Studienprojekt Dokumentation für das Projekt BuchladenTester

# Wie man das Projekt aufsetzt

## 1.1 Mit Interner Datenbank und Docker aufsetzen

Zuerst muss Docker Installiert werden. Hierfür bitte an die offizielle Docker Anleitung halten.

Wenn Docker installiert ist kann man mit dem einfachen Kommando Docker die verfügbaren Kommandos sehen welche mit Docker möglich sind und Kontrollieren ob Docker richtig installiert ist.

Wenn nun das folgende Kommando ausgeführt wird, wird das Notwendige Image automatisch vom Docker hub gepullt und gestartet.

Docker run --name buchladentesterdev –p 8080:8081 gerry1313/buchladentesterdev

Hiermit wird die Anwendung automatisch gestartet und läuft auf dem Localhost, mit dem Port welcher auf der linken Seite des –p Kommandos verwendet wird. Die rechte Seite des –p Kommandos, zeigt den internen Port und muss in diesem Fall auf 8081 zeigen da auf diesem die Anwendung läuft.

--name \*\*\* ist optional und ändern denn Containernamen welcher von Docker angezeigt wird und gearbeitet wird.

Laufende Container können mit docker ps angezeigt werden.

Um mit sich mit Laufenden Container wieder zu verbinden wird docker attach „Containername“ verwendet.

## 1.2 Mit externer Datenbank, Docker-Compose und Docker aufsetzen

Diese Installation unterscheidet sich in dem zur vorherigen einzig allein darin, dass in diesem Fall auf eine externe MYSQL Datenbank verwendet wird anstatt einer internen Datenbank.

Hierfür wird wie zuvor auch Docker benötigt und sollte dafür von der offiziellen Seite installiert werden.

Zusätzlich wird noch docker-compose verwendet wodurch es möglich wird unsere Programm spezifischen Parameter in einer YML Datei aufzuschreiben und auszuführen, wodurch lästige Kommandozeilen Parameter wegfallen. Docker-compose sollte ebenfalls von der offiziellen Seite installiert und auf Verwendbarkeit überprüft werden.

Für die Ausführung wird allerdings noch eine docker-compose.yml File benötigt, welche um es einfacher zu machen genau diesen Namen tragen sollte.

Falls eine spezifischerer Pfad und Name angegeben werden möchte, muss auf das –f Kommando zurückgegriffen werden wie im Folgenden zu sehen ist und im Anschluss das Start Kommando angefügt werden um den Code in der Datei auszuführen.

docker-compose -f .\Pfad\Name.yml up

Als alternative wechselt man direkt in den entsprechenden Pfad wo eine docker-compose.yml Datei liegt und führt: docker-compose up aus.

Zum Herunterfahren und Entfernen aller gestarteten Images wird einfach: docker-compose down ausgeführt.

## Erstellung eines eigenen Docker Images

# Verwendung des laufenden Programms

Es gibt einen Endpunkt welcher dem Nutzer zusätzliche Informationen zur Verfügung stellt, ohne dass dieser den Code sehen muss. Dieser beinhaltet Swagger Inklusive UI welcher unter Folgender Adresse aufgerufen werden kann.

<http://localhost:8080/swagger-ui.html#/>

Swagger beschreibt hierbei jegliche APIs und macht es unteranderem Möglich diese direkt im Browser auszuprobieren.

# Die Anwendung

Die Anwendung lässt einen Admin Nutzer sowohl Bücher als auch Autoren für Bücher anlegen, wobei normale User hier keinen Zugriff darauf haben.

Alle Nutzer können sich Bücher in ihren Einkaufswagen legen und diesen ebenfalls „Kaufen“, wodurch die entsprechenden Bücher aus dem Einkaufswagen entfernt und dem Nutzer hinzugefügt werden.

Nutzer die nicht Angemeldet sind können zwar sehen welche Bücher verfügbar sind allerdings diese nichts in einen Einkaufswagen legen oder auf den Nutzerbereich Zugreifen.

Die Kaufaktion wurde extra durch eine Pause erweitert um einen Schwerlaufenden Prozess zu Simulieren.

