Technische Berufsschule Zürich TBZ

KN02: Dockerfile

Beachten Sie die allgemeinen Informationen zu den Abgaben.

Grundlagen für diesen Auftrag:

- TBZ: Dockerfile Grundlagen
- Docker: Referenz

A) Dockerfile I

In KN01 hatten Sie keine eigentliche Kopie des öffentliche Images angelegt, sondern Sie haben einen Layer erstellt, welcher das öffentliche Image beinhaltet. Sie können jederzeit zusätzliche Layer erstellen und andere Images dadurch erweitern. **Achtung**: Wir arbeiten nun wieder mit den öffentlichen Images, nicht mit Ihren privaten. Löschen Sie Ihre Images aber erst, wenn dieser Auftrag abgenommen wurde.

Um bestehende Images zu erweitern, müssen Sie ein *Dockerfile* erstellen und die entsprechenden Anweisungen definieren.

Nehmen Sie das folgende Dockerfile und dokumentieren Sie die einzelnen Zeilen in **eigenen** Worten.

```
FROM nginx  # ...
COPY static-html-directory /var/www/html # ...
EXPOSE  80  # ...
```

Ändern Sie nun den Inhalt um eine Webseite zu erstellen

1. Laden Sie sich diese HTML-Datei herunter und kopieren Sie sie in Ihr Arbeitsverzeichnis.
2. Kopieren Sie nun die drei Zeilen aus dem vorgegebenen Dockerfile in eine echte Datei und passen Sie den Inhalt so an, dass die HTML-Datei kopiert wird. Der **Dateiname** ist *Dockerfile* ohne Dateierweiterung. Zusätzliche Änderungen:
3. **Wichtige** Änderung: Verwenden Sie die Anweisung WORKDIR, dafür verzichten Sie in der COPY-Anweisung auf den absoluten Pfad. Sie müssen also nun herausfinden wo das nginx-Image die HTML-Seiten ablegt. Docker-Hub sollte Auskunft geben können. Der Beispiel-Pfad oben ist nicht korrekt.
4. Führen Sie den **Befehl** `docker build` mit den korrekten Parameter aus. Ihr neues Image wird erstellt.

5. Pushen Sie Ihr Image in **Ihr privates Repository** mit dem **Tag *kn02a***
6. Erstellen Sie einen Container und rufen Sie die Seite helloworld.html auf.

Abgaben:

- Dokumentiertes Dockerfile
- Dockerfile, welches mit den entsprechenden Zeilen wie oben beschrieben.
- Notwendige Docker-Befehle für das *build*. Es ist einfacher für den nächsten Schritt, wenn Sie bereits den korrekt Tag für Dockerhub verwenden mit Benutzername, etc.
- Notwendige Befehle für den Start des Containers und dem push in das private Repository (gemäss KN01)
- Screenshot aus Docker Desktop, welcher das **Image kn02a zeigt**.
- Screenshot der HTML-Seite, der die Seite helloworld.html zeigt, nachdem der Container gestartet wurde

B) Dockerfile II

Sie werden nun wieder die beiden Seite *info.php* und *db.php* aus m346 zum laufen kriegen. Die folgende Kurzanleitung soll Ihnen eine Hilfestellung sein, ohne, dass alles vorgegeben wird. **Lesen Sie zuerst alles durch**. Machen Sie eigene Nachforschungen, falls notwendig.

Sie benötigen zwei Images - eines für die PHP/Webseite und eines für die Datenbank/Mariadb. Sie dürfen auch eine andere DB verwenden, wenn Sie möchten.

- Beide Images müssen einen Port exponieren. Sie finden diese in Ihren Unterlagen aus m346, falls Sie die nicht mehr wissen.
- Wenn Sie auf Docker Hub nach geeigneten Images suchen, wird Ihnen jeweils auch angezeigt, wie es verwendet werden kann. Lesen Sie sich ein, aber beachten Sie die folgenden Punkte
- Image für Datenbank (basierend auf mariadb Image)
    - In der Anleitung werden env-Parameter mitgegeben um den Benutzernamen und das Passwort zu setzen. Da Sie aber einen neuen Layer erstellen, können (**und sollen**) Sie die Credentials auch im Dockerfile definieren - mit Hilfe der Dockerfile-Anweisung *ENV*.
    - Schlauerweise beginnen Sie mit der Datenbank und wenn Sie von Ihrem Host eine Verbindung aufbauen können, beginnen Sie mit der Webseite.
    - Verwenden Sie den eindeutigen Namen **kn02b-db** für Ihren Container im `docker run` Befehl, weil Sie diesen Namen später verwenden werden und sich dieser nicht ändern sollte.
    - Erstellen Sie die Abgaben für den DB-Server bevor Sie weiter gehen (siehe unten)
- Image für Webseite (basierend auf PHP Image)
    - Das offzielle Image ist https://hub.docker.com/_/php. Dieses verwendet allerdings die Skripting Variante, ohne Webserver. Sie müssen herunterscrollen, bis zum Kapitel "Image Variants". Verwenden Sie die Variante *php:8.0-apache* (FROM php:8.0-apache)
    - Schauen Sie nach, wo die Dateien liegen müssen. Kopieren Sie die beiden Dateien (info.php und db.php) aus m346 in die entsprechenden Ordner.
    - Führen Sie *docker-php-ext-install mysqli*, um das PHP mysqli zu installieren. Schauen Sie in der Dockerfile-Dokumentation nach wie sie den Befehl ***RUN*** verwenden.

    - Verwenden Sie den root-Benutzer und Passwort, nicht den zusätzlichen Benutzer (weil der zusätzliche Benutzer keine Rechte auf die mysql-Datenbank hat). Sie haben das Passwort gerade eben gesetzt.
    - Damit die beiden Container miteinander kommunizieren können, werden die entsprechenden Links benötigen für das Networking. Lesen Sie sich unter diesem Link ein. Sie müssen verstanden haben, welches die Domain ist, die Sie in db.php verwenden müssen.
    - Verwenden Sie den eindeutigen Namen **kn02b-web** für Ihren Container im `docker run` Befehl, weil Sie diesen Namen später verwenden werden und sich dieser nicht ändern sollte.
- Publizieren Sie Ihre Images in das private Repository mit den Tags *kn02b-db*, resp. *kn02b-web* .

Abgaben:

- DB: telnet Befehl der zeigt, dass der Zugriff auf den DB Server funktioniert (Screenshot)
- DB: Dockerfile für Ihren DB-Container
- DB: `docker build` und `docker run` Befehle für Ihren DB-Container.
- Web: Screenshots der beiden Seiten info.php und db.php
- Web: Dockerfile für Ihren Web-Container
- Web: `docker build` und `docker run` Befehle für Ihren Web-Container.
- Web: Angepasste PHP-Dateien