Kubernetes Pod Basics - Exercises

# Exercise 1: Create a Pod Running Python HTTP Server

🧠 Question:

Create a Pod named `python-web` that runs a container from the `python:3.11-slim` image. Use it to serve an HTTP server on port 8000 using `python -m http.server`.

📄 YAML Definition:

apiVersion: v1  
kind: Pod  
metadata:  
 name: python-web  
spec:  
 containers:  
 - name: python-container  
 image: python:3.11-slim  
 command: ["python", "-m", "http.server", "8000"]  
 ports:  
 - containerPort: 8000

💻 Commands:

* kubectl apply -f python-web.yaml
* kubectl get pods
* kubectl port-forward pod/python-web 8000:8000

# Exercise 2: View Logs from a Pod Running Node.js

🧠 Question:

Create a Pod named `node-logger` that uses the `node:20-alpine` image. It should print "Hello from Node.js" and then sleep. View the log output.

📄 YAML Definition:

apiVersion: v1  
kind: Pod  
metadata:  
 name: node-logger  
spec:  
 containers:  
 - name: node-container  
 image: node:20-alpine  
 command: ["sh", "-c", "echo 'Hello from Node.js' && sleep 3600"]

💻 Commands:

* kubectl apply -f node-logger.yaml
* kubectl logs node-logger

# Exercise 3: Run a Shell Command in a Debian Pod

🧠 Question:

Create a Pod named `debian-tools` using the `debian:bullseye` image. After it's running, exec into the pod and list files in `/etc`.

📄 YAML Definition:

apiVersion: v1  
kind: Pod  
metadata:  
 name: debian-tools  
spec:  
 containers:  
 - name: debian-container  
 image: debian:bullseye  
 command: ["sleep", "3600"]

💻 Commands:

* kubectl apply -f debian-tools.yaml
* kubectl exec -it debian-tools -- ls /etc

# Exercise 4: Pod with Environment Variable Using `alpine`

🧠 Question:

Create a Pod named `env-checker` using the `alpine:3.20` image. Set an environment variable `DEMO=K8sRocks`, and print it using `echo $DEMO`.

📄 YAML Definition:

apiVersion: v1  
kind: Pod  
metadata:  
 name: env-checker  
spec:  
 containers:  
 - name: alpine-container  
 image: alpine:3.20  
 command: ["sh", "-c", "echo $DEMO && sleep 3600"]  
 env:  
 - name: DEMO  
 value: "K8sRocks"

💻 Commands:

* kubectl apply -f env-checker.yaml
* kubectl logs env-checker

# Exercise 5: CrashLoopBackOff Pod Using `ubuntu`

🧠 Question:

Create a Pod named `failer` using the `ubuntu:22.04` image that immediately fails (exits with code 1) to demonstrate the `CrashLoopBackOff` status.

📄 YAML Definition:

apiVersion: v1  
kind: Pod  
metadata:  
 name: failer  
spec:  
 containers:  
 - name: fail-container  
 image: ubuntu:22.04  
 command: ["bash", "-c", "exit 1"]

💻 Commands:

* kubectl apply -f failer.yaml
* kubectl get pods
* kubectl describe pod failer