

ABSCHLUSSPRÜFUNG SOMMER 2022

AUSBILDUNG ZUM FACHINFORMATIKER

FACHRICHTUNG: ANWENDUNGSENTWICKLUNG

PRÜFLING

Eugen Maksymenko Schillerstraße3

90409 Nürnberg

Identnummer: 3608904

AUSBILDUNGSBETRIEB

SUSE Software Solutions Germany GmbH Maxfeldstraße 5

90409 Nürnberg



SUSE

Dokumentationstools für die Verwaltung und Erstellung von wöchentlichen Berichten

ABGABEDATUM: 15. MAI 2022

Inhaltsverzeichnis

[1. Einleitung 1](#_Toc103117023)

[1.1 Vorwort 1](#_Toc103117024)

[1.2 Projektname 1](#_Toc103117025)

[2. Definitionsphase 1](#_Toc103117026)

[2.1 Projektumfeld 1](#_Toc103117027)

[2.2 Auftraggeber 1](#_Toc103117028)

[2.3 Problemstellung 1](#_Toc103117029)

[2.4 Projektziel 2](#_Toc103117030)

[2.5 Projektabgrenzung 2](#_Toc103117031)

[2.6 IST-Analyse 2](#_Toc103117032)

[2.7 Soll-Konzept 3](#_Toc103117033)

[3. Planungsphase 3](#_Toc103117034)

[3.1 Zeitplan 3](#_Toc103117035)

[3.2 Ressourcenplanung 3](#_Toc103117036)

[3.2.1 Hardware 3](#_Toc103117037)

[3.2.2 Software 4](#_Toc103117038)

[3.2.3 Personal 4](#_Toc103117039)

[3.3 Projektkosten 4](#_Toc103117040)

[3.4 Amortisation/Vorteile 4](#_Toc103117041)

[4. Entwurfsphase 5](#_Toc103117042)

[4.1 Kommunikation zwischen Benutzter und dem Tool 5](#_Toc103117043)

[4.1.1 Interaktion ohne Kommandozeilenargumente 5](#_Toc103117044)

[4.1.2 Interaktion mit Kommandozeilenargumenten 6](#_Toc103117045)

[4.2 Datenbank 6](#_Toc103117046)

[4.3 Layout des Berichts 7](#_Toc103117047)

[4.4 Module 7](#_Toc103117048)

[4.4.1 Kommandozeile Modul 7](#_Toc103117049)

[4.4.2 Datenbank Modul 7](#_Toc103117050)

[4.4.3 Bericht Modul 7](#_Toc103117051)

[4.5 Planung der Testszenarien 7](#_Toc103117052)

[4.6 Pflichtenheft 8](#_Toc103117053)

[5. Implementierung 8](#_Toc103117054)

[5.1 Kommandozeilen Modul 8](#_Toc103117055)

[5.1.1 Kommandozeilen Interaktion mit Menü ohne Unterbefehle 8](#_Toc103117056)

[5.1.2 Kommandozeilen Interaktion ohne Menü mit Unterbefehl 9](#_Toc103117057)

[5.2 Datenbank Modul 10](#_Toc103117058)

[5.3 Bericht Modul 10](#_Toc103117059)

[5.4 Vorlage für das Bericht Modul 11](#_Toc103117060)

[6. Projektkontrolle 11](#_Toc103117061)

[6.1 Zeitlicher Soll-Ist-Vergleich 11](#_Toc103117062)

[6.2 Zielerreichungsvergleich 12](#_Toc103117063)

[6.3 Kostenvergleich 12](#_Toc103117064)

[6.4 Testdurchführung 12](#_Toc103117065)

[6.4.1 Code Review als White-Box-Test 12](#_Toc103117066)

[6.4.2 Systemtest als Black-Box-Test 12](#_Toc103117067)

[7. Projektabschluss 13](#_Toc103117068)

[7.1 Übergabe an Kunden/Team 13](#_Toc103117069)

[7.2 Fazit 13](#_Toc103117070)

[7.2.1 Projekt umgesetzt 13](#_Toc103117071)

[7.2.2 Zufriedene Kollegen 13](#_Toc103117072)

[7.2.3 Gelerntes Wissen 13](#_Toc103117073)

[7.3 Ausblick 13](#_Toc103117074)

[8. Glossar 14](#_Toc103117075)

[HTML 14](#_Toc103117076)

[Codereview 14](#_Toc103117077)

[CSS 14](#_Toc103117078)

[Anhang 14](#_Toc103117079)

[Anhang A: Mockup Erstes Berichts Layout 14](#_Toc103117080)

[Anhang B: Mockup Endgültiges Berichts Layout 15](#_Toc103117081)

[Anhang C: Detaillierte Zeitplanung 15](#_Toc103117082)

[Anhang D: Datenbank Modell 16](#_Toc103117083)

[Anhang E: Klassendiagramm 17](#_Toc103117084)

[Anhang G: Zusätzliche Bearbeitungszeit 17](#_Toc103117085)

**Abbildungsverzeichnis**

[Abbildung 1 Grobe Zeitplannung 3](#_Toc103116301)

[Abbildung 2 Mockup Hauptmenü 5](#_Toc103116302)

[Abbildung 3 ER Modell Datenbank 6](#_Toc103116303)

[Abbildung 4 Implementierung des Menü Interfaces 8](#_Toc103116304)

[Abbildung 5 Implementierung eines Unterbefehls mit zwei Argumenten 8](#_Toc103116305)

[Abbildung 6 Implementierung der privaten Methode für die Tabellenerstellung 9](#_Toc103116306)

[Abbildung 7 Implementierung eines Jinja2 Dictionary 9](#_Toc103116307)

[Abbildung 8 HTML Vorlage 10](#_Toc103116308)

[Abbildung 9 Soll-IST-Vergleich 10](#_Toc103116309)

[Abbildung 10 Datenbank Modell 15](#_Toc103116310)

[Abbildung 11 Database Klassendiagramm 16](#_Toc103116311)

**Tabellenverzeichnis**

Tabelle 1 Kategorien im wöchentlichen Bericht 2

Tabelle 2 Benutzte Software 4

Tabelle 3 Projektkosten 4

Tabelle 4 Pflichtenheft 7

Tabelle 5 Zielerreichungsgvergleich 11

Tabelle 6 Kostenvergleich 11

Tabelle 7 Detaillierte Zeitplanung 15

# 1. Einleitung

Das Projekt „Dokumentationstools für die Verwaltung und Erstellung von wöchentlichen Berichten“ wurde vom Autor dieser Dokumentation, Herrn Eugen Maksymenko im Rahmen der Abschlussprüfung zum Fachinformatiker für Anwendungsentwickler im Sommer 2022 durchgeführt.

Die Dokumentation wurde mit Microsoft Word erstellt.

## 1.1 Vorwort

Innerhalb dieser Dokumentation sind *kursiv* formatierte oder abgekürzte Begriffe zu finden, welche bei erster Nennung ausführlich und alphabetisch im Glossar aufgeführt werden

## 1.2 Projektname

Bei diesem Projekt handelt es sich um ein „Open Source“ Projekt. Darunter ist zu verstehen das jeder Mensch dieses Projekts für jeden Zweck verwendet kann. Es ist erlaubt das Projekt zu inspizieren, modifizieren und weiterverbreiten. Nach der vollen Fertigstellung dieses Projektes ist es geplant das Projekt auf GitHub zu veröffentlichen. Da jedes gute Projekt einen guten Namen erfordert und da „Dokumentationstools für die Verwaltung und Erstellung von wöchentlichen Berichten“ nicht sprechend ist, wird das Projekt den Titel Reportrick erhalten. Auch innerhalb dieses Projektdokumentation wird man stets sich auf den Titel Reportrick beziehen.

