HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Web-Editor für Datenbanken

Bachelorarbeit

Abteilung Informatik
Hochschule für Technik Rapperswil

Frühjahrssemester 2013

Autoren:	Fabian Schweizer, Manuel Schweizer
Betreuer:	Dr. Daniel Keller
Projektpartner:	foryouandyourcustomers AG, Pfäffikon ZH
Experte:	Dr. Rudolf Mattmann
Gegenleser:	Prof. Hansjörg Huser

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Rapperswil, den 14. Juni 2013

Erklärung

Wir erklären hiermit,

- dass wir die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt haben, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben haben.
- das wir keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt haben.

Fabian Schweizer

Manuel Schweizer

Danksagung

Wir möchten uns bei all jenen bedanken, die uns während dieser Arbeit gefordert, unterstützt und motiviert haben.
Ganz besonders danken wir Herrn Dr. Daniel Keller, der mit seiner Idee dieses Projekt überhaupt erst möglich gemacht und während dem ganzen Semester betreut hat. Mit seinen kritischen Fragen und konstruktiver Kritik hat er uns zu Höchstleistungen animiert. Die gemeinsamen Mittagessen, bei welchen wir auch mal über unsere Zukunft, Gott und die Welt diskutiert haben, empfanden wir als sehr bereichernd. Herzlichen Dank!
Weiter möchten wir dem Experten, Dr. Rudolf Mattmann, und dem Gegenleser unserer Arbeit, Prof. Hansjörg Huser, für ihren Einsatz und die wertvollen Tipps danken.
Grosser Dank gilt zudem unseren Partnerinnen, die trotz Beteuerungen unsererseits, dass sich die Situation bald bessern werde, bis am Abgabetag deutlich vernachlässigt wurden und trotzdem grosses Verständnis zeigten.

Web-Editor für Datenbanken

Studiengang:	Informatik (I)
Semester:	FS 2013 (18.02.2013-15.09.2013)
Durchführung:	Bachelorarbeit, Studienarbeit

Fachrichtung:	Internet-Technologien und -Anwendungen
Institut:	Diverses
Gruppengrösse:	2 Studierende
Status:	zugewiesen

Verantwortlicher:	Keller, Daniel
Betreuer:	Keller, Daniel
Gegenleser:	Huser, Hansjörg
Experte:	Dr. Rudolf Mattmann, Mettler-Toledo
Industriepartner:	foryouandyourcustomers.com

Ausschreibung: Web-Editor für Datenbanken

In einer Firma enstehen viele strukturierte Daten, sie werden üblicherweise in Excel-Tabellen gehalten (Adressen und Telefonnummern der Mitarbeitenden, Liste der laufenden Projekte, Ferienplanung, Requirements in Projext X, ToDo Listen, etc.). Nun ist das herummailen von Excel-Tabellen oft keine so gute Idee: die angemailten Personen müssen etwas ausfüllen - währenddem jemand anders auch vielleicht gerade dasselbe ausfüllt - und wieder zurückschicken. Jetzt kommt die Zusammenführung der Daten, wo auch mal Konflikte auftreten. Zu guter Letzt muss die zusammengeführte Liste wieder an alle verschickt werden. Wenn jetzt jemand eine kleine Änderung angebracht haben möchte…

Viel einfachte wäre in vielen Szenarien der Web-basierte Zugriff auf eine zentrale Datenbank. Wenn es einfach wäre, schnell mal eine zentrale DB-Tabelle anzulegen, wo alle via Browser darauf zugreifen und ihre Daten eintragen könnten, dann wären viele Probleme gelöst.

Ansätze zu einer solchen Lösung sieht man in:
- Oracle Apex (http://apex.oracle.com)
- Google Docs (für Spreadsheets, nicht für DB-Tabellen)
- MS SharePoint (macht viel mehr als nur Tabellen anzeigen/editieren)
- zoho.com (http://www.zoho.com/creator/) macht auch viel mehr, ähnlich wie MS SharePoint
- phpMyAdmin als Primitiv-Lösung (wenig nutzerfreundlich, fehlende Rollen & Rechte…)

Ich habe selbst aus der Not heraus in 60 Arbeitsstunden etwa 1000 Zeilen PHP Code geschrieben, der die grundlegenden Operationen zur Verügung stellt. Aber es hat Einschränkungen, die ein Business User nicht annimmt: man muss jedes Mal PHP programieren, kein grafisches Frontend für DB Table creation, das Ding kann noch bei weitem nicht alles Gewünschte, der Code ist nicht sehr schön modular aufgebaut, u.a.m. Ein Prototyp halt.

Aufgabe:
Die ersten zwei Wochen arbeiten Sie sich in die Aufgabenstellung ein, d.h. Kundenbedürfnisse verstehen, existierende (open source) Lösungen

anschauen. Die restlichen 12 Wochen der Arbeit sollten Sie entweder auf der grünen Wiese eine neue Lösung programmieren, oder - besser noch - ein bereits existierendes open source Programm nehmen und das professionell erweitern.

Implementationssprache sollte Java oder C# sein, falls Sie es gut begründen können, dürfen Sie aber auch z.B. mit Ruby on Rails oder PHP/Symfony2 (oder etwas ganz anderem) arbeiten.

Voraussetzungen: Die Aufgabenstellung ist einerseits sehr einfach ("machen Sie einen Web-basierten DB Editor"), andererseits auch sehr anspruchsvoll: wenn es so einfach wäre, dann gäbe es bereits elegante, günstige Lösungen.

Für Student/innen, die gut programmieren können und die Herausforderungen schätzen.

Bewerbungen:

Gruppe:	Schweizer/Schweizer ✉
Einschreibung:	Bachelorarbeit
Status:	Arbeit zugewiesen (Priorität Student: 1)
Studierende:	- Schweizer, Manuel Lukas - Schweizer, Fabian
Kommentar:	Sehr geehrter Herr Keller Gemäss Besprechung bewerben wir uns für diesen Slot und freuen uns auf eine konstruktive Zusammenarbeit und spannende Zeit. Freundliche Grüsse Fabian und Manuel Schweizer

Fenster schliessen

Aufgabenstellung

Arbeit:

Bachelorarbeit FS 2013

Thema:

Web-Editor für Datenbanken

Auftraggeber und Betreuer

Industriepartner und Auftraggeber:

foryouandyourcustomers AG
Seestrasse 10
CH-8330 Pfäffikon ZH

Ansprechpartner Auftraggeber:

Dr. Daniel Keller

Betreuer HSR:

Dr. Daniel Keller

Ausgangslage

In einer Firma entstehen viele strukturierte Daten, sie werden üblicherweise in Excel-Tabellen gehalten (z.B. Adressen und Telefonnummern der Mitarbeitenden, Liste der laufenden Projekte, Ferienplanung, Requirements in Projekt X, ToDo-Listen, etc.). Das Versenden von Excel-Tabellen ist oft keine so gute Idee: die Empfänger müssen etwas ausfüllen – währenddem jemand anders vielleicht gleichzeitig dasselbe ausfüllt – und wieder zurückschicken. Es folgt die aufwändige Zusammenführung der Daten, wo auch mal Konflikte auftreten können. Zu guter Letzt muss die zusammengeführte Liste wieder an alle verschickt werden. Wenn jetzt jemand eine kleine Änderung angebracht haben möchte…

Viel einfacher wäre in vielen Szenarien der webbasierte Zugriff auf eine zentrale Datenbank. Wenn es einfach wäre, schnell mal eine zentrale DB-Tabelle anzulegen, wo alle via Browser darauf zugreifen und ihre Daten eintragen könnten, dann wären viele Probleme gelöst.

Ansätze zu einer solchen Lösung sieht man in:

- Oracle Apex (http://apex.oracle.com)
- Google Docs (für Spreadsheets, nicht für DB-Tabellen)
- MS SharePoint (macht viel mehr als nur Tabellen anzeigen/editieren)
- zoho.com (http://www.zoho.com/creator/) macht auch viel mehr, ähnlich wie MS SharePoint
- phpMyAdmin als Primitiv-Lösung (wenig nutzerfreundlich, fehlende Rollen & Rechte…)

Ziele der Arbeit

Die Studenten sollen eine Software-Lösung erstellen, welche auf einer Mehrheit der eingesetzten Webhosting-Umgebungen (LAMP-Stack) ohne Mühe installiert und betrieben werden kann. Besonderer Wert soll dabei auf die einfache Erstellung, Bearbeitung und Weitergabe von Webformularen gelegt werden.

Die folgenden Ziele sollen als Teil dieser Bachelorarbeit erreicht werden:

- Webapplikation, basierend auf einem verbreiteten PHP-Framework
- Benutzer ohne MySQL-Kenntnisse sollen in der Lage sein, einfache DB-Applikationen ohne fremde Hilfe zu erstellen (z.B. Adressverwaltung, Anmeldeformular)
- Mindestens zwei konkrete Anwendungen zu Demonstrationszwecken umsetzen

Durchführung

Mit dem HSR-Betreuer finden in der Regel wöchentliche Besprechungen statt. Zusätzliche Besprechungen sind nach Bedarf durch die Studierenden zu veranlassen.

Für die Durchführung der Arbeit ist ein Projektplan zu erstellen. Dabei ist auf einen kontinuierlichen und sichtbaren Arbeitsfortschritt zu achten. An Meilensteinen (gemäss Projektplan) sind einzelne Arbeitsresultate in vorläufigen Versionen zu präsentieren. Über die gezeigten Arbeitsresultate erhalten die Studierenden ein Feedback. Eine definitive Beurteilung erfolgt aufgrund der am Abgabetermin abgelieferten Dokumentation.

Es sollten hierbei die Hinweise aus dem abgegebenen Dokument „Tipps für die Strukturierung und Planung von Studien-, Diplom- und Bachelorarbeiten“ beachtet werden.

Dokumentation

Über diese Arbeit ist eine Dokumentation gemäss den Richtlinien der Abteilung Informatik zu verfassen. Die zu erstellenden Dokumente sind im Projektplan festzuhalten. Alle Dokumente sind nachzuführen, d.h. sie sollen den Stand der Arbeit bei der Abgabe in konsistenter Form dokumentieren. Die Dokumentation ist vollständig auf CD/DVD in 2 Exemplaren abzugeben.

Regeln und Termine

Im Weiteren gelten die allgemeinen Regeln zu Bachelor und Studienarbeiten „Allgemeine Infos Diplom-, Bachelor- und Studienarbeiten“:
https://www.hsr.ch/Allgemeine-Infos-Diplom-Bach.4418.0.html

Der Terminplan ist hier ersichtlich (HSR Intranet):
https://www.hsr.ch/Termine-Bachelor-und-Studiena.5142.0.html

Beurteilung

Eine erfolgreiche BA zählt 12 ECTS-Punkte pro Studierenden. Für 1 ECTS Punkt ist eine Arbeitsleistung von ca. 25 bis 30 Stunden budgetiert. Dies entspricht, gerechnet auf 15 Wochen à 2 Tagen mit 2 zusätzlichen Intensivwochen anfangs Semesterferien, ungefähr 17h Arbeit pro Woche.

Für die Beurteilung ist der HSR-Betreuer und ein externer Experte verantwortlich.

Rapperswil, den 25. Februar 2013

Dr. Daniel Keller

Institut für Software

Hochschule für Technik Rapperswil

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

Management Summary

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren: Fabian Schweizer, Manuel Schweizer
Betreuer: Dr. Daniel Keller
Projektpartner: foryouandyourcustomers AG, Pfäffikon ZH
Experte: Dr. Rudolf Mattmann
Gegenleser: Prof. Hansjörg Huser

Dokumentinformationen

Änderungsgeschichte

Datum	Version	Änderung	Autor
25.02.2013	0.1	Beginn	ms
02.06.2013	0.2	Abstract	ms
10.06.2013	1.0	Erste vollständige Version	ms
11.06.2013	1.1	Überarbeitung Management Summary	ms
12.06.2013	1.2	Überarbeitung	ms
13.06.2013	1.3	Review & Nachbearbeitung	fs
14.06.2013	1.4	Ergebnis	fs

Abstract

Tagtäglich werden in Firmen Excel-Files per E-Mail verschickt. Häufig werden die Empfänger dazu aufgefordert, bestimmte Felder auszufüllen und das bearbeitete Dokument an den Absender zurückzuschicken. Dabei ist der Massenversand von Excel-Tabellen zur Datenbeschaffung oft keine gute Idee: Das Zusammenführen verschiedener Datensätze birgt das Risiko von widersprüchlichen und unvollständigen Einträgen. Diese müssen dann in aufwändiger Handarbeit aufgelöst oder ergänzt werden.
Obwohl am Markt bereits zahlreiche Lösungsansätze existieren, schränken Komplexität der Benutzeroberfläche, Registrierungszwang oder eine zu starke Spezialisierung der Anwendung die Attraktivität für den durchschnittlichen Benutzer ein.
Ziel unserer Arbeit ist die Entwicklung einer Applikation, welche es dem Anwender ermöglicht, ein Webformular rasch, unkompliziert und ohne Registrierung zu erstellen.

Zu Beginn der Arbeit haben wir die Bedürfnisse der Zielgruppe mit den im Vorfeld gesammelten Ideen und Anwendungsfällen verglichen. Daraus sind entsprechend priorisierte Arbeitspakete entstanden, welche in einem Scrum-ähnlichen Vorgehen umgesetzt wurden. Gegen Ende wurde die intuitive Bedienung der Software mittels verschiedener Benutzertests verifiziert.
Die aus dieser Bachelorarbeit resultierende Applikation „Adgenero“ basiert auf dem Zend Framework 2 und wurde in der Skriptsprache PHP5 geschrieben. Für die vereinfachte Kommunikation mit der MySQL-Datenbank wurde mit Doctrine 2 ein OR-Mapper eingesetzt. Die zur Darstellung verwendeten Elemente stammen von Twitter Bootstrap. Auf Seite des Benutzers kommen HTML5, CSS und JavaScript zum Einsatz.

Adgenero erlaubt es dem Benutzer, strukturierte Informationen zu definieren, abzulegen, zu nutzen und anderen Personen zur Verfügung zu stellen. Er braucht sich dabei nicht um die zugrundeliegende Datenbankstruktur zu kümmern.
Das Resultat ist eine moderne Webapplikation, welche von jedermann kostenfrei und ohne Registrierung genutzt werden kann. Die Anwendung wird als hosted Service auf einem klassischen LAMP-Stack (Linux, Apache, MySQL, PHP) betrieben und ist dadurch von überall aus und zu jedem Zeitpunkt erreichbar.

Inhaltsverzeichnis

1 Management Summary

1.1 Ausgangslage

1.1.1 Problemstellung

Oft werden zur Datenerfassung Excel-Files an mehrere Empfänger geschickt. Diese werden aufgefordert Angaben wie Adressen, Ferientage und Abwesenheiten, Kontaktmöglichkeiten, Pendenzenlisten, Requirements - die Liste könnte beliebig fortgesetzt werden - zu überprüfen, aktualisieren und gegebenenfalls zu ergänzen. Das angepasste Dokument wird dann an den Absender zurückgeschickt und dieser muss die Datensätze aller Beteiligten in mühsamer Handarbeit zusammenführen. Dabei ist das Risiko von Konflikten oder unvollständigen Datensätzen gross.
Sind die Daten erst einmal in einem gemeinsamen Dokument konsolidiert, wird dieses wiederum an alle Empfänger zur Kontrolle verschickt. Im ungünstigsten Fall folgen weitere Korrekturrunden.

1.1.2 Bestehende Lösungsansätze

In den meisten Szenarien wäre der webbasierte Zugriff auf eine zentrale Datenbank für alle Beteiligten einfacher und würde viele der oben genannten Probleme beseitigen. Es existieren bereits zahlreiche Lösungsansätze, welche sich mit der gleichen Problemstellung befassen. Dennoch vermag keines dieser Software-Produkte den durchschnittlichen Benutzer von der Verwendung von Excel-Tabellen, zur Ablage strukturierter Daten, abzuhalten: Zu stark fokussieren diese auf der unkomplizierten Umsetzung bestimmter Anwendungsszenarien oder sind derart überladen, dass der potentielle Benutzer von der Komplexität und dem Funktionsumfang abgeschreckt wird.

