Vertraulich

Probe-IPA  
Secondhand-Plattform

autor(en) : Cassandra Corrodi

dokumentennummer : CCT-FFF-XXXX

version : 1.0

status : Draft

quelle : Atos

dokumentendatum : 24 Oktober 2018­

­

­

anzahl der seiten : 1

owner : Cassandra Corrodi

© Copyright 2018, Atos AG Alle Rechte vorbehalten. Reproduktion von Teilen oder dem Gesamten ist ohne vorherige schriftliche Genehmigung des Urhebers untersagt. Für Fragen oder Anmerkungen zu diesem Dokument wenden Sie sich bitte an Atos, 0795893817.

Inhaltsverzeichnis

[1 Teil 1: Umfeld und Ablauf 4](#_Toc528828756)

[1.1 Einleitung 4](#_Toc528828757)

[1.1.1 Siemens Schweiz AG 4](#_Toc528828758)

[1.1.2 Atos AG 4](#_Toc528828759)

[1.1.3 Kandidatin 4](#_Toc528828760)

[1.2 Beteiligte Personen 5](#_Toc528828761)

[1.2.1 Durchführung der Probe IPA 5](#_Toc528828762)

[1.2.2 Kandidatin 5](#_Toc528828763)

[1.2.3 Fachvorgesetzter 5](#_Toc528828764)

[1.2.4 Ausbildungsverantwortlicher 5](#_Toc528828765)

[1.2.5 Experte 5](#_Toc528828766)

[1.3 Aufgabenstellung 6](#_Toc528828767)

[1.3.1 Projektantrag 6](#_Toc528828768)

[1.3.2 Anforderungen 7](#_Toc528828769)

[1.3.3 Anforderungen des Moduls 8](#_Toc528828770)

[1.3.4 Technologiestack 8](#_Toc528828771)

[1.4 Projektorganisation 8](#_Toc528828772)

[1.4.1 Projektmanagement Methode 9](#_Toc528828773)

[1.5 Vorkenntnisse 10](#_Toc528828774)

[1.6 Vorarbeiten 10](#_Toc528828775)

[1.7 Firmenstandards 10](#_Toc528828776)

[1.8 Zeitplan 11](#_Toc528828777)

[1.9 Arbeitsprotokoll 12](#_Toc528828778)

[2 Teil 2: Projekt-Dokumentation 17](#_Toc528828779)

[2.1 Informieren 17](#_Toc528828780)

[2.1.1 Soll-Analyse 17](#_Toc528828781)

[2.1.2 Ist-Analyse 17](#_Toc528828782)

[2.1.3 Use-Cases 18](#_Toc528828783)

[2.2 Planen 18](#_Toc528828784)

[2.3 Entscheiden 18](#_Toc528828785)

[2.4 Realisieren 18](#_Toc528828786)

[2.5 Kontrollieren 18](#_Toc528828787)

[2.6 Auswerten 18](#_Toc528828788)

[3 Teil 3: Anhang 19](#_Toc528828789)

[3.1 Quellenverzeichnis 19](#_Toc528828790)

[3.2 Abbildungsverzeichnis 19](#_Toc528828791)

[3.3 Tabellenverzeichnis 19](#_Toc528828792)

[3.4 Glossar 19](#_Toc528828793)

[3.5 Programmcode 19](#_Toc528828794)

Änderungshistorie

| Version | Datum | Beschreibung | Autor(en) |
| --- | --- | --- | --- |
| 0.1 | 30.10.2018 | Dokumentstruktur erstellt | Cassandra Corrodi |
| 0.2 |  |  | Cassandra Corrodi |
| 0.3 |  |  | Cassandra Corrodi |

# Teil 1: Umfeld und Ablauf

## Einleitung

### Siemens Schweiz AG

1847 in einer kleinen Berliner Werkstatt durch Werner von Siemens gegründet, entwickelte sich seine Firma innerhalb weniger Jahrzehnte zu einem weltumspannenden Unternehmen. Der Zeigertelegraf, die Entdeckung des dynamoelektrischen Prinzips und weitere bahnbrechende Erfindungen verhalfen Siemens zu einem schnellen Wachstum. Mit dem Bau des Kraftwerks Wynau im Kanton Bern war das Unternehmen 1894 erstmals auch in der Schweiz aktiv. In den folgenden 123 Jahren hat Siemens seine Tätigkeitsgebiete stetig weiterentwickelt. Heute ist der petrolgrüne Schriftzug im ganzen Land präsent. An mehr als 20 Standorten in der Deutschschweiz, der Romandie und im Tessin beschäftigt Siemens mehr als 5300 Mitarbeiterinnen und Mitarbeiter, darunter über 300 Lernende. Im Geschäftsjahr 2017 generierte das Unternehmen in der Schweiz einen Auftragseingang von 2,3 Milliarden und einen Umsatz von 2,2 Milliarden Franken. Damit gehört Siemens hierzulande zu den wichtigsten und grössten industriellen Arbeitgebern.

Die Verantwortung für die Verkaufs- und Vertriebsaktivitäten liegt bei der Regionalgesellschaft Siemens Schweiz. Eine sehr starke Marktpräsenz hat das Unternehmen hierzulande vor allem in den Bereichen Energie- und Gebäudetechnik, Industrieautomatisierung und Antriebstechnik sowie in der Bahn- und Strassenverkehrstechnik.1

### Atos AG

Atos ist ein weltweit führender Anbieter im Bereich der digitalen Transformation mit 120.000 Mitarbeitern in 73 Ländern und einem Jahresumsatz von 13 Milliarden Euro. Die Gruppe, die europaweit die Nummer eins in den Bereichen Cloud, Cybersicherheit und High-Performance Computing ist, bietet über ihre Digital Transformation Factory durchgängig orchestrierte hybride Cloud-, Big Data-, Business-Applikationen und Digital Workplace-Lösungen sowie Transaktionsdienste über Worldline, den europäischen Marktführer in der Zahlungsbranche.

