Einleitung

Die Repository verwendet eine GitLab CI/CD Pipeline, die automatisch bei jedem Merge Request ausgeführt wird. Die Pipeline baut sowohl das Spring Boot Backend als auch das React Frontend.

Daniel Kovac hat beigeholfen in dem er mir erklärt hat was er bisher an seinem Code gearbeitet hat. Ebenfalls hat er mir seine Erfharung mit GitLab und CI/CD-Pipelines geteilt da er in einem vorherigen Modul bereits damit gearbeitet hat.

Realisierung

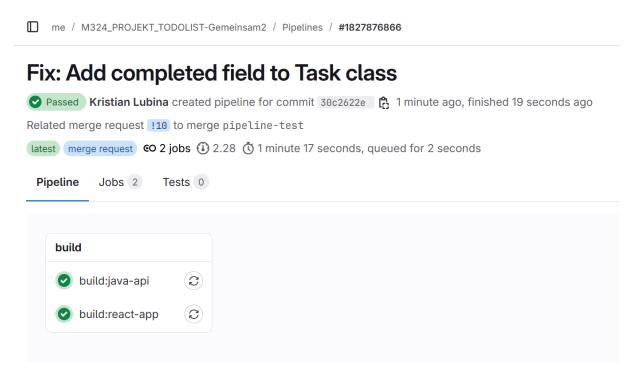
Die Pipeline wurde durch Erstellung einer .gitlab-ci.yml Datei im Wurzelverzeichnis des Projekts implementiert. Diese Konfigurationsdatei definiert zwei Build-Jobs:

- Backend-Build für das Java Spring Boot Projekt
- 2. Frontend-Build für die React-Anwendung

Während der Implementierung musste ein Fehler im Backend-Code behoben werden: In der Klasse Task.java fehlte die Deklaration der Variable completed, die für die User Story 4 (Aufgabenstatus) benötigt wird. Nach dem Hinzufügen dieses Feldes konnte die Pipeline erfolgreich ausgeführt werden.

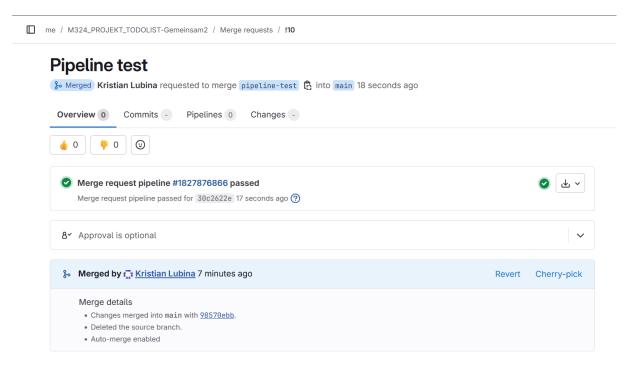
Erfolgreiches Build

Die Pipeline wurde erfolgreich mit zwei parallel laufenden Jobs ausgeführt, wie im folgenden Screenshot zu sehen ist.



Erfolgreicher Merge-Request

Nach dem erfolgreichen Build wurde ein Merge-Request mit dem Namen "Pipeline test" erstellt, der ohne Probleme gemergt werden konnte.



Verwendete Docker-Images

- Backend: maven:3.8.3-openjdk-17 Enthält Maven und Java 17
- Frontend: node:18.12.1 Enthält Node.js und npm

Pipeline-Struktur

Die Pipeline besteht aus einer Build-Phase mit zwei parallel laufenden Jobs:

1. Backend Build (Java/Spring Boot)

- o Baut das Java-Projekt mit Maven
- Erzeugt JAR/WAR Dateien
- o Speichert die Artefakte für 30 Tage

2. Frontend Build (React)

- o Installiert alle npm-Abhängigkeiten
- o Baut die React-Anwendung zu statischen Dateien (HTML, CSS, JS)
- o Speichert die Build-Artefakte für 30 Tage

README.md

Folgendes wurde in das README.md ergänzt um die Funktionsweise der Pipeline zu erläutern:

CI/CD Pipeline

Übersicht

Die Repository verwendet eine GitLab CI/CD Pipeline, die automatisch bei jedem Merge Request ausgeführt wird. Die Pipeline baut sowohl das Spring Boot Backend als auch das Reach Ernstend

Pipeline-Struktur

Die Pipeline besteht aus einer Build-Phase mit zwei parallel laufenden Jobs:

- 1. Backend Build (Java/Spring Boot)
 - Baut das Java-Projekt mit Maven
 - Erzeugt JAR/WAR Dateien
 - Speichert die Artefakte f
 ür 30 Tage

2. Frontend Build (React)

- Installiert alle npm-Abhängigkeiten
- Baut die React-Anwendung zu statischen Dateien (HTML, CSS, JS)
- Speichert die Build-Artefakte f
 ür 30 Tage

Konfiguration

Die Pipeline ist in der .gitlab-ei.ynl Datei im Wurzelverzeichnis des Projekts definiert:

```
stages:
- build
build:java-api:
 stage: build
 image: maven:3.8.3-openjdk-17
 script:
    - cd backend && mvn clean install
  artifacts:
    untracked: false
    when: on_success
    expire_in: "30 days"
      - backend/target/*.jar
     - backend/target/*.war
  only:
build:react-app:
 stage: build
  image: node:18.12.1
  script:
    - cd frontend && npm install && CI=false npm run build
  artifacts:
    untracked: false
    when: on_success
   expire_in: "30 days"
    paths:
     - frontend/build
  only:
    - merge_requests
```