

ShopNook: E-Commerce Warenkorbanwendung mit Spring Boot

ShopNook: Development of an e-commerce shopping cart application with Spring Boot

Imad-Eddine Abdessami, Mohammed Salih Mezraoui

Bachelor-Projektarbeit

Betreuer: Prof. Dr. Andreas Lux

Ort, DD.MM.YYYY

Kurzfassung

In der Kurzfassung soll in kurzer und prägnanter Weise der wesentliche Inhalt der Arbeit beschrieben werden. Dazu zählen vor allem eine kurze Aufgabenbeschreibung, der Lösungsansatz sowie die wesentlichen Ergebnisse der Arbeit. Ein häufiger Fehler für die Kurzfassung ist, dass lediglich die Aufgabenbeschreibung (d.h. das Problem) in Kurzform vorgelegt wird. Die Kurzfassung soll aber die gesamte Arbeit widerspiegeln. Deshalb sind vor allem die erzielten Ergebnisse darzustellen. Die Kurzfassung soll etwa eine halbe bis ganze DIN-A4-Seite umfassen.

Hinweis: Schreiben Sie die Kurzfassung am Ende der Arbeit, denn eventuell ist Ihnen beim Schreiben erst vollends klar geworden, was das Wesentliche der Arbeit ist bzw. welche Schwerpunkte Sie bei der Arbeit gesetzt haben. Andernfalls laufen Sie Gefahr, dass die Kurzfassung nicht zum Rest der Arbeit passt.

Inhaltsverzeichnis

1	Einleitung und Problemstellung	1
1.1	Überblick	1
1.2	Globale Anforderungen	1
1.2.1	Funktionale Anforderungen	1
1.2.2	Nicht-funktionale Anforderungen	2
1.2.3	Technische Anforderungen	2
2	Methodologie und Systementwurf	3
2.1	Entwurf und Konzeption	3
2.1.1	UML	3
2.1.2	Klassendiagramm	4
2.1.3	Use-Case Diagramme	4
2.2	Front-End Technologien	4
2.2.1	HTML/CSS	4
2.2.2	JavaScript	4
2.2.3	Angular	4
2.3	Back-End Technologien	5
2.3.1	Java	5
2.3.2	REST-API	5
2.3.3	Spring Boot	6
2.4	Datenbankstruktur	8
2.4.1	Übersicht über relationale Datenbanken und Auswahl von MySQL	8
2.4.2	Datenbankschemaentwurf mit Spring Data JPA	8
2.4.3	Datenpersistenz und Transaktionen	9
2.4.4	MySQL-Workbench und SQL	9
2.4.5	Datenbankverbindung und -konfiguration in Spring Boot	9
2.5	Entwicklungsumgebung und Versionskontrolle	10
2.5.1	Entwicklungsumgebung	10
2.5.2	Versionskontrolle	10

3	Implementierung	12
3.1	Startseite	12
3.2	Online-Shopping	12
3.3	Bezahlprozess	13
4	Ergebnisse und Analyse	14
5	Zusammenfassung und Ausblick	15
	Literaturverzeichnis	16
	Glossar	18
	Selbstständigkeitserklärung	19

Abbildungsverzeichnis

Tabellenverzeichnis

Einleitung und Problemstellung

Hier werden folgende Aspekte berücksichtigt:

- Relevanz für ein E-Commerce Shop.
- Aktueller Stand im Markt: Welche ähnlichen Anwendungen gibt es und wie unterscheidet sich unsere Anwendung von den bisherigen?
- Problem-/Fragestellung und Zielsetzung für ein zu entwickelndes

1.1 Überblick

ShopNook ist ein internetbasierter Marktplatz, der darauf abzielt, Käufern einen großartigen und angenehmen Einkaufsprozess zu bieten. Mit dieser Anwendung können Käufer ganz einfach durch die verschiedenen verfügbaren Produkte gehen, mehrere davon in ihren Einkaufskörben sammeln und sie sicher bezahlen. Das Design von ShopNook ist attraktiv und auf jeder Bildschirmgröße zugänglich, so dass jeder Nutzer, der die Website besucht, unabhängig von seinen Vorlieben bei den Geräten angesprochen wird. Die Website verfügt außerdem über eingebaute Funktionen wie ein Bestandskontrollsystem, Admin-Rollen wie z. B. einen Editor für Produktdetails und andere sowie Käuferprivilegien.

1.2 Globale Anforderungen

1.2.1 Funktionale Anforderungen

Die App wird über alle notwendigen Funktionen verfügen, um ein reibungsloses Einkaufserlebnis zu gewährleisten. Die Kunden werden die Möglichkeit haben, sich sicher zu registrieren, einzuloggen und einen Produktkatalog zu erkunden. Zu jedem Produkt werden vollständige Details wie Name, Beschreibung, Preis und Bilder angezeigt. Die Kunden können schnell Artikel in ihren Einkaufskorb legen, den Inhalt ändern und zur Zahlung übergehen. Sichere Kreditkartenzahlungen werden während des Kaufvorgangs über die Stripe-API abgewickelt. Administratoren können die Waren überwachen und die Bestellungen im Auge behalten, um sicherzustellen, dass das Online-Geschäft gut funktioniert.

1.2.2 Nicht-funktionale Anforderungen

Die App wird effizient mit gleichzeitigen Nutzern umgehen und die Ladezeiten der Seite sind minimal. HTTPS-Verschlüsselung und JWT-basierte Authentifizierung stellen die Sicherheit in den Vordergrund. Die Einheitlichkeit der Geräte wird durch eine reaktionsschnelle und einfach zu bedienende Schnittstelle gewährleistet.

1.2.3 Technische Anforderungen

Die vollwertige E-Commerce-App wird mit npm für JavaScript-Abhängigkeiten und Maven für die Java Abhängigkeiten und Build-Prozesse entwickelt. Im Frontend wird Angular verwendet, um eine dynamische und reaktionsschnelle Schnittstelle für die Benutzer bereitzustellen, während im Backend das Spring Boot-Framework verwendet wird, um RESTful APIs zu erstellen. MySQL wird die Datenbank sein, die Benutzer-, Produkt- und Bestelldaten speichert. Git und GitHub werden für die Versionskontrolle verwendet, um Änderungen zu verwalten und die Teamarbeit zu erleichtern.

Methodologie und Systementwurf

Hier werden über folgende Punkte diskutiert:

- Die Entwicklungskonzeption, die bei der Implementierung der Anwendung verwendet wurde, sowie die Tools, Technologien und Methodologien.
- Überblick über Spring Boot und seine wichtigste Funktionalitäten, die auch in dem Projekt angewendet sein werden.
- Detaillierte Erklärung der Struktur und Design der Anwendung (Bsp: Analyse-/Entwurfsdiagramm)
- Datenbanken, UI-Design und unterschiedliche Komponenten des Systems.
- Wie Spring Boot die Entwicklung und Implementation der verschiedenen Module vereinfacht.

2.1 Entwurf und Konzeption

In der Softwareentwicklung spielen Entwurf und Konzeption eine entscheidende Rolle. Diese Phase des Entwicklungsprozesses befasst sich mit der Strukturierung und Planung von Systemen, bevor die eigentliche Implementierung beginnt.

Ziel ist es, ein klares und verständliches Modell zu erstellen, das die Anforderungen und Funktionen eines Systems abbildet. In diesem Kapitel werden UML und einige Diagrammtypen vorgestellt.

2.1.1 UML

Die Unified Modeling Language (UML) ist ein wesentliches Werkzeug im Bereich Entwurf. UML ist eine standardisierte Modellierungssprache, die eine Vielzahl von Diagrammen bietet, um unterschiedliche Aspekte eines Systems darzustellen. Diese Diagramme helfen dabei, komplexe Systeme zu visualisieren, zu dokumentieren und zu kommunizieren [Bel23].

2.1.2 Klassendiagramm

2.1.3 Use-Case Diagramme

2.2 Front-End Technologien

Dieses Kapitel befasst sich mit Front-End-Technologien wie HTML, CSS, JavaScript und Angular, die für die Erstellung interaktiver, reaktionsfähiger und visuell ansprechender Benutzeroberflächen in unserer E-Commerce-Anwendung unerlässlich sind.

