# Dokumentversion

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Datum | Bearbeiter | Beschreibung | |
| V0.1 | 31.03.2014 | Sakir Olgun | Allgemein: | Dokumentstruktur erstellt, PkOrg Dokumente gelesen, erster Besuchstag Expert |
| Teil 1: | Projektorganisation dokumentiert, Zeitplan erstellt, Arbeitsjournal Tag 1 |
| Teil 2: | Projektmethode definiert, Versionsverwaltung eingerichtet, Use Cases definiert. |
| V0.2 | 02.04.2014 | Sakir Olgun | Teil 1: | Arbeitsjournal Tag 2 |
| Teil 2: | Use Cases fertiggestellt, GUI Design begonnen. |
| V0.3 | 03.04.2014 | Sakir Olgun | Teil 1: | Arbeitsjournal Tag 3 |
| Teil 2: | Klassendefinitionen erstellt + UML Design, DB Design begonnen |
| V0.4 | 04.04.2014 | Sakir Olgun | Teil 1: | Arbeitsjournal Tag 4 |
| Teil 2: | DB Design beendet + erstellt, |
| V0.5 | 07.04.2014 | Sakir Olgun | Allgemein: | Zweiter Besuchstag Expert |
| Teil 1: | Arbeitsjournal Tag 5 |
| Teil 2: | Views implementiert |
| V0.6 | 09.04.2014 | Sakir Olgun | Teil 1: | Arbeitsjournal Tag 6 |
| Teil 2: | Controller erstellt, DB Anbindung |
| V0.7 | 10.04.2014 | Sakir Olgun | Teil 1: | Arbeitsjournal Tag 7 |
| Teil 2: | Modelklassen + Hibernate Mapping implementiert |
| V0.8 | 11.04.2014 | Sakir Olgun | Teil 1: | Arbeitsjournal Tag 8 |
| Teil 2: | IIS Konfiguriert und Unittests erstellt |
| V0.9 | 14.04.2014 | Sakir Olgun | Teil 1: | Arbeitsjournal Tag 9 |
| Teil 2: | UseCases getestet + Implementationsfehler korrigiert + Kommentare hinzugefügt |
| V1.0 | 16.04.2014 | Sakir Olgun | Allgemein: | - |
| Teil 1: | Arbeitsjournal Tag 10 |
| Teil 2: | Dokumentation fertiggestellt und korrigiert |

Inhaltsverzeichnis

[1 Ziel und Zweck 3](#_Toc384633150)

[Teil 1: Umfeld und Ablauf 4](#_Toc384633151)

[2 Projektorganisation 4](#_Toc384633152)

[2.1 Durchführungstage 4](#_Toc384633153)

[2.2 Thematik 5](#_Toc384633154)

[2.3 Klassierung 5](#_Toc384633155)

[2.4 Titel der Facharbeit 5](#_Toc384633156)

[2.5 Ausgangslage 5](#_Toc384633157)

[2.6 Detaillierte Aufgabenstellung 6](#_Toc384633158)

[2.6.1 Planungsarbeiten: 6](#_Toc384633159)

[2.6.2 Realisierung: 6](#_Toc384633160)

[2.6.3 Testing: 7](#_Toc384633161)

[2.6.4 Dokumentation: 7](#_Toc384633162)

[2.7 Mittel und Methoden 7](#_Toc384633163)

[2.7.1 Hardware 7](#_Toc384633164)

[2.7.2 Software 7](#_Toc384633165)

[2.8 Vorarbeiten 7](#_Toc384633166)

[2.9 Firmen Standards 7](#_Toc384633167)

[2.10 Vorkenntnisse 7](#_Toc384633168)

[2.11 Neue Lerninhalte 7](#_Toc384633169)

[2.12 Arbeiten in den letzten 6 Monaten 7](#_Toc384633170)

[3 Zeitplan 8](#_Toc384633171)

[4 Arbeitsjournal 9](#_Toc384633172)

[4.1 Tag 1 – Montag 31.03.2014 9](#_Toc384633173)

[4.2 Tag 2 – Mittwoch 02.04.2014 10](#_Toc384633174)

[4.3 Tag 3 – Donnerstag 03.04.2014 11](#_Toc384633175)

[4.4 Tag 4 – Freitag 04.04.2014 12](#_Toc384633176)

[4.5 Tag 5 – Montag 07.04.2014 13](#_Toc384633177)

[4.6 Tag 6 – Mittwoch 09.04.2014 14](#_Toc384633178)

[4.7 Tag 7 – Donnerstag 10.04.2014 15](#_Toc384633179)

[4.8 Tag 8 – Freitag 11.04.2014 16](#_Toc384633180)

[4.9 Tag 9 – Montag 14.04.2014 17](#_Toc384633181)

[4.10 Tag 10 – Mittwoch 16.04.2014 18](#_Toc384633182)

[Teil 2: Vorgehen und Umsetzung 19](#_Toc384633183)

[5 Projektmethode 19](#_Toc384633184)

[5.1 SCRUM (Agile Softwareentwicklung) 19](#_Toc384633185)

[5.2 IPERKA 20](#_Toc384633186)

[5.2.1 Erklärung 20](#_Toc384633187)

[5.2.2 Meine Facharbeit 20](#_Toc384633188)

[6 Management Summary 21](#_Toc384633189)

[6.1 Ausgangssituation 21](#_Toc384633190)

[6.2 Umsetzung 21](#_Toc384633191)

[6.3 Ergebnis 21](#_Toc384633192)

[7 Recherchen für die Implementation 22](#_Toc384633193)

[7.1 Versionisierungstools 22](#_Toc384633194)

[7.1.1 Git: Kurze Beschreibung 22](#_Toc384633195)

[7.1.1.1 Neues Repository 22](#_Toc384633196)

[7.1.1.2 Repository auschecken 22](#_Toc384633197)

[7.1.1.3 Workflow 22](#_Toc384633198)

[7.1.1.4 Add & Commit 23](#_Toc384633199)

[7.1.1.5 Änderungen hochladen (Check-In) 23](#_Toc384633200)

[7.1.1.6 Branching 23](#_Toc384633201)

[7.1.1.7 Update & Merge 24](#_Toc384633202)

[7.1.1.8 Tagging 24](#_Toc384633203)

[7.1.1.9 Änderungen rückgängig machen 24](#_Toc384633204)

[7.1.2 Git Funktionalität (Visuelle Darstellung) 25](#_Toc384633205)

[7.2 UseCases 26](#_Toc384633206)

[7.2.1 Definition 27](#_Toc384633207)

[7.3 GUI Design 32](#_Toc384633208)

[7.3.1 Softwareergonomie 32](#_Toc384633209)

[7.3.2 Prototypen 32](#_Toc384633210)

[7.4 Klassendefinition 33](#_Toc384633211)

[7.4.1 UML Klassendiagramm 33](#_Toc384633212)

[7.4.1.1 Lieferant 33](#_Toc384633213)

[7.4.1.2 Kundennetzwerk 33](#_Toc384633214)

[7.4.1.3 User(Mitarbeiter) 34](#_Toc384633215)

[7.4.1.4 Controller 34](#_Toc384633216)

[7.4.1.5 Sonstige Klassen 35](#_Toc384633217)

[7.5 MS SQL 36](#_Toc384633218)

[7.5.1 Tabellendefinition 36](#_Toc384633219)

[7.5.2 ERD 36](#_Toc384633220)

[7.6 Implementationsmethodik 37](#_Toc384633221)

[7.6.1 ASP.NET mit VB.NET 37](#_Toc384633222)

[7.6.2 Model-View-Controller 37](#_Toc384633223)

[8 Implementation 38](#_Toc384633224)

[9 Testprotokoll 39](#_Toc384633225)

[10 Reflexion 40](#_Toc384633226)

[11 Quellverzeichnis 41](#_Toc384633227)

[12 Glossar 42](#_Toc384633228)

[13 Anhang 43](#_Toc384633229)

# Ziel und Zweck

Zum Abschluss der 4 Jährigen Ausbildung wird nun mit dieser Abschlussarbeit gezeigt was gelernt wurde. In diesem Dokument geht es um ein Projekt von der Planung bis hin zur Realisierung und kurz vor der Einführung in der Firma.

Im ersten Teil sind organisatorische Informationen dokumentiert. Der Zeitplan mit SOLL und IST Vergleich und die Arbeitsjournale für die 10 Tage sind auch in diesem Kapitel beschrieben.

Im zweiten Teil geht um die eigentliche Arbeit. Die Beschreibung des Projektes sowie der Aufbau, die Planung, Realisierung und das Testen ist der Hauptbestandteil des zweiten Teiles.

