ict-m324 - Prüfungsaufgabe: GitHub Actions und CI/CD

- Achten Sie auf die geforderten Commit Messages.
- Die Aufgabe 1-3 gehören zusammen. Die Aufgaben 4 und 5 können unabhängig voneinander gelöst werden.
- Code Vorlage für die Aufgaben 1-3: Siehe ZIP im Teams Chat.

Aufgabe 1: Repository verwenden

Verwenden Sie das GitHub Repository, das Sie in der letzten Stunde für die Prüfungsvorbereitung abgegeben haben (siehe Excel Liste). Das Repository muss einen `main` branch haben und leer sein - `README.md` ist erlaubt.

- **Abgabe**: Stellen Sie sicher, dass Sie das Repo verwenden, das Sie in der Excel Liste angegeben haben.

1. **Repo aufsetzen**
   - Öffnen Sie das Repository lokal
   - Kopieren Sie alle Dateien aus dem Prüfungs-ZIP (Teams Message) in ihr lokales Repository.

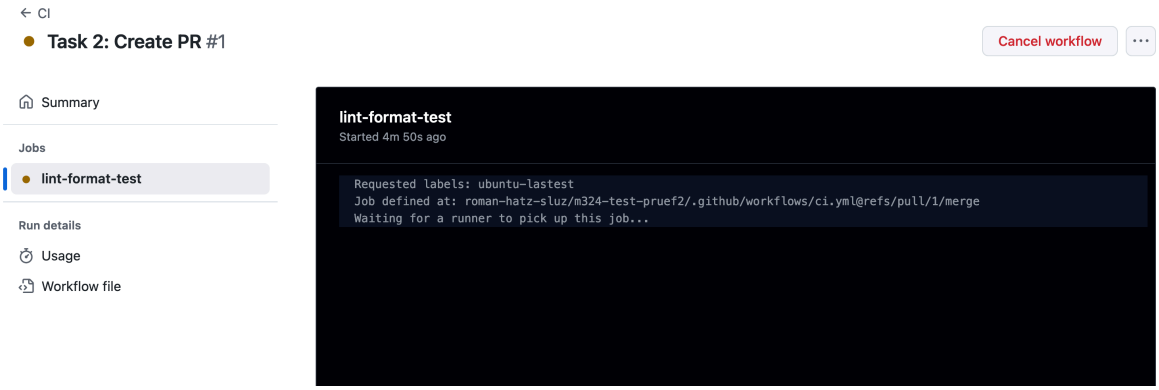
- **Commit Messages**:
  - `"Task 1: Init"` (Branch `main`)

Aufgabe 2: Setup, Updates und lokal zum Laufen bringen

- Leiten Sie einen neuen Branch namens `ci` vom `main` Branch ab.
- Fügen Sie eine leere Datei mit dem Namen `exam.md` hinzu.
- Erstellen Sie nach dem Push einen **Pull Request** auf github.com (Branch `ci`).
- **Commit Message**:
  - `"Task 2: Create PR"` (Branch `ci`)
- Erstellen Sie ein Update aller NPM Packages und committen Sie alle relevanten Dateien.
- **Commit Message**:
  - `"Task 2: Update NPM"` (Branch `ci`)

Aufgabe 3: Fehler im CI-Workflow beheben

- Der Workflow startet noch nicht. Tipp: Die Datei `ci.yml` muss im korrekten Ordner sein.
- **Commit Messages**:
  - `"Task 3: Fix workflow folder"` (Branch `ci`)
- Ihr Pull Request auf github.com sollte anzeigen, dass der Workflow startet. Leider gibt es Problem. Es sollte so aussehen wie im Bild:



1. **Fehler im CI-Workflow beheben**

- In der YML Datei sind 2 Fehler eingebaut.
- Beheben Sie beide Fehler in der YML Datei und erstellen Sie jeweils einen Commit.
- **Commit Messages**:
  - `"Task 3: Fixed CI Error 1"` (Branch `ci`)

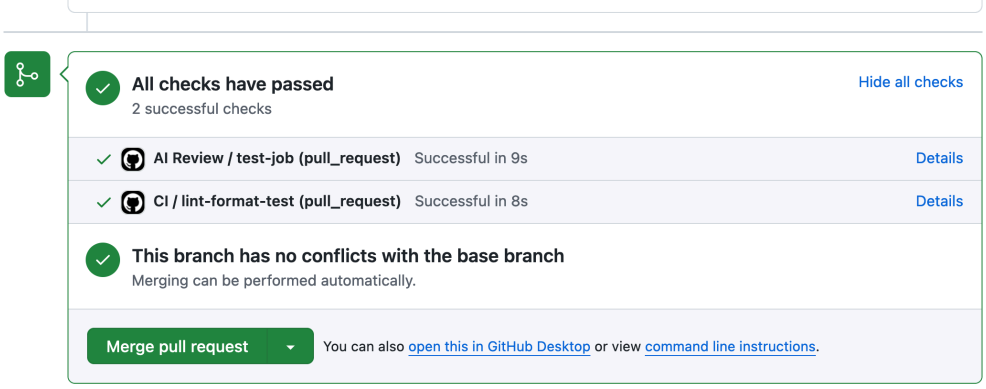
    - `"Task 3: Fixed CI Error 2"` (Branch `ci`)
- Der Workflow sollte auf github.com jetzt starten, es gibt jedoch einen Fehler beim Linter. Fixen Sie den Fehler und erstellen Sie einen Commit.
- **Commit Messages**:
    - `"Task 3: Fixed Linter Error"` (Branch `ci`)
- Als nächstes gibt es einen Fehler beim Formatter. Fixen Sie den Fehler.
- **Commit Messages**: `"Task 3: Fixed Formatting error"` (Branch `ci`)
- Der GitHub Workflow sollte jetzt funktionieren und die Nachricht "All checks have passed" anzeigen.

