Kubernetes/Openshift Demo

1. Applikation lokal starten und deployen
    a. **Applikation (Springboot) in der IDE zeigen**.
    b. Aus der IDE deployen und …
    c. …mit Browser aufrufen
2. Einstieg in Kubernetes
    a. **kubectl get pods** -> was gibt es für pods
    b. **kubectl get namespace** -> was gibt es für namespaces, alles läuft im *default* Namespace ab
    c. clear
    d. **kubectl apply -f deployment.yml** -> Standardaufruf, wenn man was ändern will.
    e. **kubectl apply -f deployment.yml** -> Nochmals ausführen -. es passiert nix, kein neuer State.
3. **Was passiert nun im Cluster: Pods, Replicaset**?
    a. Es gibt ein Deployment: **kubectl get deployment**
    -> wir sehen das es ein Deploymentprozess gibt und der will einen Container
    -> also holt er ihn und erzeugt den Container
    b. kubectl get pod
    c. kubectl get pods -o wide
    -> Interessant hier die ID, denn die ID gehört der erste Teil zum zugehörigen *Replicaset*.
    d. **kubectl get replicaset**
    -> Viele Befehle aus der Dockerwelt ähnlich vorhanden auf diesem Pod
    e. **kubectl get logs POD-ID** -> Standard out, Best Practice: nur noch dieses Log!
    f. **kubectl exec POD-ID curl http://localhost:8080** -> Das Hello aus der App
    g. **kubectl exec POD-ID curl http://localhost:80891/host** -> Host-Kommando ergibt den POD-Namen !
4. Pod braucht einen Service, damit er von aussen gefunden werden kann:
   kubectl apply-f service.yaml
5. **kubectl describe service SERVICENAME** -> Da kommt wesentlich mehr Info, inkl. IP, gebundene Endpoints
6. Routes: Wie eine reverse proxy setup / wird dann entsprechend gemappt/geleitet:
   kubectl apply -f route.yml
7. (IP Hack: minikube = IP der lokalen Minikube VM) -> Dann mit Browser den Endpoint aufrufen, z.B. mit http://minikube/host oder eben http://minikube-IP/host
8. Jetzt langweilig mit nur einem Pod 😉, replicas 1 -> 3:
   -> Im deploy.xml ändern und dann neu mit **kubectl apply -f deploy.xml**
   ODER
   -> **kubectl edit deployment** -> Ruft einen lokalen Texteditor auf und applied die gespeicherte Version.
   -> **Kein neues Replicaset**, aber es werden neue Pods gestartet, das sieht man hier:
   kubectl get pods
9. Dann mit Browser den Host-Endpoint aufrufen, z.B. mit http://minikube/host
   -> Es ändert der Hostname (Round-Robin mässig)

FRAGEN ????

1. **Health**: deployment mit **livenessProbe** (/actuator/health) und **readinessProbe** (/actuator/health) ergänzen.
   - Status RUNNING eines Pods erst wenn readinessProbe erfolgreich ist.
   - Wenn lifenessProbe fehlgeschlägt, wird der Container neu gestartet mit einem neuen ersetzt!

   kubectl apply -f deployment-health.yml
   -> Neues Deployment - es wird ein NEUES Replicaset erstellt = Rolling Upgrade
   -> Das sieht man auch an der Replicaset ID: das eine RS baut die neuen Container, das andere räumt sie weg...

2. Jetzt möchte ich noch Security und Konfiguration externalisieren:
   kubectl apply -f configmap.yml -> verlangt ROLLE für heikle Zugriffe
   -> In Spring Boot muss die Spring Security Config angepasst werden.

3. Und dann Redeplyoment mit Config und Secret
   kubectl apply -f deploy-with-config.yml
   -> es kommen auch Volumes für Configs und Properties dazu.

-> **startet aber noch nicht**, weil wir ja noch **kein secret deployt** haben (*CreateContainerConfigEror*).

4. Das können wir auch herausfinden...
   kubectl describe pod POD-ID
   -> secret 'management-user' not found.
   -> Dasselbe passiert auch wenn am Pod zu viele minimale Ressource definiert werden.

5. Also schreiben wir unser secret...
   -> Im secret legen wir Environment-Variablen an (für den Management-User mit Password) mit base64...
   kubectl apply -f secret.yml

6. Kubernetes merkt das und deployed den Container, Hurra!

   -> Er macht ein Rolling Upgrade
   -> Dazu auch das dashboard anschauen: **minishift dashboard**

7. Weiter kann man auch mit *prometheus/fluxdb* metriken von aussen einsammeln
   -> Frage, was will man alles monitoren als Entwickler?
   -> kube-prometheus operator, ähnliches auch für Grafana
   -> oder mit Elastic Stack

8. Und wie sieht der Docker Daemon darunter aus?
   minikube ssh -> ssh auf die Maschine
   docker ps -> alle normalen Docker Befehle !