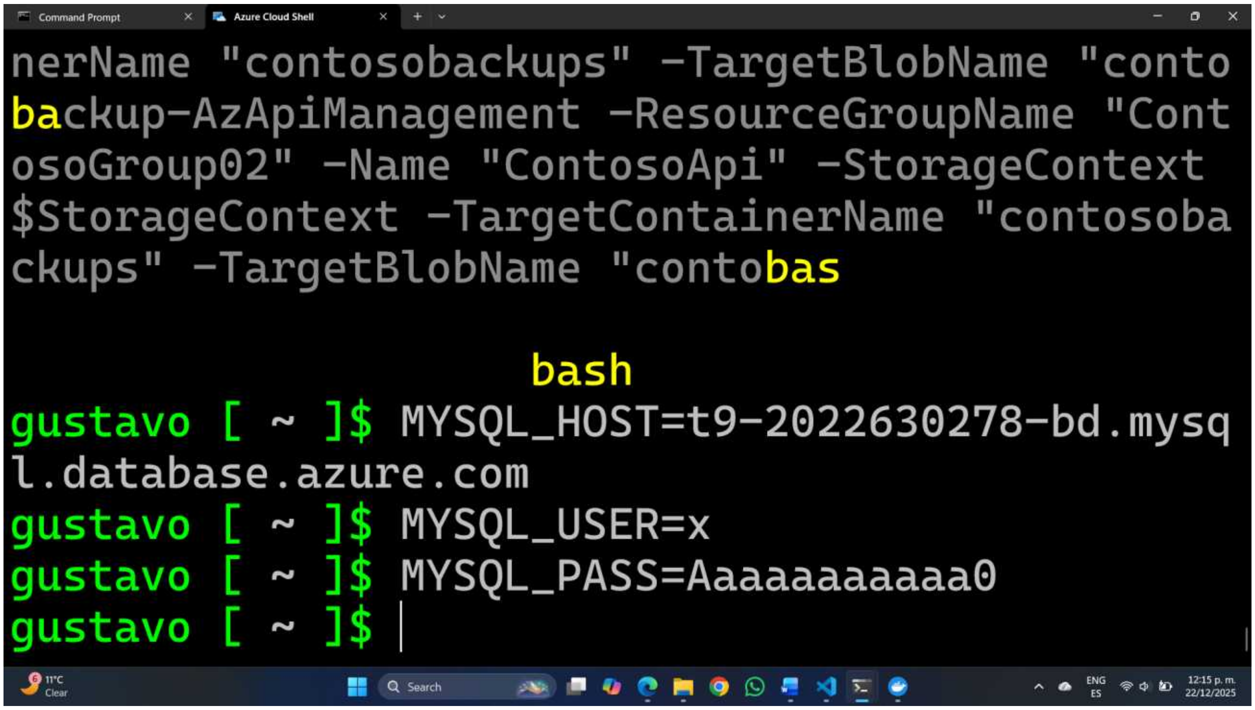
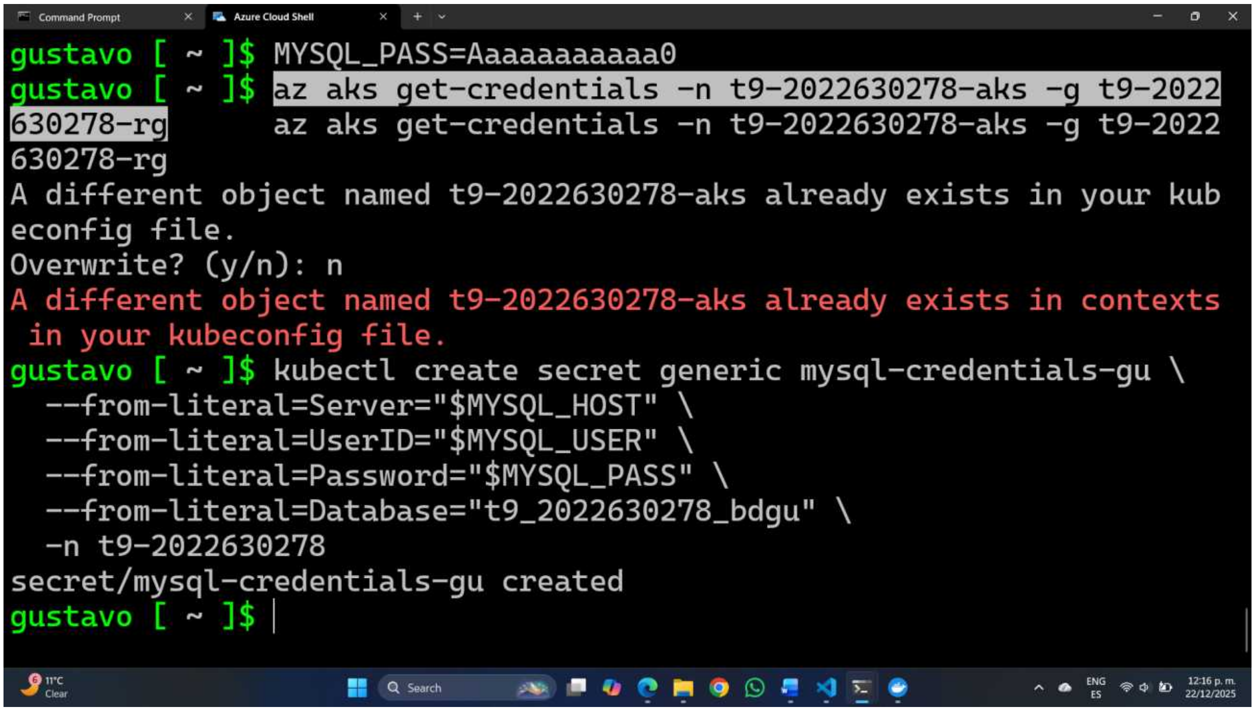