Mit seinen Spitzentechnologien und Branchenkenntnissen unterstützt Atos die digitale Transformation seiner Kunden in allen Geschäftsbereichen.

Die Gruppe ist der weltweite Informationstechnologie-Partner für die Olympischen und Paralympischen Spiele und agiert unter den Marken Atos, Atos Syntel, Unify und Worldline. Atos ist im Pariser Aktienindex CAC40 gelistet.2

### Kandidatin

Im Sommer 2015 habe ich meine Lehre zur Informatikerin Fachrichtung Applikationsentwicklung bei Siemens Schweiz gestartet. Im ersten Lehrjahr habe ich dort die Basisausbildung absolviert. Alle Informatiklehrlinge vom gleichen Lehrgang waren während dieses Jahres zusammen, und mussten theoretische wie auch praktische Arbeiten lösen.

Im zweiten Lehrjahr wurden alle in die Abteilungen verteilt. Ab dann habe ich bei Atos gearbeitet und konnte in den Abteilungen praktische Erfahrungen im Bereich C# und Angular sammeln. Zudem durfte ich auch eine Woche im First-Level-Support arbeiten, bei dem ich auch nützliche Erfahungen mit Kunden sammeln und verschiedene Probleme lösen konnte.

Im Moment arbeite ich im SWP Team, und Implementiere mit ASP.NET MVC.

## Beteiligte Personen

### Durchführung der Probe IPA

Lehrbetrieb: Siemens Schweiz AG  
 Berufsbildung  
Adresse: Freilagerstrasse 40  
PLZ/Ort: 8047 Zürich

### Kandidatin

Name: Cassandra Corrodi  
Adresse: Alte Bergstrasse 7  
PLZ/Ort: 8707 Uetikon am See  
Telefon: 079 517 38 89

### Fachvorgesetzter

Name: Patrick Maurer  
Adresse: Freilagerstrasse 28  
PLZ/Ort: 8047 Zürich  
E-Mail: patrick.maurer@atos.net

### Ausbildungsverantwortlicher

Name: Jonas Knoll  
Adresse: Freilagerstrasse 40  
PLZ/Ort: 8047 Zürich  
Telefon: 058 558 38 58  
E-Mail: jonas.knoll@siemens.com

### Experte

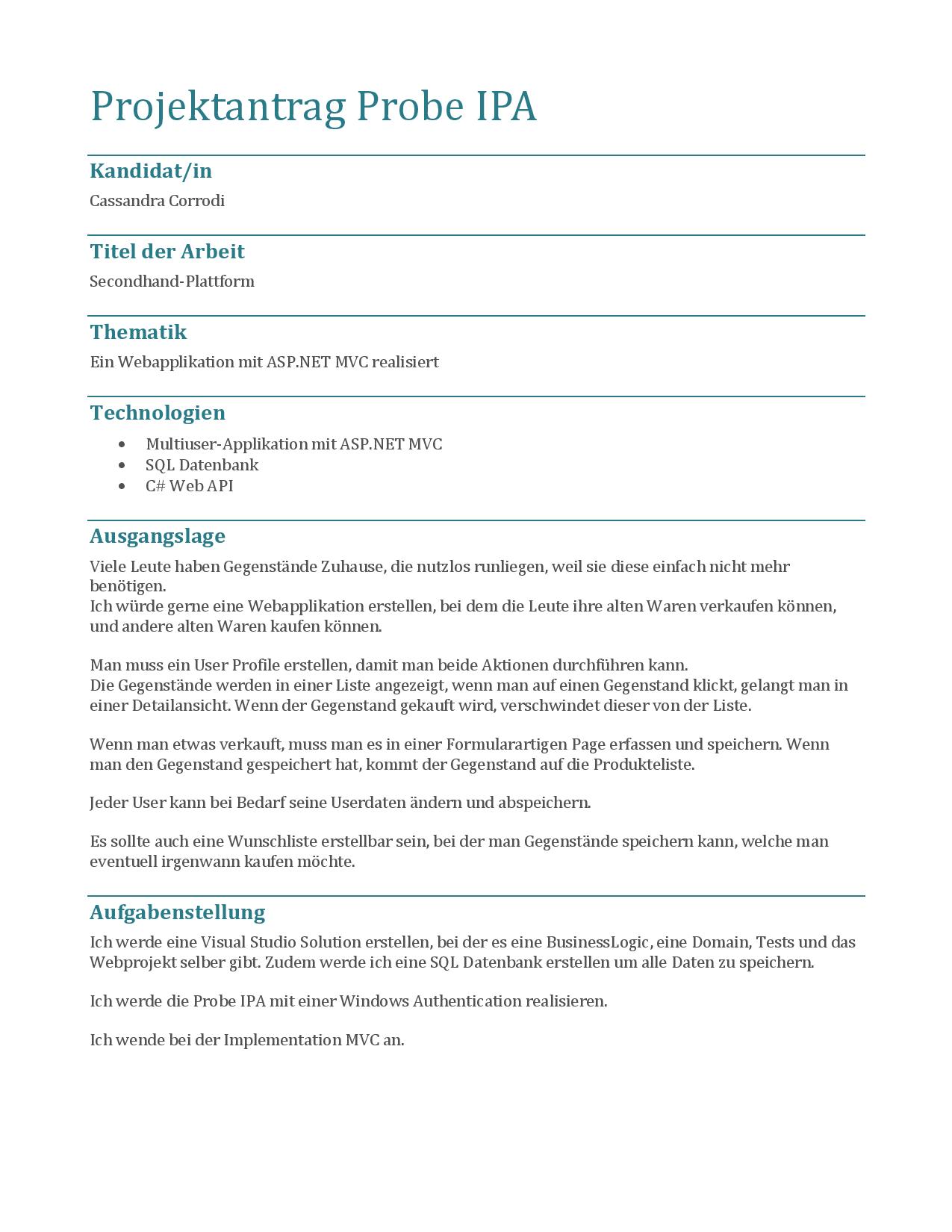
Name: Remo Steinmann  
Adresse: Freilagerstrasse 40  
PLZ/Ort: 8047 Zürich  
E-Mail: remo.steinmann@siemens.com

