CaseStudy BusinessStreamline

- Semester 1 Webtechnologie -

BusinessStreamline

B2B PLATFORM

Ivan Hörler, Harayanamoorthy Prashath, Ciullo Alessio

31. März 2017

Inhaltsverzeichnis

1 Ausgangslage

1.1 Zielsetzung an die Fallstudie

Die Webtech Case Study im dritten Semester der Ausbildung zum Dipl. Techniker IT befasst sich mit den Technologien HTML, PHP und SQL aus den Fächern Webtechnologie und Datenbanken. Die Zielsetzung ist vom Fachgruppenleiter dokumentiert und wird gemäss diesem umgesetzt. Es sind vier Ziele zu verfolgen:

- Der Studierende hat ein Geschäftsmodell in einer einfachen dynamischen Weblösung abgebildet.
- Der Studierende hat einen sinnvollen Testplan entworfen und diesen auf seine Lösung angewendet.
- Der Studierende hat die zur Verfügung stehenden Stunden in einer Planung den Aufgaben zugeordnet und eine Abweichungsanalyse erstellt.
- Der Studierende hat die Lösung dokumentiert..

1.2 Rahmenbedingungen

Thema	Festlegung
Art der Arbeit	Gruppenarbeit zu dritt
Umfang	je 80 Stunden
Präsentation	15min. vor der Klasse
Abgabe	Elektronisch
Benotung	gemäss Beurteilungsschema

1.3 Umschreibung der Aufgabenstellung

Folgendes Geschäftsmodel ist umzusetzen:

- Firmen (Nachfrager), welche Teile zur Herstellung Ihrer Produkte benötigen (Schrauben, Muttern, Nieten...) können diese auf unserer Plattform erfassen.
- Erfasst werden dabei die genaue Bezeichnung, die benötigte Menge und Qualität und der gewünschte Lieferzeitpunkt.
- Anbieter solcher Teile können nun nach gewissen Teilen suchen (z.B. sucht jemand Schrauben des Typ II-C ?)
- Findet der Anbieter einen Eintrag, so kann er, anonymisiert, ein Angebot hinterlegen (ich biete Schraube II-C zu 12 Fr. per 100 Stück an)
- Die Nachfrager können anschliessend alle Angebote, welche zu einem bestimmten Teil abgegeben worden sind, auflisten und das Ihnen passende auswählen.
- Ein so ausgewähltes Angebot wird dem Anbieter als Bestellung im XML- Format übermittelt

2 Projektorganisation

Wir entscheiden uns für eine agile Entwicklung nach SCRUM.

2.1 Personen, Aufgaben und Werkzeuge

Prashath Harayanamoorthy wird die Aufgabe des SCRUM Masters übernehmen, während sich Ivan Hörler als Developer integriert und Alessio Ciullo die Sprint Reviews übernimmt.
Dabei wurde die Daily- auf Weeklymeetings ausgedehnt, um den Ansprüchen der Aufgabe und der nebenberuflichen Erledigung der Case Study gerecht zu werden.
Die Dokumentation wird jeweils aktuell mitgeführt und über die gleiche Versioning Software verteilt wie der Code. Geschrieben wird in Latex, um die Mitarbeit aller direkt über commits ins ”Git” Repository sicherzustellen. Programmiert wird ausschliesslich in HTML, PHP und MYSQL Sprache.
Der SCRUM Master erstellt die zu erledigenden Arbeitspakete (nachfollgend ”AP” genannt), dazu das Usecase Diagramm und die User Storys und übergibt diese dem Developer der den Sprint damit beginnt. Durch den Revisor wird die jeweilige Itteration beglaubigt, bevor zum nächsten AP übergegangen wird.

2.2 Kommunikationsstrukturen

Zur Kommunikation untereinander wird ein Telegramm Messenger eingesetzt. AP und Diagramme werden direkt im Dokument eingefügt und stehen somit mit dem nachgehenden Bericht des Developers allen zur Einsicht zur verfügung.

2.3 Qualitätsmanagement

Nach jedem AP wird das Geleistete untersucht und eventuelle Abschweifer von der Aufgabenstellung gleich behoben. Untersucht wird anhand von User Stories und Usecase Diagrammen, ob die Lösung den Vorgaben entspricht.

2.4 Abgrenzung

Die Platform ist anonym. Die BusinessStreamline speichert keine Daten die auf die wahre Identität des Benutzers hinweist. Die Nutzung ist freiwillig und basiert auf einer Hohlschuld.

3 Analyse

3.1 Interessensgruppen

Die BusinessStreamline als B2B Partner verfolgt das Ziel Anbieter und Nachfrager zu verbinden und kommerziellen Handel über die Platform anonym abzuwickeln.

3.2 Vision

Die Vision beschreibt den Vorgang von Anwendern auf der Website der Firma BusinessStreamline graphisch als Kundenbeziehung.

3.4 User Stories

Ausgangspunkt ist diese Syntax:

> Als <Benutzerrolle> will ich <das Ziel>, so dass <Grund für das Ziel>.

3.3 Kundenbeziehung

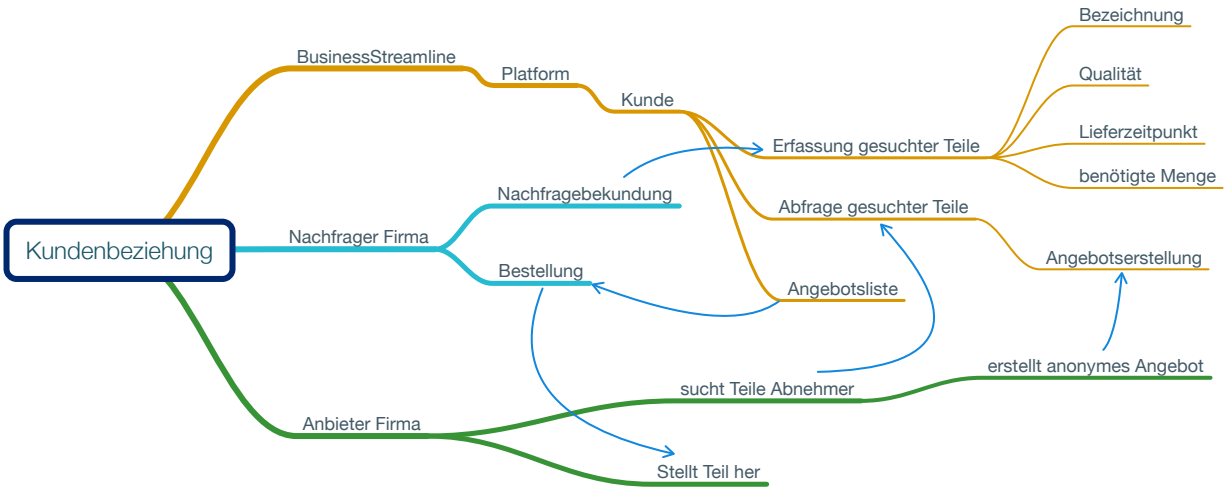


Abbildung 1: Kundenbeziehung

3.4.1 BusinessStreamline

- Als Dienstleister will ich, dass der Kunde sich anmelden muss, so dass kein unbefugter Zugang besteht.
- Als Dienstleister will ich Kontrolle über die Kundenanmeldung haben, so dass kein Kunde zweimal registriert sein kann.
- Als Dienstleister will ich die gesuchten Teile abfragen können, so dass jederzeit Gewissheit über die Anzahl Nachfragen besteht.
- Als Dienstleister will ich die Angebote überprüfen können.
- Als Dienstleister will ich, dass alle Nachfrager und Anbieter anonym verhandeln können.

3.4.2 Nachfrager

- Als Nachfrager will ich eine Bezeichnung eines gesuchten Teils hinterlegen können, so dass ein Anbieter das Teil spezifisch suchen kann.
- Als Nachfrager will ich die Qualität eines gesuchten Teils hinterlegt haben, so dass ich mich an Qualitätsmerkmalen orientieren kann.
- Als Nachfrager will ich einen Lieferzeitpunk hinterlegen können, so dass ein Anbieter über die Lieferfrist informiert ist.
- Als Nachfrager will ich die benötigte Menge angeben können, so dass ich diese mit einer Bestellung bekomme.

- Als Nachfrager will ich ein Angebot annehmen können, so dass dem Anbieter eine Bestellung zugesandt wird.
- Als Nachfrager will ich das Teil innerhalb der Lieferzeit zugesandt bekommen.

3.4.3 Anbieter

- Als Anbieter will ich abfragen, welche Teile gesucht sind, so dass ein Angebot erstellt werden kann.
- Als Anbieter will ich, dass alle Eckpunkte geklärt sind, so dass keine unvorhergesehenen Kosten entstehen.
- Als Anbieter will ich ein Angebot abgeben können, so dass ein schriftlicher Vertrag entsteht.
- Als Anbieter will ich anonym bleiben, so dass die Nachfragefirma nur anhand des Preises entscheidet wer sie berücksichtigt.
- Als Anbieter will ich bei der Bestellung eine schriftliche Antwort erhalten, so dass der Vertrag gültig ist.

3.5 Zielsetzung

3.5.1 Harte Ziele

- Benutzer muss sich einloggen können.
- Benutzer muss Nachfrage mit genannten 4 Faktoren hinterlassen können.
- Benutzer muss ein Angebot für eine Nachfrage erstellen können.
- Benutzer muss ein Angebot annehmen können.
- Benutzer muss das angenommene Angebot übermittelt bekommen.

3.5.2 Weiche Ziele

- Benutzer kann das Angebot per E-Mail übermittelt bekommen.
- Benutzer kann in Angebot die Telefonnummer und Anschrift hinterlassen.
- Benutzer kann das Angebot anpassen.
- Benutzer kann sein Passwort ändern.

4 Lösungsvarianten

4.1 Variante 1

XML Storage - Das XML Fileformat erlaubt das Speichern und abrufen von Datensätzen anhand von DOM Objekten. Da das angestrebte Ausgabeformat ein XML File ist, kann die Speicherung aller Datensätze auch in XML Nodes geschehen.

Vorteile:

- Keine Verbindung zu externen Datensätzen nötig.
- Die Sprache für den Zugriff, sowie das Speichern von Datensätzen kann einzig mit PHP geschehen.
- Keine Konvertierung der Ausgabe nötig, da die Daten in XML schon bestehen, müssen sie nur neu angeordnet werden.

Nachteile:

- Das Abfragen von XML Strukturen ist im Netz weniger gut dokumentiert als Datenbankabfragen.
- Das Speichern von Files birgt eventuell Sicherheitsrisiken die wir noch nicht abschätzen können.

4.2 Variante 2

MYSQL Database - MYSQL Datenbanken sind opensource Datenbanken, die einfache und standardisierte Abfragen und Speicherungen über bestehende und in PHP integrierte Schnittstellen erlauben.

Vorteile:

- Gut dokumentierte und Standardisierte Zugriffe.
- Tabellen mit Spalten und Reihen, die eine geordnete Datenstruktur vereinfacht.
- SQL Datenbanken wurden im Unterricht schon behandelt.

Nachteile:

- Infrastruktur muss vorhanden sein.
- dezentrale Daten.
- das Thema MYSQL queryes wurde noch nicht behandelt.

5 Detailplanung

5.1 Benutzerführung

Damit der Kunde sieht, wie seine Webseite funktioniert, wurde ein Mockup zu den beiden Kundentypen erstellt.

5.1.1 Nachfrager

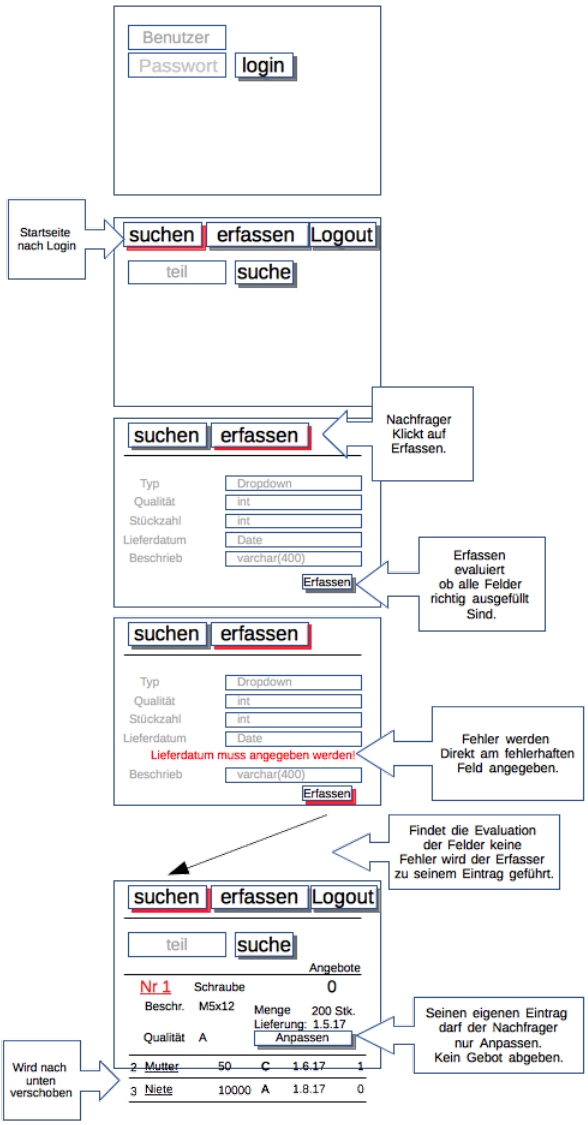


Abbildung 2: Benutzerführung des Nachfragers

5.1.2 Anbieter

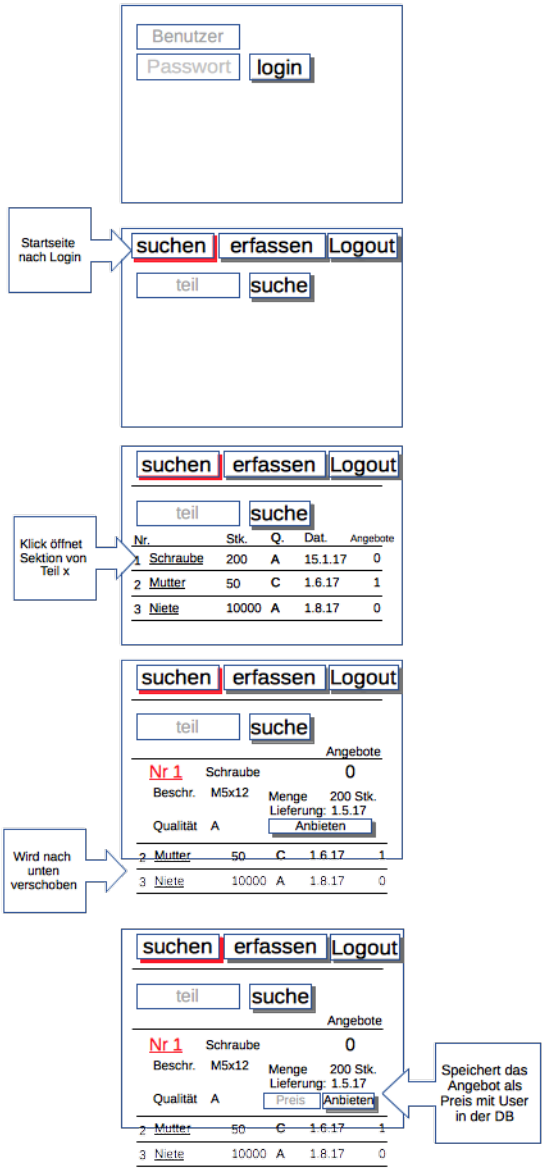


Abbildung 3: Benutzerführung des Anbieters

5.2 Usecase Diagramme

5.2.1 Login

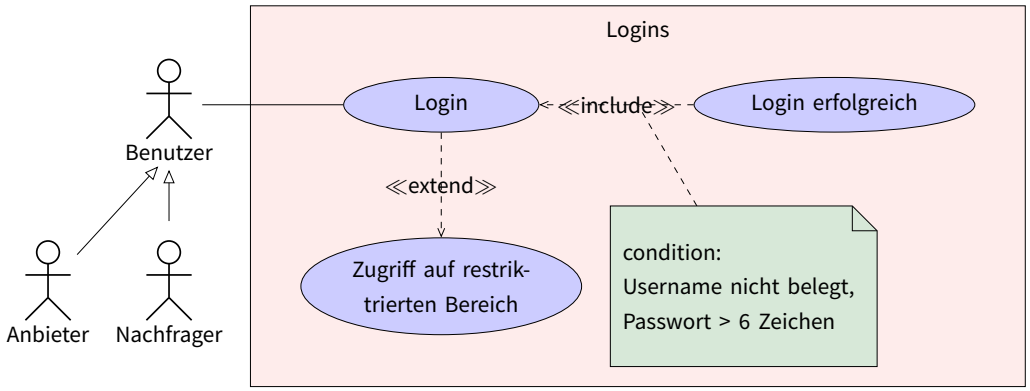


Abbildung 4: Login vorgang

5.2.2 Nachfrage

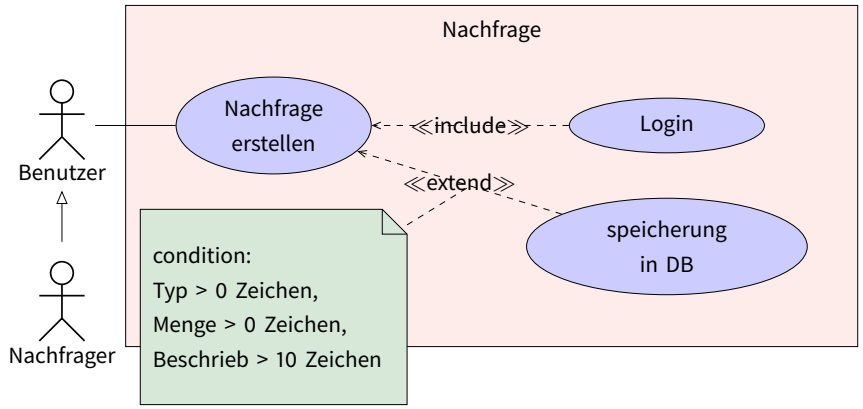


Abbildung 5: Vorgang zur Nachfrageerfassung

5.2.3 Suche

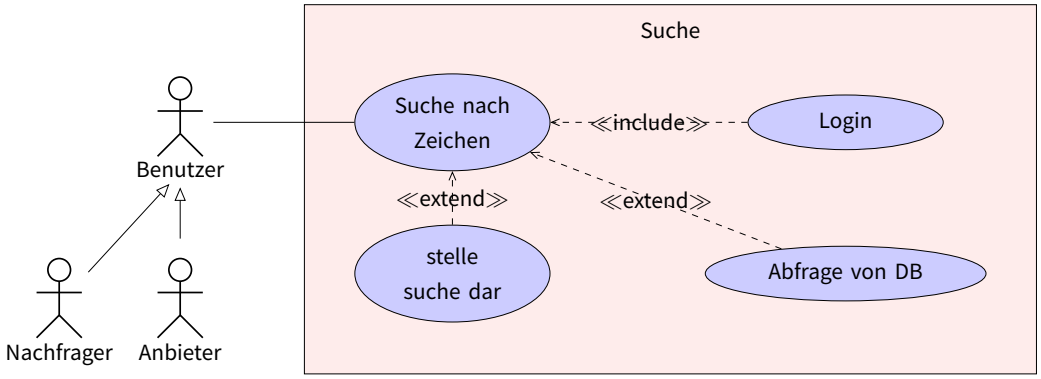


Abbildung 6: Vorgang der Suche

5.2.4 Angebot

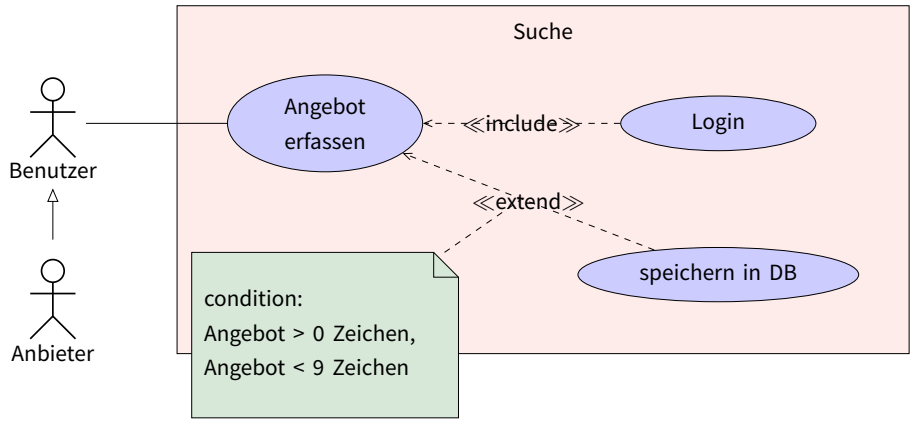


Abbildung 7: Vorgang des Angebots

5.2.5 Profile

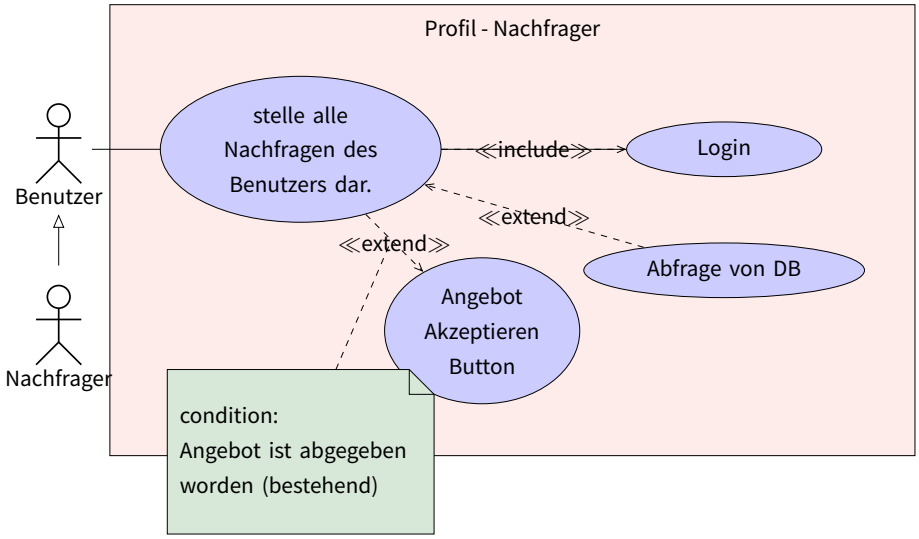


Abbildung 8: Abruf der Profilseite aus Sicht des Nachfragers

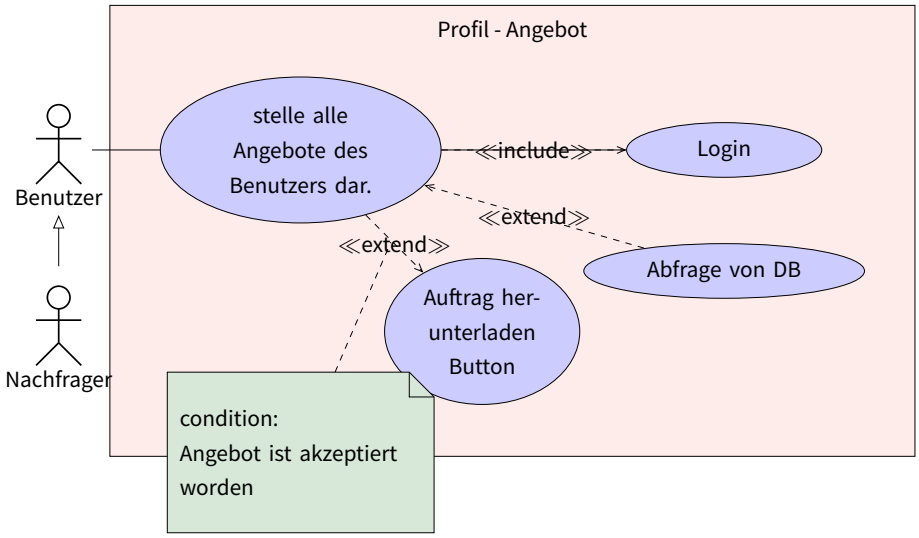


Abbildung 9: Abruf der Profilseite aus Sicht des Anbieters

5.2.6 XML

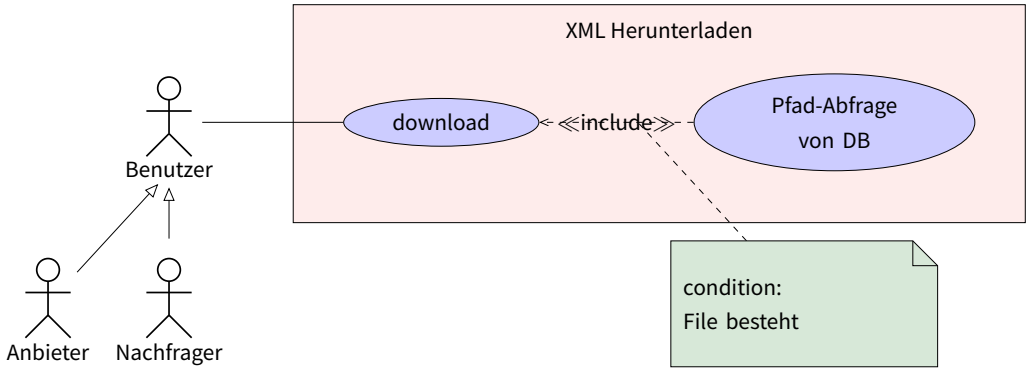


Abbildung 10: Login Vorgang

5.3 Architektur

Die Datenstruktur ist flach auf einer Ebene aufgebaut.
Einzig die XML Files werden in einen Ordner abgelegt.

Die Software-Architektur der eingelogten Benutzerseiten ist auch Seitenorientiert. Das bedeutet, dass jede Seite die aufgerufen wird, komplett geladen wird, einen eigenen header/body hat und nur die Navigation verschachtelt eingefügt wird.

5.4 Datenbank

Die Gruppe hat sich entschieden, das Projekt auf Basis einer MySQL Datenbank mit PHP zu erstellen. Da ein Hoster über ein Team-Mitglied zur Verfügung stand.

Zur Visualisierung wurden RM und ERM mit Dia erstellt:

5.4.1 ERM

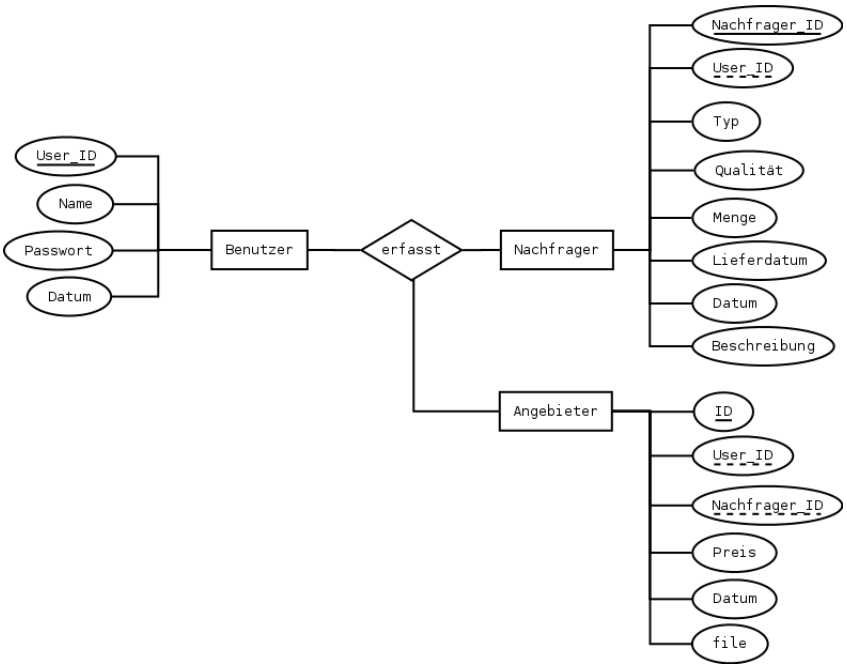


Abbildung 11: ERM zur Datenbank für BusinessStreamline

5.4.2 RM

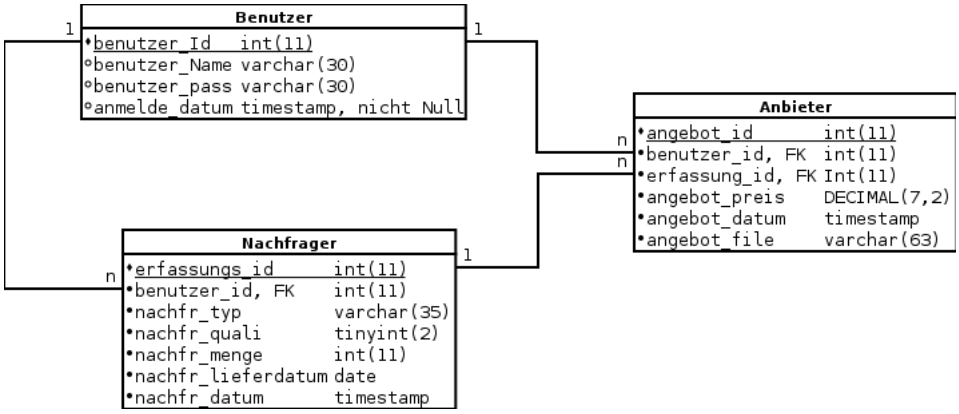


Abbildung 12: RM zur Datenbank für BusinessStreamline

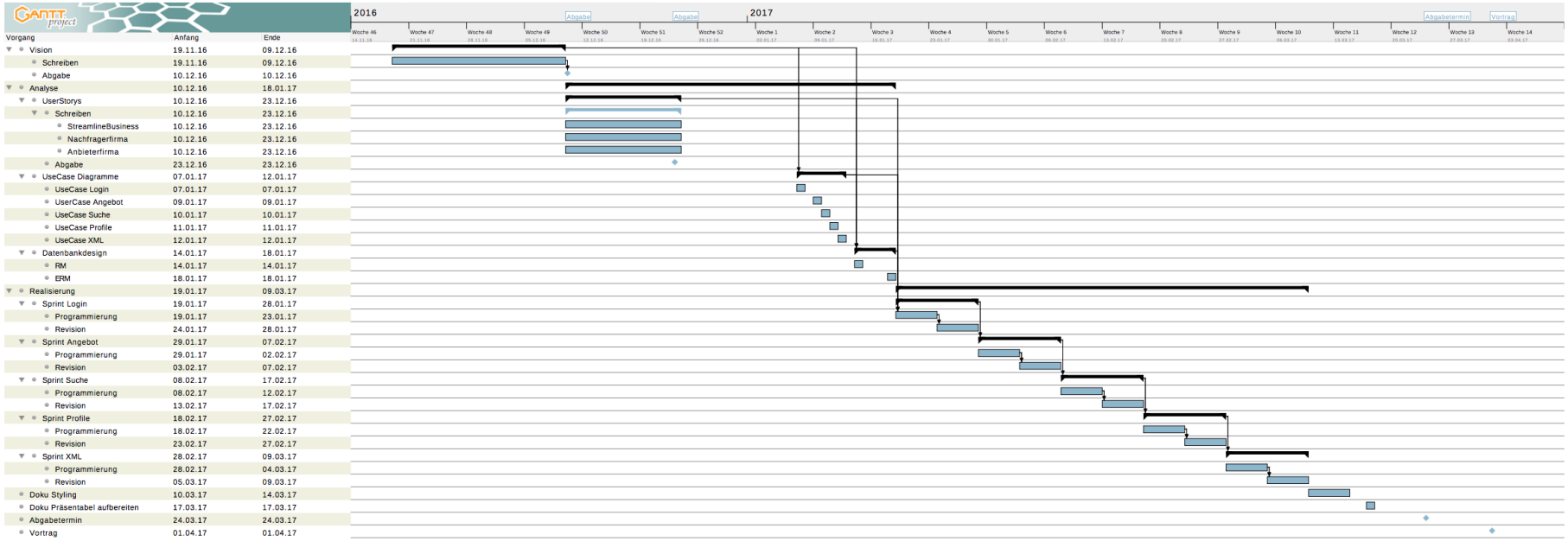
5.5 Webspace

Die Seite ist online im Web unter folgender Adresse zu finden:

http://wt1cs1.hoerler.us/

5.6 Termine und Meilensteine

5.7 Zeitplan



Vorgang	Anfang	Ende
Vision	19.11.16	09.12.16
Schreiben	19.11.16	09.12.16
Abgabe	10.12.16	10.12.16
Analyse	10.12.16	18.01.17
UserStorys	10.12.16	23.12.16
Schreiben	10.12.16	23.12.16
StreamlineBusiness	10.12.16	23.12.16
Nachfragerfirma	10.12.16	23.12.16
Anbieterfirma	10.12.16	23.12.16
Abgabe	23.12.16	23.12.16
UseCase Diagramme	07.01.17	12.01.17
UseCase Login	07.01.17	07.01.17
UserCase Angebot	09.01.17	09.01.17
UseCase Suche	10.01.17	10.01.17
UseCase Profile	11.01.17	11.01.17
UseCase XML	12.01.17	12.01.17
Datenbankdesign	14.01.17	18.01.17
RM	14.01.17	14.01.17
ERM	18.01.17	18.01.17
Realisierung	19.01.17	09.03.17
Sprint Login	19.01.17	28.01.17
Programmierung	19.01.17	23.01.17
Revision	24.01.17	28.01.17
Sprint Angebot	29.01.17	07.02.17
Programmierung	29.01.17	02.02.17
Revision	03.02.17	07.02.17
Sprint Suche	08.02.17	17.02.17
Programmierung	08.02.17	12.02.17
Revision	13.02.17	17.02.17
Sprint Profile	18.02.17	27.02.17
Programmierung	18.02.17	22.02.17
Revision	23.02.17	27.02.17
Sprint XML	28.02.17	09.03.17
Programmierung	28.02.17	04.03.17
Revision	05.03.17	09.03.17
Doku Styling	10.03.17	14.03.17
Doku Präsentabel aufbereiten	17.03.17	17.03.17
Abgabetermin	24.03.17	24.03.17
Vortrag	01.04.17	01.04.17

Abbildung 13: Zeitplanung

6 Realisation/Dokumentation

6.1 Erstes Login

Nachdem wir einige Tutorials durchgearbeitet hatten, fanden wir mit codeingcage [**?**] ein Tutorial, das die Abläufe genügend einfach gestaltet hatte, so dass wir auch in der Lage waren, es zu verstehen, verändern und unseren Bedürfnissen anzupassen. Wir haben es zuerst nach dem vorgezeigten Schema erstellt und dann gemerkt, dass wir die E-Mail Adresse als Login Feld eher als lästig empfinden. Also machten wir uns an die Arbeit, um den Login nach User-Namen umzukrempeln und entfernten alle E-Mail Felder und Variablen auch aus der Datenbank. Als spass fügten wir noch ein Bild ein und entfernten alle bootstrap CSS Klassen und ersetzten sie mit unseren aus dem lokalen style.css File um das ganze auch schön aussehen zu lassen ohne externe Abhängigkeiten.

6.2 Datenbankeinträge

Um die Datenbankeinträge gemäss RM erstellen zu können wurde als erstes das .sql File zur Erstellung der Datenbank erstellt. Danach wurden von Hand drei Einträge gemacht, um die Abfrage testen zu können. Da der Unterricht die SQL-Querrys noch nicht abgedeckt hatte, schob der Dozent eine kurze Einführung samt Beispiel ein, was als Vorlage für die weitere Arbeit diente. Zudem entnahm ich dem youtube Video [**?**] wichtige Informationen zum Vorgehen mit PDO. Um die Darstellung zu gliedern, wurde die Abfrage in HTML Tabellen integriert. Das Eintragsdatum musste von YYYY-MM-DD-hh-mm gekürzt werden und wird nun als Datum ohne Zeitangabe dargestellt. Die Eintrags Erfassung wird mit dem gleichen Vorgehen gemacht und fängt Fehler wie leergelassene Felder oder zu kurze Beschreibungen ab.

6.3 Angebotsabgabe

Die Selektion von einem Eintrag gestaltete sich schwierig. Nachdem die erste Idee mit onclick auf der Tabellenzeile nicht funktionierte, weil das Thema JavaSkript noch nicht angeschnitten wurde und mir nicht klar war, wie es umzusetzen ist, wurde auf einen einfachen Link zurückgegriffen und per ”GET” Befehl die ID der Zeile übermittelt. Im Formular des Angebots wird zuerst geprüft, ob das Angebot nicht vom eingelogten Nutzer stammt und falls doch, eine Meldung platziert, damit keine Angebote auf eigene Nachfragen gemacht werden können. Ein Nutzer, der die Nachfrage nicht erstellt hat, kann per ”POST” befehl ein Angebot abgeben, das in der ”Anbieter” Tabelle gespeichert wird. Vorgängig werden leere Übermittlungen und Angebote mit über 9 Ziffern abgefangen und Fehlermeldungen platziert.

6.4 Profile

Der Ersteller einer Nachfrage muss nun noch ein Interface zur Akzeptierung eines Angebots erhalten, das dann die Bestellung auslösen wird. Da die erforderlichen Werte für die Auflistung der Einträge aus zwei Tabellen kommen musste ein Join der Tabellen erarbeitet werden. Durch IF/ELSE abfragen wird ermittelt, ob für eine Nachfrage schone ein Angebot erstellt wurde und in dem Falle der Download Button für das geforderte XML angezeigt, oder dann k.A. für kein Angebot angezeigt.

6.5 XML

Da XML eine eigene Skript art ist, mussten wir uns erst damit auseinandersetzen. Gut gibts dazu mehrere Tutorials. Als wir dann mit dem Umsetzen der Aufgaben anfingen, hatten wir schon einige fertig erstellte Skripts gefunden, die diese Aufgabe automatisch durch Zuweisung der Tabellenspaltennamen erstellen. Jedoch waren alle für mehrere Tabellen-Zeilen ausgelegt und wir mussten diese dann an die Gegebenheit, dass nur für ein Teil (und damit eine Zeile) iteriert wird, anpassen. Die beste Dokumentation fanden wir unter [**?**] http://php.net/manual/en/book.dom.php, welche mit einer Beschreibung auf Stackexchange zusammenpasste und uns so eine Grundlage zur Erstellung bot. [**?**].

6.6 Filedownload

Diese Problemstellung haben wir komplett unterschätzt. Die Art und Programmierung für den Trigger eines Downloads mit PHP ist kompliziert. Durch Ausprobieren und der Kombination von verschiedenen Lösungen aus dem Internet, konnte es dann bewerkstelligt werden. [**?**]

6.7 Testing

Das Testen der Funktionen wurde auf Basis der Use-Case Diagramme durchgeführt. Dabei stellt jeder Usecase ein Testszenario dar. Diese Testszenarien wurden aufbauend auf der agilen Entwicklung jeweils am Ende des Sprints als Revision angehängt und genau so wie das Diagramm es darstellt für den jeweiligen Benutzertypen durchgespielt. Bei Falschinterpretationen oder Fehlern in der Programmierung wurde bilateral darauf eingegangen und das Problem durch eine Iteration behoben. Zu den einzelnen Fehlern die behoben werden mussten, findet man das Protokoll in der Abweichungsanalyse **??**

7 Zeitaufwandsliste

Arbeitspaket	geplant	geleistet	delta	Erklärung
Projektorganisation	1	1		
Projekt Analyse	5	5		
Lösungsvarianten	5	3	-2	Schell wurde klar das V2 im Web besser dokumentiert ist.
Detailplanung	40	60	20	Die Usecase Diagramme wurden wiederholt überarbeitet.
Dokumentation	40	66	26	LaTeX verstehen und einrichten
Vision	1	1		
UserStories	3	3		
Benutzerführung	2	3	1	gezeichnet mit Libreoffice
Datenbank	1	1		
erstes login/out	2	8	6	
Datenbankeinträge	12	16	4	der Schulische hintergrund fehlte zu dem Zeitpunkt noch.
Abgebotsabgabe	10	8	-2	
User Profile seite erstellt	6	16	8	
Angebotabgabe	6	6	0	
Angebote für eigene Nachfr. deakt.	2	4	2	
suche erstellt	10	8	-2	
XML übersetzer	14	16	2	musste erst XML verstehen.
Filedownload	3	10	10	Aufgabe unterschätzt.
Abweichungsanalyse	1	1		
Dokumentation zusammenführen	2	12	10	Aufgabe unterschätzt.
Total	166	248	83	h

Abbildung 14: Zeitaufwand

7.1 Abweichungsanalyse

Produkt:

- Das Login ist mit einer separaten dbconfig.php angebunden, da zu dem Zeitpunkt die Vererbung und deren Anwendung noch nicht im Unterricht thematisiert wurde, konnte damit nicht weiter gearbeitet werden.
- Die Angebotsabgabetaste sollte gemäss Planung auf der gleichen Seite wie die Nachfragen stehen. Das konnte aufgrund fehlender Kenntnisse nicht so gelöst werden. Der Einfachheit halber wurde dazu eine neue Seite erstellt. Das eigentliche Problem war, dass der benötigte Schulstoff zu dem Zeitpunkt nicht behandelt war, als wir diesen Sprint abhandelten.

Projekt/Dokumentation:

- Insgesamt hatten wir zu wenig Zeit eingerechnet. Dies auch deshalb, weil der nötige schulische Hintergrund noch fehlte.
  Der Umstand, dass die Arbeit dem Schulstoff immer um ca. 4 Wochen vorauseilte und die nötigen Informationen im Netz recherchiert werden mussten, erschwerte das Zusammenarbeiten und die Arbeit enorm.
  Das Zusammenarbeiten insofern, dass nie jedes Mitglied der Gruppe auf dem selben Stand war und die Arbeit weil der Schulstoff oft nachträglich Fragen beantwortete, die wir uns über das Netz nicht beantworten konnten, und dadurch den Arbeitspunkt im Nachhinein nochmals überarbeiten mussten. Das generierte einen erheblichen Mehraufwand, der nicht nötig gewesen wäre.

8 Referenzen

THIS DOCUMENT IS TYPSET WITH

LaTeX