

IPA Bericht

Webtool zur Arbeitszeiterfassung

**Zürich, Donnerstag, 27. Mai 2021**

**Lars Gächter**

Inhaltsverzeichnis

[Teil 1 Umfeld und Ablauf 4](#_Toc72925174)

[1.1. Ausgangslage 4](#_Toc72925175)

[1.2. Detaillierte Aufgabenstellung 4](#_Toc72925176)

[1.3. Projektaufbauorganisation 7](#_Toc72925177)

[1.4. Projektmanagement Methode 7](#_Toc72925178)

[1.5. Mittel und Methoden 7](#_Toc72925179)

[1.6. Vorkenntnisse 7](#_Toc72925180)

[1.7. Vorarbeiten 7](#_Toc72925181)

[1.8. Neue Lerninhalte 8](#_Toc72925182)

[1.9. Arbeiten in den letzten 6 Monaten 8](#_Toc72925183)

[1.10. Dokumentablage 8](#_Toc72925184)

[1.11. Zeitplan 8](#_Toc72925185)

[1.12. Arbeitsprotokoll 10](#_Toc72925186)

[1.12.1. Tagesprotokoll vom 12.5.2021 10](#_Toc72925187)

[1.12.2. Tagesprotokoll vom 14.5.2021 12](#_Toc72925188)

[1.12.3. Tagesprotokoll vom 17.5.2021 14](#_Toc72925189)

[1.12.4. Tagesprotokoll vom 18.5.2021 15](#_Toc72925190)

[1.12.5. Tagesprotokoll vom 19.5.2021 16](#_Toc72925191)

[1.12.6. Tagesprotokoll vom 20.5.2021 17](#_Toc72925192)

[1.12.7. Tagesprotokoll vom 21.5.2021 18](#_Toc72925193)

[1.12.8. Tagesprotokoll vom 25.5.2021 19](#_Toc72925194)

[1.12.9. Tagesprotokoll vom 26.5.2021 20](#_Toc72925195)

[1.12.10. Tagesprotokoll vom 27.5.2021 21](#_Toc72925196)

[1.12.11. Tagesprotokoll vom 28.5.2021 22](#_Toc72925197)

[Teil 2 Projekt-Dokumentation 23](#_Toc72925198)

[2.1. Kurzfassung 23](#_Toc72925199)

[2.2. Informieren 24](#_Toc72925200)

[2.3. Planen 29](#_Toc72925201)

[2.3.1. Testkonzept 33](#_Toc72925202)

[2.4. Entscheiden 34](#_Toc72925203)

[2.5. Realisieren 34](#_Toc72925204)

[2.6. Kontrollieren 37](#_Toc72925205)

[2.6.1. Testprotokoll 37](#_Toc72925206)

[2.7. Auswerten 37](#_Toc72925207)

[2.7.1. Reflexion 37](#_Toc72925208)

[2.7.2. Schlusswort 37](#_Toc72925209)

[2.8. Glossar 38](#_Toc72925210)

[2.9. Quellenverzeichnis 38](#_Toc72925211)

[2.10. Literaturverzeichnis 38](#_Toc72925212)

[Teil 3 Anhang 39](#_Toc72925213)

[3.1. Abbildungsverzeichnis 39](#_Toc72925214)

[3.2. Tabellenverzeichnis 39](#_Toc72925215)

[3.3. Listing des Programmcodes 39](#_Toc72925216)

# Umfeld und Ablauf

## Ausgangslage

Die Rafisa Informatik bietet seit diesem Jahr ein betreutes Wohnen an. Die BetreuerInnen für das Wohnen absolvieren mehrmals über den Tag verteilt ihre Arbeitseinsätze. Diesen Fall deckt die Applikation die die Rafisa für die Zeiterfassung nutzt nicht ab. Da die Applikation "monolithisch" strukturiert ist und der Ursprüngliche Entwickler mitlerweile pensioniert ist die Applikation nur schwer erweiterbar. Deshalb wünscht sich die Geschäftsleitung eine Applikation mit der die WohnbetreuerInnen ihre Arbeitszeiten dokumentieren können.

## Detaillierte Aufgabenstellung

Code-Konventionen:

- Namensgebung analog zu Java. Klassen sind 'upper camel case', Methoden und Variabeln sind 'lower camel case' und Konstanten 'upper case underscore seperated'.

-Zugriff auf Variabeln nur über Methoden

- Authentifizierung: Benutzer müssen sich über das LDAP der Rafisa authentifizieren können.

- Eingeloggte Benutzer können die Zeiterfassung starten falls sie nicht aktiv ist und beenden falls sie aktiv ist

- Pro Tag können mehrere Arbeitseinträge pro Benutzer erfasst werden.

- Am Tagesende müssen Aufgrund der Einträge das jeweilige Tages-, Wochen- und Monatssaldo berechnet werden.

Dokumentation:

Zur Planung soll ein Use-Case Diagramm Verwendet werden.

Damit das Programm später erweiterbar ist soll auch ein ERM und ein Aktivitätsdiagramm erstellt werden.

Eingabemasken:

Die Eingabemasken sind analog zu denen in der regulären Zeiterfassung zu halten.

Loginmaske:

-Ein Textfeld für den Benutzernamen.

-Ein Passwortfeld für das Passwort

-Einen Button zum Absenden des Formulars

Zeiterfassungsmaske:

-Einen Button mit der Beschriftung "Zeiterfassung starten" falls die Zeiterfassung nicht gestartet ist.

-Einen Buttom mit der Beschriftung "Zeiterfassung stoppen" falls die Zeiterfassung gestartet ist.

Übersichtsmaske:

-Eine Tabelle mit den Daten des Ausgewählten Monats. Die Einträge sollen in chronologischer Reihenfolge dargestellt werden. Titel der Tabelle soll der ausgewählte Monat und die Jahreszahl sein und als Standardwert soll der derzeitige Monat gesetzt sein.

Links soll zum vorherigen Monat navigiert werden können, rechts zum folgenden. Wenn der aktuelle Monat ausgewählt ist soll das nach vorne Navigieren deaktiviert und ausgegraut sein.

-Die Einträge sollen für jeden Arbeitseinsatz folgende Attribute auflisten: Datum des Eintrages, Startzeit des Eintrages, Endzeit des Eintrages, Soll- und Istarbeitszeit des dazugehörigen Tages sowie die Differenz dieser zwei Werte und die Zeitsaldi für die jeweilige Woche, Monat und Total. Daneben noch jeweils ein Feld für allfällige Textkommentare. Neben jedem Eintrag soll ein klickbares Symbol sein das beim anklicken die Maske zur Korrektur des Eintrages aufruft.

-Unter den Einträgen der Arbeitszeit soll eine Zeile das Totalwerte dieses Monats angezeigt werden.

Korrekturmaske:

Die Korrekturmaske soll erneut alle Attribute eines Eintrages auflisten. Neben jedem Eintrag soll ein Feld zur Korrektur des Wertes vorhanden sein. Falls in einem oder mehreren Feldern Korrekturen vorgenommen werden soll der Kommentar ein Pflichtfeld sein. In einer dritten Spalte soll jeweils eine kurze Erklärung des jeweiligen Attributes stehen. Die Beschreibung kann 1:1 aus der vorhandenen Zeiterfassung kopiert werden. Zusätzlich soll angezeigt werden wann und von welchem Benutzer zuletzt eine Korrektur für diesen Eintrag vorgenommen worden ist.

Im Falle einer Korrektur ist ein Textkommentar zwingend.

Tägliche verarbeitung:

Am Ende jeden Tages soll das Saldo ausgerechnet werden. Arbeitsstunden an Sonn- und Feiertagen werden 1.5-fach gezählt.

Fehlerbehandlung

- Ist keinen Kontakt zum LDAP möglich, soll eine Fehlermeldung erscheinen die den Benutzer auf diesen Umstand aufmerksam macht.

- Bei Erhalt von Fehlerhaften Daten, wird eine Fehlermeldung angezeigt die auf die Inkompatibilität der

Daten aufmerksam macht.

- Bei internen PHP-Fehlermeldungen sollen diese nicht angezeigt werden. Werden solche

Fehlermeldungen generiert, muss der Programmablauf abgebrochen werden und zur „Einloggen“ Maske navigiert

werden.

Tests

Die Aufgabe umfasst die Installation eines Webauftrittes der funktionell ist. Das Ganze soll ausgiebig getestet werden.

- Dazu ist ein Testkonzept zu erstellen.

- Die Tests sind gemäss Konzept durchführen.

Mindestens folgende Testszenarien müssen durchgeführt werden:

- Korrekte Funktion der Masken.

- Fehler abfangen.

- Fehlermeldungen nach Vorgabe anzeigen.

