Graphical user interface

Description automatically generated

**ULLA**

Firmeninternes Management Interface zur Urlaubsverwaltung & Antragserstellung als Webapplikation zum Zwecke der Digitalisierung

*Datum: 22.04.2021*

Dokumentation zur Betrieblichen Projektarbeit

**Prüfungsbewerber:**

Maximilian Müller

**Prüf-Nr.:**

4087

Inhalt

[Glossar 3](#_Toc69916968)

[Technisches Abkürzungsverzeichnis 4](#_Toc69916969)

[1 Projektbeschreibung 5](#_Toc69916970)

[1.1 Projektbezeichnung 5](#_Toc69916971)

[1.2 Projektumfeld 5](#_Toc69916972)

[1.3 Projektziel 5](#_Toc69916973)

[1.4 Projektbegründung 5](#_Toc69916974)

[1.5 Projektschnittstellen 6](#_Toc69916975)

[1.6 Projektabgrenzung 6](#_Toc69916976)

[2 Projektplanung 6](#_Toc69916977)

[2.1 Projektphasen 6](#_Toc69916978)

[2.2 Abweichungen vom Projektantrag 6](#_Toc69916979)

[2.3 Ressourcenplanung 7](#_Toc69916980)

[2.4 Entwicklungsprozess 7](#_Toc69916981)

[3 Analysephase 7](#_Toc69916982)

[3.1 Ist-Analyse 7](#_Toc69916983)

[3.2 Wirtschaftlichkeitsanalyse 8](#_Toc69916984)

[3.2.1 „Make or buy“ Entscheidung 8](#_Toc69916985)

[3.2.2 Projektkosten 8](#_Toc69916986)

[3.2.3 Amortisationsdauer 8](#_Toc69916987)

[3.3 Anwendungsfälle 9](#_Toc69916988)

[3.4 Lastenheft 9](#_Toc69916989)

[4 Entwurfsphase 9](#_Toc69916990)

[4.1 Zielplattform 9](#_Toc69916991)

[4.2 Server 10](#_Toc69916992)

[4.2.1 Spezifikationen 10](#_Toc69916993)

[4.3 Architekturdesign 10](#_Toc69916994)

[4.4 Benutzeroberfläche 10](#_Toc69916995)

[4.5 Datenmodell 11](#_Toc69916996)

[Literaturverzeichnis 12](#_Toc69916997)

[A Anhang 13](#_Toc69916998)

[A.1 Grobe Zeitplanung 13](#_Toc69916999)

[A.2 Detaillierte Zeitplanung 14](#_Toc69917000)

[A.3 Ist-Analyse - Aktivitätsdiagramm 14](#_Toc69917001)

[A.4 Kalkulatorische Aufstellung 15](#_Toc69917002)

[A.5 Zeiteinsparung 15](#_Toc69917003)

[A.6 Anwendungsfälle – Use Case Diagramm 16](#_Toc69917004)

[A.7 Auszug Lastenheft 17](#_Toc69917005)

[A.8 Mockup - Formular 18](#_Toc69917006)

[A.9 Mockup - Dashboard 18](#_Toc69917007)

[A.10 Datenbankschema 19](#_Toc69917008)

# Glossar

Alle im Glossar zu findenden Fachbegriffe sind in der nachfolgenden Dokumentation *kursiv-unterstrichen* formatiert.

**CMS** *-* Ein Content-Management-System (kurz CMS, deutsch Inhaltsverwaltungssystem) ist eine Software zur gemeinschaftlichen Erstellung, Bearbeitung, Organisation und Darstellung digitaler Inhalte (Content) zumeist zur Verwendung in Webseiten, aber auch in anderen Medienformen.

**Mail2Many** – Das von Atrivio entwickelte Newsletter-System mail2many ermöglicht die Kommunikation zwischen Herstellern, ihren Händlern und Kunden.

**Geneva** – Ein von Atrivio entwickeltes Content-Management-System.

**Compliance** - Compliance ist die betriebswirtschaftliche und rechtswissenschaftliche Umschreibung für die Regeltreue von Unternehmen, also die Einhaltung von Gesetzen, Richtlinien und freiwilligen Kodizes.

**Laravel** - Laravel ist ein freies PHP-Webframework, das dem MVC-Muster folgt. Es wurde 2011 von Taylor Otwell initiier.

