Prüfungsphase:	**Winter 2019/20**

Name/Vorname:	Bekefi, Barbara Anna
Ausbildungsberuf:	Fachinformatikerin-Anwendungsentwicklung
Prüflingsnummer:	**(183) 05381**
Id.Nr.:	**1496126**

Erstellen einer Webseite in WordPress, Programmierung der gewünschten Funktionen in PHP und der dazugehörigen Datenbank in MySQL

Projektgeber:

T-NET SOLUTIONS
Bahnstraße 11
16348 Wandlitz

Ansprechpartnerin:

Dagmar Schumacher
Telefon: +49 33397 23 99 99
E-Mail: info@it-net.expert

Ausbildungsbetrieb:

BFW Berlin/Brandenburg
Kastanienallee 25
16567 Mühlenbeck

Ansprechpartner:

Herr Alwin Dettling
Telefon: +49-033056 86-316
E-Mail: alwin.dettling@bfw-brandenburg.de

Eidesstattliche Erklärung:

Hiermit erkläre ich, Barbara Bekefi, die hier vorliegende Arbeit (Dokumentation) selbstständig und nur unter Zuhilfenahme der aufgeführten Quellen angefertigt zu haben. Der aufgeführte zeitliche Rahmen wurde eingehalten.

Ort/Datum

Unterschrift

IHK Potsdam

Winterprüfung 2019

Ausbildungsberuf
Fachinformatiker/-in Anwendungsentwicklung

Prüfungsbezirk
Potsdam FI 1196-1 (AP T2V1)

Frau Barbara Bekefi
Identnummer: 1496126
E-Mail: barbara.bekefi@gmail.com, Telefon: +49 1575 6697814

Ausbildungsbetrieb: bfw
Projektbetreuer: Herr Alvin Dettling
E-Mail: alwin.dettling@bfw-berlin-brandenburg.de, Telefon: +49 33 056 86-316

Thema der Projektarbeit
Erstellen einer Webseite in Wordpress, Programmierung der gewünschten Funktionen in PHP und der dazugehörigen Datenbank in MySql

IHK Potsdam
Frau Barbara Bekefi
Identnummer: 1496126
17.10.2019

1 Thema der Projektarbeit

Erstellen einer Webseite in Wordpress, Programmierung der gewünschten Funktionen in PHP und der dazugehörigen Datenbank in MySql

2 Geplanter Bearbeitungszeitraum

Beginn: 16.09.2019
Ende: 10.10.2019

3 Projektbeschreibung

IST-Zustand:

Der Projektgeber betreibt zurzeit eine Webseite, it-net.solutions, die in mehreren Bereichen fehlerhaft ist.
Unter anderem zählt dazu, dass diese Seite weder für die Verwendung auf dem Handy, als auch dem Tablet optimiert wurde. Viele Mängel wurden auch bei der Codierung der Webseite eingebaut, z.B. gibt es redundante Menüpunkte, Inkonsistenzen der Menüführung und Schriftgrößen, oder Buttons, die nicht anwählbar sind. Die Chatfunktion ist ebenso wenig optimiert, wie auch gestalterische und ästhetische Verwendung von Textelementen. Bausteine der Seite sind nicht konsistent umgesetzt und Verlinkungen sind fehlerhaft.
Die Alarm-Webseite hat ähnliche Probleme. Sie soll so gestaltet werden, dass die Kunden über alle Produkte und Dienstleistungen der Firma informiert werden (darauf wird im folgenden Soll-Zustand ausführlich eingegangen).
Die existierende it-net.solutions Seite soll dann den Gestaltungsregeln der Alarm-Webseite angepasst werden, was aber nicht Inhalt des Projektes ist.

SOLL-Zustand:

Ziel des Projektes ist es, mit dem Content-Management-System Wordpress eine komplette Firmenwebseite zu erstellen. Dafür werde ich sowohl eine neu zu erstellende Homepage (Alarm-Webseite) als auch eine Startseite entwickeln. Dadurch kann der Besucher bzw. Kunde wählen, ob er zur Alarm-Solutions Seite - was mein Kern-Projekt beinhaltet - gehen oder zum Dienstleistungsbereich der It-net.solutions Seite (im Ist-Zustand betrachtet) wechseln will.
Der Projektschwerpunkt ist die Entwicklung der Alarm-Webseite.
Die Nutzung beider Webseiten soll sowohl für das Smartphone, als auch für die Desktopanwendung optimiert werden, um die Kosten für die Webseitenunterhaltung zu verringern. Dazu plane ich, kostenfreie Wordpress Templates und Plugins heranzuziehen.

Zusätzlich soll während der Umsetzung des Projekts ein kennwort-geschützter Bereich erstellt werden; für diesen wird mit PHP eine Login-Funktion programmiert. Der Projektgeber beabsichtigt, die Daten der Webseite und die Kundendaten auf dem eigenen sicheren Cloud-Server zu speichern. Dafür werde ich eine entsprechenden MySQL Datenbank mit

IHK Potsdam
Frau Barbara Bekefi
Identnummer: 1496126
17.10.2019

Dummy-Kundendaten erstellen. Dies plane ich mit phpMyAdmin zu realisieren. Diese Dummy-Kundendaten werden nach dem Entwicklungsprozess durch reale Kundendaten ersetzt. Die Kennwörter in der Datenbank werden durch der PHP-native Password-Hash Funktion verschlüsselt.
Der Plan des Projektgebers ist, den Kunden eine Datenbank von Gebrauchsanweisungen zur Verfügung zu stellen. Diese werde ich mithilfe der Wordpress-nativen Datenbank entwickeln (Custom Post Type oder CPT). Es gibt Plugins in Wordpress für das Uploaden von verschiedenen Dateien zu einen Custom Post Type; diese Plugins können pdf-Formate verwenden, jedoch nur in eingeschränkten Dateigrößen. Daher muss für diesen Zweck ein eigenes %E2%80%9APlugin‘ von mir programmiert werden.
Die Gebrauchsanweisungen sollen von Kunden suchbar sein; deswegen plane ich, für diesen Zweck zwei Drop-Down Boxen zu programmieren, um die Suche nach verschiedenen Taxonomien (Hersteller und Produkttyp) zu ermöglichen.
Weiterhin werde ich ein Child-Theme erstellen, um meine selbst programmierte Funktionen bei einem Update des Wordpress-Themes zu schützen.
Einige Menü-Funktionen sollen nur für ein- bzw. für ausgeloggte Kunden sichtbar sein. Dies werde ich durch Programmierung von Sessions und einem Plugin (Menu Items Visibility) lösen.
Die Gestaltung der gesamten Webpräsenz erfolgt nach den Wünschen und Vorstellungen des Projektgebers.

Projektziele/Aufgaben

Folgende Aufgaben müssen durchgeführt werden, um den gewünschten Anforderungen des Soll-Zustandes zu erreichen.

- Analyse der existierenden Homepage und Datenbank
- Installation von XAMPP
- Erstellung einer Datenbank mit zugehöriger Tabelle
- Aufbau einer Testumgebung mit dem neuesten Wordpress CMS-System
- Import der derzeitigen Webseiten-Daten auf die lokale Testumgebung
- Analyse der Kundendaten
- Entwicklung eines Login-Bereichs mit einem PHP nativen Verschlüsselungsalgorithmus
- Programmierung von Custom Post Type (CPT) und dazugehörigen Plugin für das Hochladen, Speicherung, und Herunterladen von Gebrauchsanweisungen in PDF
- Programmierung von Suchtaxonomien
- Programmierung von Dropdown-Box Logik für Suchtaxonomien
- Sessions programmieren mit Menüprüfung
- Erstellung des Child-Themes
- Migration von Localhost zu Live-Umgebung

Personalplanung

Die Bearbeitung des Projekts werden von mir in Eigenarbeit durchgeführt. Weitere beteiligte Personen stehen außerhalb des Projekts und werden bei den Schnittstellen aufgeführt.

Sachmittelplanung

- PC mit WLAN-Verbindung
- Notepad++ Editor
- Xampp für Speichern der Webseite auf Localhost
- Microsoft Office (Power Point für Präsentation, Outlook für Emails, Word und Excel für Dokumentation)
- Microsoft Photo Editor/Photoshop für Zuschneiden von Logos

Schnittstellen

Technische Schnittstellen

- Zugriff zum Wordpress-Konto des Praktikumbetriebs
- Kommunikation: Kerio Mailserver. (Email, chatten). Microsoft Office. Outlook für Kommunikation und Terminkalender.
- Synology Cloud Station Drive der Firma (zur Synchronisierung mit lokalem Rechner)
- Securepoint SSL VPN Netzwerk
- PC visit Remote-Host Fernwartungstool
- Zugriff zum Google-Konto des Projektgebers (z.B. für Google Maps, Recaptcha)

Personelle Schnittstellen

- Frau Dagmar Schumacher – Projektleiterin
- Herr Jörn Schmidt – Leiter des Betriebs, bei einigen Fragen z.B. Produktkategorien, Zugangsdaten

4 Projektumfeld

Der Praktikumsbetrieb it-solutions ist ein Dienstleister mit zwei Tätigkeitsfeldern: Objektsicherheit bzw. IT- und Netzwerklösungen. Dementsprechend möchte die Firma zwei Webseiten haben: Alarm-Solutions und IT-net.solutions.
Der Betrieb mit Sitz in Wandlitz beschäftigt drei Angestellte und einige freie Mitarbeiter.
Die Dienstleistungen richten sich an Privatpersonen (inkl. Wohngemeinschaften) und Kleinunternehmen. Der Betrieb bietet Komplettlösungen an, von Planung und Installation bis zu Wartung. Diese Lösungen werden durch Produkte von verschiedenen Kooperationspartnern realisiert.
Einige Kooperationspartner, z.B. Elbe Haus, bauen die vom Betrieb geplante komplette Sicherheitslösung für einen Pauschalpreis ein, und je nach Bedarf des Kunden übernimmt der Betrieb die benötigte Instandhaltung, Reparatur- und Wartungsaufgaben.
Andere Kooperationspartner sind Entwickler, wie z.B. Jovision, ein deutscher Hersteller von Kameras und Videoüberwachungssystemen, oder der größte Kooperationspartner Jablotron, eine internationale Firma mit umfassender Produktpalette rund um Sicherheit.
Das Portfolio umfasst Produkte und Dienstleistungen im Bereich Sicherheit. U.a. Bewegungsmelder, Tür- und Fensterkontakte, Videoüberwachung, Vernebelung, CO – und Rauchmelder, Fernbedienung, Videoüberwachung, Sirenen, usw.

5 Projektphasen mit Zeitplanung

Vorläufiger Zeitplan für das Projekt:
1.1 Planung
1.1.1 Initiale Projektbesprechung und Zielsetzung (1 Stunden)
1.1.2 Analyse des Ist-Zustandes/Ausgangssituation (1 Stunde)
1.1.3 Erstellung des Soll-Konzepts (1 Stunde)
1.1.4 Ableitung der Projektzieles, -Ablaufes, -Aufgaben (1 Stunde)
1.2 Durchführung
1.2.1 Wordpress-Update auf neueste Version (1 Stunde)
1.2.2 Strukturelle Erstellung der Test-Webseite (5 Stunden)
1.2.3 Umsetzung von Änderungswünschen (2 Stunden)
1.2.4 Erstellung von Test-Datenbank (Stammkunden) (1 Stunde)
1.2.5 Programmierung von Login und Password-Hash Verschlüsselung (8 Stunden)
1.2.6 Programmierung von Sessions und Sichtbarkeit der Menü-Elemente anpassen (6 Stunden)
1.2.7 Einbetten von Google-Funktionen, Maps und Recapcha (1 Stunde)
1.2.8 Programmierung von Custom Post Type (CPT) (4 Stunden)
1.2.9 Programmierung von Custom Anhang (.pdf) zum Custom Post Type, und Anhang lösbar machen (6 Stunden)
1.2.10 Programmierung von Suchtaxonomien für CPT (8 Stunden)
1.2.11 Programmierung von Dropdown-Box Logik für Suchtaxonomien (8 Stunden)
1.2.12 Layout Gestaltung von selbst programmierten Funktionen (CSS) (1 Stunde)
1.2.13 Testen und Fehlerbehebung (1 Stunden)
1.2.14 Anpassungen (1 Stunde)
1.2.15 Integration/Migration (1 Stunde)
1.2.16 Qualitäts-Sicherung/Testen (0,5 Stunden)
1.3 Übergabe/Auswertung/Dokumentation (1 Stunde)
1.3.1 Erstellung des Soll-Ist-Vergleiches (1 Stunden)
1.3.2 Abnahme/Übergabe (0,5 Stunden)
1.3.3 Erstellung des Ausblicks/Fazit (1 Stunde)
1.3.4 Erstellung der Dokumentation (8 Stunden)
Summe: 70 Stunden

6 Dokumentation zur Projektarbeit

Aus den Projektnotizen wird am Ende eine Dokumentation erstellt. Änderungen zum Projektantrag werden in der Dokumentation benannt und begründet.

7 Anlagen

siehe Anlage 1

IHK Potsdam

Frau Barbara Bekefi
Identnummer: 1496126 17.10.2019

8 Präsentationsmittel

- Notebook
- Power Point
- Beamer
- Flipchart

9 Hinweis!

Ich bestätige, dass der Projektantrag dem Ausbildungsbetrieb vorgelegt und vom Ausbildenden genehmigt wurde. Der Projektantrag enthält keine Betriebsgeheimnisse. Soweit diese für die Antragstellung notwendig sind, wurden nach Rücksprache mit dem Ausbildenden die entsprechenden Stellen unkenntlich gemacht.

Mit dem Absenden des Projektantrages bestätige ich weiterhin, dass der Antrag eigenständig von mir angefertigt wurde. Ferner sichere ich zu, dass im Projektantrag personenbezogene Daten (d. h. Daten über die eine Person identifizierbar oder bestimmbar ist) nur verwendet werden, wenn die betroffene Person hierin eingewilligt hat.

Bei meiner ersten Anmeldung im Online-Portal wurde ich darauf hingewiesen, dass meine Arbeit bei Täuschungshandlungen bzw. Ordnungsverstößen mit „null“ Punkten bewertet werden kann. Ich bin weiter darüber aufgeklärt worden, dass dies auch dann gilt, wenn festgestellt wird, dass meine Arbeit im Ganzen oder zu Teilen mit der eines anderen Prüfungsteilnehmers übereinstimmt. Es ist mir bewusst, dass Kontrollen durchgeführt werden.

IHK Potsdam

Frau Barbara Bekefi
Identnummer: 1496126 17.10.2019

Anlage 1

1. Persönliche Daten

- Ausbildungsberuf: FAAN
- Name: Barbara Bekefi
- Ident-Nummer: 1496126
- Prüflingsnummer:
- Ausbildungsbetrieb: BFW
- Projektgeber: It-Net.solutions

2. Thema/Titel der Projektarbeit

Erstellen einer Webseite in Wordpress, Programmierung der gewünschten Funktionen in PHP und der dazugehörigen Datenbank in MySql

3. Geplanter Bearbeitungszeitraum

Beginn: 16.09.2019

Ende: 10.10.2019

4. Projektbeschreibung

4.1. IST-Zustand:

Der Projektgeber betreibt zurzeit eine Webseite, it-net.solutions, die in mehreren Bereichen fehlerhaft ist.

Unter anderem zählt dazu, dass diese Seite weder für die Verwendung auf dem Handy, als auch dem Tablet optimiert wurde. Viele Mängel wurden auch bei der Codierung der Webseite eingebaut, z.B. gibt es redundante Menüpunkte, Inkonsistenzen der Menüführung und Schriftgrößen, oder Buttons, die nicht anwählbar sind. Die Chatfunktion ist ebenso wenig optimiert, wie auch gestalterische und ästhetische Verwendung von Textelementen. Bausteine der Seite sind nicht konsistent umgesetzt und Verlinkungen sind fehlerhaft.

Die Alarm-Webseite hat ähnliche Probleme. Sie soll so gestaltet werden, dass die Kunden über alle Produkte und Dienstleistungen der Firma informiert werden (darauf wird im folgenden Soll-Zustand ausführlich eingegangen).

Die existierende it-net.solutions Seite soll dann den Gestaltungsregeln der Alarm-Webseite angepasst werden, was aber nicht Inhalt des Projektes ist.

4.2. SOLL-Zustand:

Ziel des Projektes ist es, mit dem Content-Management-System Wordpress eine komplette Firmenwebseite zu erstellen. Dafür werde ich sowohl eine neu zu erstellende Homepage (Alarm-Webseite) als auch eine Startseite entwickeln. Dadurch kann der Besucher bzw. Kunde wählen, ob er zur Alarm-Solutions Seite - was mein Kern-Projekt beinhaltet - gehen oder zum Dienstleistungsbereich der It-net.solutions Seite (im Ist-Zustand betrachtet) wechseln will.

Der Projektschwerpunkt ist die Entwicklung der Alarm-Webseite.

Die Nutzung beider Webseiten soll sowohl für das Smartphone, als auch für die Desktopanwendung optimiert werden, um die Kosten für die Webseitenunterhaltung zu verringern. Dazu plane ich, kostenfreie Wordpress Templates und Plugins heranzuziehen.

Zusätzlich soll während der Umsetzung des Projekts ein kennwort-geschützter Bereich erstellt werden; für diesen wird mit PHP eine Login-Funktion programmiert. Der Projektgeber beabsichtigt, die Daten der Webseite und die Kundendaten auf dem eigenen sicheren Cloud-Server zu speichern. Dafür werde ich eine entsprechenden MySQL Datenbank mit Dummy-Kundendaten erstellen. Dies plane ich mit phpMyAdmin zu realisieren. Diese Dummy-Kundendaten werden nach dem Entwicklungsprozess durch reale Kundendaten ersetzt.

Die Kennwörter in der Datenbank werden durch der PHP-native Password-Hash Funktion verschlüsselt.

Der Plan des Projektgebers ist, den Kunden eine Datenbank von Gebrauchsanweisungen zur Verfügung zu stellen. Diese werde ich mithilfe der Wordpress-nativen Datenbank entwickeln (Custom Post Type oder CPT). Es gibt Plugins in Wordpress für das Uploaden von verschiedenen Dateien zu einen Custom Post Type; diese Plugins können pdf-Formate verwenden, jedoch nur in eingeschränkten Dateigrößen. Daher muss für diesen Zweck ein eigenes ‚Plugin' von mir programmiert werden.

Die Gebrauchsanweisungen sollen von Kunden suchbar sein; deswegen plane ich, für diesen Zweck zwei Drop-Down Boxen zu programmieren, um die Suche nach verschiedenen Taxonomien (Hersteller und Produkttyp) zu ermöglichen.

Weiterhin werde ich ein Child-Theme erstellen, um meine selbst programmierte Funktionen bei einem Update des Wordpress-Themes zu schützen.

Einige Menü-Funktionen sollen nur für ein- bzw. für ausgeloggte Kunden sichtbar sein. Dies werde ich durch Programmierung von Sessions und einem Plugin (Menu Items Visibility) lösen.

Die Gestaltung der gesamten Webpräsenz erfolgt nach den Wünschen und Vorstellungen des Projektgebers.

4.3. Projektziele/Aufgaben

Folgende Aufgaben müssen durchgeführt werden, um den gewünschten Anforderungen des Soll-Zustandes zu erreichen.

- Analyse der existierenden Homepage und Datenbank
- Installation von XAMPP
- Erstellung einer Datenbank mit zugehöriger Tabelle
- Aufbau einer Testumgebung mit dem neuesten Wordpress CMS-System
- Import der derzeitigen Webseiten-Daten auf die lokale Testumgebung
- Analyse der Kundendaten
- Entwicklung eines Login-Bereichs mit einem PHP nativen Verschlüsselungsalgorithmus
- Programmierung von Custom Post Type (CPT) und dazugehörigen Plugin für das Hochladen, Speicherung, und Herunterladen von Gebrauchsanweisungen in PDF
- Programmierung von Suchtaxonomien
- Programmierung von Dropdown-Box Logik für Suchtaxonomien
- Sessions programmieren mit Menüprüfung
- Erstellung des Child-Themes
- Migration von Localhost zu Live-Umgebung

4.4. Personalplanung

Die Bearbeitung des Projekts werden von mir in Eigenarbeit durchgeführt. Weitere beteiligte Personen stehen außerhalb des Projekts und werden bei den Schnittstellen aufgeführt.

4.5. Sachmittelplanung

- PC mit WLAN-Verbindung
- Notepad++ Editor
- Xampp für Speichern der Webseite auf Localhost
- Microsoft Office (Power Point für Präsentation, Outlook für Emails, Word und Excel für Dokumentation)
- Microsoft Photo Editor/Photoshop für Zuschneiden von Logos

4.6. Schnittstellen

4.6.1. Technische Schnittstellen

- Zugriff zum Wordpress-Konto des Praktikumbetriebs
- Kommunikation: Kerio Mailserver. (Email, chatten). Microsoft Office. Outlook für Kommunikation und Terminkalender.
- Synology Cloud Station Drive der Firma (zur Synchronisierung mit lokalem Rechner)
- Securepoint SSL VPN Netzwerk
- PC visit Remote-Host Fernwartungstool
- Zugriff zum Google-Konto des Projektgebers (z.B. für Google Maps, Recaptcha)

4.6.2. Personelle Schnittstellen

- Frau Dagmar Schumacher – Projektleiterin
- Herr Jörn Schmidt – Leiter des Betriebs, bei einigen Fragen z.B. Produktkategorien, Zugangsdaten

5. Projektumfeld

Der Praktikumsbetrieb it-solutions ist ein Dienstleister mit zwei Tätigkeitsfeldern: Objektsicherheit bzw. IT- und Netzwerklösungen. Dementsprechend möchte die Firma zwei Webseiten haben: Alarm-Solutions und IT-net.solutions.

Der Betrieb mit Sitz in Wandlitz beschäftigt drei Angestellte und einige freie Mitarbeiter.

Die Dienstleistungen richten sich an Privatpersonen (inkl. Wohngemeinschaften) und Kleinunternehmen. Der Betrieb bietet Komplettlösungen an, von Planung und Installation bis zu Wartung. Diese Lösungen werden durch Produkte von verschiedenen Kooperationspartnern realisiert.

Einige Kooperationspartner, z.B. Elbe Haus, bauen die vom Betrieb geplante komplette Sicherheitslösung für einen Pauschalpreis ein, und je nach Bedarf des Kunden übernimmt der Betrieb die benötigte Instandhaltung, Reparatur- und Wartungsaufgaben.

Andere Kooperationspartner sind Entwickler, wie z.B. Jovision, ein deutscher Hersteller von Kameras und Videoüberwachungssystemen, oder der größte Kooperationspartner Jablotron, eine internationale Firma mit umfassender Produktpalette rund um Sicherheit.

Das Portfolio umfasst Produkte und Dienstleistungen im Bereich Sicherheit. U.a. Bewegungsmelder, Tür- und Fensterkontakte, Videoüberwachung, Vernebelung, CO – und Rauchmelder, Fernbedienung, Videoüberwachung, Sirenen, usw.

6. Projektphasen mit Zeitplanung

Vorläufiger Zeitplan für das Projekt:

1.1 Planung
1.1.1 Initiale Projektbesprechung und Zielsetzung (1 Stunden)
1.1.2 Analyse des Ist-Zustandes/Ausgangssituation (1 Stunde)
1.1.3 Erstellung des Soll-Konzepts (1 Stunde)
1.1.4 Ableitung der Projektzieles, -Ablaufes, -Aufgaben (1 Stunde)

1.2 Durchführung
1.2.1 Wordpress-Update auf neueste Version (1 Stunde)
1.2.2 Strukturelle Erstellung der Test-Webseite (5 Stunden)
1.2.3 Umsetzung von Änderungswünschen (2 Stunden)
1.2.4 Erstellung von Test-Datenbank (Stammkunden) (1 Stunde)
1.2.5 Programmierung von Login und Password-Hash Verschlüsselung (8 Stunden)
1.2.6 Programmierung von Sessions und Sichtbarkeit der Menü-Elemente anpassen (6 Stunden)
1.2.7 Einbetten von Google-Funktionen, Maps und Recapcha (1 Stunde)
1.2.8 Programmierung von Custom Post Type (CPT) (4 Stunden)
1.2.9 Programmierung von Custom Anhang (.pdf) zum Custom Post Type, und Anhang lösbar machen (6 Stunden)
1.2.10 Programmierung von Suchtaxonomien für CPT (8 Stunden)
1.2.11 Programmierung von Dropdown-Box Logik für Suchtaxonomien (8 Stunden)
1.2.12 Layout Gestaltung von selbst programmierten Funktionen (CSS) (1 Stunde)
1.2.13 Testen und Fehlerbehebung (1 Stunden)
1.2.14 Anpassungen (1 Stunde)
1.2.15 Integration/Migration (1 Stunde)
1.2.16 Qualitäts-Sicherung/Testen (0,5 Stunden)

1.3 Übergabe/Auswertung/Dokumentation (1 Stunde)
1.3.1 Erstellung des Soll-Ist-Vergleiches (1 Stunden)
1.3.2 Abnahme/Übergabe (0,5 Stunden)
1.3.3 Erstellung des Ausblicks/Fazit (1 Stunde)
1.3.4 Erstellung der Dokumentation (8 Stunden)
Summe: 70 Stunden

7. Projektphasen mit Zeitplanung

Aus den Projektnotizen wird am Ende eine Dokumentation erstellt. Änderungen zum Projektantrag werden in der Dokumentation benannt und begründet.

8. Anlagen

Keine

9. Präsentationsmittel

- Notebook
- Power Point
- Beamer
- Flipchart

10. Hinweis

Ich bestätige, dass der Projektantrag dem Ausbildungsbetrieb vorgelegt und vom Ausbildenden genehmigt wurde. Der Projektantrag enthält keine Betriebsgeheimnisse. Soweit diese für die Antragstellung notwendig sind, wurden nach Rücksprache mit dem Ausbildenden die entsprechenden Stellen unkenntlich gemacht.

Mit dem Absenden des Projektantrages bestätige ich weiterhin, dass der Antrag eigenständig von mir angefertigt wurde. Ferner sichere ich zu, dass im Projektantrag personenbezogene Daten (d.h. Daten über die eine Person identifizierbar oder bestimmbar ist) nur verwendet werden, wenn die betroffene Person hierhin eingewilligt hat.

Bei meiner ersten Anmeldung im Online-Portal wurde ich darauf hingewiesen, dass meine Arbeit bei Täuschungshandlungen bzw. Ordnungsverstößen mit „null“ Punkten bewertet werden kann. Ich bin weiter darüber aufgeklärt worden, dass dies auch dann gilt, wenn festgestellt wird, dass meine Arbeit im Ganzen oder zu Teilen mit der eines anderen Prüfungsteilnehmers übereinstimmt. Es ist mir bewusst, dass Kontrollen durchgeführt werden.

Zeitmitschreibung der Projektarbeit

Dieses Formular ist mit der Projektdokumentation einzureichen!

Prüfungsteilnehmer/-in

Name: Bekefi

Vorname: Barbara Anna

Login: 1831496126

(Login des Online Portals)

	Tätigkeit	Zeit in Stunden
16.9	Initiale Projektbesprechung und Zielsetzung	0,5
16.9	Analyse des Ist-Zustandes/Ausgangssituation	1
16.9	Erstellung des Soll-Konzepts	1
16.9	Ableitung des Projektzieles, -Ablaufes, -Aufgaben	1
17.9	Ableitung des Projektzieles, -Ablaufes, -Aufgaben	2
17.9	WordPress-Update auf neueste Version	0,5
18.9	WordPress-Update auf neueste Version	0,5
18.9	Strukturelle Erstellung der Test-Webseite	0,5
18.9	Umsetzung von Änderungswünschen	0,5
18.9	Anpassungen	0,5
23.9	Umsetzung von Änderungswünschen	0,5
23.9	Erstellung der Test-Datenbank von Stammkunden	1
23.9	Programmierung von Login und Password-Hash Verschlüsselung	1
23.9	Programmierung von Sessions mit Menüprüfung	1
23.9	Einbetten von Google-Funktionen: Maps und ReCAPTcha	0,5
23.9	Layout Gestaltung von selbst programmierten Funktionen (CSS)	0,5
24.9	Programmierung von Sessions mit Menüprüfung	4
25.9	Strukturelle Erstellung der Test-Webseite	2,5
25.9	Programmierung von Sessions mit Menüprüfung	2
25.9	Erstellung der Dokumentation	1
30.9	Umsetzung von Änderungswünschen	1
30.9	Programmierung von Dropdown-Box Logik für Suchtaxonomien	4
2.10	Strukturelle Erstellung der Test-Webseite	0,5
2.10	Umsetzung von Änderungswünschen	1
2.10	Testen und Fehlerbehebung	0,2
2.10	Erstellung der Dokumentation	1
5.10	Strukturelle Erstellung der Test-Webseite	0,5
5.10	Anpassungen	1
5.10	Erstellung des Soll-Ist Vergleich (Zeitlich, Ressource, Inhaltlich)	1
5.10	Abnahme/ Übergabe	0,5
5.10	Erstellung der persönlichen Fazit/Ausblick	0,5
8.10	Layout Gestaltung von selbst programmierten Funktionen (CSS)	0,5
9.10	Testen und Fehlerbehebung	0,2
9.10	Erstellung der Dokumentation	4
18.9.	Programmierung von Login und Password-Hash Verschlüsselung	3
18.9.	Layout Gestaltung von selbst programmierten Funktionen (CSS)	1
18.9.	Testen und Fehlerbehebung	0,2

2.10.	Anpassungen	1
21.9.	Programmierung von Login und Password-Hash Verschlüsselung	2
23.9.	Programmierung von Custom Post Type	0,5
24.9.	Programmierung von Custom Post Type	0,5
25.9.	Anpassungen	0,5
25.9.	Programmierung von Dropdown-Box Logik für Suchtaxonomien	2
25.9.	Testen und Fehlerbehebung	0,2
28.9.	Programmierung von Custom Anhang (PDF) zum Custom Post Type, und Anhang lösbar machen	4
28.9.	Programmierung von Suchtaxonomien für CPT	3
29.9.	Programmierung von Dropdown-Box Logik für Suchtaxonomien	8
29.9.	Layout Gestaltung von selbst programmierten Funktionen (CSS)	0,5
5.10.	Umsetzung von Änderungswünschen	1
5.10.	Testen und Fehlerbehebung	0,2
6.10.	Layout Gestaltung von selbst programmierten Funktionen (CSS)	0,5
7.10.	Anpassungen	2
8.10.	Erstellung der Dokumentation	4
	Summe	**72**

Dokumentation für das Projekt

Erstellen einer Webseite in WordPress, Programmierung der gewünschten Funktionen in PHP und der dazugehörigen Datenbank in MySql

Abschlussprojekt für die Ausbildung zum
Fachinformatiker/in Fachrichtung Anwendungsentwicklung

Autorin
Barbara A. Bekefi

Projektzeitraum
16.09.2019 bis 10.10.2019

Inhaltsverzeichnis

Dokumentation zur Projektarbeit

Zum oben aufgeführten Projekt habe ich eine prozessorientierte Dokumentation erstellt, welche die einzelnen Arbeitsschritte in einer zeitlichen Abfolge beschreibt. Die Dokumentation wurde teilweise während der Projektarbeit mitgeschrieben, oder teilweise am Ende des Projektes aus Notizen erstellt. In der letzten Phase folgte abschließend die Formatierung gem. IHK Vorgaben sowie die Korrektur. Die jeweils gewählte Zeitform der einzelnen Abschnitte, ist dem Dokumentations-Erstellungsprozess geschuldet und wurde bewusst nicht angepasst, da es den realistischen Ablauf zeigen soll. Die im Rahmen der Projektdokumentation erstellten Quelltexte, Diagramme, Gesprächsprotokolle und ähnliches werden im Projektkontext oder im Anhang der Projektdokumentation dargestellt.

Verwendungsvermerk zur Projektarbeit

In Absprache mit dem Projektgeber wurde genehmigt, dass die vorliegende Projektarbeit zum Zweck der IHK-Prüfung der Prüfungs-Kommission vorgelegt werden darf. Die namentlich genannten Personen dürfen in Absprache der betroffenen verwandt werden. Kunden der Firma dürfen nur in Absprache mit dem Projektgeber namentlich genannt werden. Die Quellen der nicht selbst erstellten Quelltexten, Bilder usw. werden im Anhang genannt.

Quellenverzeichnis

- Becker, Jochen (2001): Marketing-Konzeption, 7. Aufl., München 2001
- Diller, Herrmann (2008): Preispolitik, 4. Aufl., Stuttgart/Berlin/ Köln 2008
- Lingenfelder, Michael (1995): Lebensstile, in: Handwörterbuch des Marketing, 2. Aufl., hrsg. Tietz, B./Köhler, R./Zentes, J., Stuttgart 1995, Sp. 1377 – 1392.

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

- CMS — Content Management System
- CPT — Custom Post Type
- CSS — Cascading Stylesheet
- KZA — Kundenzufriedenheitsanalyse
- QA — Quality Assurance
- UAT — User Akzeptanz Test
- WP — WordPress

1. Ausganssituation

1.1. Projektumfeld

Der Projektgeber IT-Net.solutions ist ein Kleinunternehmen mit dem Geschäftsführer Herrn Jörn Schmidt. Die drei permanenten Angestellten führen den ca. 15 externen Mitarbeitern die jeweiligen Aufgaben in den Bereichen, wie zum Beispiel der Elektroinstallation oder aber der Buchhaltung zu. Die Firma ist ein IT-Dienstleister mit Sitz in Wandlitz und betätigt sich in den folgenden Tätigkeitsfeldern:

- Alarm-Solutions (Objektsicherheit)
- IT-Net.solutions (IT- und Netzwerklösungen)

Die Dienstleistungen richten sich an Privatpersonen, klein- sowie mittelständige Unternehmen. Dem Kunden werden Komplettlösungen von Planung über die Installation bis zu Wartung in den beiden genannten Geschäftsbereichen angeboten. So umfasst das Portfolio im Bereich Alarm-Solutions Dienstleistungen, welche u.a. Bewegungsmelder, Tür- und Fensterkontakte, Videoüberwachung, Vernebelung, CO_2 – und Rauchmelder, Fernbedienung, Sirenen bei dem Kunden integrieren. Lösungen werden durch Produkte mit verschiedenen Kooperationspartnern realisiert. Zu einem werden die Produkte der Partner, wie z.B. Jovision, ein deutscher Hersteller von Kameras und Videoüberwachungssystemen, oder der größte Kooperationspartner Jablotron, eine internationale Firma mit umfassender Produktpalette rund um Sicherheit, benutzt. Zum anderen baut ein deutsches Fertighausunternehmen die vom Betrieb geplanten Sicherheitslösung zu einem Pauschalpreis in die Kundenhäuser ein. Je nach Bedarf des Bau-Kunden, werden die notwendigen Instandhaltungen, Reparatur- und Wartungsaufgaben von Alarm-Solutions übernommen.

1.2. Ausgangslage (Ist-Zustand)

Der Projektgeber betreibt zurzeit eine Webseite (https://it-net.solutions), welche viele Fehler aufweist. Bei dem Definitionsgespräch mit dem Projektgeber wurden folgende Fehler grob erörtert:

- Die Seite ist nicht für unterschiedliche Bildschirmgrößen optimiert (Responsives Design)
- Die Seite beinhaltet inhaltlich inkonsistente und redundante Menüpunkte
- Die Seite beinhaltet funktionell defekte Menüpunkte und Elemente
- Das Corporate Design ist inkonsequent ausgeführt
- Die Erfahrung mit der vorhandenen Chatfunktion hat sich als wenig praktikabel erwiesen.
- Die oben genannten zwei Geschäftsbereiche (Alarm-Solutions, IT-Net.solutions) werden nicht durch die vorhandene WWW-Firmenpräsens wiedergespiegelt. Der Projektgeber wünscht eine grundsätzliche Teilung, da gemäß seiner Aussage es oft in der Vergangenheit vorgekommen ist, dass dieser Umstand die Kunden irritierte.

In der Besprechung wurde beschlossen, dass primär der Teil der Alarm Solutions Webseite funktionell neu entwickelt werden soll. Es soll darauf geachtet werden, dass die oben aufgezählten Defizite der alten Seite, nicht auftreten sollen. Weiterhin wurde genehmigt, das zu Zweck der IHK-Prüfung dieses Projekt dokumentiert werden soll und der IHK vorgelegt werden darf.

