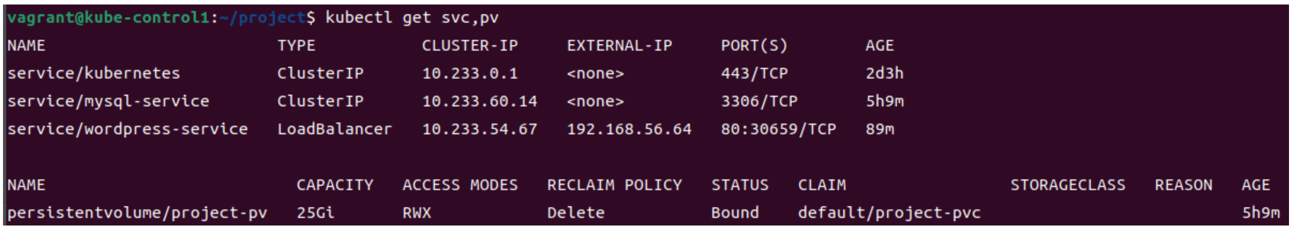
#vim wordpress-ingress.yml



```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: wordpress-ingress
spec:
  rules:
  - host: mw.db.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: wordpress-service
            port:
              number: 80
```

13. 서비스 확인.

#kubectl get svc,pv

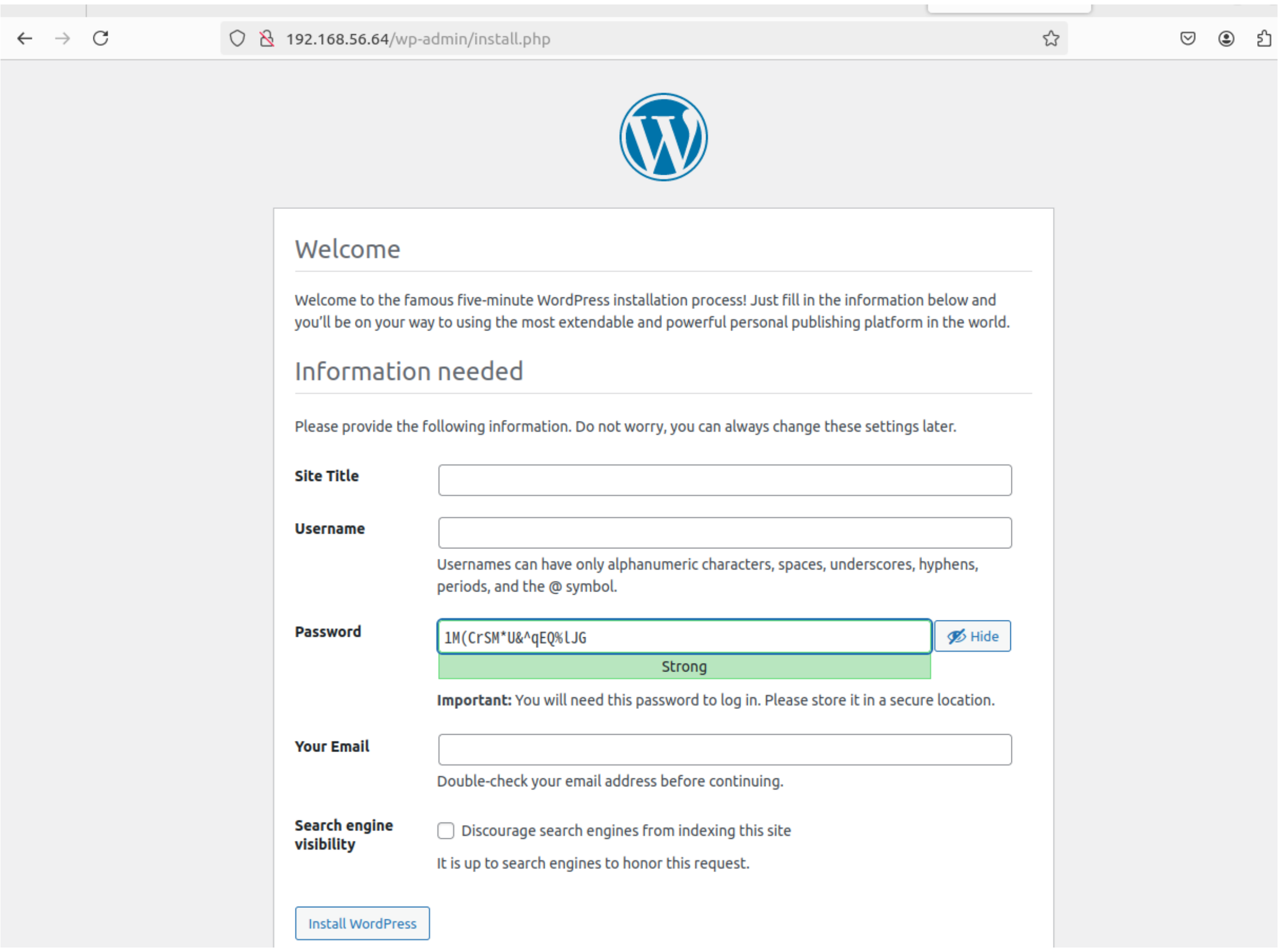


```
vagrant@kube-control1:~/project$ kubectl get svc,pv
NAME                       TYPE           CLUSTER-IP     EXTERNAL-IP    PORT(S)        AGE
service/kubernetes         ClusterIP      10.233.0.1     <none>         443/TCP        2d3h
service/mysql-service      ClusterIP      10.233.60.14   <none>         3306/TCP       5h9m
service/wordpress-service  LoadBalancer   10.233.54.67   192.168.56.64  80:30659/TCP   89m

NAME                         CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS   REASON   AGE
persistentvolume/project-pv  25Gi       RWX            Delete           Bound    default/project-pvc                            5h9m
```

14. 접속 확인.

192.168.56.64 접속 화면.



192.168.56.64/wp-admin/install.php

Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

Site Title

Username

Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password 1M(CrSM*U&^qEQ%lJG

Strong

Important: You will need this password to log in. Please store it in a secure location.

Your Email

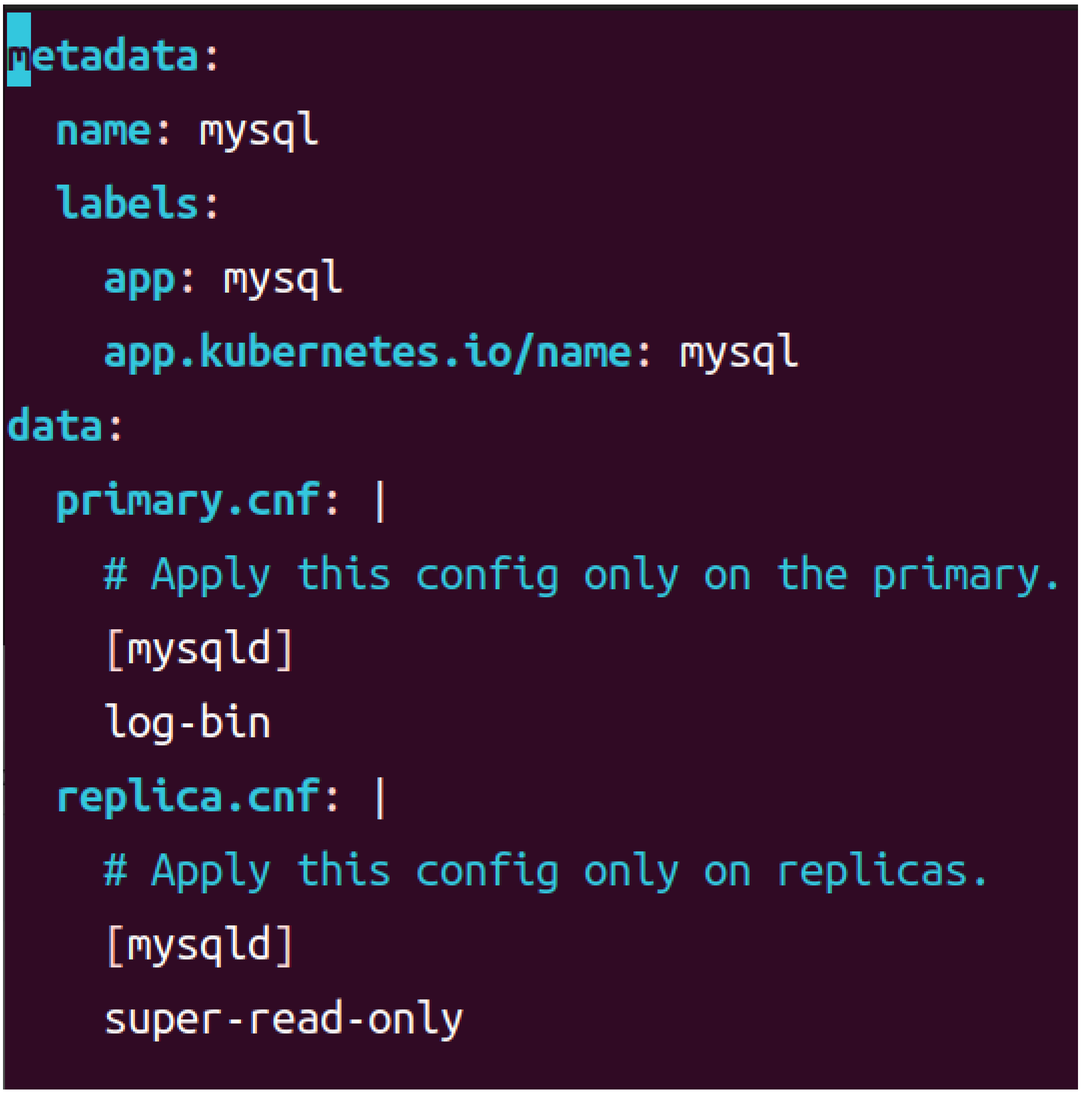
Double-check your email address before continuing.

Search engine visibility □ Discourage search engines from indexing this site

It is up to search engines to honor this request.

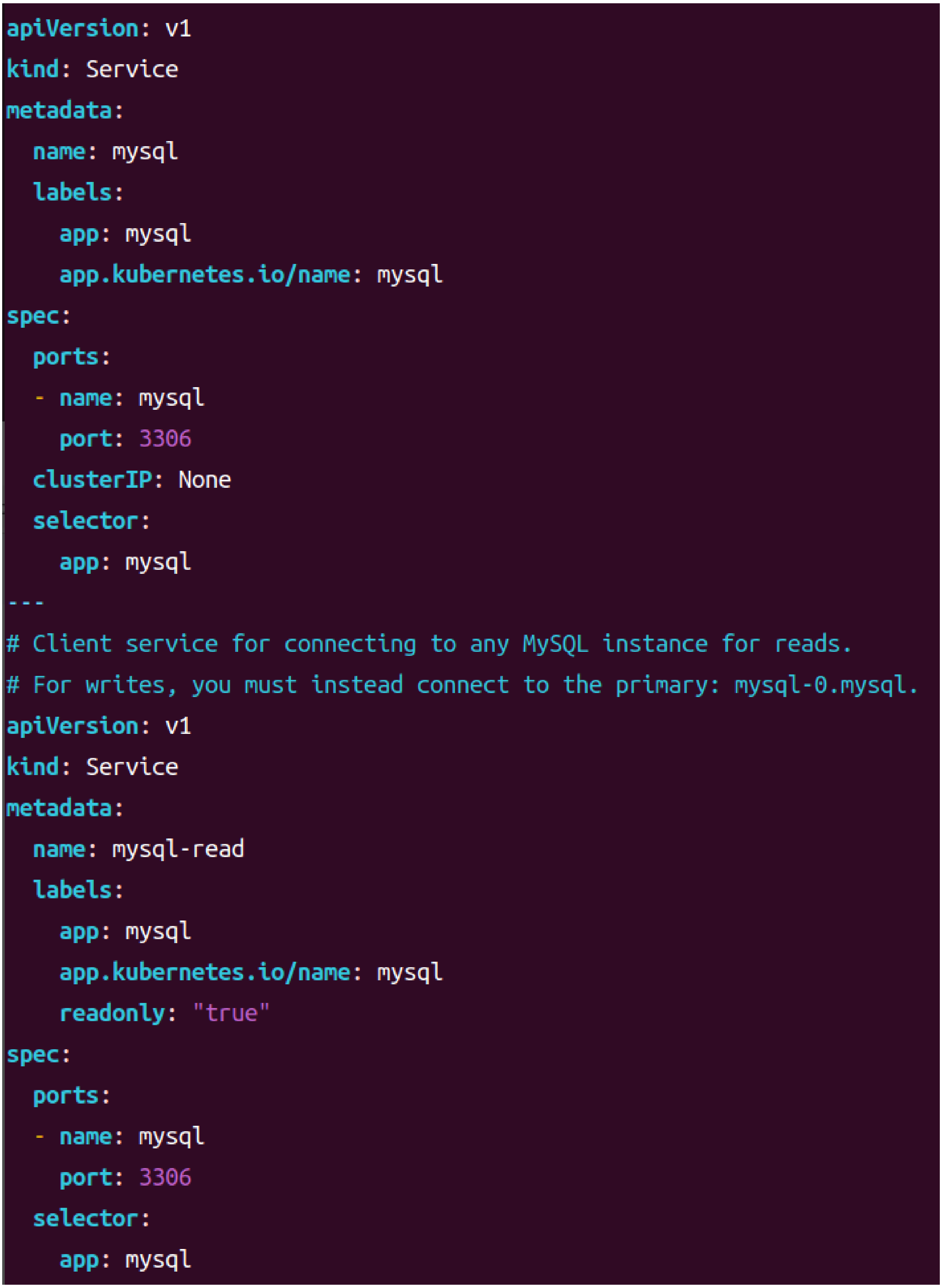
15. DB 이중화를 위한 config 파일 생성.

```
#vim mydb-cm-mysql.yml
```



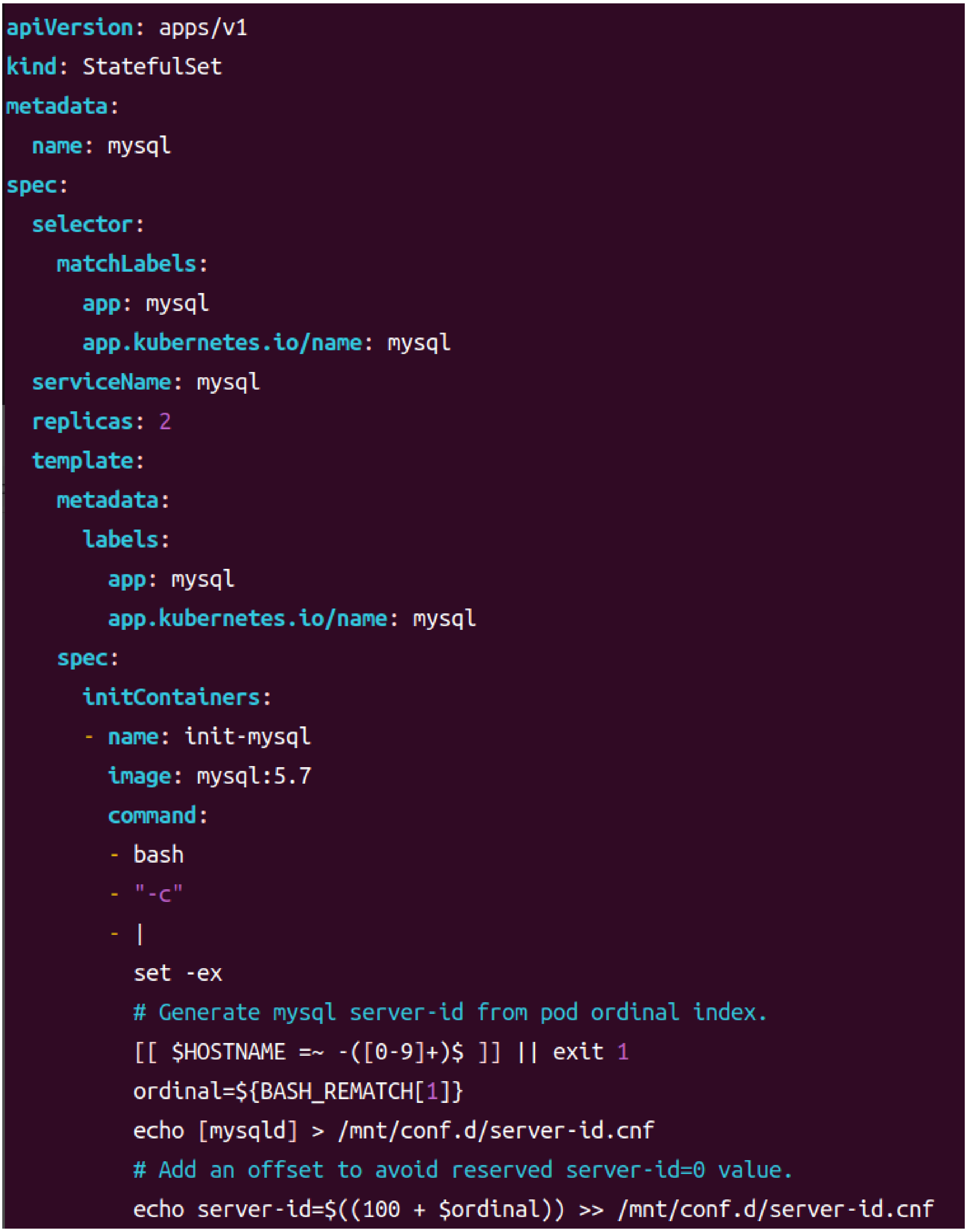
```yaml
metadata:
  name: mysql
  labels:
    app: mysql
    app.kubernetes.io/name: mysql
data:
  primary.cnf: |
    # Apply this config only on the primary.
    [mysqld]
    log-bin
  replica.cnf: |
    # Apply this config only on replicas.
    [mysqld]
    super-read-only
```