**Blade** - Blade ist die standardmäßig mit Laravel mitgelieferte Template Engine. Mit Blade lassen sich Views in einer Laravel Anwendung implemetieren. Template Engines wie Blade ermöglichen die Wiederverwendbarkeit von Views und erhöhen damit die Produktivität bei der Entwicklung sowie die Wartbarkeit einer Anwendung. View-Teile die sehr ähnlich oder identisch sind, können als einzelne Blöcke definiert und innerhalb anderer Views wiederverwendet werden. Beim richtigen Einsazt von Blade entsteht gut strukturierter und damit gut lesbarer Code, der neuen Entwicklern den Einstieg in bestehende Projekte deutlich erleichtern kann.

**ULLA** – Projektname (Urlaubsverwaltung & Antragserstellung)

# Technisches Abkürzungsverzeichnis

Alle im technischen Abkürzungsverzeichnis zu findenden Fachbegriffen sind in der nachfolgenden Dokumentation unterstrichenformatiert.

**PHP** Hypertext Preprocessor

**MVC** Model View Controller

**HTML** Hypertext Markup Language

**ERM** Entity Relation Modell

# Projektbeschreibung

## Projektbezeichnung

Firmeninternes Management Interface zur Urlaubsverwaltung & Antragserstellung als Webapplikation zum Zwecke der Digitalisierung.  
Kurzform ULLA (Akronymisiert aus Urlaubsverwaltung & Antragserstellung)

## Projektumfeld

Auftraggeber ist die Atrivio GmbH (im Folgenden durch Atrivio abgekürzt). Das Projekt wurde in den Räumen der Atrivio in 87437 Kempten durchgeführt.  
Seit über 18 Jahren begleitet die Atrivio Unternehmen bei der Umsetzung kundenspezifischer, digitaler Lösungen. Dabei werden zum einen Dienstleistungen wie die Umsetzung Webbasierter [*CMS*](#_Glossar) Lösungen, E-Commerce-Plattformen bis hin zur Cloud Infrastruktur angeboten, aber auch eigene Produkte wie [*Mail2Many*](#_Glossar)oder [*Geneva*](#_Glossar)betriebseigen entwickelt und vertrieben.

## Projektziel

Der gesamte Prozess eines Antrages von der Erstellung bis zur Archivierung soll zukünftig webbasiert – digitalisiert werden.   
Primär sollen die Abteilungsleiter durch den Digitalisierungsprozess von den bürokratischen und organisatorischen Mehraufwänden des bisherigen Prozesses entbunden werden. Außerdem werden durch die digitale Archivierung in Zukunft Druckkosten- und material reduziert. Durch das neue System wird eine strikte [*Compliance*](#_Glossar) und Ordnung in den Ablauf der Antragserstellung eingebracht.

Durch die Umsetzung soll eine prozessorientierte, schlanke und zukünftig erweiterbare intern gehosteten Web-Anwendung mit Anbindung einer Datenbank und grafischer Benutzeroberfläche entstehen.

## Projektbegründung

Derzeit steht allen Mitarbeitern eine Word Vorlage für die Antragserstellung zur Verfügung. Diese Vorlage ist zentral und für alle Mitarbeiter erreichbar gespeichert.   
Soll ein neuer Antrag eingereicht werden wird die Vorlage aufgerufen, befüllt und ausgedruckt. Der unterschriebene Ausdruck wandert dann durch die Abteilung und wird schlussendlich in einem der Verwaltungsordner archiviert. Durch den bisherigen Prozess entstehen mehrere wirtschaftliche Makel. Beispielsweise wird die Prozessdauer durch die händische Weitergabe des Antrags unnötig in die Länge gezogen. Außerdem werden, wie in [Punkt 1.3](#_Projektumfeld) erwähnt, vermehrt Unternehmensressourcen wie Büromaterialien (Druckermittel, Papier, Ordner) oder Stellplatze (Regale) verbraucht.

Ein Organisatorischen Makel stellt die fehlende Zentralisierung der Anträge da. Momentan besteht keine Möglichkeit Anträge Abteilungsübergreifend zu vergleichen u. o. abzustimmen.

Mit der Umsetzung von [*ULLA*](#_Glossar)werden die organisatorischen und wirtschaftlichen Makel beseitigt und eine zentrale, zukunftssichere, Kosten und Ressourcensparende Anlaufstelle geschaffen.