1.3. Soll-Zustand

Ziel des Projektes ist es, mit dem Content-Management-System WordPress eine komplette Web-Präsens für den Alarm-Dienstleistungsbereich zu erstellen. Dafür sollte ich sowohl die Alarm-Solutions Webseiten als auch die Startseite[1] entwickeln. In Absprache mit dem Projektgeber soll darauf geachtet werden, dass die oben genannten vorhandenen Probleme eliminiert werden sollen. Zusätzlich ist darauf zu achten, dass die zu erarbeitende Lösung eine geringe Webseitenunterhaltung (Arbeitsaufwand, Kosten) nach sich zieht.

1.4. Projektziele/Aufgaben

Folgende Aufgaben wurden abgeleitet, um die gewünschten Anforderungen des Soll-Zustandes zu erreichen.

[1] Aus der Startseite soll der Kunde durch JS Onclick-Event die Alarm-Webseiten inkl. Alarm-Homepage (hier beschriebenes Durchgeführtes Projekt) oder die it-net.solutions Seiten (nicht Teil des Projekts) erreichen.

- Installation von XAMPP, Notepad++, und PhpMyAdmin (siehe Ressourcenplanung)
- Analyse der Daten und Anforderungen des Projektgebers und deren Kundschaft (IST)
- Erstellung eines Modells und einer Sitemap (SOLL)
- Import der derzeitigen Webseiten-Daten auf die lokale Testumgebung
- Erstellung der responsiven Webseite in WordPress (Startseite, Homepage, Unterseiten)
- Erstellung einer Datenbank mit zugehöriger Tabelle
- Erstellung des Child-Themes
- Entwicklung eines Login-Bereichs mit einem PHP nativen Verschlüsselungsalgorithmus
- Programmierung von Custom Post Type (CPT) und dazugehörigem Plugin für das Hochladen, Speichern, und Herunterladen von Gebrauchsanweisungen in PDF-Format
- Programmierung von Suchtaxonomien - hierarchisch und nicht hierarchisch
- Programmierung von Dropdown-Box Logik für Suchtaxonomien
- Sessions programmieren mit Menüprüfung
- Migration von Localhost in die Live-Umgebung

2. Technische Planung des Projektziels

2.1. Auswahl von WordPress als CMS

Bevor mit der technischen Planung begonnen wurde, musste ausgehende der vorhandenen Ressourcen analysiert werden, welche Betriebsmittel für den Einsatz sinnvoll sind. Die Firma hatte ein existierendes WordPress-Konto. So stellte sich die Frage, ob WordPress in der Lage ist, die mit dem Projektgeber besprochenen Forderungen zu erfüllen. Folgende Punkte sprachen für den Einsatz:

- Die vorhandene Seite it-net.solutions benutzt schon WordPress. Der Aufwand einer erforderlichen Migration (Kosten) auf ein neues System würde bei einem Weiterbetrieb so entfallen.
- Vorhandene funktionelle und gestalterische Komponenten können auf die neue Seite übernommen werden.
- Die Mitarbeiter sind auf WordPress geschult, sodass ein neues Einarbeiten entfällt.
- WordPress ermöglicht den Mitarbeitern, neue Seiten oder notwendige Änderungen ohne fortgeschrittenen Programmierkenntnissen zu erstellen
- Es ist erkennbar, dass WordPress zunehmend populärer unter Firmen wird, weil es durch kostenfreie *Plugins* viele Funktionen zur Verfügung stellt oder können ggf. selbst erstellt oder ggf. angepasst werden. Weiterhin bietet die neue WordPress-Programmversion die Möglichkeit, die Darstellung für unterschiedliche Endgeräten (Desktop, Handy oder Tablett) zu optimieren.

Bei der Darlegung dieser Fakten, wurde mit dem Projektgeber beschlossen, mit dem vorhandenen System weiterzuarbeiten und auf die neuen Erfordernisse jeweils anzupassen.

2.2. Auswahl Programmiersprache und Editor statt IDE

Die Programmierung der gewünschten Features musste in PHP stattfinden, weil das WordPress-System auf PHP basiert. Die Autorin brauchte hierfür einen einfachen Editor. Alternativ hierzu wäre die Verwendung einer IDE-Umgebung möglich gewesen. Jedoch wurde in allen Vorgängerprojekten erfolgreich Notepad++ verwendet. Durch diese vorhandene Erfahrung musste sich nicht aufwendig in eine neue Arbeits-Umgebung eingearbeitet werden. Ein weiterer Vorteil ist, dass Notepad++ kostenfrei verwendet werden darf, und einfach zu installieren und zu benutzen ist. Aus diesen Gründen wurde die Entscheidung getroffen, das Editor Notepad++ zu verwenden.

2.3. Technische Anforderungsanalyse und Vorgehensplanung

Bei weiteren Gesprächen mit dem Projektgeber wurden folgende Wünsche erörtert. Es wurde nun das jeweilige technische Vorgehen geplant:

- Projektgeberseitiger Wunsch ist es, einen kennwortgeschützten User-Bereich zu erstellen. So beabsichtigt der Projektgeber, die Daten der Webseite und die Kundendaten auf der eigenen sicheren Cloud zu speichern. Für die Speicherung der Kundendaten hatte ich geplant, mit

phpMyAdmin eine entsprechenden MySQL Datenbank mit Dummy-Kundendaten zu erstellen. Diese Dummy-Daten werden nach dem Entwicklungsprozess durch reale Kundendaten ersetzt werden. Die Kennwörter in der Datenbank sollen die PHP-native Password-Hash Funktion verschlüsselt werden. Um diese Funktionalitäten zu realisieren, soll zu diesem Zweck mit PHP eine Login-Funktion programmiert werden.

- Der Projektgeber hat auch geplant, den Kunden eine Datenbank von Gebrauchsanweisungen für die Produkte zur Verfügung zu stellen. Diese plante ich mithilfe der WordPress-nativen Datenbank entwickeln (Custom Post Type - CPT). Die Default Dateitypen, die einem WordPress-Post angehängt werden können, sind .JPG und .PNG; für das Uploaden von anderen Dateitypen stehen Plugins zur Verfügung.
- Die Gebrauchsanweisungen sollen von Kunden durchsuchbar sein. Für diesen Zweck sollen in PHP zwei Drop-Down Boxen programmiert werden, entsprechend dem von Projektgeber gewünschten zwei Suchkategorien. Diese Suchkategorien sind in WordPress Taxonomien genannt. Der Projektgeber hat eine Suchfunktionalität nach Hersteller und eine nach Produkttyp erwünscht, die voneinander abhängig sind. Geplant ist eine hierarchische (sogenannte Kategorie) und eine nicht hierarchische (sogenannte Tags) Suchtaxonomie zu erstellen.
- Einige Menü-Funktionen sollen nur für ein- bzw. für ausgeloggte Kunden sichtbar sein. Dies soll durch die Programmierung von Sessions und einem Plugin (Conditional Menus) gelöst werden.
- Um meine selbst programmierte Funktionen nach einem eventuellen Update des WordPress-Themes oder bei der Auswahl eines neuen Themes zu schützen, plante ich ein Child-Theme des von mir verwendeten WordPress-Themes zu erstellen und in diesem Child-Theme weiterzuarbeiten. Dies ist laut WordPress-Codex der empfohlene Weg, wenn zu einem WordPress-Theme selbst programmierte Funktionen gebraucht werden.[2]

2.4. Gestalterische Planung

Die Leitung der Firma hatte keine klar definierte Vorstellung, wie die Webseite gestalterisch und inhaltlich aussehen soll. Nach dem Kennenlernen der gestalterischen Präferenzen habe ich beschlossen, unter Zuhilfenahme des Balsamiq Mockup Tool[3] ein Mockup-Plan der Startseite zu erstellen. Der Vorteil von Mockup-Tools liegt daran, dass die Änderungen ganz einfach, ohne Programmieren und bei Bedarf sogar direkt durch die Kunde durchgeführt werden können. Diese Vorentwürfe wurden dem Projektgeber zur Genehmigung vorgelegt.

Abbildung 1: Mit Mockup-Tool erstellter Header-Bereich der ersten Version

2.5. Strukturelle Planung durch Sitemap

Eine Sitemap (Baumstruktur-Modell) von Webseiten und Unterseiten, bzw. Menüfunktionen wurde mithilfe eines Modelling-Tools (Umlet[4]) erstellt (siehe Seite 19 im Anhang)

2 Quelle: https://developer.wordpress.org/themes/advanced-topics/child-themes/ (Stand: Oktober 2019)
3 https://balsamiq.com/wireframes/ (Stand: Oktober 2019)
4 https://www.umlet.com/ (Stand: Oktober 2019)

3. Ressourcen- und Ablaufplanung

3.1. Personalplanung

Die Bearbeitung des Projekts wurde von mir in Eigenarbeit durchgeführt. Weitere beteiligte Personen stehen außerhalb des Projekts und sind bei den personellen Schnittstellen aufgeführt worden.

3.2. Personelle Schnittstellen

Um das Projekt durchführen zu können sind die folgenden genannten Personen notwendig. Diese stehen außerhalb des Projektes und sind mit den genannten Aufgaben beteiligt.

- Projektleiterin: Frau Dagmar Schumacher
  Aufgaben: Projektdefinition, Ansprechpartner bei projekttragenden Fragen, Abnahme des Projektes.
- Fachinformatiker im Bereich Systemintegration[5]
  Bereitstellung der Hard- und Software (Lizenzen) sowie der erforderlichen Benutzerzugänge (Konten und Passwörter)

3.3. Technische Schnittstellen

Folgende technische Schnittstellen mit dem jeweiligen Bezug zum Projekt sind unabdingbar:

- Alte Webseite mit dem WordPress-Konto, um gewünschte Funktionen und die Gestaltungen zu übernehmen sowie diese in die neue Seite zu integrieren.
- Da die Ansprechpartner bei IT-Net.solutions oft off-site arbeiten, war auch für die Kommunikation eine Kerio Mailserver erforderlich; es ermöglicht, bei Fragen die Ansprechpartner per Email oder Chat zu erreichen.
- Das firmeneigene Synology Cloud Station Drive und der Securepoint SSL VPN-Netzwerk ermöglichte bei Home-Office Arbeit die Synchronisierung mit dem Secure Server der Firma.

3.4. Projektkosten

Die Selbstkosten eines Unternehmens enthalten alle Kosten, die für die Realisierung des Produkts notwendig sind. Innerhalb dieser breiten Kategorie wurden nur folgende projektrelevante Unterkategorien berücksichtigt. [6]

Gemeinkosten sind in breitem Sinne Kosten, die aus mehreren Aufträgen mit mehreren Kostenträgern bestehen. Das fertige Produkt kann nicht einer Einheit zugeordnet werden. Gemeinkosten können zu fixen - projektunabhängigen und variablen - projektabhängigen - Kosten weiter unterteilt werden. Für die Kalkulation können nur für die Durchführung des Projekts benötigte variable Gemeinkosten berücksichtigt werden. Diese setzen sich sowohl aus Personal-, als auch aus projektbezogenen Ressourcenkosten zusammen. Für die projektbezogenen Ressourcen wird ein pauschaler Stundensatz von brutto 10 € Platzkosten angenommen. Dieser beinhaltet den Arbeitsplatz (Raum- und deren Betriebskosten), Arbeitsplatzrechner inkl. der benötigten Softwarelizenzen. Der Ausbildungssituation geschuldet entstehen dem Projektgeber keine Personalkosten. Um trotzdem eine reale Kostenabschätzung zu erstellen, wurde für die Kalkulation der Stundenlohn eines Azubis[7], bzw. von Fachinformatikers für Anwendungsentwicklung[8] wie folgt ermittelt:

> **Jahresarbeitsstunden für Azubi** (8 h/Tag * 220 Tage/Jahr = 1760 h/Jahr):
> 1000 €/Monat * 12 Monate/Jahr = 12000 €/Jahr
> 12000 €/Jahr: 1760 h/Jahr ≈ 6,82 €/h

Der Stundensatz für einen Mitarbeiter musste auch abgeschätzt werden. Auf Nachfrage wurde auf diese persönlichen Daten kein Einblick gegeben:

---

5 Es liegt keine Genehmigung vor, den Systemintegrator namentlich zu nennen zu dürfen.
6 Es ist anzumerken, dass der Projektgeber auf Nachfragen selbst keine detailliert kategorisierte Übersicht über diese Kosten hat. Aus diesem Grund wurden die Kosten ggf. möglichst aus vergleichbaren Werten abgeleitet.
7 https://www.ausbildung.de/berufe/fachinformatiker-anwendungsentwicklung/gehalt/ (Stand: Oktober 2019)
8 https://www.gehalt.de/beruf/fachinformatiker-fachinformatikerin (Stand: Oktober 2019)

Jahresarbeitsstunden für Mitarbeiter (8 h/Tag * 220 Tage/Jahr = 1760 h/Jahr):
3000 €/Monat *12 Monate/Jahr = 36000 €/Jahr
36000 €/Jahr :1760 /h/ Jahr ≈ 20,45 €/h

Somit ergibt sich ein Stundenlohn von 6,82 € für die Auszubildende und 20,45 € für die Mitarbeiter. Diese Kosten sollen die zugehörigen Lohnnebenkosten (Rentenversicherung, Gesetzliche Krankenversicherung, Arbeitslosenversicherung, Pflegeversicherung, Gesetzliche Unfallversicherung) beinhalten (Bruttowerte). Die Durchführungszeit des Projekts beträgt 70 Stunden. Somit setzen sich die Personalkosten wie folgt zusammen und betragen 1.463,70 €:

Personalkosten	Geplante Zeit (in Mitarbeiter-Stunde)	Kosten pro Stunde	Gesamtkosten
AZUBI Projektdurchführung	70	16,82 €	1177,40 €
Fr. Schumacher für Meilenstein-Gespräche[9]	10	20,45 €	204,50 €
Systemintegrator für Ressourcenbeschaffung	2	20,45 €	40,90 €
2 Mitarbeiter Fr Schumacher, Systemintegrator für Projekt-Abnahme	2	20,45 €	40,90 €
Gesamt			**1.463,70€**

Tabelle 1: Personalkostenaufstellung

Werden alle Kosten zusammengeführt, ergeben sich Projektgesamtkosten von 2.163,70 €.

Projektkosten	Gesamtkosten
Platzkosten (10,00 € x 70h)	700,00 €
Personalkosten	1.463,70 €
Gesamt	**2.163,70 €**

Tabelle 2: Gesamtkosten

3.5. Projektprozess/Ablauf

Die Autorin hat für die Planung und Durchführung des Projekts neben dem Vorgehensmodell *Prototyping* entschieden. Prototyping ist eine Methode der Softwareentwicklung, die schnell zu ersten Ergebnissen führt und *frühzeitiges Feedback* bezüglich der Eignung eines Lösungsansatzes ermöglicht[10]. Dadurch sind Probleme und Änderungswünsche frühzeitig zu erkennen und mit weniger Aufwand zu beheben, als es nach der kompletten Fertigstellung möglich gewesen wäre. Diese Methode eignet sich besonders gut in Projekt-Situationen, wo der Projektgeber keine klare Vorstellung über das Endprodukt hat – was bei der Erstellung von Webseiten oft der Fall ist. Ein weiterer Vorteil von Prototyping ist, dass es für kurzfristigen Projekte, wie das gegenwärtige Projekt, gemeint ist. Um die Projektwünsche zu klären, war eine intensive und regelmäßige (mindestens einmal pro Woche) Feedback- und Rücksprache mit der Geschäftsführung zwingend erforderlich. Im Rahmen dieser Besprechungen wurden die umgesetzte Projektziele vorgestellt, und Änderungswünsche und weitere Projektschritte erläutert. So habe ich zum Beispiel bei dem ersten Meeting den Projektgebern eine Sammlung verschiedener Typen von Partner-Webseiten gezeigt, um den Projektgebern Ideen zu geben und zu helfen, bezüglich Gestaltung der Webseite konkrete Entscheidungen zu treffen.

3.6. Sachmittelplanung

Folgende Sachmittel wurden benötigt, um das Projekt durchführen zu können. Vorhandene notwendige Ressourcen, welche nur in der vorhandenen Funktion verwendet werden, sind in den technischen Schnittstelen benannt. Die zu erwartenden Kosten werden in den Projektkosten ggf. aufgeführt.

- **Notepad++ Editor**:
  Skripterstellung, gute Projekterfahrungen, kostenfreie Benutzung.
- **XAMPP**:
  Lokale Testumgebung, gute Projekterfahrungen, kostenfreie Benutzung.

[9] Geschätzter zeitlicher Aufwand der geplanten regelmäßigen Besprechungen
[10] Quelle: https://de.wikipedia.org/wiki/Prototyping_(Softwareentwicklung) (Stand Oktober 2019)

- **Microsoft Office 365**:
  Power Point für Projektpräsentation, Outlook für Emails, Word und Excel für die Dokumentation, gute Projekterfahrungen, in den Platzkosten berücksichtigt.
- **Umlet**:
  Dokumentation der UML Diagramme, gute Projekterfahrungen, kostenfreie Benutzung.
- **Kerio Mailserver**:
  Email- und Chatkommunikation aller beteiligten Personen, Einarbeitung fand außerhalb des Projektes statt, in den Platzkosten berücksichtigt.
- **Microsoft Photo Editor** und **Adobe Photoshop**:
  Bildbearbeitung für Projektinhalte und Dokumentation, gute Projekterfahrungen, in den Platzkosten berücksichtigt.
- **Arbeitsplatzrechner**:
  wurde vom Projektgeber gestellt, in den Platzkosten berücksichtigt.
- **Arbeitsplatz**:
  wurde vom Projektgeber gestellt, in den Platzkosten berücksichtigt.

3.7. Projekt-Amortisation

Basis der Kalkulation ist der Projektgeber-Wunsch, Unterlagen an Kunden jederzeit zur Verfügung zu stellen. Zurzeit werden diese Informationen durch folgende Möglichkeiten übermittelt:

- Kundentelefonat bezüglich Produktinformationen
- Zusendung von Gebrauchsanweisungen per Post/Email
- Wegen schlechter Nachvollziehbarkeit durch zu spät übergebene Dokumente (z. B. Zeitdruck) kommt es zu unnötigen Kundenbesuchen

Auf Nachfrage, wie lange dieser Arbeitsprozess in etwa dauern, wurde geschätzt, dass drei Mitarbeiter wöchentlich ca. je 3 Stunden mit diesen Arbeiten verbringen.

