



Abbildung 1: Logo ABS-Informatik

Realisierung

Realisierung

Version 1.0

Autor des Dokuments	Aaron Inauen, Ben Zürcher, Severin Schneider		Erstellt am	17.05.2023
Dateiname	M169_Realisierung_ABS.docx			
Seitenanzahl	22	© 2023 Aaron, Ben, Severin <i>ABS Informatik</i>		Vertraulich!

Historie der Dokumentversionen

Version	Datum	Autor	Änderungsgrund / Bemerkungen
0.1	17.05.2023	Ben Zürcher	Dokument erstellt
0.2	24.05.2023	Ben Zürcher	Kapitel erstellt
0.3	26.05.2023	Aaron Inauen	Kapitel 1 (Konfigurationsfiles) geschrieben
0.4	31.05.2023	Aaron Inauen, Ben Zürcher, Severin Schneider	Anforderungen, Screenshots der Realisierung und Migration der DB geschrieben
0.5	01.06.2023	Severin Schneider	Kapitel «Verschiedene Container» geschrieben
0.6	02.06.2023	Aaron Inauen, Ben Zürcher	Kapitel Fallback-Szenario, Reflexion und Verzeichnisse erstellt
0.7	02.06.2023	Severin Schneider	Kapitel «Migration der Daten» geschrieben
0.8	04.06.2023	Severin Schneider	Reflexion geschrieben
1.0	04.06.2023	Aaron Inauen, Ben Zürcher, Severin Schneider	Überprüfung und Abgabe

Inhalt

Historie der Dokumentversionen	2
1 Konfigurationsfiles	4
1.1 Docker Compose	4
1.2 db_password.txt	4
1.3 Dump.sh	4
1.4 Script.sh	4
2 Anforderungen	5
3 Screenshots der Realisierung	6
3.1 Altes System über Port 8080 erreichbar machen	6
3.2 Inbetriebnahme des neuen Systems	9
4 Migration der Daten	15
4.1 Migration der DB	15
4.2 Migrieren des Moodle-data-Volumen	17
4.3 Vollständigkeit der migrierten Daten	17
5 Fallback-Szenario	18
6 Verschiedene Container	18
7 Reflexion	19
7.1 Aaron Inauen	19
7.2 Ben Zürcher	19
7.3 Severin Schneider	19
7.4 Erfolge / Misserfolge	20
8 Verzeichnisse	21
8.1 Glossar	21
8.2 Literaturverzeichnis	22
8.3 Abbildungsverzeichnis	22

1 Konfigurationsfiles

Alle Konfigurationsfiles, die man zur Migration des Moodles benötigt werden, sind in unserem Git-Repository abgelegt: <https://github.com/AaronInauen/Projekt-M169>. Dies ist ein privates Git-Repository, weshalb wir die Lehrperson darauf berechtigt haben. In den folgenden Kapiteln werden die Scripts genauer beschrieben.

1.1 Docker Compose

Für die Erstellung von unserem Docker Compose haben wir zuerst angefangen einen einfachen Moodle und einen MySQL Container in Betrieb zu nehmen und diese miteinander zu verbinden. Wir hatten schon viele verschieden ähnliche Aufgaben in dieser Art in der Schule gemacht. Aus diesem Grund konnten wir dies in kurzer Zeit erstellen. Damit wir testen konnten, ob dies auch funktioniert, haben wir bereits den ersten SQL-Dump in die Datenbank geladen. Wir konnten die Daten danach im Moodle sehen.

Wie wir es in unserem Lösungsvorschlag beschrieben haben, empfehlen wir zusätzlich einen Container mit phpMyAdmin einzurichten, welches einem die Administration erleichtert. Darum haben wir dies zu unserem Docker Compose hinzugefügt.

In einem nächsten Schritt haben wir unser Docker Compose optimiert:

- Secrets: Mithilfe von Secrets müssen Passwörter nicht mehr im Klartext im Dockercompose stehen, was zusätzliche Sicherheit bietet. Die Passwörter werden in einer separaten Datei «mitgegeben».
- Networks: Durch ein selbst erstelltes Netzwerk können die Container (innerhalb dieses Netzwerkes) von anderen abgetrennt werden, was zusätzliche Sicherheit bietet.
- Ports: Der Kunde möchte das Moodle über bestimmte Ports erreichen können. Wir haben diese im Docker Compose definiert.

1.2 db_password.txt

In dieser Datei werden die Passwörter gespeichert, welche die Secrets benötigt. Das Passwort kann über diese Datei beliebig angepasst werden.

1.3 Dump.sh

Diese Datei ist ein Bash-Script, welches ein Backup (SQL-Dump) der bestehenden Datenbank erstellt. In diesem Script werden zuerst verschiedenste Variablen deklariert (z.B. Datenbankzugangsdaten, Speicherort des Dumps, etc.). In der Zeile 12 wird der SQL-Dump erstellt. Dieser Befehl verwendet die zuvor deklarierten Variablen.

1.4 Script.sh

Mit diesen drei Konfigurationsdateien:

- Docker-Compose
- db_password.txt
- dump.sh

Hätte man bereits alle «Bauteile», um das neue Moodle zu erstellen und die Daten zu migrieren. So müsste man jedoch noch viele einzelne Schritte manuell ausführen. Um dieses Problem zu umgehen haben wir ein zusätzliches Script geschrieben, welches die ganze Migration automatisieren soll. Bei diesem Script wird zuerst Docker Compose gestartet. Danach dauert es ca. 5 Minuten, bis alle Container gestartet sind. Sobald dies abgeschlossen ist, wird der SQL-Dump der alten Datenbank erstellt. Dieser wird in den neuen MySQL-Container kopiert. Wenn dies fertig ist, kann dieser in die neue MySQL-Datenbank eingefügt werden. Zum Schluss haben wir noch einen Cronjob konfiguriert. Dieser erstellt täglich um 22:00 Uhr ein Backup der Datenbank.

2 Anforderungen

Die detaillierten Anforderungen an das Projekt haben wir im Dokument ModulM158_Projektkonzept_ABS.pdf genau beschrieben. Diese Anforderungen haben wir in den Testfällen im Dokument M158_Einführung_ABS.pdf getestet und die Mängel (falls diese vorhanden waren) dokumentiert. In folgendem Abschnitt haben wir die Testfälle zusammengefasst:

Testfall-ID	Name des Testfalls
01	Container löschen ohne Datenverlust
02	Erreichbarkeit des Systems über Port 80
03	Erreichbarkeit des Systems über Port 8080
04	User Accounts auf Funktionalität testen
05	Funktionsfähiger DB-Container
06	Funktionsfähige Moodle-Instanz
07	Funktionsfähiges und fehlerfreies Moodle-Image
08	Vollständigkeit des neuen Moodles (wurden alle Daten migriert)
09	Moodle Version 4.1.2 ist installiert.