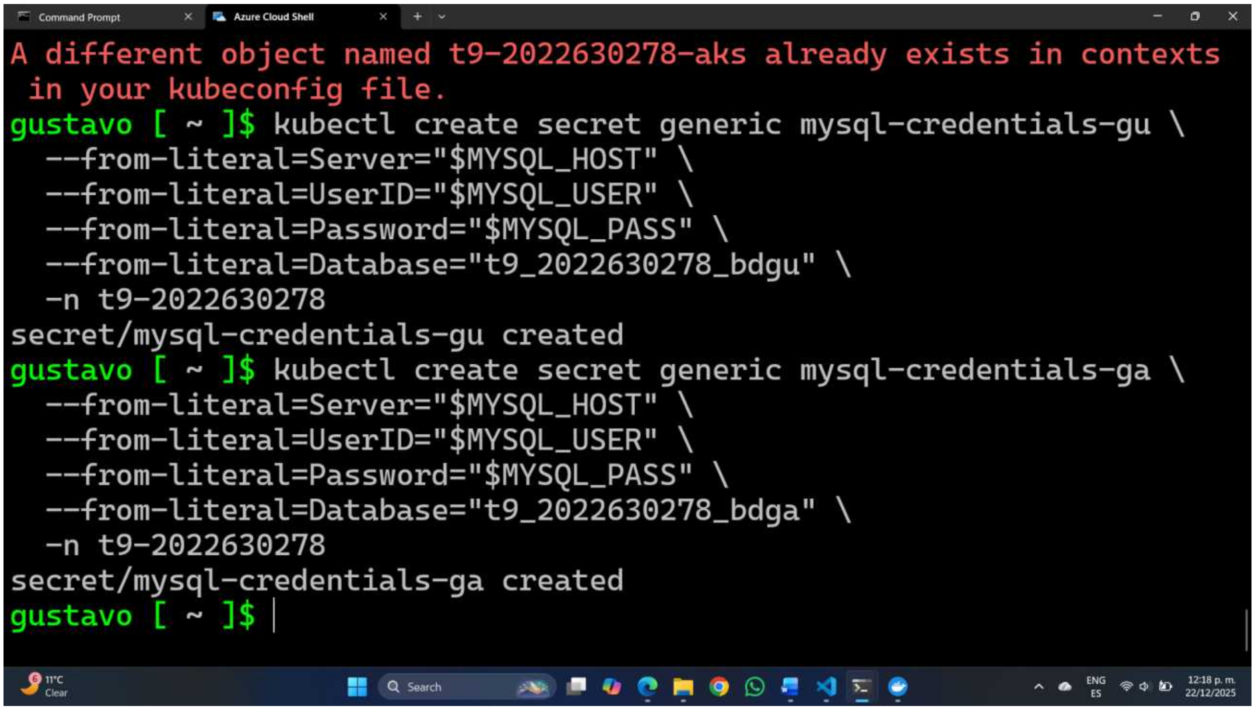
```
nerName "contosobackups" -TargetBlobName "conto
backup-AzApiManagement -ResourceGroupName "Cont
osoGroup02" -Name "ContosoApi" -StorageContext
$StorageContext -TargetContainerName "contosoba
ckups" -TargetBlobName "contobas

                           bash
gustavo [ ~ ]$ MYSQL_HOST=t9-2022630278-bd.mysq
l.database.azure.com
gustavo [ ~ ]$ MYSQL_USER=x
gustavo [ ~ ]$ MYSQL_PASS=Aaaaaaaaaaa0
gustavo [ ~ ]$ |
```

Figura 10.6 Creación de variables de entorno para ingresar a MySQL

Para GU:



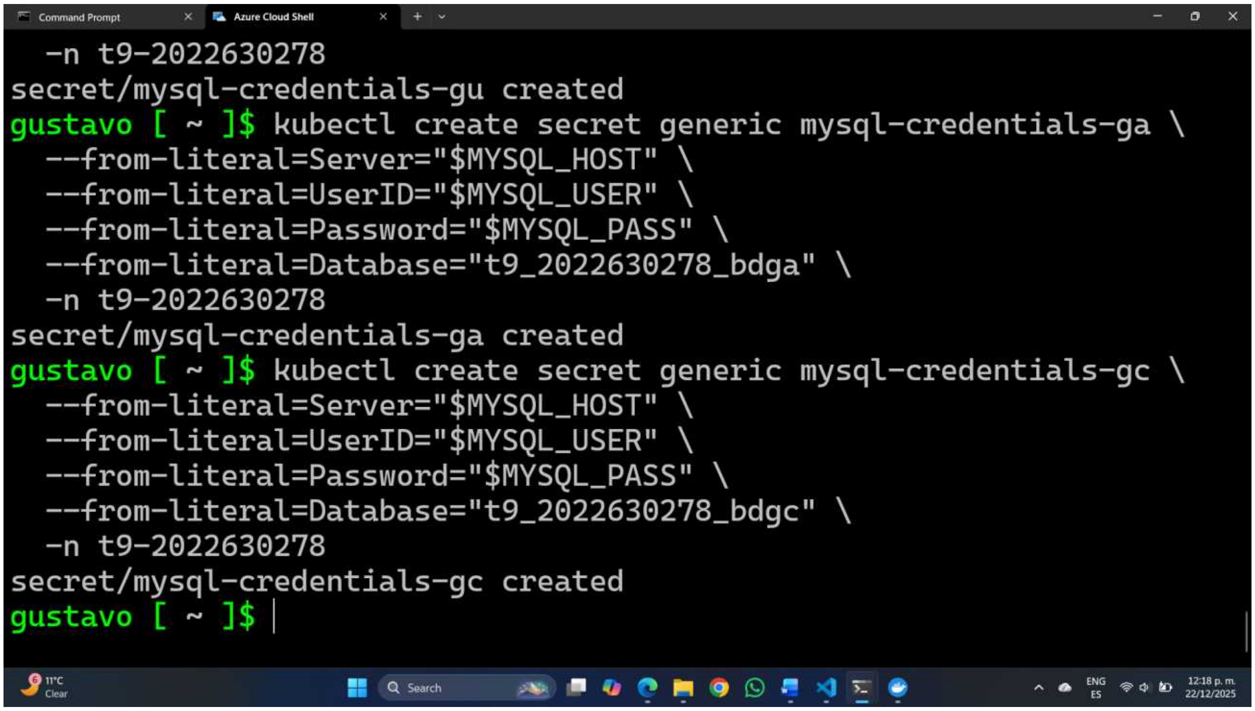
```
gustavo [ ~ ]$ MYSQL_PASS=Aaaaaaaaaaa0
gustavo [ ~ ]$ az aks get-credentials -n t9-2022630278-aks -g t9-2022
630278-rg      az aks get-credentials -n t9-2022630278-aks -g t9-2022
630278-rg
A different object named t9-2022630278-aks already exists in your kub
econfig file.
Overwrite? (y/n): n
A different object named t9-2022630278-aks already exists in contexts
 in your kubeconfig file.
gustavo [ ~ ]$ kubectl create secret generic mysql-credentials-gu \
  --from-literal=Server="$MYSQL_HOST" \
  --from-literal=UserID="$MYSQL_USER" \
  --from-literal=Password="$MYSQL_PASS" \
  --from-literal=Database="t9_2022630278_bdgu" \
  -n t9-2022630278
secret/mysql-credentials-gu created
gustavo [ ~ ]$ |
```



```
A different object named t9-2022630278-aks already exists in contexts
 in your kubeconfig file.
gustavo [ ~ ]$ kubectl create secret generic mysql-credentials-gu \
  --from-literal=Server="$MYSQL_HOST" \
  --from-literal=UserID="$MYSQL_USER" \
  --from-literal=Password="$MYSQL_PASS" \
  --from-literal=Database="t9_2022630278_bdgu" \
  -n t9-2022630278
secret/mysql-credentials-gu created
gustavo [ ~ ]$ kubectl create secret generic mysql-credentials-ga \
  --from-literal=Server="$MYSQL_HOST" \
  --from-literal=UserID="$MYSQL_USER" \
  --from-literal=Password="$MYSQL_PASS" \
  --from-literal=Database="t9_2022630278_bdga" \
  -n t9-2022630278
secret/mysql-credentials-ga created
gustavo [ ~ ]$
```

Figura 10.8 Creación de secretos para gu

Para GC:

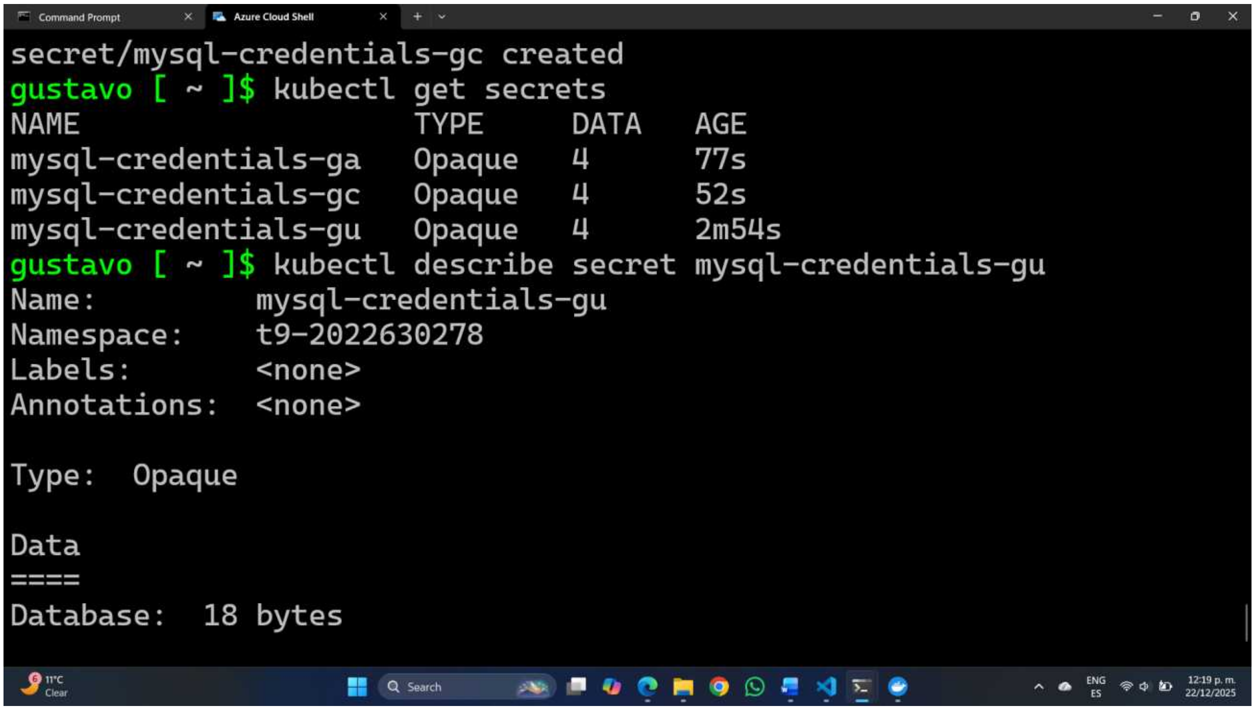


```
  -n t9-2022630278
secret/mysql-credentials-gu created
gustavo [ ~ ]$ kubectl create secret generic mysql-credentials-ga \
  --from-literal=Server="$MYSQL_HOST" \
  --from-literal=UserID="$MYSQL_USER" \
  --from-literal=Password="$MYSQL_PASS" \
  --from-literal=Database="t9_2022630278_bdga" \
  -n t9-2022630278
secret/mysql-credentials-ga created
gustavo [ ~ ]$ kubectl create secret generic mysql-credentials-gc \
  --from-literal=Server="$MYSQL_HOST" \
  --from-literal=UserID="$MYSQL_USER" \
  --from-literal=Password="$MYSQL_PASS" \
  --from-literal=Database="t9_2022630278_bdgc" \
  -n t9-2022630278
secret/mysql-credentials-gc created
gustavo [ ~ ]$
```

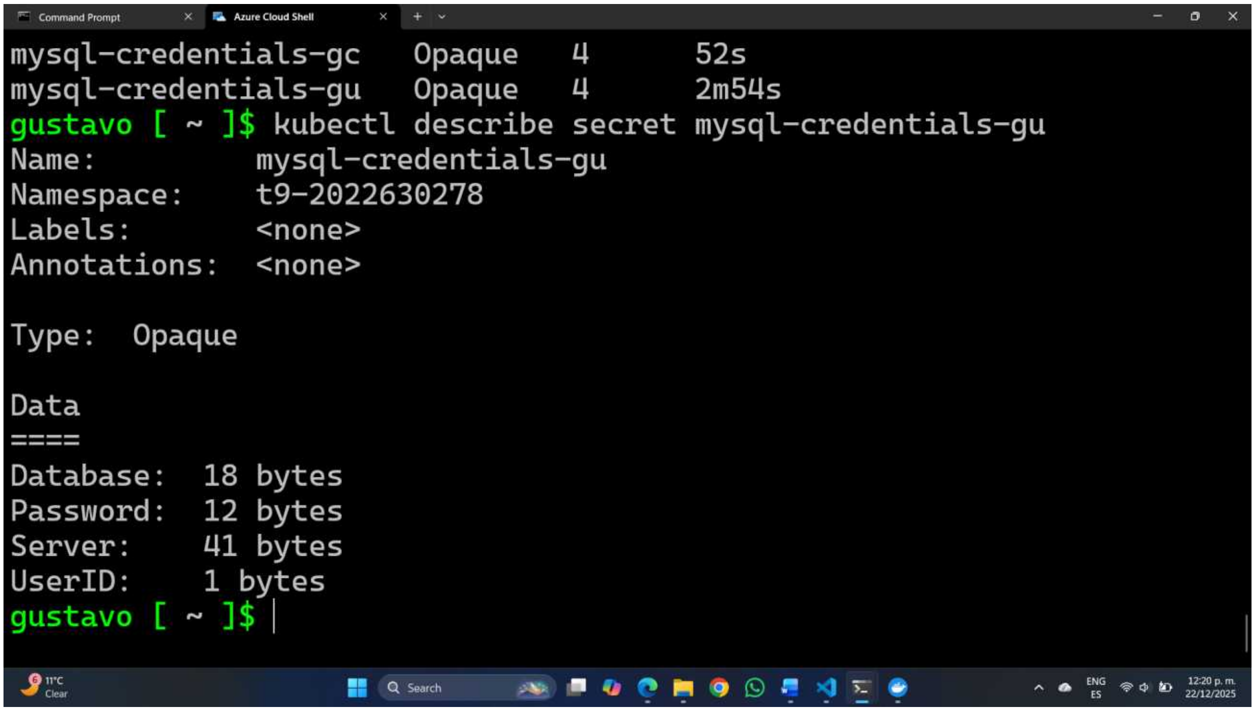
Figura 10.9 Creación de secretos para gc

Verifica que existen:

