Projektdokumentation

Contact Search Tool

Henry Wünsche
Auszubildender Fachinformatiker für Anwendungsentwicklung

Projektzeitraum
24.09.2021 bis 19.11.2021

Ausbildungs- und Praktikumsbetrieb
Unitedprint.com SE
Friedrich-List-Straße 3
01445 Radebeul

22. 11. 2021

Inhalt

1 Einleitung

Im Folgenden wird das Projekt „Contact Search Tool“ dokumentiert. Dieses Projekt wurde im Rahmen der Abschlussprüfung zum Fachinformatiker für Anwendungsentwicklung[1] erstellt. Der Ausbildungsbetrieb ist Unitedprint.com SE in Radebeul. Unitedprint zählt zu den großen E-Commerce-Unternehmen im Bereich Druck in Europa. Das Unternehmen besitzt eine eigene IT-Abteilung, welche sich unter anderem mit der Entwicklung neuer Features und der Wartung der eigenen Webportale sowie der Pflege der unternehmenseigenen Administration befasst. Dieses Projekt ist eines der zahlreichen internen Projekte, in denen das Unternehmen gleichzeitig auch der Auftraggeber ist.

1.1 Auftraggeber

Auftraggeber und zugleich Kunde des Projektes ist das Backend Relaunch Team, welches Teil der Abteilung IT-Development des ausbildenden Unternehmens ist. Dieses befasst sich mit dem Neubau der unternehmenseigenen Administration, eine Sammlung verschiedener für den internen Betrieb benötigter Tools, welche den jeweiligen Abteilungen für die Bearbeitung der unterschiedllichen Abläufe zur Verfügung stehen.
Das Team steht während der Entwicklung und Testphase des Projekt-Tools als Ansprechpartner für den Auszubildenden zur Verfügung.

1.2 Problembeschreibung

Die momentan genutzte Unternehmens-Administration ist technisch überholt und über ihre Nutzungszeit hinweg immer weiter durch neu hinzugekommene Werkzeuge gewachsen, die zwischenzeitlich teils kaum oder gar keine Verwendung finden, sich aber trotzdem noch im Codebestand befinden und bestenfalls lediglich im Frontend deaktiviert wurden.

1.3 Projektbeschreibung

Im Zuge des Relaunch der Unternehmens-Administration soll der Bestand bereitgestellter Tools überarbeitet und die Neuimplementierung dieser mit angepassten Anforderungen vorgenommen werden. Eines der benötigten Tools ist das Kontakt-Tool, über welches sich Mitarbeiterdaten abrufen lassen. Dieses bietet, neben den für alle Mitarbeiter einsehbaren Daten, der Abteilung Human Resources die Möglichkeit, eine umfangreichere Ansicht der sogenannten Stammdaten aufzurufen.

[1] äquivalent Übungsprojekt

1.4 Projektbegründung

Da es zum einen immer schwieriger wird, Perl-Entwickler zu finden und zum anderen Perl technisch von anderen verfügbaren Sprachen in seiner Leistungsfähigkeit überholt wurde, hat sich der Auftraggeber dazu entschieden, die Admin auf PHP- und Next.js-Basis von Grund auf neu zu implementieren. Damit geht einher, dass der Bestand an verfügbaren Tools geprüft und überarbeitet wird und bei Tools, die übernommen werden, die jeweiligen Anforderungen neu bewertet werden und Funktionalitäten entfallen oder hinzugefügt werden müssen.
Das Kontakt-Tool ist eines der Tools, welches neu und ob dessen abteilungsübergreifender Wichtigkeit für den reibungslosen betrieblichen Ablauf auch als eines der Ersten implementiert werden soll.

1.5 Projektziele

Mit Abschluss des Projektes soll ein funktionierendes Tool zum Suchen von aktiven Mitarbeitern und dem Anzeigen zugehöriger Daten entstehen. Die neue Anwendung soll die gleiche Funktionalität bezüglich der Suchfunktion wie die derzeit Genutzte aufweisen, jedoch in technisch optimierter Form im Relaunch der Administration implementiert werden. Des Weiteren soll der Umfang der angezeigten Daten reduziert werden, da viele bisherig angezeigte Informationen nicht mehr mit aktuellen Datenschutzbestimmungen vereinbar sind.
Das Tool wird jedoch zum Zeitpunkt der Abgabe noch nicht produktiv genutzt werden, da im Relaunch derzeitig noch keine Rechteverwaltung implementiert ist und es dadurch nicht authorisierten Nutzern einen zu großen Einblick in die Stammdaten der Mitarbeiter gewährt.

2 Projektplanung

Bei der Projektplanung wurde ein passendes Vorgehensmodell gewählt, der Projektablauf in Phasen unterteilt und die benötigten Ressourcen geplant.

2.1 Vorgehensmodell

Es wurde sich für das iterative Modell entschieden, da innerhalb des Entwicklungsprozesses Funktionen parallel implementiert und getestet wurden. Das Vorgehensmodell beinhaltet, dass Phasen mehrmals durchlaufen werden und innerhalb jeder Iteration die Software geändert, weiterentwickelt und somit erweitert wird. Hierdurch werden zwischenzeitliche Tests während der Entwicklung ermöglicht und die Erweiterung der Software schrittweise auf Grundlage bereits umgesetzter Funktionalitäten vorangetrieben.