# 2. Definitionsphase

## 2.1 Projektumfeld

Die SUSE Software Solutions Germany GmbH, kurz SUSE, ist ein im Jahr 1992 in Fürth gegründetes Softwareunternehmen und entwickelt unter anderem eigene Linux Distributionen mit dem Namen SUSE Linux Enterprise (SLE) und der Binärkompatiblen Community Version openSUSE. Eine Linux Distribution ist eine Zusammenstellung von über 5000 Programmen und Bibliotheken, die auf einander abgestimmt sind und ein Allzweckbetriebssystem bilden.  
Bei SUSE arbeiten weltweit ca. 2200 Mitarbeiter an verschiedenen Standorten auf unterschiedlichen Kontinenten, z.B. in Beijing, Vancouver und Prag. Die Kommunikation zwischen Mitarbeitern findet unter anderem im „Slack“ statt. Slack ist ein webbasierter Instant-Messaging-Dienst. Dieses Chatsystem unterstützt moderne Features wie einen serverseitigen Nachrichtenverlauf, Desktopbenachrichtigungen und die Aufbereitung von Nachrichten.

Die Gesamte Projektumsetzung findet in der SUSE Software Solutions Germany GmbH im Standort Nürnberg statt.

## 2.2 Auftraggeber

Der Auftraggeber dieses Projektes ist das Dokumentation Team der SUSE. Dieses Team entwickelt das openSUSE DAPS (DocBook Authoring and Publishing Suite) Projekt und schreibt Dokumentation für die SUSE und ihre große Anzahl an Projekten. Der Teamleiter des Dokumentationsteams ist der Frank Sundermeyer. Auftraggeber und Betreuer des Projekts ist ein Teammitglied des Dokumentationsteams, Herr Thomas Schraitle.

## 2.3 Problemstellung

Aktuell muss jeder Mitarbeiter eine eigene Lösung suchen, für die Erfassung seiner Aktivität im  
Team. Das hat den großen Nachteil, dass dadurch oft Details und Tätigkeiten verloren gehen, da  
die Berichte meistens erst gegen Ende der Woche aus dem Gedächtnis angefertigt werden.

Uneinheitliche Berichte erschweren und verlangsamen zusätzlich das Zusammenfassen und die Erstellung eines einheitlichen Berichtes für den Team Leiter.

## 2.4 Projektziel

Aufgrund des Problems aus 2.3 entstand dieses Projekt.

Das Ziel dieses Projektes besteht darin ein Kommandozeilentool für die Erfassung der  
wöchentlichen Tätigkeiten zu entwickeln.

Das Tool soll Einträge in eine lokale Datenbank abspeichern damit auch bei Bedarf nachträglich auf die Berichte zugegriffen werden kann.  
Der Nutzer kann direkt über Kommandozeilenargumente Beiträge hinzufügen, löschen und  
ausgeben. Die Beiträge werden in vordefinierte Kategorien für eine Kalenderwoche abgespeichert. Die so erfassten Daten werden am Ende der Woche zu einem Wochenbericht im gewünschten Format (HTML/PDF/TEXT), generiert.

## 2.5 Projektabgrenzung

Die Funktion einer automatisierten E-Mail-Versendung nach Berichtserstellung ist nicht ein Teil dieses Projektes.

Die Erstellung eines offiziellen RPM Paketes für die openSUSE ist auch nicht ein Teil dieses Projektes.

## 2.6 IST-Analyse

Nach mehreren Gesprächen mit verschiedenen Teamleitern der SUSE Software Solutions Germany GmbH und der Auswertung eines, von Teammitgliedern ausgefüllten Fragebogens.

Ergibt sich die folgende Situation:

Mitarbeiter müssen in ihrem Team einen wöchentlichen Bericht für die aktuelle Kalenderwoche anfertigen. Der Bericht enthält Einträge mit ihren Tätigkeiten in verschiedenen Kategorien.

|  |  |  |
| --- | --- | --- |
| **Kategorie** | **Bedeutung** | **Beispiel** |
| RED | Problembehaftete Tätigkeiten und „Stopper“ | Merge mit Feature XY. Paket XYZ noch nicht upgedatet, dadurch kein Release bis. |
| AMBER | Problematischen Tätigkeiten | Internetausfall |
| GREEN | Erledigte Tätigkeiten, die normal verlaufen sind. | Github commit im Projekt XY |
| MEETING | Meetings, Team Events, Besprechungen | Codereview mit Mitarbeiter XY |
|  |  |  |

Tabelle 1 Kategorien im wöchentlichen Bericht

Dieser ist jeden Freitag bis 13 Uhr abzusenden.

Der Inhalt Bezieht sich immer auf die Aktuelle Kalenderwoche von Montag-Freitag.

Nach der Erstellung, sendet der Mitarbeiter seinen Bericht per E-Mail an den Teamleiter. Dieser Teamleiter liest jeden einzelnen Wöchentlichen Bericht. Alle Berichte der Mitarbeiter werden vom Teamleiter zu einem gemeinsamen Bericht erfasst. Dadurch existiert ein schriftlicher Überblick über die Aktuelle Situation im Team. Diese Zusammenfassung verbraucht 0.5-2 Stunden durch die variable Team Größe und aktuellen Auslastung. Das sogenannte Teambericht (Zusammenfassung und Sortierung der Mitarbeiter Berichte) wird an das höhere Management gesendet. Das Teambericht enthält eine Liste an aktuellen Problemen, Stopper oder mögliche/zukünftige Komplikationen. Außerdem werden besondere Leistungen von Mitarbeitern in diesem Bericht hervorgehoben und Festgehalten. Ein Tool der die Erfassung der Tätigkeiten und automatische Erstellung der Berichte wird dringend gewünscht. Einheitliche Berichte eröffnen viele weiteren Optionen zur Automatisierung dieses Prozesses.

## 2.7 Soll-Konzept

Ein Open Source Kommandozeilen Tool zum Erfassen der Tätigkeiten und Generierung eins wöchentlichen Berichtens soll realisiert werden.

Die Anpassung...

…muss ein Kommandozeilen Menü anbieten.

…muss Interaktion direkt durch Kommandozeilen Argumente anbieten.

…muss eine Datenbank Anbindung haben.

…muss eine HTML und CSS-Template besitzen haben.

…muss automatisiert ein Bericht erstellen können in mehreren Formaten (HTML/PDF/TEXT)

…muss modular aufgebaut sein damit weitere Funktionalität „einfach“ erweiterbar sind.

# 3. Planungsphase

## 3.1 Zeitplan

Für das Abschlussprojekt stehen insgesamt 70 Stunden zur Verfügung. Um einen Überblick über den zeitlichen Aufwand zu erhalten, wurde das Projekt in folgende Phasen unterteilt, welche während des Projektes durchlaufen werden. Die grobe Zeitplanung lässt sich aus der folgen Grafik entnehmen.

Abbildung 1 Grobe Zeitplannung

Die Detaillierte Zeitplan ist im Anhang Table 7 Dataillierte Zeitplannung zu finden.

## 3.2 Ressourcenplanung

Eine Sammlung aller benötigten Ressourcen für eine Erfolgreiche Durchführung des Projekts

### 3.2.1 Hardware

Büroarbeitsplatz mit Laptop „Dell Latitude 5300 2-in-1“

### 3.2.2 Software

|  |  |
| --- | --- |
| **Name** | **Verwendungszweck** |
| OpenSUSE Tumbleweed | Betriebsystem |
| Visual Studio Code | Quelltext-Editor |
| Git | Versionskontrolle |
| GitHub | Webbasierter Filehosting-Service für die Softwareentwicklung und das Teilen von Softwarecode |
| Python | Programmiersprache zur Programmlaufzeit |
| Jinja2 | Web-Template-Engine für die Programmiersprache Python |
| HTML | Textbasierte Auszeichnungssprache zur Strukturierung elektronischer Dokumente wie Texte mit Hyperlinks |
| CSS | Stylesheet-Sprache für elektronische Dokumente und zusammen mit HTML |
| SQLite | Datenbankmanagementsystem für die Entwicklung |
| Draw.io | Erstellung von Diagrammen und Mockups |
| Microsoft Word | Erstellung der Dokumentation und Tabellen |

Tabelle 2 Benutzte Software

### 3.2.3 Personal

Mitarbeiter– Absprache, Festlegung der Anforderungen Konzeption und Entwicklung

Entwickler – Code review

## 3.3 Projektkosten

In diesem Abschnitt werden die voraussichtlichen Kosten berechnet, die bei der Durchführung des Projektes anfallen. Da es sich um ein Firmen internes Projekt handelt kann man keine pauschalen Kosten angeben die normalerweise einem Kunden in Rechnung gestellt werden.