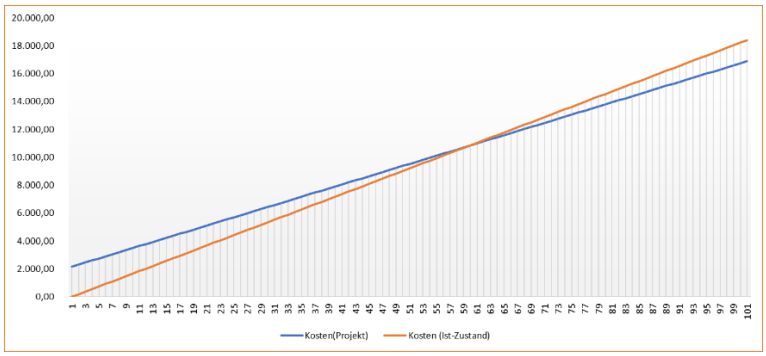


Abbildung 2: Amortisation-Entwicklung (Diagramm)
(siehe Anhang Tabelle 7: Amortisation-Entwicklung)

Wird der Stundensatz von 20,45 €/h zugrunde gelegt, ergibt sich ein monetärer Aufwand von: 20,45 €/ * 9h Mitarbeiterstunden = 184,05 € pro Woche. Die geschätzt berechneten Gesamtkosten für das Projekt betragen 2.163,70 €. Würden sich die Supportanfragen nur um 20% auf 147,24 € minimieren, würde sich das hier dokumentierte Projekt nach ca. 60 Wochen amortisieren und der Projektgeber hätte eine jährliche Einsparung von 1.914,12 € (52 Wochen * 36,81 € Einsparung pro Woche).

3.8. Make or Buy Entscheidung

Im Vorfeld des Projektes wurde erörtert, ob das Projekt an einen externen Dienstleister beauftragt werden sollte. Für die Projektleiterin sprachen zwei Gründe dagegen:

- Durch die Ausbildungssituation entstanden dem Projektbetrieb keine Personalkosten für Projekterstellung (1.463,70 €).
- Das Knowhow soll in der Firma bleiben, um so Folgekosten und Abhängigkeiten gegenüber dritten zu minimieren.

Vor dem Projektbeginn hatte der Projektgeber für die alte Seite eine Vorlage vom Dienstleister Templatemonster[11] erworben. Bei weiterem Gebrauch der Vorlage würden weitere Kosten anfallen, siehe Abbildung 3. Bei der Analyse des technischen Ist-Zustands ist mir aufgefallen, dass sehr oft Ressourcen an das Template angepasst werden mussten, anstatt sich das Template dem Inhalt anpasst. Diese Vorlage ist somit nicht hinreichend für das Projekt geeignet. Außerdem entstehen beim Gebrauch von Templates weitere Kosten von Dienstleistungen, die auch kostenfrei, wie z. B. bei WordPress verfügbar wären.

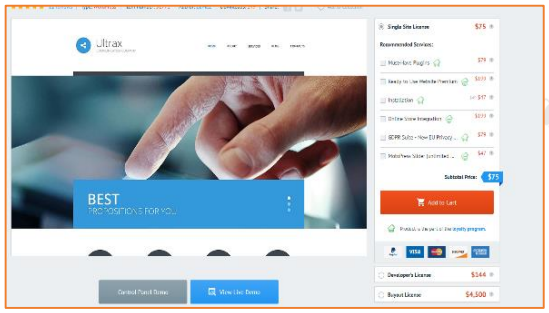


Abbildung 3: Kosten der derzeitigen Webseite

3.9. Ablauf- und Zeitplanung

Ausgehend von der eingerechneten Ablauf- und Zeitplanung des Projektantrages, wurden die angedachten Vorgänge nach dem technischen Sollkonzept zeitlich neu bewertet und ggf. im Ablauf neu einsortiert. Die jeweilig geplanten Vorgänge wurden den folgenden Projektphasen zugeteilt:

Im Anhang 6.1 Zeit und Ablaufplan auf Seite 18 wird das Gant-Diagramm gezeigt, welches über das gesamte Projekt mitgeschrieben worden ist. Der Ablauf sowie die Soll-Zeitplanung wurde nach der Planungsphase eingetragen. Die Ist-Zeiten wurden nach der Fertigstellung des jeweiligen Vorgangs nachgetragen. Die aufgetretenen Differenzen wurden im ablaufzeitlichen Soll-Ist-Vergleich berechnet und in der Auswertungsphase inhaltlich ausgewertet.

4. Umsetzung des Projektziels

Nach Abschluss der Planungsphase konnte nun mit der Umsetzung des Projektes begonnen werden. Geschuldet dem begrenzten Platz einer IHK-Dokumentation entschloss ich mich, exemplarisch in der Dokumentation den Entwicklungsverlauf detailliert zu zeigen. Um die gewünschten Funktionalitäten herzustellen, fiel mir für alle Arbeitsprozesse auf, dass die erforderlichen Arbeiten sich wie folgt kategorisieren lassen:

- Analysieren und Anpassung von Fremdcode
- Neuentwicklung

Selbst bei Neuentwicklungen greift man oft auf das Benutzen vorhandener Funktionalitäten, welche dann zu eigenen Funktionalitäten zusammengesetzt werden. Weiterhin musste ich mich zwischen der Nachvollziehbarkeit der einzelnen Schritte und den IHK-Vorgaben der Platzbegrenzung entscheiden.

4.1. Erstellung der Webseiten in WordPress

Eine schnelle Einarbeitung in WordPress, um eine erste grobe Version der Webseite zu erstellen, gelang mithilfe von YouTube-Videos[12]. Um die benötigten Seiten sachlogisch einzupflegen, erstellte ich das folgend abgebildete Sitemap-Modell (Siehe Abbildung 8: Sitemap-Modell). Das Modell zeigt eine während der Bearbeitungszeit entstandenen Version und wurde in dem Prozess je nach Anforderung angepasst. Die so entstandene erste Version der Webseite diente noch als Mockup, um dem Projektgeber ein erstes Bild zu zeigen (Gespräch mit Projektgeber), um eventuelle strukturelle Fehler frühzeitig auszuschließen.

4.2. UML-Modellierung der Zugriffe

Um funktionell die verschiedenen User-Rollen der bestehenden Benutzer- und Kundenzugriffe klar voneinander abzugrenzen, entstand das folgend gezeigte UML Use-Case Diagramm.

[11] www.templatemonster.com (Stand Oktober 2019)
[12] https://www.youtube.com/watch?v=v62Rd7gGh0Q (Stand Oktober 2019)
https://www.youtube.com/watch?v=8AZ8GqW5iak (Stand Oktober 2019)

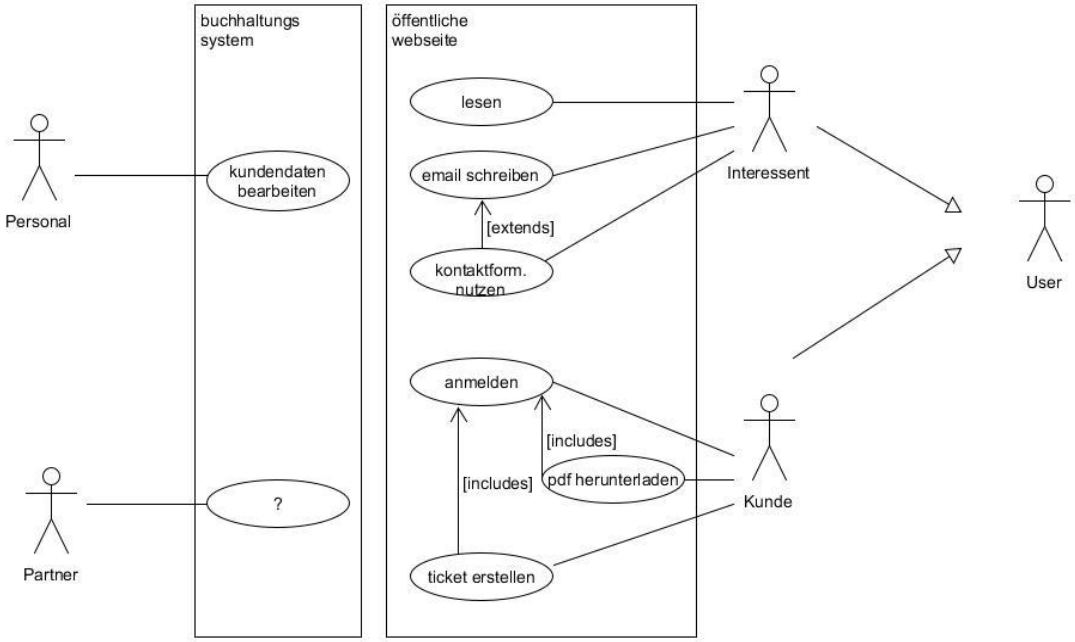


Abbildung 4: Rollen der Benutzer- und Kundenzugriffen

4.3. Implementierung des Kundenlogins (Kennwort-Hash [13])

Ausgehend von den genannten User-Rollen (siehe Use-Case-Diagramm) war es wichtig, die zu erwartenden Kundenzugriffe abzusichern. Hierzu zeigte ich dem Projektgeber, wie die Kundendaten zukünftig abgespeichert werden sollen. Zweck Schutz von Kennwort-Angriffen entschied ich mich, die Kennwörter mit einer serverseitigen Verschlüsselung zu schützen. Hierzu gibt es u.a. die Methoden MD5 und SHA1 sowie die PHP-native Password Hash Funktion. Diese basiert auf Verschlüsselungs-algorithmen wie Blowfish, Argon 2, und Argon2I. Ich wählte Blowfish aus, da die Hash-Funktion ohne Parameter-eingabe auskommt. Zusätzlich bietet diese Funktion auch das sogenannte Salt[14] an. In dieser Kombination ist es zurzeit einer der sichersten Verschlüsselungs--Algorithmen. Die folgend dargestellte Tabelle zeigt die gehashten Kennwörter in der Kunden-Datenbank. Es ist zu sehen, dass zwei Kunden (testbenutzer1, testbenutzer5) sich schon einmal erfolgreich eingeloggt habe. Bei diesem ersten Login wurden die originalen Kennworte aus der Tabelle gelöscht. Bei den anderen Kunden (testbenutzer2 bis -6) ist das noch nicht der Fall.

kunden_id	benutzername	kennwort	eingeloggt	hashedpassword
1	testbenutzer1		0	$2y$10$flKVFU.WCO2Zb7wF8RBlo.HNqjAbGjQl4CY2NQ1NLXK...
2	testbenutzer2	testkennwort2	0	$2y$10$8pqXOilq25Os/0OhR98rseJbLzT2I4wO3svVCvsigwo...
3	testbenutzer3	testkennwort3	0	$2y$10$TAKe7f59xzf5Xq14dMqlWeSydg7xP.UYWWuHpp5/hQN...
4	testbenutzer4	testkennwort4	0	$2y$10$d4937Y4utM3DD5BwYtLWCuT3RhBEFgKc80L0ddSx6ui...
5	testbenutzer5		0	$2y$10$Qa5OvXDaPPAXpvnxYYdIsuL1CqMnx78ucTB9ZC7BT5g...
6	testbenutzer6	testkennwort6	0	$2y$10$WXkIk5Kqn8HNSJHK0SJnFe10.5/u8FJ2iU.g5ry785E...

Abbildung 7: Kundentabelle mit Hash-Kennwörtern

Eine Besonderheit der Aufgabe war, dass es in diesem Login-System weder Registration- noch Self-Service Funktionen gibt; es war ein spezieller Wunsch des Projektgebers, die Zugangsdaten ihren Kunden selbst zu übergeben.

4.4. Implementation von Sessions:

Folgend wird gezeigt, wie ich das Sessions- und Menu-Items-Visibility-Plugin implementiert habe. Ziel ist es, externen selbst erstellten PHP-Code in WordPress auszuführen. Da ich mich in dieses

[13] https://www.php.net/manual/de/faq.passwords.php (Stand: Oktober 2019)

[14] *Salt* (englisch für *Salz*) bezeichnet in der Kryptographie eine zufällig gewählte Zeichenfolge, die an einen gegebenen Klartext vor dessen weiterer Verarbeitung (z. B. Eingabe in eine Hashfunktion) angehängt wird, um die Entropie der Eingabe zu erhöhen. Quelle: https://de.wikipedia.org/wiki/Salt_(Kryptologie) (Stand: Oktober 2019)

Thema einarbeiten musste, habe ich u.a. ausprobiert, die WordPress-Datei *functions.php* unter den Child-Theme umzuschreiben[15]. Dies hat funktioniert, soll aber nicht zuverlässig sein und zu Sicherheitsproblemen führen. So sollen laut der Kritik, nicht autorisierte Personen auf den Source-Code zugreifen können. Ein anderer Weg ist es, das Plugin *PHP-Everywhere*[16] zu installieren. Hierzu muss das Plugin in den einem selbst erstellten Ordner eingebunden werden. Hierzu wurde der Ordner *includes* erstellt, indem gearbeitet werden soll:

```
D:\xampp\htdocs\dev-alarmsol\wp-content\themes\sydney-child-files\includes
```

Die dazu erstellte Testdatei heißt team.php, und wurde in den includes Ordner kopiert. Nun gibt es zwei Wege, auf Dateien in diesem Ordner zu zugreifen. Beide Möglichkeiten müssen in das PHP-Everywhere Plugin eingegeben werden, damit sie ausgeführt werden können:

Möglichkeit 1:
von Root aus, und den ganzen absoluten Pfad definieren:

```
include_once $_SERVER['DOCUMENT_ROOT'].'/dev-alarmsol/wp-content/themes/sydney-child-files/includes/team.php';
```

Möglichkeit 2:
aus dem Ordner des Plugins PHP-Everywhere. Dort liegt die Datei widget.php, welche die Zeile eval(' ?>'.$content.'<?php ')[17] ausführt. Nun muss noch auf folgenden Code eingebunden werden, welcher auf den Ordner zeigt:

```
include __DIR__ . '/../../themes/sydney-child-files/includes/team.php';
```

Nach Erfolg mit Testdatei können wir nun die gewünschten Funktionalitäten eingebunden werden. Ziel ist es zu kontrollieren, welcher Seitenbereich (Download-Bereich oder Ticket-Bereich[18]) nach einem erfolgreichen Login benutzt werden soll:

```
1. $login_dest='/test-kundenbereich/downloads';
2. include_once $_SERVER['DOCUMENT_ROOT'] .'/dev-alarmsol/wp-content/themes/sydney-child-files/includes/BB-loginform.php';
```

Danach wird der jeweilige Pfad verketten[19]:

```
echo "<script>";echo "document.location='".site_url().$login_dest."'"; echo "</script>";
```

Nun können jeweils die zwei folgenden Snippets auf zwei separaten Seiten benutzt werden:

- *Kundenbereich*-Seite (page_id=35)[20]:

```
3. $login_dest='/test-kundenbereich/ticket-erstellen';
4. include_once $_SERVER['DOCUMENT_ROOT'] .'/dev-alarmsol/wp-content/themes/sydney-child-files/includes/BB-loginform.php';
```

- *Downloads*-Seite (Custom Post Type, post_id='downloads')

Diese CPT hat eine Kind-Seite: Login-Downloads (page_id=322), die nur zwecks Login erstellt worden ist, und ist nicht aus dem Hauptmenu sichtbar oder erreichbar ist:

```
5. $login_dest='/downloads';
6. include_once $_SERVER['DOCUMENT_ROOT'] .'/dev-alarmsol/wp-content/themes/sydney-child-files/includes/BB-loginform.php';
```

Die Datei BB-loginform.php beinhaltet die Login-Logik sowie das Formular. Ob Login-Downloads öffnen soll, entscheidet eine Session. Für den Login habe ich folgende Code implementiert:

```
$_SESSION['userloggedin']=true;
```

$_SESSION ist eine globale variable mit assoziativem Array. Bei erfolgreichem Login wird der Schüssel userloggedin zum Wert True gesetzt. Ausgeloggte Benutzer können anstatt mit

---

15 https://www.emanueleferonato.com/2011/04/11/executing-php-inside-a-wordpress-widget-without-any-plugin/ (Stand Oktober 2019)
16 https://www.alexander-fuchs.net/projects/php-everywhere/ (Stand Oktober 2019)
17 https://developer.wordpress.org/cli/commands/eval/ (Stand Oktober 2019)
18 Während Erstbesprechung war der Kundenwunsch, die Login-Funktion für das Ticket-Bereich zu erstellen. Später wegen Kostenproblemen wurde das Erwerb des von Projektgeber ausgewählte Ticket-Software verschoben. Der Bestandkunde landet deswegen in mit dem gleichen Login in einer leeren Ticket Seite ('coming soon') bzw. im Downloads-Bereich.
19 Hier wird JavaScript mit PHP schon zusammen verwendet.
20 Von Wordpress-Datenbank automatisch hinzugefügter Primärschlüssel

`if(!$user_id)` nun mit `if(!$_SESSION['userloggedin'])` in der ersten if-Anweisung identifiziert. Für eingeloggte Nutzer wird in der Else-Anweisung die variable `$login_dest` die entsprechende Destination-Seite definiert:

```
echo"<script>"; echo"document.location='".site_url().$login_dest."'";echo"</script>";
```

Auf *Downloads* wird es immer geprüft, ob der Benutzer eingeloggt ist. Wenn nicht, dann wird der Benutzer zu Login-Downloads (mit Login-Formular) umgeleitet, von dort bei erfolgreichem Login zurück zu Downloads. Ein eingeloggter Benutzer bleibt auf Downloads, wo kein Login-Formular erscheint.

