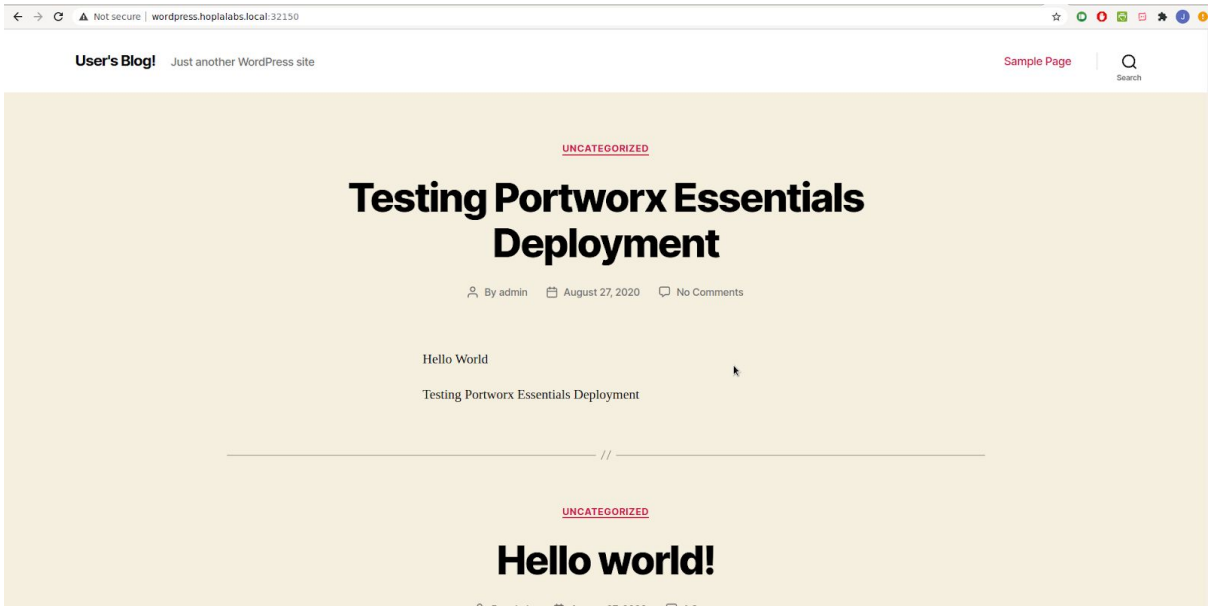
PORTWORX ESSENTIALS

Accedemos de nuevo a WordPress y vemos que la entrada que habíamos creado está disponible.



```
$ kubectl get nodes
NAME        STATUS   ROLES    AGE     VERSION
kube-node1   Ready    master   10m     v1.18.2
kube-node2   Ready    <none>   7m45s   v1.18.2
kube-node3   Ready    <none>   5m21s   v1.18.2
kube-node4   Ready    <none>   2m57s   v1.18.2
```

```
helm install --name ingress-nginx ingress-nginx/ingress-nginx
```

```
$ kubectl apply -f
'https://install.portworx.com/2.5?mc=false&kbver=1.18.0&oem=esse&user=d4f15375-d0bb-11ea-a2c5-c24e499
c7467&b=true&f=true&j=auto&c=px-cluster-d06e65db-5b34-4692-ba04-b3862269ebdc&stork=true&lh=true&st=k8s'
service/portworx-service created
customresourcedefinition.apiextensions.k8s.io/volumeplacementstrategies.portworx.io created
```

```
serviceaccount/px-account created
clusterrole.rbac.authorization.k8s.io/node-get-put-list-role created
clusterrolebinding.rbac.authorization.k8s.io/node-role-binding created
namespace/portworx created
role.rbac.authorization.k8s.io/px-role created
rolebinding.rbac.authorization.k8s.io/px-role-binding created
secret/px-essential created
daemonset.apps/portworx created
service/portworx-api created
daemonset.apps/portworx-api created
serviceaccount/stork-account created
clusterrole.rbac.authorization.k8s.io/stork-role created
clusterrolebinding.rbac.authorization.k8s.io/stork-role-binding created
deployment.apps/stork created
storageclass.storage.k8s.io/stork-snapshot-sc created
service/stork-service created
configmap/stork-config created
serviceaccount/stork-scheduler-account created
clusterrole.rbac.authorization.k8s.io/stork-scheduler-role created
clusterrolebinding.rbac.authorization.k8s.io/stork-scheduler-role-binding created
deployment.apps/stork-scheduler created
serviceaccount/px-lh-account created
clusterrole.rbac.authorization.k8s.io/px-lh-role created
clusterrolebinding.rbac.authorization.k8s.io/px-lh-role-binding created
service/px-lighthouse created
deployment.apps/px-lighthouse created
configmap/autopilot-config created
serviceaccount/autopilot-account created
clusterrole.rbac.authorization.k8s.io/autopilot-role created
clusterrolebinding.rbac.authorization.k8s.io/autopilot-role-binding created
deployment.apps/autopilot created
service/autopilot created
vagrant@kube-node1:~$
```

```
provision@kube1:~$ kubectl get pod -A
NAMESPACE     NAME                                READY   STATUS    RESTARTS   AGE
default       ingress-nginx-controller-5d6fbbddb6-n4twx   1/1     Running   0          55m
kube-system   autopilot-58db5f4dbb-q8qsq                  1/1     Running   0          54m
kube-system   calico-kube-controllers-57546b46d6-j47gw    1/1     Running   0          61m
```

```
kube-system   calico-node-4nsmf                  1/1     Running   0          61m
kube-system   calico-node-dqbdp                  1/1     Running   0          61m
kube-system   calico-node-n4g6s                  1/1     Running   0          61m
kube-system   calico-node-nd5hb                  1/1     Running   0          61m
kube-system   coredns-66bff467f8-5x2mc           1/1     Running   0          61m
