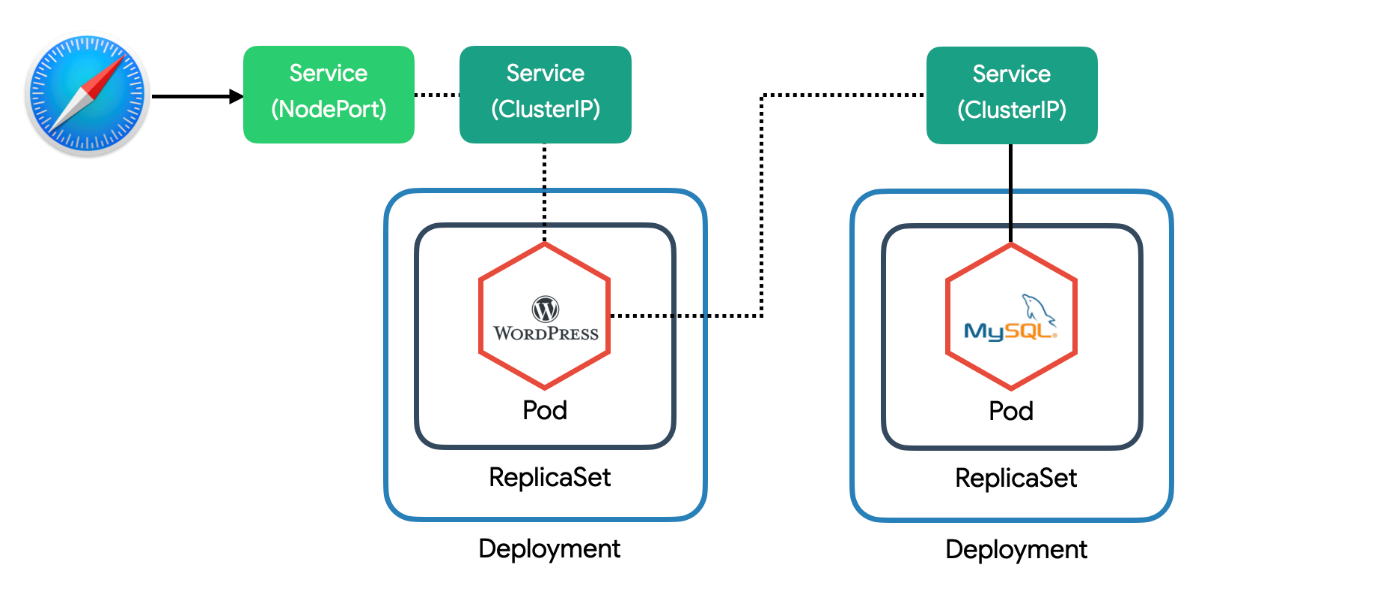
**Architecture**



위와 같은 Architecture로 Wordpress를 배포 해주자.

**Create spec files**

wordpress-k8s.yml

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: wordpress-mysql  labels:  app: wordpress  spec:  selector:  matchLabels:  app: wordpress  tier: mysql  template:  metadata:  labels:  app: wordpress  tier: mysql  spec:  containers:  - image: mysql:5.6  name: mysql  env:  - name: MYSQL\_ROOT\_PASSWORD  value: password  ports:  - containerPort: 3306  name: mysql  ---  apiVersion: v1  kind: Service  metadata:  name: wordpress-mysql  labels:  app: wordpress  spec:  ports:  - port: 3306  selector:  app: wordpress  tier: mysql  ---  apiVersion: apps/v1  kind: Deployment  metadata:  name: wordpress  labels:  app: wordpress  spec:  selector:  matchLabels:  app: wordpress  tier: frontend  template:  metadata:  labels:  app: wordpress  tier: frontend  spec:  containers:  - image: wordpress:5.5.3-apache  name: wordpress  env:  - name: WORDPRESS\_DB\_HOST  value: wordpress-mysql  - name: WORDPRESS\_DB\_PASSWORD  value: password  ports:  - containerPort: 80  name: wordpress  ---  apiVersion: v1  kind: Service  metadata:  name: wordpress  labels:  app: wordpress  spec:  type: NodePort  ports:  - port: 80  selector:  app: wordpress  tier: frontend |

해당 파일을 배포해주자.

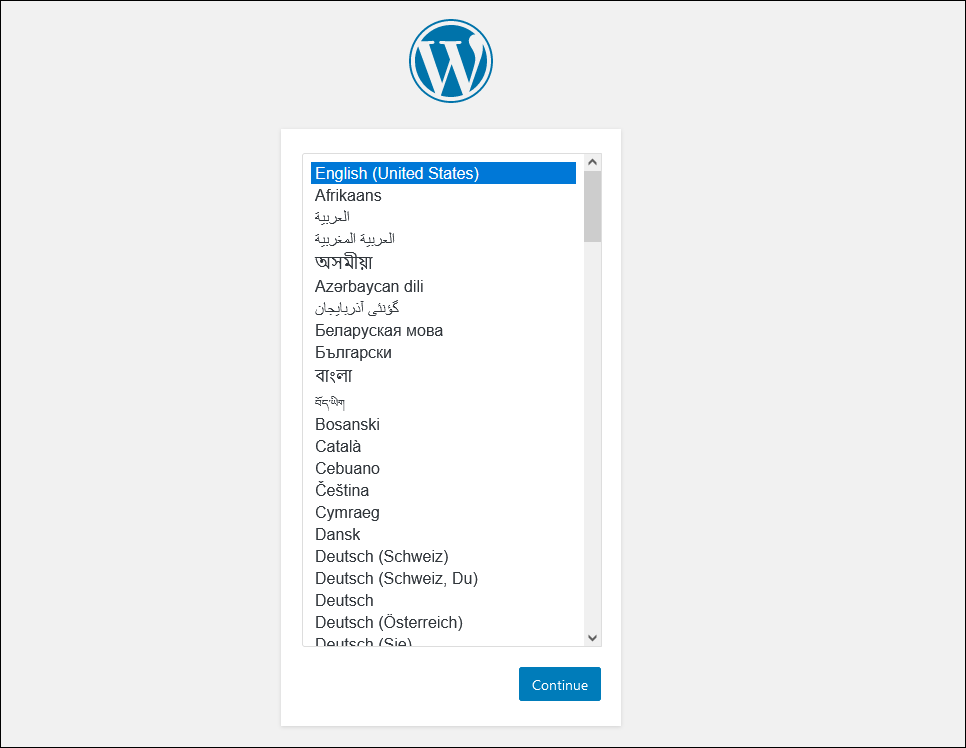
|  |
| --- |
| $ kubectl apply -f wordpress-k8s.yml |

그리고 배포가 완료 되었다면 아래와 같은 명령어로 확인 해보자.

|  |
| --- |
| $ kubectl get all |

|  |
| --- |
| $ kubectl get svc wordpress -o yaml | grep nodePort |

그리고 localhost:<nodeport> 로 접근해보자.



다음과 같이 출력되면 wordpress 배포가 된 것이다.