2.2 Projektphasen

Für die Umsetzung des Projektes wurde eine Dauer von 120 Arbeitsstunden vorgegeben. Diese konnten frei innerhalb des Projektzeitraumes von Projektbeginn (24. September 2021) bis Projektabgabe (19. November 2021) für die Umsetzung aufgewendet werden. Es erfolgte eine zeitliche Einteilung in die Phasen Analyse und Konzeption, Implementierung und Testen, Release-Testing, Übergabe und Verfassen der Dokumentation. Eine detaillierte zeitliche Gliederung ist in Anlage A zu finden.
Innerhalb des Projektes gab es 5 Meilensteine, namentlich das „Kickoff-Meeting“, „Abschluss Analysephase“, „Abschluss Implementierungsphase“, „Abschluss Testphase“, die jeweils am Ende der jeweiligen Projektphasen erreicht wurden. Der Meilenstein „Projektabschluss“ stellt auch zugleich die Beendigung des Projektes dar. Die für das Projekt verfügbaren Arbeitsstunden wurden wie folgt auf die unterschiedlichen Phasen aufgeteilt:

Phase	Zeit in Stunden
Analysephase	10
Implementierung	60
Testphase	30
Projektabschluss (Übergabe und Dokumentation)	20

Tabelle 1: Zeitplanung

2.3 Ressourcenplanung

Für dieses Projekt wurde ausschließlich bereits im Unternehmen vorhandene, lizenzierte oder freie Software sowie Hardware genutzt, wodurch keine Neuanschaffungen im Projektrahmen notwendig sind. Eine genaue Aufschlüsselung befindet sich in Anlage B. Das Projekt wurde von dem Auszubildenden an einem vorhandenden PC-Arbeitplatz durchgeführt. Zur Betreuung und für Rückfragen standen insgesamt 3 Entwickler zur Seite.
Das Projekt wurde unter Zuhilfenahme der Versionsverwaltung Git bzw. der Versionierungsplatform Gitlab umgesetzt, um eine adäquate Codeverwaltung und fachgerechte Verteilung zu gewährleisten.

3 Analyse

Vor der Implementierungsphase wurden die vorherrschenden Gegebeneiten sowie der angestrebte Soll-Zustand analysiert und Kalkulationen betreffend der Implementierungskosten durchgeführt.

3.1 Projektschnittstellen

Die Projektschnittstellen können in Personen- und technische Schnittstellen unterteilt werden. In diesem Rahmen wirkte die Fachbetreuerin, Frau Bitterlich, bei der Projektplanung und Anforderungsausarbeitung sowie in nachträglichen Absprachen zum Projekt mit. Unterstützung während der Entwicklung und zur Code Review wurde durch Herrn Schilde, Herrn Schubert und Herrn Makarov gestellt. Anwendertests wurden durch Mitarbeiter der Abteilung Human Resources durchgeführt.
Technische Schnittstellen sind die relationalen Datenbanken des Unternehmens sowie der Relaunch der unternehmenseigenene Administration.

3.2 Ist-Analyse

Die derzeit produktiv genutzte Unternehmens-Administration (im Folgenden „die Admin“) basiert backendseitig technisch auf Perl mit dem Framework „Catalyst“, welches ergänzt wird durch das hauseigen entwickelte und ebenfalls darauf aufbauende Framework „Mariposa“. Dieses stellt die jeweiligen Tools nach dem MVC-Design Pattern[1] bereit. Das HTML der Anwendung wird weitestgehend serverseitig mit Hilfe der Template Engine „Template Toolkit“ gerendert. Für clientseitige Veränderungen der angezeigten Web-Anwendung kommt JavaScript und jQuery zum Einsatz.
Eines der bereitgestellten Tools ist das Kontakt-Tool, über welches die Mitarbeiter des Unternehmens gesucht und unter anderem die Kontaktdaten angezeigt werden können. Die Fachabteilung Human Resources kann über dieses Tool auch weiterführende Informationen, wie den Urlaubsanspruch oder Fehlgründe, für einzelne Mitarbeiter abrufen. Diese sogenannten Stammdaten werden momentan doppelt gepflegt - ein Mal in einer MySQL Datenbank und zusätzlich über das im Unternehmen genutzte Zeiterfassungssystem der Tempras AG. Die Pflege soll zukünftig ausschließlich über das Zeiterfassungssystem erfolgen. Anschließend sollen die Datensätze in der Datenbank aus diesem Bestand aktualisiert werden.

3.3 Soll-Analyse

Um die Admin zu modernisieren, wurde sich dazu entschieden, das bestehende System nicht zu überholen, sondern dieses zu relaunchen. Dadurch wird nicht nur die Codebasis entschlackt, auch die insgesamte Performance der Admin steigt durch den damit einhergehenden Sprachwechsel sowohl auf PHP mit dem Framework „Laravel“ als auch auf Node.js zum Bereitstel-

[1] Model View Controller

len serverseitig vorgerenderter Webseiten. Das JavaScript-Framework „React“ übernimmt das clientseitige Rendern der Web-App. Die Anwendung folgt somit weiterhin dem MVC-Prinzip.
Das neue Kontakt-Tool soll, im Gegensatz zum alten, auch keine Spreadpartner, sondern nur noch Angestellte anzeigen und zum Zeitpunkt der Abgabe technisch soweit vorbereitet sein, dass ein späteres Hinzufügen eines Rollen- und Rechtesystems möglichst einfach umzusetzen ist. Die sonstige Usability soll im Vergleich zur alten Kontaktsuche weitestgehend erhalten bleiben.