2.2.1 HTML/CSS

HTML¹ und CSS² sind die grundlegenden Technologien zur Erstellung und Gestaltung von Webseiten. HTML liefert die Struktur und den Inhalt einer Webseite, während CSS für die visuelle Darstellung, Layout, Farben und Schriftarten, verwendet wird. Sie ermöglichen es optisch ansprechende und gut strukturierte Webseiten zu erstellen [HTM24].

2.2.2 JavaScript

JavaScript³ ist eine vielseitige High-Level-Programmiersprache, die häufig für die Webentwicklung verwendet wird. Ursprünglich entwickelt, um Webseiten interaktiv zu gestalten, hat sie sich zu einer leistungsstarken Sprache entwickelt, die sowohl für die clientseitige als auch für die serverseitige Entwicklung verwendet werden kann [Jav24].

Die Programmiersprache ist auch für ihre dynamische Natur bekannt und ermöglicht es Entwicklern, interaktive Benutzeroberflächen und robuste serverseitige Anwendungen zu erstellen. Ihre Flexibilität und Ihre umfangreiches Ökosystem, zu dem Bibliotheken und Frameworks wie React, Angular und Node.js gehören, machen sie zu einem unverzichtbaren Werkzeug in der modernen Webentwicklung.

2.2.3 Angular

Angular⁴ ist ein Open-Source-Framework für Webanwendungen, das von Google entwickelt und gepflegt wird. Es wird für die Erstellung dynamischer, einseitiger Anwendungen (SPAs)⁵ mit TypeScript und HTML verwendet. Das Framework bietet eine robuste Plattform für die Entwicklung komplexer Anwendungen, indem es einen umfassenden Satz von Tools und Funktionen wie Datenbindung, Dependency Injection und eine modulare Architektur bietet [GOO24].

Eine der Hauptstärken von Angular ist seine komponentenbasierte Struktur, die

¹ HyperText Markup Language

² Cascading Style Sheets

³ <https://www.javascript.com>

⁴ <https://angular.io>

⁵ Single Page Application

es Entwicklern ermöglicht, wiederverwendbare UI-Komponenten zu erstellen, die die Wartbarkeit und Skalierbarkeit verbessern. Darüber hinaus bietet Angular leistungsstarke Funktionen wie reaktive Programmierung mit RxJS⁶, Zustandsverwaltung und ein reichhaltiges Ökosystem von Bibliotheken, was es zu einer idealen Wahl für Anwendungen auf Unternehmensebene macht.

2.2.3.1 Komponenten

2.3 Back-End Technologien

In diesem Kapitel werden Back-End-Technologien untersucht, wobei der Schwerpunkt auf Spring Boot liegt, um robuste, skalierbare und sichere serverseitige Logik und APIs für unsere E-Commerce-Anwendung zu erstellen.

2.3.1 Java

Java ist eine weit verbreitete Programmiersprache, die für ihre Plattformunabhängigkeit, Stabilität und umfassende Bibliotheken bekannt ist. Aufgrund ihrer Vielseitigkeit und Leistung wird Java häufig für die Entwicklung von Back-End-Anwendungen verwendet. In der E-Commerce-Anwendung (Shop-Nook) wird Java genutzt, um eine robuste und skalierbare serverseitige Logik zu gewährleisten.

2.3.2 REST-API

REST⁷ ist ein Architekturstil für die Entwicklung vernetzter Anwendungen. Er basiert auf einem zustandslosen, Client-Server- und cachefähigen Kommunikationsprotokoll, und in fast allen Fällen wird das HTTP-Protokoll verwendet. REST-APIs sind so konzipiert, dass sie einfach, leichtgewichtig und skalierbar sind, was sie zu einer beliebten Wahl für Webdienste macht.

