# **Word Docker To-Do-Listen Dokumentation**

## **Klonen des Repositories**

Zuerst habe ich einen Fork erstellt des Repositories mit diesem knopf auf Github:Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

KI-generierte Inhalte können fehlerhaft sein.

Danach habe ich **Git bash** gestartet.

Zuerst habe ich den Command *cd C:/git* geschrieben um diesen Ordner rein zu gehen.

Nachdem den Command *git clone* [*git@github.com:philippesaxer/docker-nodejs-sample.git*](mailto:git@github.com:philippesaxer/docker-nodejs-sample.git) um mein Repository lokal zu klonen.

Ein Bild, das Text, Screenshot, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

## **Erstellung der README.md**

Eine README.md war bereits vorhanden. Ich musste den Text darin löschen, um die Anleitung direkt dort reinzuschreiben. Immer nach jedem Schritt, den ich machte schrieb ich die Anleitung.

## **Verwendung von Git (Commit, Push)**

Beim README.md habe ich die Commits direkt in Github gemacht.

Den Rest konnte ich immer mit:

* git status
* git add .
* git commit -m «»
* git push

Leider ging es teilweise nicht und die commits gingen verloren bzw. gingen auf das falsche Repository.

## **Docker Anleitung Schritt 1-5**

Zuerst musst du wie vorher im richtigen Speicherort sein. Dannach muss man den Command *docker init* im Git Bash eingeben.

Es stellt dir dann Fragen, die man so Antworten soll:

Ein Bild, das Text, Screenshot, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

Danach muss man *docker compose up --build -d* eingeben und es startet

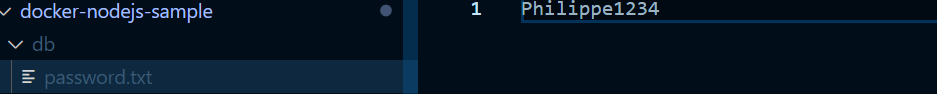
Um es zu stoppen den Command *docker compose down*

Ich musste danach das compose.yaml file ändern.

In den **docker-nodejs-sample** directory, einen directory namens **db erstellen**.

In den **db** directory, einen File namens **password.txt**. Dieses File hat dein Passwort Data drin.

Die Datei password.txt öffnen in einer IDE oder einem Texteditor und dann habe ich ein Passwort eingegeben. Das Passwort muss in einer einzigen Zeile stehen.



Dann Änderungen an allen Dateien speichern.

Zum Schluss den Befehl ausführen, um Anwendung zu starten: *docker compose up –build*

Später musste ich Änderungen am Dockerfile machen sowie am Compose.yaml. Ich musste ein mehrstufiges Dockerfile erstellen und einen Bind-Mount für die Entwicklung einrichten.

Im nächsten Abschnitt musste ich GitHub Actions und workflow für mein Node.js machen. Genauer gesagt musst ich ein neues Repository erstellen. Dann musste ich eine Variable erstellen und einen Personal Access Token erstellen und ins Repostitory einfügen. Dannach einen Workflow erstellen mit main.yaml und ihn ausführen.

Für den letzten Schritt musste ich zuerst Kubernetes im Docker Desktop aktivieren. Danach musste ich ein Kubernetes Yaml file erstellen.

Als letztes musste ich diese Commands machen.

* Um in einem Terminal zu dem Ort navigieren , an dem man **docker-node-kubernetes.yaml** Anwendung erstellt haben. Um in Kubernetes bereit zu stellen Comand: *kubectl apply -f docker-node-kubernetes.yaml*
* Mit den danachfolgenden Commands sollte man eine Ausgabe sehen, die zeigt dass Ihre Kubernetes-Objekte erfolgreich erstellt wurden. Commands: *deployment.apps/docker-nodejs-demo created  
  service/todo-entrypoint created*
* Um sicher zu stellen, dass alles funktioniert hat, kann man mit dem Command Bereitstellungen auflisten. *kubectl get deployments*
* Die gleiche Prüfung für die Dienste durch den Command: *kubectl get services*
* Jetzt kann man einen Browser öffnen und die App unter **localhost:30001** sehen. Die Anwendung sollte angezeigt werden.
* Um die Anwendung zu beenden. Den Befehl: *kubectl delete -f docker-node-kubernetes.yaml*