An diesem Projekt arbeitet ein Prüfling im dritten Lehrjahr voraussichtlich 70h.  
Nach einer Anfrage in der Human Resources Abteilung des Unternehmens verbraucht der Azubi 9,34€ pro Stunde. Darin sind pauschale Gemeinkosten wie Stromkosten, Mietkosten ebenfalls enthalten. Nach der Umsetzung wird ein *Codereview* von Herr Thomas Schraitle durchgeführt. Der interne Stundensatz von einem Entwickler beträgt durchschnittlich 30 €.

Tabelle 3 Projektkosten

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Vorgang | Mitarbeiter | Stundensatz | Zeit | Kosten |
| Entwicklungskosten | 1x Auszubildender | 9.34 € | 70 h | 653.80 € |
| Codereview | 1x Mitarbeiter | 30.00 € | 2 h | 60.00 € |
| **Gesamtkosten** | | | | 713.80 € |

## 3.4 Amortisation/Vorteile

Da das Projekt nicht in direktem Bezug zum Umsatz steht, lässt sich über die Amortisierung der Kosten keine konkrete Aussage treffen.  
Es lassen sich jedoch offensichtliche Vorteile für das Unternehmen ableiten:

* Die Teamleiter können den aktuellen Status schneller einschätzen und in der Folge darauf präventiv auf Unzugänglichkeiten reagieren
* Die Management-Ebene kann durch vollständige und zusammengefasste Berichte viel schneller auf Probleme aufmerksam gemacht werden.
* Die Mitarbeiter des Teams können das neue Tool viel einfacher in ihren Workflow einsetzten

Aufgrund dieser Vorteile ist abzusehen, dass sich die Verbesserung zügig amortisieren wird, weil sie Arbeitsabläufe vereinfacht und zu einer Vereinfachung des Workflows führt.

# 4. Entwurfsphase

In dieser Phase wird festgelegt wie die Kommunikation zwischen Benutzter und dem Berichtstool stattfinden soll. Dabei unterscheiden wir nochmal die Interaktion des Anwenders mit und ohne Kommandozeilenargumente. Der nächste große Schritt ist der Entwurf einer relationalen Datenbank für die persistente Speicherung von Daten. Um einheitliche Berichte zu erfassen ist ein Template mit Layout erforderlich welches mit den Werten aus der Datenbank befüllt werden kann. Zuletzt wurde eine Aufteilung in Module vorgenommen und die Schnittstellen zwischen den einzelnen Modulen entworfen. Die Ergebnisse der Entwurfsphase sind in den folgenden Abschnitten aufgeführt.

## 4.1 Kommunikation zwischen Benutzter und dem Tool

Der Nutzer kommuniziert mit dem Tool direkt per Kommandozeile. Wird das Programm ohne Kommandozeilenargumente gestartet, so wird auch das Hauptmenü aufgerufen.

Die Interaktion mit dem Tool bezieht sich standartmäßig immer auf die aktuelle Woche. Es ist möglich aber auch auf Berichte aus der Vergangenheit zuzugreifen.   
Siehe 4.1.1.2 Wechsel der Kalenderwoche oder des Jahres

Eine Alternative Methode mit dem Programm zu arbeiten soll eine direkte Übergabe an Befehlen mit Parametern.

### 4.1.1 Interaktion ohne Kommandozeilenargumente

Im Kommandozeileninterface wird der Nutzer begrüßt. Aktuelles Datum und die jetzige Kalenderwoche sollen angezeigt werden. Darunter ist ein Hauptmenü für die Interaktion zu sehen. Durch simple Nutzer Eingabe soll der User durch das Interface navigieren können.

In der folgenden Abbildung ist ein Beispiel für das Hauptmenü dargestellt.

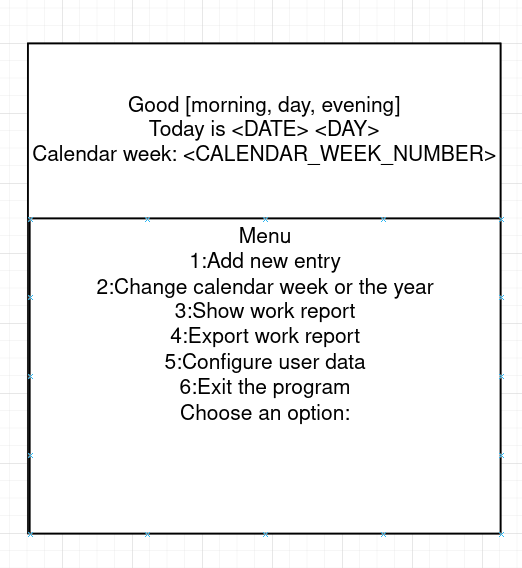


Abbildung 2 Mockup Hauptmenü

Die genauere Funktionsweise der einzelnen Punkte wird unten festgehalten

#### 4.1.1.1 Hinzufügen der neuen Einträge

Unter Nummer 1 kann die Option „Add new entry“ ausgewählt werden. Das beinhaltet das

hinzufügen von neuen Beiträgen. Beim Auswählen von dieser Option wird der Nutzer aufgefordert dein Eintrag in die Kommandozeile einzutragen. Nach erfolgreicher Eingabesoll der Anwender entscheiden unter welcher Kategorie er den eingegebenen Eintrag abspeichern will. Nach einer Kategorieneingabe wird der Eintrag für die eingestellte Kalenderwoche/Jahr in der Datenbank abgespeichert. Der Nutzer kriegt eine Rückmeldung zu den abgespeicherten Daten.

Danach kann der Benutzter zum Hauptmenü zurückkehren oder direkt das Programm beenden.

#### 4.1.1.2 Wechsel der Kalenderwoche oder des Jahres

Unter Nummer 2 kann die Option „Change calendar week or the year“ ausgewählt werden.

Damit kann der Nutzer bei Bedarf auf alte Berichte aus der Datenbank zugreifen.   
Der Nutzer soll ein Jahr und eine Kalenderwoche eingeben. Danach kehrt der Nutzer zurück ins Hauptmenü. Im Hauptmenü ist ein Hinweis zu sehen der betonnt das der Benutzer auf einen Bericht aus der Vergangenheit zugreift. Die normale Interaktion mit dem Tool kann weiterhin geführt werden mit dem Unterschied das man auf den alten Bericht zugreift und nicht auf die aktuelle Kalenderwoche. Das beenden des Programms setzt immer das Programm wieder auf das aktuelle Jahr und die aktuelle Kalenderwoche zurück.

#### 4.1.1.3 Einsicht in den aktuellen Wochenbericht

Unter Nummer 3 kann die Option („Show work report“) ausgewählt werden.

Beim Aufruf werden alle Einträge für diese Woche kategorisiert ausgegeben. Die Kategorien sollen farblich dargestellt werden für eine bessere Lesbarkeit. Der Benutzer bekommt die Möglichkeit das Programm zu beenden und zum Hauptmenü zurückzukehren. Da der Nutzer die Ansicht aller Einträge vor sich hat, darf er eine weitere Möglichkeit auswählen und zwar Einträge zu löschen.

Durch die Auswahl dieser Option im Untermenü wird der Anwender aufgefordert eine Kategorie auszusuchen, in welcher der zu löschende Eintrag sich befindet. Durch die Auswahl werden aus der Datenbank alle Einträge in der zugehörigen Kategorie angezeigt. Der Nutzer muss den zu löschenden Eintrag eingeben. Somit soll verhindert werden, dass ein unbeabsichtigter Eintrag gelöscht wird. Nach der Bestätigung dieser Eingabe wird der Eintrag gelöscht und der Anwender bekommt eine aktualisierte Ansicht aller Einträge. Bei Bedarf kann er diesen Vorgang wiederholen, das Programm beenden oder ins Hauptmenü zurückkehren.

#### 4.1.1.4 Erstellung des Wochenberichtes

Unter Nummer 4 kann die Option („Export work report“) ausgewählt werden. Diese Funktion dient dazu den Wochenbericht für die aktuell ausgewählte Kalender Woche und Jahr zu erstellen.

Nach der Auswahl dieser Option wird der Benutzer gefragt, in welchem Format dieser Bericht erstellt werden soll. Ein Untermenü wird auf der Kommandozeile ausgegeben wo der Anwender zwischen drei benötigten Formaten, wie HTML, PDF und Text entscheiden. Nach der Selektierung des Formates werden alle Einträge aus der Datenbank zu einem Bericht erstellt. Eine Rückmeldung soll den Pfad angeben in dem der Wochenbericht abgespeichert worden ist.

#### 4.1.1.5 Konfiguration der Nutzerdaten

Unter Nummer 5 kann die Option („Configure user data“) ausgewählt werden. Der Nutzer kann hier seine eigenen Daten wie Vornamen, Nachname und das Team in dem er Arbeitet festhalten. Wenn der Nutzer eine Konfiguration vornimmt, dann werden diese Daten auch in den Bericht aufgenommen.

#### 4.1.1.6 Beenden des Programms

Unter Nummer 6 kann die Option („Exit the program“) ausgewählt werden. Das Programm wird beendet.

### 4.1.2 Interaktion mit Kommandozeilenargumenten

Die Befehle des Nutzers an das Tool müssen einer festgelegten Syntax einhalten, damit das Tool diese als Befehl interpretieren kann. Der Befehl „help“ zeigt dem Nutzer eine Bedienungsanleitung, die alle verfügbaren Befehle auflistet und beschreibt. Es soll möglich seine Einträge zu hinzufügen und zu löschen. Der Nutzer soll Konfigurierbar sein und natürlich die Exportierung des Berichtes.

Aufruf von vergangenen Wochenberichten ist in dieser Interaktionsart erstmalig nicht geplant.

## 4.2 Datenbank

Das Tool benötigt eine eigene Datenbank, in welcher alle Nutzer Daten und Einträge abgespeichert werden können.  
Abbildung 3 zeigt ein ER Diagramm ohne Attribute.

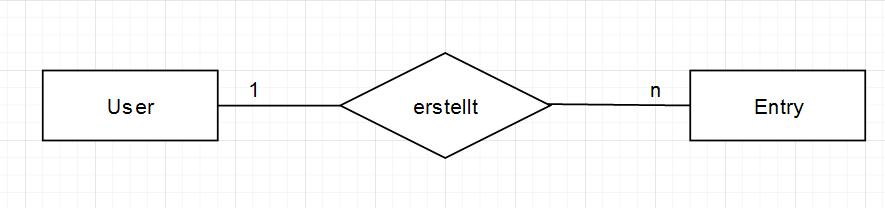


Abbildung 3 ER Modell Datenbank

Eine genaue Darstellung mit dem ganzen Anhang ist in Abbildung 4 Datenbank ER Diagramm enthalten

## 4.3 Layout des Berichts

Für den wöchentlichen Bericht ist eine Vorlage notwendig. In dieser Struktur sollen Platzhalter fest definiert werden, die dynamisch mit Einträgen aus der Datenbank befüllt werden können.

Nach der Aufnahme der Kundenwünsche wurde ein Mockup für das Layout angefertigt.

Nach Rücksprache mit dem Auftraggeber wurden einige Änderungen vorgenommen.

Dadurch entstand dann das finale Layout.  
Anhang: Abbildung 7 Final Mockup

## 4.4 Module

Zur besseren Wart- und Erweiterbarkeit wird Reportrick in Programmmodule aufgeteilt. Mehrere kleinere Module können leichter verstanden und verändert werden, als eine große Datei, in der die gesamte Programmlogik enthalten ist.

### 4.4.1 Kommandozeile Modul

Das „Main“ Modul beinhaltet die Logik für die Interaktion mit der Kommandozeilen und aller ihrer Funktionalität. Beim Programstart wird dieses Modul aufgerufen.

### 4.4.2 Datenbank Modul

Das Datenbank Modul enthält die Logik der Datenbank in einer Klasse. Siehe Beispiel unter Abbildung 5 Database Klassendiagramm.

### 4.4.3 Bericht Modul

Das Bericht Modul ist für die Erstellung von Berichten in verschiedenen Formaten verantwortlichen. Durch die Auslagerung, ist bei Bedarf die Erweiterung in zusätzliche Formate ohne Probleme möglich.

## 4.5 Planung der Testszenarien

Um möglichst effiziert in der Testphase zu sein, werden die Testszenarien bereits in der Planungsphase entworfen. Dazu werden die erwarteten Funktionalitäten festgelegt. Diese Punkte werden in der Testphase als Black-Box-Test durchlaufen und die Ergebnisse werden im Feld „Ergebnis“ protokolliert. Ein Beispiel der geplanten Testszenarien lässt sich aus der folgenden Tabelle entnehmen

|  |  |  |
| --- | --- | --- |
| Anpassung | Erwartete Funktionalität | Ergebnis |
| Hinzufügen von Daten in der Datenbank | Die Daten werden korrekt abgespeichert. |  |
| Löschen der Daten in der Datenbank | Die Daten werden erfolgreich aus der Datenbank gelöscht |  |
| Erstellung des Berichtes | Die Datei wurde erstellt |  |
| Überprüfung der Einträge | Der Bericht hat alle Einträge aus der Datenbank eingespeichert. |  |
| Layout Format einhalten | Überprüfung der Schriftgröße, Zeilenumbrüche und des Layouts. |  |

Tabelle 1 Beispiel Testszenarien

## 4.6 Pflichtenheft

Im Pflichtenheft werden alle Punkte festgehalten, welche umgesetzt werden. Zusätzlich wird es ebenfalls beschrieben, wie diese Punkte realisiert werden. Ein Auszug aus dem Pflichtenheft lässt sich aus der unterstehenden Tabelle entnehmen:

|  |  |
| --- | --- |
| Funktionalität | Umsetzung |
| Hinzufügen und Entfernen von Einträgen | Die Daten sollen über die Kommandozeile erfasst werden und in einer SQLITE Datenbank abgespeichert. Einträge können gelöscht werden. |
| Zugriff auf alte Berichte | In der Kommandozeile können Bericht aus vergangener Zeit aus der Datenbank aufgerufen werden. Hinzufügen/Löschen und Erstellung ist wiederholt möglich. |
| Anzeige aller Einträge für den aktuellen Bericht | Die Kommandozeile hat eine Unterfunktion zur Anzeige von Einträgen für die gewählte Woche. |
| Erstellung von Berichten in HTML/PDF oder Text | Die Kommandozeile hat eine Unterfunktion für die Erstellung eines Berichtes. Dafür wird eine HTML Vorlage für die Struktur und eine CSS-Vorlage für das Design und Formatierung genutzt. |
| Konfiguration von Nutzer Daten | Der Anwender hat eine Unterfunktion wo der Nutzer Konfiguriert wird und diese Daten in der Datenbank abgespeichert werden. |

Tabelle 4 Pflichtenheft

# 5. Implementierung

Reportrick wird in der Programmiersprache Python programmiert, da diese frei und für alle gängigen Betriebssysteme verfügbar ist. Python ist universell, üblicherweise interpretiert und eine höhere Programmiersprache, die alle Werkzeuge mit sich bringt um dieses Projekt erfolgreich durchzuführen. Eins der benötigten Werkezuge ist das Modul SQLite, welches eine dateibasierte Datenbank ist. Durch die geringe Größe kann Reportrick dadurch auf eine ausgewachsene Datenbank zugreifen. Dadurch lässt sich das Datenbanksystem vollständig in die Anwendung integrieren. Zuletzt bietet uns das Python Modul Jinja2, eine sehr schnelle Template Engine, die Möglichkeit aus einer HTML Vorlage ein Bericht zu erzeugen. Dazu werden spezielle Platzhalter in der HTML Vorlage genutzt um ein finales Dokument zu erzeugen. Durch Hinzufügen von CSS ist es möglich das Layout und Design jederzeit anzupassen.