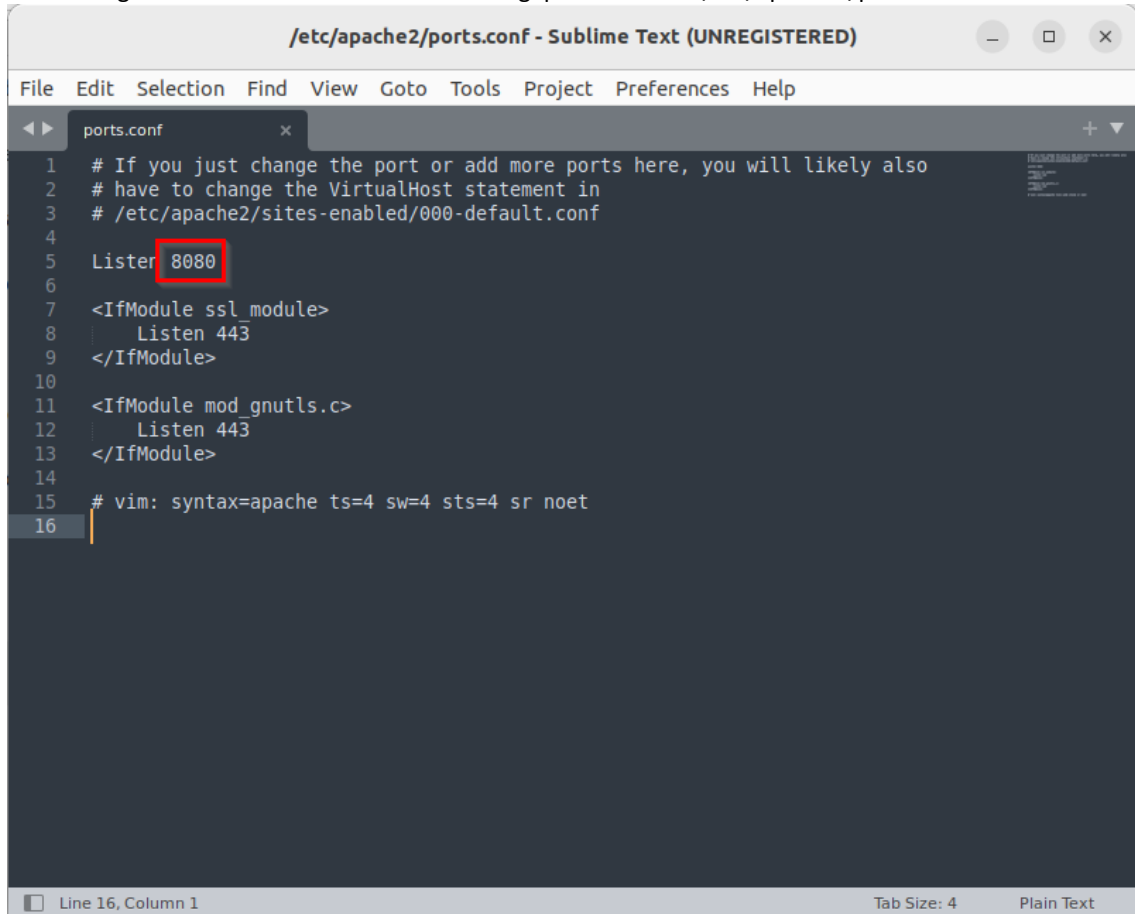
3 Screenshots der Realisierung

Bevor man mit der Realisierung beginnt, ist es wichtig, dass man entweder eine neue VM verwendet oder alte Konfigurationen, welche nicht mehr benötigt wird, löscht. So treten keine Fehler auf.

3.1 Altes System über Port 8080 erreichbar machen

Damit das alte System unter Port 8080 erreichbar wird, müssen folgende Schritt befolgt werden:

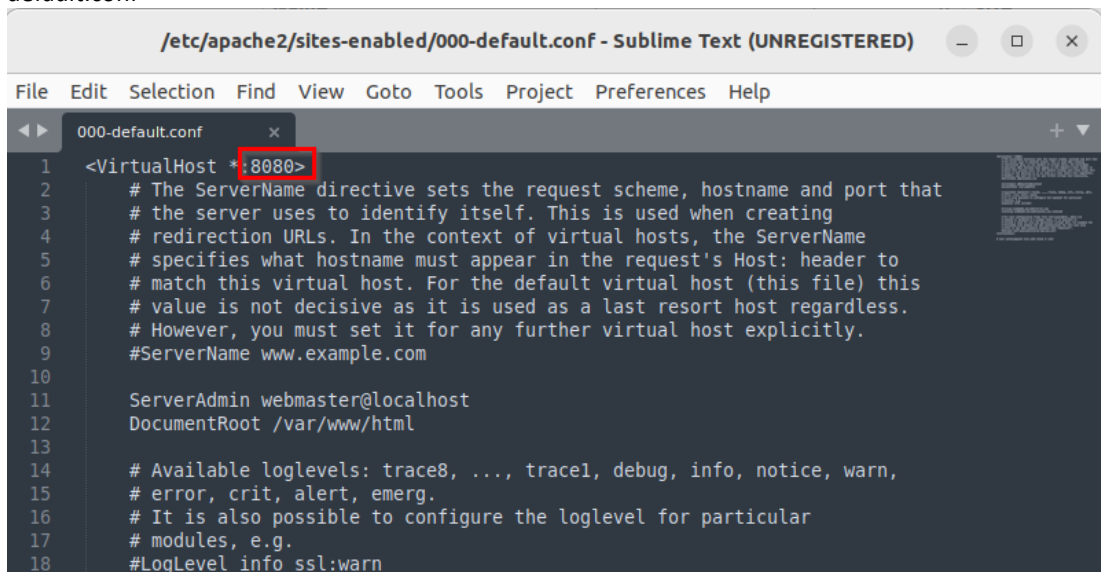
1. Bei der folgenden Datei muss Listen zu 8080 angepasst werden: `/etc/apache2/ports.conf`



```
/etc/apache2/ports.conf - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
ports.conf x
1 # If you just change the port or add more ports here, you will likely also
2 # have to change the VirtualHost statement in
3 # /etc/apache2/sites-enabled/000-default.conf
4
5 Listen 8080
6
7 <IfModule ssl_module>
8 | Listen 443
9 </IfModule>
10
11 <IfModule mod_gnutls.c>
12 | Listen 443
13 </IfModule>
14
15 # vim: syntax=apache ts=4 sw=4 sts=4 sr noet
16
Line 16, Column 1 Tab Size: 4 Plain Text
```

Abbildung 2: ports.conf

- Bei dieser Datei muss der Port ebenfalls auf 8080 geändert werden: /etc/apache2/sites-enabled/000-default.conf



```

/etc/apache2/sites-enabled/000-default.conf - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

000-default.conf x
1 <VirtualHost *:8080>
2     # The ServerName directive sets the request scheme, hostname and port that
3     # the server uses to identify itself. This is used when creating
4     # redirection URLs. In the context of virtual hosts, the ServerName
5     # specifies what hostname must appear in the request's Host: header to
6     # match this virtual host. For the default virtual host (this file) this
7     # value is not decisive as it is used as a last resort host regardless.
8     # However, you must set it for any further virtual host explicitly.
9     #ServerName www.example.com
10
11     ServerAdmin webmaster@localhost
12     DocumentRoot /var/www/html
13
14     # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
15     # error, crit, alert, emerg.
16     # It is also possible to configure the loglevel for particular
17     # modules, e.g.
18     #LogLevel info ssl:warn

