ToDo-Application

Inhalt

[1 Git 2](#_Toc177739787)

[1.1 Fork 2](#_Toc177739788)

[1.2 Clone 2](#_Toc177739789)

[1.3 Git befehle 2](#_Toc177739790)

[2 Erstellung der Docker-Umgebung 2](#_Toc177739791)

[2.1 Einrichten der Docker-Umgebung für Node.js 2](#_Toc177739792)

[2.2 Bauen und Ausführen im Docker-Container 3](#_Toc177739793)

[3 Erstellung von README.md 3](#_Toc177739794)

# 1 Git

## 1.1 Fork

Ich bekam einen Link zu einem GitHub Repository vom BLJ. In diesem Repository ist der Node.js Code für die ToDo-Application gespeichert und auch schon eine README.md Datei. Dieses Repository musste ich Forken.

Ein Fork auf GitHub ist eine Kopie von einem Repository. Somit kann man den genau gleichen Code von einem Anderen User auf sein GitHub Account kopieren. Danach kann man ein neues Repository auf dem eigenen Account erstellen und hat dann den gleichen Code bei sich selbst. So kann man den Code unabhängig vom anderen bearbeiten. Der Fork bleibt jedoch mit dem Original-Repository verbunden, sodass man später Änderungen zwischen beiden Repositorys synchronisieren kann.

## 1.2 Clone

Danach musste man das Repository auf den Rechner Lokal Clonen. Dies macht man, indem man beim Repository bei «Code» den SSH-Key kopiert und dann in der Kommandozeile zum Verzeichnis navigiert, in welchem der Clone sein soll (bei mir: «C:\Daten\020\_ZLI\010\_projects\010\_ToDoApplication»). Dann muss man den Befehl «git clone ssh://username@host.xz/absolute/path/to/repo.git/» eingeben. Nun hat man die Verbindung von GitHub zum eigenen Rechner.

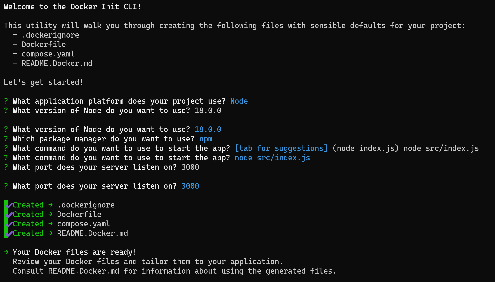
## 1.3 Git befehle

Nun sind alle Dateien auf GitHub und bei mir Lokal auf dem Rechner. Wenn man nun den Code Lokal auf dem Laptop anpasst, ist dies jedoch noch nicht auf GitHub. Für dies benötigt man verschiedene Befehle. Als erstes muss man einen neuen Branch erstellen. Für dies wird der Befehl «git checkout -b name». Danach kann man im neuen Branch den Code anpassen. Danach muss man mit «git add .», «git commit -m ‘’Kommentar’’» und «git push» den Branch auf GitHub hochladen. Dies kann man sich wie ein Packet vorstellen, wo alles hinzugefügt wird (Add .) dann Beschriftet (Commit) und abgesendet (Push) wird. Nun hat man auf GitHub ein Pull Request. Hier hat man nochmal die letzte Chance etwas anzupassen, denn wenn man ihn merged dann wird der Main aktualisiert und mit allen geänderten Daten. Nun ist der Main auf dem Aktuellen Stand auf GitHub. Um nun Lokal auch noch den Main aktuell zu haben kann man mit «git pull» die Daten wieder herunterholen und ist auch da auf dem neusten Stand.

# 2 Erstellung der Docker-Umgebung

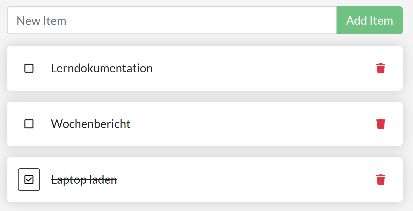
## 2.1 Einrichten der Docker-Umgebung für Node.js

Für das Einrichten der Docker-Umgebung für das Projekt mit dem Node.js gibt es einen [Anleitungspfad](https://docs.docker.com/guides/language/nodejs/containerize/) auf Docker Docs. Für das Projekt, welches wir umsetzten, war dieser genau das richtige. Als erstes muss man Docker Desktop installieren. Hierbei ist es wichtig, dass man die richtige Version herunterladet. Es gibt von Windows eine für ARM und für Intel Prozessoren. Wenn man das falsche installiert, geht es nicht. Nach dieser Installation muss man im Terminal beim richtigen Verzeichnis den Befehl «docker init». Dann kommen Verschiedene Fragen, die man die man für die Initialisierung beantworten muss. Ich hatte bei der Docker Docs Anleitung die Vorgaben, was ich antworten musste. Nachdem man diese Initialisierung abgeschlossen hat, wurden verschiedene Files im Verzeichnis erstellt und verschiedene wichtige Pakete erstellt. Bei mir wurde bei so einem Paket eine zu neue Version erstellt, und somit konnte der nächste Schritt nicht ausgeführt werden und es funktionierte alles nicht. Die Version von «sqlite3» war zu neu für die Docker-Version. Somit konnten nicht alle Pakete geladen werden.



Nach erfolgreicher Initialisierung

## 2.2 Bauen und Ausführen im Docker-Container

Der nächste Befehl ist «docker compose up –build» welcher die zuvor initialisierte Umgebung erstellt. Nun läuft die Applikation unter der URL localhost:3000. Bei der Initialisierung musste man den Standardort angeben (3000), deshalb ist es nun :3000. Links ist ein Bild von dem UI mit einer Beispiel ToDo-List. Nun ist die gesamte Docker Umgebung aufgebaut und nutzbar.

# 3 Erstellung von README.md

Eine README.md Datei ist in fast jedem Projekt vorhanden. In dieser Datei wird beschrieben, um was es beim Projekt geht. In dem File oft kurz dokumentiert was man wie gemacht hat oder auch was der Code macht. Wenn man einen Fehler im Code entdeckt hat oder eine mögliche Fehlerquelle ist es praktisch, wenn man dies im README festhaltet. Es ist wie eine kurze Zusammenfassung des Projekts und dem Code. Dies wird meist mit der Markdown Sprache gemacht, weil es sehr einfach und übersichtlich ist damit zu schreiben.