# Projekt Anleitung





## 1. Klonen des Repositories

Um das Repository zu klonen, verwenden Sie den folgenden Befehl (ersetzen Sie `<SSH-Schlüssel>` durch den tatsächlichen SSH-Link des Repositories):

git clone <SSH-Schlüssel>

Falls der SSH-Schlüssel fehlt, können Sie diesen in den GitHub-Einstellungen erstellen und Ihrem Konto hinzufügen.

## 2. Einrichtung der Entwicklungsumgebung

Nach dem Klonen des Projekts sollten Sie folgende Schritte durchführen, um die Entwicklungsumgebung einzurichten:

1. Navigieren Sie in das Verzeichnis des geklonten Projekts:

cd <repository-ordner>

2. Installieren Sie die nötigen Pakete (hier beispielsweise für eine Node.js-Umgebung):

npm installa

3. Optional: Erstellen Sie eine Konfigurationsdatei (.env), falls das Projekt Umgebungsvariablen benötigt.

## 3. Erstellung der README.md

Die README.md dient als Dokumentation für Ihr Projekt. Typischerweise enthält sie:

- Projektbeschreibung: Kurze Zusammenfassung des Projekts.

- Installationsanweisungen: Anweisungen zum Klonen des Repositories und Installieren notwendiger Pakete.

- Nutzungsanweisungen: Wie das Projekt gestartet und genutzt wird.

Beispiel:

# Projekt Setupa  
  
## Installation  
```bash  
npm install  
```

- Contributing: Informationen für andere Entwickler, wie sie beitragen können.

## 4. Verwendung von Git (Commit, Push)

Um Änderungen zu committen und zu pushen, folgen Sie diesen Schritten:

1. Änderungen hinzufügen:

git add .

2. Commit erstellen:

git commit -m "Beschreibung der Änderungen"

3. Änderungen pushen:

git push origin main

Hinweis: Ersetzen Sie `main` durch den Namen des Ziel-Branches, falls Sie einen anderen Branch verwenden.

## 5. Erstellung und Nutzung von Docker-Containern

Docker ermöglicht es, Anwendungen in Containern zu isolieren und bereitzustellen. Die grundlegenden Schritte für Docker sind:

1. Dockerfile erstellen (falls noch nicht vorhanden): Dieses enthält alle Befehle, um das Projekt in einem Container auszuführen.

2. Image erstellen:

docker build -t <image-name> .

3. Container starten:

docker run -d -p 3000:3000 <image-name>

Dieser Befehl startet den Container im Hintergrund (`-d`) und leitet den Port `3000` vom Container auf `3000` des Hosts weiter.

4. Docker-Compose verwenden: Um mehrere Container zu orchestrieren, erstellen Sie eine `docker-compose.yml` und starten alle Container mit:

docker-compose up --build