kube-system   coredns-66bff467f8-vdfj4           1/1     Running   0          61m
kube-system   etcd-kube1                         1/1     Running   0          61m
kube-system   kube-apiserver-kube1               1/1     Running   0          61m
kube-system   kube-controller-manager-kube1      1/1     Running   2          61m
kube-system   kube-proxy-4lp2s                   1/1     Running   0          61m
kube-system   kube-proxy-4x9nf                   1/1     Running   0          61m
kube-system   kube-proxy-8rffq                   1/1     Running   0          61m
kube-system   kube-proxy-8xt9d                   1/1     Running   0          61m
kube-system   kube-scheduler-kube1               1/1     Running   2          61m
kube-system   portworx-api-4rfqx                 1/1     Running   0          54m
kube-system   portworx-api-6nr97                 1/1     Running   0          54m
kube-system   portworx-api-97glw                 1/1     Running   0          54m
kube-system   portworx-kzqgf                     1/1     Running   0          54m
kube-system   portworx-m7rjt                     1/1     Running   0          54m
kube-system   portworx-srw8r                     1/1     Running   0          54m
kube-system   px-lighthouse-5896bc9b8d-fhqf8     3/3     Running   0          54m
kube-system   stork-f4799f469-c4jm6              1/1     Running   0          54m
kube-system   stork-f4799f469-d5cj6              1/1     Running   0          54m
kube-system   stork-f4799f469-t6tv5              1/1     Running   0          54m
kube-system   stork-scheduler-7bd6564484-9rn5v   1/1     Running   1          54m
kube-system   stork-scheduler-7bd6564484-bjfmm   1/1     Running   0          54m
kube-system   stork-scheduler-7bd6564484-k6bxh   1/1     Running   0          54m
```

```
$ cat<<EOF|kubectl create -f -
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: portworx-sc
provisioner: kubernetes.io/portworx-volume
parameters:
  repl: "1"
EOF

storageclass.storage.k8s.io/portworx-sc created

$ kubectl get sc
```

```
NAME                PROVISIONER                    RECLAIMPOLICY   VOLUMEBINDINGMODE
ALLOWVOLUMEEXPANSION   AGE
portworx-sc         kubernetes.io/portworx-volume   Delete          Immediate          false
54s
stork-snapshot-sc   stork-snapshot                 Delete          Immediate          false
61m

  25  curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash
  26  helm help
  27  helm repo add stable https://kubernetes-charts.storage.googleapis.com/


provision@kube1:~$ helm repo add bitnami https://charts.bitnami.com/bitnami
"bitnami" has been added to your repositories
provision@kube1:~$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "bitnami" chart repository
...Successfully got an update from the "stable" chart repository
Update Complete. ⎈ Happy Helming!⎈

provision@kube1:~$ helm install nginx-controller --set service.type=NodePort
bitnami/nginx-ingress-controller
NAME: nginx-controller
LAST DEPLOYED: Wed Aug 26 20:13:06 2020
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
** Please be patient while the chart is being deployed **

The nginx-ingress controller has been installed.