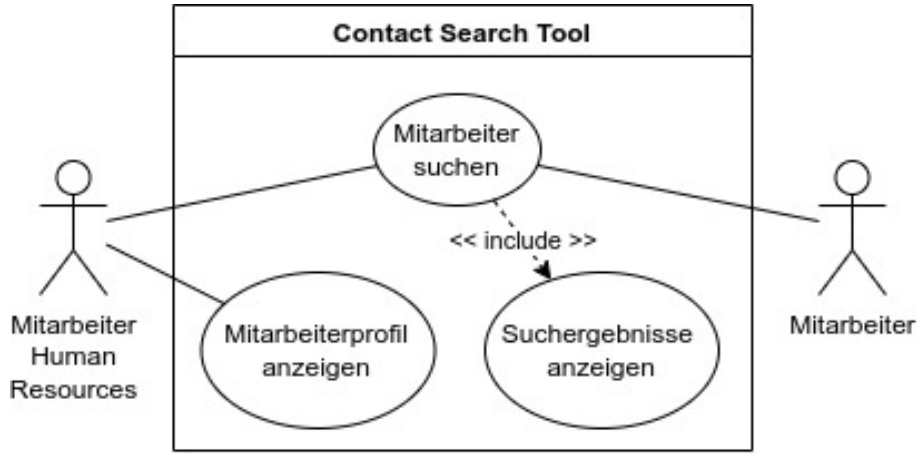

Abbildung 1: Anwendungsfalldiagramm „Kontaktsuche“

3.4 Make-Or-Buy Analyse

Die Option, eine externe Software für diesen Anwendungsfall einzukaufen, wurde schnell verworfen, da die Vorteile einer Eigenentwicklung überwiegen. Ein fremdentwickeltes Produkt bringt beispielsweise mit, dass zusätzliche Schnittstellen zum restlichen System geschaffen werden müssten und die Software entweder in einer gesonderten Form in den Admin Relaunch eingebettet werden müsste, was zusätzliche Kosten verursachen würde, oder als Standalone Anwendung betrieben wird, wodurch der Gedanke einer zentralen Tool-Sammlung nicht konsequent umgesetzt wird.
Eine interne Entwicklung des Kontakt-Tools hat mehrere hervorzuhebende Vorteile. Zum einen ist das zugrundeliegende System mit seinen Schnittstellen bekannt. Zum anderen kann die Software von Anfang an in die Codebasis des Admin Relaunch integriert und punktgenau auf die Anforderungen der Fachabteilung zugeschnitten werden. Auch die spätere Wartung ist einfacher zu realisieren, da dies durch die IT-Entwicklungsabteilung erfolgen kann und kein Bedarf an Wartungsverträgen vorliegt, der ebenfalls wieder Kosten verursacht. Hinzu kommt, dass im Unternehmen vorhandene Technologien wie React, Node.js, CSS, PHP und MySQL genutzt werden und dadurch bereits ausreichend Know-How vorhanden ist, um eine kompetente Wartung zu gewährleisten. Insofern sind die Kosten einer Eigenentwicklung tendenziell geringer als der Kauf eines externen Produktes.

3.5 Kostenanalyse

Für die Kostenanalyse werden Personal- und Betriebskosten zur Software-Entwicklung im Rahmen des Projektes in Betracht gezogen. Zur Berechnung der Personalkosten wurden Bruttogehälter in Höhe von monatlich 1.350 € für Auszubildende, 1.900 € für Mitarbeiter der Fachabteilung Human Resources und 2.500 € für Entwickler veranschlagt. Des Weiteren wurde eine tägliche Arbeitszeit von 8 Stunden je Mitarbeiter und monatlich durchschnittlich 21 Arbeitstage veranschlagt.
Je Arbeitsplatz wurden pauschal 20 € Betriebskosten pro Tag veranschlagt. Darin sind Kosten für vorhandene Soft- und Hardware sowie Strom- und Heizkosten beinhaltet.
Aus diesen Eckdaten ergibt sich ein Stundensatz von 10,54 € für den Auszubildenden, 13,81 € pro Human Resources Mitarbeiter und 17,38 € pro Software-Entwickler.
Der Auszubildende hat insgesamt 120 Mannstunden für das Projekt aufgewendet. Hinzu kommen insgesamt 8 Mannstunden, die Mitarbeiter der Software-Entwicklung in betreuender Form und für Tests aufgewendet haben und 4 Stunden eines Mitarbeiters von Human Resources, der ebenfalls ausführliche Anwendertests durchgeführt und an Projektmeetings teilgenommen hat. Dadurch ergeben sich Gesamtkosten von 1423,80 €.
Eine detaillierte Kostenrechnung ist in Anlage C zu finden.

4 Entwurf

Im folgenden Kapitel wird auf die Zielplattform sowie deren Architektur und verwendete Technologien eingegangen und der angestrebte Funktionsumfang des Tools erläutert.

4.1 Zielplattform

Die neu zu implementierende Kontaktsuche soll im Relaunch der unternehmensinternen Administration umgesetzt werden.

4.1.1 Architektur

Diese folgt einem modularen Ansatz und stellt Webanwendungen nach dem MVC-Prinzip bereit. Dabei wird die Software in 3 logische Komponenten unterteilt: Die View (Ansicht, welche beispielsweise im Browser angezeigt wird), das Model (übernimmt backendseitige Programmlogik wie Datenbankabfragen oder Be- und Verrechnung von Daten) und den Controller (dient als Schnittstelle zwischen Model und View und reicht Anfragen und Antworten zu der jeweiligen Gegenseite durch).

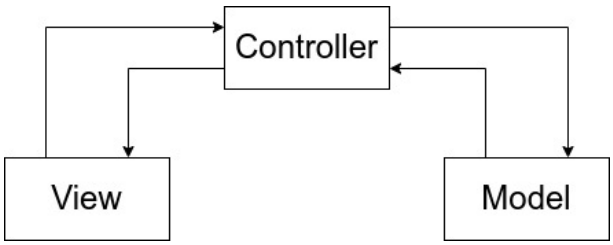

Abbildung 2: Model View Controller - schematische Darstellung

4.1.2 Technologien

Die zugrundeliegenden Technologien sind einerseits PHP in der Verion 8.0 mit dem Framework „Laravel“ in Version 8.72.1, einer MySQL Datenbank in Version 5.5.62 und Node.js zum Bereitstellen des Frontends. Die auszuliefernden Dateien liegen als Typescript-Quelldateien vor. Clientseitig kommt das JavaScript-Framework „React“ und die CSS-Bibliothek „Bootstrap“ zum Einsatz.
Die Entscheidung, PHP und Laravel zu verwenden, viel mit Hinblick auf den Relaunch der Unternehmensportale „Print24“ und „Easyprint“, da dadurch eine unternehmensweite einheitliche Basis in den zugrundeliegenden Technologien besteht. Das gleiche gilt für die Wahl des Datenbankmodells, was noch näher in Abschnitt 4.1.3 erläutert wird. Die Wahl für Typescript anstelle von JavaScript fiel ebenfalls in der Planungsphase des Admin-Relaunches. Durch den objektorientierten Ansatz und die vorhandene Typensicherheit ist der resultierende Code robuster und sicherer als reiner JavaScript-Code. Als Äquivalent für das im Altsystem genutzte Template Toolkit kommt Node.js zum Einsatz. Dies rendert die Websites der Anwendung mit dynamischen Inhalten bereits serverseitig vor und liefert sie dem Client in fertiger Form aus.