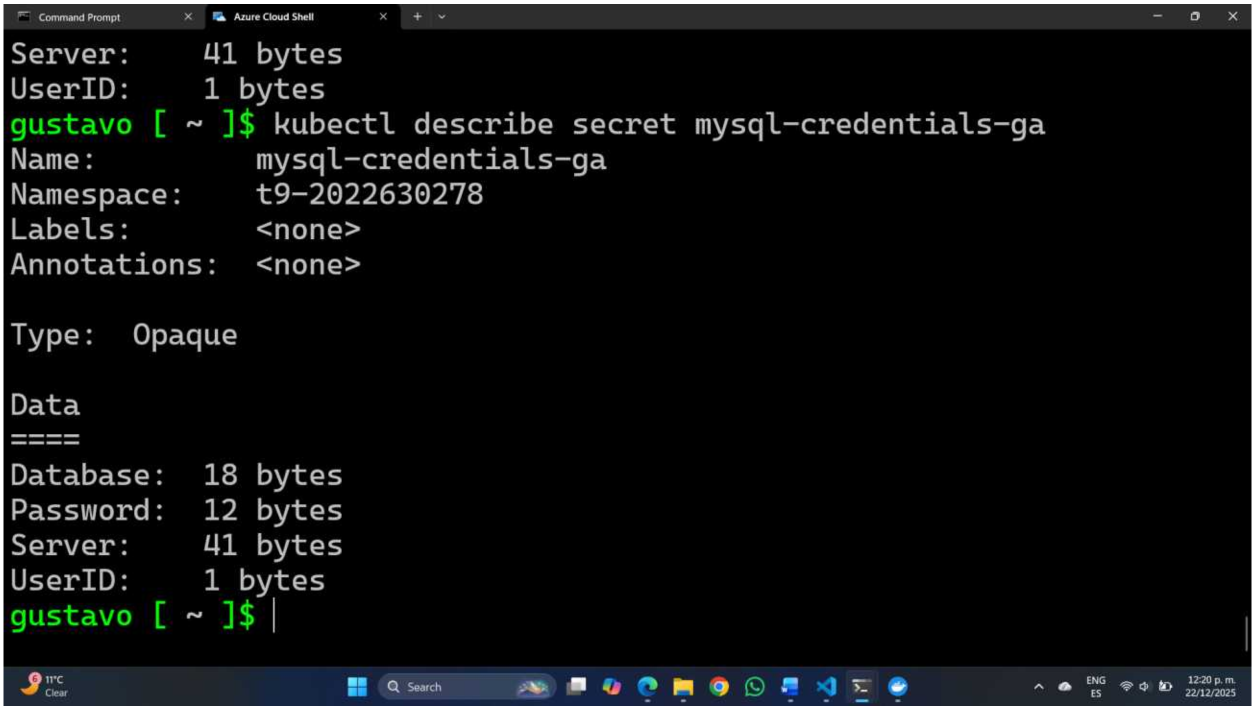
- kubectl get secrets
- kubectl describe secret mysql-credentials-gu
- kubectl describe secret mysql-credentials-ga
- kubectl describe secret mysql-credentials-gc



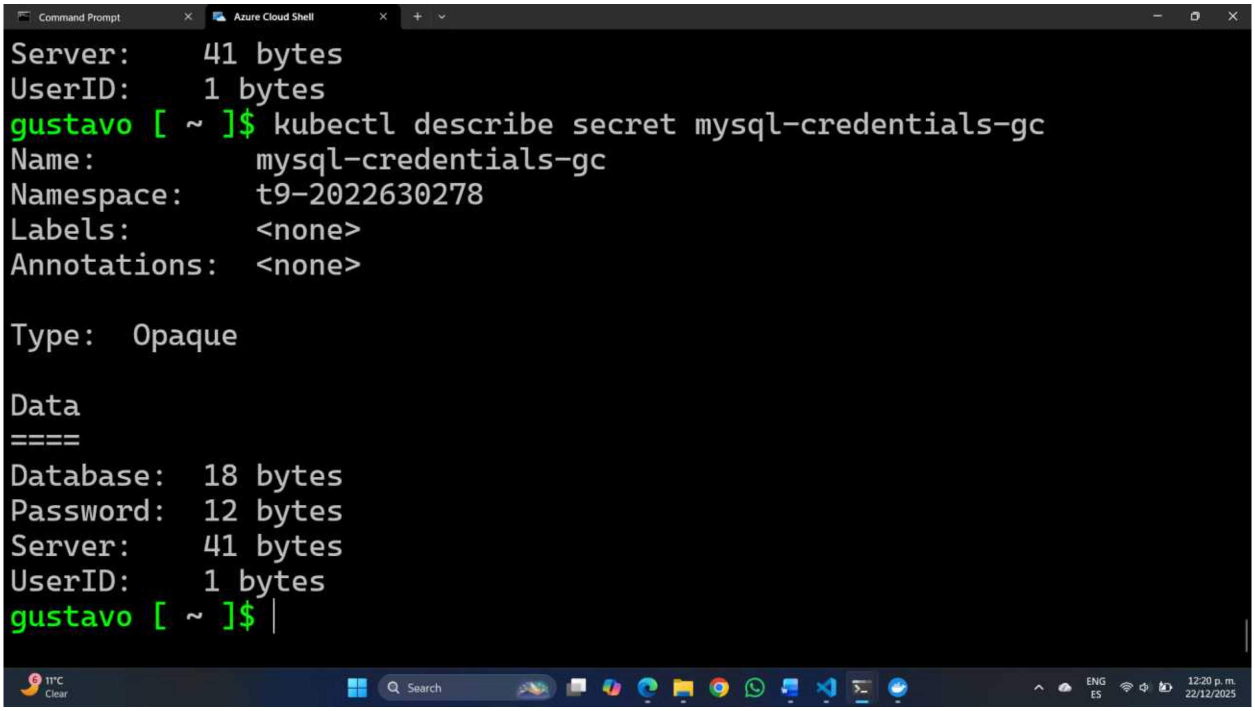
```
secret/mysql-credentials-gc created
gustavo [ ~ ]$ kubectl get secrets
NAME                    TYPE      DATA   AGE
mysql-credentials-ga    Opaque    4      77s
mysql-credentials-gc    Opaque    4      52s
mysql-credentials-gu    Opaque    4      2m54s
gustavo [ ~ ]$ kubectl describe secret mysql-credentials-gu
Name:         mysql-credentials-gu
Namespace:    t9-2022630278
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
Database:  18 bytes
```



```
mysql-credentials-gc    Opaque    4      52s
mysql-credentials-gu    Opaque    4      2m54s
gustavo [ ~ ]$ kubectl describe secret mysql-credentials-gu
Name:         mysql-credentials-gu
Namespace:    t9-2022630278
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
Database:  18 bytes
Password:  12 bytes
Server:    41 bytes
UserID:    1 bytes
gustavo [ ~ ]$
```



