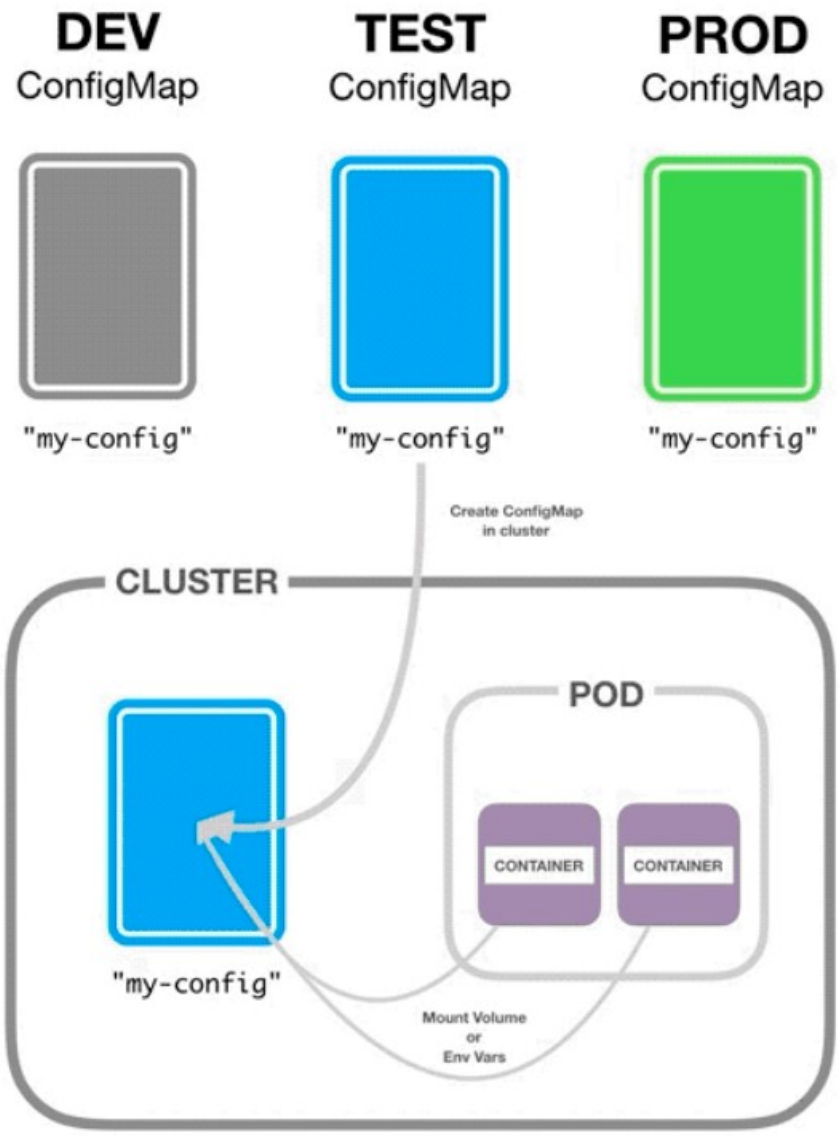
ConfigMaps and Secrets

What are ConfigMaps?

A ConfigMap is an API object used to store non-confidential data in key-value pairs. How do you manage your application's configuration? For a Python or Node.js application, where do you store the configuration? ConfigMaps allow us to make a single change to the file itself, which can then effect many different pods using the same ConfigMap, thus cutting down on user error and deployment times.



Kubernetes pods can use ConfigMaps as configuration files, environment variables or command-line arguments. ConfigMaps allow you to decouple environment-specific configurations from containers to make applications portable. Notice in the picture below we're

creating a ConfigMap for a video game. The player lives, types of enemies, and even colors are defined and would be applied to the application running on any pods using this particular ConfigMap.

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: game-demo
data:
  # property-like keys; each key maps to a simple value
  player_initial_lives: "3"
  ui_properties_file_name: "user-interface.properties"

  # file-like keys
  game.properties: |
    enemy.types=aliens,monsters
    player.maximum-lives=5
  user-interface.properties: |
    color.good=purple
    color.bad=yellow
    allow.textmode=true
```

What are Secrets?

Secrets are a Kubernetes object intended for storing a small amount of sensitive data. To create a Secret ad-hoc choose a password and convert it to base64.

This is an imperative method to create a Secret, followed be the declarative YAML file. The command `echo -n 'KubernetesRocks!' | base64` is taking the string and converting it into a hashed state.

```
$ echo -n 'KubernetesRocks!' | base64
S3ViZXJuZXRlc1JvY2tzIQ==
```

We would then save that hash within our Secrets.yaml file, but keep in mind this is not how we will create Secrets in our lab.

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: mariadb-root-password
type: Opaque
data:
  password: S3ViZXJuZXRlc1JvY2tzIQ==
```

Storing sensitive data in Secrets is more secure than in plain text ConfigMaps or in Pod specifications. Using Secrets gives you control over how sensitive data is used, and reduces the risk of exposing the data to unauthorized users.

You can also encrypt Secrets at the application layer using a key you manage in Cloud KMS.

Hands On - Create a ConfigMap and Secret

I. Create a ConfigMap and a pod that will utilize it.

```
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ kubectl create configmap doms-configmap --f
rom-literal=owner=dom
configmap/doms-configmap created
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ kubectl get configmap doms-configmap
NAME            DATA   AGE
doms-configmap   1      25s
```

II. Create a Secret for DB and a pod using the username and password.