Auf Clientseite wird React verwendet, welches durch das Einbinden von zusätzlichen Modulen schnell, einfach und zielgerichtet in dessen Funktionalität angepasst und erweitert werden kann. Die Module sind hierbei nicht global verfügbar, sondern müssen explizit in jeder App eingebunden werden. Dies gewährleistet eine erhöhte Sicherheit der Gesamtumgebung, da eventuelle Schwachstellen in Modulen nur dann eine Gefahr darstellen, wenn diese auch aktiv in einer Anwendung genutzt werden. Weiter reduziert sich durch React der zu schreibende Code auf ein Minimum, was der Wartbarkeit der Anwendung durchaus zugute kommt. Außerdem können dadurch auch dynamische Inhalte besser verarbeitet und dargestellt werden.
Für das Styling der Seite fiel die Wahl auf das von Twitter entwickelte freie CSS-Framework Bootstrap. Dieses ist global verfügbar und bringt eine einfach zu lernende Syntax zum Einbinden von CSS-Klassen und eine vielseitige Auswahl bereits vordefinierter Styling-Klassen mit sich. Durch die einheitliche Verwendung von Bootstrap wird sichergestellt, dass im gesamten Admin-Relaunch ein einheitlicher optischer Stil vorherrscht. Gleichzeitig kommt dies wieder der Wartbarkeit und einer reduzierten Codebasis zugute, da dadurch das manuelle Anlegen und Pflegen von CSS-Quelldateien und -Klassen entfällt.

4.1.3 Datenbankmodell

Die Mitarbeiterdaten werden bereits in einer relationalen MySQL Datenbank in der dritten Normalform gespeichert und gepflegt. Die Wahl für dieses Datenbankmodell ist darin begründet, dass durch dieses Design Redundanzen in den Datensätzen vermieden werden. Die Mitarbeiterdaten werden nämlich nicht zentral in einer Tabelle, sondern über mehrere Tabellen verteilt gespeichert. So gibt es beispielsweise eine Tabelle für Kontaktdaten, eine für Fehlgründe und eine für die im Unternehmen vorhandenden Abteilungen. Die Beziehhung zwischen Mitarbeiter und Abteilung wird über eine Relationstabelle dargestellt. Über diesen Ansatz wird der Speicherbedarf der Daten und die Gefahr von Inkonsistenzen innerhalb der Datensätze erheblich reduziert. Ein entsprechendes Datenbankdiagramm befindet sich in Anlage [NUMMER].

4.1.4 Funktionsumfang

Die Kontaktsuche soll, was die Usability anbelangt, sich möglichst wenig von der alten Suche unterscheiden. Das heißt, dass Mitarbeiter mit Hilfe des Vor- und Nachnamens und / oder über die Abteilung gesucht und diese in alphabetischer Reihenfolge aufgelistet werden. Die Auswahl der Abteilung erfolgt über ein Drop-Down Menü in Form eines Selects. Dabei stellt auch ein unvollständiger Vor- oder Nachname, keine Vorauswahl der Abteilung oder auch eine „leere Suche“ ein valides Suchkriterium dar, da davon auszugehen ist, dass nicht immer der volle Name oder die Abteilung, in der ein Mitarbeiter arbeitet, bekannt ist. Die Suchergebnisse sollen tabellarisch aufgelistet werden und die Mitarbeiternummer, den Namen, die Abteilung, den Anwesenheitsstatus und Kontaktdaten wie die dienstliche Telefonnummer und Email-Adresse sowie ein Bild des Angestellten anzeigen.
Der Nachname soll, wie im alten Tool auch, ein Link sein, der zu einer Profilübersicht des jeweiligen Angestellten führt, in der weiterführende Informationen wie private Kontaktdaten angezeigt werden oder die Möglichkeit besteht, Login-Passwörter und Pins zu ändern. Der Aufruf

der Profilseite soll auch mit einer direkten Eingabe der URL inklusive Mitarbeiternummer im Schema „backend.com/contacts/1234“ möglich sein, um den Suchprozess zu beschleunigen. Diese Ansicht wird aus Datenschutzgründen nur authorisierten Mitarbeitern von Human Resources zugänglich sein, da nur diese entsprechende Daten verarbeiten.
Im Gegensatz zum aktuell noch genutzten Tool soll eine Bearbeitung der Mitarbeiterstammdaten über das Such-Tool nicht mehr möglich sein, sondern nur noch eine Weiterleitung in das Personalmanagement-Tool der Tempras AG existieren. Der Hintergrund ist, dass Stammdaten zukünftig nur noch über diese Software und nicht zusätzlich noch über die Suche in der Datenbank bearbeitet werden können, da diese Praktik sehr fehleranfällig ist und zu Inkonsistenzen im Datenbestand führen kann. Die in der Personalmanagement-Software gepflegten Daten sollen dafür dann regelmäßig in den Datenbanken eingepflegt und auf diese Weise auch da aktuell gehalten werden.
Da die detaillierte Profilansicht nur Mitarbeitern von Human Resources zugänglich sein soll, wird die Software so konzipiert, dass eine später umgesetzte Rollen- und Rechteverwaltung so einfach wie möglich nachgerüstet werden kann. Da so ein System beim aktuellen Entwicklungsstand der Admin jedoch noch nicht implementiert ist, können maximal allgemein gehaltene Schnittstellen definiert werden. Möglich wäre hier die Nutzung einer Middleware, die bei entsprechenden Routen zwischengeschalten wird.