2. **Verbesserung der Paketinstallation**
    - Ersetzen Sie `run: npm install` durch eine bessere Variante, um einen reproduzierbaren Build sicherzustellen.
    - **Commit Message**:
        - `"Task 3: Improved package installation in CI"`

3. **Pull Request mergen**
    - Mergen Sie den Pull Request in den `main` Branch.
    - Löschen Sie den `ci` Branch nach dem Merge *nicht*.

Aufgabe 4: KI-Code-Review einrichten

- **Commit Messages**: Mehrere Commit sind erlaubt. Alle Commits zu dieser Aufgabe müssen mit `"Task 4:"` beginnen. Branch: `ai-review`. Schreiben Sie aussagekräftige und nachvollziehbare Commit Messages.
- Erstellen Sei einen neuen Workflow `"ai_review.yml"`
- Erstellen Sie einen neuen Pull Request.
- Verwenden Sie eine Action, um ein KI-generiertes Code-Review zu Ihrem Pull Request zu erhalten. Bei jedem Push in einen offenen Pull Request wird automatisch ein Code Review hinzugefügt.
- Am Schluss sollte es ungefähr so aussehen:



- Deaktivieren Sie im yml den Workflow mit einer `if` Bedingung sobald das Review einmal erfolgreich durchgelaufen ist.
- Hilfe
  - Wenn Sie schon einen API Key für einen AI Service haben, können Sie diesen nutzen. OpenAI Key: Siehe Nachricht im Teams Chat. Geben Sie als Modell für diesen Key `gpt-3.5-turbo` an.
  - Ein Key geben Sie als Secret bei den Settings ein.

Pull Request mergen

- Mergen Sie den Pull Request in den `main` Branch.
- Löschen Sie den `ai-review` Branch nach dem Merge *nicht*.

Aufgabe 5: DevOps Pipeline

Ausgangslage

Ihre Firma nutzt aktuell eine externe GitHub Action mit OpenAPI aus Aufgabe 4. Aus Datenschutz-, Kosten- und Compliance-Gründen soll das künftig intern betrieben werden: Ein eigener Service (Container) stellt ein internes LLM/AI-Review bereit. Sie erhalten den Auftrag, dafür eine DevOps Pipeline zu erstellen.

Ziel

Sie erstellen ein eigenes Docker-Image, pushen es auf Docker Hub und führen es **automatisch in GitHub Actions** aus.

- Der Container gibt beim Start **`This is an AI Review`** aus (Minimal-Prototyp für einen späteren AI-Review-Service).
- Die Nachricht `This is an AI Review` erscheint als Kommentar im GitHub Pull Request

Anforderungen

- **Commit Messages**: Mehrere Commit sind erlaubt. Alle Commits zu dieser Aufgabe müssen mit `"Task 5:"` beginnen. Branch: `ai-custom-review` . Schreiben Sie aussagekräftige und nachvollziehbare Commit Messages. Erstellen Sie einen Pull Request. Löschen Sie den branch `ai-custom-review` nicht.

1. Dockerfile

- Erstellen Sie ein `Dockerfile` .
- Beim Start muss **`This is an AI Review`** auf STDOUT ausgegeben werden.
- Der Container darf danach sofort beenden.

2. DockerHub

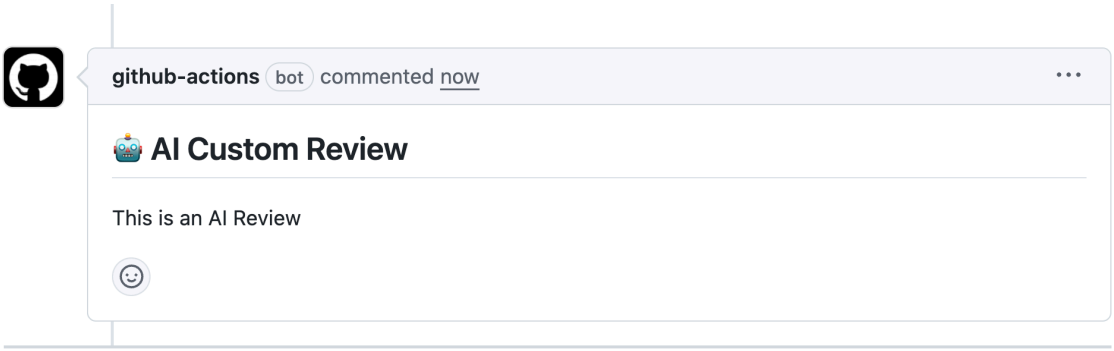
- Nutzen Sie ihr DockerHub-Repository wie in der Übung
- Image lokal bauen, taggen und auf DockerHub pushen.

3. GitHub Actions Workflow - Setup

- Erstellen Sie die Datei `ai_custom_review.yml`
- Der Workflow muss:
  - Starten, wenn in den Pull Request gepusht wird.
  - Das Image **pullen und als Container ausführen**
  - Die Ausgabe des Containers muss im GitHub Actions Log sichtbar sein.

4. GitHub Actions Workflow - Setup PR Comment

- Die Ausgabe des Containers erscheint als Kommentar im GitHub Pull Request. Es sollte so aussehen



5. GitHub Actions Workflow - Next steps

- Die Ausgabe des Containers ist neu ein Diff des geänderten Codes. Die Ausgabe des Container beginnt mit `This is an AI Review` , und fügt die ersten 100

Zeichen des Code Diffs an. Diese Transformation muss im Container erfolgen.