**Attach a Volume to the Pod – Go to pods directory**

Lets create a pod for database and attach a volume to it. To achieve this we will need to

* create a **volumes** definition
* attach volume to container using **VolumeMounts** property

Local host volumes are of two types:  
\* emptyDir  
\* hostPath

We will pick hostPath. [Refer to this doc to read more about hostPath.](https://kubernetes.io/docs/concepts/storage/volumes/#hostpath)

File: db-pod.yaml

1. **apiVersion: v1**
2. **kind: Pod**
3. **metadata:**
4. **name: db**
5. **labels:**
6. **app: postgres**
7. **role: database**
8. **tier: back**
9. **spec:**
10. **containers:**
11. **- name: db**
12. **image: postgres:9.4**
13. **ports:**
14. **- containerPort: 5432**
15. **volumeMounts:**
16. **- name: db-data**
17. **mountPath: /var/lib/postgresql/data**
18. **volumes:**
19. **- name: db-data**
20. **hostPath:**
21. **path: /var/lib/pgdata**
22. **type: DirectoryOrCreate**

**apiVersion: v1kind: Podmetadata: name: db labels: app: postgres role: database tier: backspec: containers: - name: db image: postgres:9.4 ports: - containerPort: 5432 volumeMounts: - name: db-data mountPath: /var/lib/postgresql/data env: - name: POSTGRES\_PASSWORD value: test123 volumes: - name: db-data hostPath: path: /var/lib/pgdata type: DirectoryOrCreate**

To create this pod,

1. kubectl apply -f db-pod.yaml
2. kubectl get pods –o wide
4. kubectl describe pod db
6. kubectl get events

**Exercise** : Examine **/var/lib/pgdata** on the systems to check if the directory is been created and if the data is present.