5 Implementierung

Der grundsätzliche Aufbau der Suche war im Vorfeld durch das Vorhandensein im Altsystem bereits bekannt und musste nicht komplett neu konzipiert werden. Zu Beginn wurde der Code des Alt-Tools analysiert und begonnen, diesen im Admin-Relaunch in Typescript (mit React) und PHP neu zu schreiben. Beispielsweise wurden Algorithmen effizienter gestaltet und Suchkriterien optimiert. Des Weiteren wurden im Zuge des Neubaus auch Bugs gefixt.

5.1 Model View Controller Design Pattern

Wie bereits in Absatz 4.1.1 erwähnt, basieren auch die Anwendungen im Admin-Relaunch auf dem Model View Controller Design Pattern. Allerdings werden in Laravel bestimmte Bezeichnungen anders genutzt als im Altsystem.

5.1.1 Model, Repositories & Libraries

Anders als im auf Perl und dem Framework Catalyst basierenden Altsystem, wo Models Funktionen für die Programmlogik beinhalten, sind diese im PHP-Framework Laravel vielmehr Datenbankobjekte. Diese werden mittels Eloquent, einer ORM[1]-Schnittstelle von Laravel, oder über den Laravel Query Builder verwendet. Jedes Model bildet eine Tabelle der Datenbank mit all ihren Attributen ab und stellt entsprechende Zugriffsmethoden bereit. Die Eloquent- bzw. Query Builder-Aufrufe werden in sogenannten Repository-Klassen definiert, welche ausschließlich Funktionen für Datenbankabfragen beinhalten (näheres in Absatz 5.1.4).
Diese Funktionen werden wiederum aus Library-Klassen aufgerufen, die die restliche Programmlogik beinhalten - zum Beispiel die Umformatierung von Datenstrukturen und Vorbereitung dieser zur Weitergabe an das Frontend. Somit erfüllen Libraries und Repositories noch am ehesten die Funktionen, die Models im Altsystem erfüllten.

5.1.2 View

Die View beschreibt die Ansicht der Anwendung, also das, was der Benutzer im Browser angezeigt bekommt. Die Anzeige wurde durch den Einsatz von serverseitigem Rendern via Node.js und dynamischer Anpassungen zur Laufzeit mittels React realisiert. Dabei werden per Node.js aus Typescript-Quelldateien Standard-Ansichten kompiliert und diese dann an den Client ausgeliefert. Der Gebrauch dieser Technologien hat den entscheidenden Vorteil, dass wenn sich eine Ansicht ändert, beispielsweise wenn eine Suche gestartet oder eine Unterseite aufgerufen wird, nicht die komplette Anwendung neu gerendert werden muss, sondern nur der betreffende Teil der View. Daraus ergeben sich ein geringerer Datenverkehr und insgesamt schnellere Ladezeiten.
So wurde die Suchmaske in eine sogenannte „Component“ ausgelagert, was bewirkt, dass diese nur beim Seitenaufruf gerendert und nicht bei jedem Event, welches auf der Seite ausgewertet wird (z. B. Nutzereingaben).

[1] Object-Relational Mapping - Objektrelationale Abbildung

5.1.3 Controller

Der Controller nimmt, wie auch im Altsystem, nur eine vermittelnde Rolle zwischen der View und zumeist den in Library-Klassen befindlichen Funktionen ein. Controller-Funktionen werden über verschiedene API-Routen, die in der Datei „api.php “ definiert sind, angesprochen. In dieser Datei wird auch die Art der Anfrage pro Route festgelegt, also ob es sich beispielsweise um eine Post- oder Get-Anfrage handelt, und werden durch das Framework auch abgewiesen, wenn die Art nicht zu der Route passt.
Es wurde versucht, den Controller so schlank wie möglich zu halten und jegliche Logik in die Model-Komponenten auszulagern. Dies war ein teils stetiger Prozess, da anfangs die komplette Logik im Controller verblieb und mit wachsendem Kenntnisstand des Auszubilden ein schrittweiser Umbau vonstatten ging.