## Projektschnittstellen

Der Endbenutzer kann die Webapplikation in jedem aktuellen Browser und auf einer Vielzahl von Geräten wie beispielsweise PC‘s, Notebooks über Tablets bis hin zu Smartphones im internen Firmennetzt aufrufen. Dabei ist die Applikation im ersten Schritt zwar responsiv fähig aber für die Ansicht auf Desktopendgeräten optimiert.

Um an die Applikationsdaten zu gelangen, muss die Anwendung mit einer MySQL-Datenbank kommunizieren und Werte abfragen können. Diese befindet sich im internen Firmennetz und kann direkt angesprochen werden.

Weitere Schnittstellen sind im ersten Entwicklungsschritt nicht geplant.

## Projektabgrenzung

[*ULLA*](#_Glossar) ist eine von Grund auf eigenständig, im Rahmen der praktischen Abschlussprüfung der IHK Schwaben bzw. der Beauftragung durch Atrivio programmierte Applikation ohne Abgrenzung oder Abhängigkeiten von bereits bestehenden Applikationen oder Applikationen Drittanbieter.

# Projektplanung

## Projektphasen

Das Projekt wurde innerhalb den, von der IHK Schwaben, vorgegebenen 70 Stunden durchgeführt. Eine grobe sowie eine detaillierte Zeitplanung finden sich im [Anhang A.1](#_A.1__Detaillierte).

## Abweichungen vom Projektantrag

Zum Zeitpunkt der Abgabe gab es keine Abweichung vom Projektantrag.

## Ressourcenplanung

Die benötigten Ressourcen sowie die verwendete Hard- und Software können im [Anhang A.2](#_A.2__Ressourcenplan) im Ressourcenplan werden.

## Entwicklungsprozess

Als letzter Schritt vor dem tatsächlichen Beginn des Projektes musste durch den Autor ein geeigneter Entwicklungsprozess gewählt werden. Durch diesen wird die Herangehensweise und Entwicklung bei der Umsetzung des Projektes definiert.

Grundsätzlich kann die Umsetzung im Wasserfallmodell erfolgen da die einzelnen Projektabschnitte, durch die gegebenen Rahmenbedingungen, klar definiert und konzeptioniert sind. Somit kann mit dem gewählten Umsetzungsmodell die Entwicklung präzise und zeiteffektiv durchgeführt werden. Im Bereich der Oberfläche und Prozessabläufe soll es jedoch durch regelmäßige Rücksprache mit dem Fachbereich einen agilen Prozess geben. Ebenso wird in Bezug auf die fachliche Logik und Prüfung der verarbeiteten Daten durch Plausibilitäten in iterativen Zyklen nach Erreichen von Projektmeilensteinen ein Anwendertest durch den Fachbereich schon in die Implementierungsphase integriert um den generellen Aufwand der Abnahmephase und zu minimieren.

Auf Modultest (auch von englisch unit test als Unittest oder als Komponententest bekannt) wurde aus zeitlichen Gründen verzichtet, allerdings fanden in der Entwicklungsphase und Abnahme des Projektes Codereviews mit der Ausbildungsleitung statt.

# Analysephase

## Ist-Analyse

Wie bereits unter [1.4 (Projektbegründung)](#_Projektbegründung) beschrieben stellt der aktuelle Ablauf einen wirtschaftlich und organisatorisch suboptimalen Prozess dar. Der genaue Ablauf des aktuellen Prozesses, die Akteure und Problemstellungen sollen in der folgenden Ist-Analyse genau dargestellt werden.

Am Prozess der Antragserstellung sind drei Akteure beteiligt. Der Prozess startet bisher mit dem Aufruf der Word Vorlage durch eine/en Mitarbeiter:in (Benutzergruppe: Mitarbeiter:in, Akteur 1). Auf das öffnen der Vorlage folgt die eingabe der Antragsparameter und abspeichern der Datei. Ein Ausdruck jener Datei wird anschließend unterschrieben dem Vorgesetzten (Benutzergruppe: Vorgesetzter, Akteur 2) der Abteilung vorgelegt. Entscheidet der Vorgesetzte den Antrag nicht freizugeben nimmt der Prozess ein vorzeitiges Ende. Wird der Antrag freigegeben unterschreibt der Vorgesetzte den Antrag ebenfalls und informiert den/die Antragsteller:in bzw. Mitarbeiter:in mündlich über die Freigabe. Der Unterschriebene Antrag wird dann an die Verwaltung übergeben. Ein/e Verwaltungsmitarbeiter:in vermerkt die genommen Urlaubstage in der Personalverwaltung und archiviert den Antrag, außerdem wird der/dem Antragsteller:in bzw. Mitarbeiter:in eine Kopie zur schriftlichen Bestätigung und Absicherung vorgelegt.