1.1.3 Vision

Um den durchschnittlich versierten Excel-Benutzer von der Verwendung einer Alternativlösung zu überzeugen, müssen bestimmte Rahmenbedingungen gegeben sein:
So muss zum Beispiel der Zugriff auf die Daten einfach, zentral und jederzeit erfolgen können und die Benutzeroberfläche intuitiv zu bedienen sein. Zudem muss die Applikation generisch ausgelegt werden, damit sie den Benutzer nicht in den Anwendungsmöglichkeiten einschränkt.
Unsere Vision ist, dass jedermann kostenlos und zu jedem Zeitpunkt Zugriff auf eine Webapplikation hat, welche genau diese Rahmenbedingungen erfüllt. Daraus entstand ein Slogan, welcher genau diese Vorstellung verkörpert: einfach online verwalten.

1.1.4 Motivation

Aus dem Büroalltag und unserer Vereinstätigkeit kennen wir die Problematik, die beim Versand von Excel-Tabellen entstehen kann und waren selber bereits Leidtragende bei der stundenlangen Zusammenführung von Datensätzen.
Wir wollen mit unserer Applikation solche Situationen in Zukunft vermeiden und anderen Mitarbeitern, Vorstandsmitgliedern und sonstigen Freunden und Bekannten genau diesen Aufwand ersparen.

1.1.5 Ziele der Arbeit

Die nachfolgenden Ziele wurden zu Beginn der Arbeit festgelegt und definieren den minimalen Funktionsumfang der Software:

- Ohne Registrierung ein Formular erstellen
- Felder definieren (z.B. Textfeld, Auswahlfeld)
- Eigenschaften von Feldern definieren (z.B. Pflichtfeld, internes Feld)
- Felder und Feldeigenschaften bearbeiten
- Felder entfernen
- Direktlink zur Formularverwaltung, Manageransicht genannt
- Direktlink zum Formular, auch Gastansicht genannt
- Datensätze werden zentral abgelegt
- Daten sind exportierbar

1.1.6 Vergleichbare Produkte

Es existieren zahlreiche kostenlose und kommerzielle Applikationen, welche den gleichen Ansatz verfolgen, aber noch nicht die Bekanntheit und Verbreitung eines Microsoft Excels geniessen.
Folgend eine unvollständige Liste möglicher Konkurrenten:

- Google Forms
- Google Docs
- Doodle
- Oracle Apex
- MS Sharepoint
- FileMaker Bento

1.2 Vorgehen

1.2.1 Methode

Da die Aufgabenstellung zum Zeitpunkt der Ausschreibung sehr breit formuliert war, sammelten wir in einer ersten Phase verschiedene Ideen und Lösungsmöglichkeiten. Diese verglichen wir mit möglichen Anwendungsfällen. Daraus entstand dann die definitive Aufgabenstellung und unser Projektplan.
Aus den Bedürfnissen der Zielgruppe leiteten wir entsprechend priorisierte Arbeitspakete ab, welche in einer agilen, Scrum-ähnlichen Vorgehensweise umgesetzt wurden.
Die intuitive Bedienung der Software wurde mittels Benutzertests verifiziert. Diese Tests wurden gegen Ende der Arbeit durchgeführt und gaben Aufschluss über fehlende Funktionen und unlogische Abläufe innerhalb der Applikation.

1.2.2 Zielgruppe

Die Zielgruppe von Adgenero besteht hauptsächlich aus Privatpersonen und Vereinen.

1.2.3 Prototyp

Mit einem Prototyp wurden die wichtigsten architektonischen Grundlagen und der Ablauf der Applikation rudimentär implementiert. Dank der Erstellung des Prototyps war eine frühzeitige Erkennung von möglichen Problemen gegeben, was sich positiv auf die Entwicklung ausgewirkt hat.

1.2.4 Technologien

Es wurden aktuellste Webtechnologien eingesetzt. Insbesondere kamen die nachfolgend erwähnten Technologien zum Einsatz:

- HTML5
- Twitter Bootstrap
- Zend Framework 2
- Doctrine 2

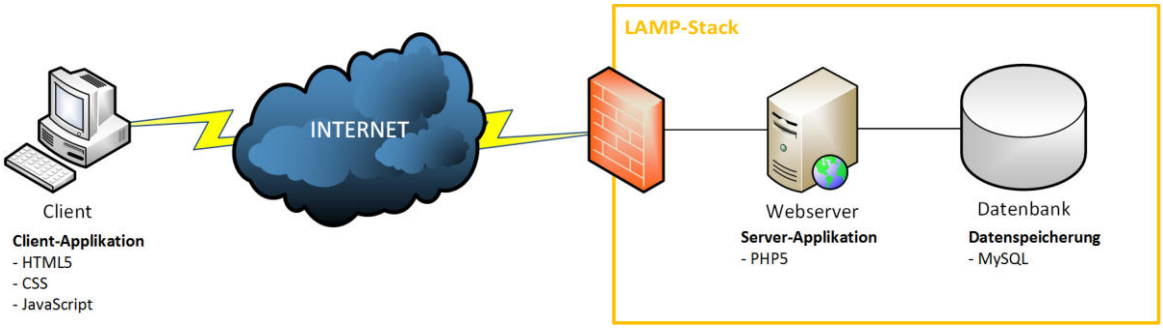


Abbildung 1: Physische Architektur

1.2.5 Infrastruktur

Im Rahmen dieser Bachelorarbeit standen zwei Arbeitsplätze mit jeweils einem Computer zur Verfügung. Zusätzlich zu diesen PCs wurden private Notebooks eingesetzt.

Folgende Software wurde im Zuge dieser Arbeit verwendet:

- Eclipse als Entwicklungsumgebung
- GIT als Versionierungssystem
- Jenkins als Buildserver
- Redmine zur Projektverwaltung
- Dropbox als Dokumentablage mit automatischer Versionierung
- Astah Community zur Erstellung von UML-Diagrammen
- Microsoft Office zur Erstellung von Texten und Bildern

Von der Hochschule wurde ein virtueller Server zur Verfügung gestellt, auf welchem das Versionierungssystem und der Buildserver installiert waren.

1.3 Ergebnis

Entstanden ist eine moderne und einfach zu bedienende Webapplikation, deren Augenmerk klar auf Usability gelegt wurde. Die Bedienung ist schlicht gehalten und kommt mit wenigen Worten aus.

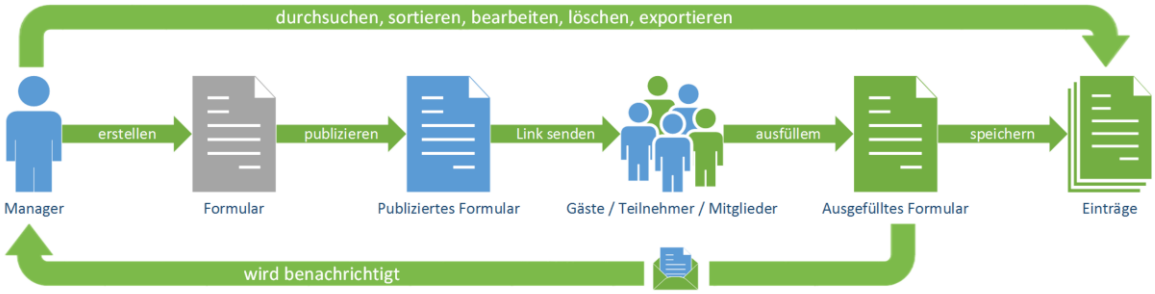


Abbildung 2: Workflow

1.3.1 Erreichte Ziele

Die Ziele wurden vollumfänglich erreicht. Darüber hinaus wurden fast doppelt so viele Funktionen umgesetzt, als ursprünglich in den Anforderungen festgelegt.

Nutzbar ohne Registrierung
Ein spezieller Mechanismus erlaubt die unkomplizierte Nutzung der Anwendung ohne vorgängige Registrierung. Dies macht die Applikation insbesondere für weniger versierte und erstmalige Nutzer attraktiv.

Moderne Architektur
Die entwickelte Webapplikation wurde in vier Schichten, welche wiederum in verschiedene Partitionen unterteilt sind, gegliedert. Damit wurde eine tiefe Kopplung zwischen den Software-Komponenten erreicht. Dadurch ist die einfache Anpassbarkeit und Erweiterbarkeit der Applikation in der Zukunft gewährleistet.

Responsive Layout
Die bei der Benutzeroberfläche eingesetzten Komponenten ermöglichen eine flexible Nutzung der Applikation bei unterschiedlichen Bildschirmgrössen und -auflösungen. Dadurch wurde die Grundlage für eine mühelose Anpassung der Software für mobile Endgeräte geschaffen.

1.3.2 Möglichkeiten

Es wurde eine Vielzahl von möglichen Anwendungsfällen abgedeckt. Der nächsten Einladung zu einem Grillfest oder der Mitgliederverwaltung des Vereins steht damit nichts mehr im Weg.

1.4 Ausblick

Obwohl wir deutlich mehr als den Pflichtumfang und sogar etliche Funktionen implementiert haben, welche in den ersten Benutzertests vermisst wurden, gibt es noch zahlreiche Erweiterungs- und Optimierungsmöglichkeiten.

1.4.1 Mögliche Erweiterungen

Zusätzliche Feldtypen
Im Rahmen der Bachelorarbeit haben wir erste Feldtypen wie Text- und Auswahlfelder umgesetzt. Zusätzliche Feldtypen wie die Möglichkeit der Mehrfachauswahl oder Textfelder, welche eine spezielle Validierung benötigen (z.B. ISBN), würden die Anwendungsmöglichkeiten der Software erweitern.

Benutzerverwaltung
Ungeachtet der Tatsache, dass unsere Applikation ohne Registrierung vollumfänglich genutzt werden kann, wird früher oder später der Wunsch nach einer Benutzerverwaltung, mit entsprechenden Möglichkeiten zur Vergabe und Einschränkung von Rechten, aufkommen.

Mobile-App
Mit zunehmend mobiler Nutzung des Internets wächst auch das Bedürfnis, Daten unterwegs zu erfassen. Mit einer nativen App für iOS und Android könnte dieser Nachfrage entsprochen werden.

1.4.2 Optimierungen

Geschwindigkeit erhöhen
In dieser Arbeit wurde kein Wert auf die Verbesserung der Geschwindigkeit gelegt. Mit Optimierungen im Code und der Einführung von Caching könnte die Performance deutlich gesteigert werden.

Webseite
Der erste Eindruck zählt. Unsere Webseite wurde aus finanziellen Gründen nicht professionell gestaltet und weckt deshalb zu wenig Emotionen bei den Besuchern.

Übersetzungen
Nicht alle Fehlermeldungen und Hinweise konnten in der vorgegebenen Zeit übersetzt werden. Um die Applikation einem breiten Nutzerkreis zur Verfügung zu stellen, bedarf es einiger Ergänzungen in den jeweiligen Sprachdateien.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

Projektplan

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren:	Fabian Schweizer, Manuel Schweizer
Betreuer:	Dr. Daniel Keller
Projektpartner:	foryouandyourcustomers AG, Pfäffikon ZH
Experte:	Dr. Rudolf Mattmann
Gegenleser:	Prof. Hansjörg Huser

Dokumentinformationen

Änderungsgeschichte

Datum	Version	Änderung	Autor
25.02.2013	0.1	Beginn Projektplan	ms
10.03.2013	0.2	Überarbeitung Projektplan	ms
11.06.2013	1.0	Erster Release	ms
12.06.2013	1.1	Review	fs
13.06.2013	1.2	Featurebeschreibung	ms
13.06.2013	1.3	Qualitätssicherung überarbeitet	ms
14.06.2013	1.4	Referenzen korrigiert	fs
14.06.2013	1.5	Last Review	ms

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument gibt Aufschluss über den geplanten Verlauf des Projekts Adgenero. Der Projektplan ist das wichtigste Planungsinstrument. Hier werden unter anderem die phasen- und iterationsübergreifenden Ziele, Meilensteine und Organisationsstrukturen festgelegt.

1.2 Gültigkeitsbereich

Das Dokument ist während der gesamten Projektdauer gültig, wird periodisch aktualisiert und ergänzt.

1.3 Referenzen

Die folgenden Dokumente werden in diesem Projektplan referenziert:

- 01-Aufgabenstellung/Aufgabenstellung.pdf
- 03-Projektplan/Risikoanalyse.pdf
- 10-Glossar/Glossar.pdf

1.4 Aufteilung

Das vorliegende Dokument beginnt nach einer kurzen Übersicht mit einer Beschreibung der Projektorganisation. Es folgen die Iterationsplanung und das Risikomanagement. Schliesslich wird näher auf die Infrastruktur und Qualitätsmassnahmen eingegangen.

2 Projekteckwerte

Das folgende Kapitel soll eine Übersicht über das Projekt Adgenero geben. Dabei werden Zweck und Ziele des Projektes aufgezeigt und die Motivation erläutert. Schlussendlich wird der Umfang der Arbeit festgelegt und einzelne Funktionen definiert.

2.1 Zweck und Ziel

2.1.1 Projektziele

Gemäss Aufgabenstellung des Betreuers und Anforderungen des Auftraggebers soll „eine Software-Lösung erstellt werden, welche auf einer Mehrheit der eingesetzten Webhosting-Umgebungen (LAMP-Stack) ohne Mühe installiert und betrieben werden kann. Besonderer Wert soll dabei auf die einfache Erstellung, Bearbeitung und Weitergabe von Webformularen gelegt werden.“

2.1.2 Persönliche Ziele

Praxis
Wir wollen das an der HSR gelernte Wissen in diesem Projekt praktisch anwenden. Dies betrifft sowohl die Planung wie auch die Realisierung des Projekts.

Machbarkeit
Da die Idee auch über die Dauer des Studiums hinaus Potential hat, soll diese Bachelorarbeit als Machbarkeitsstudie für eine mögliche kommerzielle Nutzung zu einem späteren Zeitpunkt dienen.

2.2 Motivation

Aus dem Alltag und unserem Umfeld kennen wir die Problematik, die beim Versand von Excel-Tabellen entstehen kann und waren selber bereits Leidtragende bei der Zusammenführung von Datensätzen.
Mit Adgenero wollen wir eine Plattform schaffen, welche ohne Kostenfolge und für jedermann frei zugänglich ist und diesem Problem entgegenwirkt. Der Fokus liegt dabei auf der einfachen Handhabung. Die Zielgruppe sind weniger versierte Benutzer.

2.3 Lieferumfang

Nach Abschluss des Projekts steht folgender Lieferumfang zur Verfügung:

- Webapplikation als hosted Service
- Projektdokumentation
- Entwicklungsanleitung
- Sourcecode

2.4 Ideensammlung

2.4.1 Grundidee

Formular erstellen
Die Webapplikation soll dem Benutzer die Möglichkeit bieten, einfach ein Webformular zu erstellen und dieses weiter zu geben. Dafür ist kein Login und keine Registrierung notwendig.

Formular ausfüllen
Der Benutzer kann das Formular ausfüllen, dieses wird validiert und die Angaben in der Datenbank gespeichert.