```

Abbildung 3: 000-default.conf

- Sobald man dies gemacht hat, muss eine Konfigurations-Datei vom Moodle angepasst werden. Diese ist standardmässig in folgendem Pfad zu finden: /var/www/html/config.php



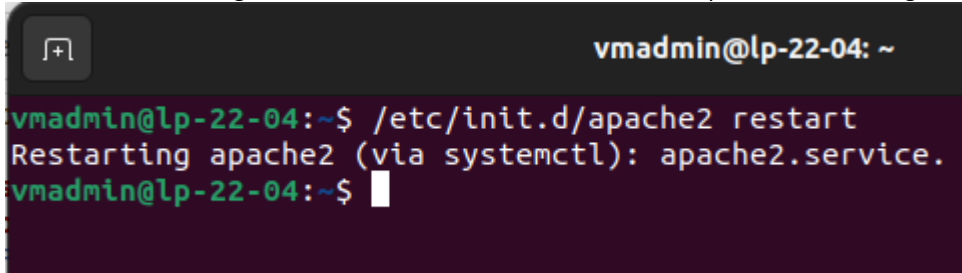
```

*config.php [Read-Only] /var/www/html
Open [+] Save
1 <?php // Moodle configuration file
2
3 unset($CFG);
4 global $CFG;
5 $CFG = new stdClass();
6
7 $CFG->dbtype      = 'mysqli';
8 $CFG->dblibrary   = 'native';
9 $CFG->dbhost      = 'localhost';
10 $CFG->dbname      = 'moodle';
11 $CFG->dbuser      = 'debian-sys-maint';
12 $CFG->dbpass      = 'vaIdfgRPSXzKbPPd';
13 $CFG->prefix      = 'mdl_';
14 $CFG->dboptions   = array (
15     'dbpersist' => 0,
16     'dbport'    => '',
17     'dbsocket'  => '',
18     'dbcollation' => 'utf8mb4_unicode_ci',
19 );
20
21 $CFG->wwwroot      = 'http://localhost:8080';
22 $CFG->dataroot     = '/var/www/moodledata';
23 $CFG->admin        = 'admin';
24
25 $CFG->directorypermissions = 0777;
26
27 require_once(__DIR__ . '/lib/setup.php');
28
29 // There is no php closing tag in this file,
30 // it is intentional because it prevents trailing whitespace problems!
31

```

Abbildung 4: config.php

4. Damit diese Änderungen auch übernommen werden, muss der Apache-Server neu gestartet werden:



```
vmadmin@lp-22-04: ~  
vmadmin@lp-22-04:~$ /etc/init.d/apache2 restart  
Restarting apache2 (via systemctl): apache2.service.  
vmadmin@lp-22-04:~$
```

Abbildung 5: Restart Apache 2

5. Nun sollte das Moodle über Port 8080 erreichbar sein: <http://localhost:8080/>

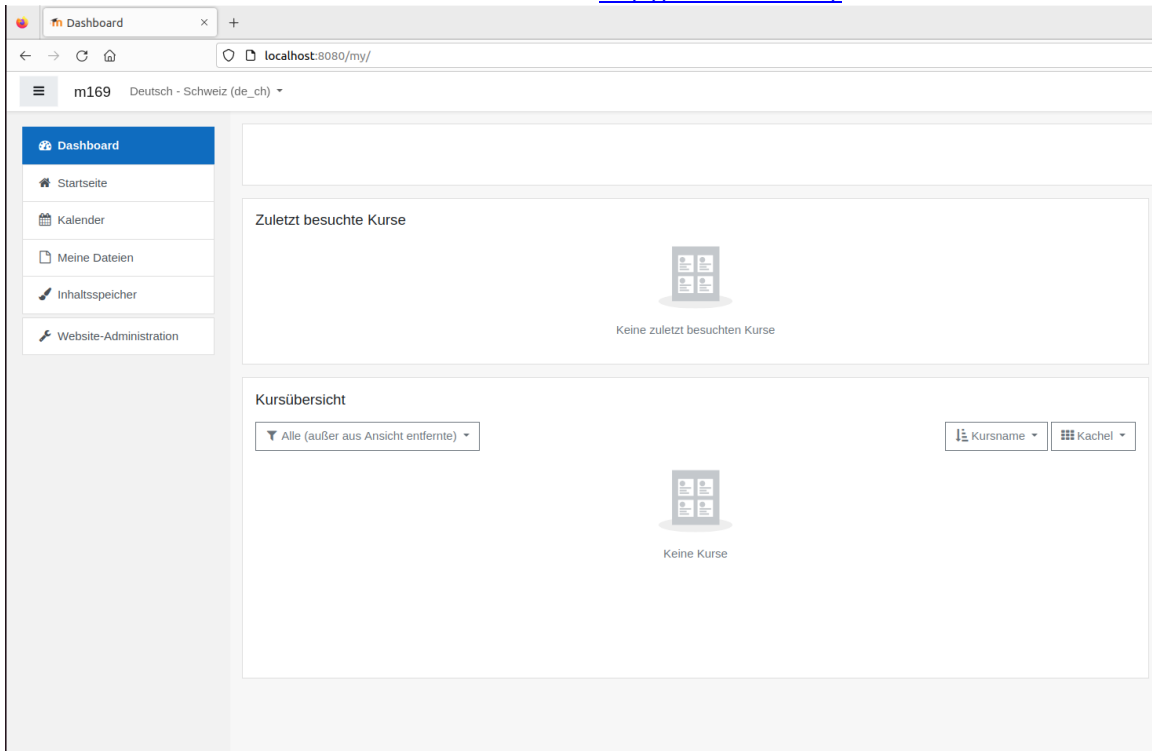


Abbildung 6: Moodle erreichbar Port 8080

3.2 Inbetriebnahme des neuen Systems

1. Folgende 4 Dateien müssen auf den (Ubuntu) Host in den Pfad /home/vmadmin/NewMoodle kopiert werden (der Ordner NewMoodle muss zuvor manuell erstellt werden):
 - db_password.txt
 - docker-compose.yml
 - Dump.sh
 - script.sh

Hinweis: Da die Dateien von Windows zu Ubuntu kopiert werden, können Probleme beim Ausführen der Datei «script.sh» auftreten. Dies kann man umgehen, wenn man den Inhalt der script.sh-Datei kopiert. Dann eine neue Datei erstellt und dort den Inhalt einfügt. Wichtig ist, dass die neue Datei denselben Namen hat und sich im gleichen Pfad wie die anderen Dateien befindet.
2. Wenn diese kopiert wurden, müssen die Berechtigungen so angepasst werden, dass alle Dateien ausgeführt werden können. Wir sind darum im Terminal in den Ordner gewechselt werden, in welchen die Dateien kopiert wurden. Danach kann man mit folgendem Befehl allen Dateien (in diesem Ordner) volle Berechtigungen geben: `chmod 777 *.*`

