A close up of a logo

Description automatically generated

**Lesson 4 Demo 5**

**Understanding init Containers**



Steps to be followed:

1. Adding init container entries to the pod configuration

**Step 1: Adding init container entries to the pod configuration**

1. Create a pod yaml file like below:

***vi demofile.yaml***

1. Add the following lines of code to the file:

***apiVersion: v1***

***kind: Pod***

***metadata:***

***name: node-server***

***labels:***

***app: myapp***

***spec:***

***containers:***

***- name: node-server***

***image: jjino/node-redis:v1.0***

***ports:***

***- containerPort: 8080***

***initContainers:***

***- name: init-myservice***

***image: jjino/test:latest***

***args:***

***- bash***

***- -c***

***- while true; do STATUS=$(redis-cli -h redis-service -p 6379 ping); if [ $STATUS = "PONG" ]; then echo "Connected"; break; else echo "Not connected"; fi; sleep 10; done***

***---***

***apiVersion: v1***

***kind: Service***

***metadata:***

***name: myservice***

***spec:***

***selector:***

***app: myapp***

***ports:***

***- protocol: TCP***

***port: 8080***

***type: NodePort***

**init Containers** is the new section where init containers are configured  ******

| **Note:** Press the **esc** button and type **:wq** to save and exit the file. |
| --- |

1. Execute this command to configure your yaml file:

***kubectl apply -f demofile.yaml***