2.4.2 Brainstorming

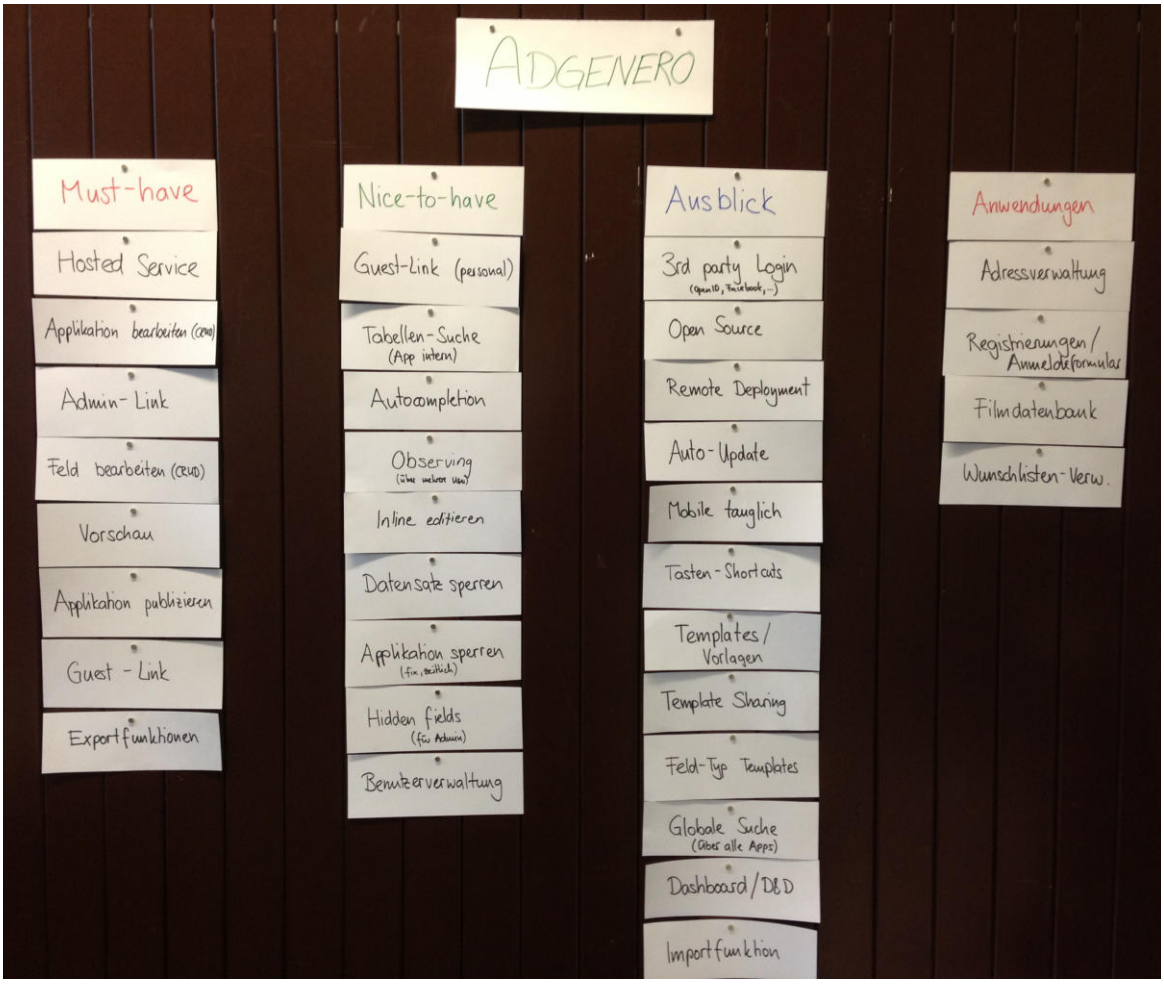


Abbildung 1: Taskboard

2.5 Funktionen

Die Applikation vereinfacht das Erstellen von Formularen und beseitigt Probleme und Risiken, die bei der Datenerfassung mittels Excel-Tabellen entstehen.

Funktionalitäten

- Ohne Registrierung ein Formular erstellen
- Felder definieren (z.B. Textfeld, Auswahlfeld)
- Eigenschaften von Feldern definieren (z.B. Pflichtfeld, internes Feld)
- Felder und Feldeigenschaften bearbeiten
- Felder entfernen
- Direktlink zur Formularverwaltung, Manageransicht genannt
- Direktlink zum Formular, auch Gastansicht genannt
- Datensätze werden zentral abgelegt
- Daten sind exportierbar

2.6 Annahmen und Einschränkungen

Es wird davon ausgegangen, dass jedes Teammitglied pro Woche ungefähr 18 Stunden in dieses Projekt investiert; dazu kommen zwei Intensivwochen zu Beginn der Semesterferien. Es werden somit insgesamt rund 720 Stunden Arbeit geleistet.
Das Projekt muss spätestens am 14. Juni 2013 um 12:00 Uhr abgegeben werden.

2.7 Verwandte Anwendungen

Folgende Produkte bieten Lösungsansätze, welche den Versand von Excel-Sheets zur Datenerfassung ebenfalls erübrigen:

- Google Forms
- Google Docs
- Oracle Apex
- MS Sharepoint
- FileMaker Bento

3 Projektorganisation

3.1 Organisationsstruktur

Das Projektteam setzt sich aus zwei Informatik-Studenten zusammen. Die nachfolgend aufgeführten Teammitglieder sind einander gleichgestellt.

Fabian Schweizer

Manuel Schweizer

Der Fokus des jeweiligen Teammitglieds ist wie folgt definiert:

- Fabian Schweizer ist verantwortlich für den Backend-Bereich der Applikation und sorgt dort für eine gute Testabdeckung und Codequalität.
- Manuel Schweizer ist verantwortlich für den Frontend-Bereich der Applikation und sorgt dort dafür, dass sich die Applikation auch von wenig versierten Usern einfach bedienen lässt.

3.2 Externe Schnittstellen

Das Projekt wird von Herrn Dr. Daniel Keller vom Institut für Software an der Hochschule Rapperswil betreut. Er ist zugleich Auftraggeber in seiner Funktion als CTO der foryouandyourcustomers AG in Pfäffikon ZH.

4 Management-Abläufe

4.1 Zeitliche Planung

Dieses Projekt wird nach dem Unified Process gemäss Larman geplant. Die Umsetzung der einzelnen Arbeitspakete soll in agiler, Scrum-ähnlicher Vorgehensweise erfolgen.

4.2 Phasen und Iterationen

Gemäss dem UP wird das Projekt in folgende vier Phasen unterteilt:

- Inception
- Elaboration
- Construction
- Transition

4.2.1 Phasenbeschreibung

Inception
Alle Teammitglieder haben eine konkrete Vorstellung des Produktes, welches abgeliefert werden soll. Ausserdem geht es darum, erste grundlegende Entscheidungen in den Bereichen Funktionalität und Architektur in Zusammenarbeit mit dem Auftraggeber zu fällen. Die Aufgabenstellung wird finalisiert, Ideen für die Projektumsetzung werden gesammelt und die Entwicklungswerkzeuge vorbereitet.

Elaboration
Zentrale Punkte dieser Iteration sind: Domainmodell erstellen, Use Cases beschreiben und Grundlagen der Architektur definieren. Sobald wie möglich wird mit der Entwicklung des Prototyps begonnen. Dabei soll der Fokus zu Beginn auf die unteren Architekturebenen gelegt werden.
Ziel dieser Phase ist es, einen funktionierenden Prototyp zu haben.

Construction
Zu Beginn der Construction-Phase soll in einem Scrum-ähnlichen Vorgehen zuerst ein Backlog mit Features erstellt werden. Diese werden dann den einzelnen Construction-Iterationen zugewiesen und abgearbeitet. Dies erlaubt eine agile Vorgehensweise während der Umsetzung.

Transition
Da es nicht das Ziel dieses Projektes ist, die Software in die Produktion zu überführen, dient diese Iteration primär dazu, die Dokumentation abzuschliessen und den Code aufzuräumen.

4.2.2 Iterationsplanung und Meilensteine

Geplant ist, das Projekt wie in nachfolgender Abbildung aufgezeigt in Iterationen zu unterteilen. Das Ende jeder Phase stellt dabei jeweils einen Meilenstein innerhalb des Projekts dar.

- **Inception:** Eine zweiwöchige Iteration
- **Elaboration:** Eine dreiwöchige Iteration
- **Construction:** Sechs zweiwöchige Iterationen
- **Transition:** Eine einwöchige Iteration

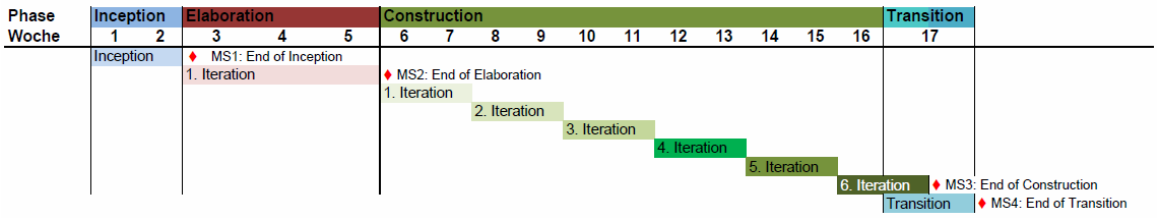


Abbildung 2: Iterationsplanung

5 Risikomanagement

Informationen zu Risiken wurden in ein separates Dokument ausgelagert. Es enthält eine Beschreibung der Risiken, Kosten und Massnahmen. Das Dokument wird laufend aktualisiert und ergänzt.

6 Infrastruktur

6.1 Entwicklung

Jedes Projektmitglied arbeitet mit dem eigenen Computer respektive Notebook.

Folgende Software kommt für die Umsetzung dieser Arbeit zum Einsatz:

- Eclipse (Entwicklungsumgebung)
- GIT (Versionsverwaltung)
- Redmine (Projektverwaltung)
- Dropbox (Dokumentverwaltung)
- Astah Community (Diagrammsoftware)
- LaTeX (Textsatzsystem)

6.2 Produktion

Das Projekt wird bei der Firma Cyon GmbH in Basel betrieben.

Folgende Software kommt beim Betrieb zum Einsatz:

- Linux
- Apache2
- PHP5
- MySQL5
- phpMyAdmin

7 Qualitätsmassnahmen

7.1 Dokumentation

Sämtliche Dokumente werden in der Team-Dropbox abgelegt. Die Qualitätssicherung wird gewährleistet durch die automatische Versionierung der Dokumente sowie mit Reviews durch die Teammitglieder.
Als Vorlage dient ein Template, das zur Erstellung der Dokumente verwendet werden muss. Dies gewährleistet ein einheitliches Design.

7.2 Projektmanagement

Für das Management des Projektes wird Redmine eingesetzt. Sämtliche Arbeiten werden als Tickets erfasst und der geleistete Aufwand direkt dort eingetragen. Bugs werden ebenfalls erfasst und der für diesen Teil verantwortlichen Person zugewiesen.
Dem Betreuer steht ein separater Zugang zur Verfügung.

7.3 Entwicklung

Für die Versionsverwaltung wird ein GIT-Server eingesetzt. GIT lässt sich direkt in Betriebssysteme und in eine Vielzahl von IDEs integrieren. Ein Repository kann auf dem von der HSR zur Verfügung gestellten, virtuellen Server eingerichtet werden.

7.3.1 Vorgehen

Zuerst soll ein Prototyp entwickelt werden, welcher die wichtigsten Funktionen implementiert. Ist dieser soweit funktionstüchtig wird das Endprodukt von Grund auf neu aufgebaut.
Bei der Entwicklung des Endprodukts wird jeweils neue Funktionalität zur bestehenden hinzugefügt. Dabei soll der Ablauf der Applikation ausschlaggebend für die Reihenfolge der Funktionalität sein.

Die Benutzeroberfläche wird parallel zur Logik entwickelt. Als Vorlage für das UI werden vereinfachte GUI-Skizzen erstellt.

Das Testen der Applikation erfolgt jeweils vor Commits auf Unit Test Ebene und immer nach Abschluss eines Features auf Funktionsebene.

7.3.2 Unit Testing

Für die Implementation werden phpUnit-Tests geschrieben. Es wird nicht nach Test Driven Development (TDD) gearbeitet sondern folgender Ablauf verwendet:

1. Die Funktionalität wird implementiert.
2. Gegebenenfalls wird ein Test zur Prüfung der Funktion geschrieben.
3. Sämtliche Tests werden ausgeführt.
4. Code wird korrigiert bis Schritt 3 erfolgreich abläuft.

7.3.3 Systemtests

Nach jeder Iteration werden Systemtests durchgeführt. Der letzte Systemtest erfolgt am Ende der Construction Phase und wird ausführlich dokumentiert.

7.3.4 Benutzertests

Gegen Ende der Construction Phase werden Benutzertests mit verschiedenen Personen durchgeführt, um die intuitive Bedienung der Software zu testen und Feedback zu erhalten. Die Erkenntnisse aus diesen Benutzertests werden schriftlich festgehalten.

7.3.5 Code Style Guidelines

Für dieses Projekt halten wir uns an die Zend Framework 2 Coding Standards.

8 Verzeichnisse

8.1 Abbildungsverzeichnis

Risiko Kosten Template

Risiko Analyse
Projektname: Adgenero Projektmanager: Fabian Schweizer, Manuel Schweizer Datum Kalkulation: 21.03.2013

Risiko Bewertungen

Risk ID	Risiko	Auswirkung	Massnahme	Kosten der Massnahmen in Mann-Stunden	Max. Schaden in Mann-Stunden	Wahrschein-lichkeit des Eintreffens	Gewichteter Schaden in Mann-Stunden	Priorität
R01	Ausfall eines Teammitglieds für eine Woche	Mehr Arbeit für andere Teammitglieder	Kompensation des Zeitverlusts durch andere Teammitglieder	20	20	0.5	8.5	Mittel
R02	Zerwürfnis im Team	Moral- und Produktivitätsverlust, weniger Umfang	Daniel Keller als Vermittler einsetzen, um Differenzen aus dem Weg zu schaffen	5	30	0.1	6	Tief
R03	Ausfall der HSR-Infrastruktur (Server)	GIT, Jenkins und Remine nicht mehr verfügbar. Allenfalls Datenverlust	DB-Dump per E-Mail (jeweils Mo und Do)	5	50	0.05	0.8	Hoch
R04	Probleme mit Webtechnologien (ZF2, JavaScript, HTML5)	Zusätzlicher Aufwand für Einarbeitung, Hilfeleistung muss bezogen werden	Prototyping der kritischen Funktionen	50	50	0.75	9.6	Mittel
R05	Probleme mit Cyon-Infrastruktur (Server, E-Mailservices)	Präsentation nicht möglich	Lokale Installation als Backup für Schlusspräsentation	5	20	0.2	6.4	Hoch
	Total Kosten in Arbeitspaketen enthalten			85				
	Total Rückstellungen						31.3	

Privates Notebook Team-Mitglied (Diebstahl, Defekt)
Build-Server für PHP
Technische Probleme bei Präsentation
- Kein Internet
- Probleme mit Hardware

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

Anforderungsspezifikation

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren: Fabian Schweizer, Manuel Schweizer
Betreuer: Dr. Daniel Keller
Projektpartner: foryouandyourcustomers AG, Pfäffikon ZH
Experte: Dr. Rudolf Mattmann
Gegenleser: Prof. Hansjörg Huser

Dokumentinformationen

Änderungsgeschichte

Datum	Version	Änderung	Autor
25.02.2013	0.1	Erstellen des Templates	ms
13.05.2013	0.2	Einfügen der Use Cases	fs
10.06.2013	1.0	Erste vollständige Version	ms
12.06.2013	1.1	Überarbeitung	ms
12.06.2013	1.2	NF aus LaTeX portiert	fs
14.06.2013	1.3	Last Review	ms

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument spezifiziert die Anforderungen für das Projekt Adgenero. Es dient zur vertieften Analyse der Benutzeransprüche und legt die definierten Use Cases fest.

1.2 Gültigkeitsbereich

Das Dokument ist während der gesamten Projektdauer gültig und wird periodisch aktualisiert.

1.3 Referenzen

Die folgenden Dokumente werden in diesen Anforderungsspezifikationen referenziert:

- 01-Aufgabenstellung/Aufgabenstellung.pdf
- 03-Projektplan/Projektplan.pdf
- 10-Glossar/Glossar.pdf

1.4 Aufteilung

Das vorliegende Dokument beginnt mit einer allgemeinen Beschreibung der Funktionsumfanges. Danach werden die Use Cases erläutert und schliesslich wird auf weitere, spezifische Anforderungen eingegangen.

2 Allgemein

Adgenero ermöglicht es Firmen, Vereinen und Privatpersonen auf intuitive Weise, Formulare zu erstellen und einem definierten Kreis von Benutzern zur Verfügung zu stellen. Dabei müssen sie sich nicht um technische Details wie die zugrundeliegende Datenbank kümmern.

Die Anwendung wird als Webapplikation umgesetzt und ist dadurch mit aktuellen Browsern von überall aus erreichbar. Das Webinterface ist responsive und bietet dem Nutzer damit eine höchstmögliche Freiheit bei der Wahl seiner Bildschirm-, bzw. Fenstergrösse.
Die Umsetzung als native App für Android und iOS ist zu einem späteren Zeitpunkt denkbar und würde die mobile Tauglichkeit weiter erhöhen.

2.1 Must-Have Funktionalität

Der Pflichtumfang wurde gemeinsam mit dem Betreuer festgelegt und umfasst die folgenden must-have Features.

Ziel	Beschreibung
Applikation bearbeiten (CRUD)	Ein neues Formular kann angelegt, der Titel und die Beschreibung editiert und das Formular wieder gelöscht werden.
Manager-Link	Der Manager des Formulars erhält einen eindeutigen Link, mit welchem er Zugriff auf die Verwaltung des Formulars hat.
Feld bearbeiten (CRUD)	Zu einem Formular können Felder hinzugefügt, angepasst und gelöscht werden.
Vorschau	Vor der Publikation eines Formulars sieht der Manager eine Vorschau des Formulars und kann diese bestätigen.
Applikation publizieren	Das Formular kann veröffentlicht oder widerrufen werden.
Gast-Link	Der Manager erhält einen Link zur leeren Formularmaske. Diesen stellt er den Gästen zur Verfügung, welche dann via diesem Link das Formular ausfüllen können.
Formular ausfüllen	Das Formular kann von einem Gast ausgefüllt und abgeschickt werden. Vorher werden die Angaben durch das System validiert.
Exportfunktion	Die Datensätze können zur Weiterverarbeitung in anderen Programmen exportiert werden.