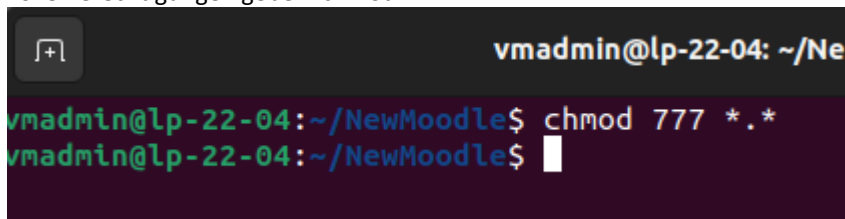


Abbildung 7: Berechtigung aller Dateien ändern

3. Damit man Dockercompose verwenden kann, muss es zuerst installiert werden. Wir haben dafür diesen Befehl im Terminal ausgeführt: `sudo apt install docker-compose`
4. Sobald Docker-Compose installiert ist, kann das script.sh mit dem Befehl `./script.sh` ausgeführt werden.
5. Während dem Ausführen des Scripts werden zwei Passwörter abgefragt. Die Passwörter lauten:
 - Docker-DB: Secret
 - Lokale-DB: Riethuesli>12345

```
vmadmin@lp-22-04:~/NewMoodle$ sudo ./script.sh
[sudo] password for vmadmin:
Bitte geben Sie das Passwort für die Docker-DB ein: Secret
Bitte geben Sie das Passwort für die lokale-DB ein: Riethuesli>12345
newmoodle_db_1 is up-to-date
pma is up-to-date
Starting newmoodle_moodle_1 ... done
```

Abbildung 8: Script ausführen

6. Es kann sein, dass das Script manchmal für ein paar Minuten stehen bleibt. Dies ist aber normal, da das Script manchmal automatisch pausiert wird, bis z.B. die Moodle-Datenbank eingefügt wird.

7. Anweisungen des Scripts befolgen. Man wird aufgefordert, den Browser zu öffnen, um dort die Moodle Datenbank zu updaten:
 - a. Moodle über Port 80 öffnen (<http://localhost:80>)

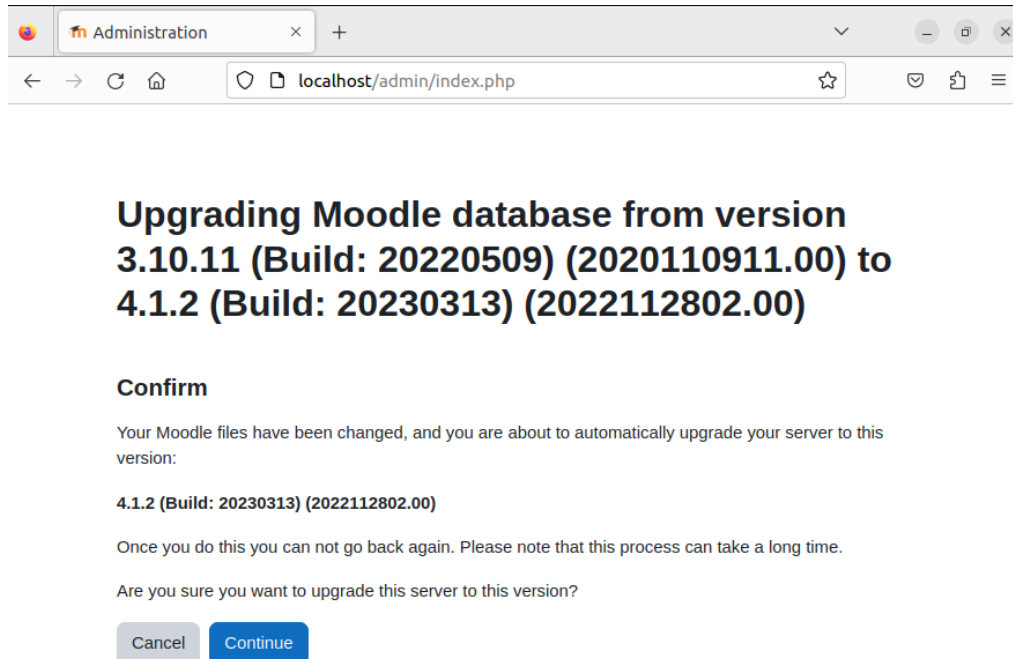


Abbildung 9: Moodle update start

- b. Mit «Continue» fortfahren.
- c. Danach ganz nach unten scrollen und wieder auf «Continue» drücken:

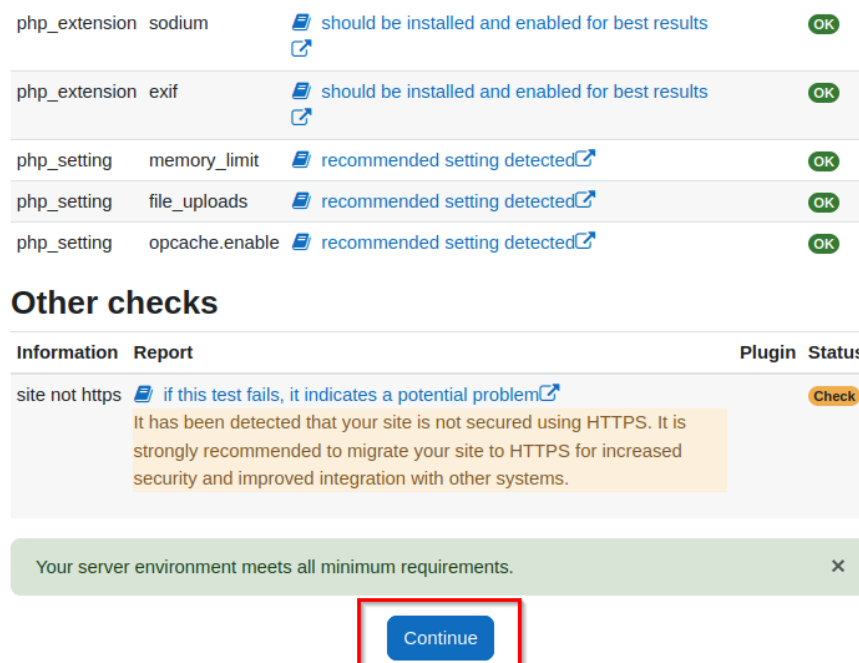


Abbildung 10: Überprüfung der Komponenten

- d. Im nächsten Fenster wieder ganz nach unten scrollen und mit «Upgrade Moodle database now» fortfahren. (Dieser Schritt kann einige Minuten in Anspruch nehmen)

H5P	2020110900	2022112800	• Moodle 2022111800	Standard To be upgraded
Themes				
Boost	2020110900	2022112800	• Moodle 2022111800	Standard To be upgraded
Classic	2020110900	2022112800	• Moodle 2022111800 • theme_boost (2022111800)	Standard To be upgraded
H5P frameworks				
H5P framework v1.24	2020110900	2022112800	• Moodle 2022111800	Standard To be upgraded
Payment gateways				
PayPal	2020110901	2022112800	• Moodle 2022111800	Standard To be upgraded

[Reload](#)

[Upgrade Moodle database now](#)

?