16. DB 이중화를 위한 master, slave 서비스 생성.

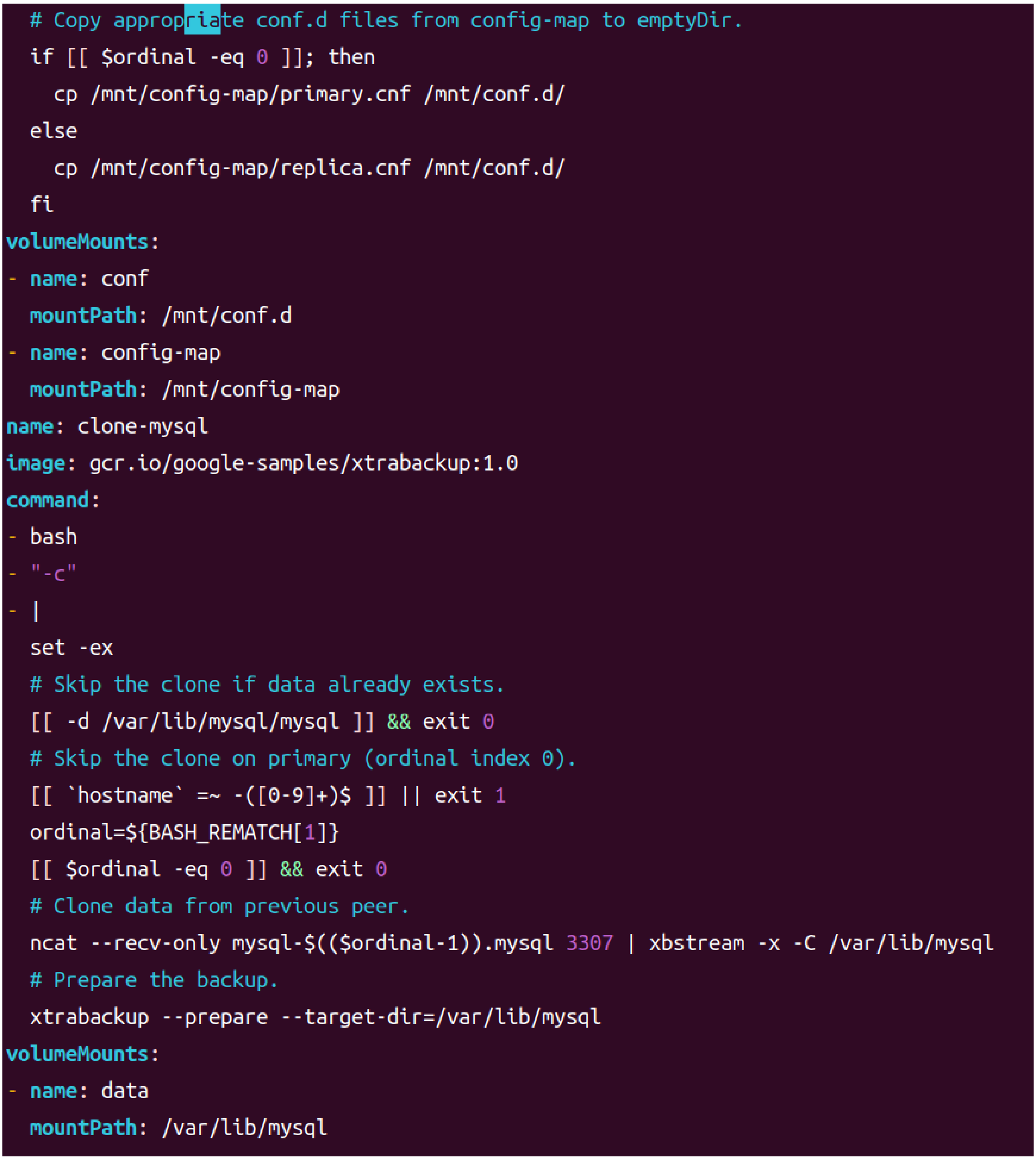


```
apiVersion: v1
kind: Service
metadata:
  name: mysql
  labels:
    app: mysql
    app.kubernetes.io/name: mysql
spec:
  ports:
  - name: mysql
    port: 3306
  clusterIP: None
  selector:
    app: mysql
---
# Client service for connecting to any MySQL instance for reads.
# For writes, you must instead connect to the primary: mysql-0.mysql.
apiVersion: v1
kind: Service
metadata:
  name: mysql-read
  labels:
    app: mysql
    app.kubernetes.io/name: mysql
    readonly: "true"
spec:
  ports:
  - name: mysql
    port: 3306
  selector:
    app: mysql
```