2.2 Nice-To-Have Funktionalität

Features, welche optional umgesetzt werden können, werden nachfolgend beschrieben.

Ziel	**Beschreibung**
Datensatz bearbeiten	Der Manager kann einen Datensatz bearbeiten und löschen.
Datensätze durchsuchen	Der Manager kann alle Datensätze zu einem bestimmten Formular durchsuchen, bzw. filtern.
Interne Felder	Der Manager hat die Möglichkeit, ein Feld anzulegen, welches dem Gast nicht angezeigt wird, d.h. lediglich für interne Zwecke verwendet wird.
Datensatz drucken	Der Manager kann einen bestimmten Datensatz in einer druckoptimierten Ansicht öffnen und drucken.
Datensatz sperren	Der Manager kann einen bestimmten Datensatz vor ungewollter Modifikation schützen, indem er diesen sperren kann.
Applikation sperren	Der Manager kann ein Formular vor ungewollter Modifikation schützen, indem er dieses sperren kann.
Datensatz nachträglich editieren	Der Gast kann seinen Datensatz dank einem personalisierten Link nachträglich editieren.
Inline editieren	Der Manager kann einen Eintrag innerhalb eines Datensatzes direkt in der Tabelle editieren.

2.3 Benutzer

Die Zielgruppe von Adgenero besteht in erster Linie aus Privatpersonen und Vereinen, welche bisher auf den Versand von Excel-Tabellen zur Datenerfassung zurückgegriffen haben. In einer späteren Ausbauphase ist es denkbar, dass auch Firmen mit höheren Ansprüchen an die Sicherheit (z.B. zwingendes Login, nur bestimmte Empfänger) und die Abbildung komplexer Vorgänge (z.B. wenn-dann Felder) angesprochen werden, indem entsprechende Features abgebildet werden. Der Fokus ist und bleibt aber der nicht versierte Anwender.

2.4 Mögliche Anwendungsszenarien

Dank der Realisierung verschiedener Feldtypen sind zahlreiche Anwendungsszenarien denkbar:

- Mitglieder- und Adressverwaltung (in einem Verein, privat, geschäftlich)
- Anmeldeformular für einen Event (Pfadilager, Geburtstagsfest, Hochzeit)
- Filmdatenbank (z.B. Titel, Medium, Erscheinungsjahr, Regisseur, Dauer)
- Bugreports (z.B. Problembeschreibung, Komponente, Reporter, Assignee)

2.5 Einschränkungen

Adgenero ist nur übers Internet erreichbar und setzt deshalb eine permanente Internetverbindung voraus.
Zudem legt die Applikation, bedingt durch die gewählte Art und Weise wie ein Formular erstellt und der Link dazu weitergegeben werden kann, bewusst keinen speziellen Wert auf Sicherheit. Da die URL zur Manageransicht aber selbst bei aktiver SSL-Verbindung unverschlüsselt übertragen würde, ist es mit der gewählten Architektur und ohne Registrierungspflicht praktisch ausgeschlossen, ein höheres Mass an Sicherheit zu bieten.

2.6 Annahmen

Wir gehen davon aus, dass der Benutzer über einen aktuellen und modernen Browser verfügt.

3 Use Cases

3.1 Farbdefinition

Die im nachfolgenden Diagramm definierten Farben sind während des ganzen Projekts gültig und werden für alle Diagramme gleich gehandhabt.

3.2 Use Case Diagramm

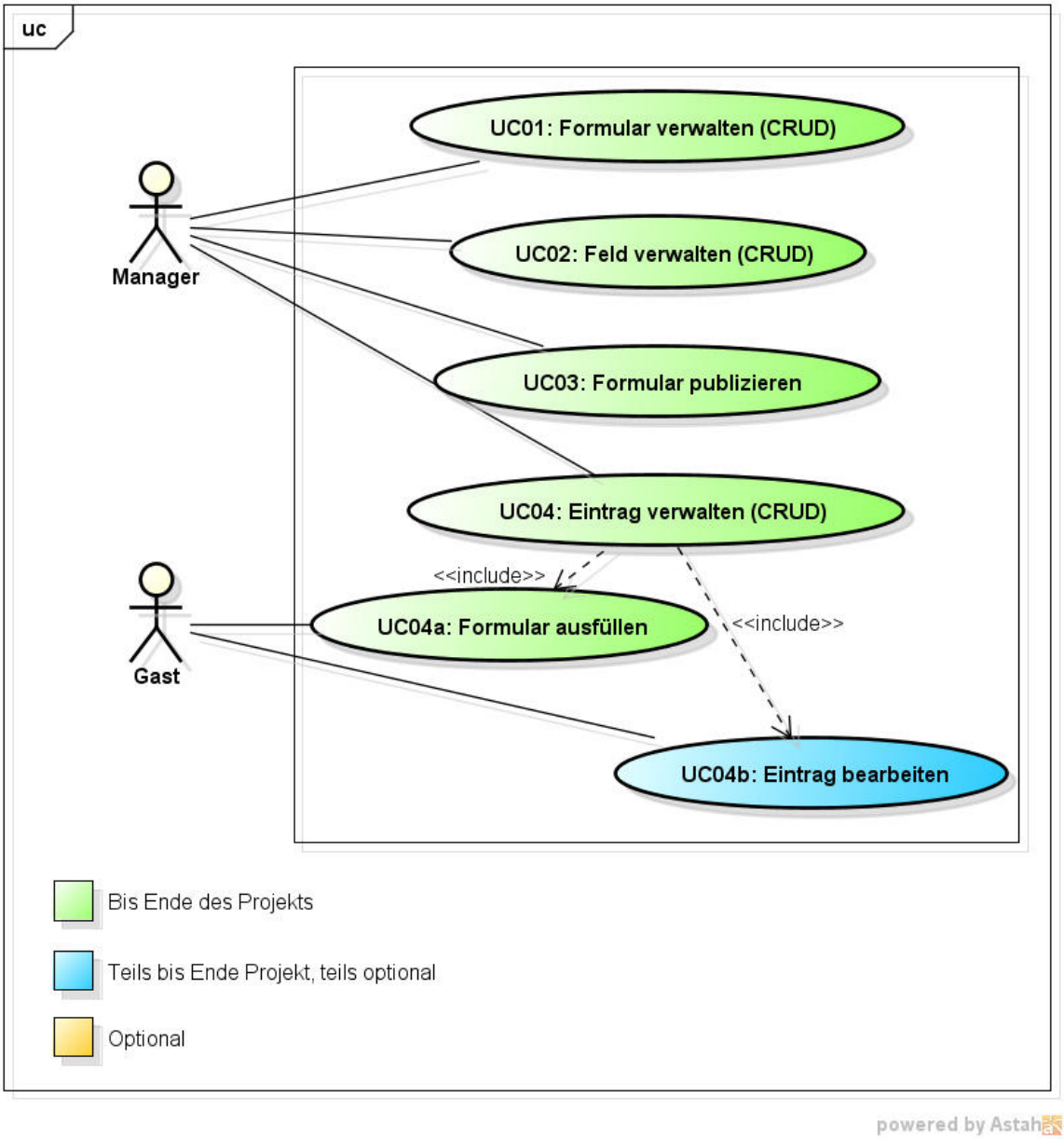


Abbildung 1: Use Case Diagramm

3.3 Aktoren & Stakeholder

Manager
Person, welche über ein eigenes Formular (App) verfügt. Der Manager legt das Formular an, verwaltet es und hat Zugriff auf sämtliche zugehörigen Einträge (Record). Das ausfüllbare Formular wird vom Manager mittels Gast-URL an die Gäste gegeben.

Gast
Aussenstehende Person, welche über einen Gästelink verfügt (den diese Person von einem Manager erhalten hat).

3.4 Beschreibungen

Die Beschreibung der Use Cases erfolgt im Format „fully dressed“.

3.4.1 UC01: Formular verwalten (CRUD)

Use Case Name	UC01: Formular verwalten (CRUD)
Primärakteur	Manager
Stakeholder und Interessen	Manager will mit minimalem Aufwand ein Formular erstellen oder bearbeiten.
Vorbedingungen	• System muss konfiguriert sein. • Manager ist in der Applikation.
Nachbedingungen	• Manager hat Formular erstellt oder bearbeitet.
Standardablauf	1. Manager wählt die entsprechende Funktion des Systems um ein Formular zu erstellen. 2. Manager gibt Werte für Formular und Kontaktdaten an. 3. Manager verwaltet Felder des Formulars (*UC02: Feld verwalten (CRUD)*). 4. Manager betrachtet Vorschau und publiziert das Formular (*UC03: Formular publizieren*). 5. Manager verwaltet Einträge (*UC04: Eintrag verwalten (CRUD)*) oder bearbeitet Felder (*UC02: Feld verwalten (CRUD)*)
Erweiterungen	• Jederzeit, wenn das System einen Fehler hat. 1 System informiert über den Fehler. 2 System versucht früheren Zustand wiederherzustellen. 1. Manager wählt die entsprechende Funktion des Systems, um ein Formular zu a. bearbeiten. b. löschen. Danach ist der Use Case beendet. 2. Manager bricht Vorgang ab. Danach ist der Use Case beendet. 4. Manager bricht Vorschau ab und publiziert Formular nicht. Kehre zurück zu Schritt 3 des Standardablaufs.
Spezielle Anforderungen	• Mögliche Touchscreen-UI für Tablets & Mobile (grosse Schaltflächen). • Mögliche Internationalisierung der Sprache.
Eintrittshäufigkeit	Durchschnittlich

3.4.2 UC02: Feld verwalten (CRUD)

Use Case Name	UC02: Feld verwalten (CRUD)
Primärakteur	Manager
Stakeholder und Interessen	Manager will mit minimalem Aufwand ein Feld eines Formulars erstellen oder bearbeiten.
Vorbedingungen	• System muss konfiguriert sein. • Formular muss vorhanden sein. • Manager ist in der Applikation. • Manager ist in der Verwaltungsansicht.
Nachbedingungen	• Manager hat Feld im Formular erstellt oder bearbeitet.
Standardablauf	1. Manager wählt die entsprechende Funktion des Systems um ein Feld zu erstellen. 2. Manager gibt Werte des Felds an.
Erweiterungen	• Jederzeit, wenn das System einen Fehler hat. 1 System informiert über den Fehler. 2 System versucht früheren Zustand wiederherzustellen. 1. Manager wählt die entsprechende Funktion des Systems um ein Formular zu… a. … bearbeiten. b. … löschen. Danach ist der Use Case beendet. 2. Manager bricht Vorgang ab. Danach ist der Use Case beendet.
Spezielle Anforderungen	• Mögliche Touchscreen-UI für Tablets & Mobile (grosse Schaltflächen). • Mögliche Internationalisierung der Sprache.
Eintrittshäufigkeit	Häufig

3.4.3 UC03: Formular publizieren

Use Case Name	UC03: Formular publizieren
Primärakteur	Manager
Stakeholder und Interessen	Manager will mit minimalem Aufwand ein Formular publizieren.
Vorbedingungen	• System muss konfiguriert sein. • Formular muss vorhanden sein. • Manager ist in der Applikation. • Manager ist in der Verwaltungsansicht.
Nachbedingungen	• Manager hat Formular publiziert.
Standardablauf	1. Manager wählt die entsprechende Funktion des Systems um ein Formular zu publizieren.
Erweiterungen	• Jederzeit, wenn das System einen Fehler hat. 1 System informiert über den Fehler. 2 System versucht früheren Zustand wiederherzustellen. 1. Manager bricht Vorgang ab. Danach ist der Use Case beendet.
Spezielle Anforderungen	• Mögliche Touchscreen-UI für Tablets & Mobile (grosse Schaltflächen). • Mögliche Internationalisierung der Sprache.
Eintrittshäufigkeit	Durchschnittlich

3.4.4 UC04: Eintrag verwalten (CRUD)

Use Case Name	UC04: Eintrag verwalten (CRUD)
Primärakteur	Manager
Stakeholder und Interessen	Manager will mit minimalem Aufwand einen Eintrag erstellen oder bearbeiten.
Vorbedingungen	• System muss konfiguriert sein. • Formular muss vorhanden sein. • Manager ist in der Applikation. • Manager ist in der Verwaltungsansicht.
Nachbedingungen	• Eintrag wurde erstellt oder bearbeitet.
Standardablauf	1. Manager wählt die entsprechende Funktion des Systems um einen Eintrag zu erstellen. 2. Siehe *UC04a: Formular ausfüllen*.
Erweiterungen	• Jederzeit, wenn das System einen Fehler hat. 1 System informiert über den Fehler. 2 System versucht früheren Zustand wiederherzustellen. 1. Manager wählt die entsprechende Funktion des Systems um einen Eintrag zu… a. … bearbeiten. b. … löschen. Danach ist der Use Case beendet.
Spezielle Anforderungen	• Mögliche Touchscreen-UI für Tablets & Mobile (grosse Schaltflächen). • Mögliche Internationalisierung der Sprache.
Eintrittshäufigkeit	Sehr häufig

3.4.5 UC04a: Formular ausfüllen

Use Case Name	UC04a: Formular ausfüllen
Primärakteur	Gast, Manager
Stakeholder und Interessen	Gast und Manager wollen mit minimalem Aufwand Daten in ein Formular eintragen.
Vorbedingungen	• System muss konfiguriert sein. • Formular muss vorhanden sein. • Gast respektive Manager ist in der Applikation. • Gast ist in der Gastansicht. • Manager ist in der Verwaltungsansicht.
Nachbedingungen	• Eintrag wurde erstellt.
Standardablauf	1. Gast respektive Manager füllt Formularfelder aus.
Erweiterungen	• Jederzeit, wenn das System einen Fehler hat. 1 System informiert über den Fehler. 2 System versucht früheren Zustand wiederherzustellen. 1. Gast respektive Manager bricht Vorgang ab. Der Use Case ist beendet. Manager kehrt zurück zu *UC01: Formular verwalten (CRUD)*.
Spezielle Anforderungen	• Mögliche Touchscreen-UI für Tablets & Mobile (grosse Schaltflächen). • Mögliche Internationalisierung der Sprache.
Eintrittshäufigkeit	Sehr häufig

3.4.6 UC04b: Eintrag bearbeiten

Use Case Name	UC04b: Eintrag bearbeiten
Primärakteur	Gast, Manager
Stakeholder und Interessen	Gast und Manager wollen mit minimalem Aufwand eingetragene Daten bearbeiten.
Vorbedingungen	• System muss konfiguriert sein. • Formular muss vorhanden sein. • Gast respektive Manager ist in der Applikation. • Gast ist in der persönlichen Datenansicht. • Manager ist in der Verwaltungsansicht.
Nachbedingungen	• Eintrag wurde bearbeitet.
Standardablauf	1. Gast respektive Manager bearbeitet Eintrag.
Erweiterungen	• Jederzeit, wenn das System einen Fehler hat. 1 System informiert über den Fehler. 2 System versucht früheren Zustand wiederherzustellen. 1. Gast respektive Manager bricht Vorgang ab. Der Use Case ist beendet. Manager kehrt zurück zu *UC01: Formular verwalten (CRUD)*.
Spezielle Anforderungen	• Mögliche Touchscreen-UI für Tablets & Mobile (grosse Schaltflächen). • Mögliche Internationalisierung der Sprache.
Eintrittshäufigkeit	Häufig

4 Nicht funktionale Anforderungen

4.1 Geschwindigkeit

Die durchschnittliche Ladezeit der Webseite soll unter zwei Sekunden und damit im üblichen Rahmen für Webapplikationen dieser Grösse liegen.
Im Falle von längeren Verarbeitungszeiten soll die Applikation sich wie unter *Reaktionszeit* beschrieben verhalten.

4.2 Skalierbarkeit

Mit heutigen Technologien ist die unterbruchfreie Migration eines Webhostings in die Cloud problemlos möglich. Aus diesem Grund liegt der Fokus bei dieser Arbeit nicht auf der Skalierbarkeit der Applikation.

4.3 Sicherheit

Sicherheit ist kein wesentlicher Bestandteil dieser Arbeit.