Abbildung 11: Datenbank upgraden

- e. Sobald unten die Schaltfläche «Continue» erscheint, kann wieder fortgefahren werden:

logstore_database

Success (0.18 seconds)

logstore_legacy

Success (0.17 seconds)

logstore_standard

Success (0.64 seconds)

[Continue](#)

?

Abbildung 12: Nach Überprüfung fortfahren

- f. Danach öffnet sich ein Anmeldefenster. Dort kann man sich mit dem Benutzernamen (vmadmin) und dem Passwort (Riethuesli>12345) anmelden:

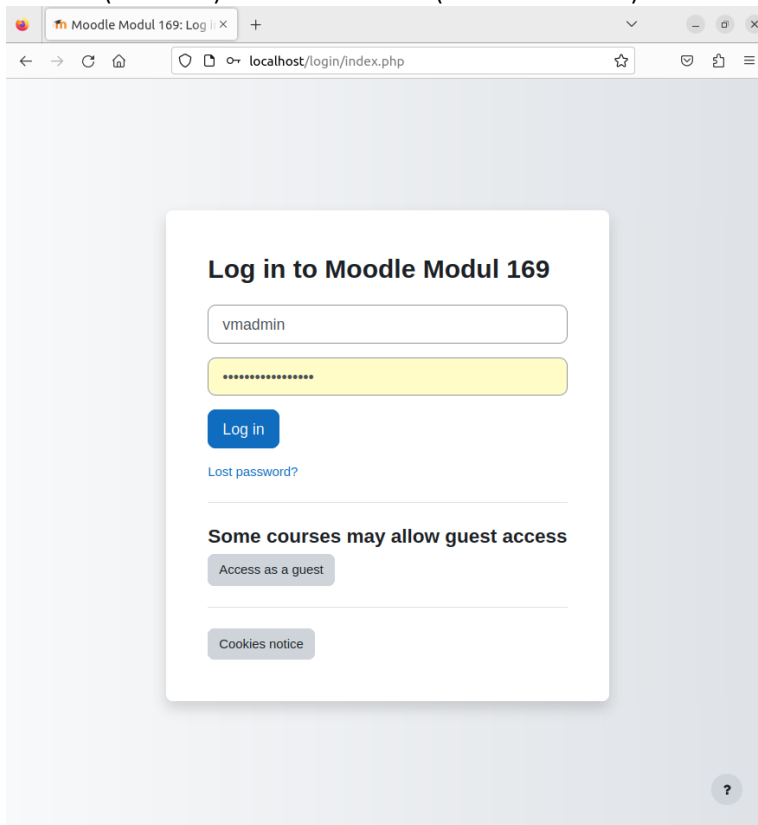


Abbildung 13: Login in Moodle

- g. Sobald man sich angemeldet hat, muss man neue Einstellungen bestätigen. Für diesen Schritt kann man auf der Seite ganz nach unten Scrollen und «Save changes» drücken.
- h. Auf der nächsten Seite muss man eine E-Mail-Adresse für den Support des Moodles eingeben und mit «save changes» bestätigen.
- i. Nun ist das Update abgeschlossen:

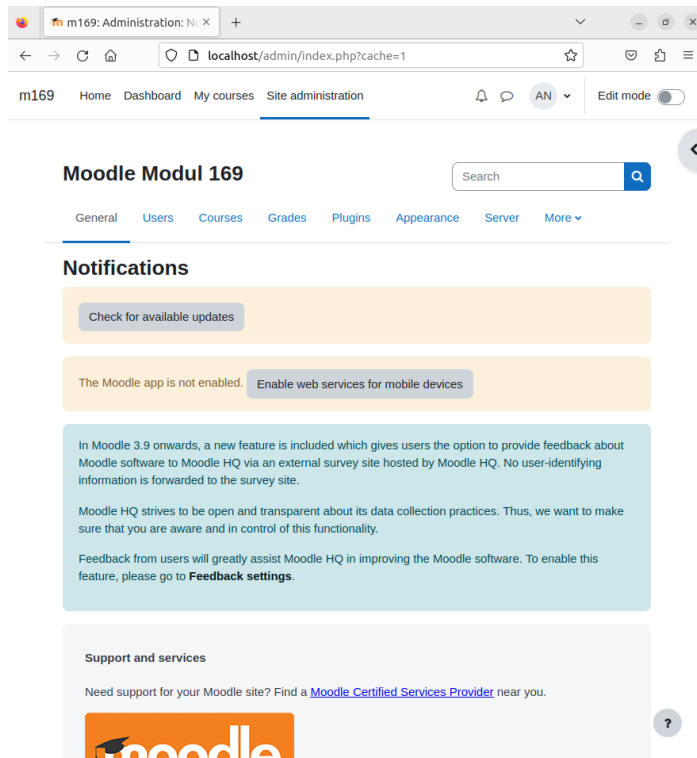


Abbildung 14: Administrationsseite Moodle

- j. Danach kann man wieder zum Terminal wechseln und das Script fortsetzen, indem man die Enter-Taste drückt.
8. Sobald man die Enter-Taste gedrückt hat, dauert es wenige Sekunden, bis das Script abgeschlossen ist.

```
Digest: sha256:ed87921184b59f7d8fc85c6a5f041c22758a4d4419c0ee3bac38eb7e133eae3
Status: Downloaded newer image for phpmyadmin/phpmyadmin:latest
Creating newmoodle_db_1 ... done
Creating pma ... done
Creating newmoodle_moodle_1 ... done
mysqldump: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
Öffnen Sie http://localhost:80 in ihrem Browser und führen Sie das Datenbankupgrade aus.
Das Skript wird pausiert. Drücken Sie Enter, um fortzufahren.

Fortsetzung des Skripts...
vmadmin@lp-22-04:~/NewMoodle$
```