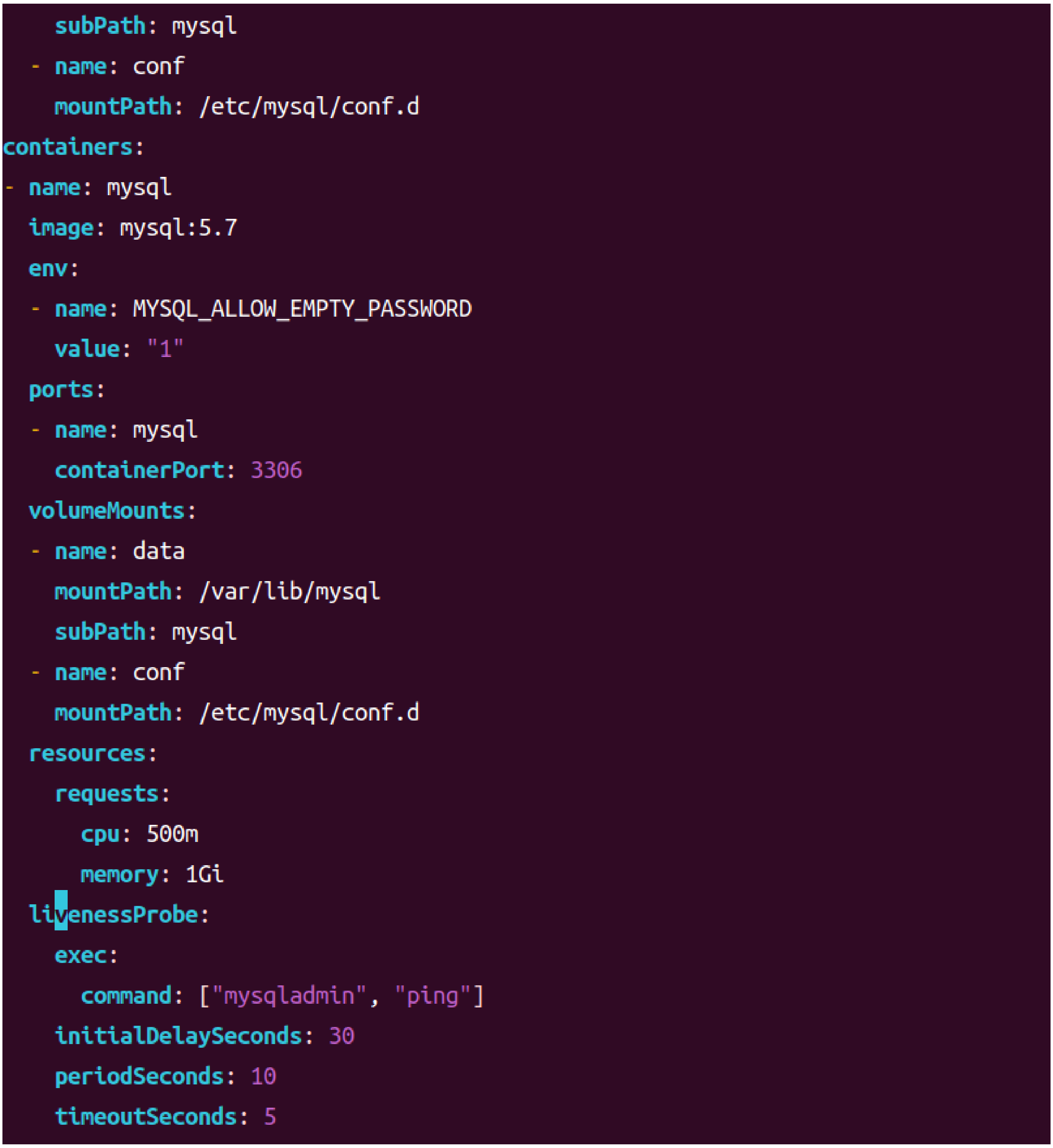
17. DB 이중화를 위한 스테이트풀셋 생성.



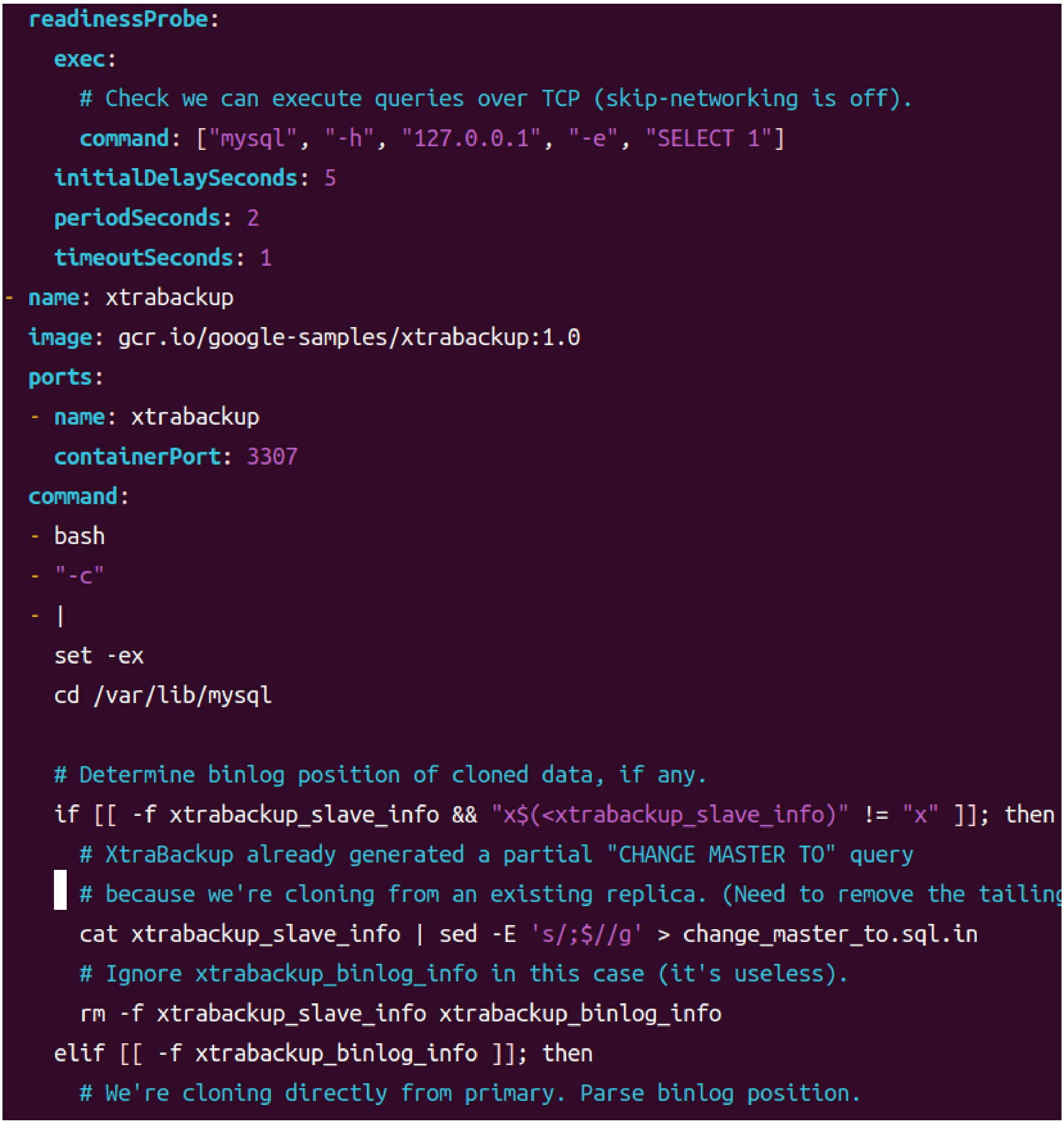
```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  selector:
    matchLabels:
      app: mysql
      app.kubernetes.io/name: mysql
  serviceName: mysql
  replicas: 2
  template:
    metadata:
      labels:
        app: mysql
        app.kubernetes.io/name: mysql
    spec:
      initContainers:
      - name: init-mysql
        image: mysql:5.7
        command:
        - bash
        - "-c"
        - |
          set -ex
          # Generate mysql server-id from pod ordinal index.
          [[ $HOSTNAME =~ -([0-9]+)$ ]] || exit 1
          ordinal=${BASH_REMATCH[1]}
          echo [mysqld] > /mnt/conf.d/server-id.cnf
          # Add an offset to avoid reserved server-id=0 value.
          echo server-id=$((100 + $ordinal)) >> /mnt/conf.d/server-id.cnf
```



```
  # Copy appropriate conf.d files from config-map to emptyDir.
  if [[ $ordinal -eq 0 ]]; then
    cp /mnt/config-map/primary.cnf /mnt/conf.d/
  else
    cp /mnt/config-map/replica.cnf /mnt/conf.d/
  fi
volumeMounts:
- name: conf
  mountPath: /mnt/conf.d
- name: config-map
  mountPath: /mnt/config-map
name: clone-mysql
image: gcr.io/google-samples/xtrabackup:1.0
command:
- bash
- "-c"
- |
  set -ex
  # Skip the clone if data already exists.
  [[ -d /var/lib/mysql/mysql ]] && exit 0
  # Skip the clone on primary (ordinal index 0).
  [[ `hostname` =~ -([0-9]+)$ ]] || exit 1
  ordinal=${BASH_REMATCH[1]}
  [[ $ordinal -eq 0 ]] && exit 0
  # Clone data from previous peer.
  ncat --recv-only mysql-$(($ordinal-1)).mysql 3307 | xbstream -x -C /var/lib/mysql
  # Prepare the backup.
  xtrabackup --prepare --target-dir=/var/lib/mysql
volumeMounts:
- name: data
  mountPath: /var/lib/mysql
```



```
    subPath: mysql
  - name: conf
    mountPath: /etc/mysql/conf.d
containers:
- name: mysql
  image: mysql:5.7
  env:
  - name: MYSQL_ALLOW_EMPTY_PASSWORD
    value: "1"
  ports:
  - name: mysql
    containerPort: 3306
  volumeMounts:
  - name: data
    mountPath: /var/lib/mysql
    subPath: mysql
  - name: conf
    mountPath: /etc/mysql/conf.d
  resources:
    requests:
      cpu: 500m
      memory: 1Gi
  livenessProbe:
    exec:
      command: ["mysqladmin", "ping"]
    initialDelaySeconds: 30
    periodSeconds: 10
    timeoutSeconds: 5
```



