

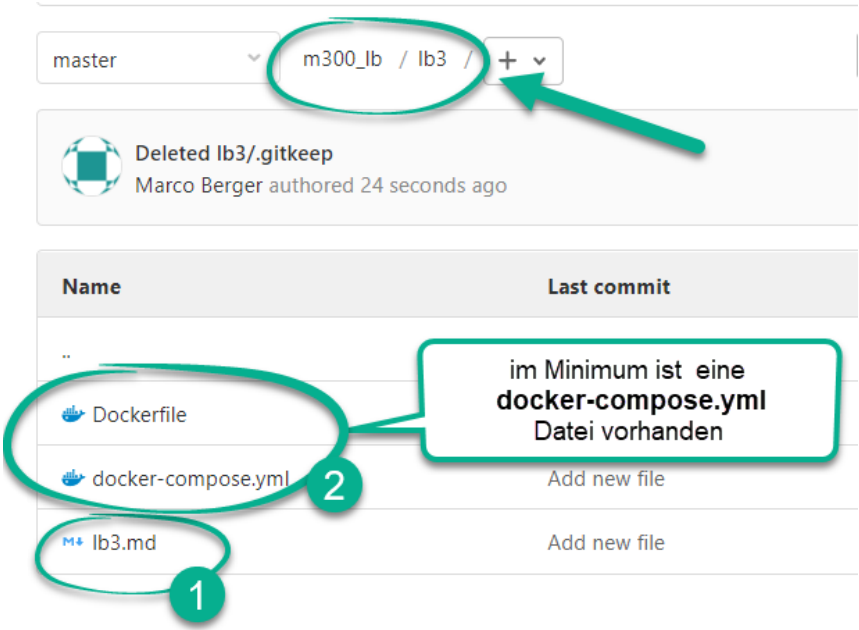
M300 LB3 - Container

Prüfungsform

Praktische Einzelarbeit. Für die Arbeit sind **12 Lektionen vorgesehen**, wovon sie **7 Lektionen während dem Schulunterricht (DL & Präsent)** zur Verfügung haben.

Für die LB3 wird die Theorie der Kapitel 10/20/25/30/35 der [M300 Dokumentation](#) vorausgesetzt

Formale Vorgaben

Abgabetermin	<ul style="list-style-type: none"> • Sonntag 25.04.
Speicherort	<ul style="list-style-type: none"> • Repository: m300_lb (auf GitLab oder GitHub)¹ • Folder: lb3
Abgabe Objekte	<ol style="list-style-type: none"> 1. lb3.md (README.md ist auch ok) 2. Mindestens eine docker-compose.yml Datei ² 3. Dockerfile(s) - sofern verwendet 

¹ Falls Repository oder Freigabe abweichend zur LB2 sind muss das der LP per Mail mitgeteilt werden

² Name von docker-compose Datei(en) darf abweichend sein

Auftrag - Container Service (Kapitel 30 / 35)

Sie erstellen ein selbst gewähltes Projekt, welches auf der **Docker Container-Technologie** basiert.

Dabei erstellen sie einen Service, der innerhalb eines oder mehrerer Container implementiert wird.

Teamarbeit ist erwünscht. Die Implementation des **Container-Projekts** erfolgt hingegen **als Einzelarbeit**. Der erstellte Code sowie die gesamte Dokumentation wird versioniert, entweder auf [GitLab](#) oder [GitHub](#), hinterlegt und der Lehrperson zugänglich gemacht (Lese-Rechte).

Das Internet ist eine wichtige Ressource für solche Projekte. Entsprechend dürfen sie auch Codebeispiele aus dem Internet verwenden, sofern sie entsprechende Quellenangaben machen.

Der verwendete Code muss aber von ihnen vollständig dokumentiert sein. Das gilt auch für Code, welchen sie aus fremden Quellen verwenden.

Das bedeutet, sie können über den verwendeten Code Auskunft geben.

Eine Vagrant Definition ist nicht verlangt, darf aber verwendet werden.

Es gelten folgende Bedingungen:

- Vagrant Code wird nicht bewertet (das war Teil der LB2)
- Es gelten die Abgabe-Objekte der LB3
- Eine Vagrant-VM und die Docker Services müssen über "private_network", ip:"192.168.60.101" erreichbar sein

Service Anforderungen

- Wiederholbar und konsistent ausführbar auf jedem Rechner³ der Docker und Docker-Compose installiert hat.
- Startet mit ***docker-compose up*** ohne weitere Interaktion.
- Die **Entwicklungsschritte** des Codes und der Dokumentation sind in der *Git History* durch **regelmässige und dokumentierte Commit**⁴ nachvollziehbar.
- Die **Projektdokumentation** erfolgt in **Markdown**.
- Der Service kann unabhängig getestet werden, alle dazu nötige ist vorhanden.
- Die Funktion und Anwendung ist in der Projektdokumentation beschrieben.
- Sämtlicher Code ist in der Projektdokumentation beschrieben⁵.

³ Zum Verifizieren kann die Vagrant VM unter M300/Docker nach Installation von Docker-Compose verwendet werden

⁴ Commit müssen in regelmässigen Abständen über die gesamte LB3 erfolgen

⁵ Nicht Inline in den Konfigurationen

Bewertung

Das Projekt wird anhand folgender Kriterien bewertet

60% Komplexität Umfang Funktionalität Umsetzung	<ul style="list-style-type: none"> • Komplexität, Umfang und Anteil Eigenleistung der Arbeit⁶ • Auf jedem Linux <i>Docker-Enabled</i> Rechner lauffähig • Korrekte Netzwerkkonfiguration • Keine Interaktion nach dem Start nötig • Eigenes Image erzeugt und verwendet • Image wird mit docker-compose erstellt • Service benötigt mehrere Container (Bsp. Webserver mit Datenbank Backend) • Verwendung von merged Compose Files • Verwendung von persistenten Volumes • Dokumentierte Sicherheitsmerkmale aus Kapitel 35 implementiert • Der Fortschritt ist in der Git History nachvollziehbar
30% Dokumentation	<p>Es wird eine technische Dokumentation über ihr Container Projekt erwartet die alle relevanten Informationen beinhaltet. Sie beschreibt, wie der Service funktioniert, welche Rahmenbedingungen erfüllt sein müssen und wie der Service auf korrekte Funktion getestet wird. Die grafische Übersicht gibt Auskunft über verwendete Server, Applikationen und Netzwerk-Konfiguration.</p> <ul style="list-style-type: none"> • Inhalt <ul style="list-style-type: none"> ○ Inhaltsverzeichnis ○ Service Beschreibung ○ Service Anwendung ○ Grafische Übersicht ○ Code Beschreibung ○ Testen ○ Quellenangaben wo nötig • Markdown <ul style="list-style-type: none"> ○ Verwendung von min. 5 verschiedenen Markdown Elementen (etwa Tabellen, Links, Bilder usw....) • Darstellung <ul style="list-style-type: none"> ○ Optisch ansprechend ○ Übersichtlich dargestellt
10% Einhaltung der Formalitäten	<ul style="list-style-type: none"> • Freigabe auf GitLab od. GitHub erteilt • Abgabe-Termin eingehalten • Alle Abgabe-Objekte korrekt hinterlegt

⁶ Die Komplexität und Anteil Eigenleistung wird von der LP unter Berücksichtigung aller abgegebenen Arbeiten individuell beurteilt