## Aufgabenstellung

Im eingereichten Projektbeschrieb ist beschrieben was das Programm schlussendlich können muss.

### Projektantrag

Der Projektantrag wurde schon im Voraus dem Auftraggeber zugesendet.



### Anforderungen des Moduls

* Multiuser-Applikation
* Objektorientierte Programmiersprache
* Relationale Datenbank muss erstellt werden
* Viele Clients müssen auf die Datenbank zugreifen können

### Technologiestack

IDE: Visual Studio 2017  
Datenbank: SQL Server Management Studio  
Fronend: Razor  
Backend: ASP.NET MVC  
Versionierung: GitHub und TortoiseGit

## Projektorganisation

### Projektmanagement Methode

Viele Projektmanagement Methoden sind eher darauf ausgerichtet, dass man diese im Team verwenden kann. Deswegen habe ich mich entschieden die Probe IPA mit IPERKA zu realisieren, da es sich sehr gut dafür eignet wenn man alleine arbeitet.  
Zudem kann man seine Arbeitsschritte im IPERKA genau auf jede Phase zuteilen, denn es ist ganz einfach aufgebaut.

Man muss den Auftrag verstehen und sich ein Bild des zu ereichenden Ziels machen. Dazu muss man sich ausführlich informieren.

Ziel der Planung ist es, einen Lösungsweg für das Endprodukt zu erstellen. Hier muss man die Arbeitsmittel und Arbeitsschritte einschätzen können.

Wenn man geplant hat, muss man sich entscheiden wie das Projekt genau realisiert werden muss.

Das ist der Zeitauswändigste Schritt von allen, denn die geplanten Arbeitsschritte werden einzeln ausgeführt. Wichtig ist, dass man die Planung wenn möglich einhält.

Wenn die Arbeit erledigt hat, muss man diese kontrollieren. Man muss schauen ob alles erledigt wurde, und die Qualität des Programmes gewährleistet ist.

Zum Schluss wertet man aus, wie die Arbeit gegangen ist. Welche Schritte sind mir gelungen? Welche eher nicht?

## Vorkenntnisse

Folgende Vorkenntnisse habe ich während der Lehre gesammelt:

* ASP.NET MVC (6 Monate)
* SQL (Erfahrungen aus dem Basislehrjahr)
* Visual Studio mit ReSharper (3 Jahre)
* Git & TortoiseGit (2,5 Jahre)
* Angular Typescript (1 Jahr)
* Visual Studio Code (1 Jahr)
* C# (1 Jahr)
* jQuery (3 Monate)

## Vorarbeiten

Bevor das Projekt offiziell begonnen hat, wurden schon kleine Vorarbeiten geleistet. Damit kann man den Stress während der Endspurtphase etwas geringer machen.

* Die Gliederung dieser Dokumentation wurde schon erstellt. Jede IPA ist verschieden, doch einige Punkte braucht es bei jeder Dokumentation
* IPERKA Abbildung, da dieses Thema bereits im ersten Lehrjahr bearbeitet wurde
* Git Repository wurde bereitgestellt, Link ist mit Remo Steinmann geteilt
* Inhalte welche schon im Voraus dokumentiert werden können, wurden erstellt

Die Struktur dieser Dokumentation wurde erstellt. Alle IPA Dokumentationen sind zwar verschieden, aber es hat ein paar Titel welche alle haben sollten. Über die wichtigsten Punkte wurde in einem Kurs bei Atos informiert und anschliessend umgesetzt.  
Ein Teil dieser Doku ist die Abbildung und Beschreibung vom IPERKA. Da dieses Thema bereits im ersten Lehrjahr behandelt wurde, konnte das Wissen benutzt werden um die Grafik schon im Voraus zu erstellen.

## Firmenstandards

Bei diesem Projekt stellt die Firma keine besonderen Standards. Tools und Technologie konnte selbt bestimmt werden.  
Atos hat ein Standard Template für Dokumentationen. Dieses wurde verwendet um die Dokumentation dieses Projektes zu schreiben.