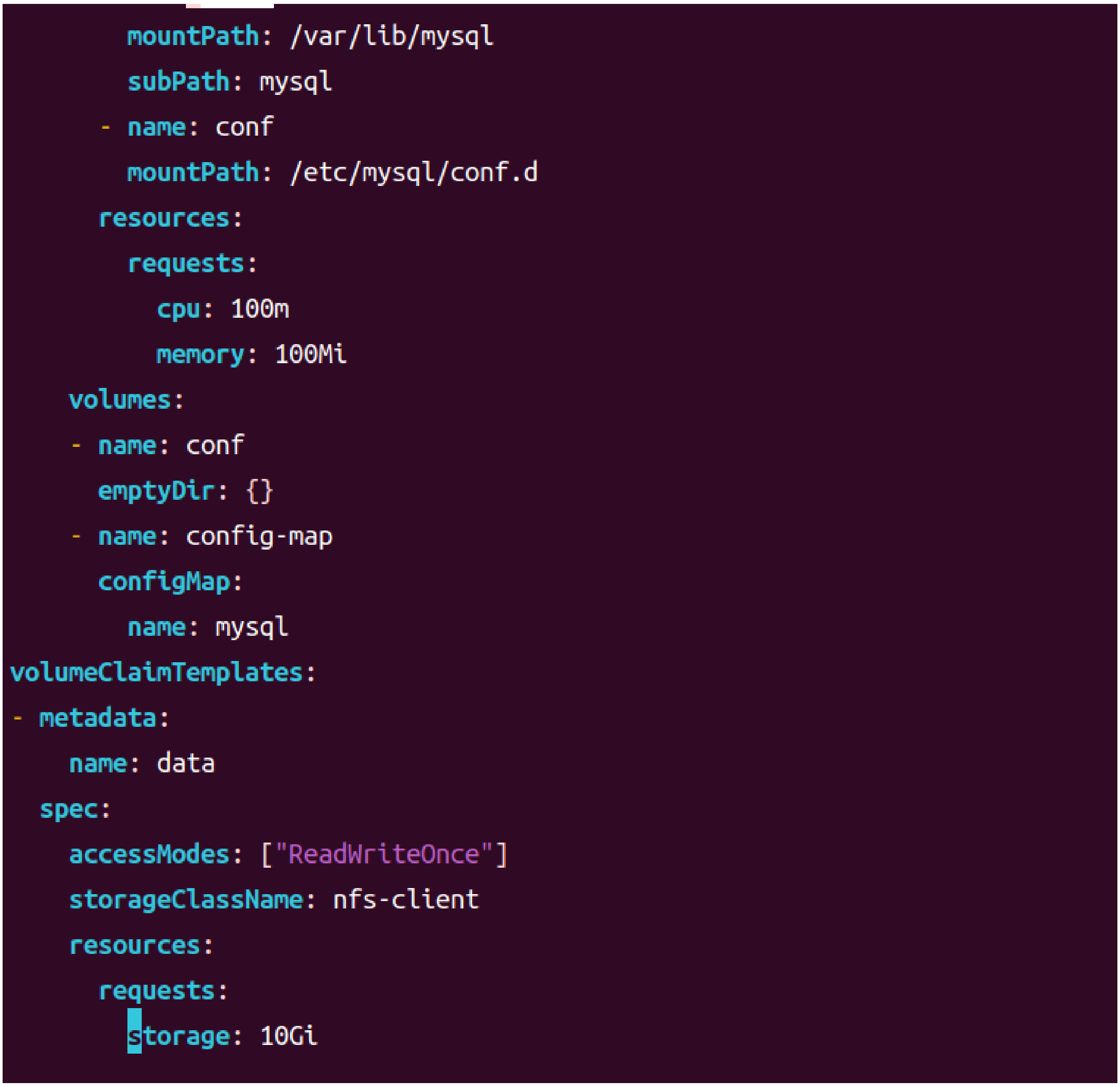
```
  readinessProbe:
    exec:
      # Check we can execute queries over TCP (skip-networking is off).
      command: ["mysql", "-h", "127.0.0.1", "-e", "SELECT 1"]
    initialDelaySeconds: 5
    periodSeconds: 2
    timeoutSeconds: 1
- name: xtrabackup
  image: gcr.io/google-samples/xtrabackup:1.0
  ports:
  - name: xtrabackup
    containerPort: 3307
  command:
  - bash
  - "-c"
  - |
    set -ex
    cd /var/lib/mysql

    # Determine binlog position of cloned data, if any.
    if [[ -f xtrabackup_slave_info && "x$(<xtrabackup_slave_info)" != "x" ]]; then
      # XtraBackup already generated a partial "CHANGE MASTER TO" query
      # because we're cloning from an existing replica. (Need to remove the tailing
      cat xtrabackup_slave_info | sed -E 's/;$//g' > change_master_to.sql.in
      # Ignore xtrabackup_binlog_info in this case (it's useless).
      rm -f xtrabackup_slave_info xtrabackup_binlog_info
    elif [[ -f xtrabackup_binlog_info ]]; then
      # We're cloning directly from primary. Parse binlog position.
    [[ `cat xtrabackup_binlog_info` =~ ^(.*?)[[:space:]]+(.*?)$ ]] || exit 1
    rm -f xtrabackup_binlog_info xtrabackup_slave_info
    echo "CHANGE MASTER TO MASTER_LOG_FILE='${BASH_REMATCH[1]}',\
          MASTER_LOG_POS=${BASH_REMATCH[2]}" > change_master_to.sql.in
  fi

  # Check if we need to complete a clone by starting replication.
  if [[ -f change_master_to.sql.in ]]; then
    echo "Waiting for mysqld to be ready (accepting connections)"
    until mysql -h 127.0.0.1 -e "SELECT 1"; do sleep 1; done

    echo "Initializing replication from clone position"
    mysql -h 127.0.0.1 \
          -e "$(<change_master_to.sql.in), \
                  MASTER_HOST='mysql-0.mysql', \
                  MASTER_USER='root', \
                  MASTER_PASSWORD='', \
                  MASTER_CONNECT_RETRY=10; \
                START SLAVE;" || exit 1
    # In case of container restart, attempt this at-most-once.
    mv change_master_to.sql.in change_master_to.sql.orig
  fi

  # Start a server to send backups when requested by peers.
  exec ncat --listen --keep-open --send-only --max-conns=1 3307 -c \
    "xtrabackup --backup --slave-info --stream=xbstream --host=127.0.0.1 --user=root"
volumeMounts:
- name: data
```

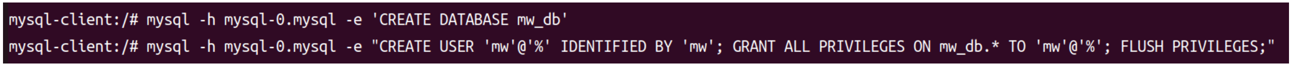


```
        mountPath: /var/lib/mysql
        subPath: mysql
      - name: conf
        mountPath: /etc/mysql/conf.d
      resources:
        requests:
          cpu: 100m
          memory: 100Mi
    volumes:
    - name: conf
      emptyDir: {}
    - name: config-map
      configMap:
        name: mysql
volumeClaimTemplates:
- metadata:
    name: data
  spec:
    accessModes: ["ReadWriteOnce"]
    storageClassName: nfs-client
    resources:
      requests:
        storage: 10Gi
```

18. 데이터베이스 생성.