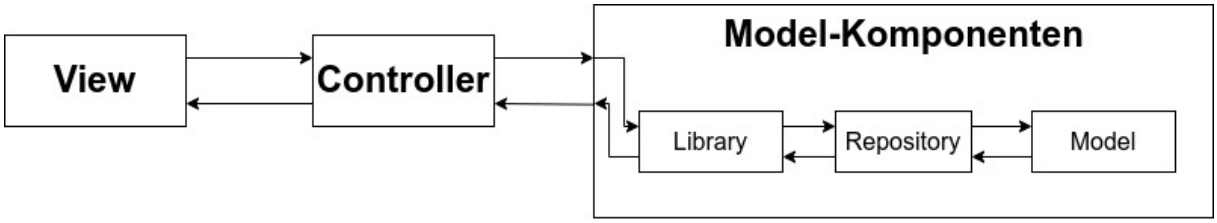

Abbildung 3: MVC im PHP-Framework Laravel

5.1.4 Eloquent und Query Builder

Laravel stellt verschiedene Möglichkeiten bereit, Datenbankabfragen zu stellen. „Eloquent“ verfolgt den objektrelationalen Ansatz. Hier werden ganze Tabellen in Objekten instanziert. Relationen und Kardinalitäten werden durch Objektmethoden dargestellt, die fest definierte Namen haben, beispielsweise „hasOne()“ oder „hasMany()“. Jedoch ist Eloquent nicht so performant wie Query Builder, da es einfache Queries auf mehrere Einzelne aufteilt, sofern weitere Tabellen gejoint werden. In diesem Fall wird für jede verbundene Tabelle ein weiteres Statement ausgeführt und erst, wenn alle zugehörigen Datensätze vorliegen eine Selektion der eigentlich abgefragten Daten durchgeführt.
Queries, die mit Query Builder geschrieben sind, ähneln in ihrer Syntax hingegen herkömmlichen MySQL Queries. Diese werden über eine Datenbank-Fassade verarbeitet. Die verfügbaren Funktionen sind in ihrer Benennung den SQL Keywords angelehnt. So gibt es Funktionen wie „select()“, „where()“ und „orderBy()“.
Aus diesem Grund ist Query Builder gerade bei großen und spezialisierten Abfragen über mehrere Tabellen weitaus leistungsfähiger, da sich der Overhead an Datenbankabfragen und übermittelten Daten auf ein Minimum reduzieren lässt.
Da Eloquent-Modelle mit Query Builder kompatibel sind, kann man durchaus eine Mischform beider Ansätze verwenden.

5.2 Aufgetretene Probleme

Während der Projektdurchführung stieß der Auszubildende auf einen Anwendungsfall, bei dem er zuerst den Ansatz verfolgte, Datenbankabfragen per Eloquent durchzuführen, jedoch durch API-Tests und Analysen feststellte, dass eine Lösung per Query Builder weitaus performanter war. In besagtem Fall wies das Model Relationen zu weiteren Tabellen auf, aus denen je nur ein Attribut benötigt wurde. Eloquent führte jedoch 3 SQL Queries aus: eines auf die Haupttabelle und zwei weitere je auf die verbundenen Tabellen. Aus jeder Tabelle wurden komplette Datensätze abgeholt und in einem weiteren Schritt dann die eigentlich angeforderten Daten selektiert.
Nach dem Umbau auf Query Builder reduzierten sich die Abfragen auf eine einzige und es wurden direkt nur die tatsächlich benötigten Daten abgeholt. Dadurch wurde die benötigte Zeit pro Abfrage um ca. 60% reduziert.

Ein weiteres Problem stellte das dynamische Befüllen des Selects für die Abteilungsauswahl dar. Die verschiedenen Abteilungen im Unternehmen sind hierarchisch organisiert. Zum Beispiel ist der Abteilung „Finance/HR/Legal“ die Abteilung „Campus“ und dieser wiederum die Abteilung „IT Education“ untergeordnet. Diese Hierarchie sollte ebenfalls in der Auswahl und auch in den Suchergebnissen dargestellt werden. Um dies zu realisieren, wurden die Datensätze der Abteilungen zuerst aus der Datenbank ausgelesen und danach durch eine sich rekursiv aufrufende PHP-Funktion in ein multidimensionales Objekt geschrieben. Anschließend wurde dieses an das Frontend ausgeliefert, wo es wieder durch eine rekursiv aufgerufene Funktion ausgelesen wurde. Mit jeder Ebene, die im Objekt tiefer abgestiegen wurde, wurde dem im Select angezeigten String ein „>“ vorangestellt, um so die Hierarchieebene optisch darzustellen.

Eine weitere Schwierigkeit war das Abrufen der Profilbilder. Da diese nicht direkt im Frontend durch simple Angabe des Quellpfades eingebunden werden können, weil dieses keine Zugriffsberechtigung auf das entsprechende Verzeichnis hat, musste ein anderer Weg gefunden werden. So wurde zuerst versucht, über eine eigene API-Anfrage einen Binary Blob der Bilder an das Frontend zu schicken, was prinzipiell auch funktioniert hat. Jedoch konnten die React-Elemente zum Anzeigen von Bildern damit nicht umgehen. Also wurden die Bilder erst base64-codiert und anschließend im Frontend in Image-Elemente eingebunden, was zum gewünschten Ergebnis führte. Abschließend wurde der erzeugte String des Bildes noch direkt an den zugehörigen Datensatz mit den Mitarbeiterdaten angefügt und so die Requests, welche das Frontend an das Backend stellt, erheblich reduziert.

Eine Anforderung der Fachabteilung war, dass das deutsche Datumsformat verwendet wird, obgleich das restliche Tool englisch gehalten wird. Nach der anfänglichen Verwendung einer eigens geschriebenen Funktion, welche das Datum vom in der Datenbank vorliegenden amerikanischen auf das deutsche Format ändert, hat der Auszubildende sich dazu entschieden, die bereits existierende Funktion „parse()“ aus der PHP Klasse „Carbon“ zu verwenden. Zum einen wird dadurch redundanter Code vermieden, zum anderen ist man damit beim Format des Eingabewertes flexibler.

6 Testing

Während der Entwicklung wurden die umgesetzten Funktionen bereits getestet, sodass nach der Implementierungsphase nur noch abschließende Gesamt- und Nutzertests durch die Mitarbeiter der Fachabteilung nötig waren.

6.1 Vorgehen

Neue Funktionen wurden während der Implementierung bereits auf viele Szenarien hin eingehend getestet. Zum Beispiel wurden API-Tests durchgeführt, in denen die korrekte Verarbeitung von übergebenen Daten und auch das Verhalten überprüft wurde, wenn Daten nicht in der erwarteten Struktur oder in einem anderen Datentyp vorlagen. Somit konnte sichergestellt werden, dass die Anwendung nicht durch falsche Eingaben zum Absturz gebracht wird, sondern ein entsprechendes Error-Handling greift. Ist die korrekte Funktionsweise eines Anwendungsfalls sichergestellt worden, konnte darauf aufbauend weiterentwickelt werden.

6.2 Code-Review

Das Tool wird im Rahmen einer Code Review von den betreuenden Entwicklern auf Schwachstellen im Code und Einhalten der internen wie externen Coding-Guides hin überprüft. Damit wird eine angestrebte Qualität des Codes gewährleistet.