## Zeitplan

### Meilensteine

Für das Projekt wurden Meilensteine festgelegt, welche die signifikanten Fortschritte eines Projektes darstellen.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### GANTT Diagramm

## Arbeitsprotokoll

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 1 31.10.2018 | |  |
| Soll | | Ist | |
| Aufgabenstellung verbessert | | Erledigt | |
| Projektbeginn | | Erledigt | |
| Vorarbeiten und Vorkenntnisse Dokumentieren | | Erledigt | |
| Firmenstandards beschreiben | | Erledigt | |
| Soll-Zeit bei Zeitplan eingefügt | | Erledigt | |
| Use-Cases erstellen | | Leider ist mit dies nicht gelungen denn ich habe zuerst das Grundgerüst mit Visual Studio erstellt. | |
| Meilensteine definieren | | Erledigt | |
| Positives (Was lief gut?) | | | |
| Ich konnte die meisten Tasks welche für den heutigen Tag geplant waren erledigen. Zudem konnte ich das Grundgerüst des Projektes mit Visual Studio bereits erstellen.  Ich habe auch noch Programme, welche für die Realisation des Projektes benötigt werden auf den Rechner installiert, damit ich vorbereitet bin für die nächsten Tage. | | | |
| Negatives (Wo gab es Probleme?) | | | |
| Heute ist es mit nicht gelungen die Use-Cases zu erstellen, denn die Zeit war knapp, und ich hatte am Feierabend noch einen Termin, also konnte ich nicht länger bleiben. Da ich kurz bevor ich gegangen bin nur noch wenig Zeit hatte, und es bestimmt nicht für die Use-Cases gereicht hätte, habe ich das Grundgerüst für die Applikation in Visual Studio erstellt, da ich gewusst habe, dass dies nicht so viel Zeit in Anspruch nimmt. | | | |
| Zeitplanung | | | |
| Ich konnte die Zeitplanung nicht ganz einhalten, denn ich habe es nicht mehr auf die Reihe gebracht, die Use-Cases zu erstellen. Sonst sind alle anderen geplanten Arbeiten und noch eine vom folgenden Tag erledigt. | | | |
| Beanspruchte Hilfestellung | | | |
| Heute habe ich keine Hilfe benötigt, da ich hauptsächlich geplant und dokumentiert habe. | | | |
| Nächste Schritte | | | |
| Für den nächsten Tag habe ich geplant, gleich als erstes mit den Use-Cases zu beginnen. Ich möchte diese so schnell wie möglich erledigt haben, damit ich mit den nächsten Schritten weitermachen kann. | | | |

Tabelle 1: Arbeitsjournal Tag 1

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 2 01.11.2018 | |  |
| Soll | | Ist | |
| Use-Cases erstellen | | Erledigt | |
| Software Architektur erstellen | | Die Software Architektur ist geplant, aber noch nicht grafisch dargestellt. | |
| ERD erstellen | | Das ERD ist ebenfalls schon geplant, aber auch noch nicht grafisch dargestellt | |
| Entscheid und Begründung für Projekt | | Erledigt | |
| ASP.NET Grundgerüst in Visual Studio erstellen | | Wurde schon am Vortag erledigt | |
| Verbesserungen am Dokument vornehmen | | Erledigt | |
| Positives (Was lief gut?) | | | |
| Ich konnte heute alle Use-Cases erstellen und diese mit Draw.io Desktop visuell darstellen. Zudem konnte ich alle Use-Cases ausführlich beschreiben.  Zudem hatte ich das erste Expertengespräch, bei dem ich viele nützliche Tipps erhalten habe, wie ich meine Dokumentation noch besser gestalten kann. Diese habe ich auch gleich umgesetzt.  Den Rest des Tages habe ich genutzt, um die restlichen Teile meines Projektes zu planen, z.B. das ERD und die Software Architektur. Ich habe mich auch am Schluss des Tages definitiv entschieden wie ich das Projekt realisieren werde. | | | |
| Negatives (Wo gab es Probleme?) | | | |
| Leider konnte ich das ERD und die Software Architektur nicht grafisch darstellen, denn die Erstellung und Darstellung der Use-Cases war sehr zeitauftreibend. Zudem hatte ich eine Stunde Besprechung mit dem Experten. Nun muss ich aufholen, damit das Projekt stressfrei laufen kann. | | | |
| Zeitplanung | | | |
| Ich konnte mich nicht an die Zeitplanung halten, denn ich habe keine Abbildungen erstellen können, weil die anderen Arbeiten sehr zeitauftreibend waren. | | | |
| Beanspruchte Hilfestellung | | | |
| Ich habe im Internet nach Use-Case Diagrammen gesucht, um auf Nummer sicher zu gehen, dass meine Use-Case Diagramme richtig aufgebaut sind. | | | |
| Nächste Schritte | | | |
| Nun werde ich mich an die Darstellung des ERDs und der Software Architektur machen, damit ich diese in die Dokumentation einfügen. Ich möchte die ersten drei Schritte des IPERKAS (Informieren, Planen und entscheiden) ganz erledigt haben. Wenn ich es schaffe widme ich mich auch schon der Entwicklung des Programmes. | | | |

Tabelle 2: Arbeitsjournal Tag 2

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 3 02.11.2018 | |  |
| Soll | | Ist | |
|  | |  | |
| Positives (Was lief gut?) | | | |
| In Sachen Dokumentation konnte ich heute viel erreichen. | | | |
| Negatives (Wo gab es Probleme?) | | | |
|  | | | |
| Zeitplanung | | | |
|  | | | |
| Beanspruchte Hilfestellung | | | |
|  | | | |
| Nächste Schritte | | | |
|  | | | |

