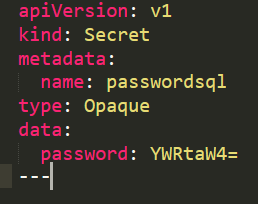
**1. CREATE A SECRET FOR MYSQL PASSWORD**

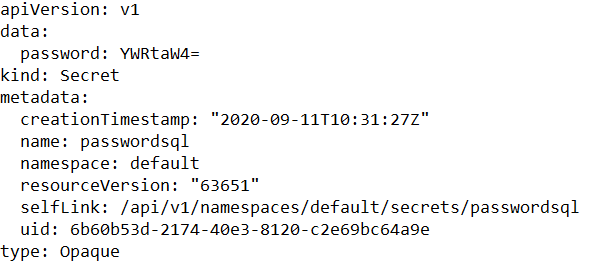
* Encode (base 64) password “admin”
* Create a secret object (use secret.yaml as a template) and set the Base64 encoded password



**$ kubectl create -f C:\k8syamls\secret.yaml**



After creation, file secret.yaml has changed:



**2. DEPLOY PERSISTENT VOLUMES FOR WORDPRESS & MYSQL (USE PV.YAML AS A TEMPLATE) WITH THE FOLLOWING SPEC:**

* storage: 1Gi (for Both)
* accessModes: ReadWriteMany (for Both)
* storageClassName: standard (for Both)
* hostPath: /data/html (for Wordpress)
* hostPath: path: /data/mysql (for MySQL)

**3. DEPLOY PERSISTENT VOLUME CLAIMS FOR WORDPRESS & MYSQL (USE PVC.YAML AS A TEMPLATE)**

**4. DEPLOY MYSQL (USE DEPLOYMENT.YAML AND SVC.YAML AS TEMPLATES FOR YOUR MANIFESTS)**

The Service

* maps MySQL’s port 3306 and
* makes it available for all containers with the labels app:wordpress and tier:mysql
* **has type “ClusterIP”**

The Deployment declares the creation strategy and specs of MySQL container:

* it’s an image from the Docker Hub: mysql:5.6
* it has app:wordpress and tier:mysql (used in Service)
* it contains an environment variable called MYSQL\_ROOT\_PASSWORD which holds the value from our secret password
* it has an open port 3306
* it has a volume claim mounted in /var/lib/mysql.

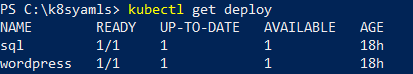
**5. DEPLOY WORDPRESS (USE DEPLOYMENT.YAML AND SVC.YAML AS TEMPLATES FOR YOUR MANIFESTS)**

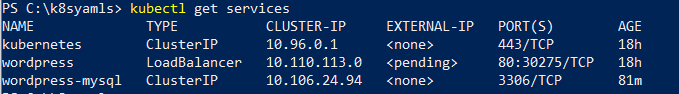
**The Service:**

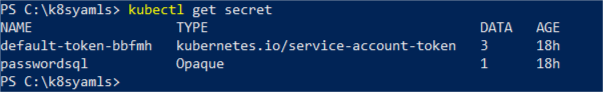
* **maps port 80 of the container to the node’s external IP:Port**
* makes it available **for all containers with the labels app:wordpress and tier:frontend**
* **has type “LoadBalancer”**

**The Deployment declares the creation spec of our WordPress container:**

* **it’s an image from the Docker Hub: wordpress:4.8-apache**
* **it has app:wordpress & tier:frontend labels (used in Service)**
* **it contains environment variables WORDPRESS\_DB\_HOST, which is the internal host name of the MySQL instance, and WORDPRESS\_DB\_PASSWORD, which holds the value from our secret password**
* **it has an open port 80**
* **it has a volume claim mounted in /var/www/html from which the WP sources are served.**



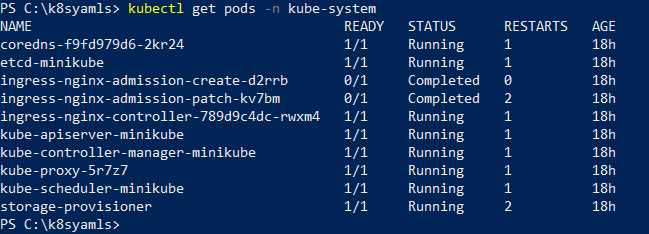




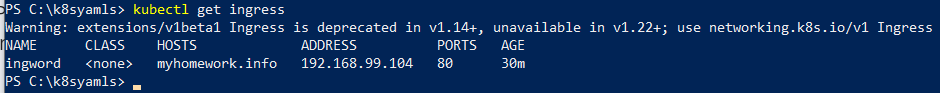
6. Deploy Nginx ingress controller to your minikube

* Expose service of the Ingress controller outside of Minikube by running “minikube service” command.