Ein erkanntes Problem war, das die Funktion `session_start()` immer zuerst initialisiert werden muss, bevor der Header und Footer ladet. Damit die Sessions in WordPress initialisiert werden, muss in die WordPress *functions.php* Datei, welche die Funktionen enthält, folgender Code eingebunden werden:

```
7.  add_action('init', 'start_session', 1);
8.  // add_action: wp eingebaute funktion: wenn etwas initialisiert wird starte Session.
9.  function start_session() {
10.             if(!session_id()) { session_start(); }
11.                             }
12. add_action('end_session_action','end_session');
13. function end_session() {session_destroy ();}
```

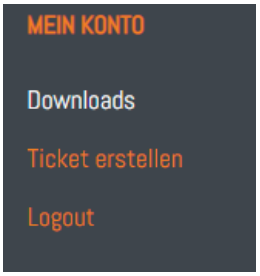
Der nächste Schritt war, die Logout-Funktionalität zur Verfügung zu stellen. Hierzu das PHP-Code-Snippet für die Logout-Seite:

```
$_SESSION['userloggedin']=false;
```

Nach dem Logout wird die Session beendet.

4.5. Integration des Menu-Items-Visibility-Plugin

Das Plugin Conditional Menus erlaubt uns für die Sessions Bedingungen zu definieren, welche die Menü-Sichtbarkeit regeln. So müssen für ein- oder ausgeloggten Kunden jeweils unterschiedliche Menüpunkte erscheinen. Für eingeloggte Kunden stellte sich das Menu wie in gezeigt dar. Für ausgeloggte Kunden wird der Login Link gemäß Kundenwunsch im Header-Menü Bereich dargestellt



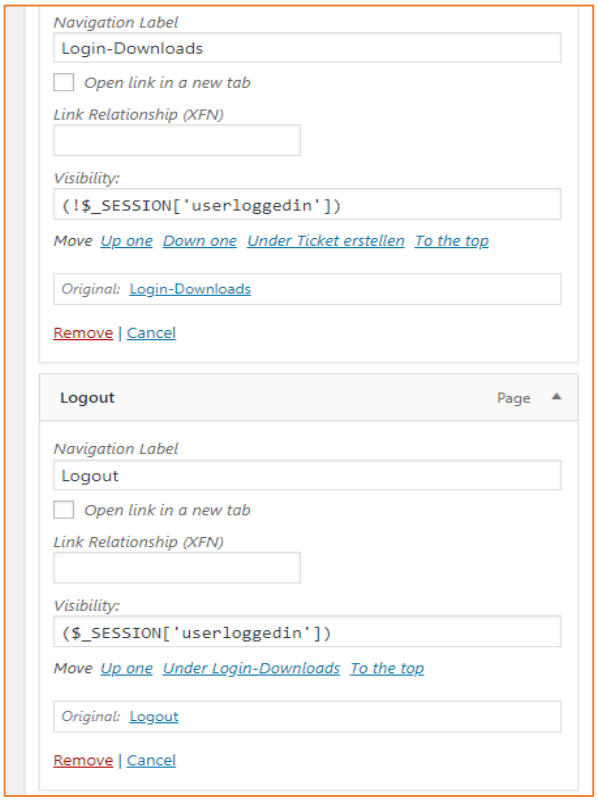


Abbildung 5:
WordPress Back- und Frontend für
ein- und ausgeloggte Kunden

4.6. Erstellung von Custom Metabox und Anhang

Um die geplante Custom Metabox zur Verfügung zu stellen, musste ich den vorhandenen Quellcode schrittweise analysieren und anpassen. Hierzu lagerte ich den geänderten Code dieser Teilaufgabe aus. Um mich fachlich für diese Aufgabe vorzubereiten, hielt ich mich an das Vorgehen des

WordPress-Tutorials https://code.tutsplus.com/articles/attaching-files-to-your-posts-using-WordPress-custom-meta-boxes-part-1--wp-22291. Diese Schritte sind zusätzlich im Dokumentation von Custom Metabox im Anhang 6.4 zu finden.

4.7. Implementierung der Dropdown-Box Logik

Diese Teilaufgabe stellt den Kern meiner Arbeit dar. Erster Schritt war zu verstehen, wie *Terms* und *Kategorien* in einer Taxonomie funktionieren[21]. Die Zusammenfassung der erarbeiteten Schritte ist folgend dargestellt. Der gesamte Code ist zusätzlich im Anhang 6.4. platziert.

Der folgende PHP und JavaScript Code[22] wurde in Archive unter Header reinkopiert:

```
1.  $categories = get_categories('taxonomy=produktgruppe');
2.  $select = "<select name='cat' id='cat' class='postform'>n";
3.  $select.= "<option value='-1'>Select category</option>n";
4.   foreach($categories as $category){
5.              if($category->count > 0){
6.  $select.= "<option value='".$category>slug."'>".$category>name."</option>";
7.                                                  }
8.   }
9.  $select.= "</select>";
10. echo $select;
11. ?> <script type="text/javascript"><!--
12. var dropdown = document.getElementById("cat");
13. function onCatChange() {
14.           if (dropdown.options[dropdown.selectedIndex]. value != -1) {
15.           location.href =
16.           "<?php echo home_url();?>/produktgruppe/"+
17.           dropdown.options[dropdown.selectedIndex].value+"/";
18.                    }
19.                }
20.     dropdown.onchange = onCatChange;
21. --></script>
```

Zusätzlich musste der Quellcode dupliziert (zweite Dropdown-Box) und komplett in einem tag gepackt werden. Damit die beiden Dropdown-Boxen nebeneinander sitzen, wurde der code mit * * getrennt.

Nun mussten die jeweiligen Variablenbezeichner geändert werden, damit die beiden Dropdown-Boxen nicht die gleichen Daten aufrufen: So wurden statt `$dropdown`, die Variablen mit `$dropdown1` und `$dropdown2` benannt. Auch die <select> Tag-Namen wurden wie folgt geändert:

Erstes Dropdown:
`$select = "<select name='hersteller_select' id='hersteller_select'…`

Zweites Dropdown:
`$select = "<select name='produkttyp_select' id='produkttyp_select'…`

Die erste Dropdown-Box wird nur mit der Elternkategorie (Hersteller) befüllt. Wenn diese Kategorie keine Oberkategorie hat, dann ist sie selbst die Ober-Kategorie.

```
22. $categories = get_categories('taxonomy=produktgruppe');//dies zu Array umwandeln
```

Für weitere Bearbeitung muss sie zu einem in ein Array umgewandelt werden [23].

```
23. $categories = get_categories(array('taxonomy'=>'produktgruppe','parent'=>0,));
```

Danach wird die zweite Dropdown-Box mit dem Kind-Kategorien-Array (=Produktgruppe) befüllt.[24]. Der Code wurde wie folgt geändert:

```
24. $categories = get_categories(array('taxonomy'=>'produktgruppe',
25.                          'parent'=>$wp_query->get_queried_object_id())),
```

[21] Quelle: https://www.wpbeginner.com/glossary/terms/ https://www.wpbeginner.com/glossary/category/ (Stand Oktober 2019)

[22] Quelle: https://pagecrafter.com/dropdown-Menü-of-all-terms-in-custom-taxonomy-wordpress/ (Stand Oktober 2019)
[23] Quelle: https://codex.wordpress.org/Function_Reference/get_queried_object (Stand 2019)
[24] Quelle: https://developer.wordpress.org/reference/functions/get_queried_object_id/ (Stand 2019)

Nun gab es das Problem, dass dieses immer die Kinder der derzeitigen Anfrage zurückgibt. Wenn keine Eltern-Query gefunden wurde, wird die WordPress Query als Eltern gesehen: ein Echo gibt viele Array-Parameter[25] der obersten Taxonomie *Downloads* zurück. Jedoch, gemäß des Soll-Zustands muss bei keiner Auswahl immer die Taxonomie *Hersteller* erscheinen. Ein Top Level (Hersteller) Auswahl soll die nicht hierarchische Unterktegorien[26] des Hersteller-Querys liefern. Außerdem sollen sich die Dropdown-Boxen an die Auswahl 'erinnern' und ausgeben.
Die Ausgabe wurde zuerst als Test außer dem Dropdown Menü geschrieben und getestet, danach das Dropdown-Menü implementiert (Auswahl von Hersteller gemäß Soll-Zustandes bleibt stehen in Dropdown-Menü). Es wird eine Schleife benötigt, die der Länge von Dropdown1 durchläuft, und den Wert ausgibt:

```
26. for(var i=0; i < dropdown1.length; i++){document.write(dropdown1.options[i].value);}
```

Mit folgender Ausgabe bei Testauswahl des Herstellers Jablotron:

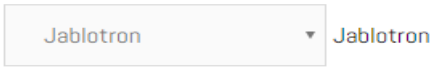


Abbildung 6:Testauswahl des Herstellers Jablotron

Jetzt muss der Wert *i* der Schleife mit dem Namen der derzeitigen Anfrage verglichen werden:

```
27. <?php echo $wp_query->get_queried_object()->name;?>
28. if(dropdown1.options[i].value=="{  dropdown1.selectedIndex = i;}
```

Dies hat jedoch nicht funktioniert. Eine Ausgabe hat den Fehler gezeigt: *name* hat großgeschriebenen Wert, Index *i* einen kleingeschrieben. Der Grund ist, das PHP eine objektorientierte Sprache ist, die die WordPress-Funktionen, -Objekte, und -Daten in der DB regelt, und evtl. die Webseiten aufruft. So muss der i-te Wert der Schleife mit dem *Slug*[27] der derzeitigen Abfrage der derzeitigen Seite verglichen werden. Jede Kategorie und Term der Suchtaxonomie hat einen kleingeschriebenen Slug, das vom *get_queried_object* in der Funktion get_queried_object()[28] geliefert wird. Erst wenn es zwischen dem zwei Objekten einen Match gibt, wird das *selectedIndex* Parameter auf Wert *i* gesetzt:

```
29. if(dropdown1.options[i].value=="
30. <?php echo $wp_query->get_queried_object()->slug;?>")
31.   {dropdown1.selectedIndex = i;}
```

Durch eindeutige Identifizierung von Top-Kategorien und Übergabe als Array-Werte in die Funktion, läuft das erste Dropdown und ruft die entsprechenden Seiten im Browser auf:

```
32. $top_categories = get_categories(array('taxonomy'=>'produktgruppe','parent'=>0,));
33. $select ="<select name='hersteller_select' id='hersteller_select' class='postform'>n";
34. $select.= "<option value='-1'>Hersteller auswählen</option>n";  …
```

Für die zweite Dropdown muss der Code modifiziert werden. Dabei hilft die WordPress-Funktion get_queried_object_id()[29]. Es wird nur die Kategorien ausgewählt, welche die eindeutig identifizierte WordPress-Query als Eltern haben. So kann nach der ersten Suche eine zweite Suche nach nicht hierarchischen Kind-kategorien ausgeführt werden.

```
35. $child_categories=get_categories(array('taxonomy'=>'produktgruppe',
36.                                    'parent'=>$wp_query->get_queried_object_id(),)
37. $select = "<select name='produkttyp_select' id='produkttyp_select' class='postform'>;
38. $select.= "<option value='-1'>Produktgruppe auswählen</option>n";  …
```

Die Dropdown Boxen haben Menüs, die mit den entsprechenden Taxonomie-Elementen (Kategorien und Tags) befüllt werden. Da alle Kategorien durch die Funktionen get_queried_object() oder get_queried_object_id() identifiziert worden sind, wird folgendes benötigt:

---

[25] Key-Value Paare
[26] Ober- und Unterkategorien sind in der Dokumentation Eltern bzw. Kinder genannt.
[27] Der *Slug* ist eine leserliche und für eine URL gültige Form des Titels eines Beitrags oder einer Seite. Der *Slug* wird auch Titelform genannt. Die Generierung des Permalinks für einen Beitrag oder eine Seite ist die bekannteste Verwendung dieser Funktion. Quelle: https://wp-bibel.de/glossar/slug/ (Stand 2019)
[28] Diese Funktion gibt ein Objekt zurück. Quelle: https://codex.wordpress.org/Function_Reference/get_queried_object (Stand 2019)
[29] https://wordpress.stackexchange.com/questions/18652/get-category-id-inside-category-template (Stand Oktober 2019)

1. Eine Funktion zum Befüllen des Dropdown1 Menüs mit der WordPress-Datenbank: ***get_ids_for menu1()***. Durch die Funktion array.push() wird das Array mit den Kategorien und die dazugehörigen Terms befüllt.
2. Eine Funktion zum Vergleich der Auswahl in Menü1: ***get_menu1_selection_id()***: ruft Funktion1 get_ids_for_menu1() auf, überprüft ob Nutzeranfrage zur Ober- oder Unterkategorie gehören und bei Auswahl von Kindern werden die Eltern, bei Auswahl von Eltern-Dummy-Daten[30] zurückgegeben. Die von der Funktion zurückgegebenen ID greift auf die Funktion get_menu2_categories zu, und listet die zur Eltern-Kategorie gehörende Produktkategorien auf.
3. Eine Funktion zum Befüllen des Dropdown2 Menüs mit der WordPress-Datenbank: ***get_menu2_categories()***: gibt bei einer Oberanfrage ein Array der Kinder zurück, und bei Unterabfrage ein Array von Siblings. Wenn es keine weitere Kindkategorien gibt, dann sprechen wir über Kindkategorie, weil die Kindkategorien nicht hierarchisch sind, sowie keine weiteren Kinder haben.
4. Die Funktion ***get_menu1_selection_slug()*** ist eine Hilfsfunktion, die durch den Aufruf von get_queried_object() und get_ids_for_menu1() überprüft, ob die Anfrage Elternkategorien hat: Wenn ja, wird Slug der Oberkategorie zurückgegeben, sonst Slug von selbst.
5. Eine Funktion, die das Slug des Child-level Querys liefert: ***get_menu2_selection_slug()***. Überprüft durch den Aufruf von get_queried_object() und get_ids_for_menu1(), ob die Kategorie Kind ist. Wenn ja, wird der Slug der Kind-Kategorie zurückgegeben.
   - Eventlistener Funktion ***onHerstellerChange()***: Rückgabe entsprechender Arraywert als URL für JS
   - Eventlistener Funktion ***onProdukttyphange()***: Rückgabe entsprechender Arraywert als URL für JS

Die von mir erstellte Hauptfunktionen sind in der Tabelle 6: Funktionsübersicht mit ihren Rückgaben sind im Anhang Seite 20 zusammenfasst.

4.8. Layout Gestaltung von selbst erstellten Elementen (CSS)

Für das Einbinden meiner eigenen CSS wurde das von WordPress-Backend Menü einfach erreichbare Customizer-Feature benutzt. Dies ermöglichte, dass ich meine Anpassungen sofort im Frontend sehen konnte. Der Vorteil dieses Vorgehens war unter anderem, dass es mit dem Feature der Syntaxfehler-Erkennung die Arbeit erleichterte. Zu der selbst erstellten Login-Logik habe ich die Stylesheets in den entsprechenden Ordner des WordPress-Verzeichnisses ausgelagert. WordPress benutzt auch Bootstrap für die Gestaltung von Taxonomien. Hierzu genügte wenig Einarbeitung[31].

```
.roll-button:hover{
background-color: #f17f23;
}

/*
Name linken Buttons für indiv. einstellungen:
*/
.roll-button.button-slider{
 background-image: url(./wp-content/themes/sydney-child-files/alarmimages/ItNetButton_noshade_InLevel-Pressed-LowQ.png);
}

/*Name rechten Buttons für indiv. einstellungen: */
.roll-button.right-border{
 background-image: url(./wp-content/themes/sydney-child-files/alarmimages/AlarmButton_noshade_InLevel.png);
}


/*
Responsive Einstellungen
für buttons
*/
@media only screen and (max-width: 767px){
.roll-button {
    padding: 5px 15px;
  margin:3px;}
}
```

Abbildung 7: WP Customizer Snippet

4.9. Testen und Fehlerbehebung

Bei der Erstellung dieses Projektes wurde fortlaufend getestet, da Testen schon zu dem gewählten Prozessmodell gehört. Mein Testverfahren wurde in der Regel folgendermaßen durchgeführt:

1. Bei der Entwicklung habe ich den Quellcode durch ständiges Sichten auf Fehler überprüft. Das kann als Whitebox-Test angesehen werden.
2. Die einzelnen Komponenten und Funktionen wurden mit selbst erstellten Dummies (Testumgebung der Prototypen) jeweils auf regelrechte Funktion getestet. Das könnte als Unit

[30] Um die Rückgabe weiterer WordPress Oberkategorien zu vermeiden, da WordPress auch von hierarchischen Kategorien besteht.

[31] https://www.w3schools.com/bootstrap/ (Stand Oktober 2019)

Tests angesehen werden. Wenn es zu Fehlern kam, habe ich in dem den Code den Fehler gesucht (Punkt 1)

3. Mit dem vom Projektgeber zur Verfügung gestellten Ressourcen wurde die gesamte Funktionalität mit Testfällen in der Testumgebung getestet. Das kann als System-Test angesehen werden.
4. User-Akzeptenztests (UAT) mit Projektgeber als 'User', am Ende jeder Teilaufgabe.
5. Fehlerbehebung, Anpassungen, widerholte UAT falls benötigt.
6. Blackbox/QA: bei Übergabe durchgeführt. Weitere Tests sind notwendig nach voller Integration der außer dem Projekt stehenden Komponenten.

4.10. Anpassungen

Sowohl die Programmierlogik als auch das Layout und Design, inkl. Farben, Fotos und Logos wurden laut Kundenwunsch angepasst. Bei gestalterischen Punkten, womit ich nicht ganz einverstanden war, musste ich das *Big Picture* im Auge behalten: Die Webseite gehört dem Projektgeber, und dessen Wünsche gilt es zu berücksichtigen.

4.11. Integration/Migration

Die Migration der WP- und Backend-Datenbank erfolgte als Hochladen des Projekts vom Localhost in die Cloud mit dem Migrationstool des Projektgebers[32]. Die volle Integration ist bei der Abgabe dieser Projektarbeit nicht durchgeführt worden[33].

5. Übergabe und Auswertung

In dieser Phase wurde überprüft, ob die geplanten Ressourcen bzw. das Vorgehen in der Planungsphase ausreichend bemessen waren.

5.1. Projektübergabe

Bei der Übergabe waren alle Beteiligten anwesend. Der gesamte Quellcode sowie die ausführlich beschriebenen Schritte der Programmierlogik und andere Instruktionen sind dem Projektgeber zur Verfügung gestellt worden. Wie im inhaltlichen Soll-Ist-Vergleich (siehe: Abschnitt 5.2 Erstellung des inhaltlichen Soll-Ist-Vergleiches) beschrieben, wurde die Migration in die Liveumgebung verschoben. Das Projektergebnis (siehe Abschnitt 5.2 Erstellung des inhaltlichen Soll-Ist-Vergleiches) wurde mit dem geplanten Soll-Zustand verglichen. Bis auf die ausstehende Migration entsprach das Ergebnis den Erwartungen und der Projektgeber beendete das Projekt als "erfolgreich durchgeführt".

5.2. Erstellung des inhaltlichen Soll-Ist-Vergleiches

Die in der Planungsphase gesetzten Projektziele wurden mit dem erarbeiteten Ist-Zustandes verglichen, und überprüft ob Sie erfolgreich realisiert worden sind. Die Migration in die Produktivumgebung hat bisher nicht stattgefunden. Zwar wurde das Projekt erfolgreich übergeben, jedoch sind Teile des Mediengestalters sowie der Erwerb von weiteren benötigten Komponenten noch nicht durchgeführt worden. Wenn diese nachgeliefert werden, müssten diese Teile eingepflegt werden. Nach einem erneuten Test könnte dann eine Migration erfolgen. Aus diesem Grund ist geplant, wenn die Migration nicht zwischenzeitlich stattgefunden hat, das Arbeitsergebnis der IHK-Prüfungskommission am Tag der mündlichen Prüfung live in der Testumgebung zu zeigen.

Projektziel	Erledigt?
Installation von XAMPP, Notepad++, und PhpMyAdmin	✓
Analyse der Daten und Anforderungen des Projektgebers und deren Kundschaft (IST)	✓
Erstellung eines Modells und einer Sitemap (SOLL)	✓
Import der derzeitigen Webseiten-Daten auf die lokale Testumgebung	✓
Erstellung der responsiven Webseite in WordPress (Webseiten, Homepage)	✓
Erstellung einer Datenbank mit zugehöriger Tabelle	✓
Erstellung des Child-Themes	✓

[32] Wurde wegen Zugriffkontrolle vom Projektgeber selber durchgeführt, ist deswegen nicht mehr Teil dieses Projekts.
[33] Siehe 5.5.3. Erstellung des inhaltlichen Soll-Ist-Vergleiches

Entwicklung eines Login-Bereichs mit einem PHP nativen Verschlüsselungsalgorithmus	✓
Programmierung von Custom Post Type (CPT) und dazugehörigem Plugin für das Hochladen, Speichern, und Herunterladen von Gebrauchsanweisungen in PDF-Format	✓
Programmierung von Suchtaxonomien – hierarchisch und nicht hierarchisch	✓
Programmierung von Dropdown-Box Logik für Suchtaxonomien	✓
Sessions programmieren mit Menüprüfung	✓
Migration von der Testumgebung (Localhost) in die Produktivumgebung – teilweise erledigt.	✓✗

Tabelle 3: Inhaltlicher Soll-Ist-Vergleich

5.3. Erstellung des Soll-Ist-Vergleiches für die eingesetzten Sachmittel

Die geplanten Sachmittel-Ressourcen waren - qualitativ und quantitativ - ausreichend bemessen. Daher muss auch der angenommene Platzkostenbetrag nicht in der Kostenauswertung angepasst werden.

5.4. Erstellung des zeitlichen Soll-Ist-Vergleiches

In dem Zeit und Ablaufplan (siehe Anhang 6.1: Zeit und Ablaufplan, Seite 18) wurden die durchgeführten Ist-Zeiten mitgeschrieben. Für die folgenden Punkte ergaben sich zeitliche Abweichungen zu den in der in Planungsphase veranschlagten Zeiten:

- Die Programmierung der Custom Post Type (-3 Stunden) und der Suchtaxonomie für CPT (-5 Stunden) war eine reine Fleißarbeit, und konnte schneller bearbeitet werden als geplant.
- Implementierung der Dropdown-Box Logik (+6 Stunde): Das fachliche Durchdringen dieser aufwendigen Thematik dauerte erheblich länger als angenommen. Jedoch empfinde ich die dadurch erworbene praktische Erfahrung als 'priceless', d.h. in monetären Mitteln nicht zu definieren. Daher sind die 6 Stunden investierte Zeit in mein fachliches Wissen als positiv zu bewerten.
- Anpassungen (+ 3 Stunden): Es ist anzumerken, dass der zeitliche Ablauf ohne Erfahrung schwer zu planen ist. So zieht zum Beispiel ein Bearbeitungsschritt oft auch eine inhaltliche Änderung in einem anderen Projektteil nach sich. Auch der zeitliche Aufwand von sich ändernden Detail-Wünsche sind oft nicht planbar.
- Erstellung der Projektdokumentation (+2 Stunden): Dieser Schritt hat mehr Zeit in Anspruch genommen als die von der IHK vorgegebene Zeit. Diese Zeitvorgabe steht im Kontrast zu der geforderten Dokumentations-Qualität meiner Ausbilder und ist nicht realistisch durchführbar gewesen.
- Entfall der Migration (-0,5 Stunden): Die Migration war mit 0,5 Stunden geplant. Wäre diese nicht entfallen, würde ich aus meiner jetzigen fachlichen Sicht min. 2 bis 3 Stunden, ohne des notwendigen Testes, veranschlagen.

Insgesamt betrachtet glichen sich die Mehr- und Minderzeiten fast gegeneinander aus. Die zeitliche Verzögerung um 2,5 Stunden fiel gemäß Projektgeber nicht ins Gewicht und ist zu vernachlässigen. Jedoch hat diese Stundenüberschreitung Auswirkung auf die Kostenplanung.

5.5. Erstellung des Soll-Ist-Vergleiches für die Kosten

Durch die zeitlichen entstandenen Verschiebungen muss die Kostenplanung mit den tatsächlich entstandenen Kosten verglichen werden.

Personalkosten	Kosten pro Stunde	Geplante Zeit (in Mitarbeiter-Stunde)	Geplante Gesamtkosten	Durchgeführte Zeit (in Mitarbeiter-Stunden)	entstandene Gesamtkosten
AZUBI Projektdurchführung	16,82 €	70	1177,40 €	72,5	1.219,45 €
Fr. Schumacher für Meilenstein-Gespräche[34]	20,45 €	10	204,50 €	10	204,50 €
Systemintegrator für Ressourcenbeschaffung	20,45 €	2	40,90 €	2	40,90 €
2 Mitarbeiter Fr Schumacher, Systemintegrator für Projekt-Abnahme	20,45 €	2	40,90 €	2	40,90 €
Gesamt			**1.463,70€**		**1.505.75 €**

Tabelle 4: Personalkostenaufstellung

[34] Geschätzter zeitlicher Aufwand der geplanten regelmäßigen Besprechungen

Werden alle Kosten zusammengeführt, ergeben sich Projektgesamtkosten von 2.205,75 €.

Projektkosten	Geplante Gesamtkosten	entstandene Gesamtkosten
Platzkosten (10,00 € x 70h)	700,00 €	700 €
Personalkosten	1.463,70 €	1.505,75 €
Gesamt	**2.163,70 €**	**2.205,75 €**

Tabelle 5:Gesamtkosten

So ergäbe sich eine tatsächliche Projektkostenerhöhung um 42,05 €. In Anbetracht der unveränderten Amortisierungsdauer von 60 Monaten läge das Projekt immer noch in einem durchführbaren Rahmen.

5.6. Technisches und persönliches Fazit

Anwendungsentwicklung gründet sich auf kreative Problemlösungen. Aus den zahlreichen Fehlern und Probleme nenne ich folgend drei Beispiele hierfür:

- **Problem von PDF Dateigröße**
  Bei der Anwendung fremder Plugins ist das Problem aufgetaut, dass angehängte PDF Dateien in ihren WP-Ordner mit neuen, nummerierten und unternummerierten Dateinamen dupliziert werden (siehe Abbildung 11: PDF Duplizierung). Dies hat mich u.a. zur Änderung der Make or Buy Entscheidung geführt, mein eigenes Plugin zu diesem einzelnen Zweck zu programmieren (siehe 6.4. 6.4Dokumentation von Custom Metabox). Mein fertiges Plugin hatte jedoch die gleiche Verhaltensweise. Im WP- Communities wurde es u.a. über Bugs im WP-DB gesprochen, und Vorschläge zum Einfügen von Code in verschiedene Dateien im WP-Verzeichnis lieferten kein Ergebnis. Die Hilfe kam eventuell aus meiner Testumgebung: mit einer 'Dummy'-Datei lief sie, und nach Isolierung der Datei und ihre Eigenschaften ist klargeworden, dass die Duplizierung von einem Überschritt der Dateigröße verursacht worden war! WordPress bietet viele Möglichkeiten an, die Dateigröße zu ändern[35]. Dieses Arbeitsergebnis möchte ich nach der Prüfungsphase mit Community-Foren teilen.
- **Logikproblem des Login-systems**
  Während der Entwicklungsphase 4.3. Implementierung des Kundenlogins (Kennwort-Hash) hat die UML Modellierung viel geholfen. Die Erstellung eines Use-Case Diagramms hat die eindeutige Schwachstelle meiner bisherigen Denkweise gezeigt: unverschlüsselte und verschlüsselte Daten dürfen nicht im gleichen Speicherort gelagert werden (siehe Anhang 6.2., Abbildung 9: Modell von Hashing Logik 1. Da der Projektgeber kein Fan von Datenbanken ist, und über die zukünftige Datenspeicherung noch keinen Plan vorhanden war, habe ich nach einer Recherche die Entscheidung getroffen, dass der Hashwert während der Erzeugung (INSERT oder UPDATE) das Kennworts erstellt werden soll. Dafür dient die Funktion $update_query. Für die Nutzerauthentifizierung bei Kennwort-Eingabe und das Vergleich mit den DB-Werten dient die Funktion password_ verify() (siehe 6.2 Abbildung 10. Modell von Hashing Logik 2.).
- **Kennwort Hashing Datenbank-Problem**
  Erst nach der Projektübergabe habe ich bemerkt, dass die Verschlüsselung in meinem Test-DB nur nach einmaligem Durchlauf funktioniert hat. Die Lösung erarbeitete ich mir als ich mich darauf mit Verschlüsselungs-Algorithmen und DB beschäftigte. Der Nullwert in einer DB ist auch ein Wert, und wird auch mitverschlüsselt und mitüberschrieben. Die Lösung war auch viel einfacher als ich dachte: mit einer *if(not empty){}* Anweisung in der richtigen Zeile war alles erledigt (Siehe 6.5., I. b. Zeile 17).

Bei Durchführung dieses Projekts hat mein Erlebnis des Flow [36] zum Erfolg beigetragen. Jedoch erscheint es Rückblickend ein Paradox zu sein. Einerseits den eigenen Flow auszunutzen und anderseits meine Aufgabe strukturiert durchzuführen, um persönliche Überforderungszustände zu vermeiden. Neben den in den personellen Schnittstellen benannten Personen bin ich meinen BFW-Projektbetreuer Herrn Dettling dankbar für den stetigen Hinweis auf eine strukturierte Arbeitsweise zu achten. Egal, wie spannend das Thema ist, Erholungszeiten tragen genauso zu einem Projekt-erfolg zu, wie die Arbeitszeiten.

[35] https://www.wpbeginner.com/wp-tutorials/how-to-increase-the-maximum-file-upload-size-in-wordpress/ (Stand Okt. 2019)
[36] Nach Mihaly Csikszentmihalyi, https://de.wikipedia.org/wiki/Flow_(Psychologie) (Stand November 2019)

6. Anhang

In dem Anhang befinden sich folgende Inhalte:

- Zeit- und Ablaufplanung (Gantt-Diagramm)
- Volle Dokumentation von Custom Metabox
- Abbildungen
- Tabellenverzeichnis
- Quelltexte

6.1. Zeit und Ablaufplan

Phase		Bezeichnung Vorgang	Soll-Stunden	Ist-Stunden	Differenz		16.9	17.9	18.9	19.9	20.9	21.9	22.9	23.9	24.9	25.9	26.9	27.9	28.9	29.9	30.9	1.10	2.10	3.10	4.10	5.10	6.10	7.10	8.10	9.10	10.10
1. Planungsphase																															
	1.	Initiale Projektbesprechung und Zielsetzung	1	0,5	-0,5	geplant	1																								
						durchgeführt	0,5																								
	2	Analyse des Ist-Zustandes/Ausgangssituation	1	1	0,0	geplant	1																								
						durchgeführt	1																								
	3	Erstellung des Soll-Konzepts	1	1	0,0	geplant	1																								
						durchgeführt	1																								
	4	Ableitung des Projektzieles, -Ablaufes, -Aufgaben	1	3	2,0	geplant	1																								
						durchgeführt	1	2																							
2. Durchführungphase																															
	12	Wordpress-Update auf neueste Version	1	1	0,0	geplant		1																							
						durchgeführt			0,5			0,5																			
	13	Strukturelle Erstellung der Test-Webseite	5	4	-1,0	geplant		1	0,5							3,5															
						durchgeführt			0,5							2,5							0,5			0,5					
	14	Umsetzung von Änderungswünschen	2	4	2,0	geplant			0,5							0,5							0,5			0,5					
						durchgeführt			0,5					0,5							1		1			1					
	15	Erstellung der Test-Datenbank von Stammkunden	1	1	0,0	geplant			1																						
						durchgeführt								1																	
	16	Programmierung von Login und Password-Hash Verschlüsselung	8	6	-2,0	geplant			1			7																			
						durchgeführt			3			2		1																	
	17	Programmierung von Sessions mit Menüpfüfung	7	7	0,0	geplant							1	2	4																
						durchgeführt								1	4	2															
	18	Einbetten von Google-Funktionen, Maps und Recapcha	1	0,5	-0,5	geplant								1																	
						durchgeführt								0,5																	
	19	Programmierung von Custom Post Type	4	1	-3,0	geplant								4																	
						durchgeführt								0,5	0,5																
	20	Programmierung von Custom Anhang (.pdf) zum Custom Post Type, und Anhang lösbar machen	6	4	-2,0	geplant										6															
						durchgeführt													4												
	21	Programmierung von Suchtaxonomien für CPT	8	3	-5,0	geplant										1			7												
						durchgeführt													3												
	22	Programmierung von Dropdown-Box Logik für Suchtaxonomien	8	14	6,0	geplant														8	1										
						durchgeführt										2				8	4										
	23	Layout Gestaltung von selbst programmierten Funktionen (CSS)	1	3	2,0	geplant										1															
						durchgeführt			1						0,5					0,5						0,5		0,5			
	24	Testen und Fehlerbehebung	1	1	0,0	geplant			0,2							0,2							0,2			0,2				0,2	
						durchgeführt			0,2							0,2							0,2			0,2				0,2	
	25	Anpassungen	1	5	4,0	geplant			0,2							0,2							0,2			0,2		0,2			
						durchgeführt			0,5							0,5							1			1		2			
	26	Integration/Migration	0,5	0	-0,5	geplant																					0,5				
						durchgeführt																									
	27	Qualitäts-Sicherung/Testen	1	0	-1,0	geplant																							1		
						durchgeführt																							0		
3. Übergabe-, Auswertungs-, Dokumentationphasephase																															
	24	Erstellung des Soll-Ist Vergleich (Zeitlich, Ressource, Inhaltlich)	1	1	0,0	geplant																				1					
						durchgeführt																				1					
	25	Abnahme/ Übergabe	0,5	0,5	0,0	geplant																				0,5					
						durchgeführt																				0,5					
	26	Erstellung der persönlichen Fazit/Aublick	1	0,5	-0,5	geplant																				1					
						durchgeführt																				0,5					
	27	Erstellung der Dokumentation	8	10	2,0	geplant																						2	2	4	
						durchgeführt										1							1						4	4	
Summen:			70	72	2																										

6.2. Abbildungen

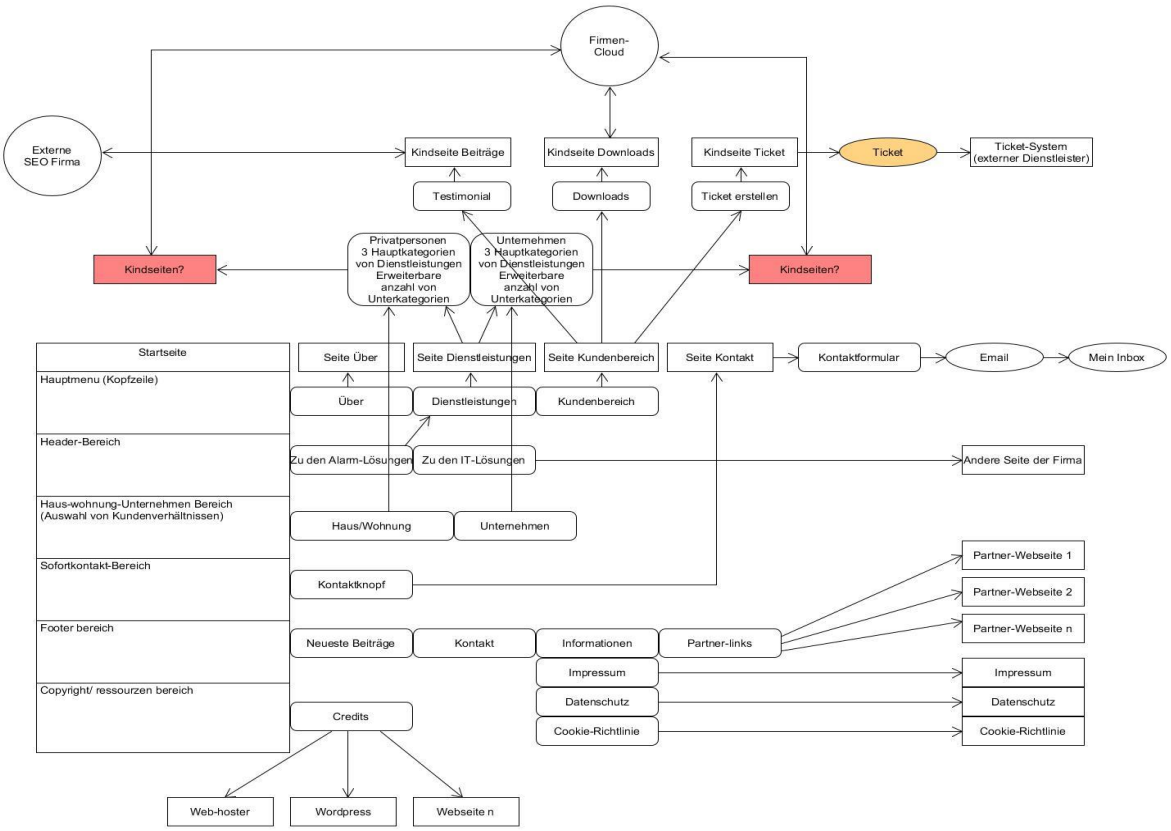


Abbildung 8: Sitemap-Modell

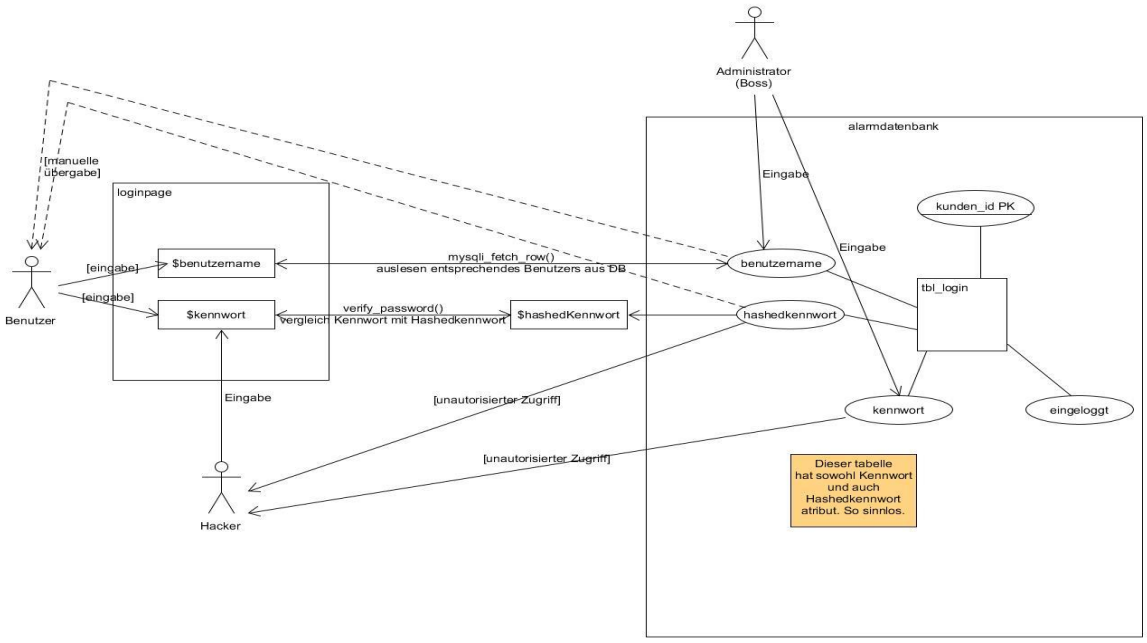


Abbildung 9: Modell von Hashing Logik 1

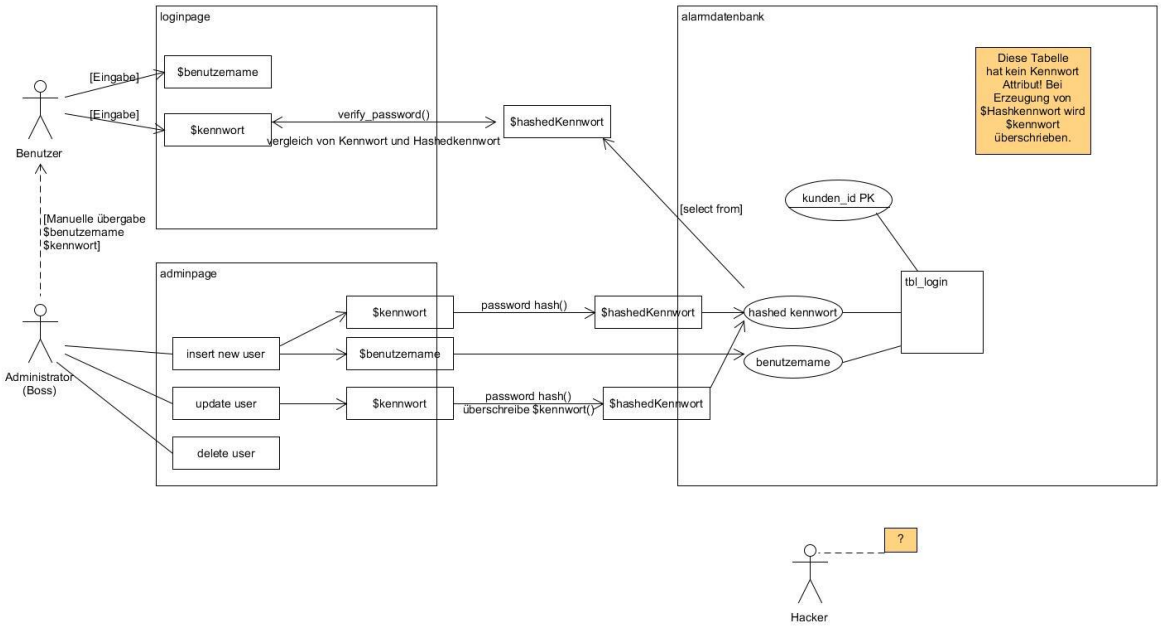


Abbildung 10: Modell von Hashing Logik 2.

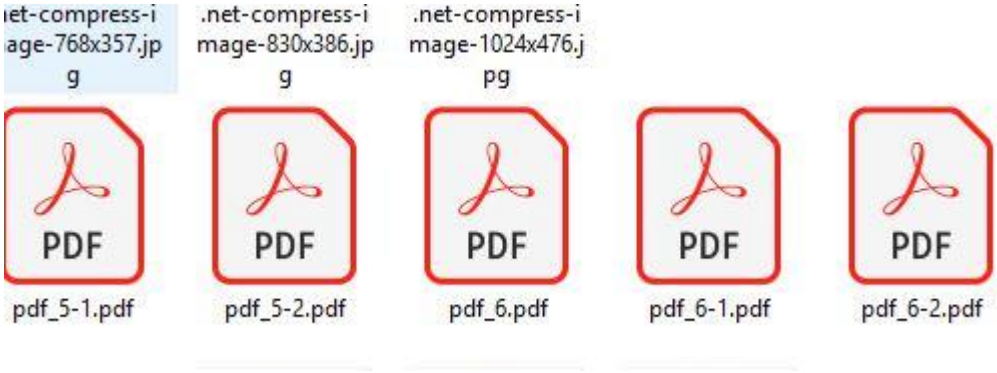


Abbildung 11: PDF Duplizierung

Anfrage Typ / Funktion	Keine Auswahl	Top-level (Hersteller) Auswahl	Child-Level (Produkttyp) Auswahl
get_ids_for menu1 [alle Ids zum Befüllen von Menü1]	Gibt nichts zurück	Gibt Top-Level Kategorien (Hersteller) zurück	Gibt Child-Level Kategorien (Produkttyp) zurück
get_menu1_selection_id() [was aus Menu1 ausgewählt wurde, ruft get_ids_for menu1 auf]	Bei keiner Auswahl gibt -8 (Dummy-Datum) zurückgegeben	Gibt termID Query zurück	Gibt termID von Parent zurück
get_menu2_categories() [IDs zum Befüllen von Menu2]	Gibt leeres Array zurück	Gibt Kinder zurück	Gibt Siblings zurück
get_menu2_selection_slug() [was in Menu2 ausgewählt wurde]	Gibt 'noparentandnosearch' zurück, Seite bleibt stehen	Gibt nichts zurück, '' (leeres String)	Gibt Slug von Child-Level get_queried_object zurück
get_menu1_selection_slug() [ruft get_queried_object() WP-funktion auf, und gibt Slug von dieser Funktion-Rückgabe zurück]	Gibt nichts zurück	Gibt Query Slug zurück	Gibt Parent Query Slug zurück

Tabelle 6: Funktionsübersicht

6.3. Amortisationsdauer – detaillierte Tabelle

Woche	Kosten(Projekt)	Kosten (Ist-Zustand)	33	7.022,62	6.073,65	67	12.028,78	12.331,35
0	2.163,70	0,00	34	7.169,86	6.257,70	68	12.176,02	12.515,40
1	2.310,94	184,05	35	7.317,10	6.441,75	69	12.323,26	12.699,45
2	2.458,18	368,10	36	7.464,34	6.625,80	70	12.470,50	12.883,50
3	2.605,42	552,15	37	7.611,58	6.809,85	71	12.617,74	13.067,55
4	2.752,66	736,20	38	7.758,82	6.993,90	72	12.764,98	13.251,60
5	2.899,90	920,25	39	7.906,06	7.177,95	73	12.912,22	13.435,65
6	3.047,14	1.104,30	40	8.053,30	7.362,00	74	13.059,46	13.619,70
7	3.194,38	1.288,35	41	8.200,54	7.546,05	75	13.206,70	13.803,75
8	3.341,62	1.472,40	42	8.347,78	7.730,10	76	13.353,94	13.987,80
9	3.488,86	1.656,45	43	8.495,02	7.914,15	77	13.501,18	14.171,85
10	3.636,10	1.840,50	44	8.642,26	8.098,20	78	13.648,42	14.355,90
11	3.783,34	2.024,55	45	8.789,50	8.282,25	79	13.795,66	14.539,95
12	3.930,58	2.208,60	46	8.936,74	8.466,30	80	13.942,90	14.724,00
13	4.077,82	2.392,65	47	9.083,98	8.650,35	81	14.090,14	14.908,05
14	4.225,06	2.576,70	48	9.231,22	8.834,40	82	14.237,38	15.092,10
15	4.372,30	2.760,75	49	9.378,46	9.018,45	83	14.384,62	15.276,15
16	4.519,54	2.944,80	50	9.525,70	9.202,50	84	14.531,86	15.460,20
17	4.666,78	3.128,85	51	9.672,94	9.386,55	85	14.679,10	15.644,25
18	4.814,02	3.312,90	52	9.820,18	9.570,60	86	14.826,34	15.828,30
19	4.961,26	3.496,95	53	9.967,42	9.754,65	87	14.973,58	16.012,35
20	5.108,50	3.681,00	54	10.114,66	9.938,70	88	15.120,82	16.196,40
21	5.255,74	3.865,05	55	10.261,90	10.122,75	89	15.268,06	16.380,45
22	5.402,98	4.049,10	56	10.409,14	10.306,80	90	15.415,30	16.564,50
23	5.550,22	4.233,15	57	10.556,38	10.490,85	91	15.562,54	16.748,55
24	5.697,46	4.417,20	58	10.703,62	10.674,90	92	15.709,78	16.932,60
25	5.844,70	4.601,25	59	10.850,86	10.858,95	93	15.857,02	17.116,65
26	5.991,94	4.785,30	**60**	**10.998,10**	**11.043,00**	94	16.004,26	17.300,70
27	6.139,18	4.969,35	61	11.145,34	11.227,05	95	16.151,50	17.484,75
28	6.286,42	5.153,40	62	11.292,58	11.411,10	96	16.298,74	17.668,80
29	6.433,66	5.337,45	63	11.439,82	11.595,15	97	16.445,98	17.852,85
30	6.580,90	5.521,50	64	11.587,06	11.779,20	98	16.593,22	18.036,90
31	6.728,14	5.705,55	65	11.734,30	11.963,25	99	16.740,46	18.220,95
32	6.875,38	5.889,60	66	11.881,54	12.147,30	100	16.887,70	18.405,00

Tabelle 7: Amortisation-Entwicklung

6.4. Dokumentation von Custom Metabox

Ein Metabox ermöglicht das Eintragen bzw. das Hochladen von Daten in das WordPress-Backend. Hierzu gibt es die Funktion add_custom_meta_boxes()[37]. Diese Funktion ist eine in WordPress eingebaute Funktion für das Einbinden von neuen eigenen Metaboxen.

```
39. function add_custom_meta_boxes() {
40. add_meta_box(
41. 'wp_custom_attachment',
42. 'Custom Attachment',
43. 'wp_custom_attachment',//WP Callback Funktion
44. 'downloads',
45. 'side' );
```

Durch einen Aufruf von bestimmten Funktionen ermöglichen das Hinzufügen von Action Hooks und das Einbinden selbst entwickelter Plugins[38]:

[37] https://developer.wordpress.org/plugins/metadata/custom-meta-boxes/ (Stand Oktober 2019)
[38] https://developer.wordpress.org/reference/hooks/ (Stand Oktober 2019)

```
add_action('add_meta_boxes', 'add_custom_meta_boxes');
```

Nun wird die WordPress Callback Funktion definiert. *Nonce* validiert, dass es nur auf dieser Seite hochgeladen werden kann:

```
wp_nonce_field(plugin_basename(__FILE__), 'wp_custom_attachment_nonce');
$html = '<p class="description">';
$html .= 'PDF hier hochladen.';
$html .= '</p>';
$html .= '<input type="file" id="wp_custom_attachment" name="wp_custom_attachment"
value="" size="25" />';
```

Auf die File-Informationen des Arrays zugreifen:

```
$doc = get_post_meta(get_the_ID(), 'wp_custom_attachment', true);//true heisst array
```

Input box erstellen und Datei URL als Wert von Textelement setzen:

```
$html .= '<input type="text" id="wp_custom_attachment_url"
name="wp_custom_attachment_url" value=" ' . $doc['url'] . '" size="30" />';
if(strlen(trim($doc['url'])) > 0) {
    $html .= '<a href="javascript:;" id="wp_custom_attachment_delete">' . __('Delete F
ile') . '</a>'; }
echo $html;
```

PDF speichern:

```
function save_custom_meta_data($id) {
```

Die Sicherheitsverifizierung mit *nonce* verhindert das Auto-save und schließt nicht autorisierte Benutzer aus:

```
if(!wp_verify_nonce($_POST['wp_custom_attachment_nonce'], plugin_basename(__FILE__))){
    return $id;}
if(defined('DOING_AUTOSAVE') && DOING_AUTOSAVE){
     return $id;}
if('page' == $_POST['post_type']) {
      if(!current_user_can('edit_page', $id)){
         return $id;}
      else {
        if(!current_user_can('edit_page', $id)){
            return $id;
         }
    }
```

Sicherstellen, dass das Datei-Array nicht leer ist:

```
if(!empty($_FILES['wp_custom_attachment']['name'])) {
```

Ein Array der unterstützten Dateitypen (PDF, JPG, PNG) erstellen

Problem hier war, dass wenn man nur PDF in das Array schreibt, dann wird auch nur dieser Dateityp unterstützt. Wenn nun eine JPG- oder PNG-Datei angehängt wird, verschwindet die PDF-Datei. Also wenn ein Dateityp genannt wird, dann wird dieser auch exklusiv unterstützt. Um dies zu vermeiden, müssen .PNG- und .JPG-Dateitypen auch benannt werden, obwohl diese in WordPress standardmäßig unterstützt werden. Die Angabe: 'application/pdf' ist der Media-Typ für PDF-Dateien (auch als MIME-Type[39] bekannt).

[39] https://www.iana.org/assignments/media-types/media-types.xhtml (Stand Oktober 2019)

```
73. $supported_types = array('application/pdf', 'image/jpg', 'image/png');
```

Nun muss überprüft werden, ob der Dateityp ein unterstützter Dateityp ist. Wenn ja, kann die Datei auf den Server kopiert werden. Wenn nicht, wird ein Fehler erzeugt.

```
74. $arr_file_type = wp_check_filetype(basename($_FILES['wp_custom_attachment']['name']));
75. $uploaded_type = $arr_file_type['type'];
76. if(in_array($uploaded_type, $supported_types)) {
```

Die Funktion wp_upload_bits kopiert eine Datei auf den Server:

```
77. $upload = wp_upload_bits($_FILES['wp_custom_attachment']['name'], null,
78.           file_get_contents ($_FILES['wp_custom_attachment']['tmp_name']));
```

Folgende Schritte werden benötigt, um eine PDF-Datei zu löschen. Voraussetzung hier ist, dass das URL Feld des Attachments in der WordPress DB befüllt ist, und das zugehörige Delete-flag im Array gesetzt ist. Dafür sorgen die folgenden Zeilen:

```
79. add_post_meta($id, 'wp_custom_attachment_url', $_POST['wp_custom_attachment_url']);
80. update_post_meta($id, 'wp_custom_attachment_url', $_POST['wp_custom_attachment_url']);
```

Die Funktion Get post meta($id, 'wp_custom_attachment', true) hat 3 Parameter. Wenn der dritte Parameter auf True gesetzt ist, bedeutet das, dass nur das erste Element eines Arrays zurückgegeben wird.

```
81. $doc = get_post_meta($id, 'wp_custom_attachment', true);
```

Auf die Post-array gespeicherten Werte (ULR Post) zugreifen und in Variable $delete_flag speichern:

```
82. $delete_flag = get_post_meta($id, 'wp_custom_attachment_url', true);
```

Gibt es ein Wert im URL des Arrays? Gibt es String in $delete_flag variable (ist Länge True?)

```
83. if(strlen(trim($doc['url'])) > 0 && strlen(trim($delete_flag)) == 0) {
```

Versuche, die Datei zu löschen – setze Wert zu NULL – sonst Fehlerausgabe:

```
84. if(unlink($doc['file'])) {
85.                 update_post_meta($id, 'wp_custom_attachment', null);
86.                 update_post_meta($id, 'wp_custom_attachment_url', '');
87. } else {
88.             wp_die('Feher beim Löschen.');
89.            }
```

Action hook der neuen Funktionen

```
90. add_action('save_post', 'save_custom_meta_data');
```

Mehrere Dateitypen akzeptiert werden können, + Funktion zum Einbinden von JS Datei und ihre Aktion Hooks:

```
91. function update_edit_form() {
92.     echo 'enctype="multipart/form-data"';
93. } // ende update_edit_form, hook:
94. add_action('post_edit_form_tag', 'update_edit_form');
95. function add_custom_attachment_script() {
```

```
96.     wp_register_script('custom-attachment-
    script', get_stylesheet_directory_uri() . '/js/custom_attachment.js');
97.     wp_enqueue_script('custom-attachment-script');
98. } //action hook
99. add_action('admin_enqueue_scripts', 'add_custom_attachment_script');
```

