

Prof. Benjamin Tannert

MI Datenbankbasierte Webanwendungen

Sommersemester 2022

Autoren: Stephane Dongmo, Mehmet Kaplan, Markus Wechsel

Studiengang: IS Medieninformatik

Fachsemester: 4. Semester  
  
Matrikelnummern: 5148680, 5148440, 5148693

Inhaltsverzeichnis

Einleitung 3

Ideenfindung 3

Projektteams 3

Backend- Übersicht 3

Datenbank Entitys 4

Nutzung unserer Datenbank 4

Schwierigkeiten 5

Frontend- Übersicht 5

Schwierigkeiten 5

Konzept 7

# Einleitung

Datenbanken sind seit langem in fast allen Bereichen, in denen Daten gesichert werden müssen, im Einsatz. Jedes Kassensystem, jeder Bankautomat und jeder Onlineshop greifen auf eine Datenbank zu. Dort werden nicht nur Kundeninformationen gespeichert werden, sondern alles Mögliches. Für das Modul Datenbankbasierte Webanwendungen haben wir uns entschieden, einen Onlineshop aufzusetzen. Dieser sollte verschiedene Funktionen haben, welche hier dokumentiert wurden.

# Ideenfindung

Onlineshops sind in der aufkommenden Zeit gefragter denn je. Jeder Tante-Emma-Laden, hat die Möglichkeit seine Produkte nicht international zu verkaufen, sondern auch einen größeren Kundenstamm aufbauen. Des Weiteren hat die aktuelle Coronapandemie uns gezeigt, wie schnell ein Lockdown den lokalen Handel einschränken und wenn nicht sogar stilllegen. Unsere Entscheidung viel explizit auf die Wahl eines Onlineshops, weil wir im Modul Programmieren zwei einen Online-Shop mit Java programmiert haben und somit unsere Kenntnisse vertiefen konnten. Wir haben zunächst keine Branche gewählt, doch schließlich uns für die Elektro-Branche entschieden.

Mit Hilfe der Vorlesungen und Laboren, wollten wir den Onlineshop mit den Frameworks Angular, Spring Boot und MySQL aufsetzen.

# Projektteams

Frontend-Entwicklung Markus Wechsel und Stephane Dongmo

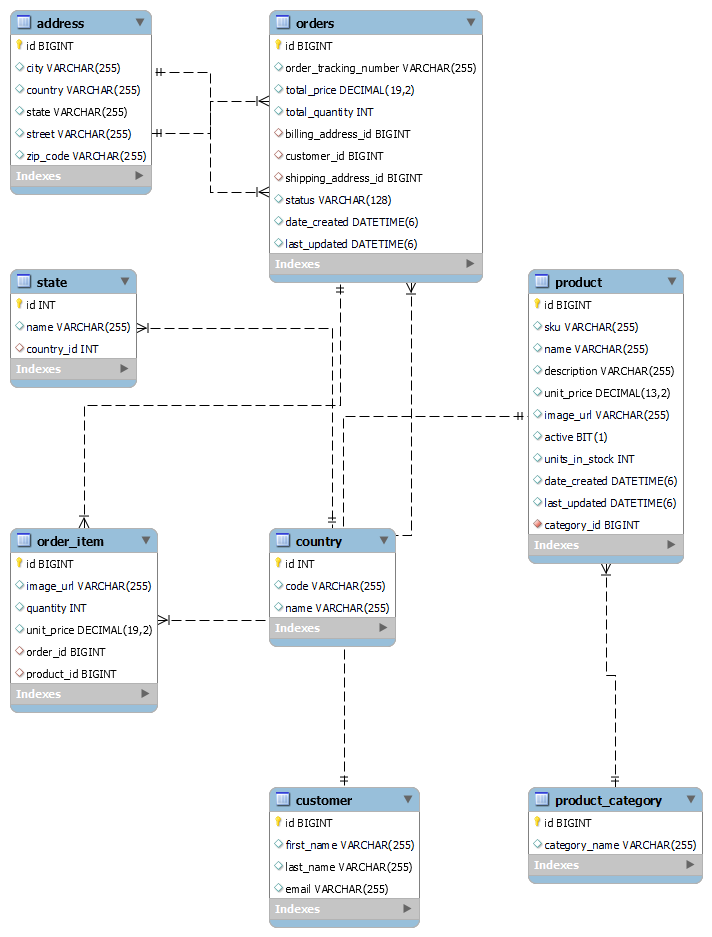
Backend-Entwicklung Mehmet Kaplan und Stephane Dongmo