4.4 User Interface

Die Applikation soll durch ein modernes, ansprechendes und frisch wirkendes User Interface bestechen: Moderne Technologien (wie z.B. AJAX oder HTML5) sollen dazu beitragen.
Dank intuitiven Abläufen und klarer Benutzerführung soll das User Interface für weniger versierte Benutzer bedienbar sein.

4.5 Qualität

4.5.1 Wartbarkeit

Unit Tests stellen die Konsistenz und Wartbarkeit der Applikation sicher.

4.5.2 Erweiterbarkeit

Dank modularem Aufbau lässt sich die Applikation mühelos erweitern.

4.5.3 Usability

Reaktionszeit
Gemäss (Stolze, 2011) sollte die Applikation folgende Richtlinien für die Reaktionszeit eines User Interfaces einhalten:

- Bei sehr kleinen Aufgaben mit einer Reaktionszeit von < 0.5 Sekunden muss dem Benutzer kein spezielles Feedback gegeben werden.
- Bei Aufgaben zwischen 0.5 und 15 Sekunden muss der Benutzer informiert werden (z.B. mit einer Animation)
- Aufgaben, die länger als 15 Sekunden dauern, müssen den Benutzer im 5-Sekunden-Takt über den aktuellen Zustand informieren (z.B. ein Ladebalken) und ihm jederzeit die Möglichkeit bieten, die laufende Aufgabe abzubrechen.

Zuverlässigkeit
Bei einem Systemausfall soll der Benutzer informiert werden und das System in einen stabilen Zustand übergehen.

Funktionalität
Es ist nicht möglich, ungültige Operationen auf der Applikation auszuführen. Das System weist den Benutzer mit aussagekräftigen Fehlermeldungen darauf hin.

Portierbarkeit
Portierbarkeit ist kein wesentlicher Bestandteil dieser Arbeit

4.6 Beschränkungen

Die Applikation wird auf einem klassischen Webhosting (LAMP-Stack) betrieben.
Es dürfen nur Software-Komponenten verwendet werden, welche eine spätere kommerzielle Nutzung des Projekts zulassen würden.

5 Verzeichnisse

5.1 Abbildungsverzeichnis

5.2 Literaturverzeichnis

Stolze, M. (2011). *LE22 Externes Design Wahrnehmen, Kognition & Handeln.*

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

UI-Skizzen

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren: Fabian Schweizer, Manuel Schweizer
Betreuer: Dr. Daniel Keller
Projektpartner: foryouandyourcustomers AG, Pfäffikon ZH
Experte: Dr. Rudolf Mattmann
Gegenleser: Prof. Hansjörg Huser

Dokumentinformationen

Änderungsgeschichte

Datum	Version	Änderung	Autor
12.06.2013	0.1	Erstellung	ms
12.06.2013	0.2	Hinzufügen der UI-Skizzen	fs
12.06.2013	1.0	Beschreibung und Review	ms

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument beschreibt die UI-Skizzen für das Projekt Adgenero.

1.2 Gültigkeitsbereich

Das Dokument ist während der gesamten Projektdauer gültig und wird periodisch aktualisiert.

1.3 Referenzen

Die folgenden Dokumente werden in diesem Dokument referenziert:

- 10-Glossar/Glossar.pdf

1.4 Aufteilung

Das vorliegende Dokument gibt eine Übersicht die geplanten Ansichten von Adgenero und beschreibt die einzelnen Elemente kurz.

2 UI-Skizzen

Da bei Adgenero die Anzahl Screens von zentraler Bedeutung sind, resultieren lediglich acht UI-Skizzen aus dieser Anforderung. Diese werden hier aufgezeigt und kurz erläutert.

2.1 Startseite

Die Startseite ist der Ausgangspunkt. Direkt auf der Startseite soll die Möglichkeit geboten werden ein neues Formular (App) zu erstellen.

Abbildung 1: Startseite von Adgenero

2.2 Applikation erstellen

Der Vorgang zum Erstellen einer Applikation besteht aus mehreren Schritten: Zuerst werden die Grunddaten der Applikation angegeben. Nach dem eigentlichen Erstellen der Applikation gelangt der Nutzer zur Verwaltung der Felder. Wurden Felder hinzugefügt kann mit der Vorschau das Formular betrachtet und danach publiziert werden.

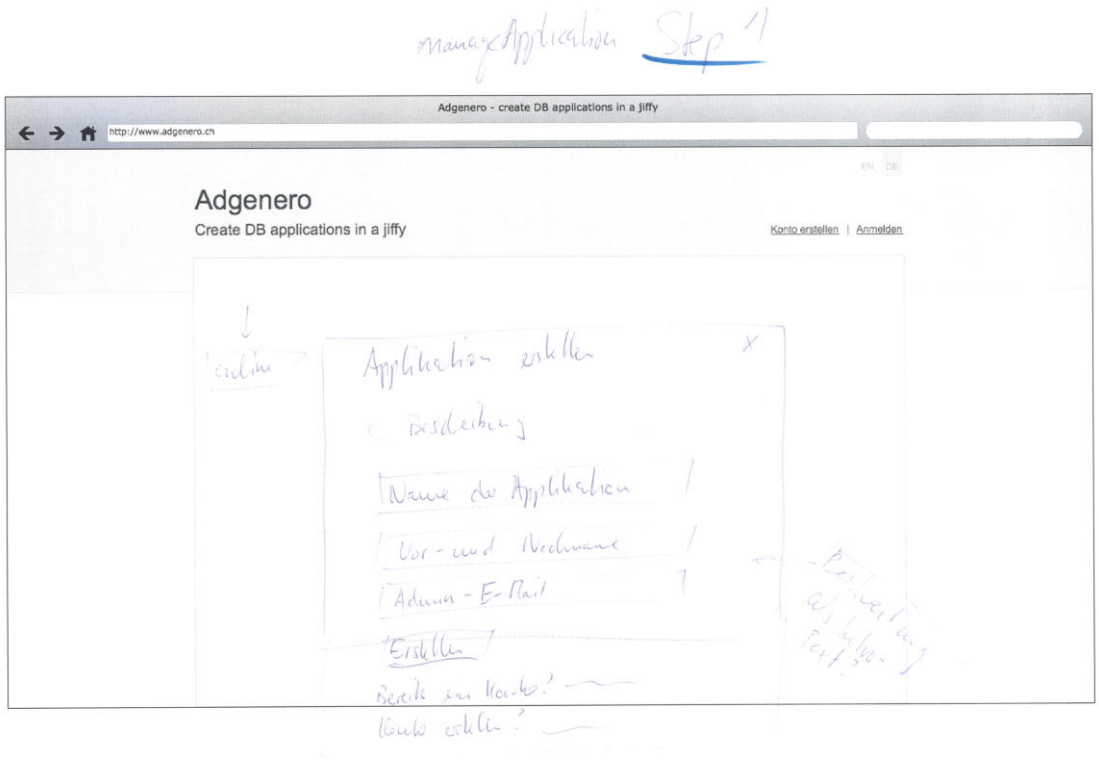


Abbildung 2: Angaben zur Applikation

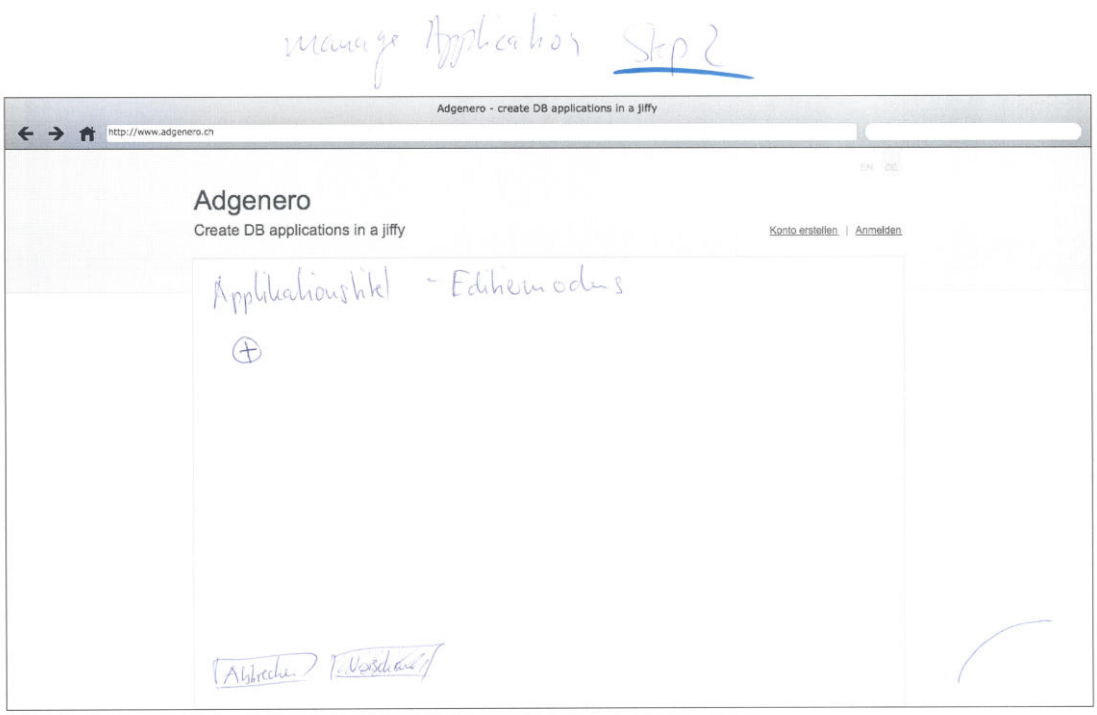


Abbildung 3: Editiermaske für Felder

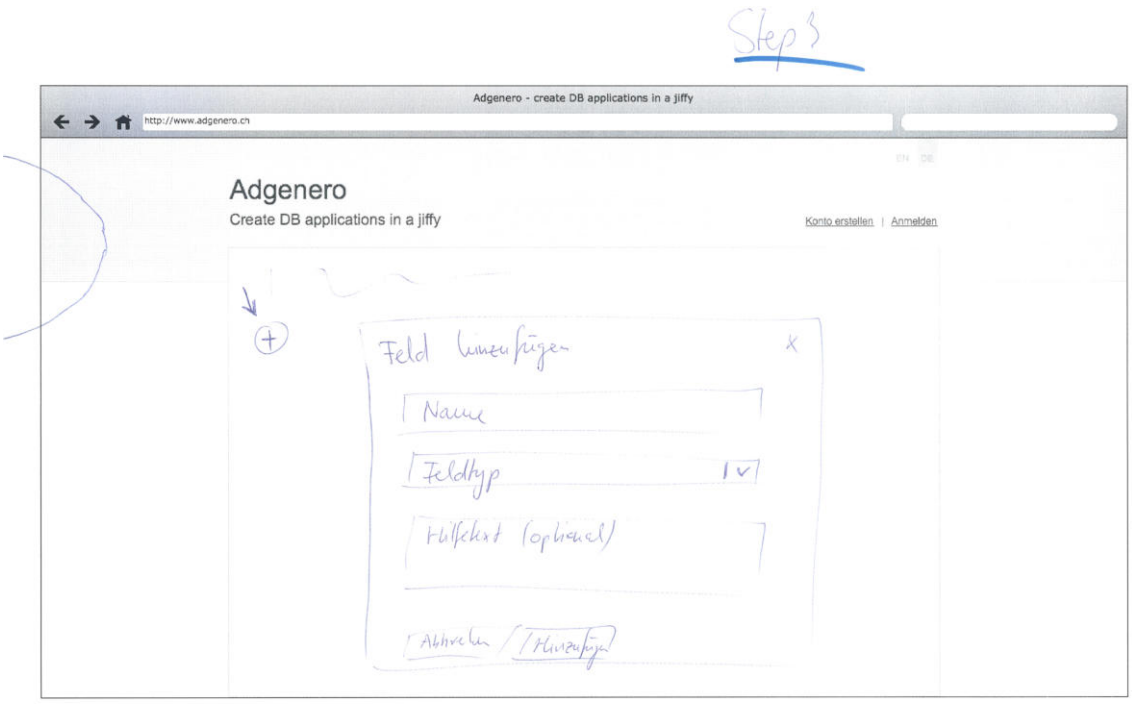


Abbildung 4: Angaben eines Felds

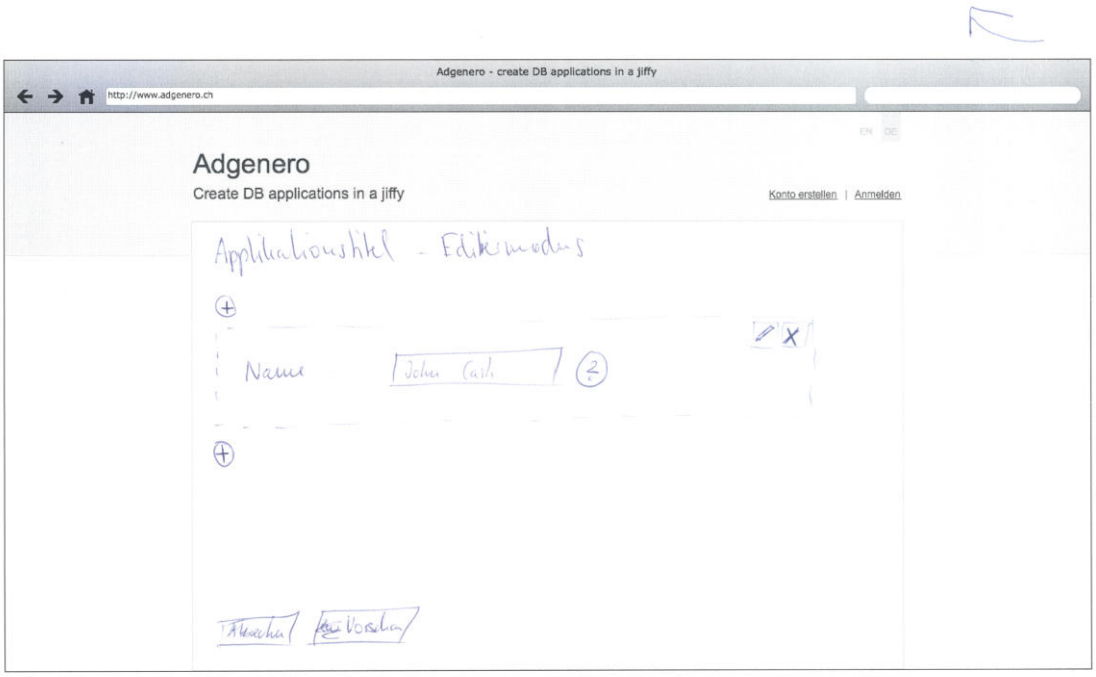


Abbildung 5: Editiermaske für Felder nach Hinzufügen eines Felds

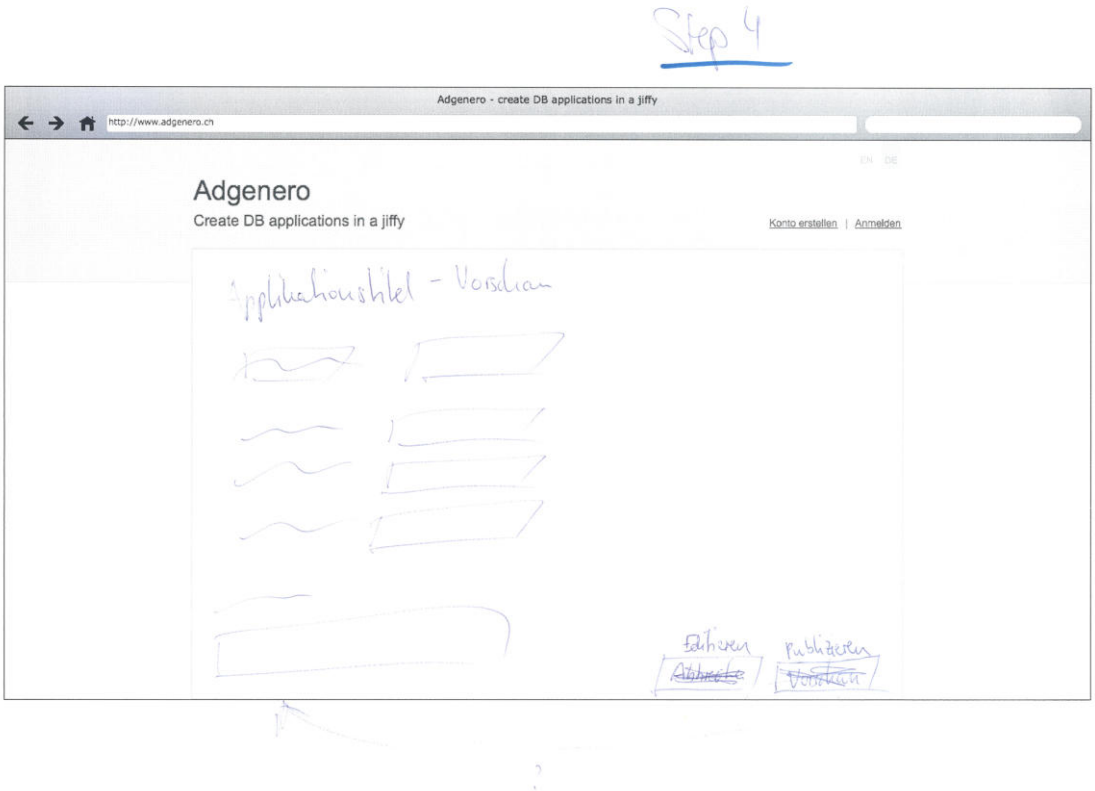


Abbildung 6: Vorschau der Applikation

2.3 Applikation verwalten

Nach der Publikation des Formulars gelangt der Manager der Applikation zur Übersicht der Einträge und kann diese verwalten.