MYSQL 클라이언트 도구가 포함된 파드 컨테이너 실행 후.
#kubectl run mysql-client it –image=ghcr.io/c1t1d0s7/networkmulittool –-rm bash

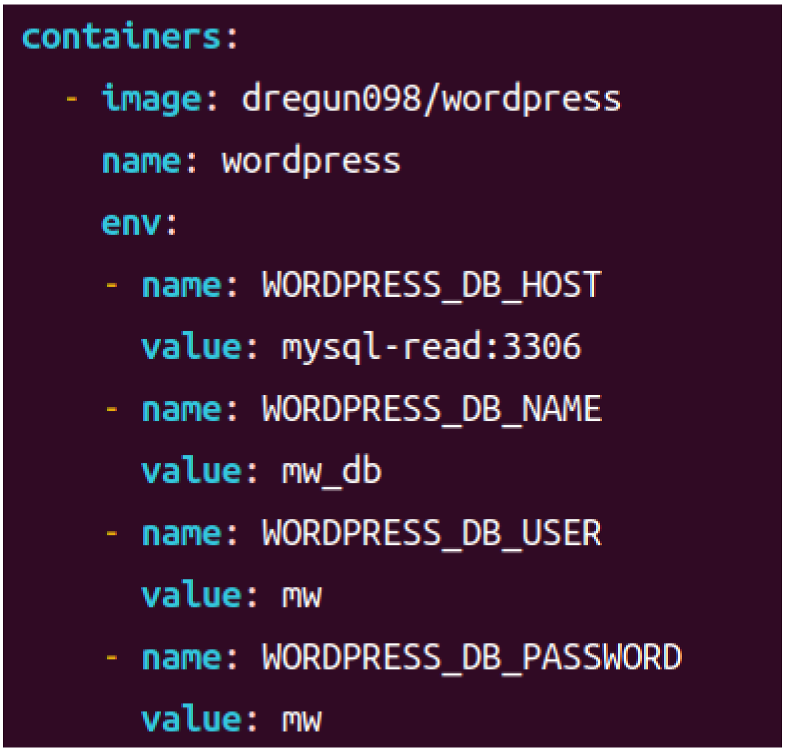
mw_db 데이터베이스 생성.(user:mw, pw:mw)



```
mysql-client:/# mysql -h mysql-0.mysql -e 'CREATE DATABASE mw_db'
mysql-client:/# mysql -h mysql-0.mysql -e "CREATE USER 'mw'@'%' IDENTIFIED BY 'mw'; GRANT ALL PRIVILEGES ON mw_db.* TO 'mw'@'%'; FLUSH PRIVILEGES;"
```

19. wordpress.yml 수정.

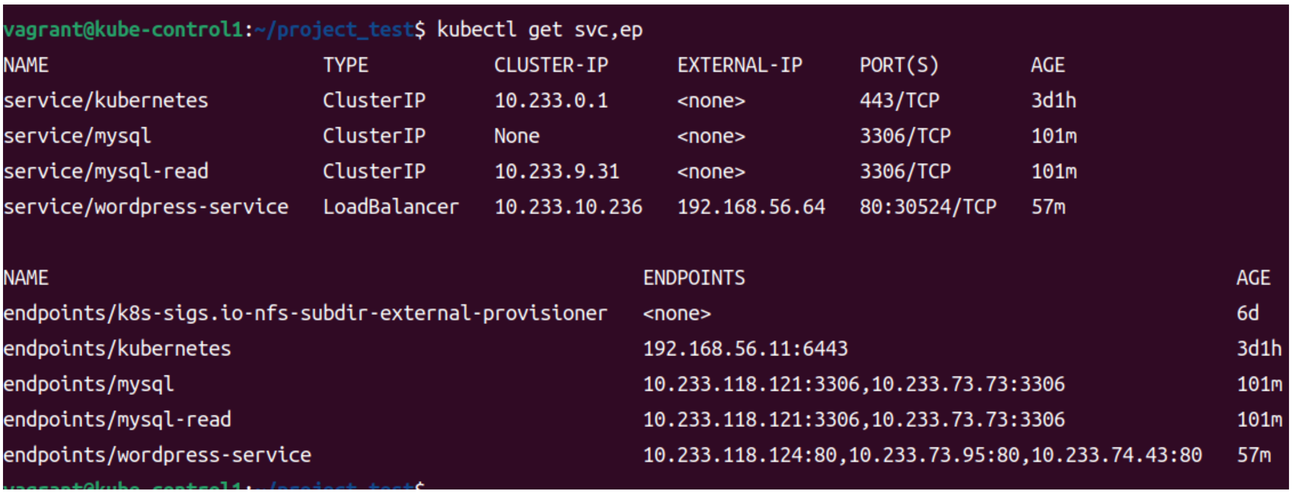
호스트 변수를 mysql-read 서비스로 수정.



```
containers:
  - image: dregun098/wordpress
    name: wordpress
    env:
    - name: WORDPRESS_DB_HOST
      value: mysql-read:3306
    - name: WORDPRESS_DB_NAME
      value: mw_db
    - name: WORDPRESS_DB_USER
      value: mw
    - name: WORDPRESS_DB_PASSWORD
      value: mw
```

20. 서비스 확인.

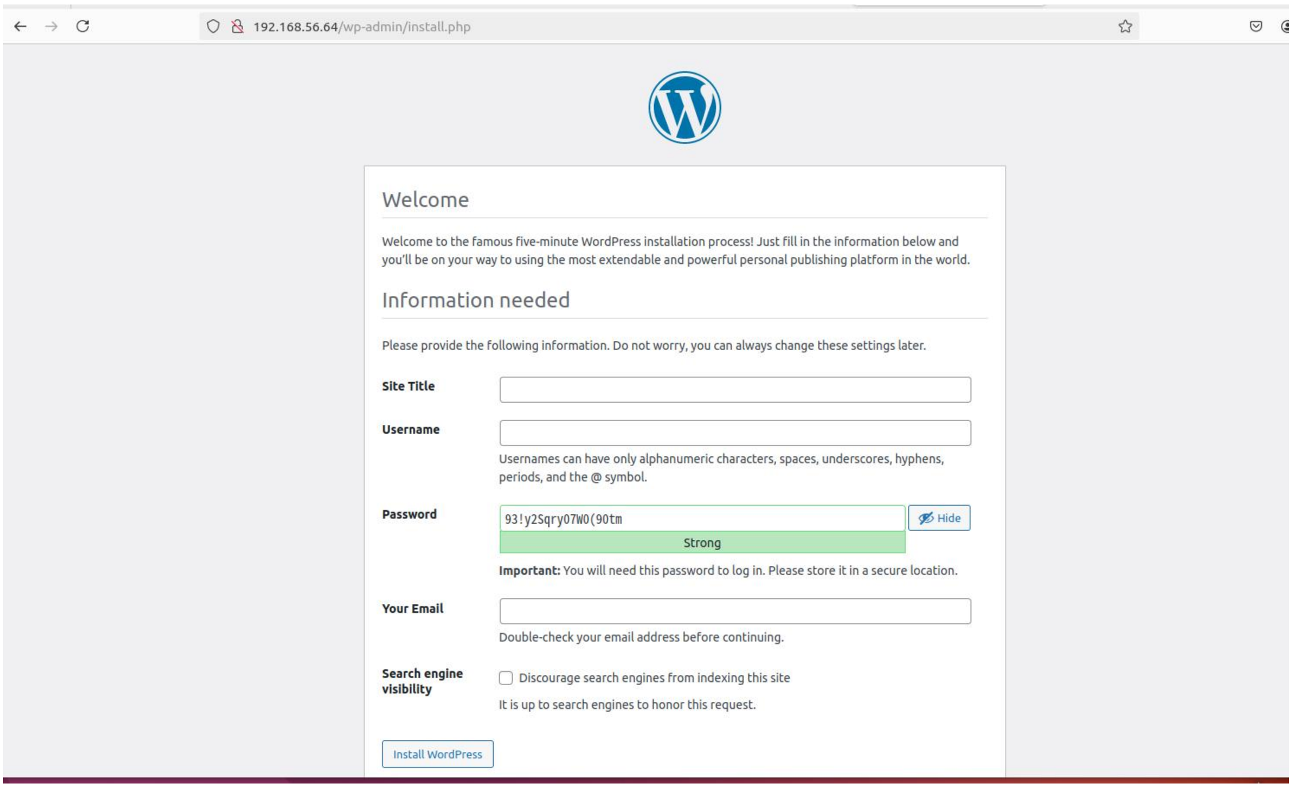
#kubectl get svc,ep



```
vagrant@kube-control1:~/project_test$ kubectl get svc,ep
NAME                        TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)        AGE
service/kubernetes          ClusterIP      10.233.0.1     <none>          443/TCP        3d1h
service/mysql               ClusterIP      None           <none>          3306/TCP       101m
service/mysql-read          ClusterIP      10.233.9.31    <none>          3306/TCP       101m
service/wordpress-service   LoadBalancer   10.233.10.236  192.168.56.64   80:30524/TCP   57m

NAME                                              ENDPOINTS                                                AGE
endpoints/k8s-sigs.io-nfs-subdir-external-provisioner   <none>                                             6d
endpoints/kubernetes                              192.168.56.11:6443                                       3d1h
endpoints/mysql                                   10.233.118.121:3306,10.233.73.73:3306                    101m
endpoints/mysql-read                              10.233.118.121:3306,10.233.73.73:3306                    101m
endpoints/wordpress-service                       10.233.118.124:80,10.233.73.95:80,10.233.74.43:80        57m
vagrant@kube-control1:~/project_test$
```