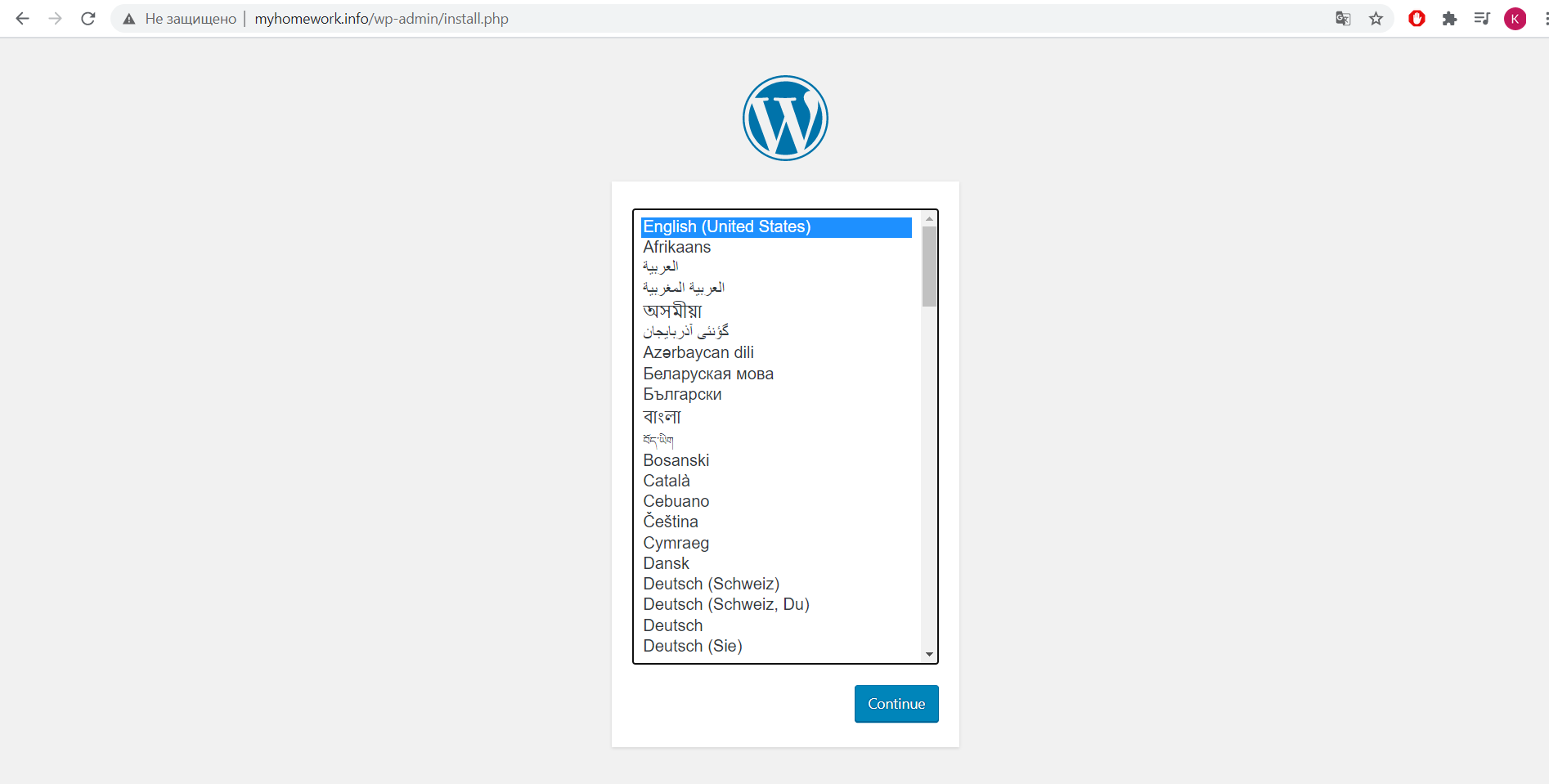
**minikube addons enable ingress**



7. Create an ingress object to expose Wordpress to outer network using Nginx Ingress Controller (use ingress.yaml as a template for your manifest)



8. Acess WordPress from your Browser and do initial configuration.



9. Upload manifests and report to Github