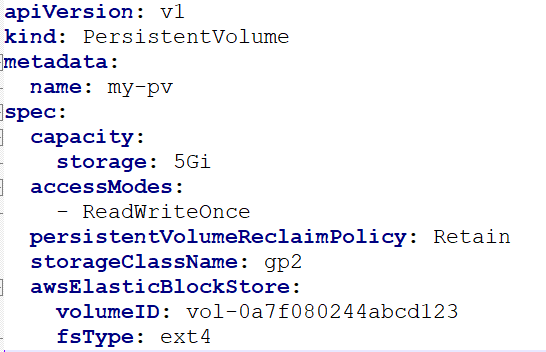
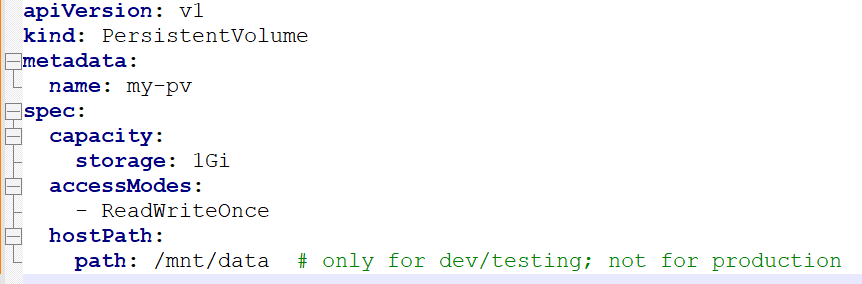
**Persistent Volumes (PV) and Persistent Volume Claims (PVC) Overview**

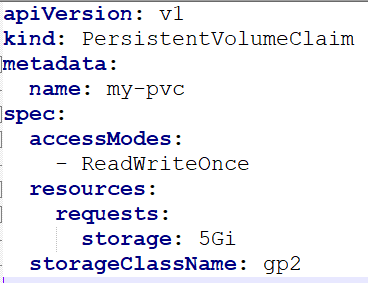
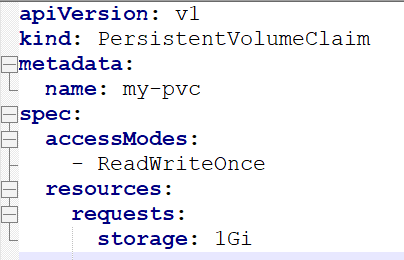
**1. Persistent Volumes (PV):**

* **PV** is a **piece of storage** in your cluster that has been provisioned by an **administrator** or dynamically via a **StorageClass**.
* It’s a resource in the **cluster** just like a Pod is a resource. However, **PV is independent** of the lifecycle of any specific Pod.
* **PV represents actual physical storage** (can be an AWS EBS, GCE Persistent Disk, etc.).
* **PV can be pre-provisioned** (manually by an administrator) or **dynamically provisioned** by Kubernetes based on a **StorageClass**.

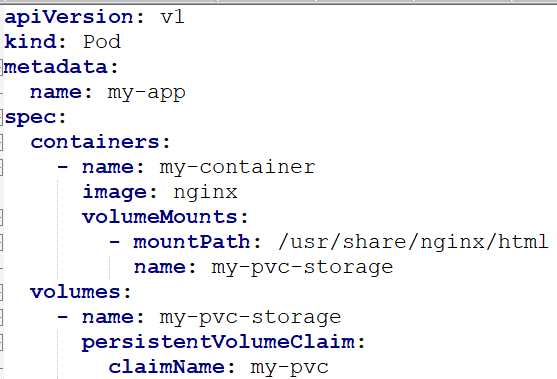
**2. Persistent Volume Claims (PVC):**

* A **PVC** is a **request** for storage by a **user/Pod**.
* A PVC specifies the **size**, **access modes**, and **storage class**. Once a PVC is created, Kubernetes tries to match it with an existing PV or dynamically provisions one if no matching PV is found.

**How PV and PVC Work Together:**

1. **Administrator** creates a **PV** that specifies the storage backend, access modes, and capacity.
2. **User** creates a **PVC** requesting storage (specifying size and access modes).
3. **Kubernetes** checks if there’s a **matching PV**. If yes, it **binds** the PVC to that PV, making the storage available to the Pod.
4. If no matching PV is found, Kubernetes may dynamically create a PV (**if configured with a StorageClass**).
5. Once bound, the PV and PVC relationship is **one-to-one**, meaning a PVC can only be bound to a single PV.