JS Datei, damit alles auch client-seitig gemacht werden kann (Nicht selbst erstellt):

```
1.  jQuery(function($) {
2.  //Checken ob ein custom_attachment_delete link existiert auf der Seite
3.  if($('a#wp_custom_attachment_delete').length === 1) {
4.  // Link da: Eventhandler Funktion muss den Nutzer-Klick handeln
5.  $('#wp_custom_attachment_delete').click(function(evt) {
6.  //Link soll Nutzer nicht aus Seite wegleiten: seine Default Funktion stoppen:
7.  evt.preventDefault();
8.  //Finde das Textelement, das den Pfad zur Datei speichert und Wert zu leerem String:
9.  $('#wp_custom_attachment_url').val('');
10. // Link mit hide() Funktion verstecken, damit Nutzer nicht mehr draufklicken kann:
11. $(this).hide();
12. });//ende funkt evt
13. } // ende if
14. });//ende funkt($)
```

6.5. Quelltexte

I. Code-Snippets von Login und Verschlüsselung

a. Dateiname: BB-Loginform - Eingabe Vergleich

```
1.  //SEHR WICHTIGE CODE DAMIT ES NICHT INS PEAR GEHT: MAGIC CONSTANT include_dir
2.  //erste'/' ist nicht root sondern gleich wie .. einmal nach oben im Filesystem
3.  include __DIR__ . "/../barb-code/hash_all_users.php";
4.  error_reporting(E_ALL|E_STRICT);
5.  //-------------verbindung datenbank:
6.  $daba_connect = mysqli_connect("localhost", "root", "", "alarmdatenbank");
7.  if($daba_connect){ //testen: kann ich den datenbank erreichen.
8.                      //print_r("yes") für Test;
9.   //============ Datensätze aus Tabelle Login auslesen =====
10. //hashed password auslesen, neue query tabelle: $_elect_hashedpw_query: 1 Spalte
11. $_elect_hashedpw_query = "SELECT hashedpassword FROM alarmdatenbank.login
12. WHERE benutzername = '$nutzername'";
13. $_delete_unhashed_pw = "UPDATE alarmdatenbank.login SET kennwort = NULL
14. WHERE benutzername = '$nutzername'";
15. mysqli_query($daba_connect, "SET NAMES UTF8");
16. //------Select von einzelbenutzer, WHERE benutzername = '$nutzername'
17. $_elect_hashedpw_for_user = mysqli_query($daba_connect, $_elect_hashedpw_query)or die(
    mysqli_error($daba_connect));
18. mysqli_query($daba_connect, $_delete_unhashed_pw)or die(mysqli_error($daba_connect));

19. //-----gibt an wieviele benutzer mit passenden benutzername in db sind.
20. $kundenzeilenr = mysqli_num_rows($_elect_hashedpw_for_user);
21. //print_r($kundenzeilenr) nur für test;
22. //if kunde ist in datenbank, kundenzeilennr ist 1. sonst 0
23. if($kundenzeilenr==1){
24.     //var $first_row_matching_user hat nur eine zeile:
25.     $first_row_matching_user=mysqli_fetch_row($_elect_hashedpw_for_user);
26. //Hier soll statt kennwort den hashed kennwort checken.
27. // eingebaute function matcht $kennwort mit hashed passwort in tbl login:
28. //0-te spalte $select_hashedpw_query ist hashed password:
29. if (password_verify($kennwort, $first_row_matching_user[0])){
30. //globalvar session hat assoc array, befüllt mit key userloggedin, wert true
31.         $_SESSION['userloggedin']=true;
```

```
        //damit nutzername in anderen seiten erreichbar, wird es teil der globalen session variable gemacht
        $_SESSION['nutzername']=$nutzername;
//TODO redirect to ticket-system!!! Login daten speichern
        echo "<br>";
        echo "Wilkommen, " . $nutzername . "!";//syntax concatination von strings.
        // todo set session login true
        sleep(1);
        echo "<script>";
        echo "document.location='".site_url().$login_dest."'";
        echo "</script>";
    }
else{ echo"<p>Kontaktieren Sie uns zum Zurücksetzen des Kennworts</p>";
    }
}
else{ echo"<p>Benutzer unbekannt. Clicken Sie <a href='".site_url()."/kontakt'>hier</a> um uns zu kontaktieren.</p>";
//todo redirect: kontakt
    }
}
else{
echo"<p>Fehler Verbindung zum Datenbank. Bitte probieren Sie noch einmal.</p>";
```

b. Dateiname: hash_all_users.php - Backend, Datenbank

```
//diese Funktion erzeugt aus Kennwort ein hashedpassword.
//argument $daba_connect wurde ins funkction reingepasst damit sie die datenbankverbindung erkennt.
function create_hashes_in_table($daba_connect){
//zurück: Strings, Select: nur 2 Spalten damit später mehr addiert werden können. User name index 0, pw index 1
$select_query='SELECT benutzername , kennwort FROM alarmdatenbank.login';
//Verbinde DB, select * users oder die und fehlermeldung
$select_result=mysqli_query($daba_connect, $select_query)or die(mysqli_error());
//loginresult ist nur zwei spalten.
$all_users=array(); //leeres array
//$loginresult ins array packen. Funk mysqli_fetch_row holt 1 Zeile aus loginresult array.
    while($row = mysqli_fetch_row($select_result)){
//while loop ausgabe true while es noch rows in $loginresult gibt. Sonst false.
        array_push($all_users,$row);
//array_push() php funktion: leeres array mit daten aus der tabelle befüllen. zu all_users array addieren wir $row
        $username = $row[0];//hilfsvariable für übersichtlichere tabelle
        $unhashed_password = $row[1];
        if(!emptyempty($unhashed_password)){ //Problem von 23 Oct!!!!!!!!!!!!!
//Tabelle hashed password updaten:
        $update_query="UPDATE alarmdatenbank.login
SET hashedpassword='". password_hash($unhashed_password,PASSWORD_DEFAULT)."'
WHERE benutzername='". $username ."'";
          } //fehlermeldung vermeiden: mysqli_query braucht 2 parameter
$update_result=mysqli_query($daba_connect, $update_query);
    }
}
//alle nutzerdaten durchlaufen und kennwort zu hashed ändern:
error_reporting(E_ALL|E_STRICT);
echo $kennwort . "";
//-------------verbindung datenbank:
$daba_connect = mysqli_connect("localhost", "root", "", "alarmdatenbank");
if($daba_connect){
    //Funktionsaufruf:
    create_hashes_in_table($daba_connect); }
```

II. Listing der Dropdown Logik:

a. Dateiname: Archive.php: selbst erstellte Archivdatei, die alle CPT listet und die benötigten Funktionen definiert. Am Ende Steht *The Loop* (Die Schleife), die WordPress selbst regelt.

```
<?php get_header(); ?>
<!--
DIES IST DAS EINZIGE ARCHIV DATEI DIE ALLE POSTS FÜR BESTIMMTE KATEGORIE LISTET
& GREIFT CONTENT-ARCHIVE.php ZU-->
<div class="row">
<div class="col-xs-12 col-sm-8">
<div id="primary" class="row text-left no-margin">
<!--class="content-area col-md-9 <?php echo esc_attr( $layout )?>;-->
<?php //aus downloads reinkopierte Snippet:
$login_dest='/login-downloads';
if(!$_SESSION['userloggedin']){
echo "<script>";
echo "document.location='".site_url().$login_dest."'";
echo "</script>";
}
else{
echo '<h5 style = "text-
align: right">'. $_SESSION['nutzername']. '<a href = http://localhost/dev-
alarmsol/logout/>  ' . '- Logout' .'</a></h5>';
}
?>
<?php if( have_posts() ): ?>
<header class="customposts-header">
<h1 class="customposts-title">Handbücher</h1>
</header>
<div id="dropdownboxen">
<?php
//Funktion 1 zum Befüllen von Menu1
function get_ids_for_menu1(){
$top_categories = get_categories(array('taxonomy'=>'produktgruppe','parent'=>0,
));
// liste von Slugs:
$categoryIdArray = [];
foreach ($top_categories as $category) {
                    array_push($categoryIdArray, $category->term_id);
                    }
                return $categoryIdArray;
                } //Ende Funktion 1
//Funktion 2 'lieferer' von Nutzerauswahl
function get_menu1_selection_id($wp_query){
        if(is_tax('produktgruppe', get_ids_for_menu1())){ //Aufruf Function 1.
 // 1. Überprüfen ob Nutzerauswahl ist top level category:
            return $wp_query->get_queried_object()->term_id;
            }
        else if(isset($wp_query->get_queried_object()->parent)){
            return get_term($wp_query->get_queried_object()->parent)->term_id;
            }
        else{
        return -8; //Dummy Wert
            }
    }  //Ende Funktion2
  //2. Überpfüfen, ob Query Eltern hat:
 //Function 4, Hilfsfunktion für Holen von Slug von Menu1
function get_menu1_selection_slug($wp_query){
        if(is_tax('produktgruppe', get_ids_for_menu1())){ //aufruf funktion1
        // 1. Überprüfen ob Nutzerauswahl ist top level category:
            return $wp_query->get_queried_object()->slug;
            }
        else if(isset($wp_query->get_queried_object()->parent)){
            return get_term($wp_query->get_queried_object()->parent)->slug;
```

```
57.                 }
58.              else{
59.              return 'noparentandnosearch';
60.              }
61.         }  //Ende Funktion 4
62.              //2. Überpfüfen, ob Query Eltern hat:
63.     //Funktion3, zum Befüllen des Kindmenüs:
64.     function get_menu2_categories($wp_query){
65.            if(!is_tax()){//is_tax überprüft ob query toplevel ist, quelle codex
66.                          return array();//leer-Array
67.            }  //ende if
68.     $child_categories=get_categories(array('taxonomy'=>'produktgruppe',
69.              'parent'=>$wp_query->get_queried_object_id(),)    );
70.            if(emptyempty($child_categories)){ //hat Keine Kinder: ist selbst Kind
71.         //Bei Kindkategorien überprüfen, ob Toplevel Kategorie gesetzt wurde
72.                    if (!empty(get_menu1_selection_id($wp_query))){
73.            $child_categories = get_categories(array('taxonomy'=>'produktgruppe',
                                    'parent'=>get_menu1_selection_id($wp_query)));
74.                    }  //ende if
75.            }  // ende if
76.     return $child_categories;
77.     }  //ende Funktion3
78.     // Funktion 5 für Ausgabe Slug in Menü2
79.     function get_menu2_selection_slug($wp_query){
80.            if(is_tax('produktgruppe', get_ids_for_menu1())){
81.     //Wenn top Level ausgewählt und 2. Level noch nicht, keine Ausgabe
82.                    return '';
83.                }
84.                //Wenn wir Eltern haben, wir sind Kind und gebe user Slug zurück
85.              else if(isset($wp_query->get_queried_object()->parent)){
86.                    return get_term($wp_query->get_queried_object())->slug;
87.                }
88.              else{
89.                return 'noparentandnosearch';
90.                }
91.         }    // ende Funktion 5
92.     //In ein Array umwandeln:
93.     $top_categories = get_categories(array('taxonomy'=>'produktgruppe',
94.                                                    'parent'=>0,));
95.     $select="<select name='hersteller_select'id='hersteller_select' class='postform'>;
96.     $select.= "<option value='1'>Hersteller auswählen</option>n"; //Wert1 noAuswahl
97.     foreach($top_categories as $category){
98.       if($category->count > 0){
99.       $select.= "<option value='".$category->slug."'>".$category->name."</option>";
100.                               }  //ende if
101.     }  //ende foreach
102.    $select.= "</select>";
103.    echo $select;
104.    ?>
105.    <script type="text/javascript"><!--
106.    var dropdown1 = document.getElementById("hersteller_select");
107.    for(var i=0; i < dropdown1.length; i++){
108.        //Aufruf funktion5, Rückgabe Slug
109.         dropdown1.selectedIndex = i;
110.                            }
111.    }
112.    function onHerstellerChange() {
113.            if (dropdown1.options[dropdown1.selectedIndex].value != -1){//kein Herst.AW
114.             location.href = "<?php echo home_url();?>
115.    /produktgruppe/"+dropdown1.options[dropdown1.selectedIndex].value+"/";
116.             }// Rückgabewert geht ins Browser
117.    }
118.    dropdown1.onchange = onHerstellerChange;//eventhandler ruft Funktion auf
119.    - -></script>
120.    <!--Zweites Dropdown-Box:-->
```

```
<?php
//Aus tax Produktruppe gebe zurück Queries die derzeitige Query als Eltern haben
$select = "<select name='produkttyp_select' id='produkttyp_select' class='postform
'>n";
$select.= "<option value='-1'>Produktgruppe auswählen</option>n";
$menu2_categories = get_menu2_categories($wp_query);
foreach($menu2_categories as $category){
       if($category->count > 0){
       $select.= "<option value='".$category->slug."'>".$category>name."</option>
        }  // ende if
}    // ende foreach
$select.= "</select>";
echo $select;
?>
<script type="text/javascript"><!--html ignoriert, nur JS führt aus:
var dropdown2 = document.getElementById("produkttyp_select");
for(var i=0; i < dropdown2.length; i++){
   if(dropdown2.options[i].value=="<?php
echo get_menu2_selection_slug($wp_query);?>"){
                              dropdown2.selectedIndex = i;
    }  // ende if
}  // ende for
function onProduktTypChange() {
    if (dropdown2.options[dropdown2.selectedIndex].value != -1 ) {
        location.href = "<?php echo home_url();?>
/produktgruppe/"+dropdown2.options[dropdown2.selectedIndex].value+"/";
       }   // ende if
}  // ende func onprodukttypchange
dropdown2.onchange = onProduktTypChange;  //Aufruf Funktion
--></script>
</div>
<hr>//WP eingebaute Funktionen, The Loop
<?php while( have_posts() ): the_post(); ?>
    <?php get_template_part('content', 'archive'); //Rufe content-archive ?>
<?php endwhile; ?> //Bootstrap grid:
<div class="col-xs-12 text-left">
    <?php the_posts_navigation(); ?>
</div>
<?php endif; ?>
</div>
</div>
<div class="col-xs-12 col-sm-4"></div>
</div>  <?php get_footer(); ?>
```

b. Dateiname: content-archive.php: CPT Titel aus WP-DB aufrufen und in JS ausgeben. Wird benötigt, damit die Downloads Seite eine Liste mit Titeln, PDF Logos und Links zu den Dateien ausgeben kann.

```
<article id="post-<?php the_ID(); ?>" <?php post_class(); ?>>
<!--Diese Seite wird von page template downloads zugegriffen -->
<?php the_title( sprintf('<h4 class="entry-
title"><a href="%s">', esc_url( get_permalink() ) ),'</a></h4>' );
//title von jedem Content-Archive
echo 'content archive'; ?>
<?php echo bb_get_terms($post->ID, 'produktgruppe');//funktion aufruf
$hersteller_tags = bb_get_terms($post->ID, 'hersteller');
/* Achtung! Oben stehende mit hersteller tag verbunden, bei änderung/löschen
taxonomie hersteller muss mitgeändert/löst werden*/
echo bb_get_terms($post->ID, 'hersteller'); ?>
<!—Damit WP die nicht hierarchische terms/tags zu tax Hersteller bekommt-->
<?php if(current_user_can('manage_options') ){  // für WP-Backend
echo '||'; edit_post_link();                }
?>
<div class="row">
```

```
    <?php if( has_post_thumbnail() ): ?>
    <div class="col-xs-12 col-sm-4">
    <div class="thumbnail"><?php the_post_thumbnail('medium'); ?></div>
    </div>
    <div class="col-xs-12 col-sm-8">
    <?php the_excerpt(); ?>
    </div>
    <?php else: ?>
    <?php $doc = get_post_meta(get_the_ID(), 'wp_custom_attachment', true);
    if (isset(($doc['url'])) && (strlen(trim($doc['url'])) > 0)){
    ?> <div style="padding-left: 15px;"
    class="wp_custom_attachment_archive"><a href="<?php echo $doc['url']; ?>">pdf heru
    nterladen
    <img src="/dev-alarmsol/wp-content/uploads/2019/06/PDF_thumbnail32.png"></a>
    </div>
    <?php
    } // ende if
    ?>
    <div class="col-xs-12">
    <?php the_excerpt(); ?>
    </div>
    <?php endif; //horizontal rule:?>
    </div><hr>
    </article>
```

c. Dateiname: Content-single.php – einzelne Beiträge in CPT – mit herunterladbaren PDF Dateien

```
    <?php
    /*** @package Sydney */?>
    <article id="post-<?php the_ID(); ?>" <?php post_class(); ?>>
    <header class="custompost-entry-header">
    <!--Achtung! Diese klasse wurde umbenannt damit sie entry-
    header klasse nicht beeinflusst: Wilkommen, testbenutzer-->
    <?php the_title( '<h1 class="title-post entry-title">', '</h1>' ); ?>
    </header><!-- .entry-header -->
    <!--klasse wurde umbenannt damit sie über-seite klasse nicht beeinflusst-->
    <div class="custompost-entry-content">
    <?php the_content();
    //hier einzelne pdf zeigen:
    $doc_single = get_post_meta(get_the_ID(), 'wp_custom_attachment', true);
    if (isset(($doc_single['url'])) && (strlen(trim($doc_single['url'])) > 0)) {
           ?>
           <div class="wp_custom_attachment_single">
           <a href="<?php echo $doc_single['url']; ?>">pdf herunterladen
           <img src="https://www.adobe.com/content/dam/acom/en/legal/
                        images/badges/PDF_32.png">
           </a></div>
           <?php echo '<img src=
     "https://www.adobe.com/content/dam/acom/en/legal/images/badges/PDF_24.png">'
           ?> <img src="
           <?php echo get_bloginfo( 'template_directory' ); ?>
           '/wp-content/uploads/2019/06/PDF32.png' />
           <?php
    } // ende if
    wp_link_pages( array(
    'before' => '<div class="page-links">' . __( 'Pages:', 'sydney' ),
    'after'  => '</div>',    ) );  ?>
    </div><!-- .entry-content --><footer class="entry-footer">
    <?php sydney_entry_footer(); ?>
    </footer><!-- .entry-footer -->
    </article><!-- #post-## -->
```