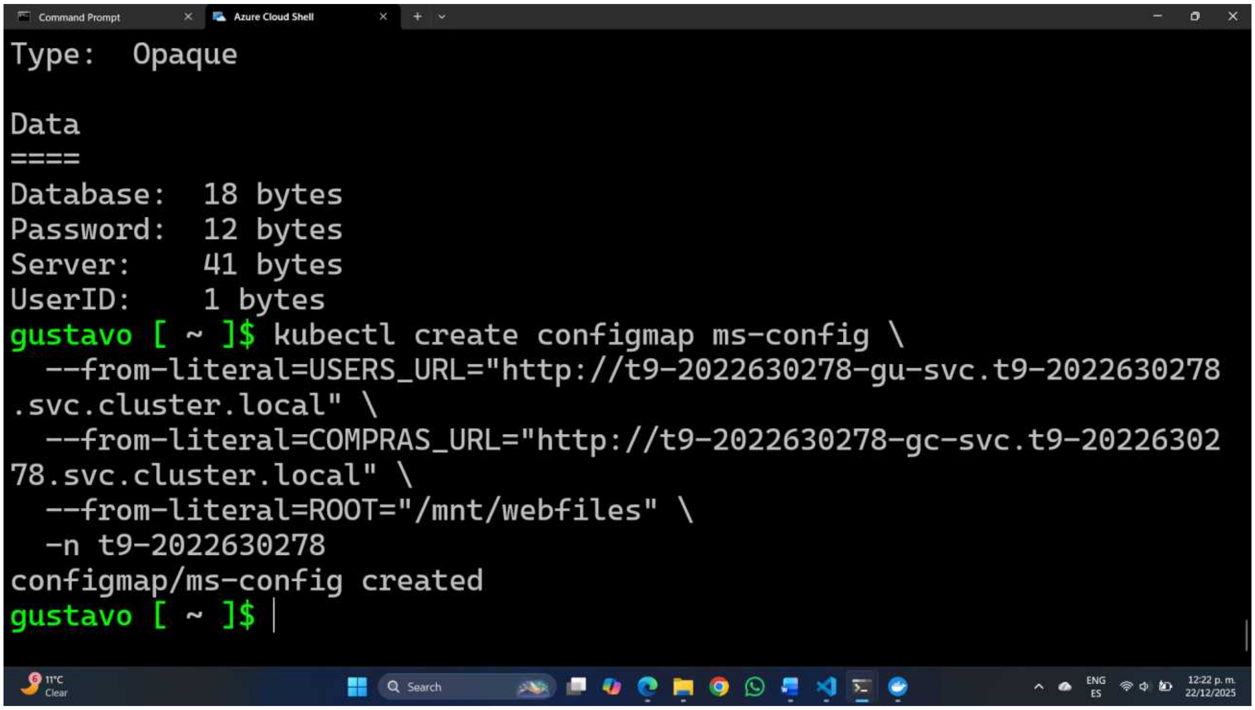
```
Server:     41 bytes
UserID:     1 bytes
gustavo [ ~ ]$ kubectl describe secret mysql-credentials-ga
Name:         mysql-credentials-ga
Namespace:    t9-2022630278
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
Database:  18 bytes
Password:  12 bytes
Server:    41 bytes
UserID:    1 bytes
gustavo [ ~ ]$
```



```
Server:     41 bytes
UserID:     1 bytes
gustavo [ ~ ]$ kubectl describe secret mysql-credentials-gc
Name:         mysql-credentials-gc
Namespace:    t9-2022630278
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
Database:  18 bytes
Password:  12 bytes
Server:    41 bytes
UserID:    1 bytes
gustavo [ ~ ]$
```

Figura 10.9 Verificar secretos

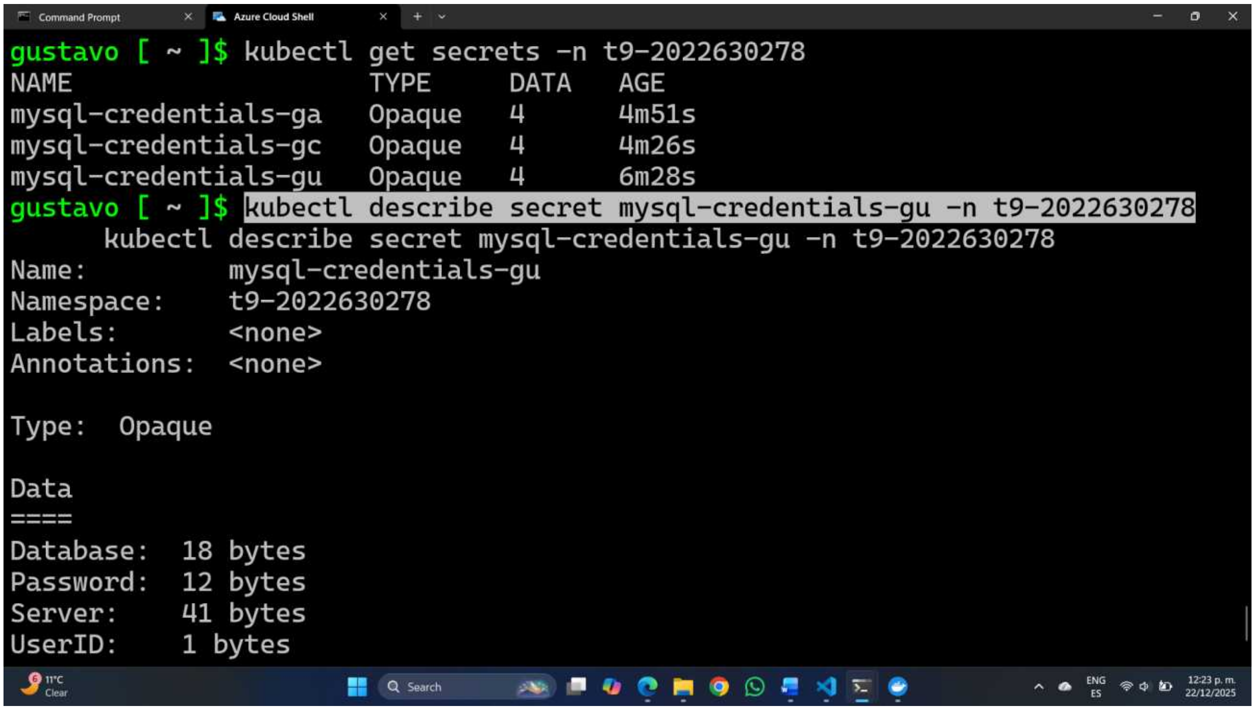
- Se instaló el ConfigMap ms-config con USERS_URL, COMPRAS_URL y ROOT.