Abbildung 15: Ende des Scripts

9. Man kann das Moodle über <http://localhost:80> erreichen.

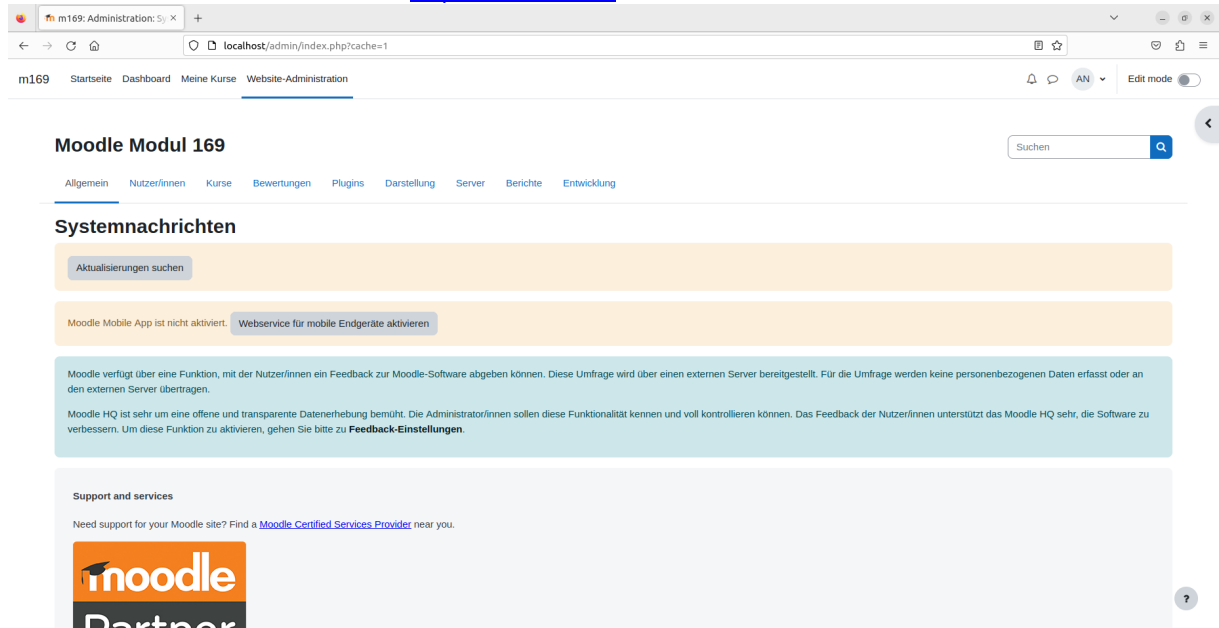


Abbildung 16: Demo Moodle erreichbar auf Port 80

4 Migration der Daten

Damit man auch auf der neuen Moodle-Version gleich weiterarbeiten kann, müssen alle Daten migriert werden. Der Hauptteil der Migration besteht aus dem Migrieren der Datenbank in die Datenbank des Containers.

Zusätzlich muss noch das Volume Moodledata (/var/www/moodledata/*) des bestehenden Moodles migriert werden, denn dieses enthält noch zusätzliche Features zB das man das Moodle auf Deutsch umstellen kann.

Die Migration all dieser Daten haben wir folgendermassen durchgeführt:

4.1 Migration der DB

Bei der Migration der DB wollten wir anfangs einen Dump der alten DB im Voraus erstellen und danach diesen Dump über unser docker-compose.yml in die neue DB importieren. Für dieses Vorhaben haben wir extrem viel Zeit aufgewendet, wir haben es zwar soweit geschafft das die DB korrekt importiert wurde, jedoch startete der Moodle Container danach nicht. Wir haben viel recherchiert und sind auf den Schluss gekommen, dass die DB erst nach der Installation des Moodle-Containers migriert werden kann.

Deshalb haben wir uns für ein zusätzliches Skript (script.sh) entschieden, welches uns genau dieses Problem und noch mehr abnimmt. In diesem File haben wir das Problem folgendermassen gelöst:

- Wir führen unser YAML aus und warten danach 200 Sekunden. damit der Moodle Container sicher erstellt und auch richtig konfiguriert wird. So sollte der alte Fehler umgangen werden.

```
7 # Docker Compose ausführen
8 docker-compose up -d
9
10 # Warten, bis die Container gestartet sind
11 sleep 200
```

Abbildung 17: Ausschnitt script.sh Sleep

- Im Anschluss wird direkt ein Dump der lokalen DB erstellt, so können wir sicherstellen das wir immer einen sehr aktuellen Dump haben.

```
13 # MySQL-Dump erstellen
14 sudo mysqldump --password=$password moodle > dump4-1-2.sql
```

Abbildung 18: Ausschnitt script.sh Dump erstellen

- Die bestehende Moodle-DB welche im YAML erstellt wurde, damit der Container starten kann wir nun gelöscht. Dies weil wir den neuen Dump importieren möchten und so sicherstellen das alles richtig funktioniert. Im Nachhinein wird die DB wieder erstellt damit der Dump importiert werden kann.

```
24 # Bestehende Moodle DB wird gelöscht und wieder erstellt
25 docker exec -i newmoodle_db_1 bash -c "mysql -u root --password=$password -e 'DROP DATABASE moodle; CREATE DATABASE moodle;'"
26
```

Abbildung 19: Neu DB überschreiben

- Der Dump wird nun in die DB geladen. Nun funktioniert die Migration der DB einwandfrei

```
# In den MySQL-Container wechseln und den Dump importieren
docker exec -i newmoodle_db_1 mysql -u root --password=$password moodle < ./dump4-1-2.sql
```

Abbildung 20: Alte DB einfügen

4.2 Migrieren des Moodledata-Volume

Um die Daten vollständig zu migrieren, muss auch das Moodledata-Volume migriert werden. In diesem sind zB Einstellungen zur Sprache gespeichert. Diese Migration haben wir folgendermassen umgesetzt:

- Im Skript (script.sh) haben wir kurz beschrieben was man im Voraus machen muss, damit diese Migration auch funktioniert. Denn bevor man die Daten kopieren kann, muss das Update durchgeführt werden, ansonsten funktioniert es nicht. Dies haben wir so gelöst, dass man während dem Ausführen folgende Meldungen bekommt, das Update macht und mit einem Enter das Skript zu Ende führt.

```
43 echo "Öffnen Sie http://localhost:80 in ihrem Browser und führen Sie das Datenbankupgrade aus."
44
45 # Ausgabe einer Nachricht
46 echo "Das Skript wird pausiert. Drücken Sie Enter, um fortzufahren."
47 S
48 # Warten auf Benutzereingabe
49 read
```

Abbildung 21: Ausschnitt script.sh warten auf Benutzereingabe

- Nach dem Update und dem Enter im Terminal bekommt man noch die Meldung das nun das Skript fortgesetzt wird. Zum Schluss wird dann noch der cp-Befehl ausgeführt, welcher die Moodle Daten kopiert.

```
51 # Fortsetzung des Skripts
52 echo "Fortsetzung des Skripts..."
53
54 # Moodle-Daten in Container verschieben
55 sudo cp -r S /var/lib/docker/volumes/newmoodle_moodledata/_data
```

Abbildung 22: Ausschnitt script.sh fortsetzen

- Zum Schluss muss man die Seite nur nochmals neu laden und sich erneut anmelden, danach wird das Moodle schon auf Deutsch angezeigt und man kann die Sprache selbst wählen.