## Projektaufbauorganisation

Graphical user interface, application

Description automatically generated

Abbildung : Baumdiagramm

## Projektmanagement Methode

## Mittel und Methoden

-Windows oder Linux-PC mit XAMP oder LAMP

PHPStorm

PHP/Symfony/Twig/Composer

HTML/CSS/Dart/Sass

Javascript/NodeJS

Git

MongoDB

LDAP

## Vorkenntnisse

PHP

HTML/CSS/Dart/Sass

Javascript/NodeJS

Git

## Vorarbeiten

Kennenlernen der LDAP authentifizierung.

Einrichten einer virtualisierten Arbeitsumgebung.

Installation von MongoDB

Evaluation der Anforderungen an die Zeiterfassung

## Neue Lerninhalte

LDAP api

MongoDB

## Arbeiten in den letzten 6 Monaten

-Webapplikation zum Ausführen und korrigieren von Mathematiktests

-Grafische Darstellung von "Nested Sets" in Kuchendiagrammen

Dafür wurden folgende Technologien verwendet:

PHPStorm

Git

VScode

CSS

javascript

## Dokumentablage

**Projekt Initialisierung**

* Ich mache jeden Tag einen Git Commit oder mehrere Git Commits.
* GitLab Repository erstellen und Vorgesetzten als Maintainer aufgenommen.
* Gitignore erstellt, um generierte Dateien von Composer, npm, PhpStorm und Autoloader im Git zu ignorieren.
* Ordnerverzeichnis “doc” für die Dokumenation angelegt und darin die Dokumentation als Word und das Gantt als Excel erstellt.
* Zeitplan anfangen zu erstellen.
* Dokumentation angefangen zu erstellen.

## Zeitplan



Abbildung : Projektzeitplan

## Arbeitsprotokoll

### Tagesprotokoll vom 12.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Das Analysieren der Aufgabenstellung und Zusatzinformationen einholen oder Ergänzugen zur Aufgabenstellung.  Wenn möglich alle Tätigkeiten festelegen.  Zeitplan zu den erfassten Tätigkeiten erstellen.  Besuchstag 1. Expertengespräch | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 09:00- 10:00 | Projekt Initialisierung | Hat alles funktioniert. | Mein Fachvorgesetzter Stefan Kuhn hat mir geholfen einen Namen für das Gitlab Repository festzulegen und wie Ihn als Maintainer vom Repository einzuladen. |
| 10:00-12:00 | Analysieren der Aufgabenstellung  Einlesen verstehen und in anderen Worten zusammenfassen. | Hat alles funktioniert. | Keine. |
| 13:00-13:30 | Analysieren der Aufgabenstellung  Einlesen verstehen und in anderen Worten zusammenfassen. | Hat alles funktioniert. | Keine. |
| 13:30-14:30 | Ergänzugen zur Aufgabenstellung.  Nachfolgende Informationen in in der Doukmentaion erwähnt,   * Abmeldung / Angemeldet * Masken-Navigation * Übersicht-Navigation * Mobiltauglichkeit * Übersicht Monatstotal * Benutzer Pflichtzeit * Korrekturmaske im Detail | Hat alles funktioniert. | Keine. |
| 14:30-16:00 | Zeitplan erstellen |  |  |
| 16:00-17:00 | Experten Gespräch |  |  |
| 17:00-18:00 | Zeitplan erstellen und Zusatzinformationen einholen |  | Mit Stefan Kuhn besprochen. |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| 9 Stunden | | 9 Stunden | |
|  | | | |
| **Projektfortschritt** | | | |
| I von IPERKA fertig  Analysieren der Aufgabenstellung, abgeschlossen  Zusatzinformationen einholen oder Ergänzugen zur Aufgabenstellung, abgeschlossen.  Zeitplan erstellen, angefangen. | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
| GitLab und Stefan Kuhn für die Erstellung vom Git Repository.  IPA Aufgabenstellung von PkOrg.  Screenshots von Masken der derzeitigen Zeiterfassung von Stefan Kuhn oder durch intern.stiftung. | | | |
|  | | | |
| **Tagesreflexion** | | | |
| Ich habe heute gelernt, dass ich bei zwei Einladungen via Teams auf die erste Einladung am ersten Termintag im Teams abrufen sollte und nicht auf irgend einen Link wie den zweiten Link für den zweiten Termintag.  Ich habe heute gelernt wie man in GitLab zu einem neu erstellten Git Repository einen Maintainer hinzufügt.  Mir ist es heute gut bei der Informierung-Phase gegangen und war im Zeitplan. | | | |

Tabelle : Tagesprotokoll vom 12.5.2021

### Tagesprotokoll vom 14.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Die Projektaufbauorganisation in Form von einem Baumdiagramm erstellt. | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 08:30-  09:00 | Zeitplan überarbeitet | Ich konnte am meisten selbständig den Zeitplan überarbeiten | Hatte Hilfe durch Stefan Kuhn |
| 09:00-  12:00 | Use-Cases, ERM Diagramm erstellt und mit dem UML Klassen Diagramm angefangen | Hat alles funktioniert. | Keine. |
| 13:15-14:45 | UML Klassen Diagramm und Aktivitätsdiagramm erstellt, Kurzfassung schreiben. | Hat alles funktioniert. | Kurze Repetition mit Stefan Kuhn, was eine Kurzfassung ist. |
| 14:45-16:00 | Manuelles Testkonzept und PHPUnit Tests |  |  |
| 16:00-17:30 | Besprechung meiner IPA mit Ausbildner und nachträglich Notizen machen. | Weiss nun mehr was ich in meiner IPA zu beachten habe. | Feedback von Egil Rüefli, auf meine heutigen erarbeiteten Elemente |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| 8 Stunden | | 17 Stunden | |
|  | | | |
| **Projektfortschritt** | | | |
| P von IPERKA fertig | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
| Durch die Informierung vom Vortag und Hilfe von Stefan Kuhn und Egil Rüefli. | | | |
| nach feedback von Egil und Stefan  Kunde Monika Schmid im Baumdiagramm aufführen  Zeitplan noch in textfrom besser verfeinern  ERM überarbeiten, Aktivitätsdiagramm verfeinern  UML Klassendiagramm überarbeiten  ERM database eigenschaften von Table und in MongoDB fields immer lower-camel-case wie die variablen in PHP  Glossar alphabetisch, alle Begriffe oder in anderen Worten Schlüsselwörter als Titel und darunter ein kurzer Text erfassen welcher wie zb. google erklärt Abkürzung auschreiben  Hinhaltverzeinchnis quellen [1]... online quellen angeben  Testkonzept genauer beschreiben was bei den PHPUnit testen möchte und was bei den manuellen tests im frontend gemacht/getestet werden sollte  Arbeitsjournal ist wichtig erfolge und misserfolge genau zu benennen und das die zeiten im zeitplan übreinstimmen  Wenn ein link nicht mehr geht, waybackmachine benutzen oder darauf hinverweisen | | | |
| **Tagesreflexion** | | | |
| Ich habe heute viel von Stefan Kuhn über die Kurzfassung gelernt und habe ein Feedback von Egil Rüefli auf den Stand meiner Dokumentation und Planung bekommen.  Mir ist es heute gut bei der Informierung-Phase gegangen und war etwas ausserhalb vom Zeitplan. | | | |

Tabelle : Tagesprotokoll vom 9.5.2017

### Tagesprotokoll vom 17.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Vorentscheidungen treffen, ERM lower camel case statt snake case, Projektaufbauorganisation Kundin hinzufügen, mit Code realisierung von Masken und Architektur beginnen. Schlüsselwörter für das Glossar notieren. | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 08:10-12:00 | ERM und Projektaufbauorganisation angepasst, ein paar entscheindungen getroffen | Die Entscheidungen welche mir beim planen eingefallen sind, konnte ich gut abwägen. | Es bleiben Entscheidungen offen welche ert bei der Realsierung zum fallen kommen. |
| 13:00-17:30 | Composer und npm mit den für das Projekt nötigen Packages eingerichtet, PHP Klassen schreiben, MongoDB einrichten, LDAP einrichten, Views und Masken erstellen | Hat alles reibungslos funktioniert. | Keine. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| Stunden 9 Minuten 20 | | Stunden 26 Minuten 20 | |
|  | | | |
| **Projektfortschritt** | | | |
| E von IPERKA fertig und mit R angefangen | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
| Aus eigener Dokumentation. | | | |
|  | | | |
| **Tagesreflexion** | | | |
| Habe heute mir überlegt welche Entscheidungen für das Projekt wichtig sein können und diese notiert. | | | |

