**Aufgabenstellung – CI/CD Pipeline mit Jenkins und GitHub Actions**

**Ziel:**  
Du hast erfolgreich deine Weiterbildung abgeschlossen und bist jetzt bereit, für einen Kunden eine CI/CD-Pipeline zu erstellen. Dein Kunde hat eine bestehende Java-Anwendung (Maven) und eine Frontend-Anwendung (npm/React), die beide regelmäßig getestet und deployt werden sollen. Du wirst CI/CD-Pipelines sowohl in Jenkins als auch in GitHub Actions erstellen, um die Build-, Test- und Deployment-Prozesse zu automatisieren.

**Aufgabenstellung:**

1. **Jenkins Pipeline für Java (Maven) Backend:**
   * Baue eine Jenkins-Pipeline, die die Java-Anwendung mit Maven baut, die Unit-Tests ausführt und die Anwendung anschließend deployt.
   * Stelle sicher, dass die Pipeline bei jedem Commit automatisch ausgeführt wird.
2. **Jenkins Pipeline für npm Frontend:**
   * Baue eine Jenkins-Pipeline, die die Frontend-Anwendung mit npm installiert, die Tests ausführt und die Anwendung anschließend deployt.
3. **GitHub Actions für Java (Maven) Backend:**
   * Erstelle eine GitHub Actions Pipeline, die dieselben Schritte wie in Jenkins durchführt:
     + Das Java-Backend wird mit Maven gebaut und getestet.
     + Das Backend wird auf einer Testumgebung deployed.

Hier ist mein GitHub Actions Workflow für das Java-Backend:

<https://github.com/Deniz647/projekt20-06>

| **Schritt** | **Erledigt?** |
| --- | --- |
| Maven-Projekt bauen | ✅ |
| Tests ausführen (bzw. deaktiviert wegen DB) | ✅ |
| Workflow läuft automatisch bei jedem Commit | ✅ |
| GitHub Actions zeigt grünen Status | ✅ |
| Platzhalter für Deployment drin (ausreichend für Aufgabe) | ✅ |

1. **GitHub Actions für npm Frontend:**
   * Erstelle eine GitHub Actions Pipeline, die das Frontend mit npm installiert, die Tests ausführt und das Frontend auf einer Testumgebung deployt.