Tabelle 3: Arbeitsjournal Tag 3

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 4 07.11.2018 | |  |
| Soll | | Ist | |
|  | |  | |
| Positives (Was lief gut?) | | | |
|  | | | |
| Negatives (Wo gab es Probleme?) | | | |
|  | | | |
| Zeitplanung | | | |
|  | | | |
| Beanspruchte Hilfestellung | | | |
|  | | | |
| Nächste Schritte | | | |
|  | | | |

Tabelle 4: Arbeitsjournal Tag 4

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tag 5 09.11.2018 | |  |
| Soll | | Ist | |
|  | |  | |
| Positives (Was lief gut?) | | | |
|  | | | |
| Negatives (Wo gab es Probleme?) | | | |
|  | | | |
| Zeitplanung | | | |
|  | | | |
| Beanspruchte Hilfestellung | | | |
|  | | | |
| Nächste Schritte | | | |
|  | | | |

Tabelle 5: Arbeitsjournal Tag 5

# Teil 2: Projekt-Dokumentation

## Informieren

### Soll-Analyse

Bei der Soll-Analyse werden die Anforderungen genau analysiert.

#### GUI

Das GUI wird in diesem Projekt mit Razor und CSS erstellt. Bei Bedarf werden einige jQuery Elemente im Frontend vorhanden sein.

Wenn die Applikation geöffnet wird, kommt man auf die Startseite eines ASP.NET Projektes. Es wurden keine Änderungen oder Vorarbeiten geleistet.

Standardmässig hat es jedoch ein gutes Grundgerüst, nämlich eine Navigation mit einer Startseite, Infoseite, Kontaktseite und eine Getting started Seite. Die getting started Seite wird bearbeitet, damit diese dann zur starseite meines Projektes wird.

Das Design des GUIs ist sehr schlicht gehalten, deswegen wird es noch einiges zum Verändern geben. Zudem sollte die Seite so Benutzerfreundlich wie möglich werden.

#### Ist-Analyse

Der Zweck der Ist-Analyse ist, herauszufinden was das vorhandene System bereits zu bieten hat.

**Hardware**

Das Entwicklungsgerät ist ein privater PC, und zwar einen Acer Aspire V 15 Nitro, mit dem Windows 10 Betriebssystem.

**Entwicklungsumgebung**

Das Projekt wird mit Visual Studio 2017 Community realisiert. In dieser Edition ist kein ReSharper vorhanden, doch sonst ist alles andere bereits konfiguriert und bereit für die Implementation des Projektes.

**SQL**

Mithilfe von Visual Studio und Microsoft MySQL Server Management Studio wird die Datenbank zum Laufen gebracht.

**Microsoft SQL Server Management Studio**

SQL Server Management Studio wurde installiert, damit man das im Visual Studio erstellte edmx ganz einfach importieren kann, und die Daten in die Datenbank einlesen kann.

**Internetbrowser**

Auf dem Gerät sind die Browser Chrome, Vivaldi, Edge und Internet Explorer installiert.

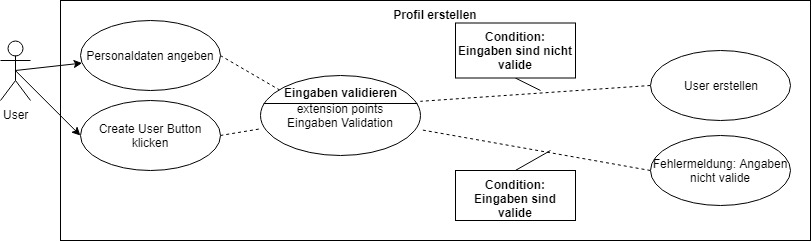
**Git**

Git ist schon seit langer Zeit auf dem Entwicklungsgerät installiert und ist bereit genutzt zu werden. Um die Arbeit mit Git zu erleichern, ist auch TortoiseGit installiert, damit Commits und Pushs schneller erledigt werden können.

### Use-Cases

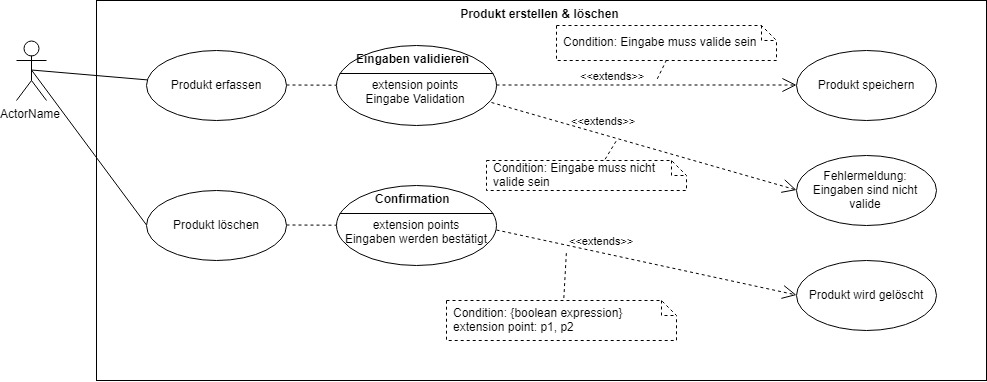
Um das Programm besser darstellen zu können, wurden Use-Cases erstellt.