Tabelle : Tagesprotokoll vom 17.5.2021

### Tagesprotokoll vom 18.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Mit der Persistenz-Logik anfangen wie die Daten ans Frontend und Backend bei den Benutzern weitergegeben werden. | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 09:15-11:45 | snake case zu lower camel case von Twig Variablen und MongoDB Keys anpassen vom Benutzer. | Die Applikation funktionierte danach weiterhin gleich. | Nimmt etwas Zeit in Anspruch. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| Stunden 2 Minuten 30 | | Stunden 28 Minuten 50 | |
|  | | | |
| **Projektfortschritt** | | | |
| R von IPERKA fortgeführt | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
| Aus eigener Dokumentation. | | | |
|  | | | |
| **Tagesreflexion** | | | |
| Habe heute Morgen alle Keys wie Variablen in lower camel case umgeschrieben. | | | |

Tabelle : Tagesprotokoll vom 18.5.2021

### Tagesprotokoll vom 19.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Client Session konfigurierbar machen und auf 8 Stunden stellen. Die Persistenz-Logik im Aktivitätsdiagram für die Korrektur-Maske verfeinern. | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 09:00-11:00 | Session in Umgebungsvariablen aufgenommen. Benutzer ist 8 Stunden nach seiner letzten Anmeldung nicht mehr angemeldet. | Hat alles bestens funktioniert | Keine |
| 11:00-12.00 | Persistenz-Logik im Aktivitätsdiagram | Speicherung ohne Berechnung auseinandergehalten | Mit was beginnen und Aufbau vom Diagramm |
| 13:00-18:00 | Persistenz-Logik im Aktivitätsdiagram | Verarbeitung besser verfeinert | Verfeinern einzelner Input Verarbeitungen |
|  |  |  |  |
|  |  |  |  |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| Stunden 8 Minuten 0 | | Stunden 36 Minuten 50 | |
|  | | | |
| **Projektfortschritt** | | | |
| R von IPERKA fortgeführt  Session Handling abgeschlossen und im groben die Speicherung und Berechnung in der Korrektur-Maske im Aktivitätsdiagram festgehalten. | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
| Keine | | | |
|  | | | |
| **Tagesreflexion** | | | |
| Habe heute am Aktivitätsdiagram mehr Zustände für die Berechnung-Kaskadierung herausgefunden. | | | |

Tabelle : Tagesprotokoll vom 19.5.2021

### Tagesprotokoll vom 20.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Die Persistenz-Logik im Aktivitätsdiagram für die Korrektur-Maske fertig machen. Beginen mit implementieren von der Persistenz-Logik für die Korrektur-Maske oder Testdaten in der MongoDB erzeugen. | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 09:00-09:45 | Dokumentieren |  |  |
| 09:45-12:00 | Persistenz-Logik im Aktivitätsdiagram |  |  |
| 13:00-13:45 | Persistenz-Logik im Aktivitätsdiagram |  |  |
| 13:45-18:00 | Testdaten erfassen |  |  |
|  |  |  |  |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| Stunden 8 Minuten 0 | | Stunden 44 Minuten 50 | |
|  | | | |
| **Projektfortschritt** | | | |
| R von IPERKA fortgeführt | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
| Keine | | | |
|  | | | |
| **Tagesreflexion** | | | |
| Ich habe heute einen Weg gefunden schneller korrekte Testdaten zu generieren mit wenigen manuellen Anpassungen in der Datenbank. | | | |

Tabelle : Tagesprotokoll vom 20.5.2021

### Tagesprotokoll vom 21.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Monatliche Pflichtzeit von jedem Arbeitstag berechnen. Weitere Testdaten exportieren uind importieren. Die beiden Methoden isSunday und isSaturdy Logikfehler beheben. Da DateTime nicht grosser als 24 Stunden unterstzt das es wie ein tages Objekt ausgelegt, eine Methode welche zum Beispiel aus dem String “64:23” ein associative Array [“hour” => 64, “minute” => 23] zurückgibt. Da meherer Tage rauskommen werden diese anzahl Tage mit 24 Stunden multiplizieren, das Total in MongoDB statt DateTime speichern. Performance von der Übersicht testen, da diese am meisten Daten aus MongoDB bezieht. | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 08:05-12:00 |  |  |  |
| 13:00 - -17:45 |  |  |  |
| 17:45-18:00 | Performance von der Übersicht testen und optimieren. | Bei 1 Monat Eintrag von 5 Sekunden Ladezeit auf 100 Millisekunden Ladezeit. | Hilfe bei Performance Optimierung in PHP von Marco Selenati |
|  |  |  |  |
|  |  |  |  |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| Stunden 9 Minuten 0 | | Stunden 53 Minuten 50 | |
|  | | | |
| **Projektfortschritt** | | | |
| R von IPERKA fortgeführt | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
| Keine | | | |
|  | | | |
| **Tagesreflexion** | | | |
| Habe heute gelernt wie man performanteren Code schreiben kann und wie man die Stelle findet im Code wo ein Performance Problem vorliegt. | | | |

Tabelle : Tagesprotokoll vom 21.5.2021

### Tagesprotokoll vom 25.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Cron Job realisieren, dabei müssen die Zeiten von gestern mit den heutigen zusammengerechnet werden, wenn der heutige Tag nicht der erste Tag vom Monat, Woche oder im Gesamten ist. | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 08:05-12:00 | Cron Job realisieren | Unterscheidung von aktiven Benutzern und inaktiven Benutzern, welche die Zeiterfassung gestartet oder nicht gestartet haben. | Die Realisierung |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| Stunden 3 Minuten 55 | | Stunden 57 Minuten 45 | |
|  | | | |
| **Projektfortschritt** | | | |
| R von IPERKA fortgeführt | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
| Keine | | | |
|  | | | |
| **Tagesreflexion** | | | |
| Ich habe heute die Unterscheidung von Benutzern Zeiterfassende, nicht Zeiterfassende und allen Benutzer für den Cron Job implementiert. | | | |

Tabelle : Tagesprotokoll vom 25.5.2021

### Tagesprotokoll vom 26.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Beim Cron Job die Berechnung von gesamter und wöchentlicher Abweichung. Checkliste durchgehen. Durchführung der Abnahme. Diagramme, unter anderem Systeme verfeinern und mit den Unit Tests beginnen. | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 08:00-10:30 | Die Berechnung vom Cron Job weitergeführt. Nach DateTime yesterday und substr(string $string , int $offset , int|null $length = null): string Funktion recherchiert. | Hat alles mit Recherche zusammen funktioniert. | Musste noch berücksichtigen das sich zwei negative Zeiten zusammenaddieren und die Vorzeichen Operatoren von der Zeit getrennt und bei der Speicherung wieder zusammengefügt werden. |
| 10:30-11:30 | Checkliste durchgehen |  |  |
| 11:30-12:00 | UAT Tests vorbereiten |  |  |
| 13:00-17:00 | Cron Job realisieren und Abnahme | Das Berechnen vom heutigen Tag, der monatlichen Pflichtzeit, die monatlichen Präsenzzeit und Abweichung der monatlichen Zeit.  Habe ein Testkonzept 1 für die erste Abnahme geschrieben. | Die Realisierung, bei der Abnahme war ich nicht beim ersten Anlauf gut vorbereitet |
|  |  |  |  |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| Stunden 8 Minuten 0 | | Stunden 65 Minuten 45 | |
|  | | | |
| **Projektfortschritt** | | | |
| K von IPERKA fortgeführt | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
|  | | | |
|  | | | |
| **Tagesreflexion** | | | |
| Habe heute gelernt wie man ein System Diagramm erzeugt. | | | |

Tabelle : Tagesprotokoll vom 26.5.2021

### Tagesprotokoll vom 27.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Unit Tests durchführen, Tagesprotokoll nachführen. Dokumentieren. Besuchstag 2. Expertengespräch | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
| 08:00 – 12:00 | Korrekturen von der Abnahme |  |  |
| 13:00 – 13:30 | Weiterführung der Abnahme mit Testkonzept 2 |  |  |
| 13:30 – 14:30 | PHPUnit erstellen, durchführen und auswerten |  |  |
| 14:30 - | Dokumentieren |  |  |
|  |  |  |  |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| Stunden | | Stunden | |
|  | | | |
| **Projektfortschritt** | | | |
| K und A von IPERKA fortgeführt | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
|  | | | |
|  | | | |
| **Tagesreflexion** | | | |
|  | | | |