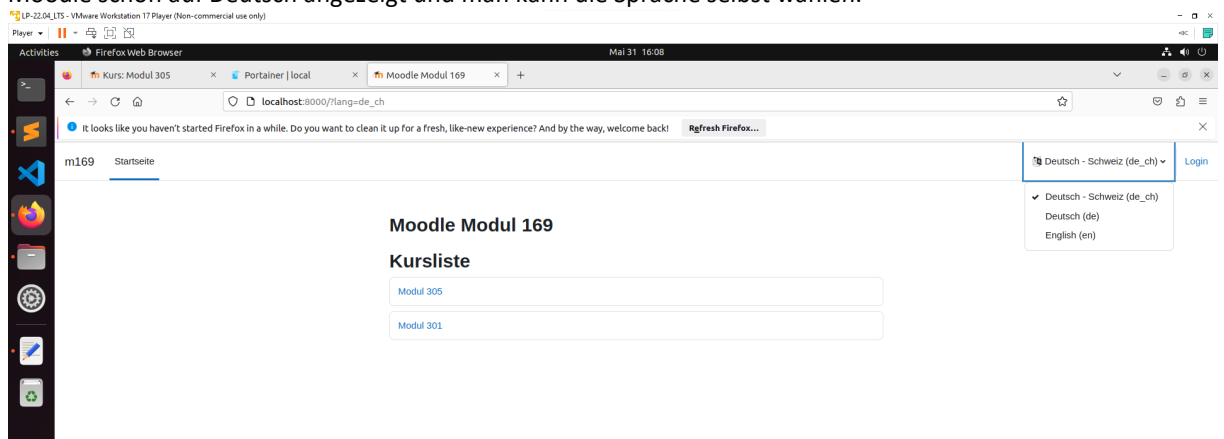


Abbildung 23: Sprache wird automatisch geändert

4.3 Vollständigkeit der Migrierten Daten

Nach all diesen Schritten der Datenmigration sind nun alle Daten komplett und korrekt migriert. Siehe dazu Testfall 08 in der Einführung

5 Fallback-Szenario

Falls bei der Migration etwas schief laufen sollte oder wir den Zeitplan nicht einhalten können, könnten die unter «3.1 Altes System über Port 8080 erreichbar machen» beschriebenen Schritte wieder rückgängig gemacht werden und die gesamte alte Umgebung wird wieder laufen.

6 Verschiedene Container

Um eine effiziente und skalierbare Umgebung für den Betrieb von Moodle zu schaffen, haben wir uns entschieden für jeden Dienst einen eigenen Container bereitzustellen. Zu den benötigten Diensten gehören das Moodle, PHPMyAdmin und die MYSQL-DB. Darüber hinaus haben wir ein zusätzliches Netzwerk namens "moodlenet" eingerichtet, um diese Container zu verbinden.

Die Aufteilung der einzelnen Services in verschiedenen Container bietet mehrere Vorteile. Erstens ermöglicht sie eine bessere Isolation und Abgrenzung der einzelnen Komponenten. Jeder Container ist eine eigenständige Einheit und kann unabhängig von anderen Containern betrieben, aktualisiert und skaliert werden. Dies erhöht die Flexibilität und ermöglicht eine einfachere Wartung und Aktualisierung der einzelnen Services.

Zusätzlich erleichtert die Verwendung von Containern die Portabilität und Konsistenz der Bereitstellung. Da jeder Service in einem eigenen Container verpackt ist, können wir ihn problemlos auf verschiedenen Umgebungen und Systemen bereitstellen, solange diese Docker unterstützen. Dies ermöglicht eine schnelle Bereitstellung und vereinfacht die Skalierung des Moodles, indem bei Bedarf weitere Instanzen des Containers gestartet werden können.

Das separate Netzwerk "moodlenet" bietet eine geschlossene Umgebung für die Kommunikation zwischen den Containern. Es isoliert den Datenverkehr und bietet zusätzliche Sicherheit, indem es den Zugriff von aussen beschränkt. Durch die Verwendung eines eigenen Netzwerks können wir auch die Netzwerkressourcen besser verwalten und Probleme mit anderen Anwendungen vermeiden, die auf demselben System laufen.

7 Reflexion

7.1 Aaron Inauen

In den vorherigen Dokumenten haben wir eine genaue Planung gemacht, was uns einen grossen Vorteil bei der Umsetzung brachte und wir schon vor der Realisierung wussten, was auf uns zukommen wird. Aus diesem Grund haben wir gleich begonnen, die ersten Container über Docker-Compose zu konfigurieren. Sobald wir einen ersten Entwurf von Docker-Compose hatten, haben wir begonnen dieses zu optimieren. Z.B. durch Secrets, Networks, etc. Zum Schluss haben wir unser Script nochmals mit den Anforderungen verglichen und die letzten Anpassungen gemacht.

Ich finde, dass uns die Umsetzung gut gelungen ist und wir ziemlich schnell einen ersten Entwurf hatten, auf welchen wir aufbauen konnten. Mir persönlich war es eine grosse Hilfe, dass wir eine gute Planung gemacht haben und uns bereits vor der Realisierung zu vielen möglichen Risiken / Schwierigkeiten Gedanken gemacht haben. Des Weiteren konnten wir wieder viel dazu lernen, wie man mit einem Git-Repository arbeitet.

Ich finde, dass es sehr schwer war nach unserem Zeitplan zu arbeiten, da wir einige Dinge in unserem Zeitplan nicht berücksichtigt haben. Ich denke, dass es sich bei einem Projekt, in welchem man ein Code schreiben muss (in diesem Fall Docker-Compose), besser eignet, einen ersten Entwurf zu schreiben (z.B. einmal alle Container installieren) und zu diesem immer wieder Dinge zu ergänzen und zu konfigurieren. Dies werde ich für ein zukünftiges Projekt auf jeden Fall mitnehmen.

7.2 Ben Zürcher

Die Umsetzung dieses Projekts haben wir meiner Meinung nach sehr gut gemeistert. Wir haben ein schönes Script erstellt, welches die meisten Aufgaben bei der Erstellung der Container und Migration der Daten übernimmt. Alle manuellen Aufgaben haben wir schön dokumentiert, sodass diese einfach vollzogen werden können.

Bis zu diesem Script mussten wir jedoch mehrere Schritte machen. Nach einem manuellen Start der Container haben wir Schritt für Schritt zuerst das Docker-Compose erstellt. Dabei haben wir auch das Secret erstellt. Danach haben wir ein Script gebaut, welches das Docker-Compose ausführt und die Daten migriert. Zum Schluss haben wir noch den Cronjob für das automatische Backup zum Script hinzugefügt.

Mich hat das Projekt technisch herausgefordert. Ich konnte mir aber immer Hilfe in der Gruppe, bei den Mitschülern, Lehrer und Internet holen. Ich habe somit viel über Docker gelernt. Mich interessiert dieses Thema und ich glaube, dass Docker eine grosse Zukunft haben wird.

7.3 Severin Schneider

Das Projekt mit dem Moodle migrieren im M169 aber auch M158 war eine sehr spannende aber auch abwechslungsreiche Aufgabe. Ich habe viele neue Sachen gelernt und mein Wissen in Docker wirklich vertiefen können und jegliche Sachen ausprobieren. So etwas in der Art macht mir persönlich sehr viel Spass.