21. 접속 확인.



192.168.56.64/wp-admin/install.php

Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

Site Title

Username

Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password: 93!y2Sqry07W0(90tm

Strong

Important: You will need this password to log in. Please store it in a secure location.

Your Email

Double-check your email address before continuing.

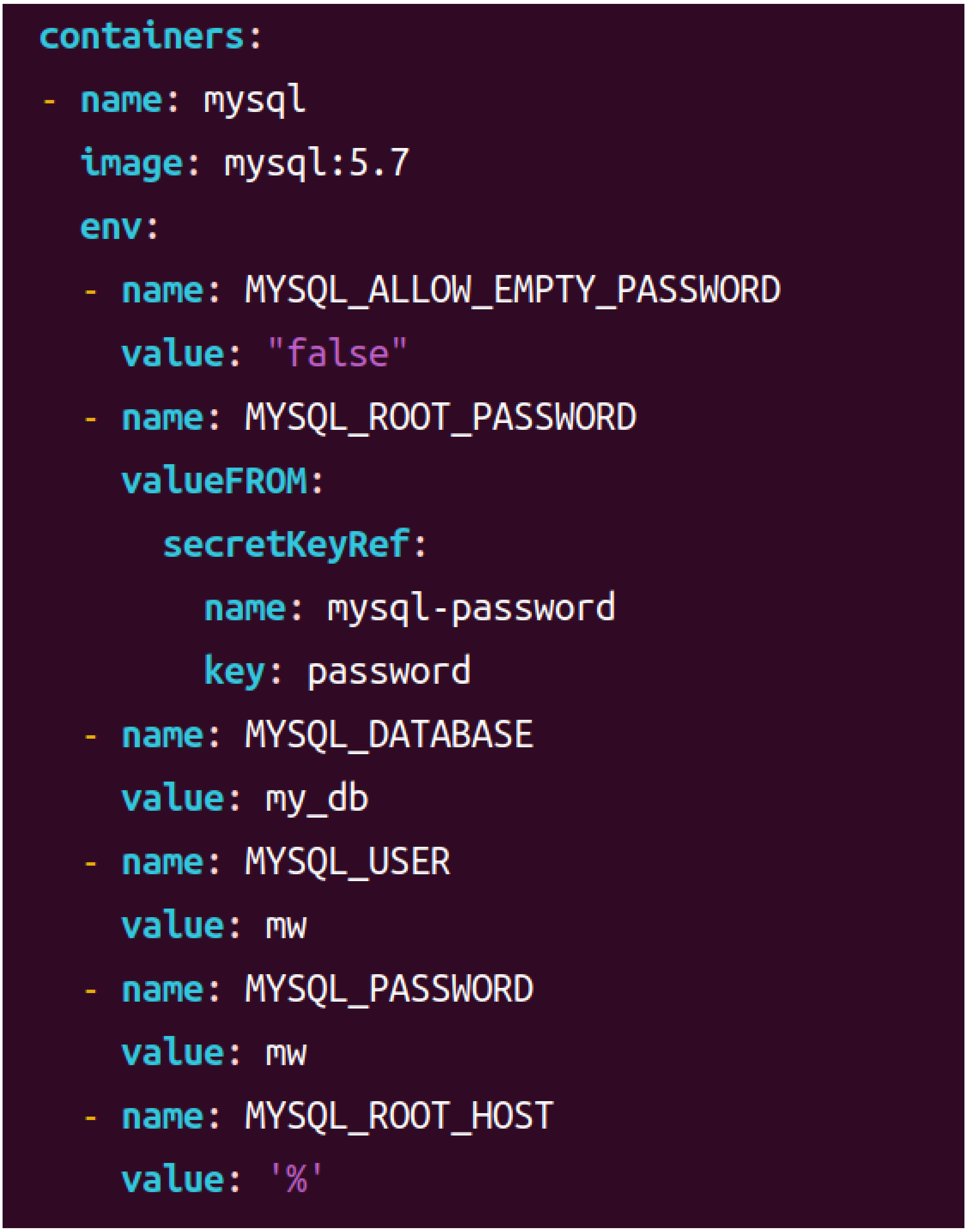
Search engine visibility: □ Discourage search engines from indexing this site

It is up to search engines to honor this request.

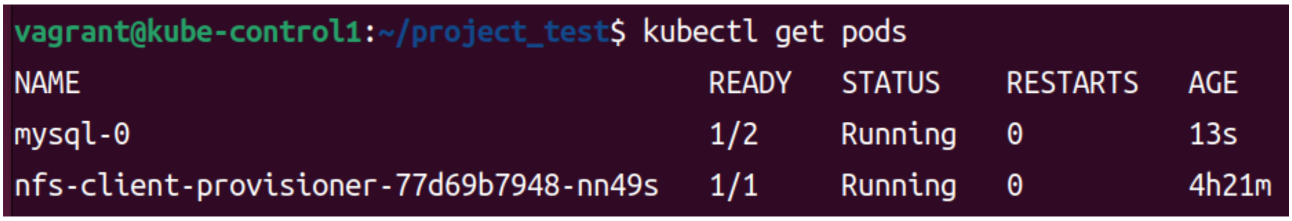
해결하지 못한 문제

문제 1.