Dokumentation Markus Wechsel, Mehmet Kaplan und Stephane Dongmo

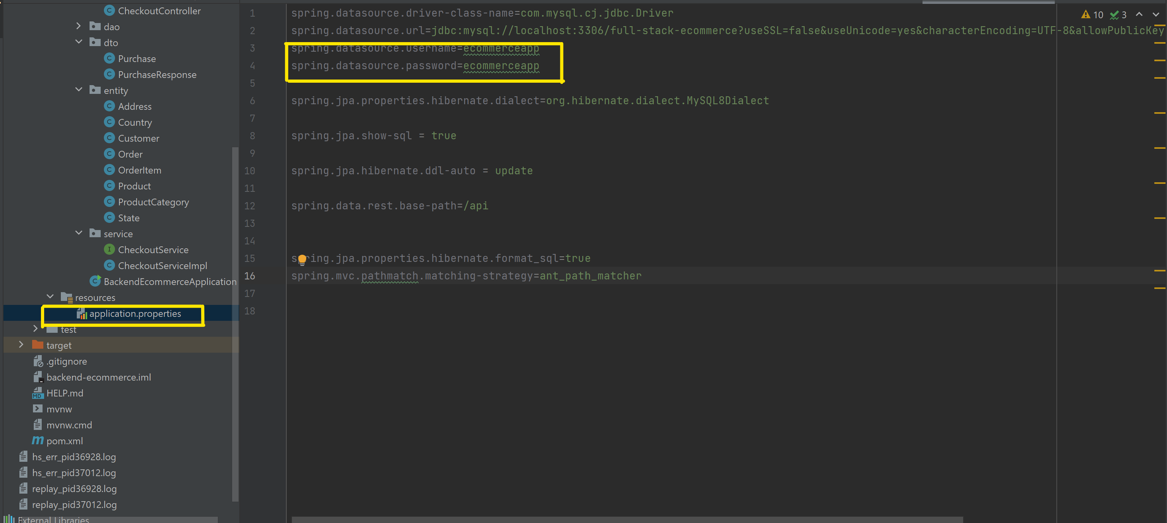
# Backend- Übersicht

Wir haben uns entschieden, unser Backend-Projekt mit Spring Boot zu entwickeln, weil es eines der besten Frameworks ist und uns die Arbeit erleichtert, indem es den Zugriff auf die Datenbank selbst verarbeitet. Im Backend haben wir eine sehr einfache Struktur implementiert, indem wir unseren API-Rest aus dem Data Access Object (dao oder Repository) senden. Wir haben den Service und den Controller nur für die Klasse CheckOut verwendet. Die Klasse CheckOut enthält einen Customer, ShippingAddress, BillingAddress, Order und OrderItems.

### Datenbank Entitys



### Nutzung unserer Datenbank



Um unserer Datenbank nutzen zu können, müssen in dem markierten Feld eingetragen werden.

### Schwierigkeiten

Aus zeitlichen Gründen haben wir es nicht mehr geschafft, eine Methode zu schreiben, welche neue Produkte in die Datenbank hinzufügt.

# Frontend- Übersicht

Das Frontend wurde mit Angular entwickelt. Angular ist eine Typescript-Framework und SPA („Single Page Application“) was uns ermöglicht mit unterschiedlichen Komponenten zu arbeiten.

Wir haben für die Umsetzung folgende Technologien benutzt:

* Models, welche die Klassen und Schnittstellen für die Typisierung sind.
* Services, welche als Grundlage für die Kommunikation zwischen den Komponenten dient.
* Komponenten ( Logik und Viev)
* Guards, welche für ein sicheres Routing zuständig sind.

### Schwierigkeiten

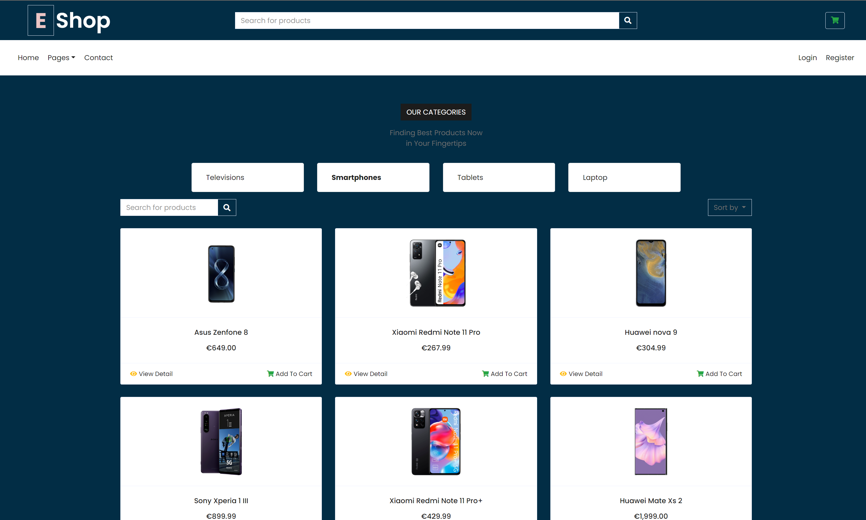
Aus Zeitgründen haben wir die Benutzer-Registrierung nicht mit einbezogen. Um sich anmelden zu können, haben wir im AuthService einen Benutzer in einem Array definiert und ihm Rollen zugewiesen. Als Zugriffslink auf unsere Datenbank haben wir die folgenden Uris: ProductService: - Http.GET liefert eine Liste von Produkten anhand einer