**Profil erstellen**



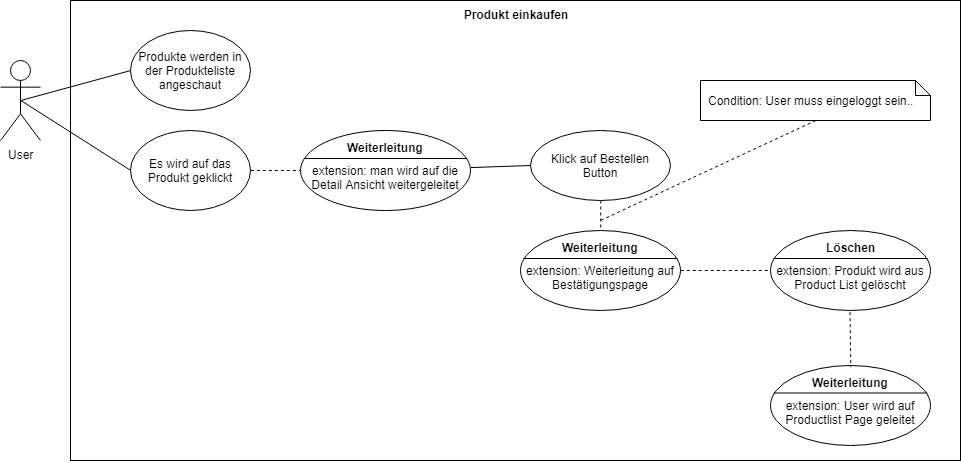
|  |  |
| --- | --- |
| Use-Case 1: Login | |
| Beschreibung | Ein neuer User muss ein Profil erstellen können |
| Personen | Neuer User |
| Eintrittsfall | Der User befindet sich auf der «User erstellen» Page. |
| Vorbedingungen | Der User ist ausgeloggt. |
| Resultat | Der User wird eingeloggt, und kann somit Aktionen ausführen, wenn er einen Gegenstand verkaufen möchte. |
| Ablauf | |  |  |  | | --- | --- | --- | | Schritt | Person | Beschreibung | | 1 | User | User öffnet die Login Page | | 2 | User | User gibt seine Personaldaten an | | 3 | System | Das System überprüft ob die eingegebenen Daten valide | | 4 | System | Erstellt den User | |

**Produkt erstellen & löschen**



|  |  |
| --- | --- |
| Use-Case 2: Produkt erstellen und löschen | |
| Beschreibung | Ein angemeldeter User muss ein Produkt erfassen und hochladen. Zudem sollte es für ihn möglich sein, dieses Produkt zu löschen, wenn er diesen doch nicht mehr verkaufen möchte. |
| Personen | Angemeldeter User |
| Eintrittsfall | Wenn sich der User auf der «Sell Product» Seite befindet. |
| Vorbedingungen | Der User ist eingeloggt |
| Resultat | Nach dem Upload können anderen User das Produkt sehen und dieses kaufen |
| Ablauf | |  |  |  | | --- | --- | --- | | Schritt | Person | Beschreibung | | 1 | User | User öffnet die «Sell Product» Page | | 2 | User | User erfasst sein Produkt | | 3 | User | User klickt auf upload | | 4 | System | Das Produkt wird in der Datenbank gespeichert und wird auf der Product List angezeigt. | | 5 | User | User klickt auf «Delete Product» Button | | 6 | System | Das Produkt wird aus der Datenbank gelöscht | | 7 | System | Das Produkt ist nicht mehr auf der Produkteliste abgebildet. | |

**Produkt einkaufen**



|  |  |
| --- | --- |
| Use-Case 3: Produkt einkaufen | |
| Beschreibung | Der User kann das Produkt in einer Detail Ansicht sehen, und kann von dort aus das Produkt kaufen. |
| Personen | User |
| Eintrittsfall | Der User klickt auf den «Buy» Button in der Detail View |
| Vorbedingungen | Der User muss eingeloggt sein |
| Resultat | Der User kehrt auf die Produkteliste zurück und der gekaufte Gegenstand ist nicht mehr verfügbar. |
| Ablauf | |  |  |  | | --- | --- | --- | | Schritt | Person | Beschreibung | | 1 | User | User klickt auf Gegenstand | | 2 | User | User klickt auf «Buy» Button | | 3 | System | Entfernt Gegenstand aus der Datenbank | | 4 | System | Leitet den User zurück auf die Produkteliste | |

## 

## Planen

### Testfälle

|  |  |
| --- | --- |
| Testfall 1: Navigation funktioniert | |
| Vorbedingungen: | Navigation mit allen nötigen Bullets sollte vorhanden sein. |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | Klick auf einen beliebigen Bullet Point | | 2 | Weiterleitung auf die korrekte Page | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Die Navigation sollte den Benutzer bei Klick auf die Bullets auf die richtige Page leiten. Testablauf sollte bei allen Bullet Points ausgeführt werden und ein erfolgreiches Resultat aufweisen. |

|  |  |
| --- | --- |
| Testfall 2: Validierung bei User erstellen | |
| Vorbedingungen: | User hat keinen Account |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | User Daten werden eingegeben | | 2 | Klick auf «Create User» Button | | 3 | Eingabe wird validiert | | 4 | Bei korrekter Validation wird der User kreiert | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Bei Erstellung eines Accounts, wird der User aufgefordert Informationen über sich einzutippen. Diese Informationen müssen korrekt validiert werden. Bei einer fehlerhaften Eingabe, kann der User nicht erstellt werden und der User wird aufgefordert seine Angaben zu überarbeiten. Wenn die Inputs korrekt ausgefüllt wurden, wird der User erstellt. |

|  |  |
| --- | --- |
| Testfall 1: Produkt hochladen, Validation | |
| Vorbedingungen: | User ist eingeloggt und hat Produkt erfasst |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | User klickt auf «Upload Item» | | 2 | Validation überprüft Eingabe | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Bei fehlerhafter Eingabe kommt eine Fehlermeldung. Bei korrekter Eingabe kann das Produkt hochgeladen werden. |