2.3.2.1 Einführung in REST

REST-APIs bieten eine Möglichkeit, über HTTP auf die Funktionalität und Daten einer Anwendung zuzugreifen. Sie folgen den Prinzipien von REST, die auf Ressourcen und deren Repräsentationen basieren. Jede Ressource wird durch eine eindeutige URI (Uniform Resource Identifier) identifiziert und kann durch standardisierte HTTP-Methoden manipuliert werden:

- GET: Abrufen von Ressourcen
- POST: Erstellen neuer Ressourcen
- PUT: Aktualisieren bestehender Ressourcen
- DELETE: Löschen von Ressourcen

REST-APIs sind leichtgewichtig und nutzen JSON (JavaScript Object Notation) oder XML (Extensible Markup Language) als Datenformat für den Datenaustausch.

⁶ <https://rxjs.dev>

⁷ Representational State Transfer

2.3.2.2 Vorteile von REST-APIs

- **Skalierbarkeit:** Durch das stateless Design können REST-APIs leicht skaliert werden. Jeder HTTP-Request enthält alle notwendigen Informationen, um ihn zu verarbeiten, ohne dass der Server den vorherigen Zustand kennen muss.
- **Flexibilität:** REST-APIs sind flexibel und können mit verschiedenen Datenformaten arbeiten. JSON ist das gebräuchlichste Format, da es leichtgewichtig und gut lesbar ist.
- **Interoperabilität:**⁸ REST-APIs nutzen standardisierte HTTP-Methoden, was ihre Interoperabilität mit verschiedenen Clients und Plattformen gewährleistet.
- **Leichte Integration:** REST-APIs lassen sich leicht in bestehende Systeme integrieren, da sie auf bekannten Webstandards basieren.

2.3.2.3 Implementierung einer REST-API mit Spring Boot

Spring Boot bietet umfassende Unterstützung für die Entwicklung von REST-APIs und vereinfacht deren Implementierung erheblich. Hier sind die wesentlichen Schritte zur Erstellung einer REST-API mit Spring Boot:

2.3.3 Spring Boot

Spring Boot ist ein Framework, das auf dem Spring Framework aufbaut und speziell entwickelt wurde, um schnelle, effiziente und skalierbare Anwendungen zu erstellen. Es bietet eine Vielzahl von Features, die den Entwicklungsprozess beschleunigen und vereinfachen, insbesondere für Java-basierte Webanwendungen und Microservices.

2.3.3.1 Warum wurde Spring Boot ausgewählt ?

Spring Boot wurde aufgrund der breiten Unterstützung in der Entwicklergemeinde ausgewählt. Hier sind einige der Hauptgründe:

- **Produktivität:** Das Framework ermöglicht es auf das Schreiben von Geschäftslogik zu konzentrieren, anstatt sich um die Konfiguration von Technologien zu kümmern.
- **Microservices:** Es eignet sich hervorragend für die Entwicklung von Microservices-Architekturen, indem es Server wie Tomcat bietet, sodass die Anwendungen ohne externen Server laufen können.
- **Integration:** Es lässt sich nahtlos in andere Spring-Projekte und eine Vielzahl von Datenbanken integrieren.

Das Framework bietet auch viele Funktionalitäten, die es von anderen abheben, es hat viele weitere Vorteile, die hier genannt werden:

⁸ Interoperabilität ist die Fähigkeit verschiedener Systeme oder Software, zusammenzuarbeiten und Informationen nahtlos auszutauschen[wik].

- **Auto-Konfiguration:** Die Autokonfigurationsfunktion von Spring Boot konfiguriert Ihre Anwendung automatisch anhand der Abhängigkeiten, die Sie dem Projekt hinzugefügt haben. Dadurch wird die Notwendigkeit einer manuellen Konfiguration minimiert und die Entwicklung beschleunigt.
- **Eigenständige Anwendungen:** Spring Boot-Anwendungen können als eigenständige Java-Anwendungen verpackt werden, was die Bereitstellung einfacher und konsistenter macht.
- **Produktionstaugliche Funktionen:** Das Framework umfasst zahlreiche produktionsreife Funktionen wie Zustandsprüfungen, Metriken und externalisierte Konfiguration.