| HTTP-Status | Access | Beschreibung |
| --- | --- | --- |
| <http://localhost:8080/api/products/search/findByCategoryId?id=1> | Http.GET | liefert eine Liste von Produkten anhand einer ID. |
| <http://localhost:8080/api/products/search/findByNameContaining?name=Huaw> | Http.GET | gibt die Liste der Produkte nach Keyword (Search Option) zurück. |
| <http://localhost:8080/api/products/1> | Http.GET | gibt ein Produkt nach der angegebenen Id zurück. |
| <http://localhost:8080/api/products/search/findByCategoryId?id=1&page=0&size=2> | Http.GET | gibt die Produktliste anhand der Id , der Seite und der Größe zurück. |
| <http://localhost:8080/api/products/search/findByCategoryId?id=1&page=0&size=2> | Http.GET | leitet die Produktliste nach Keyword, page und size weiter. |
| <http://localhost:8080/api/products/4> | Http.delete | löscht ein Produkt anhand der Id. Diese Methode wird vom Administrator verwendet!. |
| <http://localhost:8080/api/product-category> | Http.GET | gibt die Kategorien zurück. |
| <http://localhost:8080/api/countries> | Http.GET | Liste der Städte |
| <http://localhost:8080/api/states/search/findByCountryCode?code=DE> | Http.GET | Gibt eine Liste von Country mit dem Code gleich DE zurück. |
|  |  | fügt eine neue Bestellung in die Datenbank ein, d. h. einen Costumer, eine Order und ein Order\_item. |

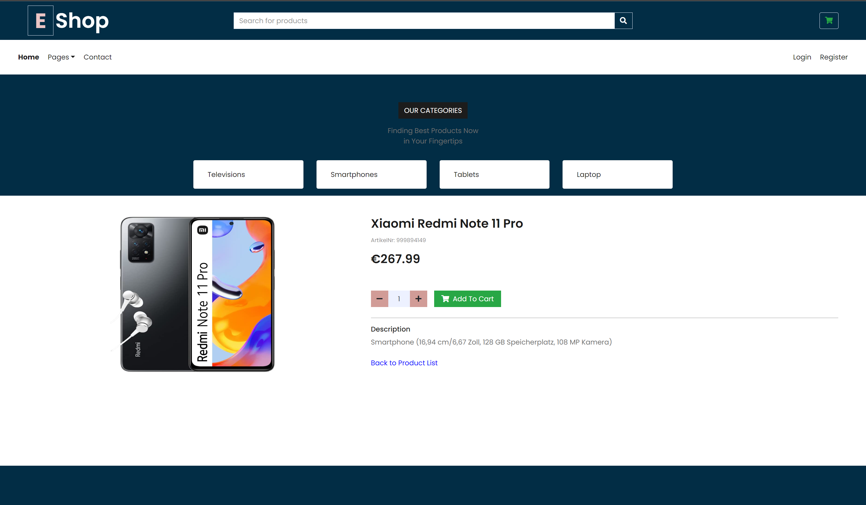
# Konzept

Es sollten folgende Dinge im Shop zu finden sein.

* Verschiedene Seiten im Shop, damit sich ein Kunde besser orientieren kann.
* Eine Suchfunktion, damit ein Kunde gezielt nach Produkten suchen kann.
* Ein Warenkorb, wo die Produkte erworben werden können.
* Ein Login, damit sich registrierte Nutzer einloggen können.
* Eine Navigationsleiste, wo die verschiedenen Kategorien ausgewählt werden können.
* Innerhalb einer Kategorie arbeiten wir mit einer Pagination, was dem Nutzer ermöglicht nur zehn Produkte auf einer Seite anzuzeigen.
* Die Produkte und eine detaillierte Ansicht der Artikel.