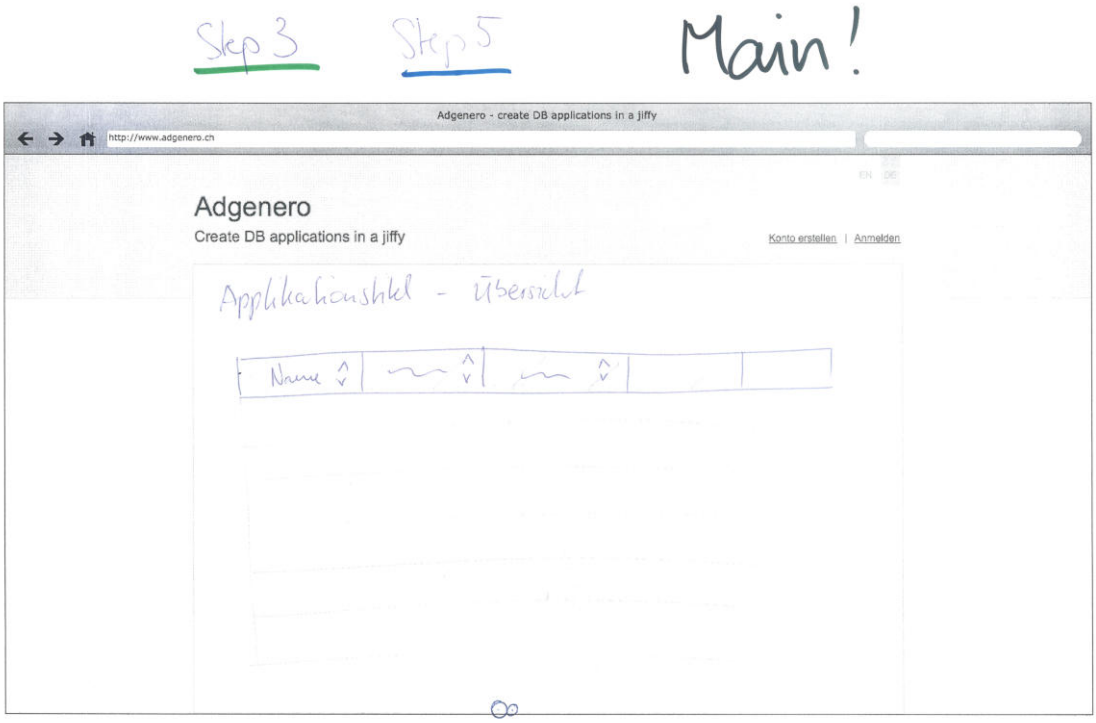


Abbildung 7: Übersicht der Einträge

2.4 Eintrag hinzufügen

Diese Skizze zeigt ein mögliches Formular für einen Gast.
Der Gast soll zudem die Möglichkeit haben, sich eine Kopie seiner Angaben per E-Mail zukommen zu lassen.

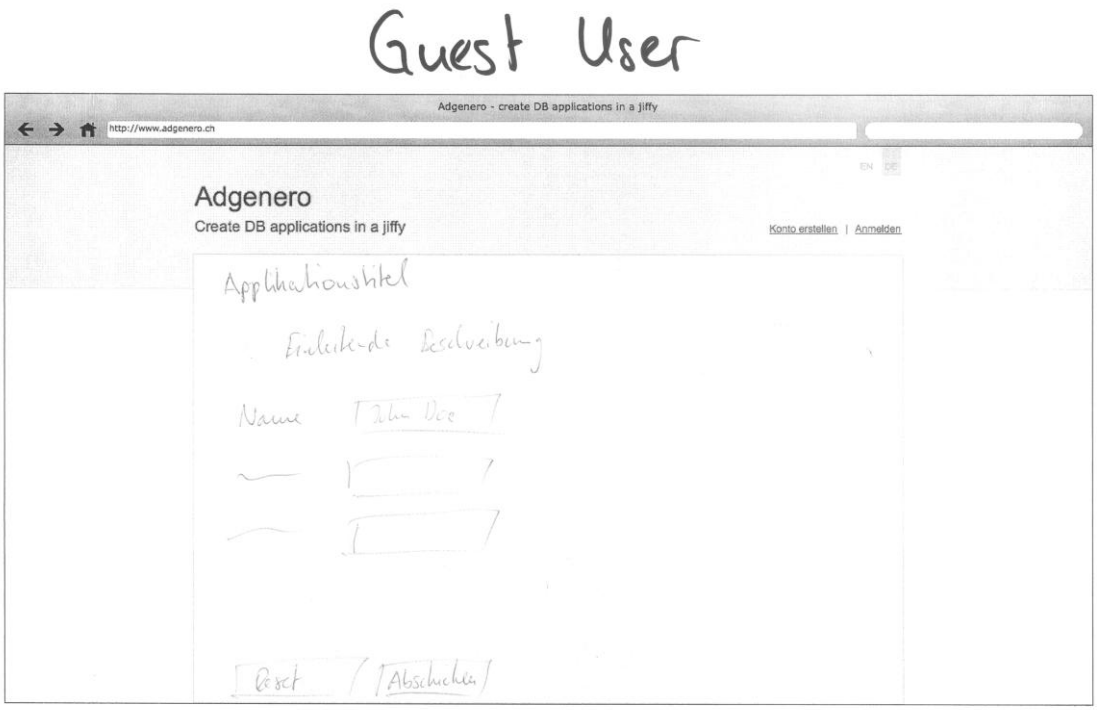


Abbildung 8: Ausfüllbares Formular

3 Verzeichnisse

3.1 Abbildungsverzeichnis

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

Domainanalyse

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren: Fabian Schweizer, Manuel Schweizer
Betreuer: Dr. Daniel Keller
Projektpartner: foryouandyourcustomers AG, Pfäffikon ZH
Experte: Dr. Rudolf Mattmann
Gegenleser: Prof. Hansjörg Huser

Dokumentinformationen

Änderungsgeschichte

Datum	Version	Änderung	Autor
28.02.2013	0.1	Template erstellt	ms
05.06.2013	1.0	Erste Version	fs
06.06.2013	1.1	Ergänzungen Domainmodell	fs
11.06.2013	1.2	Überarbeitung Konzepte	ms
12.06.2013	1.3	Überarbeitung	ms
14.06.2013	1.4	Last Review	ms

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument dient zur Analyse der Problemdomain für das Projekt Adgenero.

1.2 Gültigkeitsbereich

Das Dokument ist während der gesamten Projektdauer gültig und wird periodisch aktualisiert.

1.3 Referenzen

Die folgenden Dokumente werden in dieser Domainanalyse referenziert:

- 01-Aufgabenstellung/Aufgabenstellung.pdf
- 03-Projektplan/Projektplan.pdf
- 04-Anforderungsspezifikation/Anforderungsspezifikation.pdf
- 10-Glossar/Glossar.pdf

1.4 Aufteilung

Das vorliegende Dokument geht zuerst auf das Domainmodell und wichtige Konzepte ein. Einige Systemsequenz- und Aktivitätsdiagramme illustrieren zudem die Interaktionen mit dem Benutzer.

2 Domain Modell

2.1 Farbdefinition

Die im nachfolgenden Diagramm definierten Farben sind während des ganzen Projekts gültig und werden für alle Diagramme gleich gehandhabt.

2.2 Diagramm

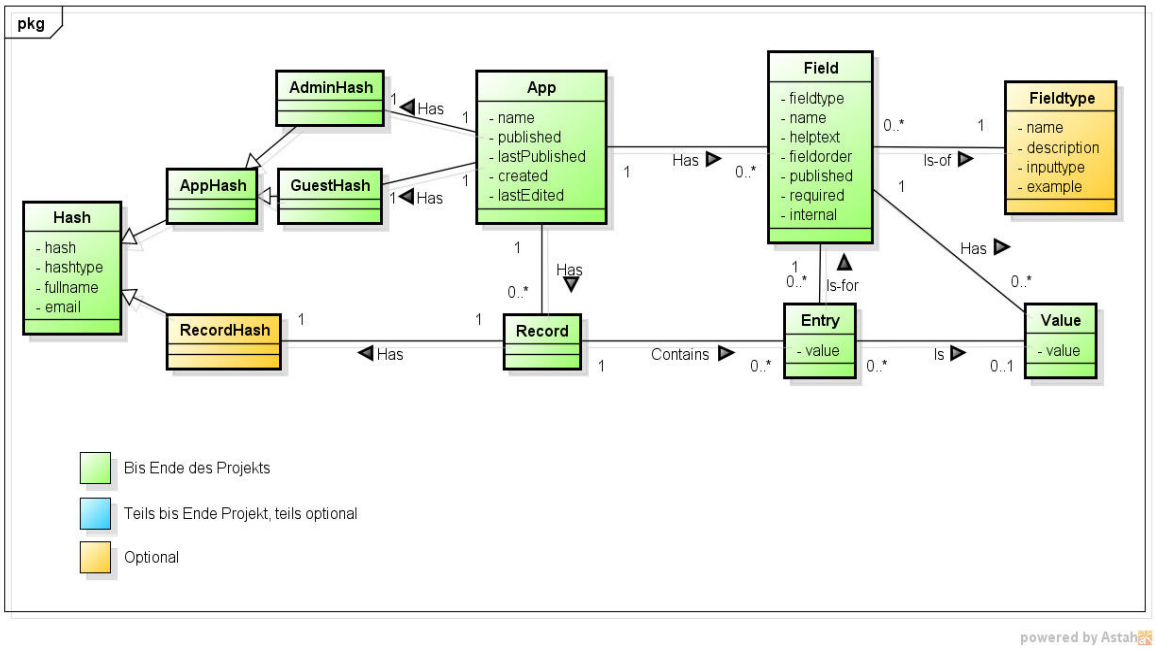


Abbildung 1: Domain Model Diagramm

2.3 Wichtige Konzepte

2.3.1 Hash

2.3.1.1 Beschreibung

Der Hash ist ein wesentlicher Bestandteil der Applikation, da dieser den Zugang zu gewissen Funktionen ermöglichen soll. Welche Funktionen das sind wird von der Logik gesteuert und ist abhängig vom Typ des Hashes.

2.3.1.2 Attribute

Attribut	Beschreibung
hash	Eindeutiger Hashwert
hashtype	Typ eines Hashs
fullname	Der volle Name des Benutzers, dem der Hash zugewiesen ist.
email	Die E-Mail-Adresse des Benutzers, dem der Hash zugewiesen ist.

2.3.1.3 Beziehungen

Multiplizität	Zu Konzept	Beschreibung
1:1	App	Ein Hash vom Typ Admin gehört zu einer App und erlaubt die Administration, bzw. Verwaltung des Formulars
1:1	App	Ein Hash vom Typ Guest gehört zu einer App und erlaubt das Eintragen in ein bestimmtes Formular.
1:1	Record	Ein Hash vom Typ Record gehört zu einem Eintrag und erlaubt es diesen zu bearbeiten.

2.3.2 App

2.3.2.1 Beschreibung

Das Konzept einer App widerspiegelt ein Formular, das Kernstück der gesamten Applikation. Formulare werden erstellt, modifiziert und von Benutzern ausgefüllt.

2.3.2.2 Attribute

Attribut	Beschreibung
name	Der Name der App respektive des Formulars.
published	Ein Flag, das angibt ob eine App publiziert ist.
lastPublished	Zeitangabe, wann die App zuletzt publiziert wurde.
created	Zeitangabe, wann die App erstellt wurde.
lastEdited	Zeitangabe, wann die App zuletzt bearbeitet wurde.

2.3.2.3 Beziehungen

Multiplizität	Zu Konzept	Beschreibung
1:1	AdminHash	Jede App hat einen Verwaltungshash.
1:1	GuestHash	Jede App hat einen Hash zur Eintragung von Daten.
1:*	Field	Eine App verfügt über Felder, welche vom Admin verwaltet werden.
1:*	Record	Eine App hat beliebig viele Einträge von Gästen.

2.3.3 Field

2.3.3.1 Beschreibung

Ein Field stellt ein Feld in einer App dar, welches ausgefüllt werden kann.

2.3.3.2 Attribute

Attribut	Beschreibung
fieldtype	Der Typ eines Felds.
name	Der Name des Felds.
helptext	Ein optionaler Hilfetext zur Unterstützung beim Ausfüllen des Formulars.
fieldorder	Die Sortierreihenfolge des Felds.
published	Ein Flag, das angibt ob ein Feld publiziert ist.
required	Ein Flag, das angibt ob ein Feld benötigt wird.
internal	Ein Flag, das angibt ob ein Feld intern ist.

2.3.3.3 Beziehungen

Multiplizität	Zu Konzept	Beschreibung
***:1**	App	Ein Feld ist einer App zugewiesen.
1:*	Entry	Ein Feld wird beliebig vielmal ausgefüllt.
1:*	Fieldvalue	Ein Feld kann vorgegebene Werte haben.
1:1	Fieldtype	Ein Feld hat einen Feldtyp.

2.3.4 Record

2.3.4.1 Beschreibung

Ein Record stellt ein ausgefülltes Formular dar.

2.3.4.2 Attribute

Records verfügen über keine Attribute.

2.3.4.3 Beziehungen

Multiplizität	Zu Konzept	Beschreibung
1:1	RecordHash	Jeder Record hat einen Bearbeitungshash.
1:1	App	Ein Record ist genau einer App zugewiesen.
1:*	Entry	Ein Record fasst beliebig viele Entries zusammen.

2.3.5 Entry

2.3.5.1 Beschreibung

Für ein Feld wird beim Ausfüllen eines Formulars ein Entry eingelegt, welcher den Wert enthält der eingetragen wurde. Entries werden zusammengefasst als Record.

2.3.5.2 Attribute

Attribut	Beschreibung
value	Stellt einen eingetragenen Wert dar.

2.3.5.3 Beziehungen

Multiplizität	Zu Konzept	Beschreibung
1:1	Record	Jeder Entry ist genau einem Record zugeteilt.
1:1	Field	Ist der Eintrag zu einem Feld.
1:0..1	Fieldvalue	Kann ein vorgegebener Wert sein.