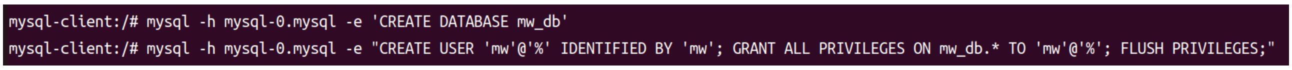
#vim mysql-statefulset.yml
스테이트 풀셋 에서 데이터베이스이름, 유저, 비밀번호 변수를 추가하고 실행하면 파드가 하나만 생성이 된다.



```yaml
containers:
- name: mysql
  image: mysql:5.7
  env:
  - name: MYSQL_ALLOW_EMPTY_PASSWORD
    value: "false"
  - name: MYSQL_ROOT_PASSWORD
    valueFROM:
      secretKeyRef:
        name: mysql-password
        key: password
  - name: MYSQL_DATABASE
    value: my_db
  - name: MYSQL_USER
    value: mw
  - name: MYSQL_PASSWORD
    value: mw
  - name: MYSQL_ROOT_HOST
    value: '%'
```



```
vagrant@kube-control1:~/project_test$ kubectl get pods
NAME                                      READY   STATUS    RESTARTS   AGE
mysql-0                                   1/2     Running   0          13s
nfs-client-provisioner-77d69b7948-nn49s   1/1     Running   0          4h21m
```

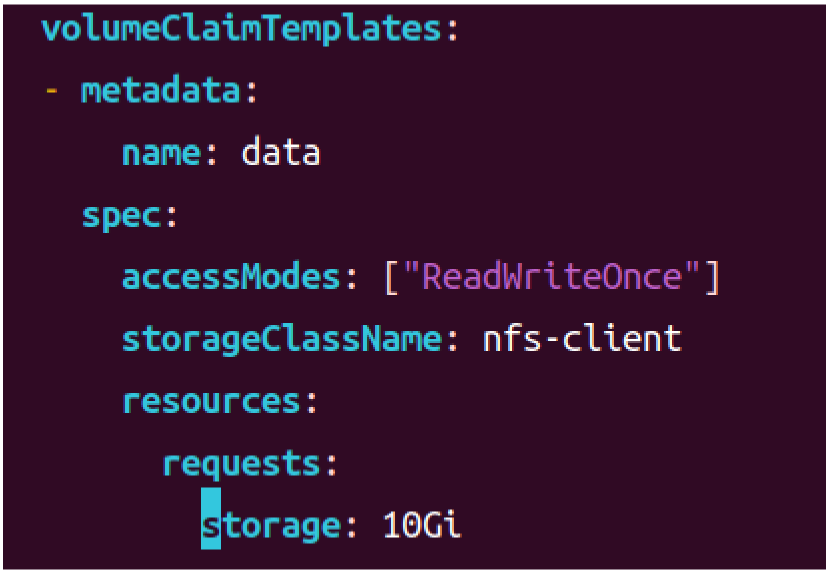


```
mysql-client:/# mysql -h mysql-0.mysql -e 'CREATE DATABASE mw_db'
mysql-client:/# mysql -h mysql-0.mysql -e "CREATE USER 'mw'@'%' IDENTIFIED BY 'mw'; GRANT ALL PRIVILEGES ON mw_db.* TO 'mw'@'%'; FLUSH PRIVILEGES;"
```

그래서 파드 생성 후 직접 db 안에 데이터베이스, 유저, 비밀번호 생성.

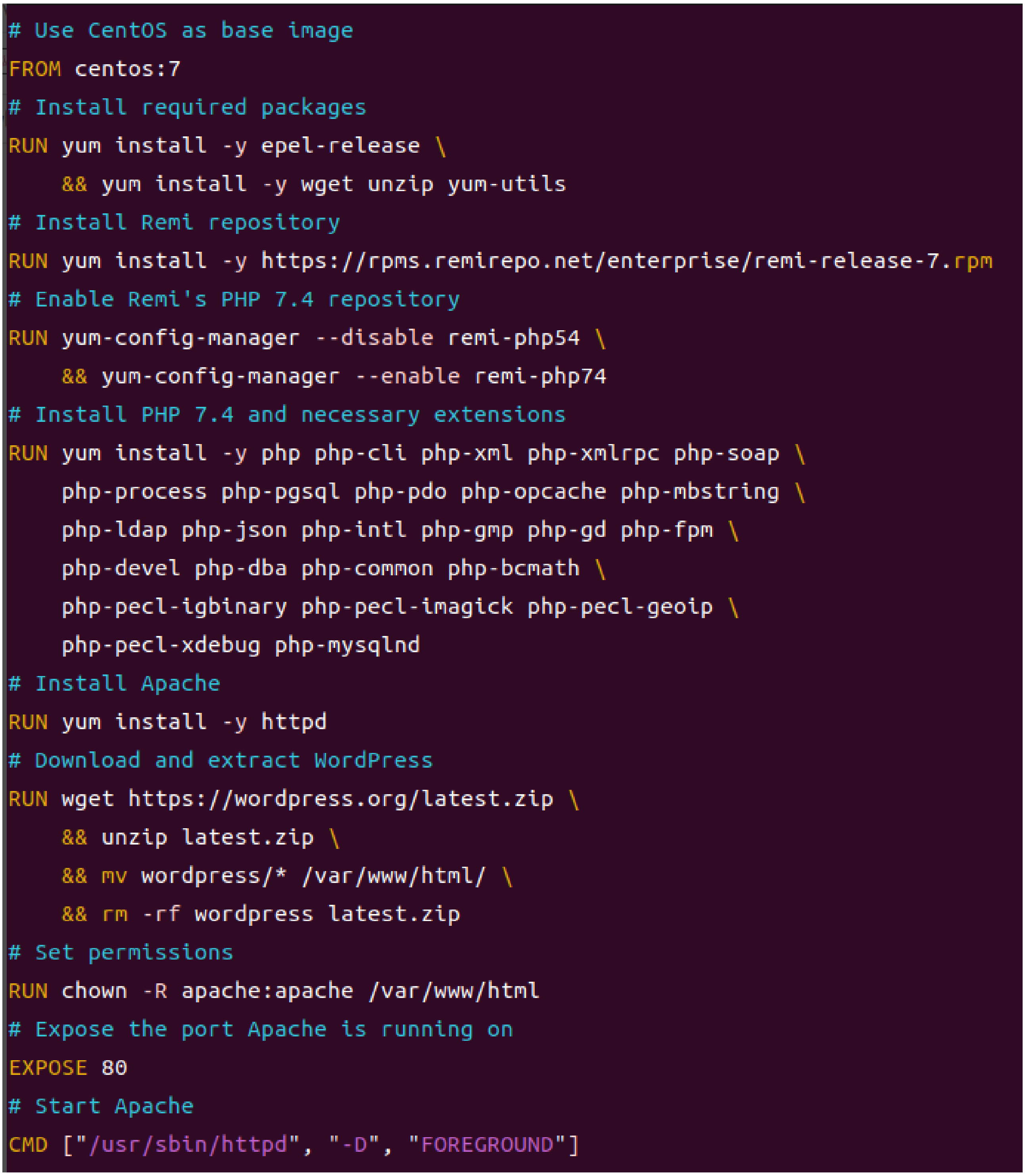
문제 2.

정적 pvc 인 project-pvc 로 스테이트풀셋을 만들고 싶었지만, 스테이트 풀셋은 동적 프로비저닝만 해당되는 거 같다.



```
volumeClaimTemplates:
- metadata:
    name: data
  spec:
    accessModes: ["ReadWriteOnce"]
    storageClassName: nfs-client
    resources:
      requests:
        storage: 10Gi
```

문제 3.



```
# Use CentOS as base image
FROM centos:7
# Install required packages
RUN yum install -y epel-release \
    && yum install -y wget unzip yum-utils
# Install Remi repository
RUN yum install -y https://rpms.remirepo.net/enterprise/remi-release-7.rpm
# Enable Remi's PHP 7.4 repository
RUN yum-config-manager --disable remi-php54 \
    && yum-config-manager --enable remi-php74
# Install PHP 7.4 and necessary extensions
RUN yum install -y php php-cli php-xml php-xmlrpc php-soap \
    php-process php-pgsql php-pdo php-opcache php-mbstring \
    php-ldap php-json php-intl php-gmp php-gd php-fpm \
    php-devel php-dba php-common php-bcmath \
    php-pecl-igbinary php-pecl-imagick php-pecl-geoip \
    php-pecl-xdebug php-mysqlnd
# Install Apache
RUN yum install -y httpd
# Download and extract WordPress
RUN wget https://wordpress.org/latest.zip \
    && unzip latest.zip \
    && mv wordpress/* /var/www/html/ \
    && rm -rf wordpress latest.zip
# Set permissions
RUN chown -R apache:apache /var/www/html
# Expose the port Apache is running on
EXPOSE 80
# Start Apache
CMD ["/usr/sbin/httpd", "-D", "FOREGROUND"]
```

해당 도커파일로 이미지를 생성이 잘 되었으나, 다시 이미지를 생성하니 에러가 발생 함.