

FACHINFORMATIKER/IN FÜR ANWENDUNGSENTWICKLUNG

## Protokoll zur betrieblichen Projektarbeit

Diese Erklärung ist der Dokumentation als erste Seite vor dem Deckblatt beizufügen.

Hiermit versichere ich,

Andreas Unger

Vor- und Zuname des Auszubildenden

die betriebliche Projektarbeit

DashBoard - Entwicklung

Genaue Bezeichnung der Projektarbeit

sowie die eingereichte Dokumentation unter Betreuung von

Oliver Bogdan, Synergie

Vor- und Zuname des Ausbilders/ Name des Ausbildungsbetriebes

**selbstständig** und **ohne fremde Hilfe** konzipiert, verfasst und durchgeführt zu haben. Teile der Dokumentation, die ich **nicht selbstständig** erstellt habe, sind von mir entsprechend **gekennzeichnet** worden. Die von der Verordnung vorgesehene Richtigkeit wurde eingehalten. Die Arbeit hat in dieser Form keiner anderen Prüfungsinstitution vorgelegen.

Ich bin darüber aufgeklärt worden, dass meine betriebliche Projektarbeit bei **Täuschungshandlungen, bzw. Ordnungsverstößen mit „Null“ Punkten bewertet** wird und als nicht bestanden gilt.

Ich bin weiter darüber aufgeklärt worden, dass dies auch dann gilt, wenn festgestellt wird, dass meine Projektdokumentation im Ganzen oder zu Teilen mit der eines anderen Prüfungsteilnehmers übereinstimmt. Ich nehme zur Kenntnis, dass ggf. stichprobenartige Kontrollen durchgeführt werden können.

Berlin, 07.12.2022

Ort und Datum

Unger

Unterschrift des Prüfungsteilnehmers

[Signature]

Unterschrift Projektverantwortliche/-r des Auftraggebers



Abschlussprüfung Winter 2022/2023

Fachinformatiker für Anwendungsentwicklung

Dokumentation zur betrieblichen Projektarbeit

## **Dashboard Entwicklung**

**Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich**

Abgabedatum: Berlin, den 07.12.2022

**Prüfungsbewerber:**

Andreas Unger

Eiserfelder Ring 32

13583 Berlin

Telefon: 0179 426 14 94

**Ausbildungsbetrieb:**

Synergie

Hohenstaufenstraße 32

10779 Berlin



## Inhaltsverzeichnis

Inhaltsverzeichnis.....	I
Abbildungsverzeichnis.....	III
Tabellenverzeichnis.....	IV
Abkürzungsverzeichnis.....	V
1 Einleitung .....	1
1.1 Projektumfeld .....	1
1.2 Projektziel.....	1
1.3 Projektbegründung .....	1
1.4 Projektschnittstellen .....	1
1.5 Projektabgrenzung .....	1
2 Projektplanung .....	2
2.1 Projektphasen .....	2
2.2 Ressourcenplanung .....	2
2.3 Entwicklungsprozess.....	3
3 Analysephase.....	3
3.1 Ist-Analyse .....	3
3.2 Wirtschaftlichkeitsanalyse .....	3
3.2.1 Make or Buy-Entscheidung .....	3
3.2.2 Projektkosten .....	3
3.2.3 Amortisationsdauer .....	4
3.3 Nutzwertanalyse.....	5
3.4 Anwendungsfälle .....	5
3.5 Qualitätsanforderungen .....	5
3.6 Lastenheft/Fachkonzept .....	5
4 Entwurfsphase .....	5
4.1 Zielplattform .....	5
4.2 Architekturdesign.....	6
4.3 Entwurf der Benutzeroberfläche .....	6
4.4 Datenmodell .....	6
4.5 Geschäftslogik.....	7
4.6 Maßnahmen zur Qualitätssicherung .....	7
4.7 Pflichtenheft/Datenverarbeitungskonzept .....	7
5 Implementierungsphase .....	8
5.1 Implementierung der Datenstrukturen .....	8
5.2 Implementierung der Benutzeroberfläche .....	8
5.3 Implementierung der Geschäftslogik .....	9

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### Inhaltsverzeichnis

---

6	Abnahmephase .....	9
7	Einführungsphase .....	9
8	Dokumentation .....	10
9	Fazit .....	10
9.1	Soll-/Ist-Vergleich .....	10
9.2	Lessons Learned.....	10
9.3	Ausblick.....	11
	Literaturverzeichnis .....	12
	Eidesstattliche Erklärung .....	13
	Anhang.....	i
A1	Detaillierte Zeitplanung.....	i
A2	Lastenheft (Auszug) .....	ii
A3	Use-Case-Diagramm.....	iii
A4	Pflichtenheft (Auszug) .....	iii
A5	Datenbankmodell .....	iv
A6	Struktogramm.....	v
A7	Oberflächenentwürfe .....	vi
A8	Screenshots der Anwendung.....	vii
A9	Entwicklerdokumentation (Oberfläche) .....	viii
A10	Entwicklerdokumentation.....	x

### Abbildungsverzeichnis

Abbildung 1: Use-Case-Diagramm .....	iii
Abbildung 2: Entity-Relationship-Model .....	iv
Abbildung 3: SQL .....	iv
Abbildung 4: Struktogramm registrieren.....	v
Abbildung 5: Screenshot Registrierung .....	vii
Abbildung 6: Screenshot Login.....	vii
Abbildung 7: HTML-Register .....	viii
Abbildung 8: HTML-Login.....	ix
Abbildung 9: Registrieren .....	x
Abbildung 10: Login.....	xi
Abbildung 11: Eingeloggt - Kontrolle.....	xii

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### Tabellenverzeichnis

---

#### Tabellenverzeichnis

Tabelle 1: Grobe Zeitplanung .....	2
Tabelle 2: Kostenaufstellung .....	4
Tabelle 3: Soll-/Ist-Vergleich.....	10
Tabelle 4: Detaillierte Zeitplanung .....	ii

## Abkürzungsverzeichnis

API .....	<i>Application Programming Interface</i>
CSS.....	<i>Cascading Style Sheets</i>
ERM .....	<i>Entity Relationship Model</i>
GUI.....	<i>Graphical User Interface</i>
HTML .....	<i>Hypertext Markup Language</i>
MVC .....	<i>Model View Controller</i>
PHP.....	<i>PHP Hypertext Preprocessor</i>
SQL.....	<i>Structured Query Language</i>
XML.....	<i>Extensible Markup Language</i>

## 1 Einleitung

### 1.1 Projektumfeld

Der Sitz der Praxis Synergie liegt in Berlin-Schöneberg. Die Praxis betreut ihre Patienten im Bereich Physiotherapie, Ergotherapie, Heilpraktiker, Rehasport, Yoga, Pilates und Sanitätshaus.

Dabei geht es in erster Linie um die interdisziplinär als Team, aber auch um das ganzheitliche Konzept, dem Patienten bei seinen Einschränkungen / Schmerzen / Erkrankung / o.ä. zu helfen, den Schmerz zu lindern oder eine Option zu bieten, mit der man seinen Alltag einfacher bewältigen kann.

### 1.2 Projektziel

Ziel des Projektes ist die Erstellung von einem Dashboard – Account Login (JavaScript) mit einer Anbindung an eine Datenbank (MySQL) zu einer bestehenden Website (Informationsseite) welcher die Registrierung, Einloggen und Ausloggen vom Kunden sowie Admin-Dashboard-Zugang übernimmt. Der Login-bereich wird zum Teil vorgefertigte auswählbare Felder erleichtert, einfachere Handhabung für den Kunden. Es wird darauf geachtet das wirklich nur die notwendigsten Daten der Kunden erfasst werden damit auch nicht zu viele Daten gespeichert werden

### 1.3 Projektbegründung

Zurzeit hat die Firma noch kein derartiges Webangebot für Ihre Kunden. Bestellungen oder Terminvereinbarungen können nur telefonisch oder persönlich abgewickelt werden. Dies kostet viel Zeit und ist auch nicht mehr zeitgemäß. Um Kundenfreundlicher zu werden entschied man sich zu diesem Schritt.

### 1.4 Projektschnittstellen

Die Anwendung läuft auf einem Webserver. Um an die Daten zu gelangen, muss die Webanwendung mit einer MySQL-Datenbank kommunizieren und Werte abfragen können. Diese befindet sich im gleichen Web-Server und kann direkt angesprochen werden.

Die Personalabteilung der Webanwendung sollen eng in den Entwicklungsprozess eingebunden werden. Durch ständigen Austausch soll eine schnelle Reaktion auf Wünsche und Änderungen erfolgen können.

### 1.5 Projektabgrenzung

Am Ende des gesamten Projekts soll ein Online-Shop für das Sanitätshaus entwickelt werden.



## 2 Projektplanung

### 2.1 Projektphasen

Für die Entwicklung des Projektes standen 70 Stunden zur Verfügung. Im Rahmen der Projektplanung wurden die Stunden auf verschiedene Phasen aufgeteilt und somit der gesamte Projektablauf abgebildet. Eine detaillierte Zeitplanung mit den einzelnen Schritten der jeweiligen Phasen befindet sich im Anhang A1 Detaillierte Zeitplanung auf Seite.

Projektphase	Geplante Zeit
Analyse	7 h
Entwurf	15 h
Implementierung	37 h
Abnahme und Einführung	2 h
Dokumentation	9 h
<b>Gesamt</b>	<b>70 h</b>

**Tabelle 1: Grobe Zeitplanung**

### 2.2 Ressourcenplanung

Räumlichkeit

- Büroarbeitsplatz

Hardware

- Desktop-PC mit 2 Monitoren und ein Laptop

Software

- Windows 11 Pro – Betriebssystem
- PhpStorm – Entwicklungsumgebung IDE
- MySQL – Datenbankverwaltungssystem der relationalen Datenbank
- XAMP - Installieren und Konfigurieren des Webserver Apache mit der Datenbank MySQL

Personal

- Auszubildender – Umsetzung des Projekts
- Anwendungsentwickler – Review der Pull-Requests
- Mitarbeiter der Marketingabteilung
- Geschäftsführerin

## 2.3 Entwicklungsprozess

Bei der Bearbeitung des Projektes wurde nach dem Wasserfallmodell vorgegangen. Das Wasserfallmodell ist ein Vorgehensmodell der klassischen Softwareentwicklung, bei dem die Projektphasen sequentiell durchlaufen und bearbeitet werden. Es wurden jedoch auch Aspekte aus einem iterativen Entwicklungsprozess eingebunden. Um sicherzustellen, dass die Umsetzung des Projektes den gewünschten Anforderungen und Vorstellungen des Betreuers entsprach, wurden ihm Zwischenergebnisse präsentiert und notwendige Änderungen besprochen. Dabei wurde allerdings auf feste Zeitfenster bzw. Sprints oder Sprintziele mit Inkrementen (wie z. B. bei Scrum) verzichtet.

# 3 Analysephase

## 3.1 Ist-Analyse

Wie bereits in Abschnitt 1.3 (Projektbegründung) erwähnt wurde, hat die Firma noch kein derartiges Webangebot für Ihre Kunden. Bestellungen oder Terminvereinbarungen können nur telefonisch oder persönlich abgewickelt werden. Dies kostet viel Zeit und ist auch nicht mehr zeitgemäß. Um kundenfreundlicher zu werden entschied man sich zu diesem Schritt.

## 3.2 Wirtschaftlichkeitsanalyse

Aufgrund des geschilderten Sachverhalts, der schon in Abschnitt 1.3 (Projektbegründung) und Abschnitt 3.1 (Ist-Analyse) beschrieben wurde, steht die Notwendigkeit des Projekts außer Frage. Im weiteren soll die Wirtschaftlichkeit des Projekts genauer untersucht werden.

### 3.2.1 Make or Buy-Entscheidung

In Zusammenarbeit mit der Personalabteilung wird in einer Internetrecherche nach Open-Source-Lösungen gesucht und bei einem Softwarelieferanten ein Angebot für die Entwicklung der gewünschten Funktionalitäten eingeholt. Aufgrund der Individualität kommen keine Branchenlösungen in Frage. Auch eine externe Entwicklung kommt aufgrund eines hohen Preises für die Umsetzung der gewünschten Funktionalitäten nicht in Frage. Daher wird die Entscheidung getroffen, die Anwendung intern zu entwickeln.

### 3.2.2 Projektkosten

Für die Durchführung des Projekts setzen sich Kosten aus Ressourcen und Personalkosten zusammen. Ein Auszubildender im dritten Lehrjahr, nach dem Tarifvertrag in Berlin verdient ca. 1120 €

$$8 \frac{\text{h}}{\text{Tag}} \cdot 220 \frac{\text{Tage}}{\text{Jahr}} = 1.760 \frac{\text{h}}{\text{Jahr}}$$
$$1.120 \frac{\text{€}}{\text{Monat}} \cdot 12 \frac{\text{Monate}}{\text{Jahr}} = 13.440 \frac{\text{€}}{\text{Jahr}}$$

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### Analysephase

---

$$\frac{13.440 \frac{\text{€}}{\text{Jahr}}}{1.760 \frac{\text{h}}{\text{Jahr}}} \approx 7,64 \frac{\text{€}}{\text{h}}$$

Es ergibt sich also ein Stundensatz von *7,56 EUR*. Die Durchführungszeit des Projekts beträgt 70 Stunden. Für die Nutzung von Ressourcen<sup>1</sup> wird ein pauschaler Stundensatz von *15 EUR* angenommen. Für die anderen Mitarbeiter wird pauschal ein Stundensatz von *25 EUR* angenommen. Eine Aufstellung der Kosten befindet sich in Tabelle 2 und sie betragen insgesamt *2.544,80 EUR*.

Vorgang	Zeit	Kosten / Stunde	Kosten
Entwicklung	70 h	7,64 € + 15 € = 22,64 €	1.584,80 €
Fachgespräch	3 h	25 € + 15 € = 40,00 €	120,00 €
Abnahme	1 h	25 € + 15 € = 40,00 €	40,00 €
Schulung	20 h	25 € + 15 € = 40,00 €	800,00 €
<b>Gesamt</b>			<b>2.544,80 €</b>

**Tabelle 2: Kostenaufstellung**

### 3.2.3 Amortisationsdauer

Einsparung von potenziellen Kunden die sich selber im Dashboard (Online-Shop) die Personen-Daten eintragen. So muss man die eingegangenen Bestellungen (Personen-Daten) manuell am PC eintragen.

Bei einer Zeiteinsparung von 30 Minuten am Tag und 220 Arbeitstagen im Jahr ergibt sich eine gesamte Zeiteinsparung von:

$$220 \frac{\text{Tage}}{\text{Jahr}} \cdot 30 \frac{\text{min}}{\text{Tag}} = 6.600 \frac{\text{min}}{\text{Jahr}} \approx 110 \frac{\text{h}}{\text{Jahr}}$$

Dadurch ergibt sich eine jährliche Einsparung von:

$$917\text{h} \cdot (25 + 15) \frac{\text{€}}{\text{h}} = 4.400 \text{ €}$$

Die Amortisationszeit beträgt also:

$$\frac{2544,80 \text{ €}}{4400 \text{ €/Jahr}} = 0,58 \text{ Jahre} = 7 \text{ Monate}$$

---

<sup>1</sup> Räumlichkeiten, Arbeitsplatzrechner etc.

### 3.3 Nutzwertanalyse

Der Unterschied zwischen automatisiert und händisch durch einen Mitarbeiter ist in diesem Fall gravierend, da automatisiert ohne größeren Aufwand nahezu Echtzeit Updates garantieren kann, während manuell unter sehr großem Aufwand eingeben muss. Zudem ist das manuelle Erstellen mit deutlich mehr Aufwand verbunden und geht nur solange gut, wie der Mitarbeiter nicht krank wird.

### 3.4 Anwendungsfälle

Es wird im Zuge der Analyse des Projektes ein Anwendungsfalldiagramm erstellt. Dies stellt Interaktionen von Benutzern mit dem System dar und zeigt somit das erwartete Verhalten der Anwendung. Das Anwendungsfalldiagramm ist im Anhang A3 Use-Case-Diagramm, Seite iii dargestellt. Es gibt die Akteure Kunde und der Administrator. Der Administrator ist ein spezialisierter Akteur, er kann zudem Benutzer anlegen, verwalten und weitere Administratoren festlegen. Der Administrator kann nicht nur sich von dem Dashboard an-/abmelden, sondern auch andere Benutzer.

### 3.5 Qualitätsanforderungen

Unter Berücksichtigung von Projektumfeld, Ziel und Begründung wurden von dem Auszubildenden und den Teams um das Dashboard herum Anforderungen in Form eines Lastenheftes definiert. In A2 findet sich ein Auszug davon.

### 3.6 Lastenheft/Fachkonzept

Zum Abschluss der Analysephase wurde vom Autor ein Lastenheft angefertigt. Als Basis dienten die Anforderungen an das Projekt, die vom Projektmanagement vor dem Projektbeginn angegeben wurden. Das Lastenheft dient der klaren Auflistung aller Anforderungen, deren Wechselwirkung untereinander und möglichen Risiken, die jeweils auftreten können. Ein Auszug des Lastenheftes befindet sich im Anhang unter A2, Seite ii Auszug Lastenheft.

## 4 Entwurfsphase

### 4.1 Zielplattform

Wie bereits in Abschnitt 1.2 (Projektziel) erwähnt, ist das festgelegte Ziel, eine Web-Anwendung mit Datenbankbindung zu entwickeln, die im Internet erreichbar sein soll. Die Programmierung wurde mit der Sprache JavaScript: Node.js umgesetzt sowie mithilfe von Bibliotheken bzw. Frameworks wie Express, Handlebars.js (Frontend-Template), im Bereich CSS (Bootstrap) und einer Datenbank mit Datenbanksystem MySQL entwickelt.

Außerdem wird die Webanwendung auf dem Node.js ausgeführt, ist eine plattformübergreifende Open-Source-JavaScript-Laufzeitumgebung, die JavaScript-Code außerhalb eines Webbrowsers ausführen kann. Damit wird ein Webserver betrieben.

#### 4.2 Architekturdesign

Das Projekt basiert auf dem Architekturmuster Model View Controller (MVC). Gemäß diesem Muster lässt sich Software in die drei Einheiten Model (Datenhaltung), View (Präsentation) und Controller (Anwendungssteuerung) unterteilen.

**Modell:** Das Modell stellt die Datenstruktur, das Format und die Einschränkungen dar, mit denen es gespeichert wird. Es verwaltet die Daten der Anwendung. Im Wesentlichen ist es der Datenbankteil der Anwendung.

**Ansicht:** Ansicht ist das, was dem Benutzer präsentiert wird. Ansichten verwenden das Modell und präsentieren Daten in einer Form, in der der Benutzer es wünscht. Einem Benutzer kann auch gestattet werden, Änderungen an den dem Benutzer präsentierten Daten vorzunehmen. Sie bestehen aus statischen und dynamischen Seiten, die gerendert oder an den Benutzer gesendet werden, wenn der Benutzer sie anfordert.

**Controller:** Controller steuert die Anfragen des Benutzers und generiert dann eine entsprechende Antwort, die dem Betrachter zugeführt wird. Typischerweise interagiert der Benutzer mit der Ansicht, die wiederum die entsprechende Anfrage generiert, diese Anfrage wird von einem Controller bearbeitet. Der Controller rendert die entsprechende Ansicht mit den Modelldaten als Antwort.

#### 4.3 Entwurf der Benutzeroberfläche

Die Benutzeroberfläche der Webanwendung soll klar strukturiert und einfach sein. Hierzu werden zunächst MockUps erstellt, die den groben Aufbau und die Anordnung der Elemente der einzelnen Seiten zeigen. Zwei MockUps werden im Anhang A7, Seite vi dargestellt. Diese werden zum einen als Orientierung für Entwickler erstellt, zum anderen werden sie intensiv in die Gespräche mit dem Fachbereich einbezogen. Die Mitarbeiter des Fachbereichs, die später als Administratoren die Webanwendung und die Benutzer betreuen, haben sich für ein Design, das in das Corporate Design der Synergie passt und schon in anderen Webanwendungen verwendet wird, entschieden. Die Website besteht aus einem Header mit Logo und Navigationsleiste. Die Inhalte werden in einem Gold-Braun abgegrenzten Bereich angezeigt. Ein nicht registrierter Nutzer sieht nur die Verifikationsabfrage, den Login-Bereich und informative Texte zum Service.

#### 4.4 Datenmodell

Im nächsten Schritt wurde das Projekt anhand der Frage analysiert, welche Daten innerhalb der jeweiligen Anwendungsfälle auftreten, wie sie zu speichern sind und in welcher Beziehung sie zueinanderstehen. In Folge dessen wurden mehrere Entitätstypen ausgearbeitet, die die Daten fassen sollen. Anhand dieser Ausarbeitung wurde ein ERM erstellt. Dieses enthält die einzelnen Entitäten und veranschaulicht grafisch die Beziehungen untereinander.

Das Datenmodell besteht aus Entitäten Kunde, Adresse, Daten und der Administrator. Kunde und der Admin speichern dabei den Namen, Passwort und die E-Mail-Adresse.

### Entwurfsphase

---

Eine Adresse gehört genau einem Kunden. Ein Kunde kann mehrere Adressen haben. Diese Beziehungen sind im Anhang A5 Entity-Relationship-Modell auf Seite iv als Entity-RelationshipModell (ERM) dargestellt.

## 4.5 Geschäftslogik

Das Dashboard besteht im Wesentlichen aus vier großen Segmenten:

- Datenbank erstellen sowie abfragen und schreiben in die Datenbank
- Registrierung
- Einloggen & Ausloggen
- Benutzerbereich: Kunde und Admin

Damit die Onlinefunktionalität des Dashboards geschaffen werden kann müssen eben diese vier Segmente vorhanden sein. Das erste Segment ist das Erstellen der Datentabellen und Datensätzen in die MySQL-Datenbank.

Nachdem die Daten über das erste Segment erstellt wurden, gilt es über das zweite und dritte Segment diese Daten in die Datenbank zu übernehmen.

Im vierten Segment muss zur Sicherung der Datenintegrität eine Prüfung der Daten stattfinden und die übernommenen Daten aus der Datenbank auszulesen und dem Benutzerbereich zu stellen.

Ein Struktogramm, welches die Klassen der Anwendung und deren Beziehungen untereinander darstellt, kann im Anhang A6, Seite v eingesehen werden.

## 4.6 Maßnahmen zur Qualitätssicherung

Vor dem Deployment testet der Autor auf generelle Funktionalität.

Beim Dashboard handelt es sich um eine Anwendung, die in Form und Funktion künftig erweitert und angepasst werden soll. Im Zuge des Entwicklungsprozesses wurde Wert daraufgelegt, dass die genutzten Funktionen ohne Abnahme der Qualität geändert und angepasst werden können. Während des Entwicklungsprozesses wurden zwei Testphasen eingeplant. Dieser Test wurde mit Personen aus dem näheren Heilpraktiker Umfeld durchgeführt, um zugleich die Usability des Programms von Personen testen zu lassen, die über keine oder gering ausgeprägte IT-Affinität verfügen. Der Abschlusstest wurde am 07.12.2022 mit dem Auftraggeber durchgeführt.

## 4.7 Pflichtenheft/Datenverarbeitungskonzept

Zum Schluss der Entwurfsphase wurde ein Pflichtenheft erstellt. Dieses ist im Anhang A4, Seite iii zu finden und beschreibt die Umsetzung des Anfangs definierten Ziele und Anforderungen. Somit dient es als Leitfaden während der Entwicklung des Projektes.

## 5 Implementierungsphase

Die in der Entwurfsphase entworfenen Skizzen werden nun genauer betrachtet und es geht an die eigentliche Umsetzung des Projektes, durch Programmcode.

### 5.1 Implementierung der Datenstrukturen

Anhand des im Abschnitt 4.4 Datenmodell erarbeiteten ERM erfolgte in der nächsten Phase zunächst die Implementierung der Datenstruktur. Dafür wurden als Erstes zu Testzwecken die benötigten Entitäten händisch vom Autor im phpMyAdmin angelegt, um unter Zuhilfenahme von Testdaten den Datenzugriff über Referenzen zu simulieren.

Nach erfolgreichem Abschluss dieser Testphase wurde die SQL-Logik zur Erstellung der Datenbankobjekte innerhalb des firmeninternen Webservers implementiert im Anhang A5 Abbildung 3, Seite iv dargestellt, welcher mit Hilfe einer fortlaufenden Versionierung das Updaten von bestehenden Kundendatenbanken realisiert.

### 5.2 Implementierung der Benutzeroberfläche

Auf Grundlage der unter A7, Seite vi (Entwurf der Benutzeroberfläche) beschriebenen Mockups konnte die Benutzeroberfläche implementiert werden. Umgesetzt wurde die Oberfläche mit der Handlebars.js (Handlebars kompiliert Vorlagen in JavaScript-Funktionen. Dadurch wird die Template-Ausführung schneller als bei den meisten anderen Template-Engines). Handlebars beinhaltet grundsätzlich selbst keinen JavaScript-Code, sondern ist eine Extensible Markup Language (XML)-Datei, die nach Hypertext Markup Language (HTML) gerendert wird, damit die Anwendung im Browser angezeigt werden kann. Über Ausdrücke, die von `{{}}`-Blöcken umgeben werden, lässt sich auf Methoden der Controller zugreifen, um so eine Verknüpfung zum Code herzustellen. Mit folgendem Tag lässt sich beispielsweise über eine Nachricht iterieren:

```
<h4 class="alert alert-danger mt-4">{{message}}</h4>
```

Die Gestaltung der Oberfläche wurde dann mit eingebundenen Cascading Style Sheets (CSS)-Dateien umgesetzt. Unter anderem wurde dabei das CSS-Framework Bootstrap eingesetzt. Im Anhang A21: Screenshots auf Seite xxii befinden sich Screenshots der Anwendung, die nach der Implementierungsphase entstanden sind.

Bei der Gestaltung der Oberfläche wie auch während der gesamten Entwicklung wurden die Softwareergonomie-Richtlinien beachtet. Dazu zählt vor allem die Aufgabenangemessenheit durch eine Minimierung der Interaktionen und geeigneter Funktionalität und die Fehlertoleranz.

Durch den Paket-Manager von Node.JS „npm“ verlief die Installation von den Handlebars.js (Handlebars.js bietet die nötige Leistung, damit semantische Vorlagen effektiv und ohne Frustration erstellt werden können) sehr einfach.

Zur Registrierung wurden vier Input Pflichtfelder angelegt siehe A10 Abbildung 9, Seite x.

Und A9 Abbildung 7, Seite viii. Name, zwei Passwortabfragen zur Abgleichung der Richtigkeit, E-Mail-Adresse und Datenschutz.

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### Abnahmephase

---

Login: Zwei Input-Felder wurden angelegt, die E-Mail-Adresse und das Passwort siehe A9 Abbildung 8, Seite ix. Und A10 Abbildung 10, Seite xi

Logout: Der Benutzer kann sich aus dem Dashboard ausloggen siehe A10 Abbildung 11, Seite xii.

Dashboard: Authentifizierte Benutzer die auch in ihrem Benutzereingang landen. Hier kann der Benutzer seine Personendaten einsehen und ändern siehe A10 Abbildung 11, Seite xii.

### 5.3 Implementierung der Geschäftslogik

Die Implementierung der Oberfläche und der Datenstrukturen ermöglicht die Implementierung der Geschäftslogik. Diese ist dabei hauptsächlich in verschiedenen Service-Funktionen implementiert worden, die von Controller-Klassen aufgerufen werden. Für jede View-Funktion, die Suche auf dem Datenbanksystem, wurden je eine Controller- und eine Service-Funktion erstellt. Außerdem gibt es Passwort-Hashing -Funktion für die Ver- und Entschlüsselung der Hashed-Passwörter (Das sog. Passwort-Hashing dient einem erhöhten Datenschutz und wandelt Passwörter in einer Art und Weise in festgelegte Zeichen- und Symbolfolgen um, die es auch Hacker:innen, die sich Zugriff zu einem Firmensystem verschaffen, nicht ohne weiteres ermöglicht, diese auszulesen. Werden Kennwörter hingegen als Klartext in einem Firmensystem gespeichert, können sie ohne Umstände missbraucht werden).

## 6 Abnahmephase

Vor der Vorführung dem Personalbereich, fand ein Code-Review durch den Anwendungsentwickler statt. Dabei wurde neben fachlichen und technischen Fehlern vor allem auch auf die Verständlichkeit des Codes in Bezug auf die Benennung von Variablen und Komplexität der Methoden.

Zudem wurde auf Fehler und Warnungen in der Kommandozeile des Servers geachtet, so wie in den Entwickler-Tools im Browser.

Die Anwendung wurde nach der Vorführung durch die Personalabteilung abgenommen.

## 7 Einführungsphase

Die Anwendung wurde weitestgehend automatisiert. Eine Schulung der Mitarbeiter war durch den ständigen Austausch während des Projekts nicht notwendig. Die fertige Anwendung wurde Geschäftsführerin und der Personalabteilung im Rahmen der Einführungsphase in der dafür vorgesehenen Zeit vorgeführt.

Parallel wurde die Anwendung durch die Personalabteilung in verschiedenen Browsern und auf einem Smartphone getestet. Dies hatte den Vorteil, dass abschließend nochmal auf potenzielle Fehler geachtet und die Anwender gleichzeitig mit der Bedienung vertraut gemacht wurden.



## 8 Dokumentation

Die Dokumentation der Anwendung setzt sich neben der Projektdokumentation, in der die während der Umsetzung des Projektes durchlaufenden Phasen beschrieben werden, auch aus der Entwicklerdokumentation zusammen. Die Entwicklerdokumentation beinhaltet eine detaillierte Beschreibung. Dazu gehören auch deren Attribute und Methoden sowie untereinander bestehenden Abhängigkeiten. Diese Dokumentation soll bei der Weiterentwicklung der Anwendung oder einer Anpassung durch einen anderen Entwickler als Übersicht und Nachschlagewerk dienen.

## 9 Fazit

### 9.1 Soll-/Ist-Vergleich

Entwickler und Mitarbeiter anderer Abteilungen haben das Dashboard-System gesehen und als nützlich empfunden. Somit wurde das Projektziel erreicht. Der Zeitplan wurde bis auf die Entwurfsphase eingehalten. Die eine Stunde Differenz ist damit zu begründen, dass das Erstellen des Pflichtenhefts schneller ging als erwartet, durch gemeinsames Brainstorming des Auszubildenden und seines Teams.

Wie in Tabelle 3 zu erkennen ist, konnte die Zeitplanung bis auf wenige Ausnahmen eingehalten werden.

Phase	Geplant	Tatsächlich	Differenz
Analyse	7 h	7 h	
Entwurf	15 h	14 h	-1 h
Implementierung	37 h	37 h	
Abnahme	1 h	1 h	
Einführung	1 h	1 h	
Dokumentation	9 h	10 h	+1 h
<b>Gesamt</b>	<b>70 h</b>	<b>70 h</b>	

**Tabelle 3: Soll-/Ist-Vergleich**

### 9.2 Lessons Learned

Anhand der Umsetzung dieses Abschlussprojektes konnte der Autor wertvolle Erfahrungen über die Arbeit an einem Projekt mit allen dazugehörigen Arbeitsschritten sammeln. Vor allem in der Planung und Entwurfsphase sowie in der Absprache mit dem Fachbereich konnten neue Erkenntnisse gewonnen werden. Spannend war vor allem der Einsatz von verschiedenen Frameworks, um alle Komponenten der JavaScript Anwendung testen zu können. Zudem konnte der Autor auch in Bezug auf die Node.js (ist eine plattformübergreifende Open-Source-JavaScript-Laufzeitumgebung, die JavaScript-Code außerhalb eines Webbrowsers ausführen kann. Damit kann zum Beispiel ein Webserver betrieben werden) viel Neues lernen. Dazu

## **DASHBOARD ENTWICKLUNG**

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### **Fazit**

---

gehört unter anderem die Anbindung an Datenquellen mit MySQL oder die Gestaltung der Oberflächen mit Handlebars.js.

### **9.3 Ausblick**

In diesem Projekt wurden ausnahmslos alle Anforderungen und Wünsche des Auftraggebers umgesetzt: Zukünftig ist ein Online-Shop geplant.

## Literaturverzeichnis

Rheinwerk Verlag GmbH, Bonn, 2018. *JavaScript lernen und verstehen. inkl. Objektorientierter und funktionaler Programmierung.*

Rheinwerk Computing, Bonn, 2021. *Node.js – Hochperformante Webanwendungen mit JavaScript.* Sebastian Springer.

Rheinwerk Computing, Bonn, 2022. *Handbuch für Softwareentwickler – Das Lehr- und Nachschlagewerk für professionelles Software Engineering.*

<https://developer.mozilla.org/>

<https://www.google.de/>

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### Eidesstattliche Erklärung

---

### Eidesstattliche Erklärung

Ich, Andreas Unger, versichere hiermit, dass ich meine Dokumentation zur betrieblichen Projektarbeit mit dem Thema

*Dashboard Entwicklung – Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich*

selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, wobei ich alle wörtlichen und sinngemäßen Zitate als solche gekennzeichnet habe. Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Abgabeort, den 07.12.2022

---

ANDREAS UNGER

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### Anhang

---

## Anhang

### A1 Detaillierte Zeitplanung

Phase	Stunden
Analyse	7 h
Durchführung der Ist-Analyse	2 h
Durchführen der Wirtschaftlichkeitsanalyse und Amortisationsrechnung	1 h
Ermittlung von Anwendungsfällen inkl. Erstellung eines Use-Cases	2 h
Unterstützung des Fachbereichs beim Erstellen des Lastenhefts	2 h
Planung / Entwurf	15 h
Planung der Ressourcen (Zeit, Kosten, Personal, Hardware und Software)	2 h
Entwerfen der Benutzeroberfläche inkl. Erstellung von Mock-Ups	2 h
Entwerfen der Datenbankstruktur inkl. Erstellung eines ER-Modells	2 h
Erstwerfen eines Sequenzdiagramms	1 h
Entwurf des Aktivitätsdiagramms	1 h
Planung der Architektur	2 h
Planung der Maßnahmen zur Qualitätssicherung	2 h
Erstellung des Pflichtenhefts	3 h
Implementierung inkl. Tests	37 h
Eine Entwicklungsumgebung (IDE) einrichten	1 h
Implementierung der Oberfläche	2 h
Implementierung der Datenbank	5 h
Implementierung des Login-Bereichs	13 h
- Implementierung der Registrierung	3 h
- Implementierung des Einloggens	3 h
- Implementierung des Ausloggens	3 h
- Implementierung des Kundenbereichs	4 h
Implementierung - des Administrator-Panels	12 h
- Implementierung des Einloggens	3 h
- Implementierung des Ausloggens	3 h
- Implementierung des Admin-Dashboards	6 h
Testphase und Fehleranalyse: Test auf Funktion der Datenbank, des Login-Bereichs, und des Admin-Dashboards	4 h
Abnahme und Einführung	2 h

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### Anhang

---

Code-Review	1 h
Abnahme und Übergabe an den Fachbereich	1 h
Dokumentation	9 h
Erstellen der Projektdokumentation	9 h
Gesamt	70 h

**Tabelle 4: Detaillierte Zeitplanung**

## A2 Lastenheft (Auszug)

Es folgt ein Auszug aus dem Lastenheft mit Fokus auf die Anforderungen:

**Die Anwendung muss folgende Anforderungen erfüllen.**

### 1. Verarbeitung der Daten

- 1.1. Die Anwendung muss Daten bezüglich der Benutzerangaben aus einer Datenbank auslesen und verarbeiten.
- 1.2. Als Benutzer möchte ich meine Personendaten ändern können.
- 1.3. Als Administrator möchte ich Benutzer einfügen, und Daten ändern.

### 2. Darstellung der Daten

- 2.1. Die Anwendung soll sich an das Corporate Design der Synergie anpassen.
- 2.2. Das Layout muss übersichtlich gestaltet sein.
- 2.3. Bei der Oberflächen-Gestaltung sollen Ergonomie-Richtlinien eingehalten werden.

### 3. Sonstige Anforderungen

- 3.1. Die Anwendung muss ohne das Installieren einer zusätzlichen Software über einen Webbrowser im Intranet erreichbar sein.
- 3.2. Die Anwendung muss im Firmennetz über einen regulären Webbrowser erreichbar sein.
- 3.3. Die Anwendung soll jederzeit erreichbar sein.
- 3.4. Die Anwendung muss in allen Webbrowser lauffähig sein

#### A3 Use-Case-Diagramm

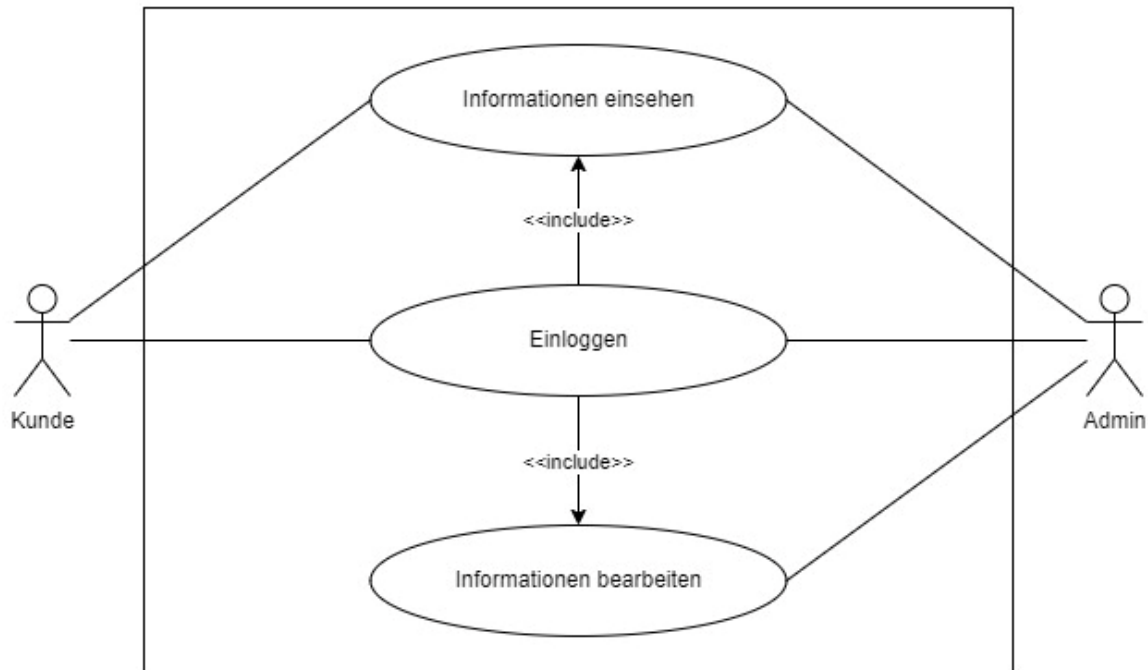


Abbildung 1: Use-Case-Diagramm

#### A4 Pflichtenheft (Auszug)

##### Zielbestimmung

###### Plattform

- Die Webanwendung muss in JavaScript (Node.js) programmiert werden.
- Die Webanwendung muss die Technologien des JavaScripts-Standards verwenden.
- Die Erzeugung der Datenbank-Tabellen soll durch phpMyAdmin erfolgen.
- Für das Gestalten der Oberfläche muss Handlebars.js genutzt werden.
- Die Daten der Webanwendung müssen in einer MySQL-Datenbank auf dem Webserver abgelegt sein.

###### Oberfläche

- Die Oberfläche muss mit HTML5 und CSS3 umgesetzt werden.
- Um dies zu generieren, wird die Handlebars.js-Technologie genutzt.

#### A5 Datenbankmodell

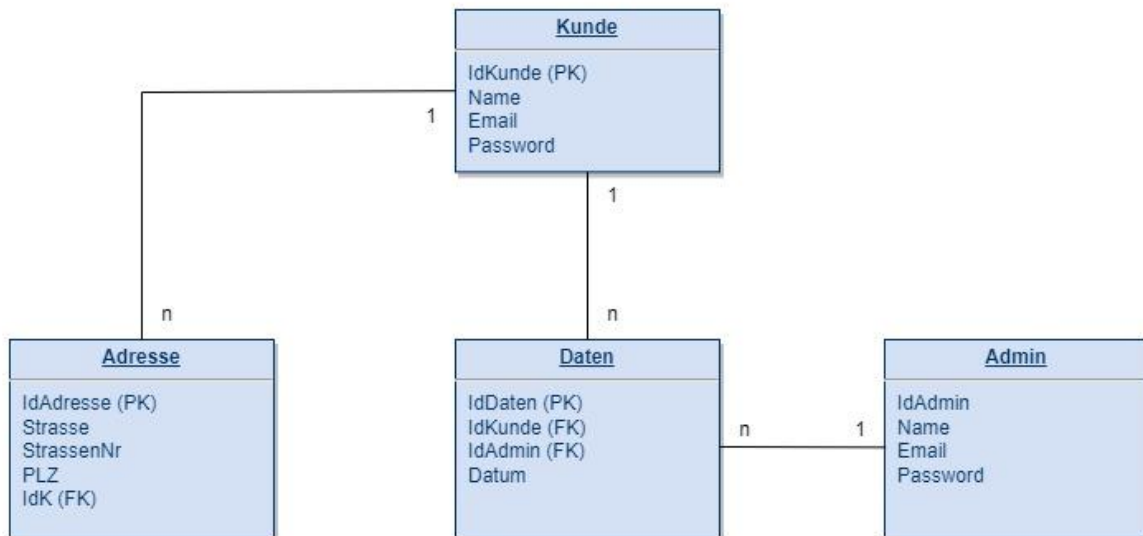


Abbildung 2: Entity-Relationship-Model

```
CREATE DATABASE dashboard;
CREATE TABLE `users`. (
  `id` INT(11) NOT NULL AUTO_INCREMENT ,
  `name` VARCHAR(100) NOT NULL ,
  `email` VARCHAR(100) NOT NULL ,
  `password` VARCHAR(255) NOT NULL ,
  PRIMARY KEY (`id`));
```

Abbildung 3: SQL



## A6 Struktogramm

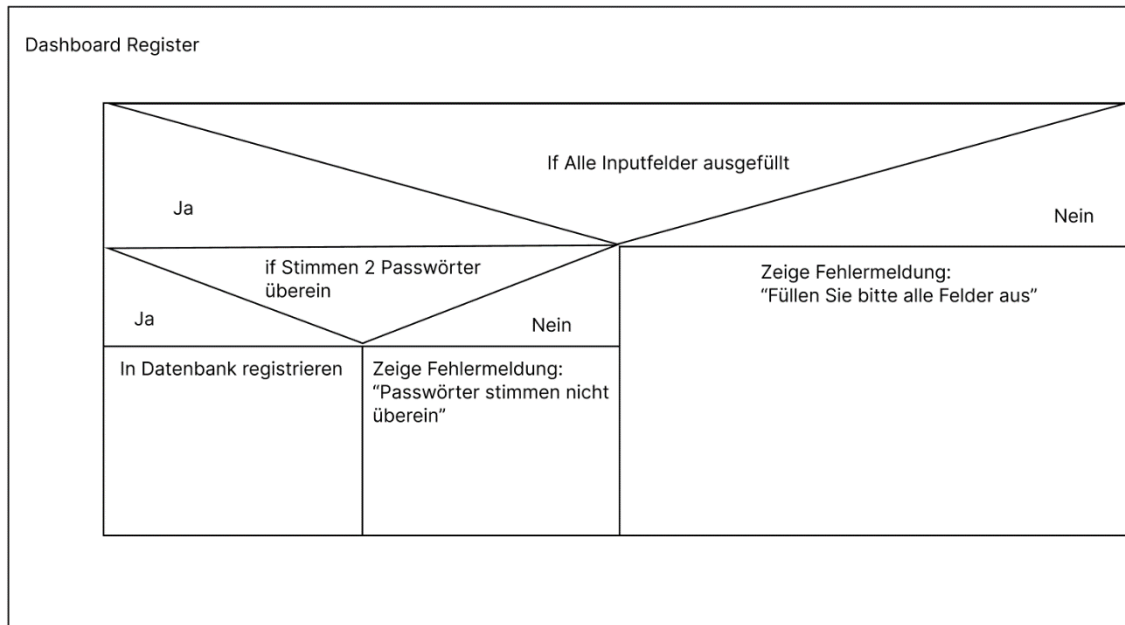


Abbildung 4: Struktogramm registrieren

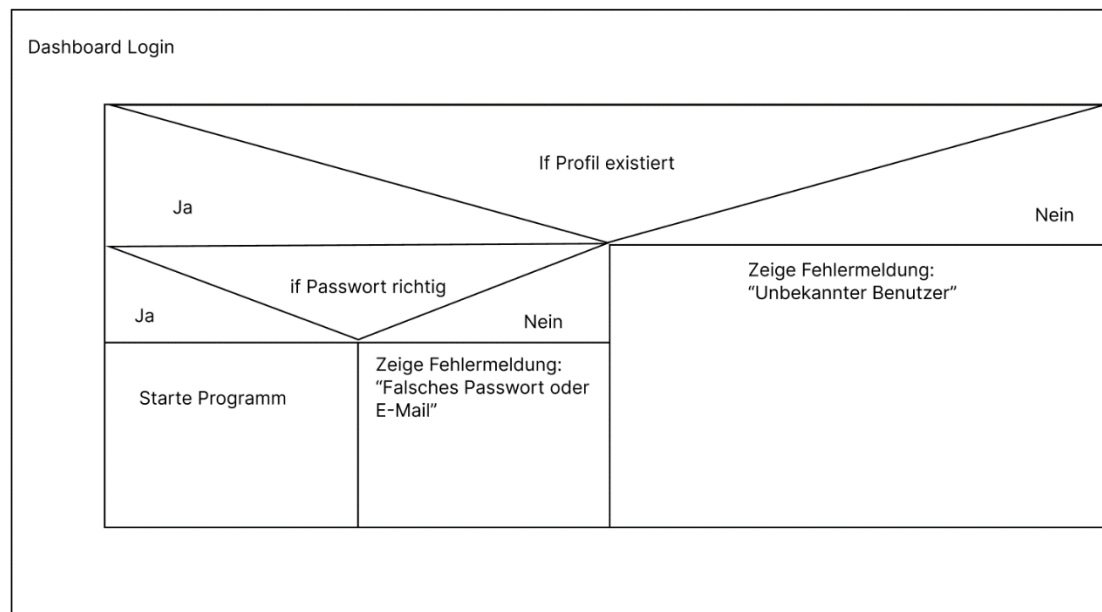


Abbildung 5: Struktogramm einloggen

## A7 Oberflächenentwürfe

The image displays two wireframe designs for a dashboard interface, specifically for registration and login. Both designs are contained within a light gray border.

**Top Wireframe (Registration):**

- Header:** On the left, it says "Logo" followed by "Dashboard-Synergie". On the right, it says "Start", "Login", and "Registrieren".
- Content Area:** Centered at the top is the text "Registrieren". Below it are four stacked, light gray rectangular input fields. The labels for these fields, from top to bottom, are "Name", "E-Mail", "Passwort", and "Passwort bestätigung".

**Bottom Wireframe (Login):**

- Header:** On the left, it says "Logo" followed by "Dashboard-Synergie". On the right, it says "Start", "Login", and "Registrieren".
- Content Area:** Centered at the top is the text "Einloggen". Below it are two stacked, light gray rectangular input fields. The labels for these fields, from top to bottom, are "E-Mail" and "Passwort".

Abbildung 6: Registrierung und Login

## A8 Screenshots der Anwendung

The screenshot shows the 'Synergie Dashboard' header with navigation links 'Start', 'Login', and 'Registrieren'. Below is a 'Register Form' with the following fields: 'Name', 'E-Mail Adresse', 'Passwort', and 'Passwort bestätigen'. There is a checkbox for 'Datenschutz akzeptiert' and a blue 'Registrieren' button.

Synergie Dashboard Start Login **Registrieren**

Register Form

Name

E-Mail Adresse

Passwort

Passwort bestätigen

☐ Datenschutz akzeptiert

Registrieren

Abbildung 5: Screenshot Registrierung

The screenshot shows the 'Synergie Dashboard' header with navigation links 'Start', 'Login', and 'Registrieren'. Below is a 'Login Form' with the following fields: 'E-Mail Adresse' (containing 'andreas82un@web.de') and 'Passwort' (masked with '.....'). There is a blue 'Einloggen' button.

Synergie Dashboard Start **Login** Registrieren

Login Form

E-Mail Adresse

andreas82un@web.de

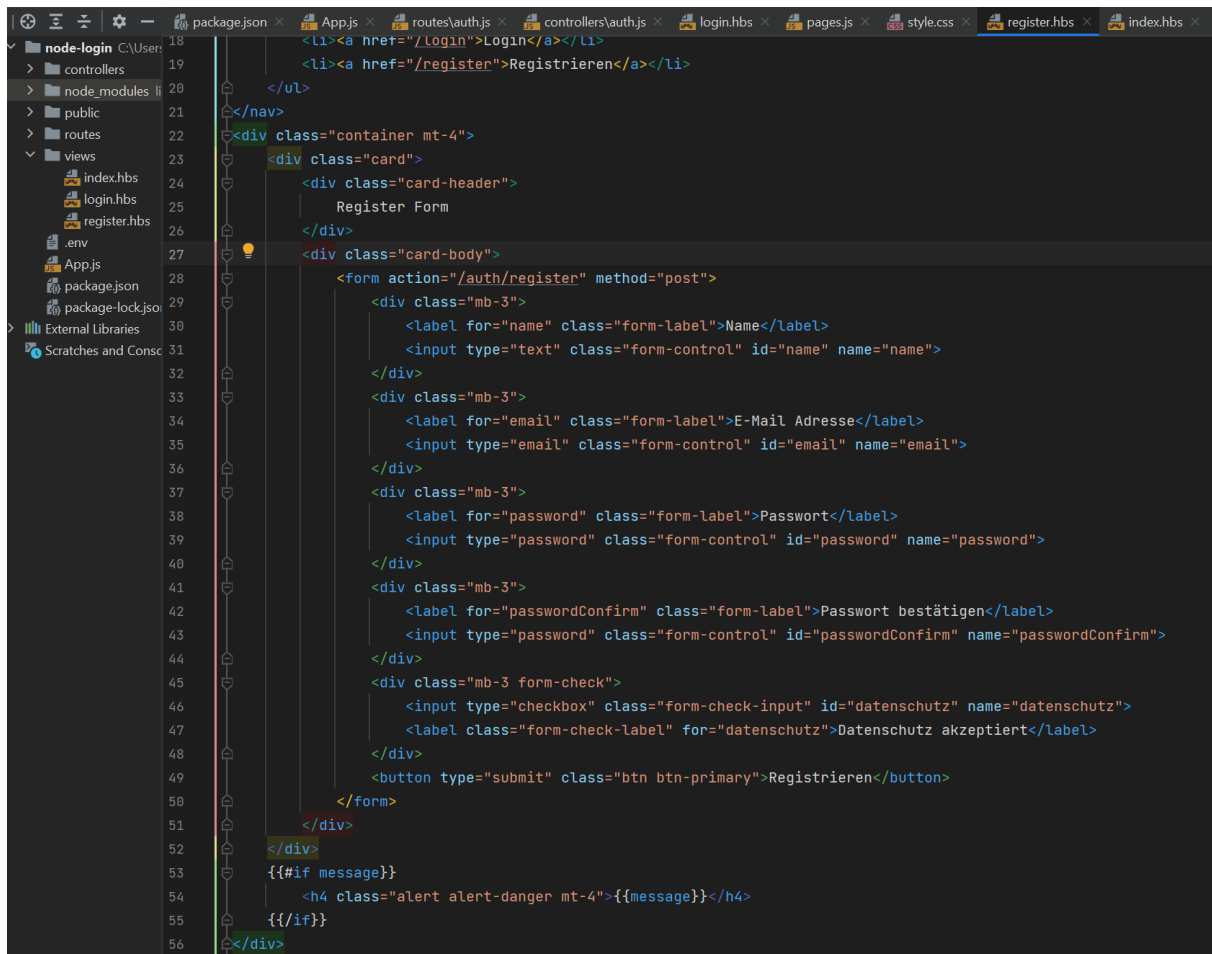
Passwort

.....

Einloggen

Abbildung 6: Screenshot Login

## A9 Entwicklerdokumentation (Oberfläche)



```
18 </li><a href="/login">Login</a></li>
19 </ul>
20 </nav>
21 <div class="container mt-4">
22 <div class="card">
23 <div class="card-header">
24 Register Form
25 </div>
26 <div class="card-body">
27 <form action="/auth/register" method="post">
28 <div class="mb-3">
29 <label for="name" class="form-label">Name</label>
30 <input type="text" class="form-control" id="name" name="name">
31 </div>
32 <div class="mb-3">
33 <label for="email" class="form-label">E-Mail Adresse</label>
34 <input type="email" class="form-control" id="email" name="email">
35 </div>
36 <div class="mb-3">
37 <label for="password" class="form-label">Passwort</label>
38 <input type="password" class="form-control" id="password" name="password">
39 </div>
40 <div class="mb-3">
41 <label for="passwordConfirm" class="form-label">Passwort bestätigen</label>
42 <input type="password" class="form-control" id="passwordConfirm" name="passwordConfirm">
43 </div>
44 <div class="mb-3 form-check">
45 <input type="checkbox" class="form-check-input" id="datenschutz" name="datenschutz">
46 <label class="form-check-label" for="datenschutz">Datenschutz akzeptiert</label>
47 </div>
48 <button type="submit" class="btn btn-primary">Registrieren</button>
49 </form>
50 </div>
51 </div>
52 </div>
53 {{#if message}}
54 <div class="alert alert-danger mt-4">{{message}}</div>
55 {{/if}}
56 </div>
```

Abbildung 7: HTML-Register

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

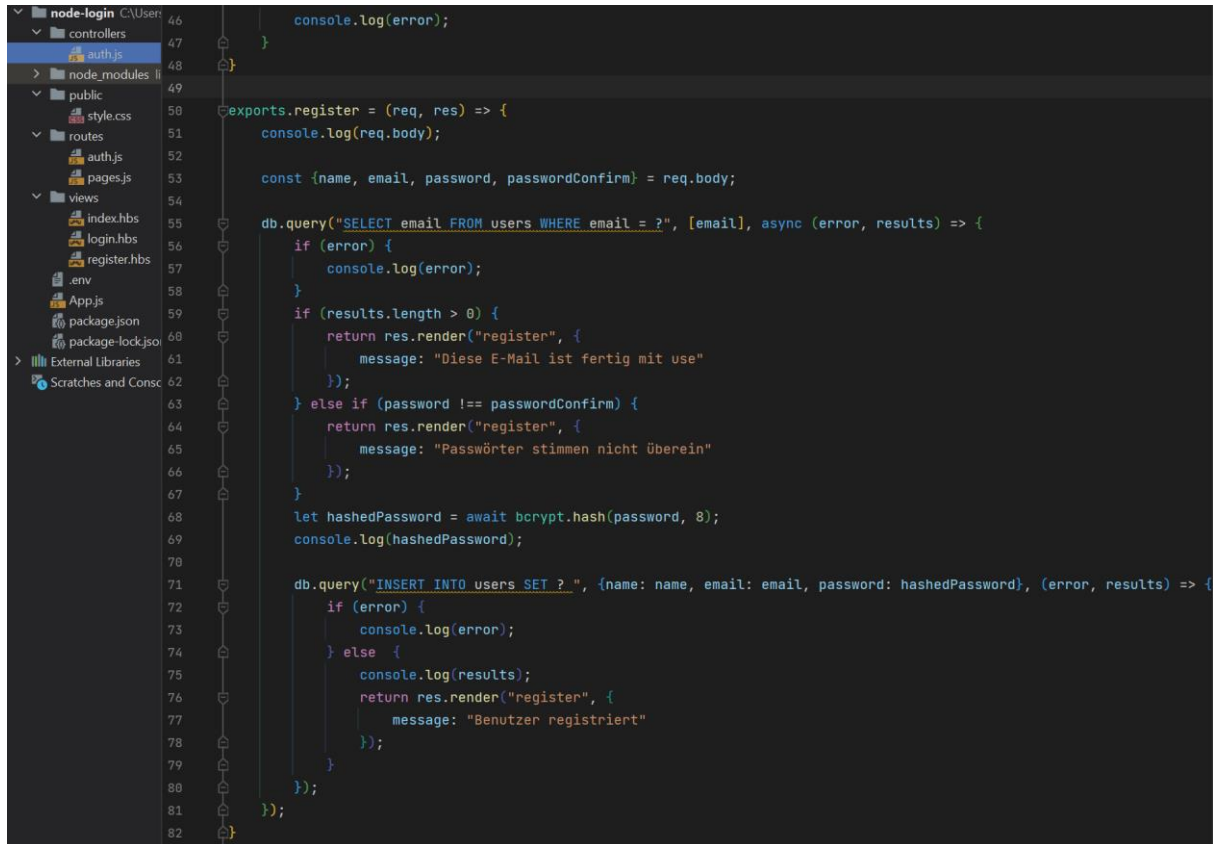
### Anhang

---

```
14 <nav>
15   <h4>Synergie Dashboard</h4>
16   <ul>
17     <li><a href="/">Start</a></li>
18     <li><a href="/login">Login</a></li>
19     <li><a href="/register">Registrieren</a></li>
20   </ul>
21 </nav>
22 <div class="container mt-4">
23   <div class="card">
24     <div class="card-header">
25       Login Form
26     </div>
27     <div class="card-body">
28       <form action="/auth/login" method="post">
29         <div class="mb-3">
30           <label for="email" class="form-label">E-Mail Adresse</label>
31           <input type="email" class="form-control" id="email" name="email">
32         </div>
33         <div class="mb-3">
34           <label for="password" class="form-label">Passwort</label>
35           <input type="password" class="form-control" id="password" name="password">
36         </div>
37         <button type="submit" class="btn btn-primary">Einloggen</button>
38       </form>
39     </div>
40   </div>
41   {{#if message}}
42   <h4 class="alert alert-danger mt-4">{{message}}</h4>
43   {{/if}}
44 </div>
```

Abbildung 8: HTML-Login

## A10 Entwicklerdokumentation



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure for 'node-login' with folders like 'controllers', 'public', 'routes', 'views', and files like 'auth.js', 'login.hbs', 'register.hbs', 'App.js', 'package.json', 'package-lock.json', and 'Scratches and Console'. The code editor shows the 'auth.js' file with the following code:

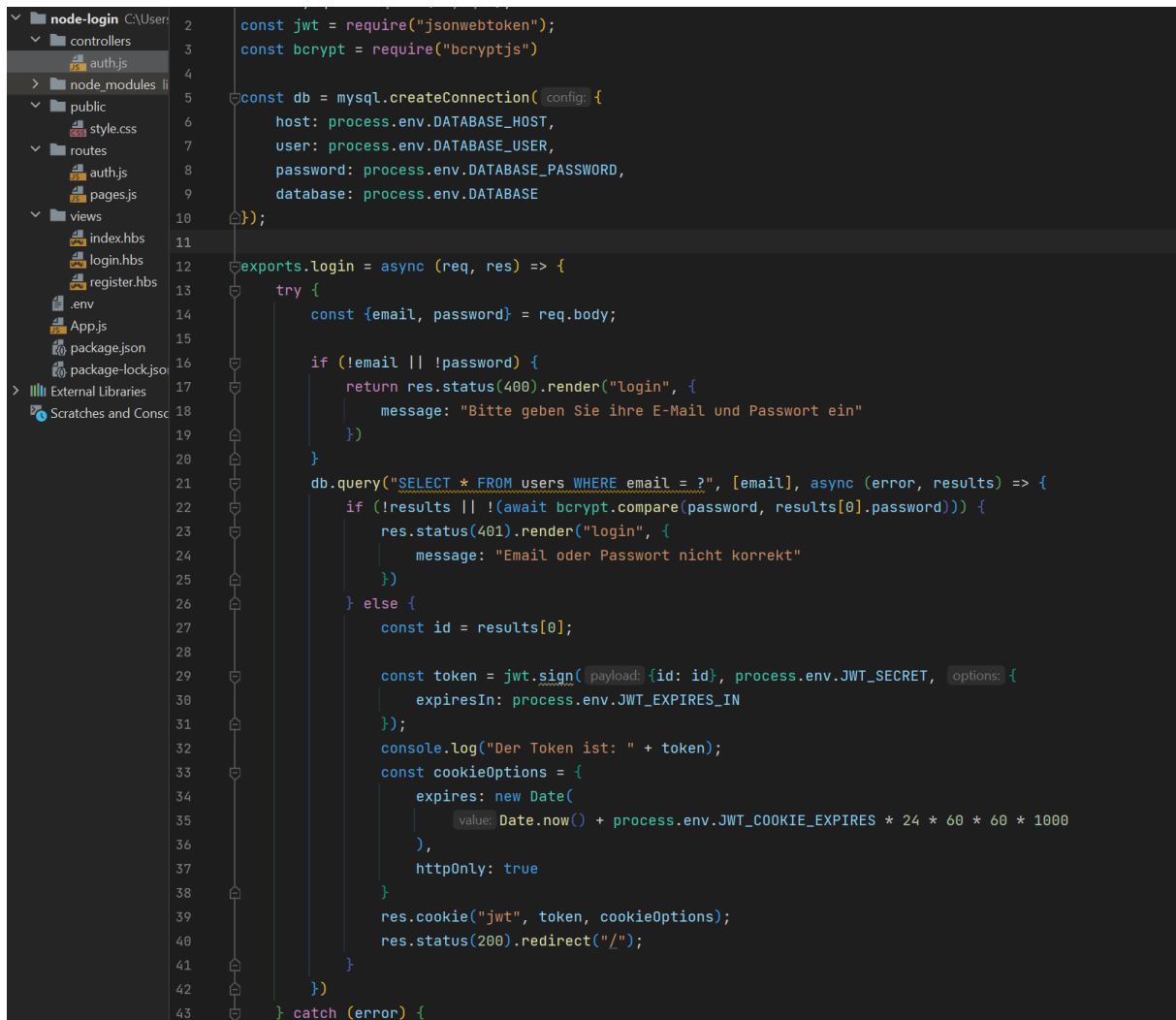
```
46 console.log(error);
47 }
48 }
49
50 exports.register = (req, res) => {
51   console.log(req.body);
52
53   const {name, email, password, passwordConfirm} = req.body;
54
55   db.query("SELECT email FROM users WHERE email = ?", [email], async (error, results) => {
56     if (error) {
57       console.log(error);
58     }
59     if (results.length > 0) {
60       return res.render("register", {
61         message: "Diese E-Mail ist fertig mit use"
62       });
63     } else if (password !== passwordConfirm) {
64       return res.render("register", {
65         message: "Passwörter stimmen nicht überein"
66       });
67     }
68     let hashedPassword = await bcrypt.hash(password, 8);
69     console.log(hashedPassword);
70
71     db.query("INSERT INTO users SET ? ", {name: name, email: email, password: hashedPassword}, (error, results) => {
72       if (error) {
73         console.log(error);
74       } else {
75         console.log(results);
76         return res.render("register", {
77           message: "Benutzer registriert"
78         });
79       }
80     });
81   });
82 }
```

Abbildung 9: Registrieren

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### Anhang



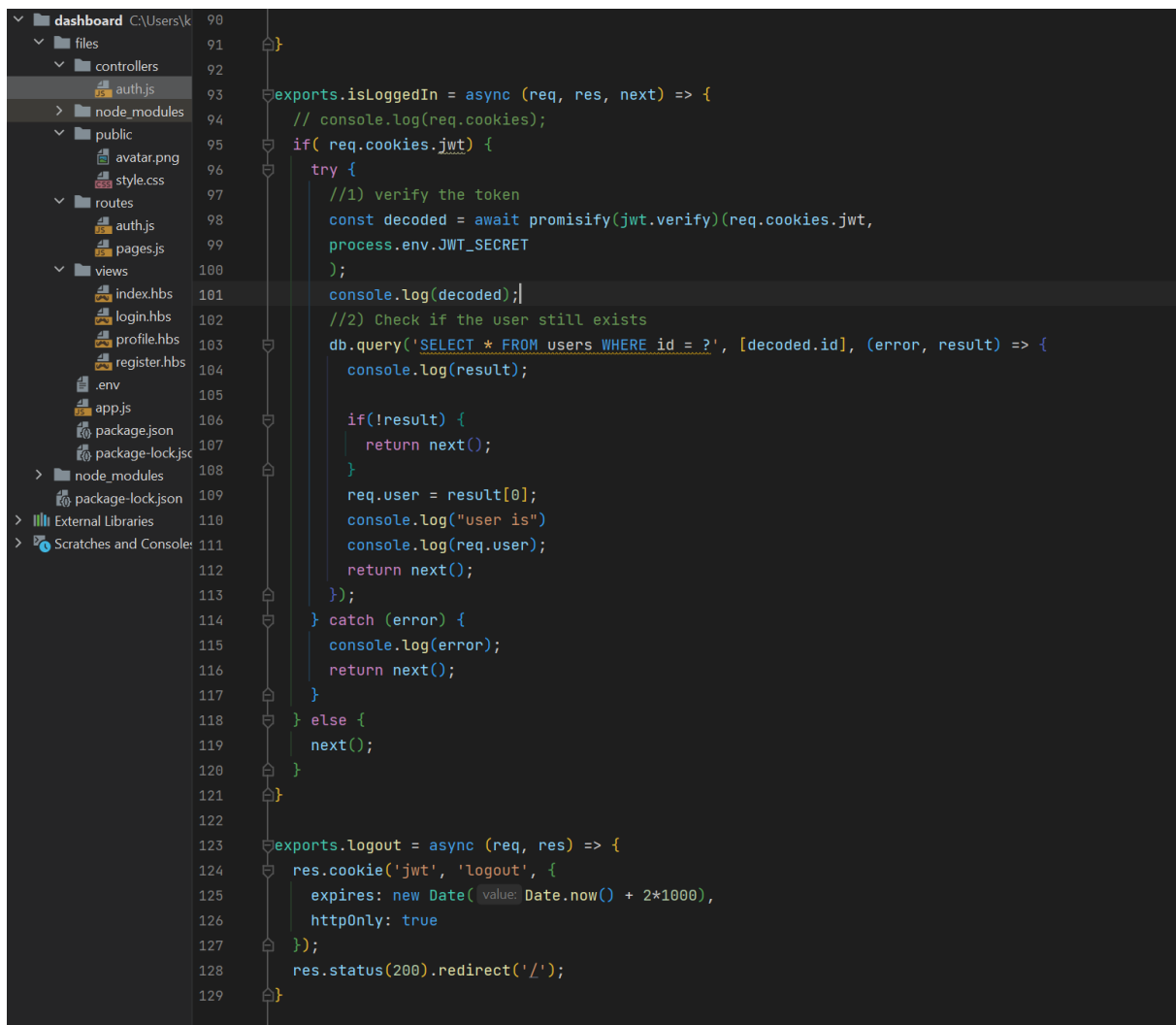
```
2  const jwt = require("jsonwebtoken");
3  const bcrypt = require("bcryptjs")
4
5  const db = mysql.createConnection( config: {
6    host: process.env.DATABASE_HOST,
7    user: process.env.DATABASE_USER,
8    password: process.env.DATABASE_PASSWORD,
9    database: process.env.DATABASE
10  });
11
12  exports.login = async (req, res) => {
13    try {
14      const {email, password} = req.body;
15
16      if (!email || !password) {
17        return res.status(400).render("login", {
18          message: "Bitte geben Sie ihre E-Mail und Passwort ein"
19        });
20      }
21      db.query("SELECT * FROM users WHERE email = ?", [email], async (error, results) => {
22        if (!results || !(await bcrypt.compare(password, results[0].password))) {
23          res.status(401).render("login", {
24            message: "Email oder Passwort nicht korrekt"
25          });
26        } else {
27          const id = results[0];
28
29          const token = jwt.sign( payload: {id: id}, process.env.JWT_SECRET, options: {
30            expiresIn: process.env.JWT_EXPIRES_IN
31          });
32          console.log("Der Token ist: " + token);
33          const cookieOptions = {
34            expires: new Date(
35              value: Date.now() + process.env.JWT_COOKIE_EXPIRES * 24 * 60 * 60 * 1000
36            ),
37            httpOnly: true
38          }
39          res.cookie("jwt", token, cookieOptions);
40          res.status(200).redirect("/");
41        }
42      });
43    } catch (error) {
```

Abbildung 10: Login

## DASHBOARD ENTWICKLUNG

Entwicklung eines Dashboards – registrieren, einloggen, ausloggen und Benutzerbereich

### Anhang



```
90 }
91
92
93 exports.isLoggedIn = async (req, res, next) => {
94   // console.log(req.cookies);
95   if( req.cookies.jwt) {
96     try {
97       //1) verify the token
98       const decoded = await promisify(jwt.verify)(req.cookies.jwt,
99         process.env.JWT_SECRET
100       );
101       console.log(decoded);
102       //2) Check if the user still exists
103       db.query('SELECT * FROM users WHERE id = ?', [decoded.id], (error, result) => {
104         console.log(result);
105
106         if(!result) {
107           return next();
108         }
109         req.user = result[0];
110         console.log("user is")
111         console.log(req.user);
112         return next();
113       });
114     } catch (error) {
115       console.log(error);
116       return next();
117     }
118   } else {
119     next();
120   }
121 }
122
123 exports.logout = async (req, res) => {
124   res.cookie('jwt', 'logout', {
125     expires: new Date( value: Date.now() + 2*1000),
126     httpOnly: true
127   });
128   res.status(200).redirect('/');
129 }
```

Abbildung 11: Eingeloggt - Kontrolle