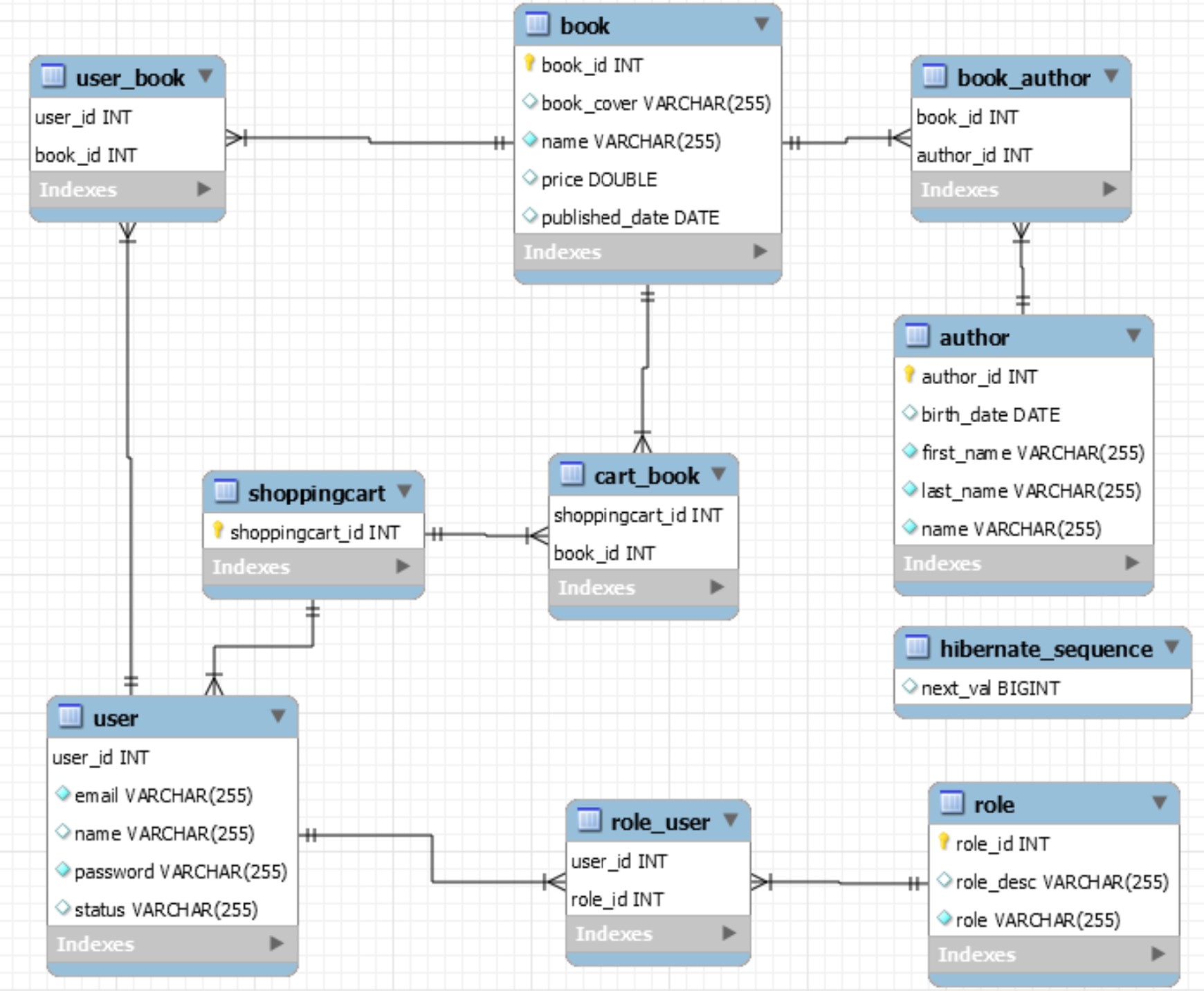
Falls Bücher von einem Admin entfernt werden, werden diese ebenfalls aus allen Einkaufswägen und Nutzerlisten entfernt.

## Aufbau

Spring Boot 2.0, Intellij, H2, MH2, Mysql, Thymeleaf, fragments, ManyToMany Problems, Registration(automatic login, Login Post, application.properties, Dockerfile, docker-compose file)

Die Anwendung Basiert auf Spring Boot 2.0

Architektur



User, Rollen