3 Systemsequenzdiagramme

3.1 UC02: Feld verwalten

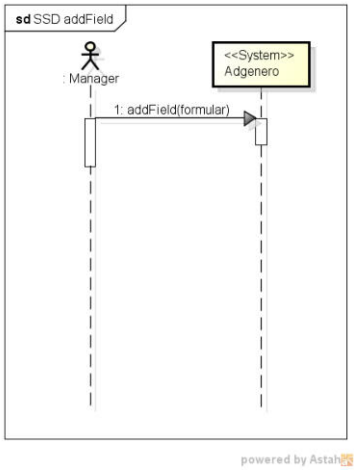


Abbildung 2: SSD addField

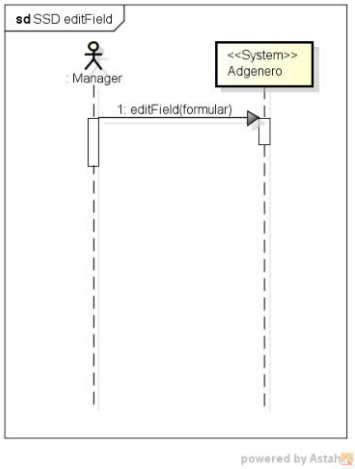


Abbildung 3: SSD editField

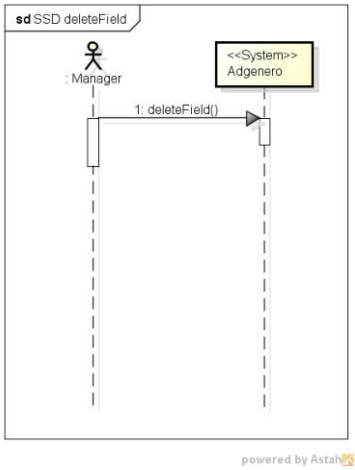


Abbildung 4: SSD deleteField

4 Aktivitätsdiagramme

4.1 App erstellen

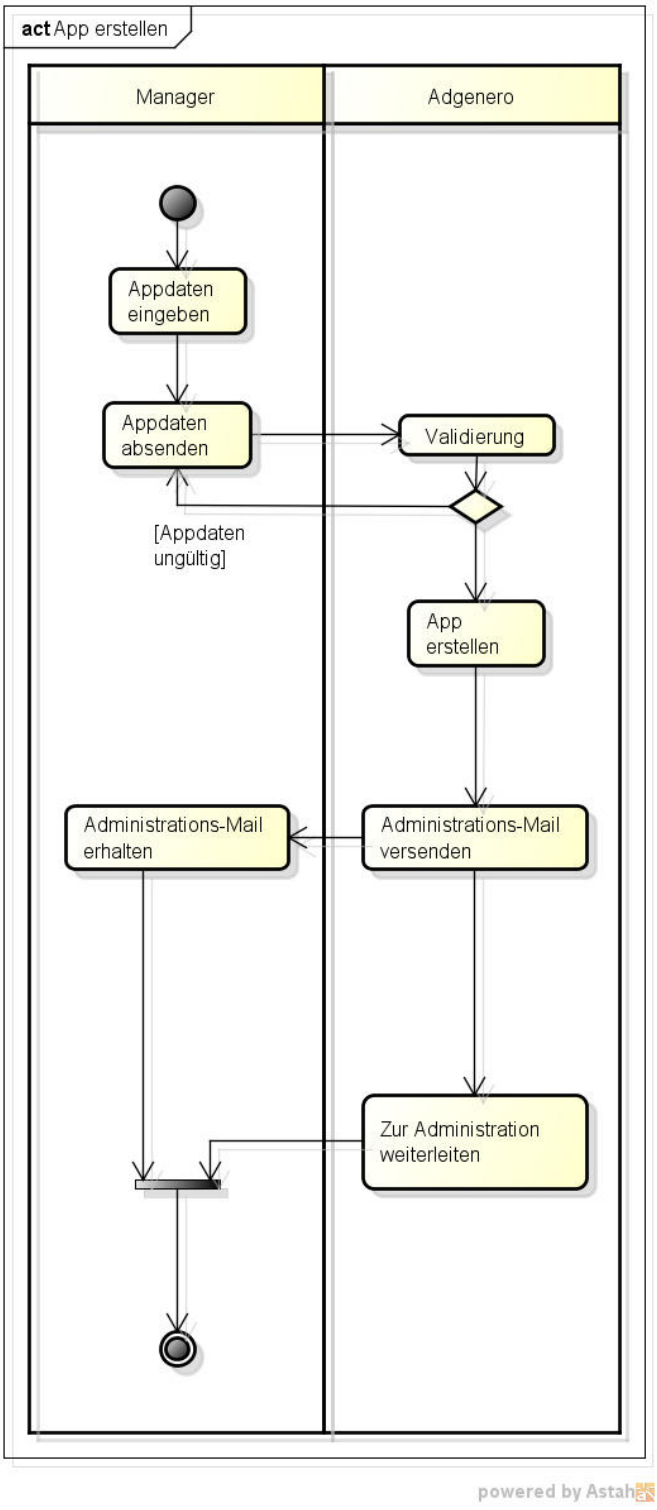


Abbildung 5: Aktivitätsdiagramm App erstellen

4.2 Feld hinzufügen

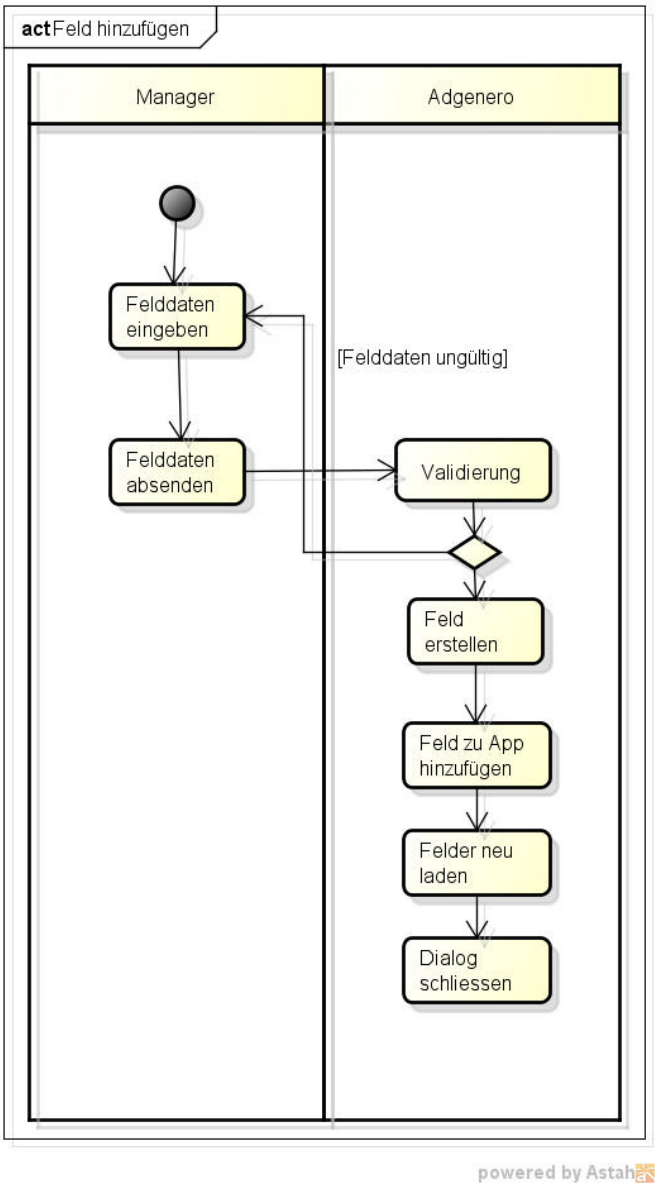


Abbildung 6: Aktivitätsdiagramm Feld hinzufügen

5 Verzeichnisse

5.1 Abbildungsverzeichnis

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

Software Architecture Document

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren:	Fabian Schweizer, Manuel Schweizer
Betreuer:	Dr. Daniel Keller
Projektpartner:	foryouandyourcustomers AG, Pfäffikon ZH
Experte:	Dr. Rudolf Mattmann
Gegenleser:	Prof. Hansjörg Huser

Dokumentinformationen

Änderungsgeschichte

Datum	Version	Änderung	Autor
28.02.2013	0.1	Template erstellt	ms
04.06.2013	1.0	Erste Version	fs
06.06.2013	1.1	Überarbeitung	fs
12.06.2013	1.2	Titelblatt und Korrekturen	ms
12.06.2013	1.3	Überarbeitung	ms
14.06.2013	1.4	Last Review	ms

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument beschreibt die Softwarearchitektur für das Projekt Adgenero.

1.2 Gültigkeitsbereich

Das Dokument ist während der gesamten Projektdauer gültig und wird periodisch aktualisiert.

1.3 Referenzen

Die folgenden Dokumente werden in dieser Domainanalyse referenziert:

- 01-Aufgabenstellung/Aufgabenstellung.pdf
- 03-Projektplan/Projektplan.pdf
- 04-Anforderungsspezifikation/Anforderungsspezifikation.pdf
- 10-Glossar/Glossar.pdf

1.4 Aufteilung

Das vorliegende Dokument schafft zuerst einen Überblick über das Gesamtsystem und geht danach näher auf die verschiedenen Schichten der Software ein.

2 Systemübersicht

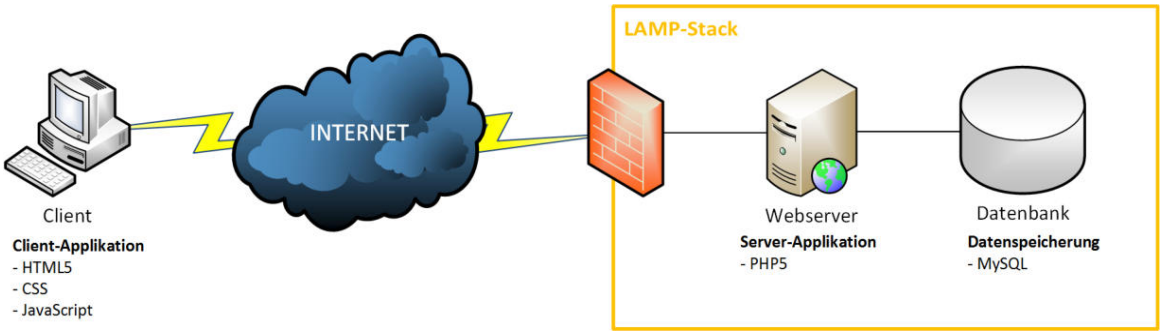


Abbildung 1: Systemübersicht

Bei Adgenero handelt es sich um eine Webapplikation. Der Client kommuniziert dabei über HTTP-Requests mit dem Server. Für die Datenspeicherung wird ein MySQL-Datenbankserver verwendet.
Die einzelnen Systembausteine werden in den folgenden Abschnitten genauer beschrieben.

2.1 Datenspeicherung

Für die Datenspeicherung wird MySQL als Datenbankserver verwendet. Der Datenbankserver liegt im vorliegenden Fall auf derselben Maschine wie der Webserver. Das Webhosting wird von der Firma Cyon GmbH in Basel betrieben.

Als Schnittstelle zwischen Businesslogik und Datenhaltung wird der OR-Mapper Doctrine 2 eingesetzt. Doctrine erlaubt das Arbeiten mit diversen Datenbankservern, wie etwa Microsoft SQL oder MySQL. Um den kompletten Funktionsumfang von Doctrine zu nutzen werden die Tabellen der Datenbank als Typ InnoDB gespeichert.

Der Datenspeicherungsprozess wird durch einen entsprechenden HTTP-Request vom Benutzer (bzw. dessen Client) ausgelöst und erfolgt innerhalb der Businesslogik der Applikation.

2.2 Server

Die Server-Applikation ist komplett in PHP geschrieben und umfasst sowohl die Businesslogik als auch die Schnittstellen zur Datenspeicherung. Für die Kommunikation wird ein Apache Webserver eingesetzt, welcher die HTTP-Requests entgegennimmt.

Die von der HSR zur Verfügung gestellte virtuelle Maschine verfügt bereits über einen Apache Webserver. Redmine und Jenkins ist bereits vorinstalliert und wird im Rahmen dieser Arbeit auch eingesetzt. Für den einfacheren Zugriff werden zusätzlich die Subdomains redmine.adgenero.ch und jenkins.adgenero.ch angelegt.
Die produktive Applikation wird auf dem Server von cyon.ch betrieben. Dieser besteht aus dem für Webhostings üblichen LAMP-Stack (Linux, Apache, MySQL, PHP).

Das von uns eingesetzte Zend Framework 2 erlaubt im Zusammenspiel mit dem Apache-Modul mod_rewrite die Verwendung von „friendly URLs“. Dadurch kann in der Applikation Request-Routing verwendet werden.

2.3 Client

Da es sich beim Endergebnis um einen hosted Service handelt, besteht der Client im Wesentlichen aus dem vom Benutzer eingesetzten Internetbrowser.
Die Client-Applikation ist eine in HTML ausgelieferte Webseite, welche dank dem Einsatz von JavaScript clientseitig um Funktionalität erweitert wird.

Als Client wird ein moderner und aktueller Browser vorausgesetzt, welcher HTML5 und JavaScript unterstützt.

3 Architektonische Ziele & Einschränkungen

3.1 Entwicklungstools

Folgende Programme werden im Rahmen dieser Arbeit verwendet:

- **eclipse** – Wird als IDE verwendet, da ein breites Spektrum an kostenlosen Plugins verfügbar ist.
- **GIT** - Als Source Code Repository wird ein GIT-Repository eingesetzt. Das Repository ist über eine SSH-Verbindung zugänglich und befindet sich auf dem von der HSR zur Verfügung gestellten virtuellen Server.
- **Astah** – Die UML-Software Astah Community Edition wird verwendet um UML-Diagramme zu erstellen.
- **Visio** – Visio wird eingesetzt um Diagramme zu erstellen, die mit Astah nicht möglich sind.
- **Redmine** – Redmine ist unser Projektmangement-Tool.
- **Jenkins** – Jenkins ist ein Build-Server. Im Rahmen dieses Projekts wird Jenkins lediglich für die Erzeugung von Code-Metriken verwendet.
- **XAMPP** – XAMPP ist ein Apache, MySQL, PHP und Perl Stack für verschiedene Betriebssysteme.
- **phpUnit** – Wird für Unit-Testing und Codecoverage verwendet.

3.2 Technologien

Folgende Technologien werden in dieser Bachelorarbeit verwendet:

- **Zend Framework 2** – Ist ein Backend-Framework geschrieben in PHP, welches die wichtigsten Konzepte wie MVC und Frontcontroller integriert.
- **Doctrine 2** – In PHP geschriebener OR-Mapper für eine breite Auswahl an Datenbanken wie MySQL oder MSSQL.
- **Twitter Bootstrap** – Das derzeit meistverbreitete Frontend-Framework entwickelt von Twitter bietet eine breite Palette an Elementen für Design und Funktionalität.
- **jQuery** – jQuery ist eine JavaScript-Bibliothek, welche nebst einem grossen Funktionsumfang auch eine einfachere Handhabung von JavaScript bietet.
- **DataTables** – DataTables ist ein jQuery-Plugin, welches es erlaubt, HTML-Tabellen zu sortieren, filtern und Pagination bietet.

4 Wichtige Architekturentscheidungen

4.1 Technologien

Zend Framework 2
Nach einem Vergleich verschiedener PHP-Frameworks haben wir uns für das Zend Framework 2 entschieden, da dieses nicht nur die wichtigsten Konzepte wie MVC integriert, sondern auch bereits Erfahrung mit dem Vorgänger Zend Framework vorhanden sind.
MVC ist derzeit der bei Webapplikationen eingesetzte Standard.

Doctrine
Zend Framework 2 empfiehlt die Arbeit mit Entitäten (Models). Wir haben uns daher für Doctrine entschieden, da die Handhabung von Joins einfacher und Lazy-Loading bereits integriert ist. Dies erlaubt es nur die wirklich nötigen Datenzugriffe zu tätigen.

Twitter Bootstrap
Um die Gestaltung des Frontends zu erleichtern, haben wir uns für Twitter Bootstrap entschieden. Twitter Bootstrap wurde so konstruiert, dass es auf sehr vielen Browsern gleich dargestellt wird und erlaubt responsive Design zu verwenden.

4.2 Mapper & Models

ZF2 empfiehlt die Verwendung von Mapper & Models. Doctrine verwendet Models.
Obwohl wir uns entschieden haben Doctrine zu verwenden, wollen uns nicht fest an diese Entscheidung binden. Daher werden Mapper & Models so abstrahiert, dass sie auch ohne Doctrine verwendet werden können.
Einzige Ausnahme bilden Models, welche eine Collection verwenden. PHP und ZF2 bringen derzeit keine uns bekannte Collection mit, welche standardmässig vorsortiert werden kann.

4.3 Ableitungen von Hash & Hashtype

4.3.1 Entscheidung

Die Hashs werden alle in einer Tabelle gespeichert um die Eindeutigkeit jedes einzelnen Hashs zu gewährleisten. Ein Problem stellt sich dabei, dass einige der Hashs über eine Referenz zu einer App, andere eine Referenz auf einen Record, aufweisen.
Grundsätzlich wäre es möglich die 3NF anzustreben und jeweils eine eigene Tabelle für die „Apphashs“ und „Recordhashs“ anzulegen mit einer Referenz auf die Hashtabelle. Diesen Ansatz wollten wir in dieser Arbeit nicht verwenden, da es bedeuten würde eine weitere Verknüpfung verwenden zu müssen.