## 5.1 Kommandozeilen Modul

Das Kommandozeilen Modul ist gleichzeitig unser Hauptmodul mit welchem der Anwender über die Kommandozeile kommuniziert. Beim Programmstart wird dieses Modul gestartet.

Das Python Argparse Modul ist hilfreich bei der Erstellung eines Kommandozeilen Programms. Handling von Optionen Argumenten und Unterbefehle werden im Argparse Modul hervorragend umgesetzt. Die Main Funktion von Reportrick übergibt ein Objekt mit allen Attributen an die parsecli Methode. Diese Methode überprüft ob das Objekt eine Unterbefehl enthält oder nicht. Abhängig davon verläuft der weitere Programmablauf unterschiedlich. Dies ist in 5.1.1 und 5.1.2 festgehalten.

### 5.1.1 Kommandozeilen Interaktion mit Menü ohne Unterbefehle

Falls kein Funktionsname im Objekt enthalten ist, dann wird auf der Konsole ein Menü ausgegeben. Der Nutzer wird begrüßt und das aktuelle Datum wird ausgeben. Durch das Nutzten der „datetime“ und „date“ Module können wir sehr einfach das Datum und die aktuelle Kalenderwoche nutzen. Der Anwender wird aufgefordert eine Eingabe in Form einer Zahl zu machen um auf Funktionen wie in der Entwurfshase 4.1.1 beschrieben.

Ein Auszug mit der Logik zeigt wie der Anwender auf die einzelnen Unterfunktionen zugreift:

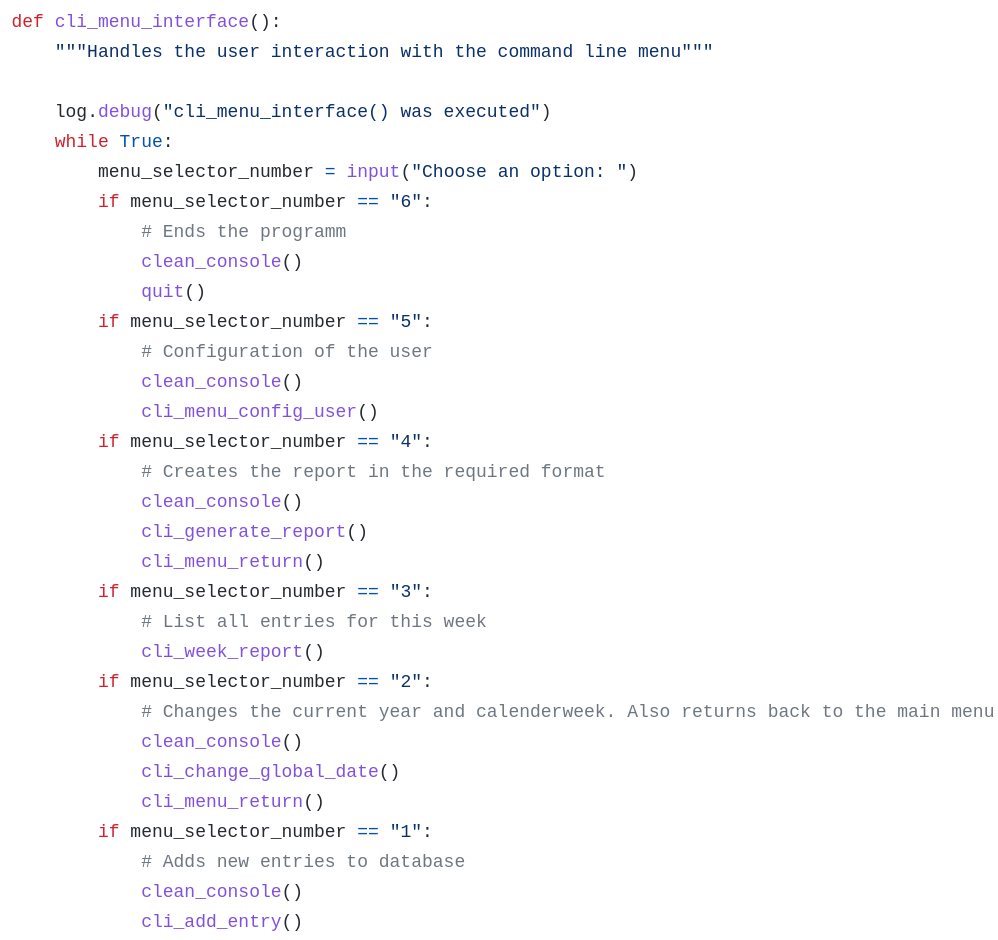


Abbildung 4 Implementierung des Menü Interfaces

### 5.1.2 Kommandozeilen Interaktion ohne Menü mit Unterbefehl

Das Übergebene Objekt aus der Main Funktion enthält ein Attribut „func“

Unterbefehle ruft man immer mit der gleichen Art und Weise auf

Reportrick add „Argument1“ „Argument2“.

Ein Unterbefehl wird immer im gleichen Muster implementiert.

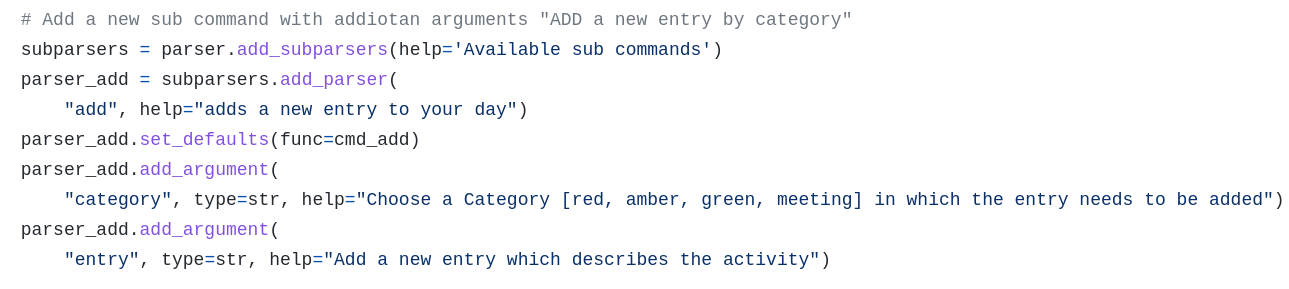


Abbildung 5 Implementierung eines Unterbefehls mit zwei Argumenten

In diesem Fall um einen Befehl direkt über die Konsole hinzufügen, muss man den Befehl in dieser Syntax eingeben.

Beispiel Aufruf

“Reportrick add red Codereview”

Somit kann der Anwender ganz schnell mit dem Programm interagieren.