2.3.3.2 Wie funktioniert Spring Boot ?

1. **Initial Setup:** Mit Spring Initializr kann man schnell ein neues Spring Boot-Projekt mit allen erforderlichen Abhängigkeiten und Konfigurationen initialisieren.
2. **Main Application Class:** Jede Spring Boot-Anwendung hat eine Hauptklasse, die mit `@SpringBootApplication` annotiert ist. Diese Annotation ist eine Kombination aus `@Configuration`, `@EnableAutoConfiguration` und `@ComponentScan`.

```
1  @SpringBootApplication
2  public class SpringBootEcommerceApplication {
3      public static void main(String[] args) {
4          SpringApplication.run(SpringBootEcommerceApplication.class, args);
5      }
6  }
```

Listing 2.1: Main-Application-Class-Implementierung in Java

3. **Application Properties:** Spring Boot ermöglicht eine einfache Konfiguration durch application.properties-Datei. Dadurch wird die Konfiguration externalisiert, was die Verwaltung verschiedener Umgebungen erleichtert.

```
1  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
2  spring.datasource.url=jdbc:mysql://localhost:3306/full-stack-ecommerce?
3  useSSL=false&useUnicode=yes&characterEncoding=UTF-8&allowPublicKeyRetrieval=
4  true&serverTimezone=UTC
5  spring.datasource.username=username
6  spring.datasource.password=password
7  spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
8  spring.data.rest.base-path=/api
```

Listing 2.2: Implementierung von Application.properties Datei

2.3.3.3 Spring Security

Spring Security ist ein robustes und anpassungsfähiges Framework, das für die Authentifizierung und Zugriffskontrolle entwickelt wurde und sich zum Standard für die Absicherung von mit Spring erstellten Anwendungen entwickelt hat.

Spring Security ist ein Framework, das sowohl die Authentifizierung als auch die Autorisierung von Java-Anwendungen übernimmt. Seine Stärke liegt in der Flexibilität, die es erlaubt, es leicht zu erweitern, um spezifische Anforderungen zu erfüllen.

Das Framework bietet eine Reihe von Funktionen, darunter umfassende Unterstützung für Authentifizierung und Autorisierung, Schutz vor verschiedenen Angriffen wie Session Fixation, Clickjacking und Cross-Site Request Forgery, Integration mit der Servlet API und optionale Kompatibilität mit Spring Web MVC [Sprd].

Zum Schutz der Endpunkte für Kundenbestellungen wurde Spring Security mit OAuth2-Authentifizierung implementiert, wobei JWT zur Gewährleistung eines sicheren Zugriffs verwendet wird. Die Einrichtung umfasste die Konfiguration von CORS zur Verwaltung von Herkunftsübergreifenden Anfragen und die Anpassung von Sicherheitseinstellungen wie die Deaktivierung des CSRF-Schutzes und die Handhabung der Inhaltsaushandlung. Dieser Ansatz gewährleistet robuste Sicherheit, während er verschiedene Inhaltstypen unterstützt und die gemeinsame Nutzung von Ressourcen über verschiedene Ursprünge hinweg effektiv handhabt.

2.4 Datenbankstruktur

2.4.1 Übersicht über relationale Datenbanken und Auswahl von MySQL

Die Daten in einer relationalen Datenbank sind in Tabellen mit Spalten und Zeilen organisiert, und auch die Primär- und Fremdschlüssel werden verwendet, um die Beziehungen zwischen den Tabellen herzustellen. Dieser Ansatz ist ideal für E-Commerce-Anwendungen, da er komplexe Abfragen und aussagekräftige Daten verarbeiten kann. Produktlieferungen, Auftragstransaktionen und Kundendaten werden alle ordnungsgemäß verwaltet. Die Möglichkeit, Tabellen zu verknüpfen, erleichtert umfassende Analysen, einschließlich der Nachverfolgung von Kaufdatensätzen und der Bestandskontrolle, während die Einhaltung von ACID-Eigenschaften eine zuverlässige Transaktionsverarbeitung gewährleistet, die für die Auftragsabwicklung und das Zahlungsmanagement entscheidend ist. Relationale Datenbanken wie MySQL bieten die Robustheit und Konsistenz, die für einen sicheren und produktiven Online-Shopping-Betrieb erforderlich sind [IBM].

2.4.2 Datenbankschemaentwurf mit Spring Data JPA

Ziel dieses Abschnitts ist es, zu erörtern, wie Spring Data JPA den Entwurf und die Verwaltung von Datenbankschemata vereinfacht. Es wird gezeigt, wie Spring Data JPA es Entwicklern ermöglicht, Entitäten und Beziehungen innerhalb eines Datenbankschemas abzubilden und zu verwalten, und wie es die Komplexität des Datenzugriffs durch seine Repository-Schicht abstrahiert.

Spring Data JPA vereinfacht den Entwurf von Datenbankschemata, indem es Entwicklern ermöglicht, Java-Klassen mithilfe von Annotationen wie `@Entity`, `@Id` und `@OneToMany` auf Datenbanktabellen abzubilden. Diese Annotationen übernehmen automatisch die Schemaerstellung und -verwaltung und stellen sicher, dass die Datenbankstruktur mit dem Domänenmodell der Anwendung übereinstimmt [Sprc, Bae].

Die Repository-Schicht in Spring Data JPA abstrahiert die Komplexität des Datenzugriffs und bietet integrierte CRUD-Operationen über Schnittstellen wie `JpaRepository`. Dies macht manuelle SQL-Abfragen überflüssig und rationalisiert die Datenbankinteraktionen, so dass sich die Entwickler auf die Geschäftslogik konzentrieren können [Sprb, Spra].

2.4.3 Datenpersistenz und Transaktionen

In diesem Abschnitt wird erläutert, wie Spring Boot und Hibernate die Datenpersistenz handhaben. Der Schwerpunkt liegt dabei auf CRUD-Operationen, der Rolle des `EntityManager` und der Interaktion des Hibernate ORM mit MySQL.

CRUD-Operationen (Erstellen, Lesen, Aktualisieren, Löschen) werden mit Spring Boot und Hibernate verwaltet. Der `EntityManager` ist in diesem Prozess von zentraler Bedeutung und ermöglicht das Erstellen, Abrufen, Aktualisieren und Löschen von Entitäten. Er verwaltet auch den Lebenszyklus von Entitäten innerhalb von Transaktionen und stellt sicher, dass Änderungen bei Bedarf ordnungsgemäß übertragen oder zurückgenommen werden [Bae24].

Hibernate ORM wird verwendet, um Java-Objekte auf Datenbanktabellen abzubilden, was die Interaktion mit der MySQL-Datenbank vereinfacht. Dieses ORM-Framework macht manuelles SQL überflüssig und bietet einen objektorientierten Ansatz für Datenbankoperationen [Hib].

2.4.4 MySQL-Workbench und SQL

In diesem Abschnitt wird die MySQL Workbench als Werkzeug für den Entwurf und die Verwaltung von Datenbanken und SQL als Sprache für die Erstellung von Abfragen und die Verwaltung der Datenbank vorgestellt.

MySQL Workbench wurde zur Visualisierung des Datenbankschemas eingesetzt und ermöglichte eine intuitive Gestaltung und Verwaltung der Datenbankstrukturen. Sie erleichterte die Erstellung und Änderung von Tabellen, Beziehungen und Indizes. Darüber hinaus wurde SQL verwendet, um Abfragen für Tests und die Datenbankverwaltung auszuführen, einschließlich Aufgaben wie Datenmanipulation und Schemaaktualisierungen [myS].

2.4.5 Datenbankverbindung und -konfiguration in Spring Boot

In diesem Abschnitt wird erläutert, wie die Datenbank innerhalb der Spring Boot-Anwendung verbunden und konfiguriert wird.

Die Datenbankverbindung wird in der Datei `application.properties` konfiguriert. Zu den wichtigsten Eigenschaften gehören die JDBC-URL, der Name der Treiberklasse und die Anmeldeinformationen, wie in Listing 2.2 dargestellt.

2.5 Entwicklungsumgebung und Versionskontrolle

2.5.1 Entwicklungsumgebung

2.5.1.1 IntelliJ IDEA

IntelliJ IDEA ist bekannt für seinen leistungsstarken Code-Editor, der sich durch eine umfassende Anfangsindizierung beim Verstehen und Verwalten von Code auszeichnet. Diese Funktion ermöglicht es ihm, Fehler in Echtzeit zu erkennen, kontextbezogene Vorschläge zur Code-Vervollständigung anzubieten und sicheres Code-Refactoring durchzuführen, neben anderen Funktionen. Es ist mit einer umfangreichen Suite integrierter Entwicklerwerkzeuge ausgestattet und bietet robuste Unterstützung für verschiedene Spring-Frameworks sowohl in Java als auch in Kotlin. Dazu gehören Spring MVC, Spring Boot, Spring Integration, Spring Security und Spring Cloud. Zu den bemerkenswerten Vorteilen gehören intelligente Codierungshilfe, sofortige Navigation innerhalb des Spring-Codes, integrierte Entwicklungswerkzeuge und erweiterte Visualisierungsfunktionen [Jet].

2.5.1.2 Visual Studio Code

Visual Studio Code ist ein vielseitiger und effizienter Quellcode-Editor, der mit Windows, macOS und Linux kompatibel ist. Er bietet native Unterstützung für JavaScript, TypeScript und Node.js und verfügt über eine breite Palette von Erweiterungen für weitere Sprachen und Plattformen [Tead].

Da die Front-End-Entwicklung Angular, ein auf TypeScript basierendes Framework, nutzt, wurde Visual Studio Code als Entwicklungsumgebung gewählt. Die starke native Unterstützung für TypeScript erleichtert die effiziente und optimierte Projektentwicklung und macht zusätzliche Erweiterungen überflüssig. [Teac].

2.5.2 Versionskontrolle

2.5.2.1 Git

Git ist ein kostenloses und quelloffenes, verteiltes Versionskontrollsystem, das entwickelt wurde, um Projekte jeder Größe schnell und effizient zu verwalten. Es hat einen minimalen Platzbedarf und liefert eine außergewöhnlich schnelle Leistung. Git übertrifft herkömmliche SCM-Tools wie Subversion, CVS, Perforce und ClearCase mit Funktionen wie kostengünstiger lokaler Verzweigung, benutzerfreundlichen Staging-Bereichen und flexiblen Arbeitsabläufen.

Eines der herausragenden Merkmale von Git ist sein Verzweigungsmodell, das es von fast allen anderen SCM-Tools unterscheidet. Git ermöglicht und fördert die Erstellung mehrerer lokaler Zweige, von denen jeder unabhängig funktionieren kann. Das Erstellen, Zusammenführen und Löschen von Zweigen geht schnell, nahtlos vonstatten und dauert oft nur Sekunden [Teaa].

2.5.2.2 GitHub

GitHub ist eine cloudbasierte Plattform zum Speichern, Freigeben und gemeinsamen Bearbeiten von Code. Die von Git unterstützten Kollaborationsfunktionen von GitHub ermöglichen es uns:

- Projekte zu präsentieren oder zu verbreiten.
- Verfolgen und Verwalten von Codeänderungen im Laufe der Zeit.
- Andere einladen, den Code zu überprüfen und Verbesserungen vorzuschlagen.
- Zusammenarbeit an Projekten ohne das Risiko, dass unbeabsichtigte Änderungen die Arbeit der anderen beeinträchtigen, bis die Änderungen integriert werden können. [Teab].

Implementierung

Hier wird das Implementationsprozess vorgestellt, zusätzlich werden die während der Implementation eingetretenen Probleme in die Diskussion eingebracht.

Die Frage: wie wurden diese Probleme behandelt?

Außerdem werden wir eine tiefe Darstellung und Beschreibung der Hauptfunktionalitäten der Anwendung (z.B : Warenkorb, Check Out Prozess, Produkt Listing...) geben.

3.1 Startseite

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua . At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

3.2 Online-Shopping

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua . At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

3.3 Bezahlprozess

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua . At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Ergebnisse und Analyse

Eine Darstellung und Analyse der Ergebnisse, sowie ein Vergleich von unserer entwickelten Anwendung mit schon im Markt existierenden E-Commerce Anwendungen.

Endlich werden wir die Strengths, Weaknesses, Opportunities und Threats (SWOT-Analyse) der entwickelten Anwendung ins Licht stellen und darüber diskutieren.

Zusammenfassung und Ausblick

Hier werden wir die Schwierigkeiten und Herausforderungen benennen, die wir bei der Konzeptentwicklung und auch bei der Umsetzung sehen. Welche Punkte sind offen geblieben? Was gilt es noch herauszufinden?

Literaturverzeichnis

- Bae. BAELDUNG: *JPA Entities*. <https://www.baeldung.com/jpa-entities/>. Abgerufen am 14.08.2024.
- Bae24. BAELDUNG: *EntityManager*. <https://www.baeldung.com/hibernate-entitymanager/>, 2024. Abgerufen am 14.08.2024.
- Bel23. BELL, DONALD: *An introduction to the Unified Modeling Language*. <https://developer.ibm.com/articles/an-introduction-to-uml/>, 2023. Abgerufen am 26.07.2024.
- GOO24. Google: *Angular - Landing Page*. <https://angular.io>, 2024. Abgerufen am 02.06.2024.
- Hib. HIBERNATE: *Hibernate*. <https://hibernate.org/orm/what-is-an-orm/>. Abgerufen am 14.08.2024.
- HTM24. *HTML For Beginners*. <https://html.com>, 2024. Abgerufen am 02.06.2024.
- IBM. IBM: *What is a relational database?* <https://www.ibm.com/topics/relational-databases/>. Abgerufen am 14.08.2024.
- Jav24. *JavaScript*. <https://en.wikipedia.org/wiki/JavaScript>, 2024. Abgerufen am 02.06.2024.
- Jet. JETBRAINS: *The IDE for Spring developers*. <https://www.jetbrains.com/idea/spring/>. Abgerufen am 14.08.2024.
- myS. MYSQL: *MySQL Workbench Manual*. <https://dev.mysql.com/doc/workbench/en/wb-intro.html>. Abgerufen am 14.08.2024.
- Spra. SPRING: *JPA Query Methods*. <https://docs.spring.io/spring-data/jpa/reference/jpa/query-methods.html/>. Abgerufen am 14.08.2024.
- Sprb. SPRING: *JPA Repositories*. <https://docs.spring.io/spring-data/jpa/docs/1.6.0.RELEASE/reference/html/jpa.repositories.html/>. Abgerufen am 14.08.2024.
- Sprc. SPRING: *Spring Data JPA*. <https://www.ibm.com/topics/relational-databases/>. Abgerufen am 14.08.2024.
- Sprd. SPRING: *Spring Security*. <https://spring.io/projects/spring-security>. Abgerufen am 14. August 2024.
- Teaa. TEAM, GIT: *Git About*. <https://git-scm.com/about>. Abgerufen am 14. August 2024.

-
- Teab. TEAM, GITHUB: *GitHub About*. <https://docs.github.com/en/get-started/start-your-journey/about-github-and-git#about-github>. Abgerufen am 14. August 2024.
- Teac. TEAM, VISUAL STUDIO: *Using Angular in Visual Studio Code*. <https://code.visualstudio.com/docs/nodejs/angular-tutorial>. Abgerufen am 14. August 2024.
- Tead. TEAM, VISUAL STUDIO: *Visual Studio Code Overview*. <https://code.visualstudio.com/docs>. Abgerufen am 14. August 2024.
- wik. *Wikipedia: Interoperability*. <https://en.wikipedia.org/wiki/Interoperability>. Abgerufen am 27.07.2024.

A

Glossar

DisASter	Distributed Algorithms Simulation Terrain, eine Plattform zur Implementierung verteilter Algorithmen
DSM	Distributed Shared Memory
AC	Atomic Consistency (dt.: Linearisierbarkeit)
RC	Release Consistency (dt.: Freigabekonsistenz)
SC	Sequential Consistency (dt.: Sequentielle Konsistenz)
WC	Weak Consistency (dt.: Schwache Konsistenz)

B

Selbstständigkeitserklärung

- ☐ Diese Arbeit wurde als Gruppenarbeit angefertigt. Meinen Anteil habe ich selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Namen der Mitverfasser:

Meine eigene Leistung ist:

Datum

Unterschrift der Kandidatin/des Kandidaten

Datum

Unterschrift der Kandidatin/des Kandidaten