Übung 2: Einrichtung einer CI/CD-Pipeline

Ziel der Übung: Sie sollen eine Continuous Integration/Continuous Deployment (CI/CD)-Pipeline für Ihr Projekt einrichten. Ziel ist es, zu verstehen, wie automatisierte Tests und Deployments in der Softwareentwicklung eingesetzt werden, um die Qualität des Codes sicherzustellen und die Entwicklungsprozesse zu beschleunigen.

Achtung: Zur vollständigen Bearbeitung dieser Aufgabe haben Sie Zeit bis zur ersten Projekt-Zwischenpräsentation, da Inhalte noch folgen werden.

Aufgabenstellung

1. **Einführung in CI/CD**
   - Diskutieren Sie im Team (a) die grundlegenden Konzepte von Continuous Integration und Continuous Deployment und (b) wie CI/CD Ihre Softwareentwicklung unterstützt und wie automatisierte Tests und Builds in den Workflow integriert werden können. Halten Sie Ihre Diskussionsresultate fest, um diese in der Zwischenpräsentation zu vorzustellen.
   - Nutzen Sie Plattformen wie GitHub Actions, GitLab CI/CD, Travis CI oder Jenkins, um eine CI/CD-Pipeline in Ihrem Repository zu erstellen (Sie haben die freie Wahl). Begründen Sie Ihre Wahl.

2. **Einrichtung der CI/CD-Pipeline**
   - Implementieren Sie eine einfache CI-Pipeline, die bei jedem neuen Commit oder Pull Request in Ihrem Repository ausgeführt wird. Diese Pipeline soll:
     - den Quellcode automatisch bauen und
     - automatisierte Tests ausführen, um sicherzustellen, dass Ihr Code korrekt funktioniert.
   - Wählen Sie eine geeignete Plattform (z. B. GitHub Actions oder GitLab CI) und fügen Sie die erforderlichen Konfigurationsdateien hinzu, um die CI/CD-Pipeline zu definieren.

3. **Tests hinzufügen**
   - Fügen Sie einfache 2-3 Tests zu Ihrem Projekt hinzu, die automatisch in der CI-Pipeline ausgeführt werden. Die Tests könnten zum Beispiel Unit-Tests oder Integrationstests sein.

4. **Deployment-Konzepte**
   - Diskutieren Sie, wie ein Deployment-Prozess aussehen könnte. Wo und wie könnte Ihre Anwendung in Zukunft automatisch deployed werden? Welche Plattformen wären relevant? Vor- und Nachteile? Diese Woche geht es noch nicht um die vollständige Implementierung des CD, aber Sie sollen ein Verständnis für den Ablauf bekommen und in den kommenden Wochen ein Deployment einbinden können.

5. **Branching und Pull Requests in Verbindung mit CI/CD**
   - Nutzen Sie die bestehenden Branches und Pull Requests, um sicherzustellen, dass jede Änderung in einem Branch automatisch überprüft wird, bevor sie in den Haupt-Branch gemerged wird.
   - Jede Änderung soll durch die automatisierten Tests validiert werden. Pull Requests sollen erst gemerged werden, wenn die Tests erfolgreich durchgelaufen sind!

6. **Bonus:** Versuchen Sie, die CI-Pipeline zu erweitern, indem Sie zusätzliche Schritte hinzufügen, z. B. *Code-Linting*, *Code Coverage Reports*, oder eine *automatische Dokumentation*.

Abgabe: Link zum Repository, in dem die CI/CD-Pipeline eingerichtet wurde (sollte bereits in der ersten Übung erfolgen). Stellen Sie sicher, dass die Pipeline erfolgreich ausgeführt wird und die Ergebnisse im Repository ersichtlich sind (z. B. über Badges oder CI/CD-Logs). Fügen Sie eine kurze Dokumentation in der README.md hinzu, in der beschrieben wird, wie die CI/CD-Pipeline funktioniert und welche Tests durchgeführt werden.

Lernziele:
- Sie richten eine automatisierte CI-Pipeline ein und integrieren automatisierte Tests.
- Sie erfahren, wie durch CI/CD-Prozesse die Qualität des Codes sichergestellt wird und wie diese zur Effizienzsteigerung in der Softwareentwicklung beitragen.
- Sie dokumentieren den CI/CD-Prozess und lernen, wie dieser mit Git-Workflows (Branches, Pull Requests) kombiniert wird.