## 5.2 Datenbank Modul

Die Nutzung der Datenbank wird in einer Klasse “Datenbank“ umhüllt. Bei dem ersten Start von Reportrick wird eine Instanz der Klasse Datenbank erzeugt und der Konstruktor erstellt mit dem Aufruf einer privaten Methode eine Leere Datenbank. SQLite Befehle werden als Strings in Variablen abgespeichert.   
Siehe Codebeispiel

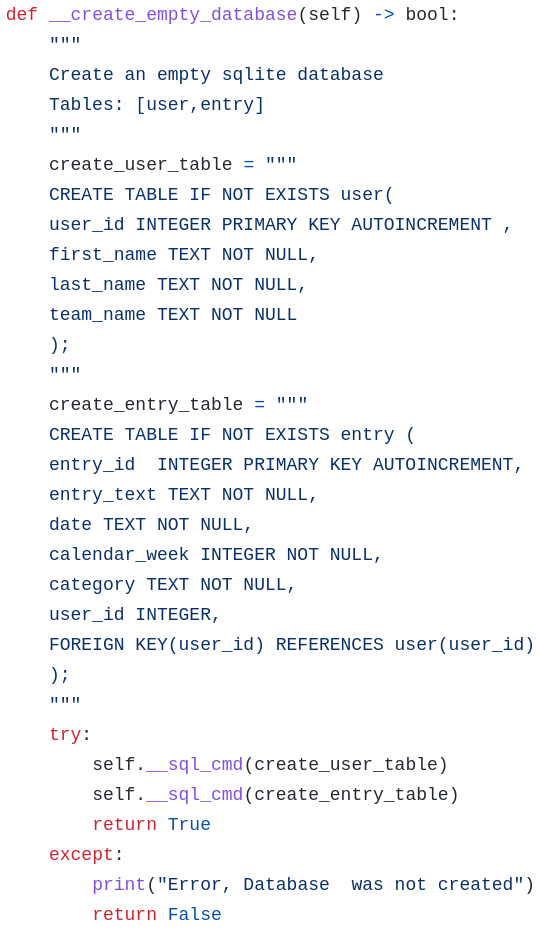


Abbildung 6 Implementierung der privaten Methode für die Tabellenerstellung

## 5.3 Bericht Modul

Das Bericht Modul ist verantwortlich dafür explizit für die Erstellung der Berichte.

Das Kommandozeilen Modul ruft Methoden des Bericht Moduls auf. Jinja2 nutzt ein Dictionary oder besser bekannt eine „Key/Value“ Dateistruktur.  
Beispiel:



Abbildung 7 Implementierung eines Jinja2 Dictionary

Das Modul beinhaltet für jedes Format eine eigene Funktion für die Erstellung. Dabei bezieht sich das Modul immer auf eine Vorlage die in einem „template“ Ordner innerhalb des Projektes abgespeichert worden ist. Unterstütze Formate, für die Erstellung von Berichten sind, HTML, PDF und Text.

## 5.4 Vorlage für das Bericht Modul

Um Berichte zu erstellen wird eine HTML Vorlage für das Bericht Modul erstellt. Eine Simple HTML Struktur mit Platzhaltern wird angelegt. Beim Ausführen greift das JInja2 Modul auf die HTML Vorlage zu und befüllt diese mit Werten aus der Datenbank. Auszug aus der jinja2 HTML Vorlage für die dynamische Befüllung aller Einträge in der Kategorie grün.



Abbildung 8 HTML Vorlage

# 6. Projektkontrolle

## 6.1 Zeitlicher Soll-Ist-Vergleich

Im Soll-Ist-Vergleich wird meine Zeitplanung mit der tatsächlichen

Zeitverglichen. In der nachfolgenden Grafik ist ein Soll-Ist-Vergleich grafisch dargestellt:

Abbildung erstellen

Abbildung 9 Soll-IST-Vergleich

Wie es sich aus der Grafik entnehmen lässt, habe ich in der Entwurfsphase 1,5 Stunden länger gebraucht als es in der Planung angesetzt war. Das liegt daran, dass ich mich erstmal in die Struktur des Farbwiegesystems einarbeiten und ein Konzept für die Übertragung entwerfen musste. Dafür konnte ich in der Durchführung durch ein detailliertes Konzept der Umsetzung 1,5 Stunden wieder einsparen. Dadurch gibt es Abweichungen von der ursprünglichen Zeitplanung, welche aber die gesamte vorgegebene Projektzeit von 70 Stunden nicht beeinflussen.

Die detaillierte Ansicht ist unter Table 8 Tatsächlicher Zeitaufwand zu finden.

## 6.2 Zielerreichungsvergleich

Am Ende der Umsetzung sollen die Anforderungen mit dem Ergebnis verglichen werden. Als Hilfsmittel dazu dient das Pflichtenheft. Hierbei werden alle festgelegten Anforderungen durchlaufen und mit dem tatsächlichen Ergebnis verglichen.

Tabelle 5 Zielerreichungsgvergleich

|  |  |
| --- | --- |
| **Funktionalität** | **Ergebnis** |
| Hinzufügen und Entfernen von Einträgen | V |
| Zugriff auf alte Berichte | V |
| Anzeige aller Einträge für den aktuellen Bericht | V |
| Erstellung von Berichten in HTML/PDF oder Text | V |

## 6.3 Kostenvergleich

In diesem Abschnitt werden die geplanten Kosten den tatsächlichen Kosten gegenübergestellt. Der Kostenvergleich lässt sich aus der folgenden Tabelle entnehmen:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Vorgang | Mitarbeiter | Stundensatz | Zeit | Kosten |
| Entwicklungskosten | 1x Auszubildender | 9.34 € | 70 h | 653.80 € |
| Codereview | 1x Mitarbeiter | 30.00 € | 2 h | 60.00 € |
| **Gesamtkosten** | | | | 713.80 € |

Tabelle 6 Kostenvergleich

Trotz kleiner Abweichungen in einzelnen Projektphasen wurden die geplanten Kosten eingehalten.

## 6.4 Testdurchführung

In der folgenden Phase wird für korrekte Funktionalität und Qualität des Quellcodes gesorgt. Dafür wird die Testdurchführung in zwei Schritte aufgeteilt, welche unten aufgeführt sind.

### 6.4.1 Code Review als White-Box-Test

Der White-Box-Test beinhaltete das Testen des Quellcodes. Dazu wurde zunächst ein Codereview von einem erfahrenen Entwickler durchgeführt, um die Performance des Codes sicherzustellen. Die Codereview ergab das die Namen nicht sprechend genug sind. Außerdem wurden Hinweise mitgegeben welche Python Bibliotheken nicht bedingt nötig sind.  
Da wir versuchen die Anzahl der Abhängigkeiten zu reduzieren wurde der Code nochmal umgeschrieben. Die Namen wurden sprechender beschrieben und unnötige Module wurden entfernt.

Die wiederholte Codereview ergab kein weiteren Anpassungsbedarf.

### 6.4.2 Systemtest als Black-Box-Test

Die Anpassung wird anhand der Spezifikation getestet. Hier wird geprüft, ob die Anforderungen erfüllt werden. Dabei werden die in der Planungsphase festgelegten Testszenarien durchlaufen. Die Testergebnisse werden zusätzlich dokumentiert.

# 7. Projektabschluss

## 7.1 Übergabe an Kunden/Team

Nach einer kurzen Einführung wurde das neue Tool vom Mitarbeiter getestet. Nachdem keine Fehler gefunden worden ist wurde das Projekt firm intern von den Mitarbeitern in ihre tägliche Arbeitsroutine aufgenommen.

## 7.2 Fazit

Mein Fazit habe ich in drei wichtigste Punkte aufgeteilt:

### 7.2.1 Projekt umgesetzt

Durch dieses Projekt wurde die Problemstellung gelöst und somit ist jetzt der Wöchentliche Bericht der Mitarbeiter einheitlich und vollständig.

### 7.2.2 Zufriedene Kollegen

Die Kollegen sind sehr zufrieden, da sowohl alle Anforderungen als auch das Projektziel erreicht wurde.

### 7.2.3 Gelerntes Wissen

Durch das Projekt konnte ich wertvolle Erfahrungen im Bereich Projektmanagement gewinnen. Bei der Analyse und Konzeption hatte ich einen tieferen Einblick in verschiedene Teams innerhalb der Firma. Zusätzlich konnte ich bei der Umsetzung mein fachliches Wissen als Programmierer erweitern.