6.3 Anwender-Tests

Zum aktuellen Zeitpunkt sind noch keine Anwender-Tests erfolgt. Diese sind jedoch als Blackbox-Tests geplant, in denen Mitarbeiter der Fachabteilungen das im Rahmen des Projektes umgesetzte Tool auf dessen Funktionalität hin testen. Dabei haben diese keine Kenntnis des Codestandes.

6.4 Test-Szenarien

Eine Auflistung durchgeführter Test-Szenarien ist im Anlage D zu finden. Daraus ist zu entnehmen, welche Anwendungsfälle vorgesehen waren und deren Testergebnisse.

7 Projektabschluss

Im Folgenden wird der erreichte Stand des Projektes analysiert und ein Ausblick auf dessen Weiterentwicklung gegeben.

7.1 Projektübergabe

Das Tool befindet sich zum Zeitpunkt der Projektabgabe in einem funktionsfähigen Zustand und weißt alle zum Projektbeginn geforderten Funktionalitäten auf. Da kurz vor Abschluss des Projektes neue Anforderungen durch die Fachabteilung gestellt wurden, konnten diese noch nicht in selbiges eingepflegt werden. Die Übergabe ist nach Umsetzung dieser geplant.

7.2 Ausblick

Nach einem weiteren Meeting mit der Betreuerin der Fachabteilung werden die entsprechenden neuen Anforderungen in Form von Nacharbeiten im Tool umgesetzt. Weiter wird, sobald eine Rollen- und Rechteverwaltung im Relaunch der Admin implementiert wurde, eine Authorisierung des Nutzers durch selbiges in das Contact Search Tool eingebaut.
Zudem soll eine Schnittstelle zur im Unternehmen genutzten Personalmanagement-Software der Tempras AG ergänzt werden, um dort die Personaldaten des ausgewählten Mitarbeiters bearbeiten zu können.

7.3 Fazit

Bei der Durchführung des Projektes hat der Auszubildende neue Einblicke in unter anderem PHP, Laravel, Typescript und React bekommen. Des Weiteren konnte er seine Fähigkeiten im Bereich ojektorienterter Programmierung vertiefen. Er hat sich einen fundierten Kenntnisstand angeeignet, der ihm als Basis in folgenden Projekten hilfreich sein wird.

8 Quellen

[1] Laravel 8 Dokumentation
letzter Aufruf 22.11.2021 10:00 Uhr
https://www.laravel.com/docs/8.x/

[2] React Dokumentation
letzter Aufruf 22.11.2021 10:15 Uhr
https://www.reactjs.org/docs/getting-started.html

[3] PHP Dokumentation
letzter Aufruf 22.11.2021 10:30 Uhr
https://php.net/docs.php

9 Abbildungsverzeichnis

10 Glossar

Anwender-Tests	Tests, die direkt von der Zielgruppe durchgeführt werden, in der Regel Blackbox Tests
API	Programmierschnittstelle, dient zur Verbindung von Programmen und Programmteilen
Array	geordnete Liste von Elementen
Blackbox Testing	Testverfahren, in dem der zugrundeliegende Quellcode nicht bekannt ist und der Tester die Funktionen „blind“ testet
Bootstrap	freies Styling Framework, welches CSS-Klassen zum Stylen von Webseiten bereitstellt.
Carbon	PHP Klasse, welche zeitbezogene Funktionen bereitstellt, beispielsweise Datumformatierungen
Catalyst	Framework für Perl
CSS	Cascading Style Sheets, Stylesheet-Sprache für elekronische Dokumente. Für Aussehen von Webseiten verantwortlich.
Code Review	Manuelles Prüfen von Arbeitsergebnissen durch Mitarbeiter der IT-Entwicklungsabteilung
Design Pattern	Entwurfsmuster zur Lösung wiederkehrender Probleme. Schablone für Softwarearchitekturen
Eloquent	ORM-Ansatz von Laravel, um Datenbankabfragen zu ermöglichen
Framework	Programmiergerüst, welches Funktionalitäten und Rahmenbedingungen zum Entwickeln von Anwendungen bereitstellt
Git	Freie Software zur verteilten Versionsverwaltung von Dateien
Human Resources	Personalabteilung im Unternehmen
HTML	Hypertext Markup Language, Grundgerüst von Webseiten
JavaScript	Skriptsprache, die im Webbrowser ausgeführt wird und durch welche HTML manipuliert und dynamische Inhalte dargestellt werden können
jQuery	JavaScript-Bibliothek, die vereinfacht Funktionalitäten bereitstellt.
Laravel	Framework für PHP
Middleware	Anwendungsneutrale Programme, die zwischen Anwendungen und Anwendungsteilen vermitteln
MVC	Das Model View Controller Design Pattern dient zur Unterteilung von Software in drei Grundkomponenten: der Ansich (View), dem Datenmodell (Model) und der Programmsteuerung (Controller).

MySQL	Relationales Datenbankmodell
Next.js	Open Source Framework, dass auf Node.js basiert und React-basierte Webanwendungsfunktionen wie serverseitiges Rendering ermöglicht.
Node.js	Plattformübergreifende Open Source JavaSript-Laufzeitumgebung, die JavaScript Code außerhalb eines Webbrowsers ausführen kann.
Open Source	Software, deren Quelltext öffentlich zugänglich ist.
ORM	Obrektrelationale Abbildung (engl. object-relational mapping), Technik Objekte in objektorienterter Programmierung in Datenbanken abzulegen bzw. diese nach objektorientiertem Ansatz im Programm abzubilden
Perl	skriptbasierte Programmiersprache
PHP	prozedurale Skriptsprache, zur Verwendung in Webanwendungen konzipiert
Query Builder	Funktion von Laravel, Datenbankabfragen zu ermöglichen. Für komplexe Queries ist Query Builder Eloquent aus Performancegründen vorzuziehen.
React	JavaScript Softwarebibliothek zur Konstruktion von Webseiten und Oberflächen
Relaunch	kompletter Neubau eines Software-Produktes ohne Berücksichtigung bestehender Versionen
Typescript	von Microsoft entwickelte Programmiersprache, die Typisierung von Variablen beinhaltet und auf Vorschlägen zum ECMAScript 6 Standard basiert, ist zu JavaScript kompatibel
Usability	Die Bedienbaarkeit einer Anwendung