Im [Anhang A.3](#_A.3__Ist-Analyse) ist ein Aktivitätsdiagramm abgebildet, in dem dieser Prozess dargestellt wird.

## Wirtschaftlichkeitsanalyse

### „Make or buy“ Entscheidung

Im Vorfeld bestand bereits großes Interesse seitens Atrivio den derzeitig zweckmäßigen Workflow digital zu verbessern. Bei Lösungen Drittanbieter ergaben sich dabei kostspielige Abo Modelle oder fehlende Schnittstellen bzw. Erweiterungsmöglichkeiten als Ausschlusskriterien. Als Software-Unternehmen bzw. IT-Dienstleister lag die strategische Entscheidung der Eigenentwicklung und die Abhängigkeiten und zusätzliche Kosten von Drittanbietern bzw. Dienstleister zu vermeiden somit nahe.

### Projektkosten

In der Kalkulation wurden alle anfallenden Entwicklungskosten von [*ULLA*](#_Glossar) im Rahmen des Umsetzungsrahmen von 70 Stunden aufgefasst. Betrachtet wurden dabei neben den personellen Kosten auch alle Software-, Hardware- und sonstige Kosten (Arbeitsplatz, Klo, ect.). Genaue personelle Kosten dürfen von der Personalabteilung nicht herausgegeben werden, daher wurden die vom Controlling verwendeten Kalkulatorischen Stundensätze verwendet. Die genaue Kalkulatorische Aufstellung kann in [Anhang A.4](#_A.4__Kalkulatorische) eingesehen werden.

### Amortisationsdauer

Durch die deutliche Zeitersparnis und die zu erwartende Nutzungsdauer von mehreren Jahren ist darauf zu schließen, dass das Projekt auch unter wirtschaftlichen Gesichtspunkten sinnvoll ist.

Die folgende Amortisationsrechnung ist auf Basis der in [Anhang A.5](#_A.5__Zeiteinsparung) Zeiteinsparungsrechnung und in [Anhang A.4](#_A.4__Kalkulatorische) entstandenen Entwicklungskosten zu verstehen.

## Anwendungsfälle

Die Anwendungsfälle von ULLA wurden zur genaueren Darstellung in einem Anwendungsfalldiagramm zusammengefast.   
Das Anwendungsfalldiagramm (Use Case Diagramm) stellt das Zusammenspiel der Anwendungsfälle eines Systems untereinander und mit den Akteuren dar[[1]](#footnote-1). Obwohl das Konzept der Anwendungsfälle vollkommen unabhängig von der objektorientierten Modellierung ist, besitzt es einen festen Platz in den meisten Vorgehensmodellen für die objektorientierte Softwareentwicklung bzw. der UML[[2]](#footnote-2) und wurde somit als Grundlage für ein internes Anforderungsdokument nach dem Muster des Lastenhefts erstellt. Die Rollenverteilung des späteren Usermanagements wird durch die Vererbungshierarchie im Diagramm ebenfalls erkennbar. Das Anwendungsfalldiagramm ist im [Anhang A.6](#_A.6__Anwendungsfälle) zu finden.

## Lastenheft

Am Ende der Analysephase wurde wie in der Konzeption und [Punkt 3.3](#_Anwendungsfälle) bereits angekündigt ein Lastenheft zum Projektanstoß erstellt. Das Lastenheft dient als zentrale Anlaufstelle um alle zu berücksichtigen Anforderungen gegliedert zusammenzufassen. Ein Auszug aus diesem Dokument befindet sich im [Anhang A.7](#_A.7__Auszug).