Tabelle : Tagesprotokoll vom 27.5.2021

### Tagesprotokoll vom 28.5.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Tagesziele** | | | |
| Bericht fertig dokumentieren. Glossar ergänzen. Bericht grammatikalische Korrektur. Alle Verzeichnisse aktualisieren. Abbildungsverzeichnis nachführen. Alle Tabellen und Diagrammen in Bericht einfügen. | | | |
|  | | | |
| **Zeit** | **Erledigte Arbeiten** | **Erfolge** | **Schwierigkeiten** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  | | | |
| **Arbeitszeit Tagestotal** | | **Arbeitszeit Total** | |
| Stunden | | Stunden | |
|  | | | |
| **Projektfortschritt** | | | |
| K und A von IPERKA fortgeführt | | | |
|  | | | |
| **Informationsbeschaffung** | | | |
|  | | | |
|  | | | |
| **Tagesreflexion** | | | |
|  | | | |

Tabelle : Tagesprotokoll vom 28.5.2021

# Projekt-Dokumentation

## Kurzfassung

**Ausgangsituation**

Es gibt derzeit eine Zeiterfassung mit welcher die Betreuung ihre Zeit erfasst. Die Frau Monika Schmid welche für das Betreute Wohnen zuständig ist, hat jeden Tag alle Ihre mehreren Arbeitsstunden am Tag manuell zusammenzurechnen und leitet dessen Totalzeit an den Entwickler und Ausbildner Herr Valentin Marolf weiter, welcher das derzeitige Zeiterfassungssystem betreut, dieser muss dann diese Arbeitszeit von Frau Monika Schmid nachträglich im System erfassen. Da entsteht ein Problem, weil es viele manuelle Vorgänge der Benutzer braucht um Ihre Arbeitsstunden korrekt zu erfassen. Ein weiteres Problem ist das der ehemaligen Entwickler der derzeitigen Zeiterfassung schon lange pensioniert ist und sich die Applikation auf altem Stand haltet und darum schlecht zu erweitern ist.

Die Betreuung wünscht sich zudem noch das die Zeiterfassung auf den mobilen Geräten bedienbar ist und den Feiertagen und den nicht Feiertagen an Samstagen für die Berechnung von 1.5-fachen Zeit berücksichtigt werden.

**Umsetzung**

Die Umsetzung erfolgt auch in PHP, schliesslich war die bisherige Zeiterfassung auch in PHP und eine Webapplikation. Die Authentifizierung des Benutzers erfolgt auch mit dem LDAP der Rafisa, nur wird hierfür eine Login Maske und kein Popup erstellt. Die Speicherung erfolgt in der Datenbank MongoDB und nicht in einer SQL rationalen Datenbank. Da in MongoDB die Dokumentenstruktur von Daten viel Dynamischer gespeichert werden können, wie das es möglich sein sollte, dass beliebig viele Einträgen von Arbeitsstunden am selben Tag gespeichert werden sollten. Genauso ist es einfacher in MongoDB die Daten von Objekten unterschiedlicher Form abzuspeichern und diese dann nach dem Benutzer der neuen Applikation zu filtern.

Um die neue Zeiterfassung auch für mobile Geräten tauglich zu machen, wird diese responsive in HTML und CSS mit dem Framework Bootstrap in SCSS umgesetzt.

**Ergebnis**

Eine neue und erweiterbare Zeiterfassung, auf dem neusten Code-Stand von PHP, welche die gleichen Anforderungen der derzeitigen Zeiterfassung erfüllt und beliebig viele Arbeitseinsätze am Tag korrekt zusammenrechnet. Die Verrechnung von Arbeitsstunden an Feiertagen oder Sonntagen auf die Betreuung zugeschnitten und auf mobilen Geräten für das starten und beenden der Zeiterfassung bedienbar ist.

## Informieren

**Vorarbeiten**

Für die Feiertage hat Herr Valentin Marolf bereits die Geschäftsleitung dazu aufgefordert die Samstage auf Feiertage miteinzuschliessen. Anschliessen hat er mir einen API Aufruf ein JSON vom derzeitigen Zeiterfassung aufbereitet, welches in der Realisierung genauer in Betracht gezogen wird. Damit ist es möglich nur in einem von zwei Zeiterfassungen die Feiertage einzutragen und die neue Zeiterfassung weiss somit automatisch Bescheid, wenn ein Feiertag ist oder nicht.

**Analysieren der Aufgabenstellung**

Das einlesen der Detaillierte Aufgabenstellung meiner IPA.

**Titel der Arbeit**

Dieser Titel wurde als Titel vom IPA Bericht aufgenommen.

**Ausgangslage**

Seit 2021 bietet die Rafusa Informatik ein betreutes Wohnen an. Mehrere Arbeitseinsätze werden gestückelt über den Tag von der Betreuung abgeleistet. Die bisherige Zeiterfassung von der Rafisa deckt diesen Fall nicht ab und ist schwer erweiterbar. Aus diesem Grund will die Geschäftsleitung eine Applikation, mit welcher die Betreuung ihre Arbeitszeit erfassen und auswerten kann.

Detaillierte Aufgabenstellung

Klassen sollten wie in Java Programmierstil entsprechen, also Anfang groß und jedes Wort im selben wieder groß angefangen werden wie zum Beispiel für eine Rafisa Internal Organisation Mongo Collection Klasse heißt der Name wie folgt “RIOMongoCollection”.

Auf die Eigenschaften eines Objekts sollten via Methoden und nicht direkt zugergriffen werden.

Für das Login wird die Authentifizierung via ein LDAP von der Rafisa realisiert werden.

Nur Benutzer welche eingeloggt sind können die Zeiterfassung starten oder beenden, wenn diese bereits aktiv ist beenden und wenn inaktiv starten.

Es sind am Tag beliebig viele Arbeitseinträge für jeden Benutzer möglich.

Am Ende von jedem Tag müssen von jeweiligem Tag die Tages-, Wochen- und Monatssaldo berechnet werden.

Zur Dokumentation in der Planung sollte ein Use-Case Diagramm konzipiert werden. Für spätere möglich anfallenden Erweiterungen, sollte ein ERM und ein Aktivitätsdiagramm erstellt werden.

Bei der Eingabemaske vom Login sollte diese gleich dem Aufbau der bisherigen Zeiterfassung entsprechen. Zu den Elementen gehören, ein Textfeld für den Benutzername, ein Passwortfeld für das Passwort und ein Absenden Button für das Anmeldeformular.

Bei der Eingabemaske von der Zeiterfassung gibt es den nicht gestarteten und den gestarteten Zustand der Zeiterfassung des jeweiligen angemeldeten Benutzers. Dabei hat als Element derselbe Button im jeweiligen Zustand die Beschriftung “Zeiterfassung starten” oder “Zeiterfassung stoppen”.

Bei der Eingabemaske von der Übersicht sollte eine Tabelle vom selektierten Monat abgebildet werden. Die Arbeitseinträge sollten in aufeinanderfolgende Reihenfolge erscheinen. Tabellentitel sollte der jeweilige Monat und die Jahreszahl sein, die Weiterleitung durch die Navigation zur Übersicht ist immer der momentane Monat als Standardwert gesetzt. Links davon sollte zum letzten angrenzenden Monat und rechts zum nächsten Monat, wenn nicht in der Zukunftsform nicht ausgegraut und navigiert werden können. Die Arbeitseinträge sollten folgende Spalten mit Eigenschaften aufgezeigt werden: Datum des Eintrages, Startzeit des Eintrages, Endzeit des Eintrages, Soll- und Ist-Arbeitszeit des dazugehörigen Tages sowie die Differenz dieser zwei Werte und die Zeitsaldi für die jeweilige Woche, Monat und Total. Daneben noch jeweils ein Feld für allfällige Textkommentare. Bei jedem Arbeitseintrag sollte ein klickbares Symbol stehen, welches als Navigierung/Aufruf zur Korrektur vom jeweiligen Arbeitseintrag dienen sollte. Unter allen Arbeitseinträgen kommt eine Zeile hin, welche die Totalwerte dieses Monats abgebildet werden.

Bei der Eingabemaske von der Korrektur, sollten nochmals alle Eigenschaften eines Arbeitseintrages aufgezeigt werden. Neben jeder Eigenschaft (erste Spalte) sollte ein Korrekturfeld (zweite Spalte) vorhanden sein. Falls eine Korrektur oder mehrere Korrekturen anfallen wird der Kommentar zu einem Pflichtfeld. In der dritten Spalte sollte eine Erklärung zur jeweiligen Eigenschaft stehen, welche eins zu eins aus der bisherigen Zeiterfassung übernommen werden kann. Nach jeder Korrektur, sollte die letzte Korrektur mit welchem Benutzer und Datum sowie Uhrzeit angezeigt werden.

Täglichen Verarbeitung

Wird am Ende von jedem Tag der Saldo ausgerechnet. Die Arbeitsstunden an Sonn- und Feiertagen werden 1.5-fach berechnet.

Fehlerbehandlung

Der Kontakt zum LDAP nicht möglich, der Benutzer wird mit einer Fehlermeldung informieren.

Anfragen mit Fehlerhaften Date, wird der Benutzer mit einer Fehlermeldung informieren, welche ihn auf die Inkompatibilität der Daten hinweist.

PHP-Fehlermeldungen werden nicht angezeigt, der Programm-Fluss wird gestoppt und der Benutzer wird zur Login-Maske weitergeleitet.

Tests

Testkonzept erstellen

Tests gemäss Konzept durchführen

Folgende Testszenarien müssen mindestens berücksichtigt werden.

Korrekte Funktion der Masken

Fehler abfangen

Fehlermeldungen nach Vorgaben anzeigen

**Zusatzinformationen einholen**

Die Absenzen nur die Differenz von Präsenzzeit und Pflichtzeit beeinflusst und nicht die Pflichtzeit selbst.

Und das die korrigierte Gesamtsaldo auf die Künftige Einträge beeinflusst solange in der Zukunft keine weitere Gesamtsaldo Korrektur anfallen, dazwischen bilden sich dann Lücken welche dann von der letzten Gesamtsaldo Korrektur überschrieben werden. Vom Tag mit Gesamtsaldo Korrektur hat alles Rückwirkende keinen Einfluss auf das Gesamtsaldo in der Zukunft.

Die letzte Gesamtsaldo Korrektur Beeinflusst das Gesamtsaldo in der Zukunft.

Gesamtsaldo Korrektur ignoriert aller Arbeitsbeginn und Arbeitsende vom selben Tag und berücksichtigen erst den vom Folgetag an alles weitere.

**Ergänzungen zur Aufgabenstellung**

Abmeldung / Angemeldet

* Nur auf einem Gerät auf einen Benutzer gleichzeitig angemeldet sein.
* Die Option Abmeldung wird direkt nach der Anmeldung in der Zeiterfassungsmaske angezeigt.

Masken-Navigation

* Eine Masken-Navigation, welche zur Pflichtzeit, Übersicht und zur Zeiterfassung führt.

Übersicht-Navigation

* Statt Monat in der Zukunft ausgegraut nichts anzeigen.

Mobiltauglichkeit

* Monika Schmid von der Betreuung möchte gerne, dass die Webapplikation auf den mobilen Geräten für das starten und beenden der Zeiterfassung, tauglich ist.
* Design Responsive mit /Bootstrap CSS Framework/ für Mobile Bedienung der Betreuung, nur für Login- und Zeiterfassungsmaske.

Übersicht Monatstotal

* Monatstotal in der Übersichtsmaske im Detail.
* Wochentage sind in der Spalte Datum auch ersichtlich.
* Unten ein Bild mit allen Attributen für die Übersicht, welche 1:1 für die neue Applikation übernommen werden kann.

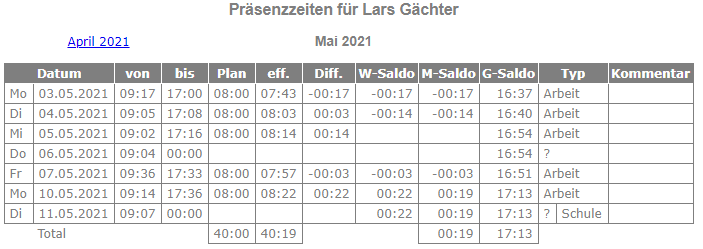
[](https://intern.stiftung-ifa.ch/betrieb/mitarbeiter/praesenzzeit_anzeige.php?popup=j&datum=11.05.2021)

Abbildung : Übersicht

Quelle: Screenshot von Lars Gächter Link <https://intern.stiftung-ifa.ch/betrieb/mitarbeiter/praesenzzeit_anzeige.php?popup=j&datum=11.05.2021>

Benutzer Pflichtzeit

* Weitere Maske mit der Anpassung der Pflichtzeit, welche ab dem nächsten Folgetag gesetzt sein wird.

Korrekturmaske im Detail

* Datum ist nicht veränderbar.
* Arbeitsbeginn ist veränderbar, darf nicht grösser als Arbeitsende sein.
* Arbeitsende ist veränderbar, darf nicht kleiner als Arbeitsbeginn sein.
* Arbeitsbeginn und Arbeitsende, können gleichgross sein, dürfen nicht andere Arbeitsbeginn und Arbeitsende am selben Tag überschneiden, kann nicht negativ ausfallen.
* Pflichtzeit ist veränderbar, kann nicht negativ ausfallen.
* Zeitgutschrift ist veränderbar und standartmässig 0 Stunden und 0 Minuten, kann nicht negativ ausfallen.
* Präsenzzeit ist veränderbar und überschreibt / ignoriert die Ausrechnung aus Arbeitsbeginn und Arbeitsende, kann nicht negativ ausfallen.
* Geleistete Arbeitszeit wird vorläufig genau wie die Präsenzzeit behandelt ohne Pausen-Abzug.
* Abweichung ist nicht veränderbar, kann negativ ausfallen.
* Abwesend?, kann die Pflichtzeit auf 0 setzen in der Differenz Rechnung der eigentlichen Präsenzzeit
* Vormittags, kann die Pflichtzeit halbieren in der Differenz Rechnung der eigentlichen Präsenzzeit
* Nachmittags, kann die Pflichtzeit halbieren in der Differenz Rechnung der eigentlichen Präsenzzeit
* Abwesend?, Vormittags und Nachmittags haben folgende Optionen
  + (nicht extern), ist immer standardmässig gesetzt
  + Ergänzungskurs, kann die Pflichtzeit halbieren oder auf 0 setzen
  + Telearbeit
  + Zügeln, kann die Pflichtzeit halbieren oder auf 0 setzen
  + Abwesend
  + Kompensation
  + offizieller Feiertag, kann die Pflichtzeit halbieren oder auf 0 setzen
  + Ferien, kann die Pflichtzeit halbieren oder auf 0 setzen
  + Betriebsferien, kann die Pflichtzeit halbieren oder auf 0 setzen
  + Bonus
  + Krank, kann die Pflichtzeit halbieren oder auf 0 setzen
  + Arzt, kann die Pflichtzeit halbieren oder auf 0 setzen
  + Unfall, kann die Pflichtzeit halbieren oder auf 0 setzen
  + Militär, kann die Pflichtzeit halbieren oder auf 0 setzen
  + Besprechung
  + †, kann die Pflichtzeit halbieren oder auf 0 setzen
* Wochensaldo ist nicht veränderbar, kann negativ ausfallen.
* Monatssaldo ist nicht veränderbar, kann negativ ausfallen.
* Gesamtsaldo ist veränderbar, kann negativ ausfallen.
* Kommentar ist veränderbar und ist bei jeglicher Korrektur oder Veränderung ein Plicht-Feld.

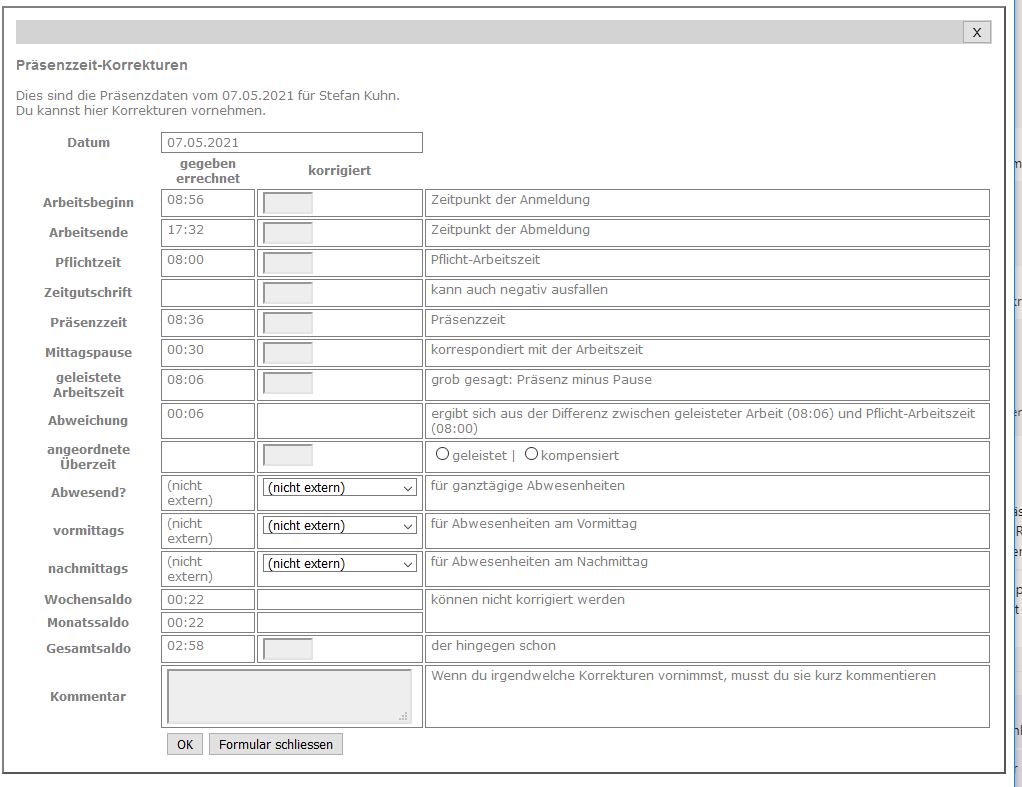


Abbildung : Korrektur

Quelle: Screenshot von Stefan Kuhn

* Nach jeder Salo beeinflussbaren Korrektur muss von diesem Datum an, jedes und jedes danach folgende Datum bis mit gestern neu berechnet und aktualisiert werden.
* Die Eingabefelder Mittagspause und angeordnete Überzeit wie im Sceenshot sind ausgeschlossen.

## Planen

Tätigkeiten festlegen für den Zeitplan

Projektmanagement-Methode: IPERKA

Use Cases

* Anmeldung/Abmeldung
* Zeiterfassung starten und beenden
* Pflichtzeit anpassen
* Tageskorrekturen

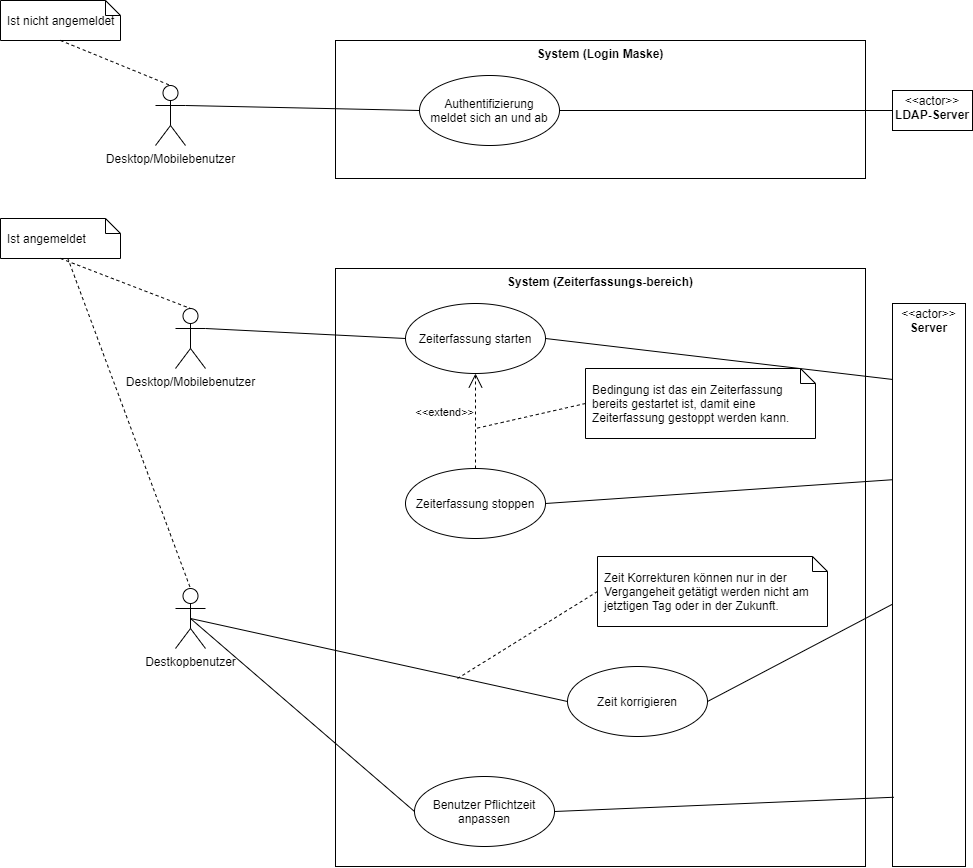


Abbildung : Use Cases

Aktivitäten Diagramm

Login

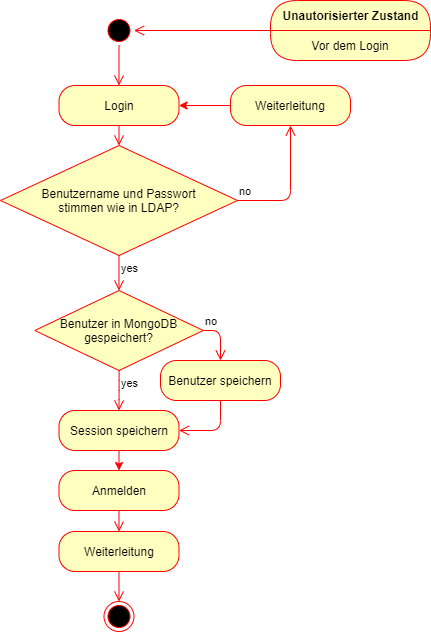
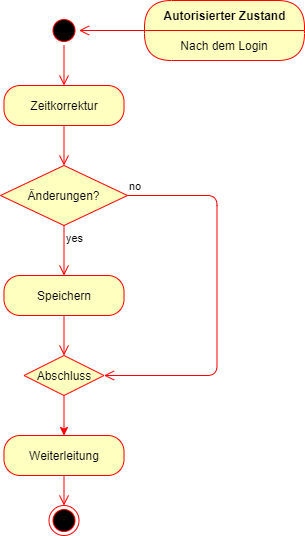


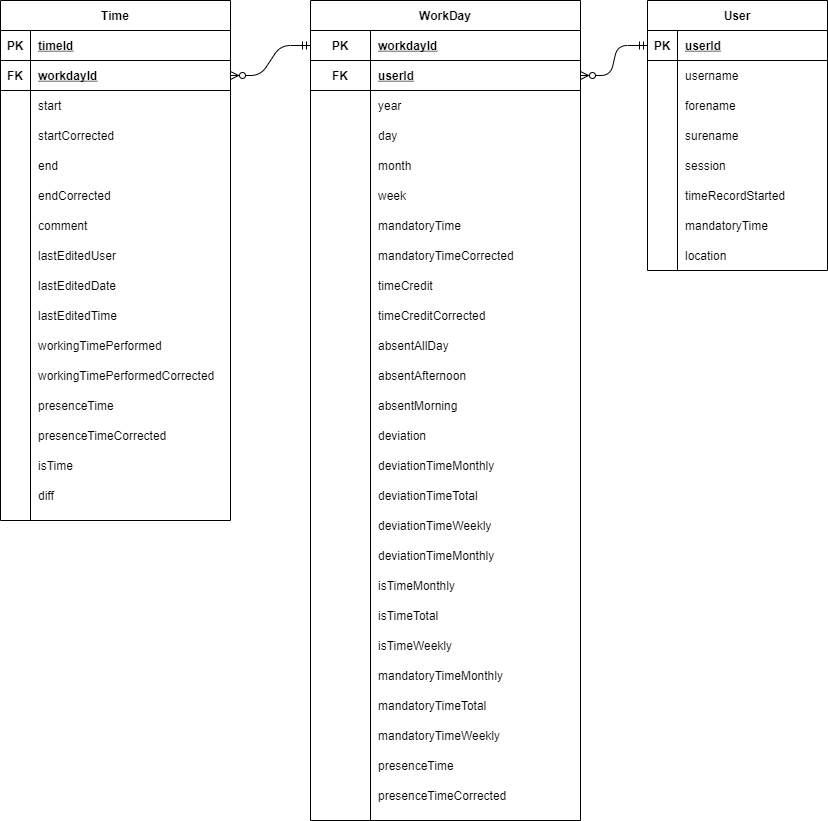
Abbildung : Aktivitäten Diagramm (Login)

Generelle Korrektur in der Zeiterfassung

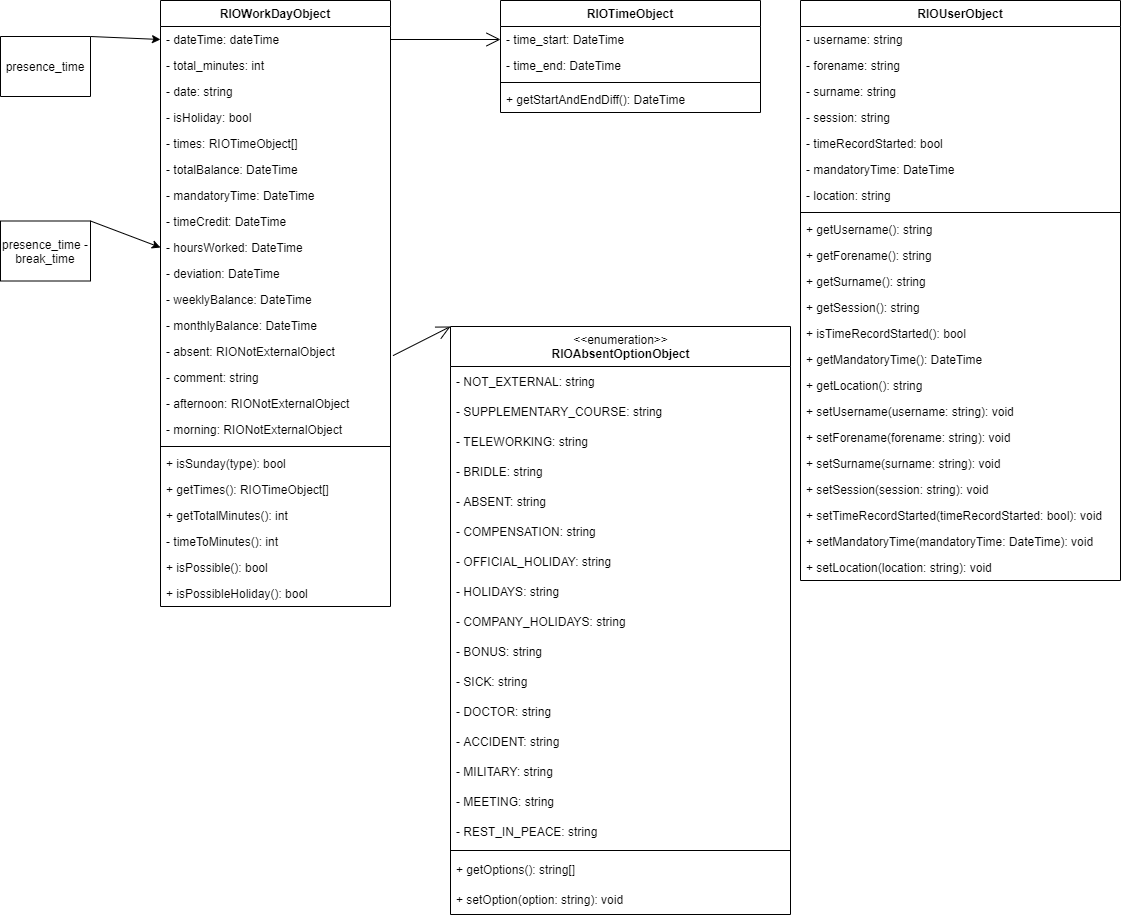


Detaillierte Korrekturen in der Zeiterfassung

ERM



UML Klassen



### Testkonzept

Ich werde auf einem Windows Rechner auf welchem PHPStorm mit PHP läuft die Backend Tests durchführen, dafür benutze ich PHPUnit.

Auf einem Rechner auf dem das OS (Betriebssystem) Windows installiert ist, werden in der IDE (Integrierte Entwicklungsumgebung) PHPStorm die Backend Tests mit PHPUnit im Git Projekt durchgeführt.

Die Manuellen Tests werden von mir Lars Gächter als Entwickler und der Kundin Monika Schmid bei der Freigabe am 26.05.2021 durchgeführt. Diese Manuellen Tests werden in einer virtuellen Maschine namens VirtualBox durchgeführt. Dabei laufen 2 Server und 1 Client im selben Netzwerk, welche miteinander kommunizieren können. Auf dem einen Server läuft nur die Datenbank namens LDAP und auf dem anderen Server läuft die Datenbank namens MongoDB, die Webservern PHP und Apache und für die Projekt-Aktualisierung das Git. Auf dem Client wird das Frontend via dem Google Chrome Browser aufgerufen, um die Mobile Anwendung im Test auch miteinzuschliessen, wird im Google Chrome der Entwickler Modus aktiviert und die Mobile Ansicht abgerufen.

Testdaten

Ein Mitarbeiter mit Arbeitseinträge vom ganzen April bis jetzt im Mai.

## Entscheiden

Entscheidungsmatrix

Zeitklasse

Zeitzone von DateTime festlegen

Curl PHP oder Symfony

MngoDB VS SQL Datenbank

Kontroller Bereiche für nicht- und angemeldeter Benutzer

## Realisieren

\*.sh Datei ausführbar machen. <https://askubuntu.com/questions/122428/how-to-run-sh-file>

Ordner rekursiv entfernen. <https://askubuntu.com/questions/217893/how-to-delete-a-non-empty-directory-in-terminal>

Entfernen von Dateien ausser einer Datei. <https://www.tecmint.com/delete-all-files-in-directory-except-one-few-file-extensions/>

**Admin-Handbuch**

Die Installationsanleitung ist im Anhang mit dem Git Repository.

Das Inhaltsverzeichnis befindet sich im Ordner «arbeitszeitserfassung\_wohnen» und heisst README.md. Dabei habe ich die Entwicklung und Produktion der Applikation in zwei weitere verlinkten \*.md Dateien unterteilt.

Intallation vom Projekt

Klonen vom Git-Repository

git clone https://gitlab.rafisa.org/l.gaechter/arbeitszeitserfassung\_wohnen.git

Einrichten der Umgebungsvariablen

cp .env.example .env

Installieren von Composer

chmod +x install\_composer.sh

./install\_composer.sh

php composer.phar i

Installieren der npm Packages, generiert einen node\_modules Ordner im arbeitszeitserfassung\_wohnen Ordner.

npm init

npm i bootstrap

npm i sass

npm i

SCSS zu komprimierten CSS parsen und JS zu komprimierten JS

Für Produktion (wandelt einmal um)

npm run css:prod

npm run js:prod

Für Entwicklung (überwacht vorlaufend Veränderungen und wandelt um)

npm run css:dev

npm run js:dev

Apache und Berechtigung

sudo chown -R www-data arbeitszeitserfassung\_wohnen

**Performance Optimierung**

Kurzbeschreibung

Nach dem alle Arbeitstage inklusive des heutigen Tags aus der Datenbank gezogen wurden, musste der der heutige Tag aus der liste entfernt werden um nicht in der Übersicht zu erscheinen. Dabei wir über eine nach Datum sortierte Liste von Arbeitstagen iteriert.

Performance Problem

Bei der Iteration wurden Mal 2 Instanzen gebildet und viele diverse Methoden aufgerufen. Die Zeit beträgt bei über 30 Tagen Einträge rund 5 Sekunden und ist inakzeptabel.

Lösung

Vor der Iteration nötige Instanzen gebildet und in der Iteration ein Array Objekt mit Datum Key mit dem Datum welches vor der Iteration gesetzt wird als String vergleichen. Dies hat viele Instanzen und Methoden Aufrufe eliminiert und Zeit gespart. Die Zeit beträgt bei über 30 Tagen Einträge rund 100 Millisekunden und ist in einem akzeptablen bereich.

**Recherchen**

Datum vom nächsten Tag [1]

Datum von gestern [2]

Zum Beispiel +07:35 zu + und 07:35 aufteilen [3]

substr(‘+07:35’,0,0);

substr(‘+07:35’,1);

Architektur entwerfen

MVC

Datenbanken

MongoDB

* Datenbank
* Sammlungen (Collection)
* Dokumentenaufbau

Geo-Location/Zeitserver

RIODateTimeFactory

Logging

Persistenzlogik

Validationsklassen

Buisnesslogik

Fehlerbehandlung

## Kontrollieren

Präsentations-Masken

Login-Maske

Desktop

Mobile

Desktop

Mobile

Masken-Navigation

Desktop

Mobile

Zeiterfassungsmaske (Logout)

Desktop

Mobile

Benutzer Pflichtzeit

Desktop

Mobile

Übersichtsmaske

Desktop

Benutzer Tageskorrektur

Desktop

### Testprotokoll

Scans von Testprotokollen

## Auswerten

### Reflexion

PHPUnit Tests für die Auswertung

|  |  |  |
| --- | --- | --- |
| ID / Bezeichnung | 1 | Ist Samstag Test |
| Beschreibung | Mit einem künftigen oder vergangenen Datum, welches ein Samstag ist wir getestet. | |
| Testvoraussetzung | Ein DateTime Objekt ist instanziiert und die isSaturday Methode wir aufgerufen. | |
| Testschritte  (Schritt / erwartetes Resultat) | Laden der Umgebungsvariablen, ein WorkDay Object mit einem Samstag Datum erstellen und das WorkDay Objekt fragen ob ein Samstag gesetzt ist. | |
| Erwartetes Ergebnis | OK | |
| Erhaltenes Ergebnis | OK | |

|  |  |
| --- | --- |
| Tester | Lars Gächter |
| Datum Testdurchführung | 27.05.2021 |
| Fehlerbeschreibung | Keine |

Code:

*/\*\*  
 \** ***@throws*** *\Symfony\Contracts\HttpClient\Exception\TransportExceptionInterface  
 \*/*public function testIsSaturday(): void  
{  
 (new Dotenv())->bootEnv(dirname(*\_\_DIR\_\_*).'\..\..\.env');  
 $dateTime = RIODateTimeFactory::*getDateTime*("2021-05-22");  
 $workDay = new RIOWorkDayObject($dateTime);  
 $result = $workDay->isSaturday();  
 $this->assertEquals(true,$result,"Test has failed!");  
}

|  |  |  |
| --- | --- | --- |
| ID / Bezeichnung | 2 | Ist Sonntag Test |
| Beschreibung | Mit einem künftigen oder vergangenen Datum, welches ein Sonntag ist wir getestet. | |
| Testvoraussetzung | Ein DateTime Objekt ist instanziiert und die isSunday Methode wir aufgerufen. | |
| Testschritte  (Schritt / erwartetes Resultat) | Laden der Umgebungsvariablen, ein WorkDay Object mit einem Sonntag Datum erstellen und das WorkDay Objekt fragen ob ein Sonntag gesetzt ist. | |
| Erwartetes Ergebnis | OK | |
| Erhaltenes Ergebnis | OK | |

|  |  |
| --- | --- |
| Tester | Lars Gächter |
| Datum Testdurchführung | 27.05.2021 |
| Fehlerbeschreibung | Keine |

Code:

*/\*\*  
 \** ***@throws*** *\Symfony\Contracts\HttpClient\Exception\TransportExceptionInterface  
 \*/*public function testIsSunday(): void  
{  
 (new Dotenv())->bootEnv(dirname(*\_\_DIR\_\_*).'\..\..\.env');  
 $dateTime = RIODateTimeFactory::*getDateTime*("2021-05-23");  
 $workDay = new RIOWorkDayObject($dateTime);  
 $result = $workDay->isSunday();  
 $this->assertEquals(true,$result,"Test has failed!");  
}

|  |  |  |
| --- | --- | --- |
| ID / Bezeichnung | 3 | Ist Feiertag Test |
| Beschreibung | Mit einem künftigen oder vergangenen Datum, welches ein Feiertag ist wir getestet. | |
| Testvoraussetzung | Ein DateTime Objekt ist instanziiert und die isRIOHoliday Methode wir aufgerufen. | |
| Testschritte  (Schritt / erwartetes Resultat) | Laden der Umgebungsvariablen, ein WorkDay Object mit einem Feiertag Datum erstellen und das WorkDay Objekt fragen ob ein Feiertag gesetzt ist. | |
| Erwartetes Ergebnis | OK | |
| Erhaltenes Ergebnis | OK | |

|  |  |
| --- | --- |
| Tester | Lars Gächter |
| Datum Testdurchführung | 27.05.2021 |
| Fehlerbeschreibung | Keine |

Code:

*/\*\*  
 \* Test  
 \*  
 \** ***@throws*** *\PHPUnit\Framework\ExpectationFailedException|\Exception  
 \** ***@throws*** *\Symfony\Contracts\HttpClient\Exception\TransportExceptionInterface  
 \*/*public function testIsHoliday(): void  
{  
 (new Dotenv())->bootEnv(dirname(*\_\_DIR\_\_*).'\..\..\.env');  
 $dateTime = RIODateTimeFactory::*getDateTime*("2021-05-24");  
 $workDay = new RIOWorkDayObject($dateTime);  
 $result = $workDay->isRIOHoliday();  
 $this->assertEquals(true,$result,"Test has failed!");  
}

|  |  |  |
| --- | --- | --- |
| ID / Bezeichnung | 4 | Ist heute Pflichtzeit Test |
| Beschreibung | Mit dem heutigen Datum wird getestet.  Ist der Test rot ist keine Pflichtzeit, ist der Test grün gilt eine Pflichtzeit. | |
| Testvoraussetzung | Ein DateTime Objekt ist instanziiert und die isRIOHoliday oder isSunday oder isSaturday Methoden werden aufgerufen. | |
| Testschritte  (Schritt / erwartetes Resultat) | Laden der Umgebungsvariablen, ein WorkDay Object mit dem heutigen Datum (siehe Datum Testdurchführung) erstellen und das WorkDay Objekt fragen ob ein Feiertag, ein Sonntag oder ein Samstag ist gesetzt ist. | |
| Erwartetes Ergebnis | OK | |
| Erhaltenes Ergebnis | OK | |

|  |  |
| --- | --- |
| Tester | Lars Gächter |
| Datum Testdurchführung | 27.05.2021 |
| Fehlerbeschreibung | Keine |

Code:

*/\*\*  
 \** ***@throws*** *\Symfony\Contracts\HttpClient\Exception\TransportExceptionInterface  
 \*/*public function testMandatoryTimeToday(): void  
{  
 (new Dotenv())->bootEnv(dirname(*\_\_DIR\_\_*).'\..\..\.env');  
 $workDay = new RIOWorkDayObject();  
 $result = $workDay->isRIOHoliday() ||  
 $workDay->isSunday() ||  
 $workDay->isSaturday();  
 $this->assertEquals(false,$result,"Test has failed!");  
}

Meldung der Resultate:

Testing started at 14:43 ...

PHPUnit 9.5.4 by Sebastian Bergmann and contributors.

Time: 00:00.476, Memory: 6.00 MB

OK (4 tests, 4 assertions)

Process finished with exit code 0

### Schlusswort

## Glossar

Composer

CSS

Curl

HTML

IPERKA-Methode

* Besteht aus 6 abgekürzten Buchstaben welchen folgenden Schritten beschreiben. Das Informieren, das Planen, das Entscheiden, das Realisieren, das Kontrollieren und das Auswerten.

JSON

LDAP

npm

MongoDB

MVC

Responsive Design

Sass

* Ausgeschrieben Syntactically Awesome Style Sheets und ist eine Stylesheet-Sprache, welche CSS mit Kontrollstrukturen vereint. Unterstützt die Syntax von SCSS, welcher zu CSS kompiliert wird.

Symfony

Twig

Ubuntu

UTF-8

## Quellenverzeichnis

Stackoverflow kommentare günder Haken

|  |  |
| --- | --- |
| [1] | J. Conde, «stackoverflow,» 7 1 2014. [Online]. Available: https://stackoverflow.com/questions/14460518/php-get-tomorrows-date-from-date?answertab=active#tab-top. [Zugriff am 25 5 2021]. |
| [2] | «php.net,» [Online]. Available: https://www.php.net/manual/en/datetime.formats.relative.php. [Zugriff am 26 05 2021]. |
| [3] | «php.net,» [Online]. Available: https://www.php.net/manual/en/function.substr.php. |
| [4] | «Composer,» 17 05 2021. [Online]. Available: https://getcomposer.org/download/. |

## Literaturverzeichnis

Php.net nicht Kommentare sondern Fakten

# Anhang

## Abbildungsverzeichnis

[Abbildung 1: Baumdiagramm 7](#_Toc73022374)

[Abbildung 2: Projektzeitplan 9](#_Toc73022375)

[Abbildung 3: Übersicht 27](#_Toc73022376)

[Abbildung 4: Korrektur 28](#_Toc73022377)

[Abbildung 5: Use Cases 29](#_Toc73022378)

[Abbildung 6: Aktivitäten Diagramm (Login) 30](#_Toc73022379)

## Tabellenverzeichnis

[Tabelle 2: Tagesprotokoll vom 12.5.2021 11](#_Toc73022039)

[Tabelle 4: Tagesprotokoll vom 9.5.2017 13](#_Toc73022040)

[Tabelle 5: Tagesprotokoll vom 17.5.2021 14](#_Toc73022041)

[Tabelle 6: Tagesprotokoll vom 18.5.2021 15](#_Toc73022042)

[Tabelle 7: Tagesprotokoll vom 19.5.2021 16](#_Toc73022043)

[Tabelle 8: Tagesprotokoll vom 20.5.2021 17](#_Toc73022044)

[Tabelle 9: Tagesprotokoll vom 21.5.2021 18](#_Toc73022045)

[Tabelle 11: Tagesprotokoll vom 25.5.2021 19](#_Toc73022046)

[Tabelle 12: Tagesprotokoll vom 26.5.2021 20](#_Toc73022047)

[Tabelle 12: Tagesprotokoll vom 27.5.2021 21](#_Toc73022048)

[Tabelle 12: Tagesprotokoll vom 28.5.2021 22](#_Toc73022049)

## Listing des Programmcodes