Abschlussprojekt – DevOps Engineering bei Kukuk Technology Future GmbH

Laufzeit: 10.06.2025 – 27.06.2025

Ort: Remote (Infrastruktur durch Dozenten bereitgestellt)

# 🎯 Ausgangslage

Herzlichen Glückwunsch zur erfolgreichen Teilnahme an der Weiterbildung zum/zur DevOps Engineer! Sie treten nun Ihre erste verantwortungsvolle Aufgabe bei der Kukuk Technology Future GmbH an – einem innovativen Unternehmen, das moderne Cloud- und Containertechnologien im Bereich smarter Industrieanwendungen einsetzt.  
  
Sie übernehmen den technischen Aufbau und die Bereitstellung einer vollständigen CI/CD-Pipeline für eine neue Microservice-Applikation mit Frontend und Backend. Die Herausforderung: alles muss automatisiert, stabil und zwischen Entwicklungs- und Produktivumgebung sauber getrennt sein.

# 🧩 Projektbeschreibung

Sie erhalten eine Projekt-ZIP-Datei mit:  
- einem Java-Backend (Spring Boot, Maven)  
- einem JavaScript-Frontend (z. B. Angular)  
- einer Basiskonfiguration für Build und Deployment  
  
Die Projektdatei ist lokal zu entpacken, in ein eigenes Git-Repository hochzuladen und mit CI/CD und Kubernetes-Deployments produktionsreif umzusetzen.

# 🛠️ Ihre Aufgaben im Projekt

## 1. Projektstruktur & Git-Repository

- Entpacken Sie das Projekt lokal  
- Erstellen Sie ein eigenes Git-Repository (z. B. GitHub, GitLab)  
- Strukturieren Sie das Projekt sauber in /backend, /frontend, /k8s, /jenkins, /docs  
- Laden Sie das Projekt hoch und geben Sie dem Dozenten Zugriff

## 2. Konfiguration über application.properties (Spring Boot)

Im Backend soll zwischen Dev- und Prod-Umgebung unterschieden werden. Dazu:  
- Legen Sie folgende Konfigurationsdateien an:  
 - application-dev.properties  
 - application-prod.properties  
- Legen Sie ein zentrales Maven-Profil für jede Umgebung in der pom.xml an:

<profiles>  
 <profile>  
 <id>dev</id>  
 <properties>  
 <spring.profiles.active>dev</spring.profiles.active>  
 </properties>  
 </profile>  
 <profile>  
 <id>prod</id>  
 <properties>  
 <spring.profiles.active>prod</spring.profiles.active>  
 </properties>  
 </profile>  
</profiles>

- Zwei sinnvolle Unterschiede zwischen Dev und Prod:  
 - Dev: server.port=8081 (für paralleles Testen)  
 - Prod: logging.level.root=ERROR (reduzierte Log-Ausgabe in Produktion)

## 3. CI/CD mit Jenkins

- Erstellen Sie ein vollständiges Jenkinsfile mit folgenden Stages:  
 - Build  
 - Backend: mit Maven und aktivem Profil (dev oder prod)  
 - Frontend: mit npm (z. B. npm install && npm run build)  
 - Test  
 - Backend: automatisierte JUnit-Tests  
 - Frontend: falls vorhanden (z. B. ng test oder npm test)  
 - Docker Build: Erstellen der Docker-Images  
 - Docker Push: Push in die vom Dozenten bereitgestellte Container Registry  
 - Deploy Dev: Deployment in den Dev-Namespace des Kubernetes-Clusters  
 - Manual Approval: manuelle Bestätigung für das Prod-Deployment  
 - Deploy Prod: Deployment in den Prod-Namespace

## 4. Kubernetes-Deployment

- Schreiben Sie Kubernetes-YAML-Dateien für:  
 - Deployments (Backend & Frontend)  
 - Services  
 - ggf. ConfigMaps oder Umgebungsvariablen (z. B. für spring.profiles.active)  
- Verwenden Sie zwei Namespaces: dev und prod  
- Nutzen Sie die vom Dozenten bereitgestellte Kubeconfig zur Verbindung mit dem Cluster

## 5. Dokumentation & Präsentation

- Schreiben Sie ein README.md mit:  
 - Projektübersicht  
 - CI/CD-Prozessbeschreibung  
 - Beschreibung der application.properties und Maven-Profile  
 - Deploymentschritte  
- Präsentieren Sie das Projekt am letzten Tag (27.06.2025) als Live-Demo oder anhand von Screenshots

# 🧾 Technische Rahmenbedingungen

- Kubernetes-Cluster: Wird vom Dozenten zur Verfügung gestellt  
- Jenkins-Server: Wird vom Dozenten bereitgestellt  
- Container Registry: Wird bereitgestellt (Zugang via Credentials)  
- Projekt-Repository: Muss selbst erstellt und gepflegt werden  
- Spring Boot Backend: Mit Maven und application.properties konfiguriert  
- JavaScript Frontend: z. B. Angular (npm-basiert)  
- Namespaces: Trennung in dev und prod für Deployments

# ✅ Abgabe

- Git-Repository mit vollständiger Struktur und CI/CD  
- application.properties sauber getrennt nach Umgebung  
- Funktionierender Build- & Deploymentprozess über Jenkins  
- Präsentation am 27.06.2025