# Entwurfsphase

## Zielplattform

Wie unter [Punkt 1.1](#_Projektbezeichnung) bereits beschrieben soll [*ULLA*](#_Glossar) als intern gehostete Webapplikation (im Folgenden als WebApp abgekürzt) entwickelt werden. Das Deployment als WebApp bringt dabei mehrere Vorteile mit sich. Durch die Webanbindung stehen allen Usern stehts dieselbe Version und somit auch Funktionen und Konfigurationen zur Verfügung. WebApp’s sind Geräteübergreifend (man spricht auch von „Cross Device“) optimiert und somit nicht an Betriebssysteme oder Bildschirmgrößen gebunden. Durch die entfallenden Abhängigkeiten durch Betriebssystemversionen (Abwärtskompatibilität) und die damit erleichterte Entwicklung sowie Fehlersuche werden zudem Entwicklungskosten gespart. Alle Projektbezogenen Daten werden auf einer – für das Projekt dediziert erstellten – MySQL Datenbank gespeichert.

Als primäres Framework der Anwendung wird das Open-Source [PHP](#_Technisches_Abkürzungsverzeichnis)-Framework [*Laravel*](#_Glossar) eingesetzt. Die Auswahl ergibt sich zum einen durch die von [Atrivio](#_Projektumfeld) vorgeschriebenen Architekturregeln, zum anderen eignet sich die Framework Architektur durch die Bindung an das [MVC](#_Technisches_Abkürzungsverzeichnis) Musters und [*Blade*](#_Glossar) Templating für die vorgesehen Anforderungen.

## Server

Anders als die Quellcodeumwandlung einer Compilerbezogenen Programmiersprache in Maschinencode ist bei WebApp’s ein Interpreter in Form eines Webservers gefragt. Von Ihm wird durch die Verwendung von [*Laravel*](#_Glossar) [PHP](#_Technisches_Abkürzungsverzeichnis)-Quellcode bei jedem Seitenaufruf eingelesen, analysiert und ausgeführt.

### Spezifikationen

Wegen der höheren Effizienz, den niedrigeren Betriebskosten und der schnellen Bereitstellung wurde vom Autor eine Virtuelle Serverumgebung (auch kurz VM für Virtuelle Maschine) gewählt. Das Virtualisierte Gastsystem baut dabei auf einem VMWare ESX Wirtsystem auf. Installiert wurde eine Ubuntu-18.04.5 Serverinstanz in Zusammenspiel mit einem Apache HTTP Server.

## Architekturdesign

Als Architekturbasis bzw. Pattern soll das in [Punkt 4.1](#_Zielplattform) bereits beschriebene [MVC](#_Technisches_Abkürzungsverzeichnis) Muster dienen. Der größte Vorteil darin liegt in der klaren Trennung der Verarbeitung von Daten in dem jeweiligen Model (z.B. Erstellen, Suchen und Löschen von Anträgen), die Darstellung der Daten im View (typischerweise [HTML](#_Technisches_Abkürzungsverzeichnis) Seiten, bzw. vor dem Rendering in Blade Templates) und der Anwendungslogik im Controller (z.B. Abfragen von Anträgen vom View und das Senden dieser ans Model). Müsst beispielsweise Datenbankstrukturen geändert werden, z.B. der Spaltenname der E-Mail-Adresse des Users wird umbenannt, so müsst diese Änderung nur in der Model-Klasse für den User geändert werden. Die Anwendungslogik (Controller) und Darstellungsschicht (View) bleibt davon unberührt.

## Benutzeroberfläche

Wie in [Punkt 2.4](#_Entwicklungsprozess) beschrieben soll es in Bezug auf die Oberfläche Rücksprache mit den Fachbereichen bzw. Projektakteuren geben. Um eine gemeinsame Diskussions- und Arbeitsgrundlage zu finden wurden erste Entwürfe angefertigt. Beispiele dieser Mockup‘s bzw. Prototypen können in [Anhang A.8](#_A.8__) und [Anhang A.9](#_A.9__Mockup) angesehen werden. Wie im Lastenheft gewünscht soll die Oberfläche ein einheitlich – minimalistisches Aussehen bekommen, wobei Farben, Schriftarten und Abstände auf die Firmen Corporate Identity abgestimmt werden. Diese Anforderungen wurden auch – bei den zuvor genannten Prototypen – berücksichtigt.

Die Anwendung wird im Intranet bzw. Internen Netz der Firma laufen. Da alle Mitarbeiter:innen an Desktopgeräten arbeiten besteht im ersten Schritt keine konkrete Anforderung an eine optimierte Responsivität der Applikation.