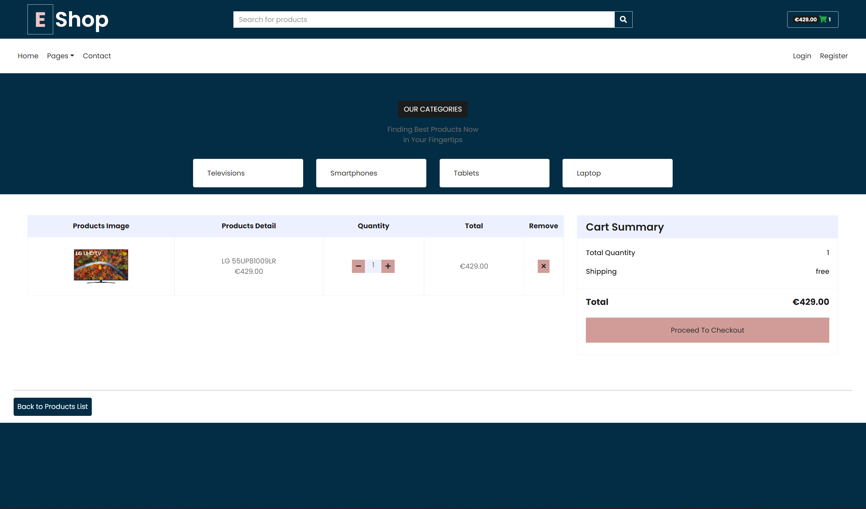


Der Onlineshop ist für jeden Nutzer frei anschaubar. Der Nutzer kann, ohne sich zu registrieren, den Onlineshop durchstöbern, die Produkte betrachten und Artikel in den Warenkorb hereinlegen. Er ist somit nicht gezwungen, sich erstmals zu registrieren, damit er einen genaueren Einblick haben kann. Um eine weite Information zu einem Produkt zu bekommen, kann der Kunde auf das gewünschte Produkt klicken.

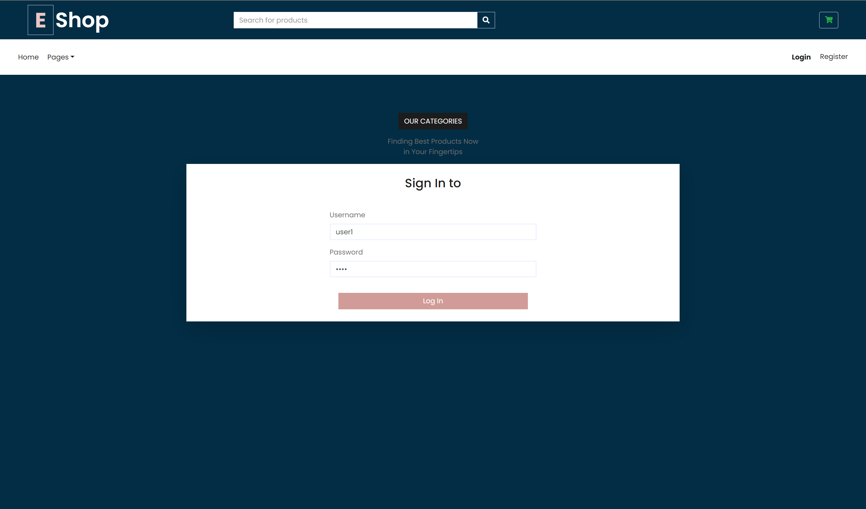


Hier kann der Kunde nicht nur die Informationen von einem Produkt erfassen, sondern die gewünschte Menge in den Warenkorb legen. Das erleichtert dem Kunden den Einkauf.

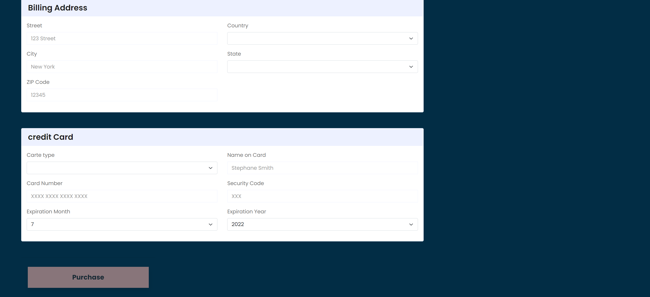
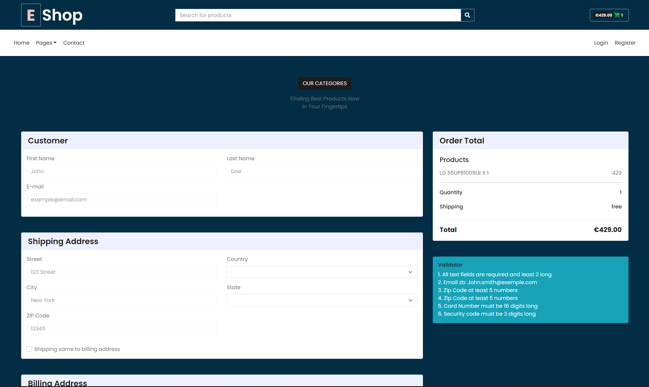
Im Warenkorb können Artikel entfernt werden und die Menge anpassen. Ist das erfolgt, kann der Kunde weiter zum Check-out.