11 Anlagen

11.1 Anlage A Zeitplanung

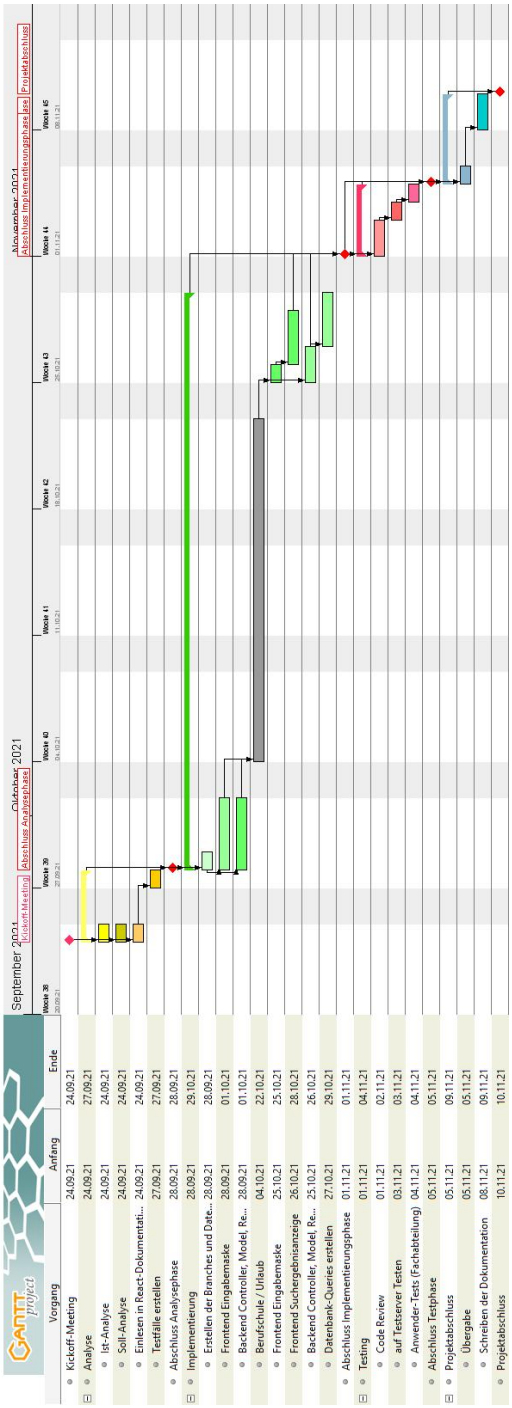

Abbildung 4: Gantt-Diagramm zur Projektzeitplanung

11.2 Anlage B Ressourcenplanung

Hardware

- Büroarbeitsplatz
- 2 Monitore
- 1 Dockingstation
- 1 Laptop
- Maus, Tastatur

Software

- Virtuelle Maschine mit Fedora 32 mit folgender Software:
  - Docker-basierte Entwicklungsumgebung
  - MySQL Datenbank & -server
  - Proxyserver (Squid)
  - Quelltext-Repositories des Backend Relaunches
- Notebook mit Windows 10 Enterprise mit folgender Software
  - PHPStorm
  - Visual Studio Code
  - SQLyog Ultimate 64 (Datenbankmanagementsystem)
  - Google Chrome
  - Mozilla Firefox
  - Git
  - Jira - Projektmanagementsystem (Webanwendung)

Personal

- Auszubildender, Entwickler - Projektumsetzung
- Fachbetreuer, Entwickler - Beratung, Code Review
- Mitarbeiter Abteilung IT - Code Review
- Mitarbeiter Fachabteilung Human Resources - Tests, Projektabnahme

11.3 Anlage C - Kostenanalyse

	Auszubildender	Mitarbeiter HR	Mitarbeiter IT
monatliches Gehalt	1.350 €	1.900 €	2.500 €
Arbeitstage / Monat	21	21	21
Stunden / Tag	8	8	8
effektiver Stundensatz	8,04 €	11,31 €	14,88 €
Betriebskosten / Tag	20 €	20€	20 €
Mannstunden	120	4	6
Kosten	1.264,29 €	55,24 €	104,29 €
Gesamtkosten			1.423,81 €

11.4 Anlage D - Testprotokoll

Nr.	Testfall	Ergebnis	Status
1	Tool in Admin Dropdown auswählen	Tool wird geladen	Erfolgreich
2	Eingabe in Inputfelder	Eingabe wird übernommen	Erfolgreich
3	Eingabe unerlaubter Zeichen	Zeichen wird nicht in Input übernommen	Erfolgreich
4	Suche ohne Vor-, Nachname oder Abteilung	Ausgabe aller Mitarbeiter	erfolgreich
5	Suche mit unvollständigem Vornamen	Ausgabe aller passenden Mitarbeiter	Erfolgreich
6	Suche mit unvollständigem Nachnamen	Ausgabe aller passenden Mitarbeiter	Erfolgreich
7	Suche nur mit ausgewählter Abteilung	Anzeige aller Mitarbeiter in Abteilung	Erfolgreich
8	Suche mit vollständigem Namen	Zutreffender Mitarbeiter wird angezeigt	Erfolgreich
9	Klick auf Nachnamen in Suchergebnissen	Weiterleitung in Profilansicht	Erfolgreich
10	Klick auf „Back“ Button	Zurück zur Suchansicht	Erfolgreich
11	Aufruf der Profilansicht über URL (backend.unitedprint.dev/other/contacts/1234)	Weiterleitung in Profilansicht	Erfolgreich