## Datenmodell

Das Datenmodell wurde mit Hilfe von Draw.IO[[3]](#footnote-3) als Entity-Relation-Modell (kurz. [ERM](#_Technisches_Abkürzungsverzeichnis)) entworfen. Um Übersichtlich- und Verständlichkeit zu wahren befindet sich das Diagramm ohne Attribute der Entitäten im [Anhang A.10](#_A.10_Entity_Relationship).  
Grundsätzlich teilt sich die Datenbank in zwei Anwendungsbereiche. Der erste und umfangreichste Anwendungsteil beschäftigt sich mit der Verwaltung der requests. Zusammengefasst mit allen Kardinalitäten ergeben sich sieben Tabellen die in der dritten Normalform (kurz. 3NF) das Administrative Tabellarium der Anträge stellen. Diese Entitätengruppe besteht aus der schon genannten requests Tabelle sowie, request\_periods, request\_types, comments, human\_resources und user\_stand\_ins.

Jeder request ist dabei einem festen request\_type zuzuordnen (Bspw. „Bezahlter Urlaub“, „Sonderurlaub“, ect.) dadurch entsteht eine 1:n-Beziehung.  
Beim Anlegen eines requests muss in der Tabelle request\_periods unter anderem der Start und End Timestamp des Antrags gesetzt sein, dadurch besteht auch hier eine 1:n-Beziehung.

In humane\_resources werden dann zunächst die user\_id des Erstellers und des zugewiesenen Vorgesetzen eingetragen. Anschließend werden über die user\_stand\_ins beliebig viele, bzw. in der Erstellungsmaske eingetragene Vertretungen verknüpft.   
Die Verbindungsstelle beherbergt dabei die user\_id, humane\_resource\_id und den sog. overhanding\_tstmp welcher das Übergabedatum des Antragstellers an die Vertretung(en) festlegt.   
Die requests Tabelle steht somit ebenfalls in einer 1:n-Beziehung zur request\_periods Tabelle, wobei die user\_stand\_ins eine n:m-Beziehung zur request\_period Entität pflegt.

Mit der comments Tabelle können alle Status Änderungen der requests durch die Akteure kommentiert werden.  
Grundsätzlich stellt die Auslagerung der 1:1 Beziehung unter Gesichtspunkten der Normalisierung keine Notwendigkeit da. Jedoch sind mit steigender Auslastung der Datenbank aufgrund der Maximalen Beschreibungsgröße von 255 Zeichen pro Datensatz in der comments Entität die größten Datensätze der Applikation zu erwarten. Somit macht die Auslagerung aus Performancetechnischer Sicht Sinn.

# Literaturverzeichnis

Prof. Dr. Gert Heinrich, Klaus Mairon (2008) *Objektorientierte Systemanalyse,*   
Oldenbourg Wissenschaftsverlag Gmbh, ISBN 978-3-486-58366-3

Dr. Dr. h. c. Niklaus Wirth *Algorithmen und Datenstrukturen*,  
B.G. Teubner Stuttgart, ISBN 3-519-02250-8

Matt Stauffer (2017) *Laravel: Up and Running*,  
O’Reilly Books USA

Dietmar Stoiber (2005) *Implementierung des Model-View-Controller Paradigmas*Diplomarbeit Universität Linz

# A Anhang

## A.1 Detaillierte Zeitplanung

Auszug aus dem Projektkonzept:

|  |  |
| --- | --- |
| Analyse | 3.5h |
| Durchführung einer Ist-Analyse | 1.0h |
| Wirtschaftlichkeitsprüfung, (z.b Amortisationsrechnung & Kostenanalyse) | 1.0h |
| Erstellung eines vereinfachten Lastenhefts | 1.0h |
| Ermittlung des Use – Case (bzw. Use – Case Diagramm) | 0.5h |
| Entwurf | 8.0h |
| Erstellung eines vereinfachten Pflichtenhefts | 3.0h |
| Aktivitätsdiagramm zu den einzelnen Projektsegmenten | 1.0h |
| Frontend-Mockup | 1.0h |
| Planung des Datenbankmodells | 1.5h |
| ER-Diagramm o. DB Schema | 0.5h |
| Entwicklung & Implementierung | 42.5h |
| Implementierung des Datenbankmodells bzw. der Datenbanklogik | 2.0h |
| Erstellung des Antrags-Interface | 5.0h |
| Erstellung des Management-Interface | 4.0h |
| Erstellung des Dashboards | 4.0h |
| Erstellung des Verwaltungs-Interface | 4.0h |
| Entwicklung der Zuweisungslogik | 2.0h |
| Entwicklung der Mail-Notifications | 3.0h |
| Entwicklung der Abnahme Logik | 2.0h |
| Entwicklung der Ablehnungs Logik | 3.0h |
| Einrichtung der Routen | 1.5h |
| Frontend Erstellung nach Mockup | 6.0h |
| Implementierung der Benutzer Authentifikation & Rechte | 6.0h |
| Abnahme und Deployment | 4.0h |
| Code-Review mit dem Ausbilder u. ggf Fachabteilung(en) | 1.0h |
| Abnahme durch einen Ausgewählten Fachbereich | 1.0h |
| Deployment der Applikation | 2.0h |
| Dokumentation | 13.0h |
| Erstellen der Projektdokumentation | 12.0h |
| Erstellen der Entwicklungsdokumentation u.a. für zukünftige Erweiterungen. Die Projektdokumentation wird hier als Basis verwendet u. ggf. erweitert. | 1.0h |
| Gesamt | **70h** |

## 

## A.2 Ressourcenplan

|  |  |
| --- | --- |
| Typ | Bezeichnung |
| **Hardware** | Tisch |
|  | Stuhl |
|  | Notebook |
|  | Monitor |
|  | Tastatur |
|  | Maus |
|  | Headset |
| **Software** | macOS Big Sur 11.2.3 |
|  | Ubuntu 20.04.02 LTS |
|  | Ubuntu Server 18.02 |
|  | VMWare ESX |
|  | Docker desktop 3.3.1 |
|  | Docker Container mysql/mysql-server (lokal) |
|  | Docker Container dockage/mailcatcher (lokal) |
|  | GIT (CLI) |
|  | Jetbrains Datagrip |
|  | Microsoft Visual Studio Code |
|  | Microsoft Visual Studio Code UMLet Extension |
|  | Microsoft Teams |
|  | Composer (CLI) |
|  | NPM (CLI) |
|  | iTerm (im Verbund mit ZSH) |
| **Personal** | Leiter der Entwicklung Kundenentwicklung, Ausbildungsleiter |
|  | Mitarbeiter:in der Verwaltung |
|  | Mitarbeiter:in Testing bzw. Projektleitung |
|  | Anwendungsentwickler, Prüfling |

## Diagram Description automatically generatedA.3 Ist-Analyse - Aktivitätsdiagramm

## A.4 Kalkulatorische Aufstellung

|  |  |  |  |
| --- | --- | --- | --- |
| Position | Kosten | Stunden | Gesamt |
| Herr Müller (Auszubildender Entwicklung, Prüfling) | 65€ | 70h | 4.550,00€ |
| Herr Tamler (IT Consultant / Head of Development) | 130€ | 8h | 1.040,00€ |
| Frau Klasen (Auszubildende Projektleitung, Testing) | 65€ | 2h | 130,00€ |
| Gesamt | | | 5.720,00€ |

## 

## A.5 Zeiteinsparung

|  |  |  |  |
| --- | --- | --- | --- |
| Stelle | Kosten in € pro Stunde | Ersparnis in Minuten | Gesamt |
| Mitarbeiter | 85 \* | 10 | 14,17€ |
| Vorgesetzer | 85 \* | 5 | 7.08€ |
| Verwaltung | 85 \* | 7 | 9.92€ |
| Ersparnis pro Antrag | | | 31,17€ |

\* Durchschnittswerte

Berechnung:

## A.6 Anwendungsfälle – Use Case Diagramm

Diagram

Description automatically generated

## A.7 Auszug Lastenheft



## A.8 Mockup - Formular

Graphical user interface

Description automatically generated

## A.9 Mockup - Dashboard

Diagram

Description automatically generated with low confidence

## A.10 Entity Relationship Modell

Diagram

Description automatically generated

1. Vgl. [Literaturverzeichnis](#_Literaturverzeichnis): *Objektorientierte Systemanalyse*, S. 59 [↑](#footnote-ref-1)
2. Vg. [Literaturverzeichnis](#_Literaturverzeichnis): *Objektorientierte Systemanalyse*, S. 58 [↑](#footnote-ref-2)
3. Kostenlose Diagramm WebApp aus dem Atlassian Ökosystem [↑](#footnote-ref-3)