Jedoch muss der Kunde eingeloggt sein, um in den Check-out zu kommen. Sollte dies man nicht eingeloggt sein, kommt der Kunde in den Login-Bereich.

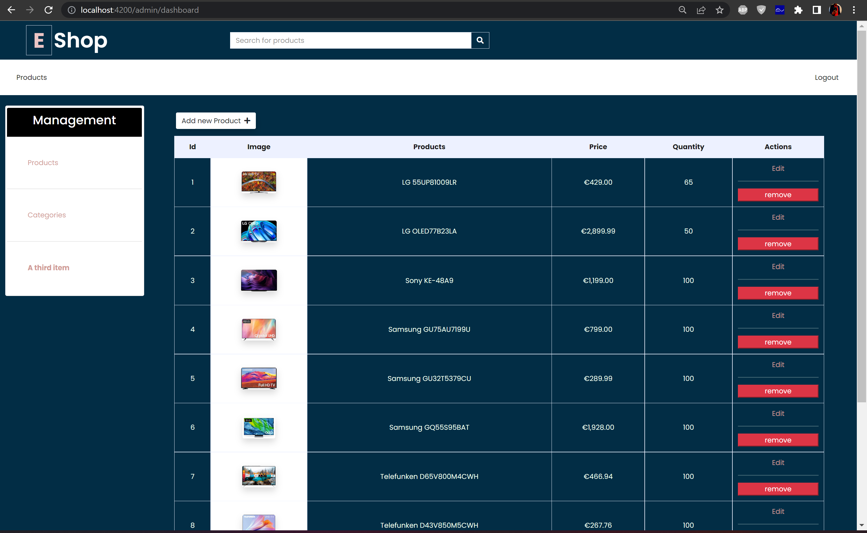


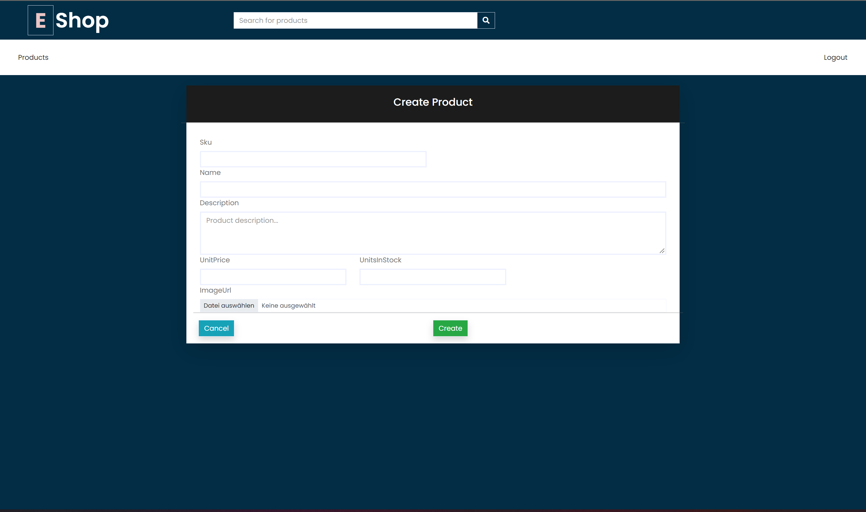
Für den Test existieren aktuell zwei Benutzer „admin“ mit dem Passwort „1234“ und „user“ mit dem Passwort „1234“. Der Benutzer wird, sobald er eingeloggt wird weitergeleitet, um den Bestellvorgang abzuschließen.



Sobald die Bestellung erfolgreich war, wird automatisch die Tracking-Nummer vom Packet generiert und über das Fenster ausgegeben.

Der Admin hat selbstverständlich eine andere Oberfläche, wo man Produkte hinzufügen, entfernen und bearbeiten kann.





Um die Sicherheit vom Admin zu garantieren, kann ein Benutzer nicht ohne Admin Rechte darauf zugreifen. Auch wenn die URL vom Admin aufgerufen wird, muss sich ein Benutzer erst mit den konkreten Daten einloggen.