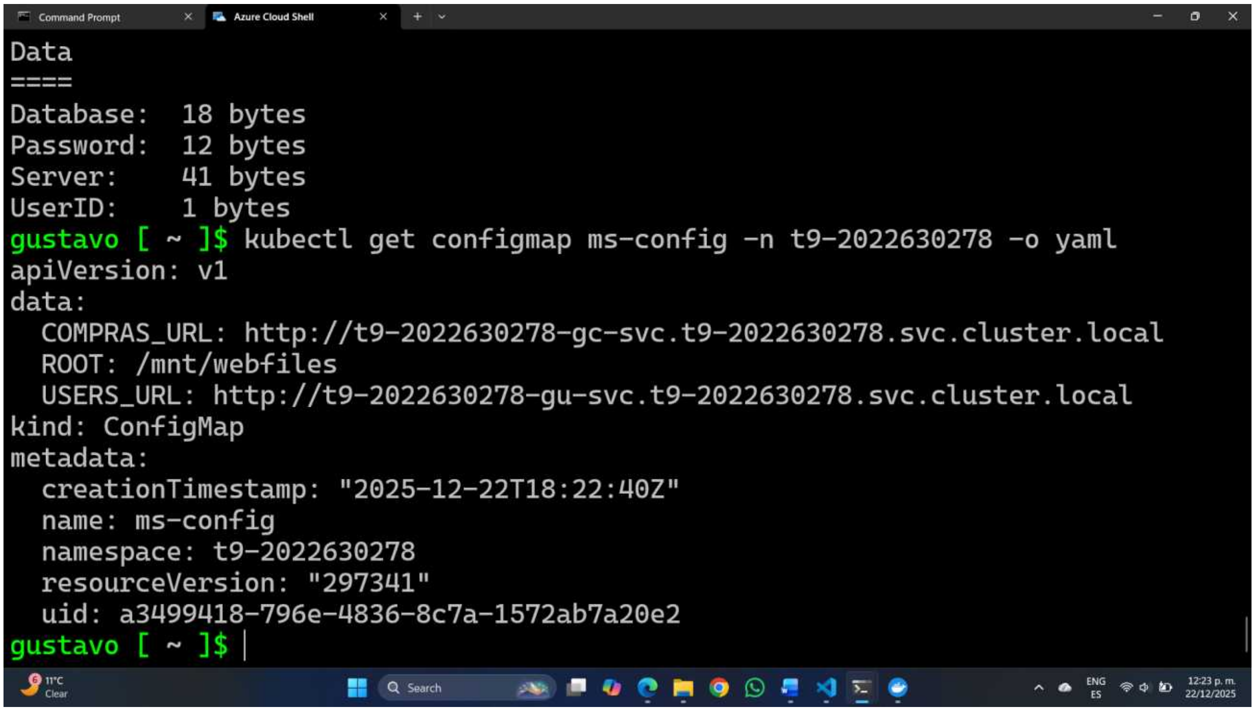
```
Type:  Opaque

Data
====
Database:  18 bytes
Password:  12 bytes
Server:    41 bytes
UserID:    1 bytes
gustavo [ ~ ]$ kubectl create configmap ms-config \
  --from-literal=USERS_URL="http://t9-2022630278-gu-svc.t9-2022630278
.svc.cluster.local" \
  --from-literal=COMPRAS_URL="http://t9-2022630278-gc-svc.t9-20226302
78.svc.cluster.local" \
  --from-literal=ROOT="/mnt/webfiles" \
  -n t9-2022630278
configmap/ms-config created
gustavo [ ~ ]$ 
```

Figura 10.10 Creación de secretos para urls

- Se accedió a la sección de configuración de cada Deployment para validar envFrom y env apuntando a Secrets y ConfigMaps.



```
gustavo [ ~ ]$ kubectl get secrets -n t9-2022630278
NAME                   TYPE     DATA   AGE
mysql-credentials-ga   Opaque   4      4m51s
mysql-credentials-gc   Opaque   4      4m26s
mysql-credentials-gu   Opaque   4      6m28s
gustavo [ ~ ]$ kubectl describe secret mysql-credentials-gu -n t9-2022630278
      kubectl describe secret mysql-credentials-gu -n t9-2022630278
Name:         mysql-credentials-gu
Namespace:    t9-2022630278
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
Database:  18 bytes
Password:  12 bytes
Server:    41 bytes
UserID:    1 bytes
```

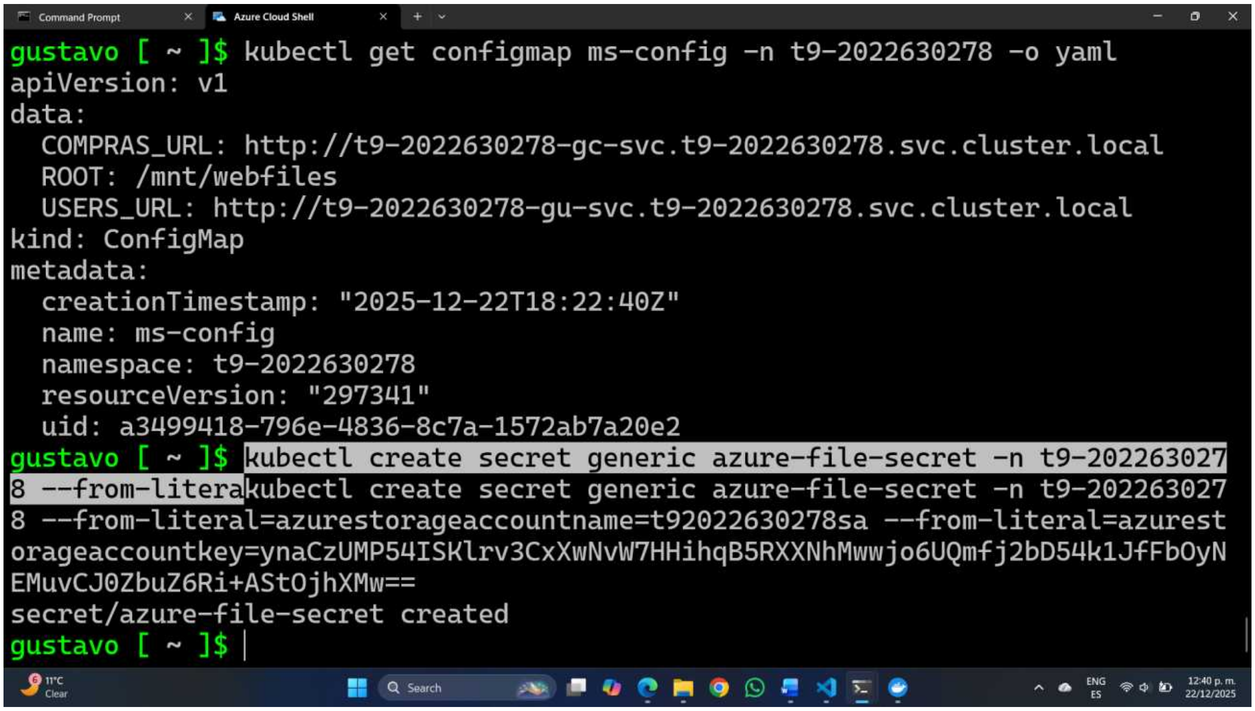


```
Data
====
Database:  18 bytes
Password:  12 bytes
Server:    41 bytes
UserID:    1 bytes
gustavo [ ~ ]$ kubectl get configmap ms-config -n t9-2022630278 -o yaml
apiVersion: v1
data:
  COMPRAS_URL: http://t9-2022630278-gc-svc.t9-2022630278.svc.cluster.local
  ROOT: /mnt/webfiles
  USERS_URL: http://t9-2022630278-gu-svc.t9-2022630278.svc.cluster.local
kind: ConfigMap
metadata:
  creationTimestamp: "2025-12-22T18:22:40Z"
  name: ms-config
  namespace: t9-2022630278
  resourceVersion: "297341"
  uid: a3499418-796e-4836-8c7a-1572ab7a20e2
gustavo [ ~ ]$
```

