# Abschlussaufgabe Development Fundamentals



Inhalt

[Abschlussaufgabe Development Fundamentals 1](#_Toc181367270)

[1. Repository klonen 3](#_Toc181367271)

[3. README.md erstellen 3](#_Toc181367272)

[4. Mit Git committen und pushen 4](#_Toc181367273)

[5. Docker-Container nutzen 4](#_Toc181367274)

1. Repository klonen  
   Zieh das Repository mit dem Befehl runter (ersetz <SSH-Link> mit dem echten SSH-Link):

bash

**git clone <SSH-Link>**

Falls du keinen SSH-Schlüssel hast, leg einen in den GitHub-Einstellungen an und verbinde ihn mit deinem Account.

1. **Entwicklungsumgebung einrichten**  
   Nachdem du das Projekt geklont hast, mach Folgendes, um die Umgebung startklar zu machen:
   * Geh ins Projektverzeichnis:

bash

**cd <projektordner>**

* + Hol dir die nötigen Pakete (bei Node.js zum Beispiel):

bash

**npm install**

* + Optional: Falls Umgebungsvariablen gebraucht werden, leg eine .env-Datei an.

Ein Bild, das Text, Elektronik, Screenshot, Multimedia enthält.

Automatisch generierte Beschreibung

1. README.md erstellen  
   Die README.md erklärt dein Projekt. Hier sollten ein paar Basics rein:
   * **Kurzbeschreibung**: Was das Projekt macht.
   * **Installation**: Kurze Anleitung zum Klonen und Installieren.
   * **Nutzung**: Wie man das Projekt startet und benutzt.

Beispiel:

markdown

# Projekt Setup

## Installation

```bash

npm install

markdown

- \*\*Mitmachen\*\*: Infos, wie andere am Projekt mitwirken können.

1. Mit Git committen und pushen  
   Änderungen speichern und hochladen geht so:
   * Alle Änderungen hinzufügen:

bash

**git add .**

* + Commit erstellen:

bash

**git commit -m "Was du geändert hast"**

* + Alles hochladen:

bash

**git push origin main**

**Ein Bild, das Text, Software, Multimedia-Software, Grafiksoftware enthält.

Automatisch generierte Beschreibung**

Hinweis: Falls du einen anderen Branch als main hast, nimm den entsprechenden Branch-Namen.

1. Docker-Container nutzen  
   Docker isoliert deine App in einem Container. Grundschritte sind:
   * Dockerfile anlegen, wenn noch keins da ist – es enthält die Anweisungen, um das Projekt im Container zu starten.
   * Docker-Image erstellen:

bash

**docker build -t <image-name> .**

* + Container starten:

bash

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

Der Befehl startet den Container im Hintergrund und leitet Port 3000 des Containers auf 3000 vom Host um.

* + Für mehrere Container docker-compose nutzen. Erstelle eine docker-compose.yml und starte die Container mit:

bash

**docker-compose up –build**

**Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung**