ONLINESHOP MIT DATENBANK

Nico Samadelli

26. JUNI 2024

SWISS RE/ ZLI

Inhalt

0 Versionierung

Datum	Version	Autor	Änderung
26.06.24	1.0	Nico Samadelli	Dokument abgeschlossen

1 Motivation

Bei meinem Abschlussprojekt wollte ich möglichst viel lernen und meine technischen Fähigkeiten erweitern. Da ich ein grosses Interesse an Datenbanken habe, war es mir wichtig, diese in mein Projekt zu integrieren und praktische Erfahrungen in der Datenbankverwaltung zu sammeln.

Ein weiteres Ziel war es, meine Kenntnisse in JavaScript zu vertiefen. JavaScript ist eine essenzielle Sprache in der Webentwicklung, und ich wollte sicherstellen, dass ich ihre vielfältigen Einsatzmöglichkeiten besser verstehe und anwenden kann. Durch die Arbeit an diesem Projekt konnte ich neue Techniken und Best Practices in JavaScript kennenlernen.

Ausserdem wollte ich meine Fähigkeiten in Node.js verbessern. Obwohl ich bereits in der Vergangenheit mit Node.js gearbeitet habe, liegt diese Erfahrung schon eine Weile zurück. Daher wollte ich mein Wissen auffrischen und erweitern. Node.js bietet viele leistungsstarke Funktionen für die Backend-Entwicklung, und ich wollte sicherstellen, dass ich diese effizient nutzen kann.

Indem ich all diese Technologien in mein Projekt einbezogen habe, wollte ich ein umfassendes Verständnis für die Entwicklung moderner Webanwendungen erlangen. Mein Ziel war es, ein funktionierendes System zu erstellen, das sowohl das Frontend als auch das Backend abdeckt und dabei die Interaktion mit einer Datenbank beinhaltet. Diese Herangehensweise sollte mir nicht nur theoretisches Wissen, sondern auch praktische Fähigkeiten vermitteln, die ich in zukünftigen Projekten anwenden kann.

2 Information

2.1 Vorgaben

Für unser Abschlussprojekt erhielten wir einen Zeitraum von insgesamt 12 Tagen. In dieser Zeitspanne hatten wir die Freiheit, ein eigenes Thema für das Projekt auszuwählen. Das Thema konnte entweder etwas sein, das uns persönlich interessiert, oder es sollte einen Bezug zu unserem Betrieb haben. Diese Entscheidungsfreiheit ermöglichte es uns, ein Thema zu wählen, das sowohl unsere Neugier weckt als auch unsere Fähigkeiten optimal fordert.

Nachdem wir ein Thema ausgewählt hatten, mussten wir es unseren Coaches präsentieren. Dies war notwendig, um sicherzustellen, dass das Projekt sowohl herausfordernd genug als auch realisierbar ist. Unsere Coaches überprüften die vorgeschlagenen Projekte und gaben uns Feedback, um sicherzustellen, dass die Anforderungen und Erwartungen erfüllt werden konnten.

Erst nach der Genehmigung durch die Coaches durften wir mit der eigentlichen Arbeit an unserem Projekt beginnen. Diese sorgfältige Planung und Abstimmung im Vorfeld half uns, den Projektverlauf effizient zu gestalten und sicherzustellen, dass wir alle

notwendigen Ressourcen und Unterstützung hatten, um unser Projekt erfolgreich abzuschliessen.

2.2 Projektidee

Meine Idee war es, einen Onlineshop zu gestalten, der auf Datenbankeinträgen basiert. Ziel war es, dass sowohl die Datenbankabfragen als auch die Einträge über einen selbst aufgesetzten Node.js-Server laufen. Dies erforderte nicht nur das Design und die Implementierung des Onlineshops, sondern auch die Einrichtung und Konfiguration des Servers.

Für die Umsetzung meines Projekts wollte ich JavaScript, HTML und CSS verwenden. Mit JavaScript sollte die Funktionalität des Shops gesteuert werden. HTML sollte für die Strukturierung der Webinhalte dienen, während CSS das Design und die visuelle Darstellung übernehmen sollte. Durch den Einsatz dieser Technologien wollte ich meine Fähigkeiten in JavaScript verbessern und ein tieferes Verständnis für die Entwicklung moderner Webanwendungen erlangen.

Ausserdem habe ich JSON-Dateien genutzt, um die Verbindung zur Datenbank zu vereinfachen. Diese Dateien ermöglichten es mir, sowohl Datenbankeinträge als auch -abfragen effizient zu verwalten. Die Verwendung von JSON für die Kommunikation mit der Datenbank erleichterte das Speichern und Abrufen von Informationen und trug zur Übersichtlichkeit und Wartbarkeit des Codes bei.

2.3 Mögliche Risiken

Das größte Risiko bei diesem Projekt ist, dass es mir nicht gelingt, die Daten richtig in die Datenbank einzutragen. Wenn dieser Schritt nicht klappt, würde ein wichtiger Teil der Anwendung nicht funktionieren.

Abgesehen davon sehe ich keine weiteren großen Risiken. Die anderen Teile des Projekts sind gut planbar und mit den vorhandenen Mitteln machbar. Dennoch sollte ich auch mögliche kleinere Risiken im Auge behalten, wie zum Beispiel unerwartete Fehler in der Frontend-Implementierung oder Schwierigkeiten bei der Kombination verschiedener Technologien.

3 Planung/Entscheidung

Bei der Planung meines Projekts habe ich zuerst ein Projekt und ein Repository auf GitHub erstellt. Diese Plattform bot mir die nötigen Werkzeuge, um meine Fortschritte zu dokumentieren und meine Aufgaben effizient zu verwalten.

Auf GitHub erstellte ich verschiedene Meilensteine, die als wichtige Etappen auf dem Weg zur Fertigstellung meines Projekts dienten. Jedem Meilenstein wies ich spezifische Aufgaben zu, was mir half, die Arbeit in überschaubare und erreichbare Schritte zu unterteilen.

Anschliessend habe ich auf der Roadmap die Zeiteinteilung für die einzelnen Aufgaben festgelegt. Diese visuelle Darstellung ermöglichte es mir, meine Zeit optimal zu planen und sicherzustellen, dass alle Aufgaben rechtzeitig erledigt werden.

Ein wichtiger Teil der Planung war die Strukturierung der Datenbank. Zuerst habe ich meine Ideen und Entwürfe auf Papier skizziert, um ein klares Bild der Datenbankstruktur zu bekommen. Diese handschriftlichen Skizzen halfen mir, die Beziehungen zwischen

den verschiedenen Datenbankelementen zu visualisieren und mögliche Probleme frühzeitig zu erkennen.

Nachdem ich eine solide Grundlage auf Papier hatte, nutzte ich Online-Tools, um ein detailliertes Datenbankmodell zu erstellen. Dieses Modell diente als Blaupause für die tatsächliche Implementierung der Datenbank und half mir, die Struktur präzise und fehlerfrei umzusetzen.

Durch diese sorgfältige und strukturierte Planung konnte ich die Basis für ein erfolgreiches Projekt legen und sicherstellen, dass alle Aspekte gut durchdacht und organisiert sind.

4 Meilensteine & Aufträge

4.1 Meilenstein 1: Datenbank erstellen

- Datenbankmodell erstellen
- Datenbank erstellen
- Daten in DB einfügen

4.2 Meilenstein 2: Website mit React erstellen

- Home-Website erstellen
- Home-Website Aussehen gestalten
- Daten auf Website anzeigen
- Warenkorb erstellen
- Shop Website Aussehen gestalten
- Bestell-Button erstellen

4.3 Meilenstein 3: PowerPoint erstellen

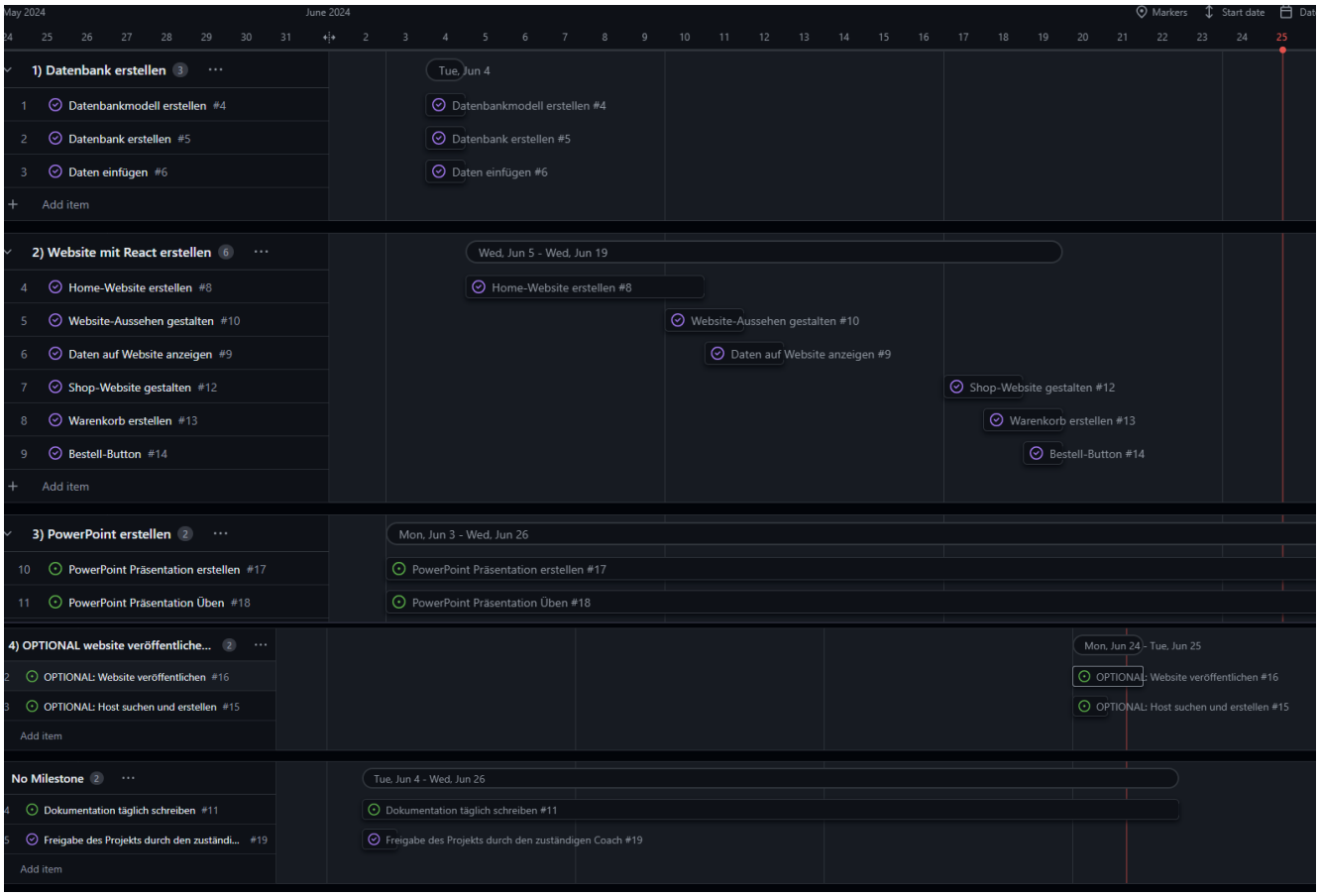
- PowerPoint Präsentation erstellen
- PowerPoint Präsentation üben

4.4 Meilenstein 4: Website mit Host veröffentlichen (Optional)

- Host suchen und erstellen (Optional)
- Website veröffentlichen (Optional)

5 Zeit Management

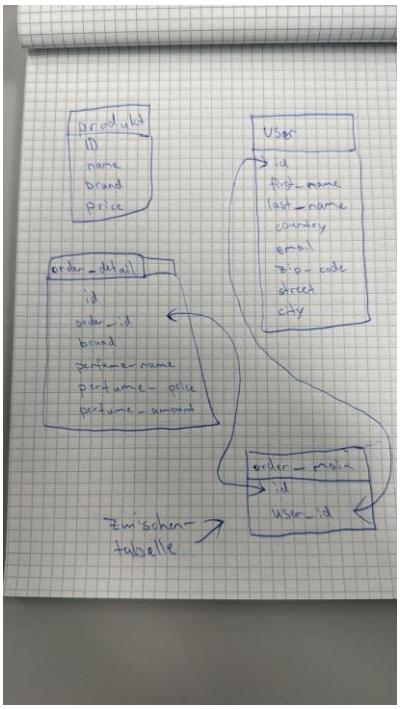
Ich habe mir einen Plan auf GitHub erstellt. Dort steht, was ich bis wann erledigt haben möchte.
So ist es einfach für mich, den Überblick über meine Aufträge zu halten. Dadurch weiss ich, wann ich gut in der Zeit liege und wann nicht.



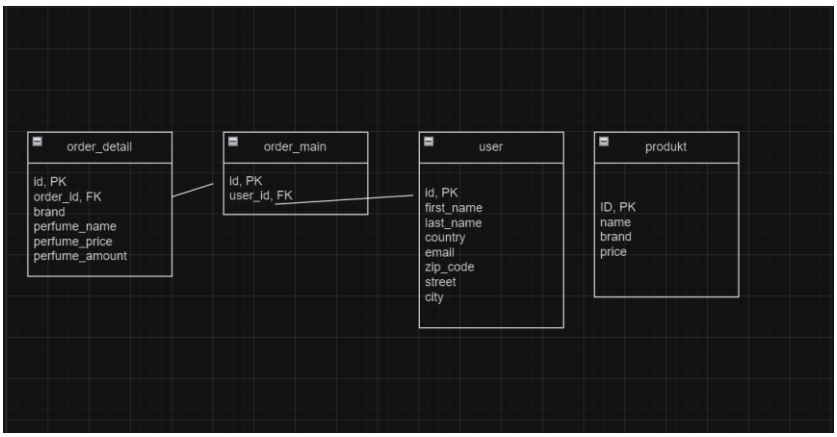
6 Umsetzung

6.1 Schritt 1: Datenbank erstellen

Ich habe zuerst auf einem Stück Papier ein Datenbankmodell erstellt. So ist es für mich einfacher, das Datenbank Modell zu erstellen. Das 2. Modell habe ich auf draw.io erstellt.

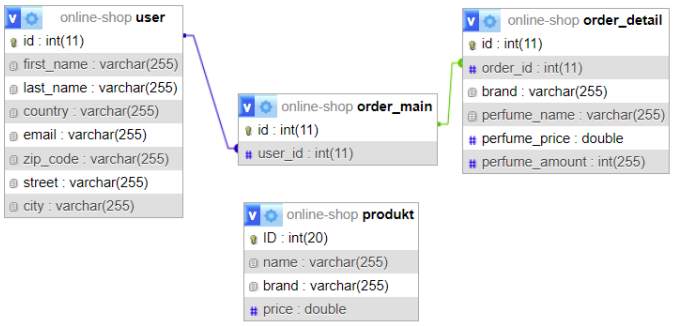


Das ist mein Datenbankmodell, welches ich auf Papier erstellt habe.
Es ist noch nicht so genau, weil es mir nur einen Überblick bringen sollte.



Dieses Datenbankmodell habe ich auf draw.io erstellt. Es sieht sehr ähnlich aus, wie das DBM auf dem Blatt Papier. Ich habe aber zusätzlich noch die PK und FK beschriftet.

Nach dem Erstellen der beiden Datenbankmodelle (DBM) habe ich die Datenbank in phpMyAdmin implementiert. Zunächst startete ich den Apache-Server in XAMPP, um sicherzustellen, dass phpMyAdmin ausgeführt wird. Anschliessend habe ich den MySQL-Dienst in XAMPP gestartet, um die Verbindung zur Datenbank herzustellen. Nachdem beide Dienste aktiv waren, habe ich die Datenbank in phpMyAdmin eingerichtet und konfiguriert. Dabei habe ich die DBM zur Hilfe geholt. Ich habe darauf geachtet, die richtigen Datenarten auszuwählen. Für Namen z.B. Varchar.



Das ist meine Datenbank in phpMyAdmin.

Als letztes habe ich dann Testdaten in die «produkt» Tabelle eingefügt.

6.2 Schritt 2: Verbindung zur DB herstellen

Nachdem ich die Datenbank erfolgreich erstellt hatte, begann ich mit der Entwicklung der Website. Zuerst erstellte ich eine leere HTML-Seite, die als Grundlage für die Anzeige der Daten dienen sollte. Mein erstes Ziel war es, die Daten aus der Datenbank abzurufen und auf der Website anzuzeigen.

Um dies zu erreichen, richtete ich einen Node.js-Server ein. Da ich noch keine Erfahrung damit hatte, recherchierte ich zunächst online nach Anleitungen und Tutorials. Nachdem ich mir das nötige Wissen angeeignet hatte, begann ich mit der Einrichtung des Servers. Ich erstellte eine neue Projektstruktur und initialisierte das Projekt mit einem package.json, um die notwendigen Abhängigkeiten zu definieren. Ich habe meine Logindaten angegeben, damit die Verbindung zur Datenbank hergestellt werden kann.

```
Node.js v20.10.0
PS C:\1.Lehrjahr\ZLI\Abschlussprojekt> cd C:\1.Lehrjahr\ZLI\Abschlussprojekt
PS C:\1.Lehrjahr\ZLI\Abschlussprojekt> node RESTDBAccess.js
Starting server...
Server is running on port 3000
Connected to the database.
```

So starte ich den Node.js-Server. Zuerst wechsle ich in das richtige Verzeichnis meines Projekts. Im Screenshot war das Verzeichnis bereits korrekt, aber ich habe den Befehl zur Demonstration erneut eingegeben.

Dann starte ich den Server über das .json-File. Der Server läuft auf Port 3000. Sobald der Server aktiv ist, erscheint eine Nachricht, die bestätigt, dass der Server gestartet wurde und bereit ist, Anfragen zu bearbeiten.

Der Server stellt dann eine Verbindung zur MySQL-Datenbank her. Wenn die Verbindung erfolgreich ist, zeigt der Server eine Nachricht an, die dies bestätigt.

Diese Bestätigungen sind wichtig, um sicherzustellen, dass der Server und die Datenbank korrekt funktionieren. Der erfolgreiche Start und die Verbindung zur Datenbank sind grundlegende Schritte, die mir zeigen, dass das System bereit ist, Daten aus der Datenbank abzurufen und auf der Website anzuzeigen.

Hier rufe ich die Daten ab:

```
// Definiert einen Endpunkt zum Abrufen aller Produkte
app.get('/products', (req, res) => {
    const sql = "SELECT * FROM produkt";
    // Führt die SQL-Abfrage aus und sendet das Ergebnis als JSON-Antwort zurück
    con.query(sql, (err, result) => {
        if (err) {
            console.error('Error executing query:', err);
            return res.status(500).json({ error: 'Internal Server Error' });
        }
        res.json(result);
    });
});
```

6.3 Schritt 3: Daten auf Website ausgeben

Mit Hilfe von Express, einem Webframework für Node.js, und dem MySQL-Modul konfigurierte ich den Server, um HTTP-Anfragen zu bearbeiten und eine Verbindung zur MySQL-Datenbank herzustellen. Ich implementierte eine Route, die Daten aus der Datenbank abruft und sie als JSON zurückgibt.

Parallel dazu schrieb ich den Frontend-Code, der eine Anfrage an den Node.js-Server sendet, die Daten empfängt und sie dynamisch in die HTML-Seite einfügt. Dadurch konnte ich sicherstellen, dass die Daten aus der Datenbank korrekt auf der Webseite angezeigt werden.

So rufe ich die daten vom node.js Server ab:

```
// Abrufen der Produktdaten vom Server
fetch("http://127.0.0.1:3000/products")
    .then(response => {
```

6.4 Schritt 4: weitere Funktionen einbauen

6.4.1 Warenkorb

Ich habe bei den Produkten einen Button hinzugefügt, der das Produkt in den Warenkorb legt. Um das zu programmieren, habe ich eine Funktion erstellt, die das ausgewählte Produkt und seine Details in den Warenkorb hinzufügt. Diese Funktion überprüft, ob das Produkt bereits im Warenkorb ist. Falls ja, erhöht sie die Anzahl, andernfalls fügt sie das Produkt neu hinzu.

Danach habe ich Funktionen hinzugefügt, mit denen die Anzahl eines Produkts im Warenkorb erhöht und verringert werden kann. Diese Funktionen werden durch die "+"

und "-" Buttons ausgelöst. Wenn die Anzahl eines Produkts auf eins ist und der "-" Button erneut gedrückt wird, wird das Produkt aus dem Warenkorb entfernt.

Anschliessend wollte ich den Gesamtpreis anzeigen. Dafür habe ich bei jedem Produkt im Warenkorb den Gesamtpreis dieses Produkts angezeigt, indem ich den Einzelpreis mit der Anzahl multipliziert habe. Diese berechneten Preise wurden unter den Produktdetails angezeigt.

```
{/* Zeigt den Gesamtpreis für das einzelne Produkt an */}
<p>Total: CHF {item.price * item.quantity}</p>
```

Um den gesamten Betrag aller Produkte im Warenkorb anzuzeigen, habe ich die Preise aller Produkte addiert und diesen Gesamtpreis am Ende der Warenkorbansicht ausgegeben.

```
// Berechnet den Gesamtpreis aller Artikel im Warenkorb
const total = cartItems.reduce((acc, item) => acc + item.price * item.quantity, 0);
```

6.4.2 Shipping Formular

Um eine Bestellung abzuschicken, müssen Bestelldetails angegeben werden, damit klar ist, wohin das Paket gesendet werden soll. Daher habe ich im Warenkorb einen Button hinzugefügt, der zur Bestellseite führt.