**Teil 1:** Umfeld und Ablauf

# Projektorganisation

|  |  |
| --- | --- |
| Lernender: | Lehrbetrieb: |
| Herr Olgun Sakir  St. Gallerstrasse 54  8853 Lachen SZ  G: 079 700 13 76  N: 079 700 13 76  Mail: sakirolgun@gmx.ch | **Atrix Consult GmbH**  Hardturmstrasse 169  8005-Zürich |

|  |  |
| --- | --- |
| Fachvorgesetzter / Berufsbildner: | Validierungsverantwortlicher: |
| Herr Jeanpierre Rosano  Welbrigring 25  8954 Geroldswil  G: 044 445 70 40  Mail: rosano@atrix.ch | **Herr Bonfranchi Renzo**  P: 043 266 61 87  N: 079 652 46 76 |

|  |  |
| --- | --- |
| Hauptexperte: | Zweit- Experte |
| Herr Ursprung Sascha Vogtackerweg 19 B  4148 Pfeffingen  G: 079 889 15 46  N: 079 889 15 46 | **Herr Gerschwiler Frank**  AXA Technology Services Schweiz AG  PL1.118  Paulstrasse 12  8401 Winterthur  G: 052 261 31 41  N: 076 377 09 73 |

## Durchführungstage

31.03.2014 – 16.04.2014

|  |  |  |
| --- | --- | --- |
| Tage: | Datum: | Bemerkung: |
| 1.Tag | 31.03.2014 | Beginn / 1. Expertenbesuch |
| 2.Tag | 01.04.2014 |  |
| 3.Tag | 03.04.2014 |  |
| 4.Tag | 04.04.2014 |  |
| 5.Tag | 07.04.2014 | 2. Expertenbesuch |
| 6.Tag | 09.04.2014 |  |
| 7.Tag | 10.04.2014 |  |
| 8.Tag | 11.04.2014 |  |
| 9.Tag | 14.04.2014 |  |
| 10 Tag | 16.04.2014 | Abgabe 18 Uhr |

## Thematik

Entwicklung eines Systems, um die Kundenanfragen oder Arbeiten strukturiert zu verwalten.

## Klassierung

* Applikationsentwicklung prozedural
* MS Windows
* Visual Basic / VBA / VB.net

## Titel der Facharbeit

Intranet mit SQL-Anbindung

## Ausgangslage

Das bestehende, eigene Intranet ist unflexibel und basiert auf statische HTML-Dateien.

Um betriebliche Informationen, welche für die Administration und technischer Kundenbetreuung benötigt werden, flexibler verwalten zu können, wird ein neues Programm programmiert. Dieses soll als Intranet-Applikation genutzt werden können. Aufgrund der SQL-Anbindung wird ein automatischer Datenaustausch mit weiteren SQL-Datenbanken bzw. weiteren internen Applikationen erfolgen können.

## Detaillierte Aufgabenstellung

Basis für die nachfolgenden Arbeiten bilden die Vorgaben ‚Anforderungen\_Intranet\_Atrix“ (Anforderungskatalog) und die „Programmierrichtlinien\_Atrix“. Der Anforderungskatalog beinhaltet die Anwendungsanalyse (Use Case, Feldbeschreibungen und Testvorgehen).

Nachstehend ergänzende Punkte zum Anforderungskatalog.

### Planungsarbeiten:

• Erstellen und nachführen eines Projektplanes mit MS Project. Die 10 Tage IPA sind tagesgenau verplant.

• Erstellen einer Word-Dokumentenvorlage mit der Dokumentenstruktur für die IPA-Dokumentation. Die Dokumentenvorlage muss ein Inhaltsverzeichnis haben, welches sich aufgrund der Gliederungsstufen 1 – 4 erstellen bzw. aktualisieren lässt. Zudem soll ein Abbildverzeichnis erstellt bzw. aktualisiert werden können.

Design/Analyse:

• Aufgrund der Anforderungen (Vorgabe) ein Entity Relation Diagram (ERD) erstellen, dass im Minimum das Kapitel ‚Funktionalität‘ aus dem Anforderungskatalog abbildet.

• Erstellen einer GUI-Vorlage. Die wesentlichen Elemente der Applikation (Navigations-, Such-, Übersichts- und Bearbeitungselemente müssen definiert und positioniert werden. Unabhängig der darzustellenden Daten werden immer die gleichen Elemente in der Applikation verwendet.

### Realisierung:

• Aufgrund des erstellten ERDs wird eine neue Datenbank in Microsoft SQL erstellt. Nach Möglichkeit soll die Tabellenstruktur bis zur 3. Normalisierungsstufe erstellt werden.

• Sämtliche Windows-Domänenbenutzer haben Lese-, Lösch-, Ausführungs- und Schreibrechte für die neue Datenbank.

• Programmierung sämtlicher Anforderungen und Klassendefinition aus der Vorarbeit. Programmiert wird mit Visual Basic mit der Programmierumgebung Visual Studio 2012.

• Die Architektur basiert auf einem 3-Schichten-Modell: Datenschicht (Datenhaltung), Programmschicht (Präsentation) und Logikschicht (Businesslogikschicht) beinhalten.

• Installation und Konfiguration der Software auf dem bestehenden, internen IIS-Server. Die Applikation wird ausschliesslich im internen Firmennetz und von allen Anwendern angewendet. Eine Verschlüsselung der Übertragung oder ein separates Anmeldeprozedere ist nicht umzusetzen.

• Die Verschlüsselung und Entschlüsselung muss als Stored Procedure (SP in T-SQL) oder als Tabellen-Trigger umgesetzt werden; um ein Passwort verschlüsselt in einer Tabelle bzw. Feld abzulegen, wird eine SP oder ein Trigger angewendet. Muss das Passwort als Klartext ausgewiesen werden, wird dies mit einer eigene SP umgesetzt.

### Testing:

• Beschreibung im Anforderungskatalog

### Dokumentation:

• Die der IPA-Dokumentation soll u.a. die nachfolgenden Punkte enthalten:

- das ERD

- alle Use Cases

- Klassendefinition (Klassendiagramm)

- Beschreibung der Schnittstellen

- Programmverweise mit Versionsangabe

- Ablagestruktur des Sourcecode (im Filesystem)

- Installationsanleitung IIS

- Allfällige Zugangsdaten für den Betrieb und Installation

## Mittel und Methoden

### Hardware

* Entwicklungs-PC

### Software

* Visual Studio 2012, Programmiersprache Visual Basic
* SQL Server 2012 mit einer gemeinsamen Entwicklungsinstanz
* MS Office 2010/2013 (Project, Word, Visio, etc.)

## Vorarbeiten

* Anforderungen für die Applikation (Intranet)
* Programmierrichtlinien

## Firmen Standards

|  |  |
| --- | --- |
| Komponente: | Beschreibung |
| Programmierrichtlinien | Implementationsregeln |

## Vorkenntnisse

* VB.NET: Mehrere Projekte Implementiert.
* ASP.NET: Im vorherigem Betrieb in einigen Projekten Erweiterungen gemacht. (Grundkenntnisse)
* SQL: Schulkenntnisse(Theorie), sowie Kenntnisse vom Selbststudium(Theorie)
* IIS Konfiguration: Neu, bisher keine vergleichbare Arbeit in der Praxis gemacht.
* Visual Studio 12: Neue Version der bisherigen benutzten Umgebung.

## Neue Lerninhalte

* Programmierung von Stored Procedure in T-SQL

## Arbeiten in den letzten 6 Monaten

Ich bin seit Ende Januar 2014 in der Firma Atrix tätig. Ich habe bisher ausschliesslich in Projekten mit VB programmiert.

# Zeitplan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Abgeschl. | Vorgangsname | Anfang | Ende | Arbeit | Ist-Arbeit |
|  | **IPA 2014 - Beginn** | **Mon 31.03.14** | **Mit 16.04.14** | **84 Std.** | **20 Std.** |
|  | **Vorbereitungsphase** | **Mon 31.03.14** | **Fre 04.04.14** | **26 Std.** | **20 Std.** |
| ✓ | Dokumentstruktur + Zeitplanung erstellen | Mon 31.03.14 | Mon 31.03.14 | 4 Std. | 4 Std. |
| ✓ | Versionierungstool einrichten | Mon 31.03.14 | Mon 31.03.14 | 2 Std. | 2 Std. |
| ✓ | Use Cases definieren | Mon 31.03.14 | Mit 02.04.14 | 4 Std. | 4 Std. |
| ✓ | GUI Design | Mit 02.04.14 | Mit 02.04.14 | 6 Std. | 3 Std. |
| ✓ | Klassendefinition (mit Use Cases) | Don 03.04.14 | Don 03.04.14 | 2 Std. | 2 Std. |
| ✓ | UML Design | Don 03.04.14 | Don 03.04.14 | 2 Std. | 2 Std. |
| ✓ | DB Design + Erstellen | Don 03.04.14 | Fre 04.04.14 | 6 Std. | 3 Std. |
|  | **Implementationphase** | **Fre 04.04.14** | **Fre 11.04.14** | **36 Std.** | **0 Std.** |
| ✓ | View implementieren (Frontend) | Fre 04.04.14 | Mon 07.04.14 | 8 Std. | 0 Std. |
|  | Controller implementieren (Backend) | Mon 07.04.14 | Mon 07.04.14 | 8 Std. | 0 Std. |
|  | DB mit Pseudodaten anbinden | Mon 07.04.14 | Mit 09.04.14 | 4 Std. | 0 Std. |
|  | Model implementieren (DB Abfragen) | Mit 09.04.14 | Don 10.04.14 | 8 Std. | 0 Std. |
|  | Unittests für die View | Don 10.04.14 | Don 10.04.14 | 2 Std. | 0 Std. |
|  | Unittests für den Controller | Don 10.04.14 | Don 10.04.14 | 2 Std. | 0 Std. |
|  | Unittests für das Model | Don 10.04.14 | Don 10.04.14 | 2 Std. | 0 Std. |
|  | IIS konfigurieren und einrichten | Don 10.04.14 | Fre 11.04.14 | 2 Std. | 0 Std. |
|  | **Testing + Korrekturen** | **Fre 11.04.14** | **Mit 16.04.14** | **22 Std.** | **0 Std.** |
|  | Use Cases testen | Fre 11.04.14 | Fre 11.04.14 | 6 Std. | 0 Std. |
|  | Implementationsfehler korrigieren | Fre 11.04.14 | Mon 14.04.14 | 10 Std. | 0 Std. |
|  | Dokumentation korrigieren | Mon 14.04.14 | Mit 16.04.14 | 6 Std. | 0 Std. |
|  | Dokumentation | Mon 31.03.14 | Mit 16.04.14 | 85 Std. | 0 Std. |

# Arbeitsjournal

## Tag 1 – Montag 31.03.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | Aufgabenstellung lesen |
| ✓ | Zeitplan erstellen |
| ✓ | Dokumentstruktur erstellen |
| ✓ | Versionisierungstool (GitHub) einrichten |
| ✓ | Entwicklungsumgebung einrichten |
| ✓ | Beginn UseCase Definition |

|  |
| --- |
| Ausgeführte Arbeiten |
| Ich habe die Aufgabenstellungen gelesen und mich über den Ablauf der IPA informiert. Anschliessend habe ich meine Zeitplanung für die IPA, sowie auch die Dokumentstruktur erstellt. Ich habe GitHub für die Implementationsphase eingerichtet.  Nach einrichten von GitHub habe ich noch meine Entwicklungsumgebung für die Realisierung vorbereitet (IDE, Tools & Plugins etc.)  Zum Schluss habe ich mit den UseCases begonnen und mich über die UML UseCases informiert. |
| Aufgetretene Probleme |
| Ich hatte ein Problem an meinem Arbeitsplatz. Ich wollte einen dritten Bildschirm anschliessen, aber leider, auf Grund von Treiberfehlern funktionierte es vorerst nicht. Nach einigen Recherchen konnte ich die nötigen Einstellungen machen und die nötigen Treiber installieren. Leider verlor ich hier 1-2 Stunden. |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
|  |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| keine |

|  |
| --- |
| Vergleich mit Zeitplan |
| Ich liege, trotz des kleinen Unterbruchs wegen den fehlenden Treibern, im Zeitplan. |

|  |
| --- |
| Reflexion |
| Ich habe gut mit der IPA begonnen konnte auch mein Soll Ziel für Heute erreichen, dennoch aber ist eine grosse Nervosität in mir. Das Problem mit den Bildschirmen war leider etwas mühsam, aber dafür funktioniert es jetzt. Heute war auch der Expert zu Besuch und hat mir allgemeine Informationen zu diesen 10 Tagen gegeben. |

## Tag 2 – Mittwoch 02.04.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | UseCases beenden. |
| ✓ | GUI Design |

|  |
| --- |
| Ausgeführte Arbeiten |
| Heute habe ich die UML UseCases studiert und entsprechend in meinem IPA Bericht dokumentiert Die Theorie (Regeln, Aufbau, Beispiele etc.) und die UseCases für meine Arbeit sind nun fertig dokumentiert. Ich habe Diagramme sowie die Definitionen erstellt und eingebunden. Anschliessend habe ich mit dem GUI Design begonnen. Ich habe mich auch mit der Softwareergonomie aus meiner Aufgabenstellung beschäftigt. Und mithilfe der gewonnenen Information habe ich nun GUI Prototypen erstellt. Diese dienen dann als Vorlage bei der Implementierung des GUIs. |

|  |
| --- |
| Aufgetretene Probleme |
| - |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
| - |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| - |

|  |
| --- |
| Vergleich mit Zeitplan |
| Gemäss Zeitplan |

|  |
| --- |
| Reflexion |
| Der 2. Tag meiner IPA lief auch ohne Probleme. Ich habe mich mit den UseCases auseinandergesetzt und habe festgestellt, dass UseCases von sehr grossem Nutzen sein können bei der Implementation von neuer Software. Beim GUI Design habe ich verschiedene Frameworks verglichen und anschliessend für einen entscheiden und das Gerüst für das Frontend erstellt. |

## Tag 3 – Donnerstag 03.04.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | Klassendefinition |
| ✓ | UML Klassendiagramm |
| ~ | DB Design |

|  |
| --- |
| Ausgeführte Arbeiten |
| Heute habe ich die Klassen definiert und ein UML Klassendiagramm erstellt. Auch mit dem DB Design konnte ich anfangen, aber leider bin ich nicht planmässig vorangekommen. Ich habe einige Zeit in die Bewertungskriterien investiert und anschliessend meine Dokumentation überarbeitet. |

|  |
| --- |
| Aufgetretene Probleme |
| Ich bin nochmals die Kriterien durchgegangen und musste anschliessend die fehlenden Informationen noch in meinem Dokument ergänzen. Dies beanspruchte ein wenig Zeit und deshalb konnte ich nicht wie im Zeitplan geplant meine Arbeit durchführen. |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
| - |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| - |

|  |
| --- |
| Vergleich mit Zeitplan |
| Ich bin etwas im Verzug mit dem Zeitplan aber dies werde ich die nächsten Arbeitstage wieder aufholen. |

|  |
| --- |
| Reflexion |
| Die Klassen zu definieren und dazu das UML Klassendiagramm zu zeichnen funktionierte ohne Probleme. Sie sind auch in meiner Doku ersichtlich. Beim DB Design bin ich leider etwas im Verzug, da ich nochmals meinen IPA Bericht korrigieren musste. |

## Tag 4 – Freitag 04.04.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | DB Design |
| ✓ | View Implementieren |

|  |
| --- |
| Ausgeführte Arbeiten |
| Ich habe heute das DB Design erstellt und anschliessend mit der Implementation der View begonnen. Beim DB Design hatte ich etwas mühe, weil MSSQL für mich etwas neu war.  Das DB Design besteht und stimmt mit den Anforderungen überein. Auch das ERD ist vorhanden und im Bericht ersichtlich. |

|  |
| --- |
| Aufgetretene Probleme |
| MSSQL Bedienung war für mich am Anfang etwas mühsam, aber innerhalb von wenigen Stunden fand ich mich zurecht und konnte auch anschliessend mit dem DB Design beginnen.  Bei der Verknüpfung von Beziehungen zwischen Tabellen hatte ich mühe mit der FK Definierung. |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
| - |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| - |

|  |
| --- |
| Vergleich mit Zeitplan |
| Im Verzug mit den Views aber konnte den bereits vorhandenen Verzug etwas einholen. |

|  |
| --- |
| Reflexion |
| Heute konnte ich das DB-Design beenden und anschliessend mit der Implementation der View anfangen. Beim DB Design hatte ich vorerst Mühe mit der Umgebung, aber nach einigen Recherchen konnte ich dieses Hindernis überwinden. |

## Tag 5 – Montag 07.04.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | View fertig implementieren |
| ✓ | Controller implementieren |
| ✓ | Dokumentation korrigieren |

|  |
| --- |
| Ausgeführte Arbeiten |
| Ich habe heute mit der Dokumentation angefangen, da noch einige Korrekturen von letzter Woche allfällig waren. Anschliessend habe ich das Frontend fertiggestellt. Zuletzt habe ich mit dem Controller begonnen. |

|  |
| --- |
| Aufgetretene Probleme |
| Ich habe einige Ergänzungen in meinem IPA Bericht gemacht und somit unter Zeitdruck gekommen |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
| - |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| - |

|  |
| --- |
| Vergleich mit Zeitplan |
| Ich konnte den Verzug einholen, da ich besser als geplant mit dem Controller vorankam. |

|  |
| --- |
| Reflexion |
| Heute habe ich die verschiedenen Views fertig implementiert. Die Views für alle CRUD Operationen sind nun vorhanden. Anschliessend habe ich mit den Controllern und deren Helfer Klassen begonnen. Die Hilfsklassen für die Controller beinhalten die Datenbeschaffung |

## Tag 6 – Mittwoch 09.04.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | Controller fertig implementieren |
| ✓ | DB Anbindung mit Pseudodaten |
| ✓ | Model Implementiert |

|  |
| --- |
| Ausgeführte Arbeiten |
| Die Controller habe ich fertig gestellt und die Zusammenarbeit mit den Views getestet.  Ich habe die DB mit Pseudodaten gefüllt und anschliessend mit der Anbindung begonnen. Für das OR Mapping habe ich das NHibernate Framework benutzt. Ich musste zuerst im Web.config die Konfiguration vornehmen und anschliessend habe ich die XML Files für das Mapping sowie auch die dazugehörigen Klassen erstellt. |

|  |
| --- |
| Aufgetretene Probleme |
| - |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
| Controllerhandling getestet und kontrolliert ob auch die richtigen Views angesprochen werden. |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| - |

|  |
| --- |
| Vergleich mit Zeitplan |
| Ich bin wieder nach dem Zeitplan auf Kurs. |

|  |
| --- |
| Reflexion |
| Ich habe heute die Anbindung der DB fertiggestellt. Am Anfang hatte ich einige Schwierigkeiten mit der Anbindung, da ich die XML Files für das Mapping falsch konfiguriert hatte. Ich hatte vergessen die Build Vorgänge der XML Files zu „Embedded Resource“ zu ändern. |

## Tag 7 – Donnerstag 10.04.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | Unittests erstellt |
| ✓ | IIS auf dem Server konfiguriert für das Testing. |

|  |
| --- |
| Ausgeführte Arbeiten |
| Ich habe die Unittests erstellt und die CRUD Operationen und die Datenanbindung getestet. Anschliessend habe ich den IIS Server auf einem von unseren Virtuellen Server installiert und konfiguriert. Auch die DB habe auf dem SQL Server erstellt (Vorher nur Lokal). |

|  |
| --- |
| Aufgetretene Probleme |
| - |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
| Die Unittests für die CRUD Operationen waren erfolgreich |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| - |

|  |
| --- |
| Vergleich mit Zeitplan |
| Gemäss Zeitplan am Projekt |

|  |
| --- |
| Reflexion |
| Heute habe ich die Unittests erstellt und anschliessend auch die Tests ausgeführt mit Erfolg. Anschliessend habe ich auf einem Virtuellen Server den IIS Konfiguriert.  Nun bin ich wieder gemäss Zeitplan am Projekt dran und kann etwas lockerer alles angehen. |

## Tag 8 – Freitag 11.04.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | Blackbox Testing |
| ✓ | Implementationsfehler korrigieren |

|  |
| --- |
| Ausgeführte Arbeiten |
| Ich habe mein Projekt durch getestet basierend auf den UseCases, welche ich am Anfang definiert hatte. Anschliessend habe ich noch die entdeckten Implementationsfehler korrigiert und nochmals getestet. |

|  |
| --- |
| Aufgetretene Probleme |
| Keine Probleme |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
| Die Funktionalität des Projektes getestet, indem ich die CRUD Operationen vom Browser aus ausgeführt habe. |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| - |

|  |
| --- |
| Vergleich mit Zeitplan |
| Gemäss Zeitplan |

|  |
| --- |
| Reflexion |
| Heute habe ich noch den letzten Schliff meinem Projekt verpasst. Das Intranet ist nun funktionsfähig und brauchbar. Ich hatte zwar die letzten 2 Wochen einige Probleme aber konnte zum Glück schnell damit klarkommen und diese auch wieder lösen. Nun Bleibt nur noch die Überarbeitung des Berichts. |

## Tag 9 – Montag 14.04.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | Source Code kommentiert |
| ✓ | Interne Dokumentation erstellt |
| ✓ | IPA Bericht überarbeitet. |

|  |
| --- |
| Ausgeführte Arbeiten |
| Heute habe ich meinen Programmcode kommentiert um es verständlicher zu machen. Ich habe auch eine Interne Dokumentation erstellt, mit den für uns relevanten Infos. Anschliessend habe ich meine Dokumentation weiter überarbeitet. |

|  |
| --- |
| Aufgetretene Probleme |
| Keine Probleme |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
| - |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| - |

|  |
| --- |
| Vergleich mit Zeitplan |
| Gemäss Zeitplan |

|  |
| --- |
| Reflexion |
| Ich habe meinen Code noch etwas kommentiert und verständlicher gemacht, dies wird anschliessend für den nächsten Programmierer eine Erleichterung sein hoffe ich. Nun wäre ich mit meinem Projekt soweit fertig. Die nächste Zeit werde ich noch für die Dokumentation beanspruchen. In grossem und ganzem bin ich mit dem Endprodukt zufrieden und hoffe, dass ich diesen auch im Betrieb einführen kann. |

## Tag 10 – Mittwoch 16.04.2014

|  |  |  |
| --- | --- | --- |
| ☺ | 😐 | ☹ |
|  |  |  |

**Stimmungsbarometer :**

|  |  |
| --- | --- |
| Tagesziele | |
| ✓ | Dokumentation fertigstellen |

|  |
| --- |
| Ausgeführte Arbeiten |
| Ich habe heute den ganzen Tag in die Dokumentation investiert. |

|  |
| --- |
| Aufgetretene Probleme |
| - |

|  |
| --- |
| Erfolgreiche oder erfolglose Tests |
| - |

|  |
| --- |
| Beanspruchte Hilfestellungen |
| - |

|  |
| --- |
| Vergleich mit Zeitplan |
| Gemäss Zeitplan fertiggeworden |

|  |
| --- |
| Reflexion |
| Ich habe heute mehrheitlich nur den IPA Bericht weitergeführt und anschliessend auf PkOrg geladen. Die letzten 2 Wochen waren recht anstrengend, da ich auch eine grosse Nervosität in mir hatte und je weniger Zeit ich hatte desto mehr Flüchtigkeitsfehler kamen unter. Aber nun kann ich etwas aufatmen und mich für die Präsentation vorbereiten. |

**Teil 2:** Vorgehen und Umsetzung

# Projektmethode

Die Projektmethode beinhaltet den Ablauf eines Projekts von der Informationssammlung - Planung bis zur Umsetzung - Auswertung. Ich habe mich, in meiner 4 jährigen Ausbildung, mehrheitlich nur mit IPERKA auseinandergesetzt und meine Projekte mit dieser Methode durchgeführt. In meinem ehemaligen Lehrbetrieb durfte ich aber auch nach der Agilen Softwareentwicklung (SCRUM) ein kleines Projekt durchführen.

## SCRUM (Agile Softwareentwicklung)

Die Projektmethode SCRUM ist, meiner Meinung nach, eine sehr effiziente Projektmethode. Mit SCRUM wird das Projekt in Sprints (vergleichbar mit Meilenstein) aufgeteilt. Diese beinhalten Teilaufgaben, welche maximal 1 Tag dauern dürfen. Auch die Kommunikation zwischen den Entwicklern und dem Kunden wird hier abgeschafft. Der Kunde kommuniziert nur mit dem sogenannten „Product Owner“. Die Aufgabenverteilung sowie auch die Aufwandsschätzung werden vom Entwicklerteam gemacht. Eine weitere wichtige Person ist der „Scrum Master“. Dieser beobachtet während des ganzen Projektes, ob wirklich auch nach SCRUM entwickelt wird, aber auch dass es keine Kommunikationsprobleme sowie persönliche Konflikte zwischen den einzelnen Teammitgliedern herrscht, ist eine Aufgabe vom „Scrum Master“. Jeden Tag gibt es ein kurzes Meeting (Daily Scrum). Hier wird kurz den Teammitgliedern erzählt was man am Tag vorher gemacht und am darauf folgenden Tag macht. Auch Probleme können hier in der Runde erwähnt werden, aber eine allfällige Lösung wird hier nicht besprochen. Der „Scrum Master“ protokolliert das ganze und auch die erwähnten Probleme, falls es welche gab. Anschliessend nach dem Meeting geht man auf die erwähnten Probleme ein. Der Sinn vom „Daily Scrum“ ist der Informationsaustausch zwischen den Teammitgliedern.

In meinem Projekt lohnt es sich nicht nach SCRUM zu entwickeln, da ich nur eine einzelne Person bin und die relevanten Kriterien somit nicht erfülle. Aus diesem Grund habe ich mich für die Projektmethode IPERKA entschieden, weil dieser wie schon vorher erwähnt mir am vertrautesten ist.

## IPERKA

### Erklärung

**▪ I**nformieren

▫Die Anforderungen sowie sonstige für das Projekt relevante Informationen werden hier genau recherchiert.

**▪ P**lanen

▫ Die Planungsphase beinhaltet den Ablauf und die Methodik der Implementationsphase. Auch das konzipieren der gewonnen Information wird hier gemacht. Zu diesen Punkten kann auch ein Zeitplan erstellt werden.

***Wichtig: Die Planung muss sehr durchdacht gemacht werden, da IPERKA nicht rekursiv ist.***

**▪ E**ntscheiden

▫ Hier werden wichtige Entscheidungen getroffen, zum Beispiel für die Realisierung.

**▪ R**ealisieren

▫ Dies beinhaltet die Implementation des Projektes.

**▪ K**ontrollieren

▫ Die Kontrolle der ganzen Implementation wird hier gemacht und auch für die Auswertung protokolliert.

**▪ A**uswerten

▫Bei der Auswertung wird die Frage gestellt, was man gelernt hat aber auch was man in Zukunft besser machen kann. In dieser Phase wird retrospektiv das ganze Projekt beurteilt.

### Meine Facharbeit

Hier erläutere ich kurz die einzelnen Phasen in Bezug auf meine Facharbeit.

**▪ I**nformieren

▫Die Bewertungskriterien des IPA Berichtes habe ich recherchiert. Auch der Ablauf dieser 10 Tage sowie die Anforderungen meiner Facharbeit habe ich hier genau analysiert, sodass ich die Planung so gut wie möglich machen kann.

**▪ P**lanen

▫ In dieser Phase meiner Facharbeit werde ich die Planung machen. Dies beinhaltet einen Zeitplan mit den Arbeiten, welche ich dann Schritt für Schritt machen werde.

**▪ E**ntscheiden.

▫ Ich entscheide wie ich meine Facharbeit realisiere (zum Teil bereits in den Anforderungen definiert). Auch was für SW, HW, DLLs sowie auch der Ablauf der Implementation wird hier entschieden.

**▪ R**ealisieren

▫ Die Implementation erfolgt in dieser Phase.

***Tipp: Je besser die vorherigen Phasen gemacht worden sind desto flüssiger erfolgt die Realisierung.***

**▪ K**ontrollieren

▫ Meine Implementation kontrolliere ich indem ich die verschiedenen UseCases durchgehe und es auch mit Testfällen prüfe. Dies beinhaltet Blackbox- sowie auch Whiteboxtesting

**▪ A**uswerten

▫ Die Schlussreflexion ist der Hauptbestandteil dieser Phase. Auch die Auswertungen der Testfälle können hier integriert werden.

# Management Summary

## Ausgangssituation

## Umsetzung

## Ergebnis

# Recherchen für die Implementation

## Versionsverwaltung

Ich habe das Versionsverwaltungssystem Git gewählt, da dieser mir am bekanntesten und auch einfacher zu verstehen ist.

Ich habe GitHub für Windows eingerichtet, damit ich mein Projekt sowie auch meine Dokumente auf einem zentralen Server lagern kann. Dies ermöglicht mir jederzeit von überall das abrufen der Daten. Auch bei lokalem Verlust kann ich meine Daten immer wieder von den GitHub Servern herunterladen. Auch die Integration ins VS12 ist ganz einfach.

Notwendige Software:

GitHub for Windows: <https://windows.github.com/>

Git Source Control Provider in VS12: http://visualstudiogallery.msdn.microsoft.com/63a7e40d-4d71-4fbb-a23b-d262124b8f4c

Visual Studio Tools for Git: http://visualstudiogallery.msdn.microsoft.com/abafc7d6-dcaa-40f4-8a5e-d6724bdb980c

GitHub ist ein Hosting-Dienst für Software-Entwicklungsprojekte und ist Webbasiert. GItHub wurde von Chris Wanstrath, PJ Hyett und Tom Preston-Werner mit Ruby on Rails und Erlang entwickelt und wird seit Februar 2008 betrieben.

### Git: Kurze Beschreibung

#### Neues Repository

Zuerst wird ein neues Verzeichnis erstellt, geöffnet und anschliessend

***git init***

ausgeführt, somit wird ein neues git-Repository angelegt.

#### Repository auschecken

Eine Arbeitskopie kann man erstellen, indem man den folgenden Befehl ausführt:

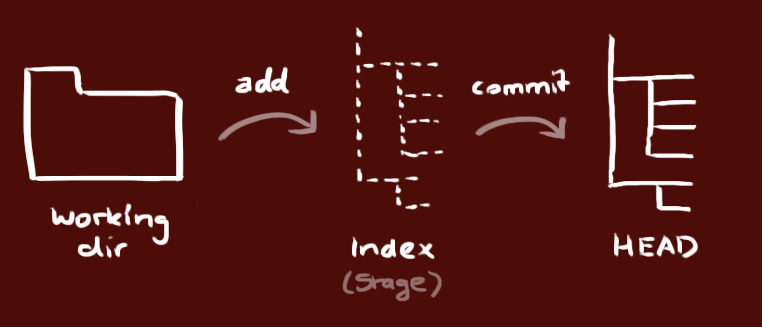
***git clone /pfad/zum/repository***

Falls man ein entferntes Repository benutzt, welches sich auf einem Server oder ähnlichem befindet:

***git clone benutzername@host:/pfad/zum/repository***

#### Workflow

Das lokale Repository besteht aus drei "Instanzen", die von git verwaltet werden. Die erste ist die **Arbeitskopie**, welche die echten Dateien enthält. Die zweite ist der **Index**, welcher als Zwischenstufe agiert und zuletzt noch der **HEAD**, der auf den letzten Commit zeigt.



#### Add & Commit

Änderungen können zum Index hinzugefügt werden mit dem Befehl

***git add <dateiname>***

***git add \****

Das ist der erste Schritt im git-Workflow, die Änderungen kann man mit:

***git commit -m "Commit-Nachricht"***

bestätigen. Jetzt befindet sich die Änderung im **HEAD**, aber noch nicht im entfernten Repository.

#### Änderungen hochladen (Check-In)

Die Änderungen sind jetzt im HEAD vom lokalen Repository. Um die Änderungen zum Server zu übermitteln benötigt man den folgenden Befehl:

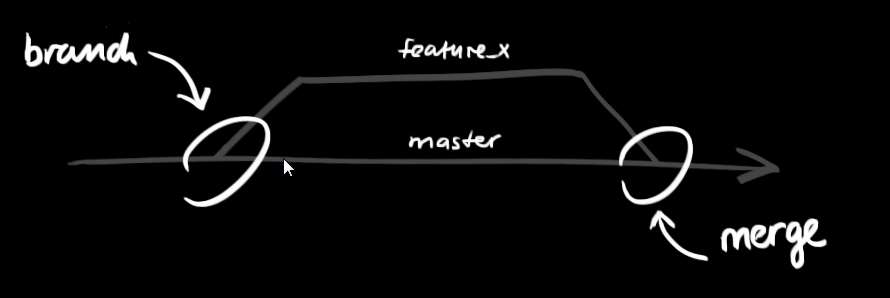
***git push origin master***

„master“ kann auch mit einem beliebigen anderen Branch ersetzt werden falls man mehrere Branches hat.

#### Branching

Branches dienen für die Entwicklung verschiedener Funktionalitäten getrennt vom „master“ Branch. Der master-Branch ist der "Standard" Branch, wenn man ein neues Repository erstellt.

Für die Entwicklung wird empfohlen neue Branches zu erstellen und anschliessend wenn die Funktionalität besteht diese dann mit dem master zusammenzuführen.



Um einen neuen Branch zu erstellen muss man folgendes eingeben:

***git checkout -b feat/controller***

Um zum Master zurück zu wechseln:

***git checkout master***

Und um den erstellten Branch wieder zu löschen:

***git branch -d feat/controller***

Ein Branch ist nicht für andere sichtbar bis man diesen in ein entferntes Repository hochlädt:

***git push origin <branch>***

#### Update & Merge

Um die neuesten Änderungen zu holen wird dieser Befehl ausgeführt:

***git pull***

Wenn man seinen Branch mit dem master zusammenführen möchte muss man folgenden Befehl ausführen:

***git merge <branch>***

Bevor man Änderungen zusammenführt, kann man Sie visuell darstellen indem man diese 2 Branches vergelicht. Dafür wird dieser Befehl ausgeführt:

***git diff <quell\_branch> <ziel\_branch>***

#### Änderungen rückgängig machen

Falls einem ein nicht behebbarer Fehler untergekommen ist, kann man die letzten Änderungen mit:

***git checkout -- <filename>***

auf den letzten stand im HEAD zurücksetzen. Änderungen welche sich bereits im Index befinden, bleiben bestehen.

Wenn man seine lokalen Änderungen ganz verwerfen möchte, kann man dies indem man den letzten Stand vom entfernten Repository übernimmt. Befehl ist wie folgt:

***git fetch origin***

***git reset --hard origin/master***

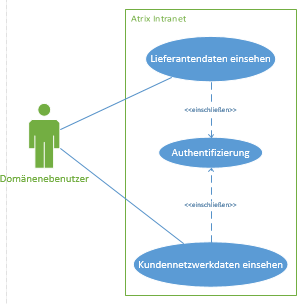
### Git Funktionalität (Visuelle Darstellung)

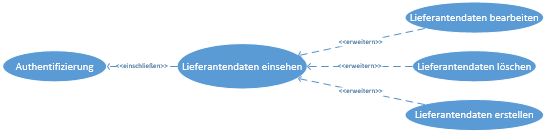
In der folgenden Abbildung kann man die Funktionalität noch visuell betrachten. Ich persönlich finde visuelle Erklärungen verständlicher und man kann somit einen besseren Einblick ins ganze erhalten.

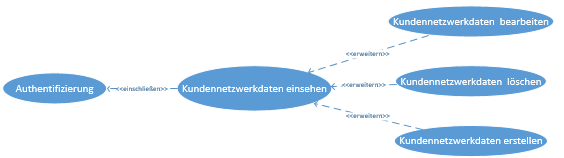


## UseCases

UseCases sind Anwendungsszenarien, welche die Interaktion zwischen Benutzer und dem System konkret mit wenig technischem Detail beschreiben. Diese UseCases benötige ich anschliessend auch für die Tests.







### Definition

UseCase Definition bedeutet eine simple Beschreibung der verschiedenen Anwendungsfälle.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 0 | | | | |
| Name | Authentifizierung | | | |
| Kurzbeschreibung | Die Intranet Seite anfordern | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Es werden bestimmte interne Daten (Kundennetzwerkdaten oder Lieferantenangaben) benötigt. | | | |
| Eingehende Daten | Authentifizierungsdaten (Benutzer + PW) | | | |
| Vorbedingungen | Benutzer muss von der Domäne sein | | | |
| Nachbedingungen, Ergebnis | Die Intranet Seite wird angezeigt | | | |
| Essentielle Schritte | 1. Benutzer ruft Seite auf 2. Authentifizierung im System | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 1 | | | | |
| Name | Daten Suchen | | | |
| Kurzbeschreibung | Domänenbenutzer möchte einen Lieferant oder Kunden suchen | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Der Kunde hat ein Problem und um schnellstmöglich Support leisten zu können wird der Kunde mit der Suchfunktion gesucht | | | |
| Eingehende Daten | Suchabfrage | | | |
| Vorbedingungen | Benutzer muss von der Domäne sein | | | |
| Nachbedingungen, Ergebnis | Resultat der Abfrage wird angezeigt | | | |
| Essentielle Schritte | 1. <<include>>: Authentifizierung 2. Kriterium mit der Suchfunktion suchen 3. <<extend>>: Lieferantenangaben anzeigen 4. <<extend>>: Kundennetzwerkdaten anzeigen | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 2 | | | | |
| Name | Kundennetzwerkdaten anzeigen | | | |
| Kurzbeschreibung | Anzeigen der Kundennetzwerkdaten für weitere Aktionen | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Die Netzwerkdaten von einem bestimmten Kunden werden für Supportarbeiten benötigt | | | |
| Eingehende Daten | - | | | |
| Vorbedingungen | Kundendaten existieren | | | |
| Nachbedingungen, Ergebnis | Die geforderten Kundendaten werden angezeigt. | | | |
| Essentielle Schritte | 1. <<include>>: Authentifizierung 2. Kunde auswählen 3. Kundennetzwerkdaten anzeigen | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 3 | | | | |
| Name | Kundennetzwerkdaten bearbeiten | | | |
| Kurzbeschreibung | Bearbeiten der vorhandenen Kundennetzwerkdaten | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Die Netzwerkdaten von einem bestimmten Kunden werden verändert, da der Kunde neuen Firmensitz hat. | | | |
| Eingehende Daten | *Überarbeitete Daten* | | | |
| Vorbedingungen | Kundendaten existieren | | | |
| Nachbedingungen, Ergebnis | Die Überarbeiteten Daten werden in die DB gespeichert und angezeigt. | | | |
| Essentielle Schritte | 1. <<include>>: Authentifizierung 2. <<extend>>: Kundennetzwerkdaten anzeigen 3. Kundennetzwerkdaten überarbeiten 4. Kundennetzwerkdaten prüfen 5. Die Daten in die DB speichern | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 4 | | | | |
| Name | Kundennetzwerkdaten löschen | | | |
| Kurzbeschreibung | Löschen der Kundennetzwerkdaten | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Die Netzwerkdaten von einem bestimmten Kunden werden gelöscht, da die Firma Konkurs gegangen ist. | | | |
| Eingehende Daten | - | | | |
| Vorbedingungen | Kundendaten existieren | | | |
| Nachbedingungen, Ergebnis | Kundendaten wurden erfolgreich aus der DB gelöscht. | | | |
| Essentielle Schritte | 1. <<include>>: Authentifizierung 2. <<extend>>: Kundennetzwerkdaten anzeigen 3. Kundennetzwerkdaten löschen | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 5 | | | | |
| Name | Kundennetzwerkdaten erfassen | | | |
| Kurzbeschreibung | Neuen Kunden mit den Kundennetzwerkdaten erfassen | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Ein neuer Kunde ist unter Vertrag | | | |
| Eingehende Daten | *Muss-Felder der Kundennetzwerkdaten Tabelle* | | | |
| Vorbedingungen | - | | | |
| Nachbedingungen, Ergebnis | Der neue Kunde wurde erfasst | | | |
| Essentielle Schritte | 1. <<include>>: Authentifizierung 2. Neuen Kundenerfassen 3. Kundennetzwerkdaten prüfen 4. Kundennetzwerkdaten in die DB eintragen | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 6 | | | | |
| Name | Lieferantenangaben anzeigen | | | |
| Kurzbeschreibung | Anzeigen der Lieferantenangaben für weitere Aktionen | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Die Lieferantenangaben von einem bestimmten Lieferant wird für künftige Bestellungen benötigt | | | |
| Eingehende Daten | - | | | |
| Vorbedingungen | Lieferantenangaben existieren | | | |
| Nachbedingungen, Ergebnis | Die geforderten Lieferantenangaben werden angezeigt. | | | |
| Essentielle Schritte | 1. <<include>>: Authentifizierung 2. Lieferant auswählen 3. Lieferantenangaben darstellen | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 7 | | | | |
| Name | Lieferantenangaben bearbeiten | | | |
| Kurzbeschreibung | Lieferant hat neuen Firmensitz | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Die Lieferantenangaben von einem Lieferant sind veraltet und müssen erneuert werden. | | | |
| Eingehende Daten | *Überarbeitete Daten* | | | |
| Vorbedingungen | Lieferantenangaben existieren | | | |
| Nachbedingungen, Ergebnis | Die Lieferantenangaben wurden erfolgreich bearbeitet | | | |
| Essentielle Schritte | 1. <<include>>: Authentifizierung 2. <<extend>>: Lieferantenangaben anzeigen 3. Lieferantenangaben überarbeiten 4. Lieferantenangaben prüfen 5. Die Daten in die DB speichern | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 8 | | | | |
| Name | Lieferantenangaben löschen | | | |
| Kurzbeschreibung | Anzeigen der Lieferantenangaben für weitere Aktionen | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Die Lieferantenangaben von einem bestimmten Lieferant wird für künftige Bestellungen benötigt | | | |
| Eingehende Daten | - | | | |
| Vorbedingungen | Lieferant existiert | | | |
| Nachbedingungen, Ergebnis | Die Lieferantenangaben wurden erfolgreich aus der DB gelöscht | | | |
| Essentielle Schritte | 1. <<include>>: Authentifizierung 2. Kunde auswählen und Daten anzeigen 3. Kundennetzwerkdaten löschen | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anwendungsfall 9 | | | | |
| Name | Lieferantenangaben erfassen | | | |
| Kurzbeschreibung | Neuen Lieferant mit den entsprechenden Angaben erfassen | | | |
| Akteure | Domänenbenutzer | | | |
| Auslöser | Ein neuer Lieferantenvertrag ist vorhanden und die Lieferantenangaben müssen nun erfasst werden | | | |
| Eingehende Daten | *Muss-Felder für die Lieferantenangaben Tabelle* | | | |
| Vorbedingungen | - | | | |
| Nachbedingungen, Ergebnis | Der neue Lieferant wurde erfasst | | | |
| Essentielle Schritte | 1. <<include>>: Authentifizierung 2. Neuen Lieferant erfassen 3. Lieferantenangaben prüfen 4. Lieferantenangaben in die DB eintragen | | | |
| Offene Punkte |  | | | |
| Änderungshistorie | Wann | Wer | Neuer Status | Was |
| Sonstiges, Anmerkungen |  | | | |

## GUI Design

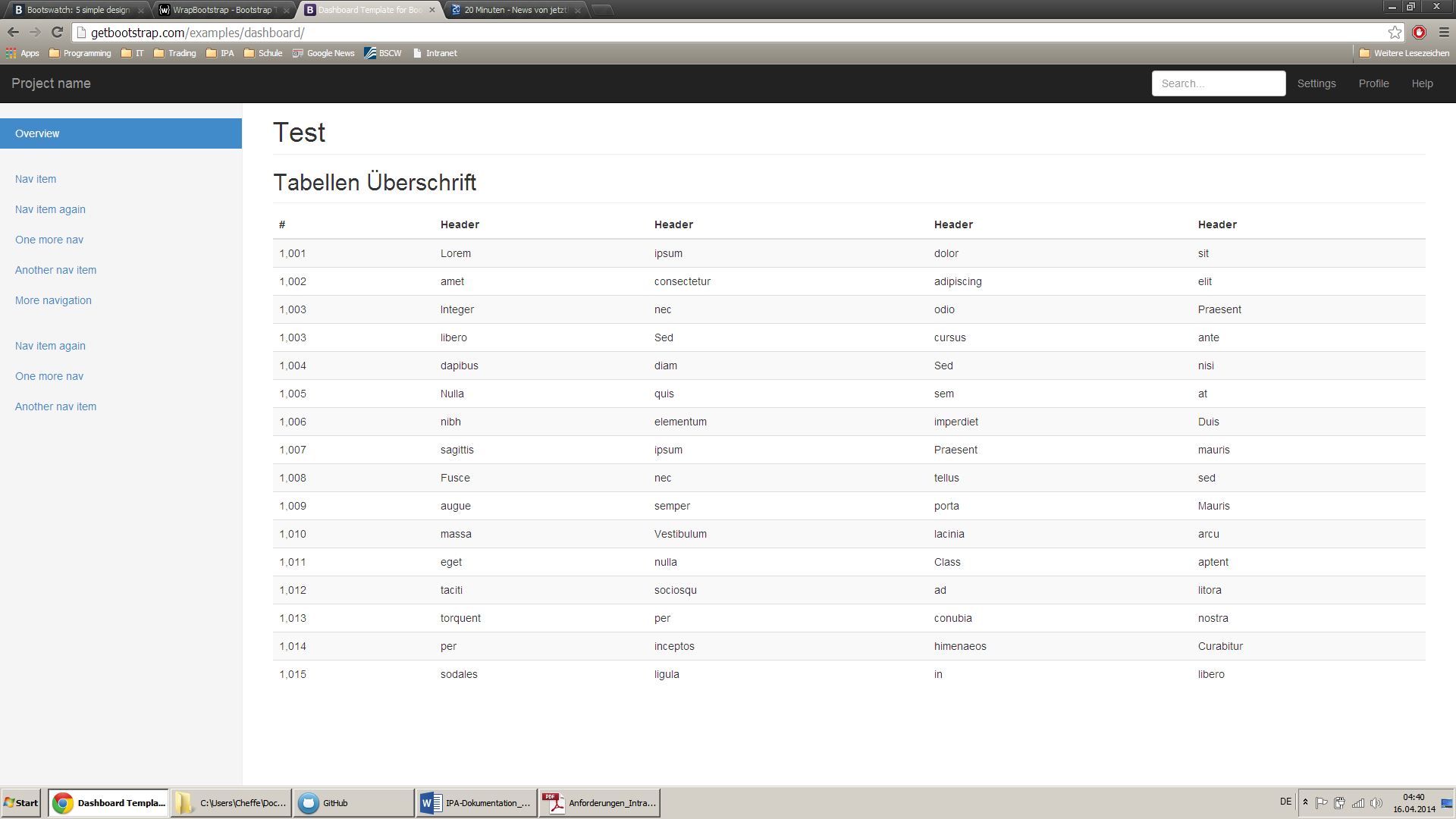
### Softwareergonomie

Die Softwareergonomie ist in der Aufgabenstellung bereits vorgegeben:

Der Aufruf durch einen Client erfolgt via Webbrowser. Die Datenübermittlung muss nicht verschlüsselt werden. Das Formular (Website) ist zweigeteilt: auf der linken Seite wird die Navigation durch die Funktionen stattfinden. Auf der rechten Seite der Website wird wahlweise eine Übersicht der gewünschten Daten oder die Bearbeitung einzelner Daten (Bearbeitungsmaske) erfolgen. Links oben soll das Atrix- Logo als Bild dargestellt werden. Dies ist auf dem Firmenlaufwerk abgelegt. In jeder Bearbeitungsmaske sollen Daten neu erfasst und bearbeitet werden können. Das Speichern von Daten soll via Button / Bestätigung explizit ausgelöst werden. Vor dem Speichern soll eine Plausibilisierung der eingegeben/mutierten Daten erfolgen. Ist die Plausibilisierung nicht vollständig, wird eine entsprechende Meldung/Hinweis ausgegeben. Dabei hat der Anwender die Möglichkeit, die eingegeben/mutierten Daten nochmals entsprechend der Plausibilisierungsmeldung anzupassen und das Speichern erneut auszulösen. Dieser Vorgang kann beliebig oft ausgeführt werden. Mit einer Abbruch-Funktion kann die Eingabe der neuen bzw. Mutation der bestehenden Daten ohne Speicherung verworfen werden. Das Löschen von bestehenden Daten kann ausschliesslich auf einer Übersicht erfolgen. Bevor die Daten gelöscht werden, muss das Löschen vom Benutzer explizit nochmals bestätigt werden. Nach dem Bestätigen der Löschung werden die Daten aus der Datenbank gelöscht. Gibt es Daten, die von den gelöschten Daten abhängig sind, müssen diese ebenfalls im gleichen Vorgang gelöscht werden. Ob dies mittels Referenzieller Integrität / Löschweitergabe erfolgt, steht dem Programmierer frei. Eine spätere Erweiterung der Funktionalität muss mit gleicher Ergonomie möglich sein. Wie die einzelnen Bereiche im Detail dargestellt werden, bleibt dem Programmierer überlassen.

### Frontend (Layout)

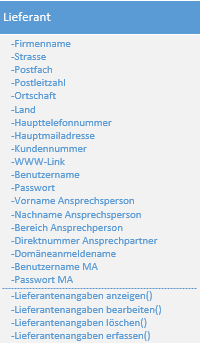
Meine Layoutseite sieht wie in der folgenden Abbildung aus.



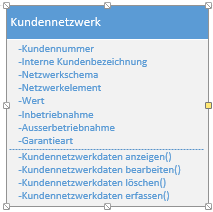
## Klassendefinition

### UML Klassendiagramm

#### Lieferant



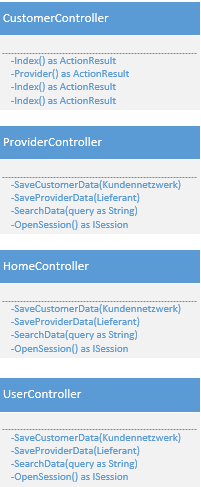
#### Kundennetzwerk



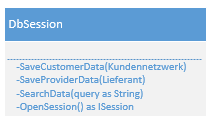
#### User(Mitarbeiter)



#### Controller



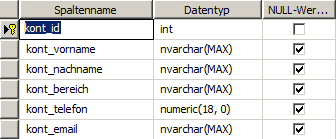
#### Sonstige Klassen



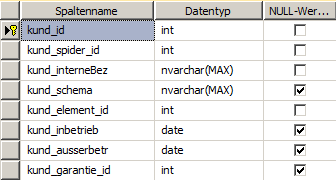
## MS SQL

### Logisches Datenmodell

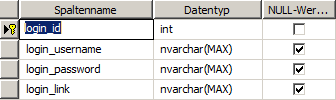
#### Kontakt



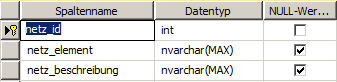
#### Kundennetzwerk



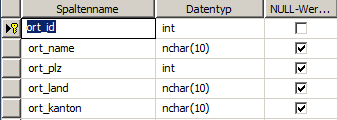
#### Login



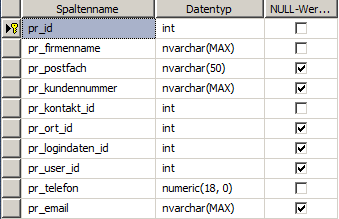
#### Netzwerkelement



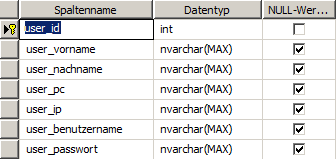
#### Ortschaft



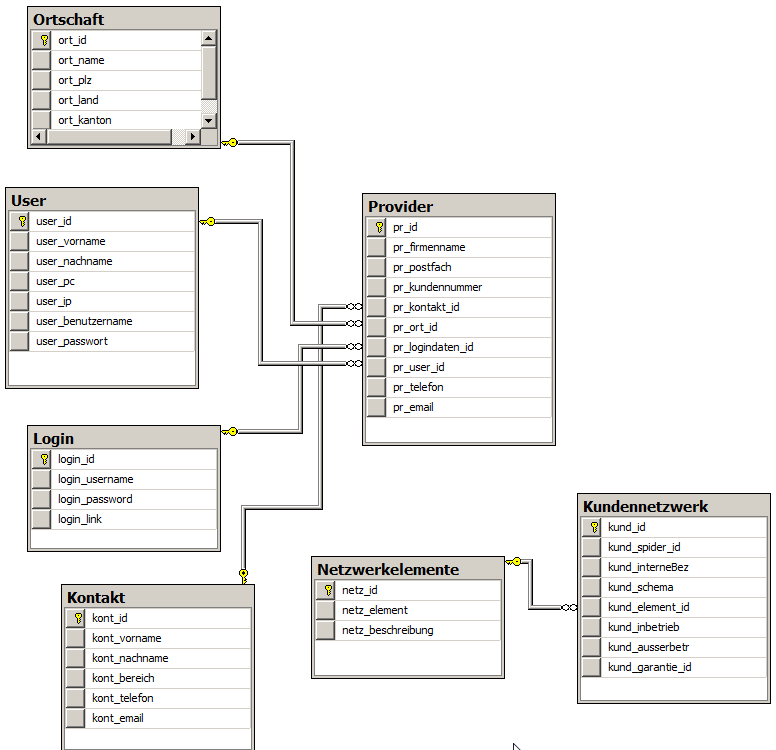
#### Provider



#### User



### ERD

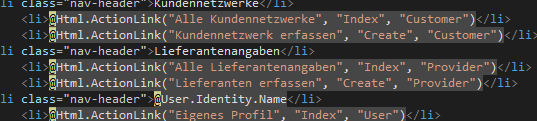


## Implementationsmethodik

### ASP.NET mit VB.NET

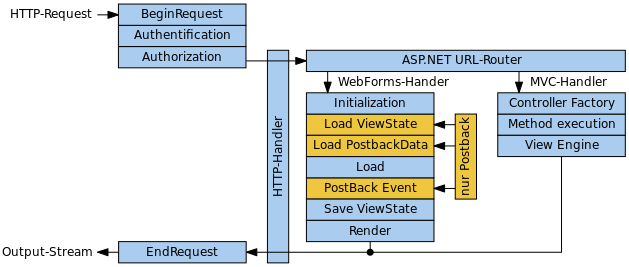
Die Implementation habe ich mit Hilfe von ASP.NET und VB.NET erstellt. Für mein Projekt habe ich dazu noch das MVC Framework benutzt.

In meinem Projekt habe ich auch die Razor Syntax benutzt. Dies war für mich zwar neu aber ich kam schnell damit zurecht. Die Razor Syntax vereinfacht die Programmierung und sieht wie folgt aus:



@HTML.ActionLink gibt einen <a></a> HTML Tag mit dem virtuellen Pfad zur dazugehörigen Aktion.

Die folgende Abbildung verschafft einen Überblick zu ASP.NET:



### Model-View-Controller

Das MVC Modell trennt eine Applikation in 3 Schichten:

**Model**: beinhaltet die DB Anbindung und die dazugehörigen Objekte

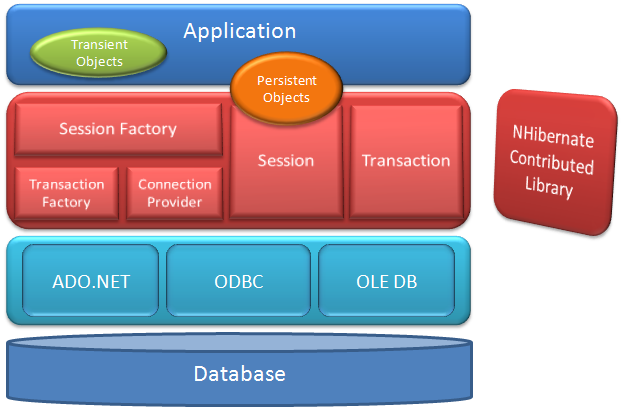
**Controller**: der Controller dient für das Zusammenspiel vom Model mit der View.

**View**: Die View ist das GUI der Applikation und ist nur für die Darstellung verantwortlich.



Das MVC Framework mit ASP.NET vereinfacht auch die Testbarkeit der Applikation.

Das Zusammenspiel von ASP.NET MVC mit NHibernate sieht wie folgt aus:



# Implementation

## Konfiguration

### Web.config

## Model

### nHibernate

#### Mappings

#### Klassen zu den Mappings

## View

### Create

### Update

### Delete

### Index

## Controller

### Aufbau

### Funktionalität

## Unit Tests

Kontrolle

# Allgemeine Vorbereitungen

# Testprotokoll

Die Testfälle beziehen sich auf die ganze Applikation und beinhalten das Verhalten der Funktionalität.

## Startseite

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr. | Name | Vorbereitung | Ausgangszustand | Aktion | Erwartetes Ergebnis |
| 1 | Falsche Logindaten |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |

## Kundennetzwerke

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr. | Name | Vorbereitung | Ausgangszustand | Aktion | Erwartetes Ergebnis |
| 1 | Falsche Logindaten |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |

## Lieferantenangaben

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr. | Name | Vorbereitung | Ausgangszustand | Aktion | Erwartetes Ergebnis |
| 1 | Falsche Logindaten |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |

## Benutzerseite

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr. | Name | Vorbereitung | Ausgangszustand | Aktion | Erwartetes Ergebnis |
| 1 | Falsche Logindaten |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |

# Testberichte

## Startseite

## Kundennetzwerke

## Lieferantenangaben

## Benutzerseite

Auswerten

# Reflexion

## Positiv

## Negativ

# Quellenverzeichnis

<http://msdn.microsoft.com/> -> .NET Klassen Referenz

<http://getbootstrap.com/> -> Bootstrap CSS für die Views

<http://Codeproject.com> -> Bei Problemen bei der Implementierung

<http://Stackoverflow> -> Bei Problemen bei der Implementierung

<https://github.com/> -> Für die Versionsverwaltung

<http://nhforge.org/> -> NHibernate Dokumentation

# Abbildungsverzeichnis

# Anhang: Listings

## Controller

### CustomerController

### HomeController

### ProviderController

### UserController

## Views

### Kundennetzwerk

#### Create

#### Update

#### Delete

#### Index

### Lieferantenangaben

#### Create

#### Update

#### Delete

#### Index

### Benutzerangaben

#### Create

#### Update

#### Delete

#### Index

## Model

### XML Mappings

### Mappingklassen

### NHibernateHelper