Die Umsetzung war zwar im ersten Moment nicht sehr einfach, denn wir haben zwar schon ähnliche Übungen gemacht aber so etwas noch nie. Erst einmal all die Konfigurationen zu kennen war nicht leicht und dann auch die unerwarteten Fehler wie das beim direkten Import der DB das Moodle nicht mehr startet, dies hat mich fast zum Verzweifeln gebracht. Dennoch habe ich all diese Probleme irgendwie lösen können. Nun denke ich ist uns die Umsetzung wirklich gut gelungen und es macht wieder Spass zusehen was wir alles gemacht habe. Auch der hohe Automatisierung Grad ist immer wieder eine Herausforderung, aber wenn es funktioniert es super. Was mir nicht so gepasst hat an der Realisierung, dass es manchmal ein wenig realitätsfern ist. Heisst z.B. das Migrieren der Daten, das man von uns erwartet das es automatisiert wird. Dies bringt sehr viel neue Arbeit, welche man bei einer Einmaligen Migration einfach von Hand machen würde und sich so viel Zeit einspart. Es war allgemein ein riesiger Aufwand.

7.4 Erfolge / Misserfolge¹

Erfolge:

- Erste Vorlage für das Dokument
- Ein Compose-File welches grundlegend funktioniert
- Script für automatische Erstellung des Dumps geschrieben
- Secrets und PHPmyadmin eingefügt
- Daten waren konnten migriert werden
- Script für automatische Erstellung des Dumps geschrieben und Cronjob erstellt
- Ein funktionierendes Script welches ein Grossteil des Projektes automatisiert.
- Alle Daten aus dem Ordner Moodledata wurden komplett migriert und im Script automatisiert.
- Beide Moodle unter dem richtigen Port erreichbar
- Ganzes Projekt fertiggestellt
- Sämtliche Lieferobjekte wurde rechtzeitig abgegeben

Misserfolge:

- Es funktioniert nicht, dass man den Dump, bzw. Moodle-DB erstellt im Skript. Denn der Moodle Container geht dann immer auf Exited, es funktioniert nur dann, wenn man den Dump im Nachhinein händisch hineinkopiert. (funktioniert auch nicht, wenn man ein Dockerfile erstellt und dort den Dump schon lädt)
- Moodledata richtig migrieren (es konnte erst nach dem Update kopiert werden)
- Falscher Zeitstempel auf SQL-Dump -->Zeit wird in UTC angegeben. Damit das ersichtlich ist, habe ich die Zeitzone im Dateinamen ergänzt
- Datenbank musste zuerst auf eine Zwischenversion geupdatet werden, bevor die neusten Updates installiert werden könnten
- Geeignete Tests für die Testfälle zu finden
- Daten konnten nicht hineingeladen werden (es sind immer verschiedenen Fehlermeldungen erschienen und wir konnten es nicht lösen)
- Ich habe es nicht ganz einfach gefunden, den Cronjob zu erstellen. Mein Fehler war, dass ich bei der Erstellung des Jobs kein sudo verwendet habe

¹ Erfolge und Misserfolge wurden aus den Arbeitsjournalen der Gruppe ABS übernommen

8 Verzeichnisse

8.1 Glossar

Begriffe	Definition
Apache-Server	Ein Apache-Server ist ein weit verbreiteter Open-Source-Webserver, welcher es ermöglicht Webanwendungen bereitzustellen.
Container	Ein Container enthält alle Informationen, welche benötigt werden, um eine Anwendung auszuführen.
Cronjob	Ein Cronjob führt eine Aktion zu einem bestimmten Zeitpunkt in unixartigen Betriebssystem (z.B. Ubuntu) aus.
Docker	Docker ist eine Open-Source-Plattform, welche es ermöglicht, Container zu erstellen, zu verteilen und auszuführen.
Docker-Compose	Docker-Compose ist ein Tool zum Definieren und Ausführen von Docker-Anwendungen mit mehreren Containern
Git-Repository	Ein Git-Repository ist ein Verzeichnis, welches eine verteilte Versionsverwaltung von Dateien ermöglicht.
Migration	Migration, bezogen auf die Informatik, meint den Umzug von Daten, Anwendungen, System oder ganze IT-Infrastrukturen auf eine andere Plattform.
PhpMyAdmin	PhpMyAdmin ist ein webbasiertes Werkzeug, welches die Administration einer Datenbank erleichtert.
Secrets	Mithilfe von Secrets müssen Passwörter nicht mehr im Klartext im Dockercompose stehen, was zusätzliche Sicherheit bietet. Die Passwörter werden in einer separaten Datei «mitgegeben».
SQL-Dump	Ein SQL-Dump ist ein Backup einer SQL-Datenbank. In der Regel ist dieses Backup als .sql-Datei gespeichert und enthält somit die SQL-Statements, welche die Datenbank wieder aufbauen können.

8.2 Literaturverzeichnis

Aaron Inauen, B. Z. (12. 03 2023). *Projekt_M346 GIT*. Von https://github.com/AaronInauen/Projekt_M346/blob/main/Projektdokumentation.md abgerufen

GBSSG. (2023). *GitLab M158*. Von <https://gbssg.gitlab.io/m158/> abgerufen

Open AI. (2023). *Chat GPT*. Von <https://chat.openai.com/chat> abgerufen

Q-Centric GmbH. (12. 03 2023). *Software Testing/Qualitätssicherung - Alle Methoden und Tools*. Von <https://q-centric.com/blogs/software-qualitaetssicherung-alle-methoden-und-tools> abgerufen

Wikipedia. (12. 03 2023). *Bild ABS Logo*. Von https://en.wikipedia.org/wiki/Antilock_braking_system#/media/File:Antilock_Braking_System.svg abgerufen

8.3 Abbildungsverzeichnis

Abbildung 1: Logo ABS-Informatik	1
Abbildung 2: prots.conf	6
Abbildung 3: 000-default.conf	7
Abbildung 4: config.php	7
Abbildung 5: Restart Apache 2.....	8
Abbildung 6: Moodle erreichbar Port 8080	8
Abbildung 7: Berechtigung aller Dateien ändern	9
Abbildung 8:Script ausführen.....	9
Abbildung 9:Moodle update start.....	10
Abbildung 10: Überprüfung der Komponenten	10
Abbildung 11: Datenbank upgraden	11
Abbildung 12: Nach Überprüfung fortfahren.....	11
Abbildung 13: Login in Moodle	12
Abbildung 14: Administrationsseite Moodle	13
Abbildung 15: Ende des Scripts	13
Abbildung 16: Demo Moodle erreichbar auf Port 80.....	14
Abbildung 17:Ausschnitt script.sh Sleep	15
Abbildung 18: Ausschnitt script.sh Dump erstellen.....	15
Abbildung 19: Neu DB überschreiben	15
Abbildung 20: Alte DB einfügen	16
Abbildung 21: Ausschnitt script.sh warten auf Benutzereingabe	17
Abbildung 22: Ausschnitt script.sh fortsetzen	17
Abbildung 23: Sprache wird automatisch geändert.....	17