Figura 10.10 ConfigMaps y Secrets del namespace del proyecto con variables de entorno aplicadas

8.1 Crear PV y PVC para Azure Files (front-end)

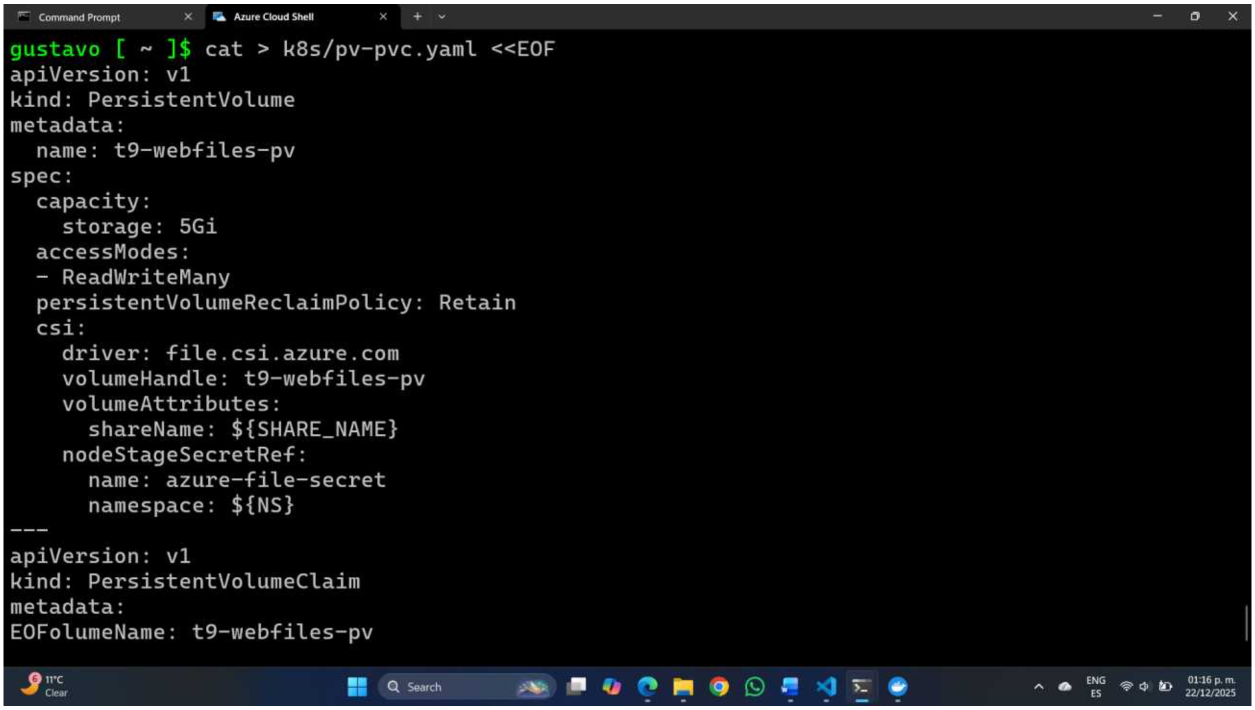
- Se realizaron los Portal > Storage account > t92022630278sa > Access keys > copia la key1
- Cloud Shell:
    - kubectl create secret generic azure-file-secret -n t9-2022630278 --from-literal=azurestorageaccountname=t92022630278sa --from-literal=azurestorageaccountkey=<KEY1>



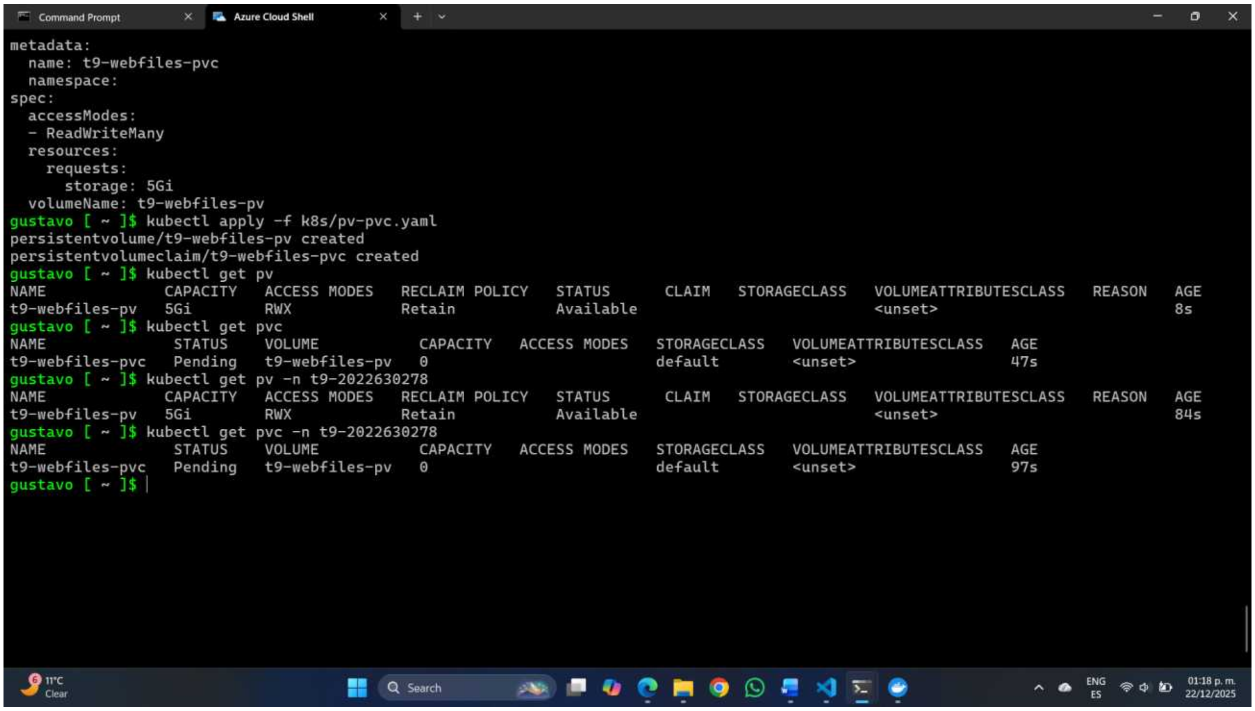
```
gustavo [ ~ ]$ kubectl get configmap ms-config -n t9-2022630278 -o yaml
apiVersion: v1
data:
  COMPRAS_URL: http://t9-2022630278-gc-svc.t9-2022630278.svc.cluster.local
  ROOT: /mnt/webfiles
  USERS_URL: http://t9-2022630278-gu-svc.t9-2022630278.svc.cluster.local
kind: ConfigMap
metadata:
  creationTimestamp: "2025-12-22T18:22:40Z"
  name: ms-config
  namespace: t9-2022630278
  resourceVersion: "297341"
  uid: a3499418-796e-4836-8c7a-1572ab7a20e2
gustavo [ ~ ]$ kubectl create secret generic azure-file-secret -n t9-202263027
8 --from-literakubectl create secret generic azure-file-secret -n t9-202263027
8 --from-literal=azurestorageaccountname=t92022630278sa --from-literal=azurest
orageaccountkey=ynaCzUMP54ISKlrv3CxXwNvW7HHihqB5RXXNhMwwjo6UQmfj2bD54k1JfFbOyN
EMuvCJ0ZbuZ6Ri+AStOjhXMw==
secret/azure-file-secret created
gustavo [ ~ ]$ |
```