**Important Concepts to Understand:**

**1. Access Modes:**

Access modes define how the storage can be mounted:

* **ReadWriteOnce (RWO):** A volume can be mounted as read-write by **only one node** at a time.
* **ReadOnlyMany (ROM):** A volume can be mounted as read-only by **many nodes**.
* **ReadWriteMany (RWM):** A volume can be mounted as read-write by **many nodes**.

**Use case:**

* **RWO** for applications where the volume should be used by only one Pod (e.g., a **database**).
* **RWX** for scenarios where the volume is shared among multiple Pods (e.g., a **shared file system**).

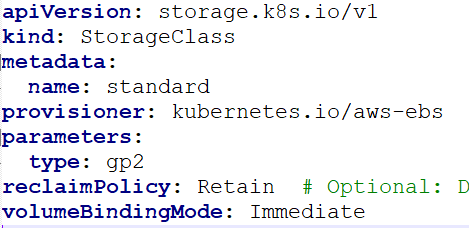
**2. Reclaim Policies:**

The **Reclaim Policy** determines what happens to the PV after the PVC is deleted.

* **Retain:** The PV is **not deleted** when the PVC is deleted. It remains available in the cluster but will require manual cleanup.
* **Delete:** The PV is **deleted** when the PVC is deleted. This works well with dynamically provisioned volumes like those backed by cloud providers.
* **Recycle (deprecated):** The PV’s contents are deleted and it is made available for another PVC (this policy is deprecated).

**3. StorageClass:**

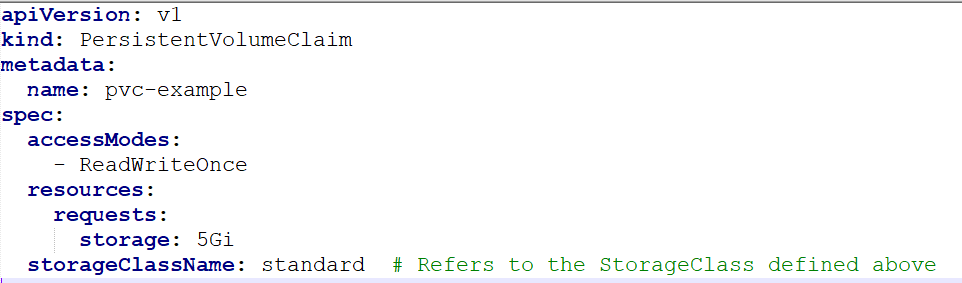
* A **StorageClass** defines the **types of storage** available in the cluster.
* It allows dynamic provisioning of PVs when a PVC is created.
* Common cloud-backed classes might be: gp2 (AWS), standard (GCE), etc.

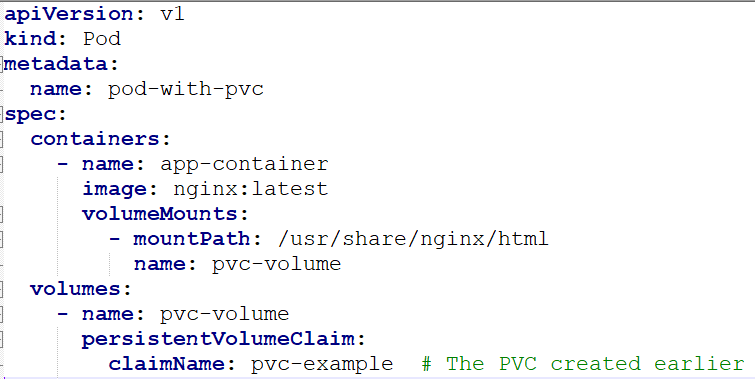


* **provisioner**: The provisioner that creates the storage (e.g., AWS EBS, GCE Persistent Disk, etc.).
* **parameters**: Specific settings for the storage (e.g., the type of disk on AWS EBS).
* **reclaimPolicy**: Defines the reclaim policy when the PVC is deleted (can be Retain or Delete).
* **volumeBindingMode**: Controls when the volume is bound (e.g., Immediate or WaitForFirstConsumer).

**4. Dynamic Provisioning:**

* If a **StorageClass** is defined and referenced in a PVC, Kubernetes will **dynamically provision** a PV that meets the requirements of the PVC (size, access mode, etc.).
* This is most commonly used with cloud providers like AWS, GCP, Azure, where storage is dynamically allocated.