Mit diesem Button wird man zu der «Order» Seite weitergeleitet.

```
    />
    <Link to="/order" className="order-btn">Place Order</Link>
</div>
```

Wenn dieser Button geklickt wird, gelangt man auf eine neue Seite. Auf dieser Seite wird oben der aktuelle Inhalt des Warenkorbs angezeigt, und darunter befindet sich ein Formular, das der Benutzer ausfüllen muss. Dieses Formular sammelt wichtige Versandinformationen wie Vorname, Nachname, Land, Postleitzahl, Strasse, Stadt und E-Mail-Adresse.

Ich habe dieses Formular mit HTML erstellt.

```
<div className="form-group">
    <label htmlFor="last_name">Last Name</label>
    <input type="text" id="last_name" name="last_name" value={last_name} onChange={this.handleChange} required />
</div>
```

6.4.3 Bestellbutton

Wenn man auf den Bestell-Button unter dem Formular klickt, werden die Daten in die Datenbank eingetragen. Die Informationen aus dem Benutzerformular, wie Name, Adresse und E-Mail, werden in die «user»-Tabelle eingetragen. Die Produkte im Warenkorb werden in die «order_detail»-Tabelle eingefügt.

Der Node.js-Server verarbeitet diese Daten. Zuerst erstellt er einen neuen Eintrag in der «user»-Tabelle mit den Benutzerdaten. Danach erhält er die ID des neuen Benutzers, um eine Verknüpfung zu den Bestelldetails herzustellen. Anschliessend werden die Produkte aus dem Warenkorb in die «order_detail»-Tabelle eingetragen, zusammen mit der Benutzer-ID und den Produktdetails wie Name, Marke, Preis und Menge.

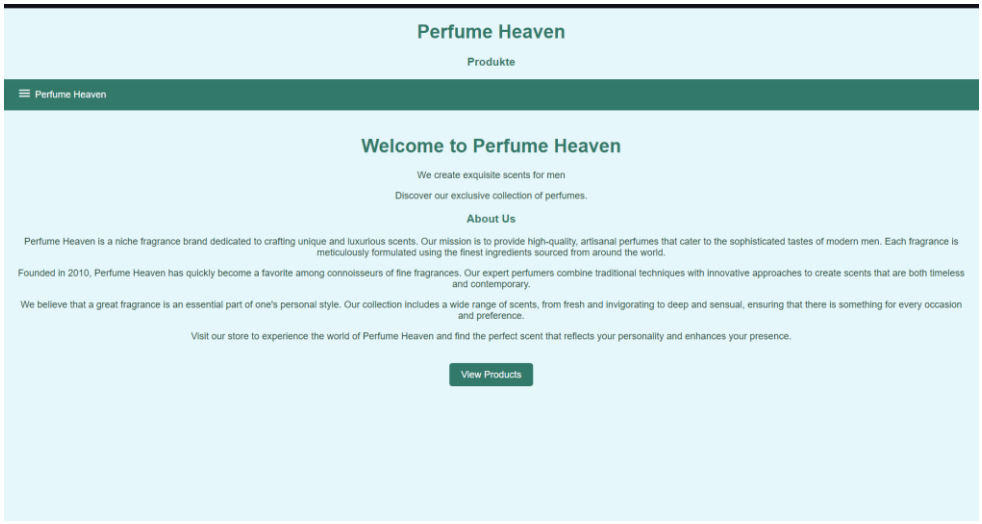
```
// Definiert einen Endpunkt zum Erstellen einer neuen Bestellung
app.post('/orders', (req, res) => {
    const { user, cart } = req.body; // Extrahiert Benutzerdaten und Warenkorbdaten aus der Anfrage
    const userSql = 'INSERT INTO user (first_name, last_name, country, zip_code, street, city, email) VALUES (?, ?, ?, ?, ?, ?, ?)'; //
    const userValues = [user.first_name, user.last_name, user.country, user.zip_code, user.street, user.city, user.email]; // Werte für

    // Führt die Einfügeabfrage für den Benutzer aus
```

Nachdem alle Daten erfolgreich in die Datenbank eingetragen wurden, zeigt die Seite eine Bestätigungsnachricht an. Dies erfolgt durch einen Alert, der bestätigt, dass die Bestellung erfolgreich aufgegeben wurde. So wissen die Benutzer, dass ihre Bestellung erfolgreich bearbeitet wurde und die Daten korrekt gespeichert sind.

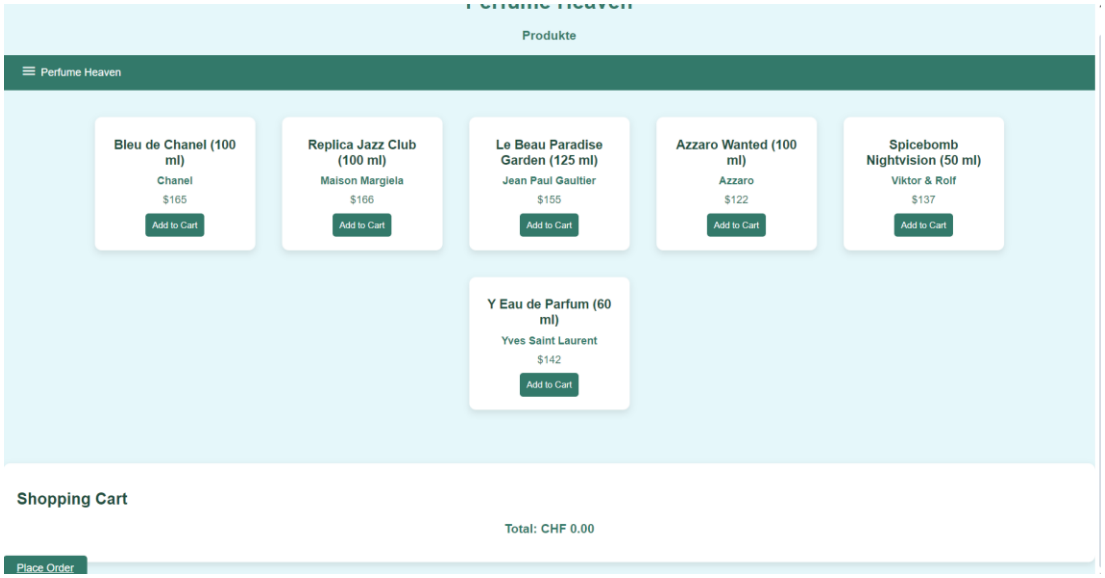
7 Demonstration meiner Website

Das ist ein Screenshot meiner Home Webseite. Der Nutzer hat 2 Möglichkeiten zu den Produkten zu gelangen. Entweder klickt man auf den «View products» Button, oder man navigiert mit der Navbar zu den Produkten. Beide Optionen führen zu der gleichen Seite.



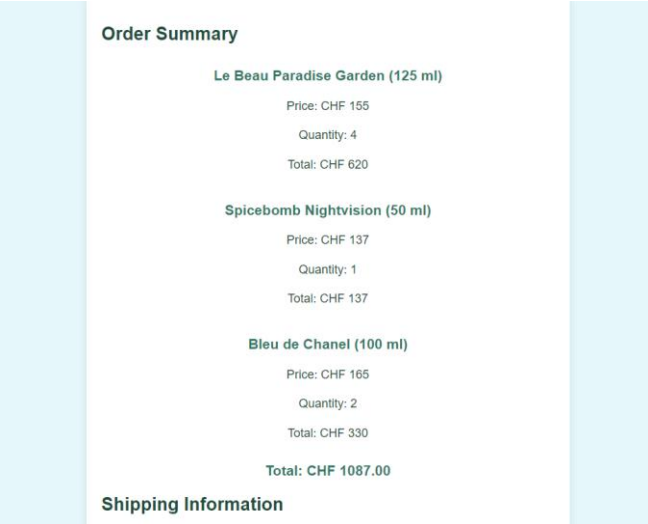
Navbar:

Hier sieht man die Website mit den Artikeln. Man kann einzelne Produkte mit dem «Add to cart» Button dem Warenkorb hinzufügen. Unter den Artikeln sieht man bereits den Warenkorb und das Total. Ebenfalls kann man den Bestell-Button schon erkennen.

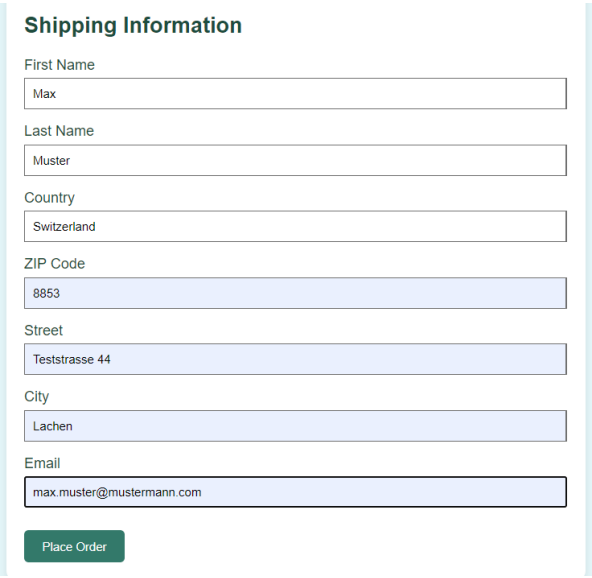


Ich habe dem Warenkorb nun ein paar Artikel hinzugefügt. Man kann mit den + und – Button ganz einfach die Artikelanzahl erhöhen und verringern.

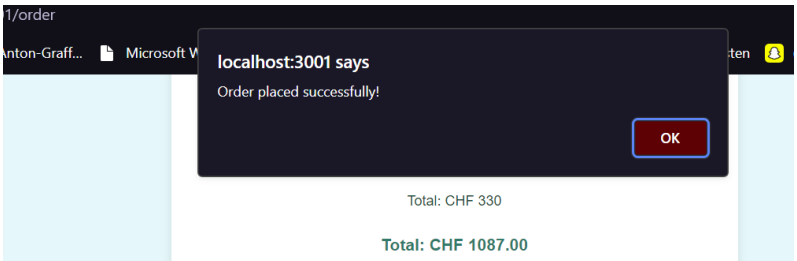
Nun kann man auf den «Place Order» Button klicken, um zu dem Userformular zu kommen. Zuallererst wird auf der Website jetzt die Bestellung erneut angezeigt.



Danach muss man die Shipping Informationen ausfüllen.



Wenn man auf «Place Order» klickt, kommt dieser Alert



Die untersten 3 einträge sind von der neuen Bestellung.

	id	order_id	brand	perfume_name	perfume_price	perfume_amount
Bearbeiten Kopieren Löschen	1	1	Chanel	Bleu de Chanel (100 ml)	165	1
Bearbeiten Kopieren Löschen	2	1	Maison Margiela	Replica Jazz Club (100 ml)	166	1
Bearbeiten Kopieren Löschen	3	2	Chanel	Bleu de Chanel (100 ml)	165	1
Bearbeiten Kopieren Löschen	4	2	Maison Margiela	Replica Jazz Club (100 ml)	166	1
Bearbeiten Kopieren Löschen	9	5	Yves Saint Laurent	Y Eau de Parfum (60 ml)	142	1
Bearbeiten Kopieren Löschen	10	6	Yves Saint Laurent	Y Eau de Parfum (60 ml)	142	1
Bearbeiten Kopieren Löschen	11	7	Yves Saint Laurent	Y Eau de Parfum (60 ml)	142	2
Bearbeiten Kopieren Löschen	12	7	Jean Paul Gaultier	Le Beau Paradise Garden (125 ml)	155	1
Bearbeiten Kopieren Löschen	13	7	Azzaro	Azzaro Wanted (100 ml)	122	3
Bearbeiten Kopieren Löschen	14	7	Viktor & Rolf	Spicebomb Nightvision (50 ml)	137	4
Bearbeiten Kopieren Löschen	15	8	Chanel	Bleu de Chanel (100 ml)	165	1
Bearbeiten Kopieren Löschen	16	8	Yves Saint Laurent	Y Eau de Parfum (60 ml)	142	1
Bearbeiten Kopieren Löschen	17	9	Jean Paul Gaultier	Le Beau Paradise Garden (125 ml)	155	1
Bearbeiten Kopieren Löschen	18	10	Jean Paul Gaultier	Le Beau Paradise Garden (125 ml)	155	4
Bearbeiten Kopieren Löschen	19	10	Viktor & Rolf	Spicebomb Nightvision (50 ml)	137	1
Bearbeiten Kopieren Löschen	20	10	Chanel	Bleu de Chanel (100 ml)	165	2

Wie man sieht, steigt die ID normal an. Die order_id bleibt jedoch gleich. Das habe ich gemacht, indem ich eine Zwischentabelle erstellt habe. In dieser Zwischentabelle wird die user_id genommen und ihr wird eine order_id hinzugefügt. Auf diesem Screenshot ist der unterste Eintrag wichtig. Ich habe das aus einem Grund so gemacht. So kann man sehen, welche Bestellung von welchem User getätigt wurde.

	id	user_id
Bearbeiten Kopieren Löschen	1	1
Bearbeiten Kopieren Löschen	2	2
Bearbeiten Kopieren Löschen	5	5
Bearbeiten Kopieren Löschen	6	6
Bearbeiten Kopieren Löschen	7	7
Bearbeiten Kopieren Löschen	8	8
Bearbeiten Kopieren Löschen	9	9
Bearbeiten Kopieren Löschen	10	10

Hier stehen die User-daten. Der unterste Eintrag ist wichtig.

	id	first_name	last_name	country	email	zip_code	street	city
Bearbeiten Kopieren Löschen	1	test	test	test	test.test@test.com	5620	test	test
Bearbeiten Kopieren Löschen	2	n	n	n	n@lskd.com	4523	n	n
Bearbeiten Kopieren Löschen	5	w	w	w	w@w.com	0834	w	w
Bearbeiten Kopieren Löschen	6	f	f	f	f@f.com	1234	f	f
Bearbeiten Kopieren Löschen	7	n	n	Switzerland	kodak@kodak.com	5620	Burgerstrasse 46 2	Zufikon
Bearbeiten Kopieren Löschen	8	p	p	p	p@p.com	23454	p	p
Bearbeiten Kopieren Löschen	9	x	x	x	x.x@x.com	2344	x	x
Bearbeiten Kopieren Löschen	10	Max	Muster	Switzerland	max.muster@mustermann.com	8853	Teststrasse 44	Lachen

8 Arbeitsjournal

8.1 Montag, 03. Juni

Ich habe mit der Projektplanung begonnen. Bereits vor Projektbeginn hatte ich erste Ideen, welches Projekt ich umsetzen möchte. Dennoch fiel es mir schwer, mich für ein passendes Projekt zu entscheiden. Nachdem ich schliesslich eine Entscheidung getroffen hatte, startete ich mit der detaillierten Planung. Diese verlief sehr gut, und ich konnte verschiedene Meilensteine und Aufgaben festlegen.

8.2 Dienstag, 04. Juni

h habe mit der Umsetzung meines Projekts begonnen. Zuerst habe ich auf Papier skizziert, wie die Datenbank aufgebaut sein soll. Anschliessend habe ich ein Datenbankmodell erstellt und die Datenbank in phpMyAdmin angelegt. Nachdem ich Daten eingetragen hatte, habe ich eine leere Website erstellt. Danach begann ich damit, die Daten auf meiner Website anzuzeigen.

8.3 Mittwoch, 05. Juni

Ich habe weiter daran gearbeitet, die Daten auf der Website anzuzeigen. Dies stellte sich als etwas kompliziert heraus, da ich zuerst mit Node.js eine Verbindung zur Datenbank herstellen musste. Anschliessend musste ich mich mit dem Node.js-Server verbinden, um die abgefragten Daten auf der Website anzuzeigen. Das habe ich dann auch geschafft, dieser Prozess brauchte jedoch viel Zeit.

8.4 Montag, 10. Juni

Ich habe meine Dokumentation weitergeführt und das Erscheinungsbild meiner Website verbessert. Anstatt alle Artikel in einer Tabelle darzustellen, was ich nicht ansprechend fand, werden sie nun in kleinen Kästchen angezeigt.

8.5 Dienstag, 11. Juni

Ich habe weiter an meinem Projekt gearbeitet. Beim Aufrufen meiner Website wurden die Artikel jedoch nicht mehr angezeigt, sondern es trat ein Fehler auf. Ich musste das Problem beheben und fand heraus, dass der für die Verbindung zur Datenbank verwendete Port von einem Docker-Container belegt war. Nachdem ich den Docker-Server beendet hatte, war das Problem gelöst. Ausserdem habe ich erneut an meiner Dokumentation weitergeschrieben.

8.6 Mittwoch, 12, Juni

Ich habe meiner Website die Funktion hinzugefügt, Artikel in einen Warenkorb zu legen. Nachdem dies erfolgreich war, stellte ich fest, dass man die Artikel nicht mehr aus dem Warenkorb entfernen konnte. Daher ergänzte ich die Seite um "+" und "-" Buttons, um die Menge der Artikel zu ändern.

Zusätzlich habe ich die Anzeige des Gesamtpreises integriert. Anfangs wurde der Gesamtpreis jedoch falsch angezeigt. Anstatt die berechneten Preise der einzelnen Artikel zu addieren, wurden sowohl die Einzelpreise als auch die Gesamtpreise zusammengezählt, was zu einem viel höheren Betrag führte. Dieses Problem konnte ich leicht beheben, indem ich die Logik zur Berechnung des Gesamtpreises korrigierte.

8.7 Dienstag 18. Juni

Ich habe meine Homepage erweitert und mehr Text hinzugefügt. Ausserdem habe ich das Aussehen der Produktseite verbessert. Da ich nur das Design verändert habe, traten keine Fehler auf. Zudem habe ich die Funktion implementiert, dass die Produkte im Warenkorb gespeichert werden. Dies war bisher nicht eingebaut. Anschliessend habe ich einen Bestellbutton hinzugefügt, der den Benutzer zu einer anderen Seite weiterleitet. Auf dieser Seite werden die Produkte aus dem Warenkorb angezeigt, zusammen mit einem Formular für die Benutzerdaten.

8.8 Mittwoch 19. Juni

Ich habe begonnen, die Funktion zu implementieren, dass die Daten aus dem Benutzerformular und die Bestellungen in die Datenbank eingetragen werden. Zusätzlich habe ich die Datenbankstruktur etwas angepasst.

8.9 Montag 24. Juni

Zuerst habe ich die Datenbank etwas abgeändert, da ich die Bestelldaten auch etwas verändert habe. Danach habe ich es endlich geschafft, durch einen Knopfdruck die Daten aus dem Bestellformular und die Produkte aus dem Warenkorb in die Datenbank einzutragen. Danach habe ich einen Alert gemacht, welcher bestätigt, dass die Bestellung abgeschickt wurde.

8.10 Dienstag 25. Juni

Ich habe den ganzen Tag an meiner Dokumentation gearbeitet. Das lief sehr gut und ich stiess auf keine Probleme.

8.11 Mittwoch 26. Juni

Ich habe den ganzen tag an meiner Dokumentation und an meiner PowerPoint Präsentation gearbeitet.

9 Schwierigkeiten

Ich hatte während meinem Projekt nicht sehr viele Probleme. Die meisten Aufgaben brauchten einfach eine Menge Zeit und Recherche.

9.1 Problem 1: Besetzter Port

Als ich auf meiner Website die Daten aufrufen wollte, erhielt ich eine Fehlermeldung. Das verwirrte mich sehr, da es am Vortag noch ohne Probleme funktionierte. Ich stoppte alle Server und Programme und versuchte, sie neu zu starten. Dann fiel mir auf, dass der Port, den ich für die Verbindung zur Datenbank benutze, besetzt war. Das fand ich merkwürdig, da ich nicht wusste, was meinen Port belegen könnte.

Mein erster Instinkt war, an Docker zu denken, und tatsächlich stellte sich heraus, dass einer meiner Docker-Container aus einem unerklärlichen Grund noch lief. Also beendete ich den Container und startete den Node.js-Server neu. Danach funktionierte das Auflisten meiner Produkte auf der Website wieder einwandfrei.

Diese Erfahrung hat mir gezeigt, wie wichtig es ist, die laufenden Prozesse auf meinem System im Auge zu behalten. Besonders bei der Entwicklung von Webanwendungen können solche Konflikte schnell auftreten, und es ist hilfreich, mögliche Ursachen systematisch auszuschliessen.

9.2 Problem 2: Zusammenrechnen der Preise

Als ich die Artikel im Warenkorb angezeigt habe, wollte ich auch den Gesamtpreis berechnen. Dafür habe ich zunächst den Gesamtpreis jedes einzelnen Produkts unter

dem jeweiligen Produkt angezeigt. Dazu habe ich einfach den Preis mit der Anzahl dieses Produkts multipliziert.

Anschliessend wollte ich alle diese Preise zusammenrechnen und ganz unten als Gesamtsumme anzeigen. Das funktionierte jedoch nicht ganz so, wie ich es wollte. Zwar wurden die Gesamtpreise der einzelnen Produkte korrekt addiert, aber es wurden auch die Einzelpreise der Produkte mit einbezogen. Daher war der Totalpreis viel zu hoch.

Durch eine kurze Anpassung in meinem Code konnte ich dieses Problem jedoch beheben. Ich stellte sicher, dass nur die berechneten Gesamtpreise der Produkte addiert und die Einzelpreise ignoriert wurden. Nachdem ich den Fehler korrigiert hatte, wurde der Gesamtpreis korrekt angezeigt.

10 Fazit

Während dieses Projekts habe ich sehr viel Neues gelernt und meine Fähigkeiten erheblich verbessert. Besonders im Umgang mit JavaScript habe ich grosse Fortschritte gemacht. Zudem habe ich meinen ersten richtigen Node.js-Server erstellt, was eine wertvolle Erfahrung war, da ich vorher nur wenig Wissen in diesem Bereich hatte.

Eine weitere wichtige Errungenschaft war das erfolgreiche Übertragen von Daten über Node.js. Vor diesem Projekt hatte ich noch nie damit gearbeitet, was mir nun jedoch deutlich vertrauter geworden ist. Auch im Bereich der Datenbankerstellung konnte ich mein Wissen vertiefen. Ich habe gelernt, effizientere und strukturiertere Datenbanken zu entwerfen und zu implementieren.

Zusätzlich habe ich mehr Erfahrung mit XAMPP gesammelt. Vor diesem Projekt habe ich XAMPP nur selten benutzt, aber durch die praktische Anwendung konnte ich mein Verständnis und meine Fähigkeiten im Umgang mit dieser Software erheblich erweitern.

Insgesamt war dieses Projekt eine grossartige Gelegenheit, verschiedene Technologien zu erlernen und anzuwenden. Die Herausforderungen, denen ich begegnet bin, haben mich dazu gebracht, neue Lösungen zu finden und mein technisches Wissen zu erweitern. Ich fühle mich jetzt sicherer im Umgang mit JavaScript, Node.js und Datenbanken, was mir in zukünftigen Projekten sicherlich zugutekommen wird.

11 Checkliste Dokumentation

Checkliste Individuelles Abschlussprojekt Dokumentation 2024

v1.2 - fraell - 29.5.2024

			Bemerkungen
Wichtige Hinweise	Dokumentation ist für Fachpersonen verständlich	x	
	Eigenleistung und Unterstützungen sind klar deklariert	o	
Allgemein	Kopfzeile Projektname Titel	x	
	Fusszeile Datum, Version, AutorIn, Seitenzahl	x	
	Seitenlayout: Keine Überlappung Seitenränder, Quer/Hochformat	x	
	Beschriftung der Bilder/Grafiken	o	
	Einsatz von aussagekräftiger Grafiken (Netzwerkplan, DB Schema)	x	
Titelblatt	Klar und übersichtlich gestaltet	x	
	Überbegriff: Individuelle Abschlussprojekt BLJ	x	
	Projektname (aussagekräftig / max 20 Zeichen)	x	
	Name, Abgabedatum, Name der Lehrfirma	x	
	Version des Dokuments	x	
Inhaltsverzeichnis	Kapitel nummeriert (z.B.: 2.1; 2.1.1 usw.)	x	
	Seitenzahlen korrekt angegeben	x	
	Formatierung überprüft	o	
Einleitung	Änderungstabelle/Versionerung (tabellenform)	x	
	Aufgabenstellung und Projektbeschreibung	x	
	Mögliche Risiken vor Projektbeginn	x	
Planung	Terminplan (Gantt Diagramm oder Screenshot Github Project) vorhanden	x	
	Entscheidungswege und Möglichkeiten (Entscheidungsmatrix)	o	
Hauptteil	Detaillierte Beschreibung des Vorgehens und der Zwischenschritte	x	
	Ergebnisse der Arbeit	x	
	Entscheidungen sind formuliert	o	
	Arbeitsjournal vorhanden	x	
	Testplan/Testfälle mit Ergebnisse	o	
	Persönliches Fazit	x	
Anhang	Quellenangaben und Literaturverzeichnis vorhanden	x	
	Glossar/Begriffserklärungen vorhanden *(Github / Dokument)*	x	
	Bildverzeichnis vorhanden	o	
	Programm-Code, Scripts, Foto-Dokumentation *(Github / Dokument)*	x	Ich habe den Link zum repositorie angegeben.
	Relevante KI-Chat-Prompts/Auszüge	x	
	Testplan/Testfälle (optional)	o	
	Ausgefüllte Checkliste	x	
Code/Konfigurationen *(Dateien bevorzugt auf Github)*	Link zum Github Repository vorhanden	x	
	Programm-Code folgt Clean-Code Richtlinien	x	
	Wichtige Module, Klassen, Funktionen sind kommentiert	x	
	Aussagekräftige Namen für Dateien, Klassen, Funktionen, DB Felder, ...	o	
	Programm-Dateien Header mit Autor, Datum, Version, Beschreibung	x	

12 Glossar/Begriffserklärung

12.1 XAMPP

XAMPP ist eine kostenlose und einfach zu installierende Software, die als lokaler Webserver oder Entwicklungsserver verwendet wird. Der Name XAMPP steht für "Cross-Platform, Apache, MariaDB, PHP, and Perl", was die Hauptbestandteile der Software sind. Es funktioniert auf Windows, macOS und Linux. Entwickler nutzen XAMPP, um Webanwendungen lokal zu testen und zu entwickeln, bevor sie auf einen Live-Server hochgeladen werden.

XAMPP enthält den Apache-Webserver, um Webseiten auszuliefern, das Datenbankverwaltungssystem MariaDB, die Skriptsprache PHP zur Erstellung dynamischer Webseiten und Perl für Systemadministration und Netzwerkprogrammierung. Die Installation und Einrichtung von XAMPP ist einfach und schnell. Ausserdem bietet XAMPP nützliche Werkzeuge wie phpMyAdmin zur Verwaltung von Datenbanken und FileZilla für den FTP-Server.

Mit XAMPP können Entwickler Webanwendungen lokal entwickeln und testen, was den Entwicklungsprozess beschleunigt und die Sicherheit erhöht, bevor Änderungen live gehen.

12.2 Apache

Der Apache HTTP Server, kurz Apache, ist einer der meistgenutzten Webserver weltweit. Seit seiner Veröffentlichung 1995 durch die Apache Software Foundation hat er sich als zuverlässige und flexible Lösung etabliert. Als Open-Source-Software ist Apache kostenlos und anpassbar, was ihn besonders attraktiv für Entwickler und Unternehmen macht.

Apache liefert Webseiten an Benutzer aus. Wenn ein Benutzer eine URL eingibt, verarbeitet Apache die Anfrage und sendet die entsprechenden Inhalte zurück. Bekannt für seine Zuverlässigkeit und Leistung, unterstützt Apache viele Module für zusätzliche Funktionen wie SSL-Verschlüsselung und Benutzerauthentifizierung.

Die Konfiguration erfolgt über Textdateien, was grosse Anpassbarkeit ermöglicht. Apache läuft auf verschiedenen Betriebssystemen wie Windows, macOS und Linux und wird oft zusammen mit MySQL und PHP verwendet, um dynamische Webseiten zu erstellen.

Dank umfangreicher Dokumentation und einer aktiven Entwicklergemeinschaft ist Apache einfach einzurichten und zu verwalten. Seine Stabilität und Flexibilität machen ihn zur bevorzugten Wahl für viele Webhosting-Dienste und Entwickler weltweit. Insgesamt bietet Apache eine leistungsstarke und vielseitige Plattform für die Bereitstellung von Webinhalten.

12.3 MySQL

MySQL ist ein kostenloses und beliebtes Datenbankverwaltungssystem. Es wird verwendet, um Daten zu speichern und zu verwalten. MySQL ist Open-Source-Software, das bedeutet, dass jeder den Quellcode einsehen und verändern kann. Es funktioniert auf verschiedenen Betriebssystemen wie Windows, macOS und Linux.

MySQL verwendet die Sprache SQL (Structured Query Language), um Datenbanken zu erstellen, zu ändern und abzufragen. Es ist bekannt für seine Zuverlässigkeit, hohe Leistung und Benutzerfreundlichkeit. MySQL wird oft in Webanwendungen verwendet, um Daten zu speichern, wie Benutzerdaten, Produktinformationen oder Blogbeiträge.

Die Installation und Einrichtung von MySQL ist einfach und schnell. Es gibt viele Tools und Programme, die mit MySQL arbeiten, wie phpMyAdmin, das eine grafische Oberfläche zur Verwaltung von MySQL-Datenbanken bietet. MySQL kann mit vielen Programmiersprachen wie PHP, Java und Python verwendet werden.

12.4 Node.js

Node.js ist eine kostenlose und beliebte JavaScript-Laufzeitumgebung. Sie ermöglicht es, JavaScript ausserhalb des Browsers auszuführen, oft auf Servern. Node.js ist Open-Source-Software, was bedeutet, dass jeder den Quellcode einsehen und verändern kann. Es funktioniert auf verschiedenen Betriebssystemen wie Windows, macOS und Linux.

Node.js verwendet die V8-Engine von Google, um JavaScript-Code schnell auszuführen. Es ist bekannt für seine hohe Leistung und Effizienz, besonders bei der Verarbeitung von vielen gleichzeitigen Verbindungen. Node.js wird oft für den Bau von Webservern und Echtzeitanwendungen verwendet, wie Chats oder Spiele.

Die Installation und Einrichtung von Node.js ist einfach und schnell. Es gibt viele nützliche Tools und Bibliotheken, die mit Node.js verwendet werden können, wie npm (Node Package Manager), das die Installation und Verwaltung von Bibliotheken erleichtert. Mit Node.js können Entwickler serverseitige Anwendungen in JavaScript schreiben.

12.5 PhpMyAdmin

phpMyAdmin ist eine kostenlose und beliebte Webanwendung zur Verwaltung von MySQL-Datenbanken. Es bietet eine benutzerfreundliche grafische Oberfläche, die es einfacher macht, Datenbanken zu erstellen, zu bearbeiten und zu verwalten. phpMyAdmin ist Open-Source-Software, das bedeutet, dass jeder den Quellcode einsehen und verändern kann. Es funktioniert auf verschiedenen Betriebssystemen wie Windows, macOS und Linux.

Mit phpMyAdmin können Benutzer MySQL-Datenbanken und -Tabellen anlegen, ändern, löschen und abfragen. Es bietet auch Funktionen zum Importieren und Exportieren von Daten, zur Verwaltung von Benutzerrechten und zur Ausführung von SQL-Befehlen. phpMyAdmin ist bekannt für seine Benutzerfreundlichkeit und die einfache Verwaltung von Datenbanken über das Web.

Die Installation und Einrichtung von phpMyAdmin ist einfach und schnell. Es wird oft zusammen mit Webservern wie Apache und Datenbankverwaltungssystemen wie MySQL in Softwarepaketen wie XAMPP verwendet. Benutzer können sich über ihren Webbrowser bei phpMyAdmin anmelden und sofort mit der Verwaltung ihrer Datenbanken beginnen.

12.6 Programmiersprachen

12.6.1 React

React ist eine kostenlose und beliebte JavaScript-Bibliothek zur Erstellung von Benutzeroberflächen. Sie wurde von Facebook entwickelt und ist Open-Source-Software. Mit React können Entwickler interaktive und dynamische Webanwendungen bauen.

React verwendet Komponenten, die wiederverwendbare Bausteine sind, um Webseiten zu erstellen. Diese Komponenten machen den Code übersichtlich und leicht zu pflegen. React

aktualisiert effizient nur die Teile der Webseite, die sich ändern, was die Leistung verbessert.

React kann mit anderen Bibliotheken und Frameworks verwendet werden und funktioniert gut mit modernen Webentwicklungs-Tools. Es ist eine gute Wahl für Entwickler, die schnell und effizient komplexe Benutzeroberflächen erstellen möchten.

12.6.2Java Script

JavaScript ist eine weit verbreitete Programmiersprache, die hauptsächlich zur Erstellung von interaktiven und dynamischen Webseiten verwendet wird. Sie läuft im Webbrowser und ermöglicht es, Webseiten lebendig und benutzerfreundlich zu gestalten.

Mit JavaScript können Entwickler Dinge wie Formulareingaben überprüfen, Animationen erstellen, Inhalte dynamisch ändern und mit Servern kommunizieren. Es ist einfach zu lernen und zu benutzen, was es zu einer beliebten Wahl für Webentwickler macht.

JavaScript funktioniert gut mit HTML und CSS, den anderen grundlegenden Technologien des Webs. Es ist eine flexible und leistungsfähige Sprache, die sowohl für einfache Aufgaben als auch für komplexe Anwendungen eingesetzt werden kann.

12.6.3HTML

HTML, kurz für HyperText Markup Language, ist die grundlegende Sprache, die zum Erstellen von Webseiten verwendet wird. Mit HTML können Entwickler den Inhalt einer Webseite strukturieren, indem sie Elemente wie Überschriften, Absätze, Links, Bilder und Listen definieren.

HTML besteht aus einer Reihe von Tags, die den verschiedenen Teilen einer Webseite Bedeutung und Funktion geben. Zum Beispiel wird der <h1>-Tag für Hauptüberschriften und der <p>-Tag für Absätze verwendet.

HTML ist einfach zu lernen und bildet die Basis für das Webdesign. Es arbeitet zusammen mit CSS (für das Design) und JavaScript (für die Interaktivität), um vollständige und funktionale Webseiten zu erstellen.

12.6.4CSS

CSS, kurz für Cascading Style Sheets, ist eine Sprache, die verwendet wird, um das Aussehen und Layout von Webseiten zu gestalten. Mit CSS können Entwickler Farben, Schriftarten, Abstände, Grössen und Positionen von HTML-Elementen festlegen.

CSS arbeitet zusammen mit HTML, um Webseiten ansprechend und benutzerfreundlich zu gestalten. Zum Beispiel kann der Hintergrund einer Webseite mit CSS eingefärbt und die Textgrösse angepasst werden.

CSS ist einfach zu lernen und ermöglicht es, das Design einer Webseite flexibel und effizient zu ändern. Es ist ein wichtiges Werkzeug für Webdesigner und Entwickler, um Webseiten attraktiv und funktional zu gestalten.

13 Quellenverzeichnis

13.1 Bilder

XAMPP Logo: https://de.m.wikipedia.org/wiki/Datei:Xampp_logo.svg

Apache Logo: https://www.brcline.com/blog/setting-up-caching-on-apache/apache-logo

MySQL Logo: https://1000logos.net/mysql-logo/

Node.js Logo: https://www.svgrepo.com/svg/376337/node-js

phpMyAdmin Logo:
https://commons.wikimedia.org/wiki/File:PhpMyAdmin_logo_2010_hidef.svg

React Logo: https://iconduck.com/icons/13180/react-original-wordmark

Java Script Logo: https://logos-world.net/javascript-logo/

HTML Logo: https://en.m.wikipedia.org/wiki/File:HTML5_logo_and_wordmark.svg

CSS Logo: https://freebiesupply.com/logos/css3-logo/

13.2 Informationen, Recherche & Programme

https://draw.io

https://www.w3schools.com

https://stackoverflow.com

https://github.com

14 Relevante KI-Chat Prompts

Prompt: Wie integriere ich Routing in einer React-Anwendung?

Prompt: Wie debugge ich Probleme mit einem Node.js-Server?

Prompt: Erkläre, wie man Daten aus einem HTML-Formular in eine Datenbank einfügt.

Prompt: Was bedeutet dieser Errorcode: node:events:492 throw er; // Unhandled 'error' event ^

15 Link zum GitHub Repositorie

https://github.com/FlyCaptainRex/Abschlussprojekt