Figura 10.11 Creación del secreto para la llave

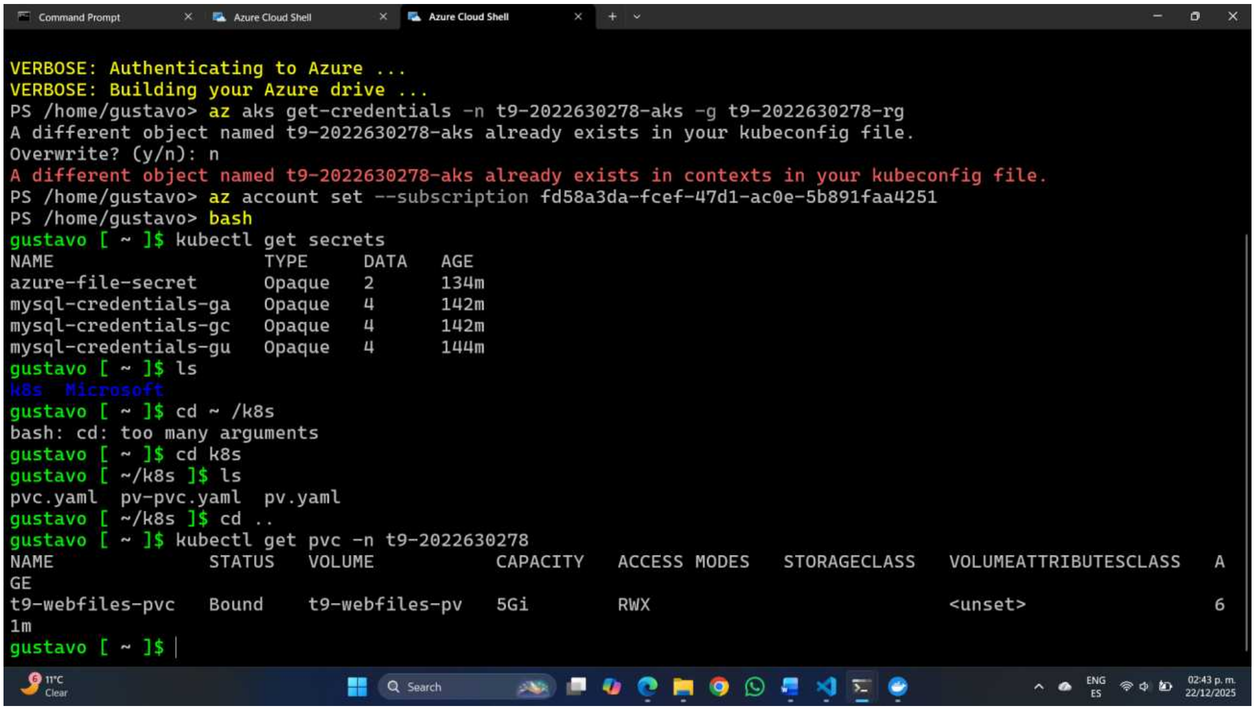
2. Aplica PV/PVC:



```
gustavo [ ~ ]$ cat > k8s/pv-pvc.yaml <<EOF
apiVersion: v1
kind: PersistentVolume
metadata:
  name: t9-webfiles-pv
spec:
  capacity:
    storage: 5Gi
  accessModes:
  - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  csi:
    driver: file.csi.azure.com
    volumeHandle: t9-webfiles-pv
    volumeAttributes:
      shareName: ${SHARE_NAME}
    nodeStageSecretRef:
      name: azure-file-secret
      namespace: ${NS}
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
EOFolumeName: t9-webfiles-pv
```



```
metadata:
  name: t9-webfiles-pvc
  namespace:
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 5Gi
  volumeName: t9-webfiles-pv
gustavo [ ~ ]$ kubectl apply -f k8s/pv-pvc.yaml
persistentvolume/t9-webfiles-pv created
persistentvolumeclaim/t9-webfiles-pvc created
gustavo [ ~ ]$ kubectl get pv
NAME             CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   VOLUMEATTRIBUTESCLASS   REASON   AGE
t9-webfiles-pv   5Gi        RWX            Retain           Available                          <unset>                          8s
gustavo [ ~ ]$ kubectl get pvc
NAME              STATUS    VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
t9-webfiles-pvc   Pending   t9-webfiles-pv   0                         default        <unset>                 47s
gustavo [ ~ ]$ kubectl get pv -n t9-2022630278
NAME             CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   VOLUMEATTRIBUTESCLASS   REASON   AGE
t9-webfiles-pv   5Gi        RWX            Retain           Available                          <unset>                          84s
gustavo [ ~ ]$ kubectl get pvc -n t9-2022630278
NAME              STATUS    VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
t9-webfiles-pvc   Pending   t9-webfiles-pv   0                         default        <unset>                 97s
gustavo [ ~ ]$ 
```



```
VERBOSE: Authenticating to Azure ...
VERBOSE: Building your Azure drive ...
PS /home/gustavo> az aks get-credentials -n t9-2022630278-aks -g t9-2022630278-rg
A different object named t9-2022630278-aks already exists in your kubeconfig file.
Overwrite? (y/n): n
A different object named t9-2022630278-aks already exists in contexts in your kubeconfig file.
PS /home/gustavo> az account set --subscription fd58a3da-fcef-47d1-ac0e-5b891faa4251
PS /home/gustavo> bash
gustavo [ ~ ]$ kubectl get secrets
NAME                   TYPE     DATA   AGE
azure-file-secret      Opaque   2      134m
mysql-credentials-ga   Opaque   4      142m
mysql-credentials-gc   Opaque   4      142m
mysql-credentials-gu   Opaque   4      144m
gustavo [ ~ ]$ ls
k8s  Microsoft
gustavo [ ~ ]$ cd ~ /k8s
bash: cd: too many arguments
gustavo [ ~ ]$ cd k8s
gustavo [ ~/k8s ]$ ls
pvc.yaml  pv-pvc.yaml  pv.yaml
gustavo [ ~/k8s ]$ cd ..
gustavo [ ~ ]$ kubectl get pvc -n t9-2022630278
NAME               STATUS   VOLUME            CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   A
GE
t9-webfiles-pvc    Bound    t9-webfiles-pv    5Gi        RWX                           <unset>                 6
1m
gustavo [ ~ ]$
```

Figura 10.12 virificacion de estatus de los yaml

4. Montar el PVC en el microservicio “Servidor web”

- En el Deployment del servicio web (t9-2022630278-sw), monta el PVC y usa ROOT=/mnt/webfiles. Si ya tienes tu manifiesto, revisa que tenga esto: