|  |
| --- |
|  |
| Probe IPA |
| Individuelle Praktische Arbeit M223 |
|  |
| **Olivia Pawlowitz** |
| **01.11.2017** |

|  |
| --- |
| [Geben Sie hier das Exposee für das Dokument ein. Das Exposee ist meist eine Kurzbeschreibung des Dokumentinhalts. Geben Sie hier das Exposee für das Dokument ein. Das Exposee ist meist eine Kurzbeschreibung des Dokumentinhalts.] |

Inhalt

[1 Vorwort 3](#_Toc497378256)

[1.1 Risiken 3](#_Toc497378257)

[1.1.2 Externe Einflüsse 3](#_Toc497378258)

[1.1.3 Falsche Zeiteinschätzung 3](#_Toc497378259)

[1.1.4 Datenverlust 3](#_Toc497378260)

[2 Aufgabenstellung 4](#_Toc497378261)

[3 Deklarationen 7](#_Toc497378262)

[3.1 Vorkenntnisse 7](#_Toc497378263)

[Fachliche Vorkenntnisse 7](#_Toc497378264)

[3.1.1 Architektur 7](#_Toc497378265)

[3 Projektmethode IPERKA 8](#_Toc497378266)

[3.1 Begründung der Wahl 8](#_Toc497378267)

[4 Zeitmanagement 9](#_Toc497378268)

[4.1 Struktur 9](#_Toc497378269)

[4.1.1 Meilensteine 9](#_Toc497378270)

[6 Management Summary 11](#_Toc497378271)

[6.1 Projektübersicht 11](#_Toc497378272)

[6.2 Lehrbetrieb 11](#_Toc497378273)

[6.3 Involvierte Personen 11](#_Toc497378274)

[7 Entwicklungsumgebung 12](#_Toc497378275)

[7.1 Rechner 12](#_Toc497378276)

[8 Planung 13](#_Toc497378277)

[8.1 Anforderungen 13](#_Toc497378278)

[8.1.1 Frontend 13](#_Toc497378279)

[8.1.2 Backend 13](#_Toc497378280)

[8.2.1 Versionsverwaltung 13](#_Toc497378281)

[8.2.2 Subversion (SVN) 13](#_Toc497378282)

[8.3 Benutzeroberfläche 15](#_Toc497378283)

[8.4 Datenbank 15](#_Toc497378284)

[ERM einfügen!! 15](#_Toc497378285)

[8.4.1 Tabellendefinitionen 15](#_Toc497378286)

[9 Entscheidung 15](#_Toc497378287)

[9.1 Versionsverwaltung 15](#_Toc497378288)

[9.2 Datenbank 15](#_Toc497378289)

[8.5 Use Case 16](#_Toc497378290)

[8.5.1 Use Case: Login 16](#_Toc497378291)

[8.5.2 Use Case: Register 16](#_Toc497378292)

[8.5.3 Use Case: Create Ticket 16](#_Toc497378293)

[8.5.4 Use Case: View Ticket 17](#_Toc497378294)

[8.5.5 Use Case: Delete Ticket 17](#_Toc497378295)

[4 18](#_Toc497378296)

[5 Analyse 18](#_Toc497378297)

[5.1 IST-Analyse 18](#_Toc497378298)

[5.1.1 SOLL-Analyse 18](#_Toc497378299)

[6 Realisierung 18](#_Toc497378300)

# 1 Vorwort

Anhand dieser Probe IPA-Dokumentation wird die gesamte, individuelle Praxisarbeit festgehalten und dokumentiert. Alle projektrelevanten Informationen sowie der eigentliche Projektverlauf sind somit klar ersichtlich. Diese Dokumentation wurde im November 2017 von Olivia Pawlowitz im Rahmen des Moduls 223 zur Vorbereitung auf den Lehrabschluss als Informatiker, Schwerpunkt Applikationsentwicklung, verfasst.

# 1.1 Risiken

## 1.1.2 Externe Einflüsse

Die Projektarbeit ist teilweise abhängig von externen Einflüssen, welche die planmässige Durchführung durcheinander bringen könnten. Zu den externen Einflüssen gehören beispielsweise Server, Netzwerk, Internet Zugang, usw.

## 1.1.3 Falsche Zeiteinschätzung

Meistens ist es so, dass der Zeitplan nicht auf das Detail genau eingehalten werden kann. Damit das Projekt trotzdem planungsgemäss erledigt werden kann, wurden Zeitreserven in die Planung mit einbezogen.

## 1.1.4 Datenverlust

Die Daten werden regelmässig gesichert, um den Verlust von relevanten Projektdaten vorzubeugen. Es besteht jedoch ein geringes Risiko, dass Daten während der Projektarbeit verloren gehen könnten.

**1.1.5 Krankheit/Unfall**

Die Projektarbeit kann aus gesundheitlichen Gründen jederzeit unterbrochen werden. Der Endtermin des Projekts verschiebt sich entsprechend der verlorenen Zeit.

# 2 Aufgabenstellung

**Projekt**

Projekttitel: Erstellen einer Multi User Applikation

Ticketing System

Im Auftrag von : Remo Steinmann Modul 223  
Auftragnehmer: Olivia Pawlowitz  
Starttermin: 01. November 2017  
Geplante Projektzeit: 5 Arbeitstage(40h) davon ca. 3 Tage Entwicklung(24h)

Erstellen einer Multi User Applikation, beinhaltend Frontend, Backend, Anbindung an relationaler Datenbank.

Wieso ein Ticket-System?

Es gibt verschiedene Varianten, Kundenanfragen in einem Unternehmen zu bearbeiten. Ticket-Systeme grenzen sich von anderen Methoden ab, sie speichern einzelne Anfragen und bieten dem User eine geniale Übersicht über alle Anfragen. Die Tickets können kategorisiert und unzählige Male abgerufen werden.

Programmiersprache : PHP, JavaScript  
IDE: Eclipse  
DB: MySQL  
Framework: Bootstrap  
Library: JQuery

**Soll Zustand**

Der User kann sich registrieren und sich mit den Anmeldedaten einloggen. Es wird ein neues Fenster geöffnet und der Benutzer sieht die Hauptseite. Dort besteht für ihn die Möglichkeit ein neues Ticket zu erstellen oder alle Tickets anzusehen. Der Benutzer kann ein Ticket ansehen und löschen.

Das Projekt wird mit PHP und JavaScript objektorientiert realisiert. Die Daten werden in einer MySQL Datenbank gespeichert. Es werden 5 Tabellen angelegt.

-Kategorie   
-Abteilung   
-Priorität   
-Tickets   
-User

Beim Login bzw. registrieren werde ich, zum übertragen der Daten, Ajax verwenden. Das Passwort wird jeweils verschlüsselt in die DB eingetragen.

Nachdem der User sich angemeldet bzw. registriert hat, wird dieser auf die Hauptseite weitergeleitet. Dort werden ihm zwei Möglichkeiten angeboten. Entweder er erstellt ein Ticket, er sieht sich alle Tickets an oder er loggt sich wieder aus.

**Fallbeispiel 1: Ticket erstellen**

Der User möchte ein neues Ticket erstellen. Er hat nun die Möglichkeit sein Ticket zu erstellen und muss den jeweiligen Bereich für das Problem auswählen, die Wichtigkeit und die Abteilung in der er sich befindet. Der User gibt dem Ticket ein Titel und fügt ausserdem eine Beschreibung hinzu. Nun kann das Ticket veröffentlicht werden.

**Spezifikationen**

Es dürfen eine maximale Anzahl Wörter in der Beschreibung enthalten sein. Diese Anzahl reduziert sich auf 50 Wörter. Ein Upload von Bildern sprich Screenshots ist möglich und auch erwünscht. Die Anzahl der Bilder, welche hochgeladen werden können ist unbegrenzt.

**Fallbeispiel 2: Ticket ansehen**

Der Benutzer möchte nun alle Tickets ansehen. Es erscheint eine Tabelle mit allen erstellten Tickets. In dieser Tabelle sieht der User, wer Ticket xyz erstellt hat, welche Wichtigkeit es hat, zu welcher Abteilung es gehört und den Titel. Er kann die Tickets löschen und bearbeiten.

In der Ticketview ist es dem User möglich zu sehen, wann ein Ticket erstellt wurde: Datum & Uhrzeit.

**Fallbeispiel 3: User abmelden**

Der User möchte sich abmelden und klickt auf den Button „Logout“.

**Anforderungen an die Applikation**

-Folgende Anforderungen müssen erfüllt werden:

-Multi-User Applikation

-Relationale Datenbank

-Objektorientiert Programmieren

-Transaktionen

-Mehrere Clients müssen gleichzeitig auf den gleichen Datenbestand zugreifen

**Zusätzliche Kriterien**

-194 Plausibilierung der Benutzer-Eingaben

-166 Coding Style – lesbarer Code

-163 Design Doku

-123 Kommentar im Code

-164 Fehlerbehandlung

-130 Vollständiges ERM bzw. Datenmodell

-125 Gliederung des Programms

**Warum dieses Projekt zum Modul Multiuser-Applikationen passt:**

Alle Anforderungen, welche Anfangs Modul gestellt wurden, können erfüllt werden.

**Herausforderung: PHP objektorientiert**

Da ich bis zu diesem Zeitpunkt erst einmal PHP Objektorientiert programmiert habe, wird dies eine Hürde sein, in kürzester Zeit die Aufgabenstellung so genau wie möglich umzusetzen.

## 3 Deklarationen

## 3.1 Vorkenntnisse

In meiner bald 4-jährigen Lehre als Applikationsentwickler konnte ich mir die in diesem Abschnitt beschriebenen Vorkenntnisse erlernen, welche für das Projekt relevant sind.   
Projektrelevante Vorkenntnisse werden stichwortartig festgehalten:

### Fachliche Vorkenntnisse

|  |  |  |
| --- | --- | --- |
| Thema | Kenntnisse | Bemerkungen |
| HTML | sehr gute Kenntnisse | Module absolviert und Selbststudium |
| CSS | sehr gute Kenntnisse | Selbststudium |
| JavaScript | gute Kenntnisse | Selbststudium |
| SQL | sehr gute Kenntnisse | Erfahrung aus Praxis, Modul |
| PHP | gute Kenntnisse | Selbststudium |
| Java | Gute Kenntnisse | 1 Jahr Erfahrung in der IT-Abteilung |

## 3.1.1 Architektur

Die Arbeitsumgebung ist im Meeting Raum „Merkur“ im Gebäude der Berufsbilder der Siemens Schweiz.

# 3 Projektmethode IPERKA

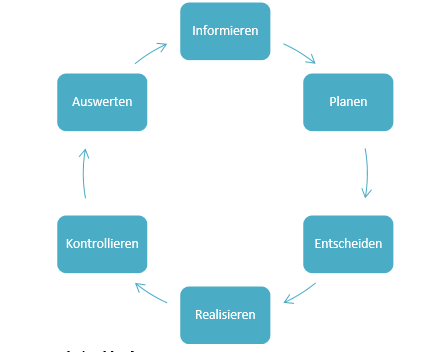
****

Abbildung : IPERKA Arbeitsablauf

In der „Abbildung 1: IPERKA Arbeitsablauf“ sind die einzelnen Projektphasen zu sehen, welche man mit IPERKA unterteilt. Das Ziel von IPERKA ist es, eine klare Struktur zu schaffen nach der gearbeitet wird. IPERKA enthält dabei alle relevanten Phasen eines Projekts.

## 3.1 Begründung der Wahl

Die Projektmanagement-Methode IPERKA haben wir schon am ersten Tag als Informatiker im Basislehrgang kennengelernt und stets bei jedem Projekt eingesetzt. Es bietet meiner Meinung nach einige Vorteile. Das Projekt lässt sich so in übersichtliche Teilschritte gliedern und es ist deshalb immer ersichtlich, in welchem Teilschritt man sich befindet.

Es gibt sicherlich weitere Projektmanagement-Methoden, die geeigneter für die Entwicklung einer Applikation wären, doch für so ein kleines Projekt, wie die IPA, ist IPERKA die einfachste und übersichtlichste Methode, um ein Projekt dieser Art durchzuführen. Zudem haben andere Methoden den Nachteil, dass sie zu viel Zeit für die Planung beanspruchen, was für die IPA nicht sehr günstig wäre.

# 4 Zeitmanagement

## 4.1 Struktur

Es macht Sinn, den Zeitplan nach der IPERKA Methode zu strukturieren, da ich diese Projektmanagement-Methode gewählt habe. Der Zeitplan ist dabei in zwei verschiedene Bereiche gegliedert, nämlich in Hauptarbeitsschritte und wiederkehrende Arbeitsschritte. Es wird in Form eines GANTT-Diagramms dargestellt.

Hier kommt der Zeitplan hinein.

## 4.1.1 Meilensteine

|  |  |  |
| --- | --- | --- |
| Meilenstein | Beschreibung | Datum |
| **Informieren**  SOLL Zeitplan | Der Zeitplan für die SOLL Stunden ist fertig definiert. | 01.11.2017 |
| **Informieren** Use Cases | Use Cases sind geschrieben und in die Dokumentation implementiert | 01.11.2017 |
| **Planen**  ERM | Klassendiagramm erstellt | 02.11.2017 |
| **Entscheiden** Klassen erstellt für DB Login implementiert Frontend implementiert |  | 02.11.2017 |
| **Realisieren** Klassen dokumentieren |  | 03.11.2017 |
| **Kontrollieren** Klassen dokumentieren |  | 08.11.2017 |
| **Auswerten**  Klassen dokumentieren |  | 10.11.2017 |

**5 Arbeitsjournale**

**5.1 Tagesablauf Mittwoch, 01.11.2017 (Tag 1)**

**5.2 Tagesablauf Donnerstag, 02.11.2017 (Tag 2)**

**5.3 Tagesablauf Freitag, 03.11.2017 (Tag 3)**

**5.4 Tagesablauf Mittwoch, 08.11.2017 (Tag 4)**

**5.5 Tagesablauf Freitag, 10.11.2017 (Tag 5)**

## 6 Management Summary

Die Experten bilden zusammen mit dem Fachausbildner und dem Kunden den Auftraggeber. Zusammen sind sie für die Formulierung der Aufgabenstellung und Bewertung der Projektarbeit als Ganzes zuständig.

6.1 Projektübersicht

Diese Dokumentation, die daraus entstandene Präsentation sowie das anschliessende Fachgespräch sind Bewertungsgrundlagen für meine praktische Lehrabschlussprüfung (Individuelle Praktische Arbeit, kurz IPA). Nach den Standards des Projekt Managements werden folgende Projektphasen dokumentiert:

* Aufgabenstellung
* Planung
* Analyse
* Realisierung
* Testphase
* Auswertung

Zusätzliche Informationen, welche in der nachfolgenden Dokumentation zu finden sind:

* Abbildungsverzeichnis
* Tabellenverzeichnis
* Quellenverzeichnis
* Protokolle
* Glossar
* Anhang

6.2 Lehrbetrieb

Siemens Schweiz AG   
Freilagerstrasse 40  
8047 Zürich

[Telefon](https://www.google.ch/search?site=async/lcl_akp&q=siemens+schweiz+ag+z%C3%BCrich+telefon&sa=X&ved=0ahUKEwiMu_bGgp3XAhWG2BoKHbaJB4AQ6BMIHzAF): 058 558 55 85

6.3 Involvierte Personen

Modulleiter: Remo Steinmann

# 7 Entwicklungsumgebung

## 7.1 Rechner

**7.1.1 Lokaler Entwicklungsserver**

**7.1.2 Entwicklungswerkzeug**

# 8 Planung

# 8.1 Anforderungen

Die Anforderungen an die Applikation werden anhand von Kriterien definiert. Die MussKriterien müssen dabei zwingend erfüllt werden.

### 8.1.1 Frontend

|  |  |  |
| --- | --- | --- |
| Kriterium | MUSS | KANN |
|  |  |  |
|  |  |  |
|  |  |  |

### 8.1.2 Backend

Für das Backend wurden folgende Kriterien definiert:

|  |  |  |
| --- | --- | --- |
| Kriterium | MUSS | KANN |
|  |  |  |
|  |  |  |

**8.2 Umsetzung**

### 8.2.1 Versionsverwaltung

Die Wahl des Sicherungssystems in der Siemens Schweiz AG ist generell dem Entwickler überlassen, da es keine Richtlinien dafür gibt. In gewissen Abteilungen haben sich aber gewisse Standards etabliert, bei den einen ist es SVN und bei den anderen ist es Git. In diesem Abschnitt werden die Funktionalität und der Unterschied zwischen diesen beiden Varianten erläutert.

### 8.2.2 Subversion (SVN)

Subversion arbeitet mit einem sogenannten Repository, in welchem die Änderungen gespeichert werden und einer „Working Copy“, die den aktuellen Stand enthält, in welcher der Entwickler seine Änderungen durchführt. Es werden dabei, aber nur die Unterschiede zu bereits vorhandenen Ständen übertragen. Dabei ist die Verbindung zum Repository erforderlich um Aktionen durchzuführen. Die Versionierung erfolgt dabei in Form einer Revisionszählung. Bei jeder Änderung wird die Revision um eins hochgezählt.

**8.2.3 Git**

Git hat viele Ähnlichkeiten zu Subversion, doch es unterscheidet sich gänzlich bei der Funktionsweise. Wie auch unter Subversion arbeitet man hier mit einem sogenannten Repository. Es gibt jedoch keine zentrale Stelle, in welcher alle Änderungen gespeichert werden. Jeder Benutzer besitzt eine lokale Kopie des gesamten Repositorys. Die meisten Aktionen können so lokal und ohne Netzwerkzugriff ausgeführt werden. Bei Git ist das Repository gleichzeitig auch die „Working Copy“. Die Versionierung erfolgt dabei in Form einer SHA1-Checksumme, da Git bei einer Änderung das ganze Dateisystem abspeichert.

**8.2.4 Unterschiede**

Die wichtigsten Unterschiede im Überblick:

- Git ist schneller als Subversion   
-Subversion lässt sich einfacher bedienen   
-Unter Subversion ist die Verbindung zum Repository zwingend notwendig   
-Git speichert bei einer Änderung das ganze Dateisystem ab, Subversion hingegen nur die Änderungen

## 8.3 Benutzeroberfläche

Die Applikation ist in zwei Teile aufgeteilt, in Frontend und Backend. Für beides gelten keine Richtlinien.

## 8.4 Datenbank

Das ERM dient dazu, Daten zu modellieren ohne technische Aspekte zu beachten. Das ERM stellt hier eine Tabellenstruktur dar, die für das Projekt in eine eigene Datenbank hinzugefügt wird.

## ERM einfügen!!

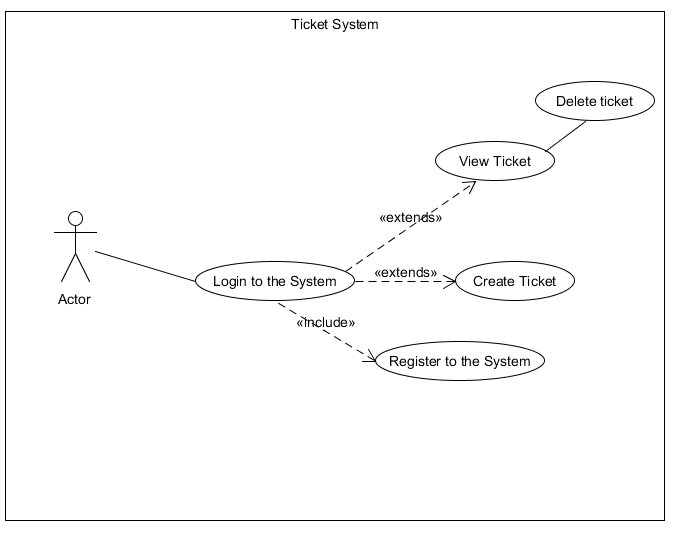
### 8.4.1 Tabellendefinitionen

### 9 Entscheidung

### 9.1 Versionsverwaltung

### 9.2 Datenbank

## 8.5 Use Case



### 8.5.1 Use Case: Login

Vorbedingung: Der User muss registriert sein

Der User kann sich anhand eines Login Fensters anmelden. Das Login Fenster besteht aus einem Feld für den Benutzernamen und einem Feld für das Passwort. Die Daten werden überprüft und nach erfolgreicher Übereinstimmung wird der User weitergeleitet.

### 8.5.2 Use Case: Register

Um auf die Applikation Zugriff zu haben, muss sich der User registrieren. Dazu gibt er seine E-Mail Adresse und sein Passwort in das Fenster eintragen. Das Passwort muss bestätigt werden.

### 8.5.3 Use Case: Create Ticket

Vorbedingung: Der User muss angemeldet sein.  
Der User kann ein neues Ticket eröffnen. Hier müssen folgende Felder ausgefüllt werden.

Titel; Abteilung; Wichtigkeit; Bereich; Beschreibung; Bilder;

Das Ticket kann gespeichert werden.

### 8.5.4 Use Case: View Ticket

Der User kann alle Tickets mit Hilfe einer Tabelle ansehen, er sieht den Titel, die Wichtigkeit und die Ticketnummer. Ausserdem hat er hier die Möglichkeit die Tickets zu bearbeiten oder zu löschen.

### 8.5.5 Use Case: Delete Ticket

Der User kann ein Ticket aus der Ticket View ansehen und löschen.

# 4

# 5 Analyse

Der erste Schritt in einem Herstellungsprozess ist die Analyse. Hier wird untersucht, was die zu erstellende Software leisten soll. Daher ist es äussert wichtig, dass die Aufgabenstellung korrekt interpretiert wird.

Zur Lösungsfindung wurden verschiedene Analyseverfahren angewandt.

## 5.1 IST-Analyse

## 5.1.1 SOLL-Analyse

# 6 Realisierung

**7 Testphasen**