**What Happens During Dynamic Provisioning:**

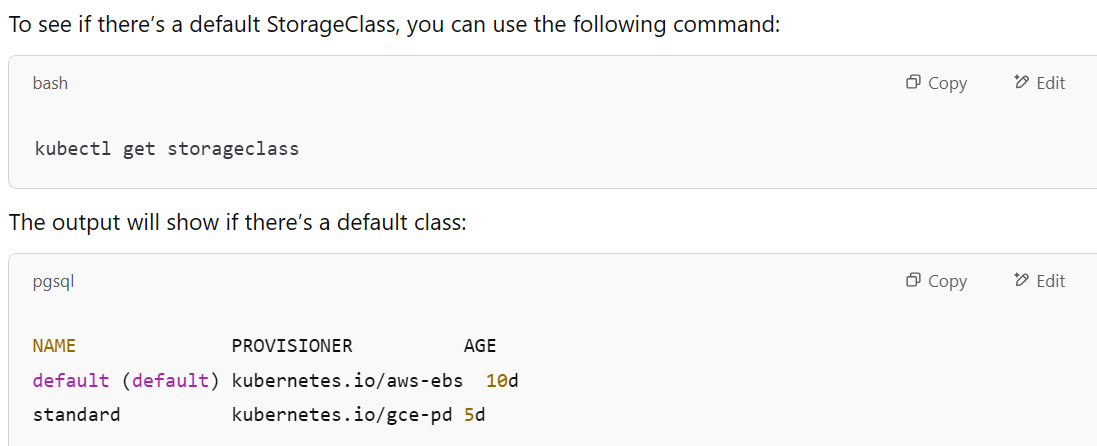
* Kubernetes will look at the **PVC** to see if it references a **StorageClass**.
* If the **StorageClass** is defined, Kubernetes will ask the **StorageClass provisioner** (e.g., AWS EBS, GCE Persistent Disk, etc.) to dynamically create a **Persistent Volume (PV)** that satisfies the request in the PVC (e.g., a 5Gi volume with ReadWriteOnce access mode).
* Once the PV is created, it gets **automatically bound** to the PVC.
* The **Pod** can now mount the volume defined by the PVC.

**Key Use Cases of PV and PVC:**

1. **Database Storage:**  
   For applications like **MySQL** or **PostgreSQL**, where the data must persist independently of Pods.
   * PVC can request persistent storage.
   * PV can be bound to a StatefulSet or Deployment to ensure data persists after Pods are deleted or rescheduled.
2. **Shared Storage for Multiple Pods:**  
   If you have multiple Pods requiring access to the same volume, you can use **ReadWriteMany (RWX)** access mode to share data between them.
   * A **distributed file system** or shared cache, where multiple Pods need access to the same storage.
3. **Dynamic Volume Provisioning:**  
   For cloud-based applications, PVs can be dynamically provisioned with a **StorageClass** (e.g., gp2 in AWS).
   * Automatically provision storage when new PVCs are created, without manual intervention.

### ****Default StorageClass:****

If no storageClassName is specified in the PVC, Kubernetes will use the **default StorageClass** if one is defined. If there is no default **StorageClass**, dynamic provisioning will not work, and you’ll receive an error.



**Quick Summary:**

* **PV** is a physical storage resource in the cluster.
* **PVC** is a request for storage, and **binds** to a PV.
* **Access Modes** determine how volumes can be accessed (by one or multiple Pods).
* **Reclaim Policies** determine the fate of PVs when PVCs are deleted.
* **StorageClasses** manage dynamic provisioning of PVs.

**Quick Summary of Access Modes and Their Supported Volumes:**

| **Access Mode** | **Volume Type** | **Use Case** |
| --- | --- | --- |
| **ReadWriteOnce (RWO)** | **EBS, GCE Persistent Disk, Azure Disk, hostPath, emptyDir** | Storage tied to a specific node or pod (single writer). |
| **ReadOnlyMany (ROM)** | **Network File System, Amazon EFS, Azure Files** | Shared read-only access across multiple pods/nodes. |
| **ReadWriteMany (RWX)** | **NFS, Amazon EFS, Google Cloud Filestore, Azure Files** | Shared read-write access across multiple pods/nodes. |