## 7.3 Ausblick

Durch die Implementierung dieses Projektes entsteht ein solider Grundbaustein der schon bereit zur Nutzung ist. Die gewählte Architektur ermöglicht Erweiterungen nach Bedarf. Wie in der Projekt Abgrenzung unter Punkt 2.5 ist die automatisierte E-Mail-Versendung weitere Automatisierung möglich. Nach der Erstellung des Berichtes kann dieser automatisiert an den Teamleiter versendet werden. Weitere möglichen Features wie Importieren eigener CSS-Dateien für die Umgestaltung der Layout Farben und Logos ist möglich. Eine weitere Möglichkeit dieses Projekt zu erweitern ist die Implementierung ein Betriebssystem unabhängigen Grafischen User Interfaces mit Hilfe von der Godot Spiele Engine. Somit kann eine Kommandozeilen unabhängige User Interaktion auf allen übliche Betriebssystem wie Windows, MacOS, Linux und Android genutzt werden.

# 8. Glossar

### **HTML**

Hypertext Markup Language, Sprache zur Auszeichnung von Inhalten für Webbrowser. Der Browser interpretiert die Sprache, um z.B. Inhalte zu positionieren

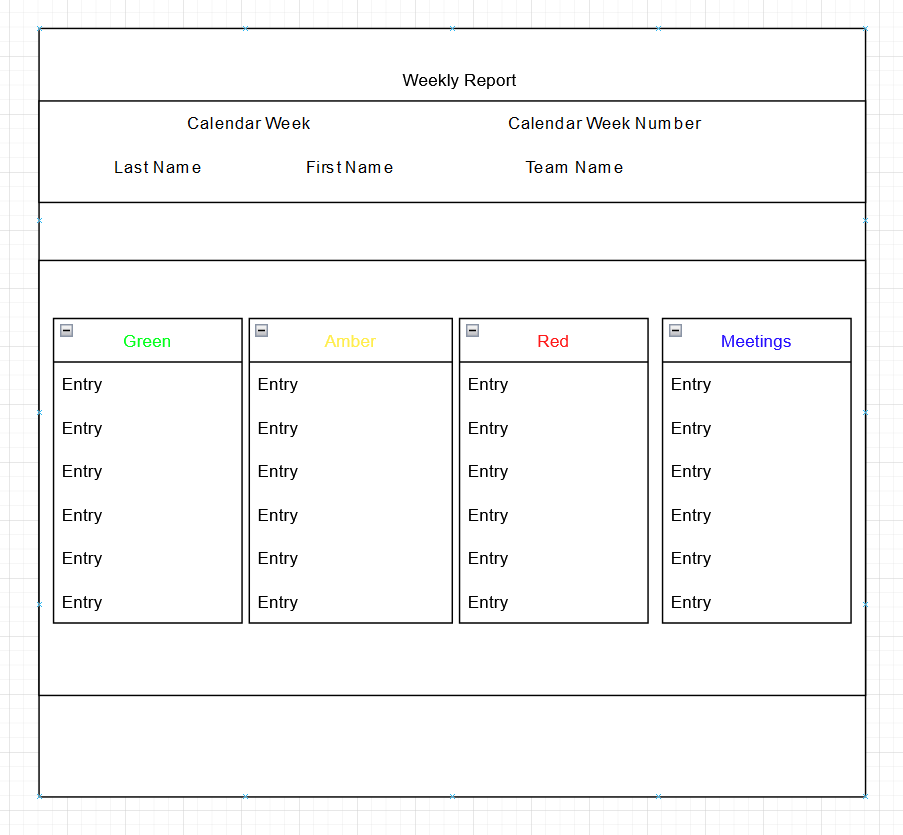
### Codereview

Die Sicherstellung der Softwarequalität durch Untersuchung von Quellcode

### CSS

# Anhang

## Anhang A: Mockup Erstes Berichts Layout

Abbildung 4 Vorläufiges Berichtslayout

## Anhang B: Mockup Endgültiges Berichts Layout

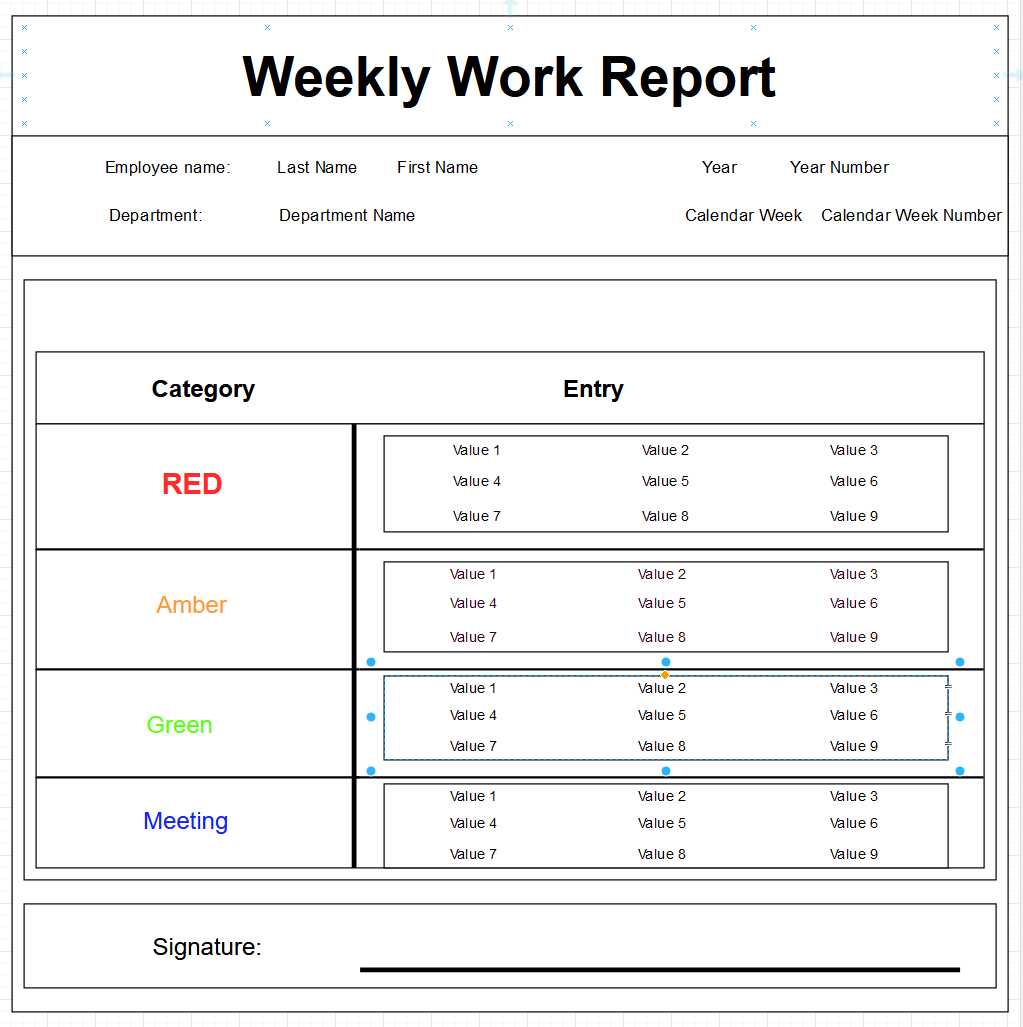


Abbildung 4 Endgültiges Berichtslayout

## Anhang C: Detaillierte Zeitplanung

|  |  |  |
| --- | --- | --- |
| Projektphase | | Zeit in Stunden |
| 1 | Erstellung der Ist-Analyse | 2 |
| 2 | Erstellung des Soll-Konzepts | 2 |
| 3 | Zeit und Aufwandsplanung | 4 |
| 4 | Entwurf der Kommunikation zwischen Benutzer und Tool | 5 |
| 5 | Entwurf der Datenbank | 1 |
| 6 | Design des Berichtes | 2 |
| 7 | Aufteilung der Logik in mehrere Module | 1 |
| 8 | Planung der Testszenarien | 1 |
| 9 | Pflichtenheft | 2 |
| 10 | Erstellung der Datenbanklasse | 5 |
| 11 | Erstellung des Kommandozeilenprogramms | 20 |
| 12 | Entwicklung der Kommandozeilenstruktur | 5 |
| 13 | Entwicklung der Datenbankklasse | 5 |
| 14 | Entwicklung der HTML Struktur | 3 |
| 15 | Entwicklung der CSS Struktur | 3 |
| 16 | Implementierung des Jinja Scripts | 3.5 |
| 17 | Zeitlicher Soll-Ist-Vergleich | 1.5 |
| 18 | Zielerreichungsvergleich | 1 |
| 19 | Kostenvergleich | 1.5 |
| 20 | Testdurchführung | 5 |
| 21 | Übergabe an Team | 1 |
| 22 | Erstellung der Projektdokumentation | 10 |
|  |  |  |
|  |  |  |