Get the application URL by running these commands:
 export HTTP_NODE_PORT=$(kubectl --namespace default get services -o
jsonpath="{.spec.ports[0].nodePort}" nginx-controller-nginx-ingress-controller)
 export HTTPS_NODE_PORT=$(kubectl --namespace default get services -o
jsonpath="{.spec.ports[1].nodePort}" nginx-controller-nginx-ingress-controller)
 export NODE_IP=$(kubectl --namespace default get nodes -o
jsonpath="{.items[0].status.addresses[1].address}")

 echo "Visit http://$NODE_IP:$HTTP_NODE_PORT to access your application via HTTP."
```

```
 echo "Visit https://$NODE_IP:$HTTPS_NODE_PORT to access your application via HTTPS."
```

An example Ingress that makes use of the controller:

```
 apiVersion: extensions/v1beta1
 kind: Ingress
 metadata:
   annotations:
     kubernetes.io/ingress.class: nginx
   name: example
   namespace: foo
 spec:
   rules:
     - host: www.example.com
       http:
         paths:
           - backend:
               serviceName: exampleService
               port: 80
             path: /
   # This section is only required if TLS is to be enabled for the Ingress
   tls:
       - hosts:
           - www.example.com
         secretName: example-tls
```

If TLS is enabled for the Ingress, a Secret containing the certificate and key must also be provided:

```
 apiVersion: v1
 kind: Secret
 metadata:
   name: example-tls
   namespace: foo
 data:
   tls.crt: <base64 encoded cert>
   tls.key: <base64 encoded key>
 type: kubernetes.io/tls
```

```
$ helm repo add bitnami https://charts.bitnami.com/bitnami
```

```
$ helm install wordpress \
--set global.storageClass=portworx-sc \
--set ingress.enabled=true \
--set ingress.hostname=wordpress.hoplalabs.local \
--set wordpressUsername=admin \
--set wordpressPassword=password \
 bitnami/wordpress


NAME: wordpress
LAST DEPLOYED: Wed Aug 26 14:40:40 2020
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
** Please be patient while the chart is being deployed **

Your WordPress site can be accessed through the following DNS name from within your cluster:

   wordpress.default.svc.cluster.local (port 80)

To access your WordPress site from outside the cluster follow the steps below:

1. Get the WordPress URL and associate WordPress hostname to your cluster external IP:

  export CLUSTER_IP=$(minikube ip) # On Minikube. Use: `kubectl cluster-info` on others K8s clusters
  echo "WordPress URL: http://wordpress.hoplalabs.local/"
  echo "$CLUSTER_IP  wordpress.hoplalabs.local" | sudo tee -a /etc/hosts

2. Open a browser and access WordPress using the obtained URL.

3. Login with the following credentials below to see your blog:

 echo Username: admin
 echo Password: $(kubectl get secret --namespace default wordpress -o jsonpath="{.data.wordpress-password}" | base64 --decode)

provision@kube1:~$ kubectl get pods -o wide
```

```
NAME                                READY   STATUS    RESTARTS   AGE   IP           NODE
NOMINATED NODE   READINESS GATES
ingress-nginx-controller-5d6fbbddb6-n4twx   1/1     Running   0          85m   10.10.9.194
worker2   <none>           <none>
wordpress-bfd5f4c68-5ptqs                   1/1     Running   1          10m   10.10.9.198   worker2
<none>           <none>
wordpress-mariadb-0                         1/1     Running   1          10m   10.10.13.8    worker3
<none>           <none>

provision@kube1:~$ kubectl get svc
NAME                                          TYPE          CLUSTER-IP       EXTERNAL-IP
PORT(S)                  AGE
kubernetes                                    ClusterIP     10.96.0.1        <none>        443/TCP
101m
nginx-controller-nginx-ingress-controller     NodePort      10.104.242.127   <none>
80:30202/TCP,443:30055/TCP   45s
nginx-controller-nginx-ingress-controller-default-backend   ClusterIP     10.96.179.54     <none>
80/TCP                   45s
wordpress                                     LoadBalancer  10.106.77.96     <pending>
80:32229/TCP,443:32708/TCP   19m
wordpress-mariadb                             ClusterIP     10.109.177.60    <none>
3306/TCP                 19m
provision@kube1:~$

[zero@antares-codegazers-local ansible]$ curl -I -H "host: wordpress.hoplalabs.local"
192.168.200.111:30202
HTTP/1.1 200 OK
Server: nginx/1.19.2
Date: Wed, 26 Aug 2020 18:14:44 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
Vary: Accept-Encoding
X-Powered-By: PHP/7.4.9
Link: <http://wordpress.hoplalabs.local/wp-json/>; rel="https://api.w.org/"
```