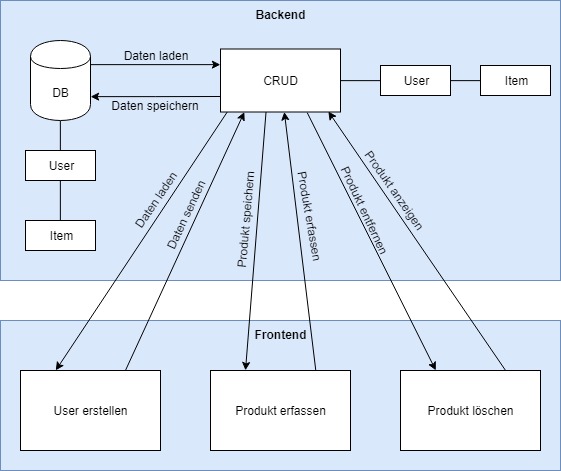
|  |  |
| --- | --- |
| Testfall 3: Produkt hochladen | |
| Vorbedingungen: | User muss eingeloggt sein |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | User befindet sich auf der «Upload Item» Page | | 2 | User erfasst Informationen für das Produkt. | | 3 | Klick auf «Upload Item» | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Item wird hochgeladen und ist auf der Produkteliste ersichtlich. |

|  |  |
| --- | --- |
| Testfall 1: Produkt verkaufen | |
| Vorbedingungen: | User muss eingeloggt sein |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | User Ist auf Detailansicht eines Produktes. | | 2 | User klickt auf «Buy» | | 3 | Handlung wird verarbeitet | | 4 | Weiterleitung auf die Produkteliste | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Wenn ein User ein Produkt kauft, wird dieses aus der Produkteliste und der Datenbank entfernt. |

|  |  |
| --- | --- |
| Testfall 1: Produkt löschen | |
| Vorbedingungen: | User muss ein Produkt hochgeladen haben |
| Testablauf: | |  |  | | --- | --- | | Schritt | Beschreibung | | 1 | Bei der Produkteliste auf «Delete Item» klicken | |
| Testmittel: | Vivaldi Browser |
| Erwartetes Resultat | Wenn der User, welcher das Produkt hochgeladen hat auf den «Delete Item» Button klickt, sollte das Produkt aus der Datenbank und somit auch von der Produkteliste gelöscht werden. |

### Software Architektur

Die Software Architektur beschreibt, wie der technische Aufbau der Applikation ist. Auf einer Grafik wurde dargestellt, wie das Projekt schlussendlich aufgebaut werden sollte.



Die Architektur ist folgendermassen aufgebaut:  
Die Applikation hat eine Datenbank (oben links) welche die Tabellen User und Item haben. Diese wird per Web.API an den Client gesendet. Die Web.API enthält alle CRUD Funktionen, damit die Daten vielseitiger genutzt werden können. CRUD steht für folgende Funktionen: Create, Read, Update und Delete. Für dieses Projekt sind zwar nicht alle notwendig, doch es werden trotzdem alle Funktionen implementiert, weil es sonst nicht CRUD wäre. Das CRUD verbindet beide, die User und die Item Tabelle mit dem Frontend.

Im Frontend sind drei verschiedene Aktionen welche die Applikation ausführen muss. Alle drei Aktionen sind mit dem CRUD verknüpft, damit die Daten mit der Datenbank ausgetauscht werden können.

**User erstellen**: Wenn ein User erstellt wird muss dieser in der Datenbank gespeichert werden.

**Produkt erfassen**: Ein User sollte ein Produkt erfassen können, welches anschliessend in die Datenbank gespeichert werden kann.

**Produkt löschen**: Wenn ein User fälschlicherweise ein Produkt hochgeladen hat, kann er diesen auch Löschen. Wenn ein Produkt gelöscht wird, sollte er auch endgültig aus der Datenbank entfernt sein.

### Datenbankdesign

Die Datenbank für dieses Projekt ist eher klein. Sie besteht aus lediglich zwei Tabellen, welche mit einer 1 zu N Verbindung verknüpft sind. Es gibt eine Tabelle für die User und eine für die Gegenstände welche auf der Seite verkauft werden.

Ein User kann mehrere Items, beziehungsweise Gegenstände haben, doch ein Gegenstand kann nur einem User gehören. Wie man auf der folgenden Abbildung sieht, ist das 1 beim User und das N beim Item.

Die Datenbank wird mithilfe von Visual Studio und Visual Studio Code realisiert.

###### Entity-Relationship-Model

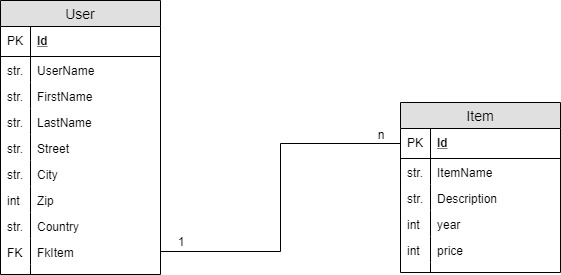


Abbildung 1: Entity-Relationship-Model

###### Tabellen

User

In dieser Tabelle sind alle User gespeichert.

|  |  |
| --- | --- |
| Attribut | Beschreibung |
| Id | Jede Tabelle braucht eine Id, damit alle Datensätze Unique sind. |
| UserName | Jeder User braucht einen UserName, damit man diesen identifizieren kann, wenn er etwas am Verkaufen ist. |
| FirstName | Braucht man um das Produkt an die richtige Person zu adressieren. |
| LastName | Braucht man um das Produkt an die richtige Person zu adressieren. |
| Street | Braucht man um das Produkt an die richtige Person zu adressieren. |
| City | Braucht man um das Produkt an die richtige Person zu adressieren. |
| Zip | Braucht man um das Produkt an die richtige Person zu adressieren. |
| Country | Braucht man um das Produkt an die richtige Person zu adressieren. |
| FkItem | Dieser Foreign Key wird benötigt, damit die Items dem richtigen User zugewiesen werden. Ein User kann unendlich viele Foreign Keys haben. |

Item

In dieser Tabelle sind alle Gegenstände gespeichert.

|  |  |
| --- | --- |
| Attribut | Beschreibung |
| Id | Jede Tabelle braucht eine Id, damit alle Datensätze Unique sind. |
| ItemName | Jedes Item braucht einen Namen, damit die Kunden wissen was es ist. |
| Description | Jedes Item braucht eine Description, damit die Kunden genau wissen was es ist. |
| Year | Die Produkte sollten gekennzeichnet sein, aus welchem Jahrgang sie sind. |
| Price | Natürlich ist es wichtig, wenn die Kunden wissen wie teuer der Gegenstand ist. |

## Entscheiden

In der Entscheidungsphase ist es das Ziel, sich für jene Sachen zu entscheiden, welche nicht vorgegeben wurden, das heisst, alles wofür der Entwickler selber verantwortlich ist.

**Entwicklungsumgebung**

**Das Projekt wird mit Visual Studio 2017 realisiert, weil ich schon viele praktische Erfahrungen damit sammeln konnte. Zudem ist es praktisch, denn man kann Client, Business Logic, Datenbank etc. hier realisieren. Diesen Vorteil kommt mir zu Gute, denn es ist zum Teil irreführend wenn man mit zu vielen Tools gleichzeitig arbeiten muss.**

**Das Entity Framework realisiere ich zwar auch mit Visual Studio, doch zur Hilfe werde ich auch Microsoft SQL Server Management Studio verwenden, um die Datenbank für die Applikation zu erstellen. Dieses Tool ist sehr praktisch, denn man kann Datenbanken in den Servern erstellen, um dann das, im Visual Studio erstellte Entity Framework einzulesen. Mit SQL Server Management Studio ist es zudem einfach Daten in die Datenbank einzufügen. Somit kann die Applikation einfacher getestet werden.**

**Versionierung**

**Damit die Sicherstellung des Projektes gewährleistet ist, habe ich mich dazu entschieden das Projekt nicht nur auf einem USB Stick zu sichern, sondern es auch auf ein Git Repository zu laden. Dieses Git Repository wird auf GitHub verfügbar sein. Der Experte wird Zugriff auf dieses Repository haben.**

**Um meine aktuellen Stände auf das Repository zu laden werde ich TortoiseGit verwenden. Ich benutze dieses Tool, denn es ist einfach damit einen Commit und einen Push zu machen. Es ist auch gut, um Änderungen der Files auf älteren Commits zu sehen. Zudem ist es einfach einen Revert zu machen, falls die Applikation während der Implementation plötzlich abstürzt.**

**Vorgehensweise**

**Als erstes wird bei der Implementation das Grundgerüst der Webapplikation erstellt. Danach wird die Datenbank implementiert.**

**Nachdem die Datenbank fertig implementiert ist, wird das Webprojekt implementiert. Wenn das Projekt fertig ist, wird alles getestet, um die Qualität des Programmes zu gewährleisten.**

**Ich habe mich für diesen Vorgang entschieden, weil ich bei Atos immer so vorgegangen bin. Ich habe mich an diese Vorgehensweise gewöhnt und es funktioniert gut für mich.**

**Testing**

**Für das Testing wird in der Visual Studio Solution ein Unit Test Projekt erstellt. Hier werden Grundfunktionen des Programmes getestet. Das sind zum Beispiel Tests, bei denen man die Kommunikation mit dem Server überprüft.**

**Doch hauptsächlich werde ich die Applikation mit Manuellen Tests testen. Diese Tests sind praktisch, denn man kann die Use-Cases zu Tests umschreiben. Man kann dann alle Schritte überprüfen, um zu sehen ob das Projekt gut implementiert wurde.**

## Realisieren

Bei der Realisation wird das ausführlich geplante Projekt umgesetzt.

## Kontrollieren

### Testprotokoll

Nach der Implementierung des Projektes, muss man die Applikation auf Herz und Nieren prüfen. Vor dem Testen wurden Testfälle verfasst. Unten sieht man die Protokolle dieser Tests und das Resultat.

|  |  |
| --- | --- |
| Testfall 1: Blabla | |
| Vorbedingungen |  |
| Test-Ablauf |  |
| Erwartetes Resultat |  |

## Auswerten

# Teil 3: Anhang

## Quellenverzeichnis

## Abbildungsverzeichnis

## Tabellenverzeichnis

## Glossar

|  |  |
| --- | --- |
| Begriff | Erklärung |
| CRUD | Create, Read, Update und Delete. Wird bei Web.APIs verwendet. |
|  |  |

## Programmcode