Tabelle 7 Detaillierte Zeitplanung

## Anhang D: Datenbank Modell

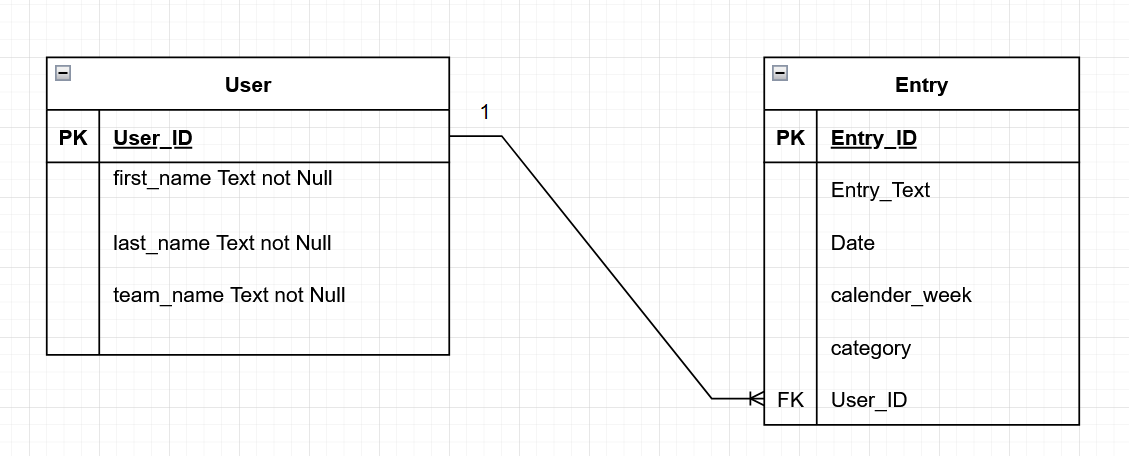


Abbildung 10 Datenbank Modell

## Anhang E: Klassendiagramm

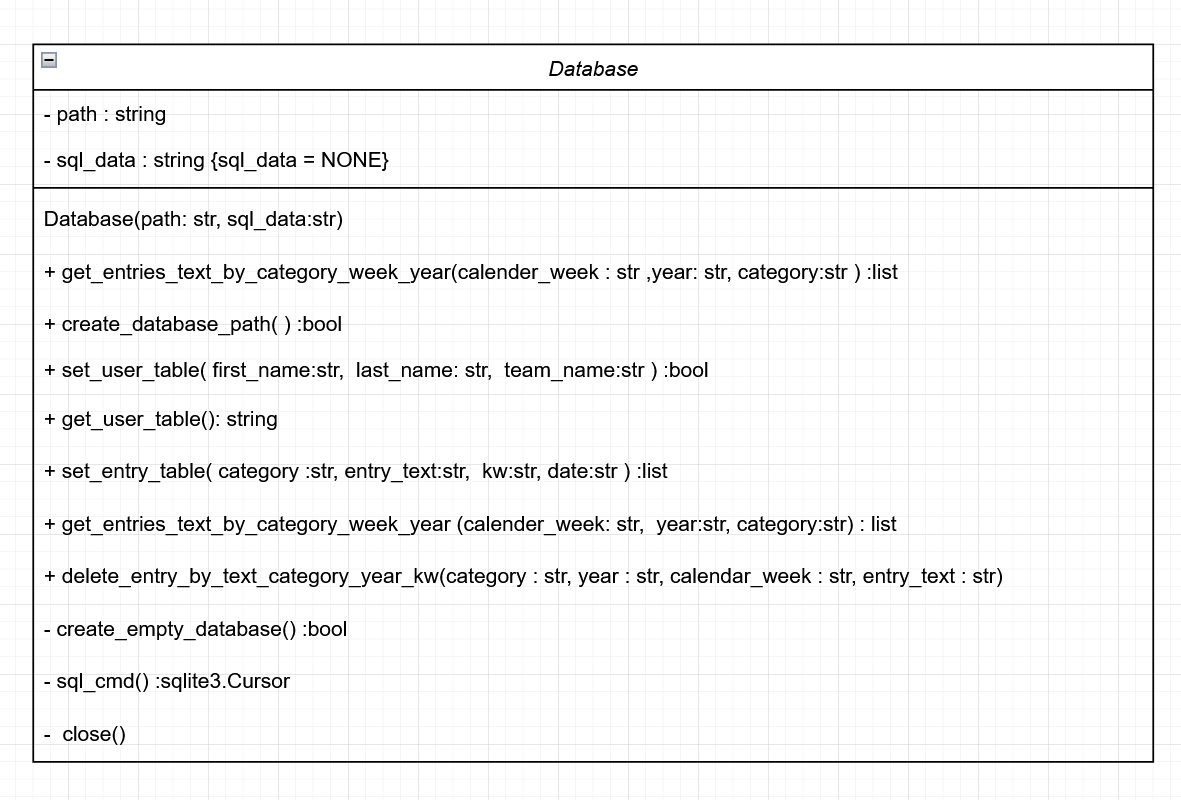


Abbildung 11 Database Klassendiagramm

## Anhang G: Zusätzliche Bearbeitungszeit

Detaillierter Zeitvergleich ist im Anhang zu finden

|  |  |  |  |
| --- | --- | --- | --- |
| Tätigkeit | | Zeit in Stunden | Abweichung |
|  | Erstellung der Ist-Analyse | 2 | 0 |
|  | Erstellung des Soll-Konzepts | 2 | +1 |
|  | Zeit- und Aufwandsplanung | 1 | +2 |
|  | Planung der Testszenarien | 1 | -2 |
|  | Struktur und Ablauf des Kommandozeilentools | 4 | 0 |
|  | Entwurf des Layouts für die HTML Vorlage | 2 | 0 |
|  | Design der Datenbank | 2 | 0 |
|  | Pflichtenheft | 2 | 0 |
|  | Entwicklung der Kommandozeilenstruktur | 5 | 0 |
|  | Entwicklung der Datenbankklasse | 5 | 0 |
|  | Entwicklung der HTML Struktur | 3 |  |
|  | Entwicklung der CSS Struktur | 3 | 0 |
|  | Implementierung des Jinja Scripts | 3.5 | 0 |
|  | Zeitlicher Soll-Ist-Vergleich | 1.5 | 0 |
|  | Zielerreichungsvergleich | 1 | 0 |
|  | Kostenvergleich | 1.5 | 0 |
|  | Testdurchführung | 5 | 0 |
|  | Übergabe an Team | 1 |  |
|  | Erstellung der Projektdokumentation | 10 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  | 14.5h fehlt |  |
|  |  | 55.5 |  |
|  | Gesamt | 70 | 0 |
|  |  |  |  |

Tabelle 7 Tatsächlicher Zeitaufwand