4.3.2 Diskriminator-Mapping

```
<entity name="Adgenero\Entity\Hash" inheritance-type="SINGLE_TABLE" table="hash">
    <discriminator-column name="hashtype" type="string" />
    <discriminator-map>
        <discriminator-mapping value="default" class="Adgenero\Entity\Hash" />
        <discriminator-mapping value="app" class="Adgenero\Entity\AppHash" />
        <discriminator-mapping value="guest" class="Adgenero\Entity\GuestHash" />
        <discriminator-mapping value="admin" class="Adgenero\Entity\AdminHash" />
        <discriminator-mapping value="record" class="Adgenero\Entity\RecordHash" />
    </discriminator-map>
```

Abbildung 2: Diskriminator-Mapping der Hashs

Mit Doctrine ist es möglich verschiedene abgeleitete Klassen in eine Tabelle zu speichern. Dazu wird ein sogenanntes Diskriminator-Mapping verwendet, bei welchem in der Basisklasse angegeben werden muss, welcher Wert zu welcher Subklasse gehört.
In diesem Projekt wurde für die Unterscheidung der Hashtype verwendet, welcher angibt um welche Entität (zum Beispiel RecordHash) es sich handelt.
Da es im Rahmen dieser Arbeit wesentlich ist, ob ein Hash vom Typ Admin oder Guest ist, um die Funktionalität entsprechend zu erlauben, wurden diese beiden Typen zusätzlich noch vom AppHash abgeleitet.

5 Logische Architektur

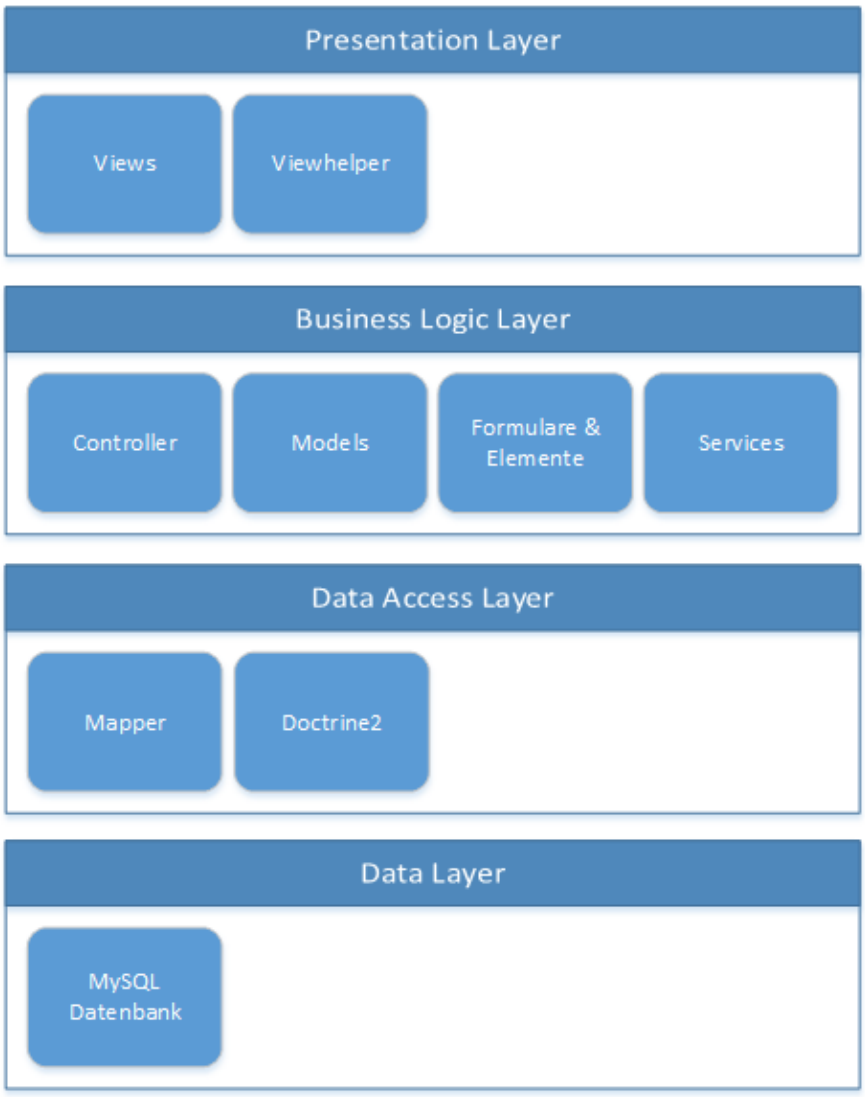


Abbildung 3: Übersicht Schichten

Die Schichten sind aufgeteilt in Presentation, Business Logic, Data Access und Data Layer.
Die Aufteilung des verwendeten MVC-Patterns ist dabei so, dass Model und Controller im Business Logic Layer sind und die View im Presentation Layer ist.

5.1 Farbdefinition

Die in nachfolgenden Diagrammen definierten Farben sind während des ganzen Projekts gültig und werden für alle Diagramme gleich gehandhabt.

5.2 Presentation Layer

Im Presentation Layer sind das gesamte Layout sowie Teile der View abgelegt. Dazu gehören auch öffentlich zugängliche Ressourcen wie Stylesheets und JavaScript-Dateien.

Hilfsklassen für die Anzeige, wie etwa der im Rahmen dieser Arbeit zu entwickelnde Viewhelper für Formulare, gehören ebenfalls zur Präsentationsschicht.

5.3 Business Logic Layer

5.3.1 Beschreibung

Diese Schicht ist aufgeteilt in Controller, Modelklassen, Formulare & Elemente sowie in Services und das Routing.

5.3.2 Klassenstruktur

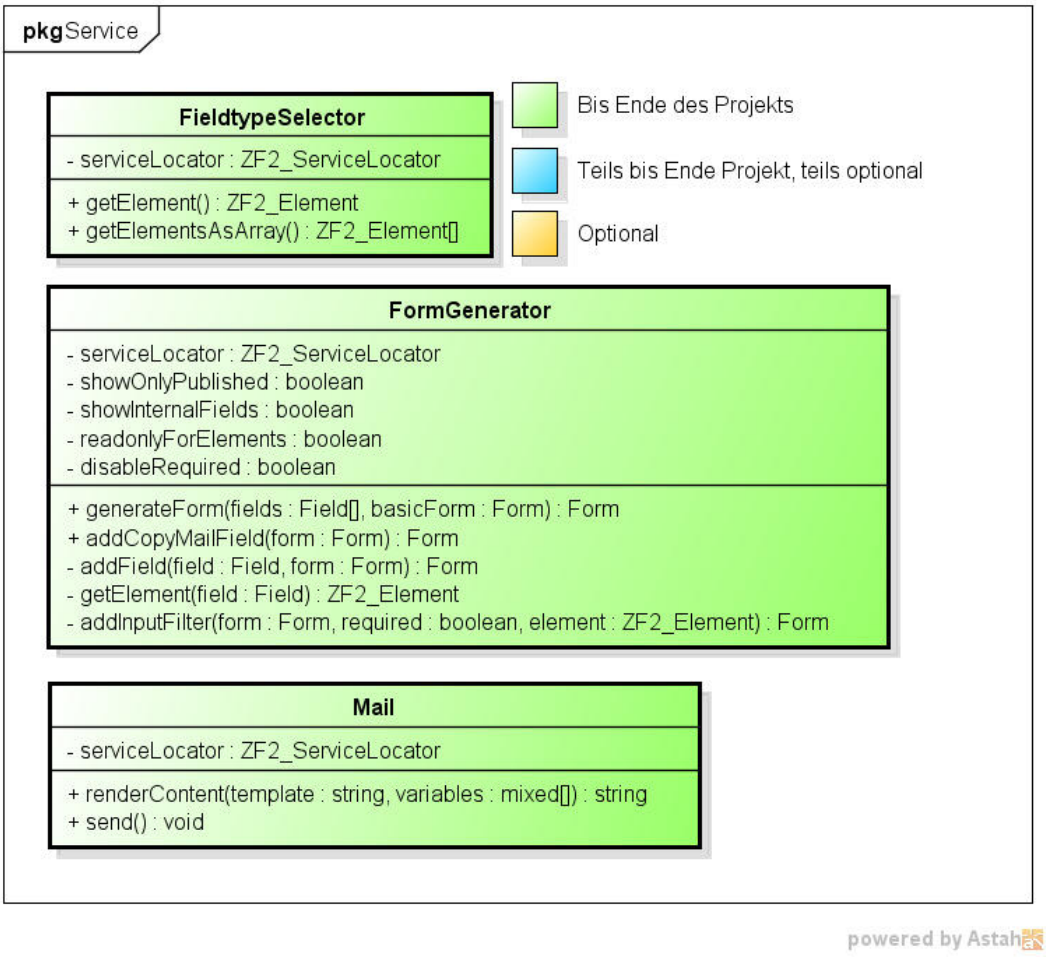


Abbildung 4: Klassendiagramm Services

Die Abbildung zeigt lediglich die Klassenstruktur der Services. Eine Klassenstruktur des gesamten Business Logic Layers ist im Anhang dieses Dokuments zu finden.

In der nachstehenden Tabelle werden die Subpakete des Busines Logic Layers kurz erläutert:

Subpaket	Beschreibung
Controller	Erhalten Requests vom Benutzer anhand des Routings und liefern einen Response zurück. Beinhalten Logik zur Verarbeitung der Anfrage.
Models	Stellen Objekte dar und enthalten Daten zum jeweiligen Objekt.
Formulare & Elemente	Sind Formulare und zugehörige Elemente und werden bei der Rückgabe gerendert.
Services	Beinhalten sämtliche Dienste wie zum Beispiel Generierung von Formularen.

5.4 Data Access Layer

5.4.1 Beschreibung

Der Data Access Layer ist zuständig für den Zugriff auf Daten sowie die Speicherung. Er gilt als Schnittstelle zwischen dem Data Layer und dem Business Logic Layer (BLL).
Doctrine sowie Mapper-Klassen werden verwendet, um Daten aus dem Data Layer zu lesen und an den BLL zu übergeben, respektive Daten vom BLL in den Data Layer zu persistieren.

5.4.2 Klassenstruktur

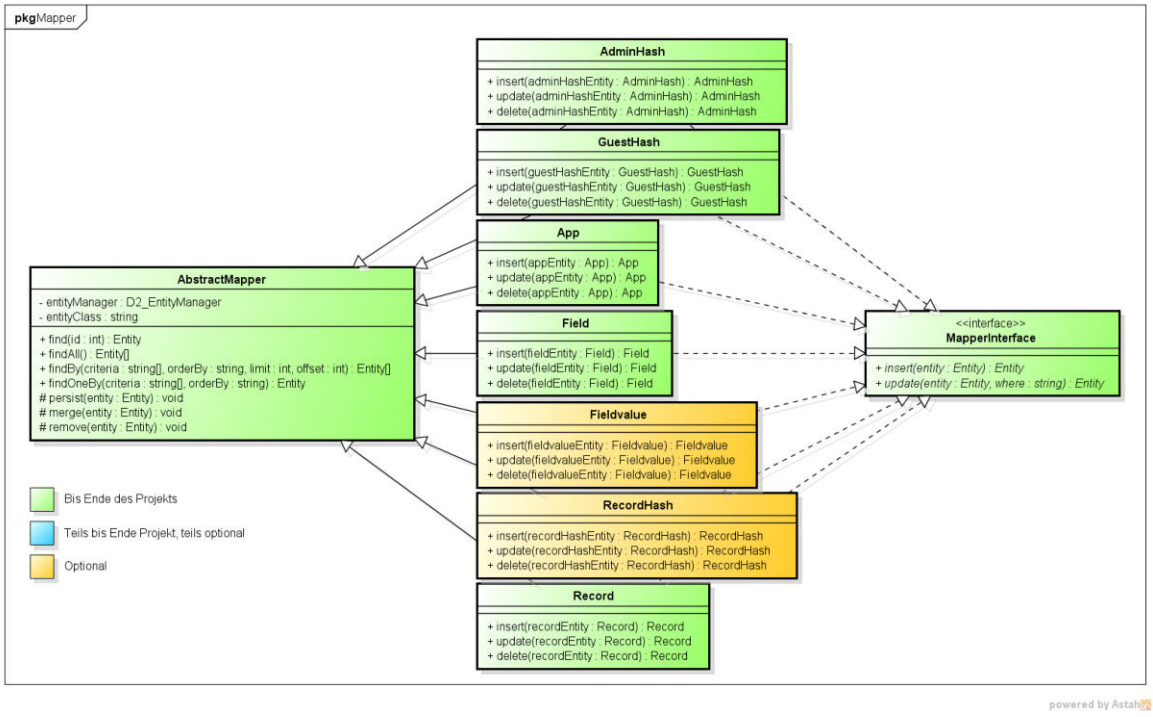


Abbildung 5: Klassendiagramm Mapper

Die Abbildung zeigt alle Mapper, welche sich im Data Access Layer befinden. Im Rahmen dieser Arbeit werden alle Mapper ausprogrammiert. Es besteht die Möglichkeit einen generischen Mapper zu verwenden um den sich wiederholenden Code der einzelnen Mapper zu umgehen.

5.5 Data Layer

Die Daten sind im Data Layer gehalten. Für diese Arbeit handelt es sich dabei um die verwendete MySQL-Datenbank.

6 Prozesse und Threads

Wie bei Webapplikationen üblich kann die Applikation mehrfach gestartet werden.
Damit nicht bei jeder Anfrage die komplette Applikation geladen werden muss, bietet das Zend Framework 2 FrontController- und Dispatch-Pattern an, welche zusammen mit dem Routing nur den zuständigen Controller laden.

Die Applikation selbst greift nicht auf Threading zurück.

7 Deployment

7.1 Entwicklungsumgebung

Während der Arbeit wird lokal auf den Entwicklermaschinen oder in Entwickler-VMs entwickelt. Die Applikation wird mit dem eclipse-Plugin „File synchronization“ bei jedem Speichervorgang in die lokale XAMPP-Installation synchronisiert.

7.2 Buildserver

Jenkins verwendet jeweils die neuste Version des Master-Branches direkt aus dem GIT-Repository.

8 Datenspeicherung

8.1 Mapper

Zend Framework 2 empfiehlt die Verwendung von Mapper im Data Access Layer. Für diese Arbeit sehen wir vor mit Doctrine, einem OR-Mapper, zu arbeiten, was die Verwendung von eigenen Table-gateways erübrigt.
Um eine mögliche Austauschbarkeit von Doctrine zu gewährleisten (Abstrahierung) werden die Mapper aber Doctrine-unabhängig entwickelt.

8.2 Doctrine Annotations

Als Schnittstelle zwischen dem Business Logic Layer und dem Data Layer wird Doctrine eingesetzt. Doctrine kennt mehrere Varianten zur Beschreibung von Modelklassen. Um eine allfällige Austauschbarkeit von Doctrine zu gewährleisten werden die Modelklassenbeschreibungen in separaten XML-Mapping-Dateien gespeichert anstelle dass die Beschreibung direkt in der jeweiligen Modelklasse vorliegt.
Dieses Prinzip wird aber bewusst durch die Verwendung von Doctrine-Collections in Modelklassen verletzt. PHP bringt derzeit keine ähnlich intelligenten Collections mit.

```
<?xml version="1.0" encoding="UTF-8"?>
<doctrine-mapping xmlns="http://doctrine-project.org/schemas/orm/doctrine-mapping"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:schemaLocation="http://doctrine-project.org/schemas/orm/doctrine-mapping
                  http://doctrine-project.org/schemas/orm/doctrine-mapping.xsd">

    <mapped-superclass name="Adgenero\Entity\App" table="app">

        <id name="id" type="integer" column="id">
            <generator strategy="AUTO" />
        </id>

        <field name="name" type="string" length="60" nullable="false" />
        <field name="description" type="text" nullable="true" />
        <field name="isPublished" column="is_published" type="boolean" nullable="false"/>
        <field name="lastPublished" column="last_published" type="datetime" nullable="true" />
        <field name="created" type="datetime" nullable="false" />
        <field name="lastEdited" column="last_edited" type="datetime" nullable="true" />

        <one-to-many field="fields" target-entity="Field" mapped-by="app">
            <cascade>
                <cascade-all />
            </cascade>
            <order-by>
                <order-by-field name="fieldorder" direction="ASC" />
            </order-by>
        </one-to-many>

        <one-to-many field="records" target-entity="Record" mapped-by="app">
            <cascade>
                <cascade-all />
            </cascade>
        </one-to-many>

    </mapped-superclass>

</doctrine-mapping>
```

Abbildung 6: Doctrine XML Annotationen

9 Anhang

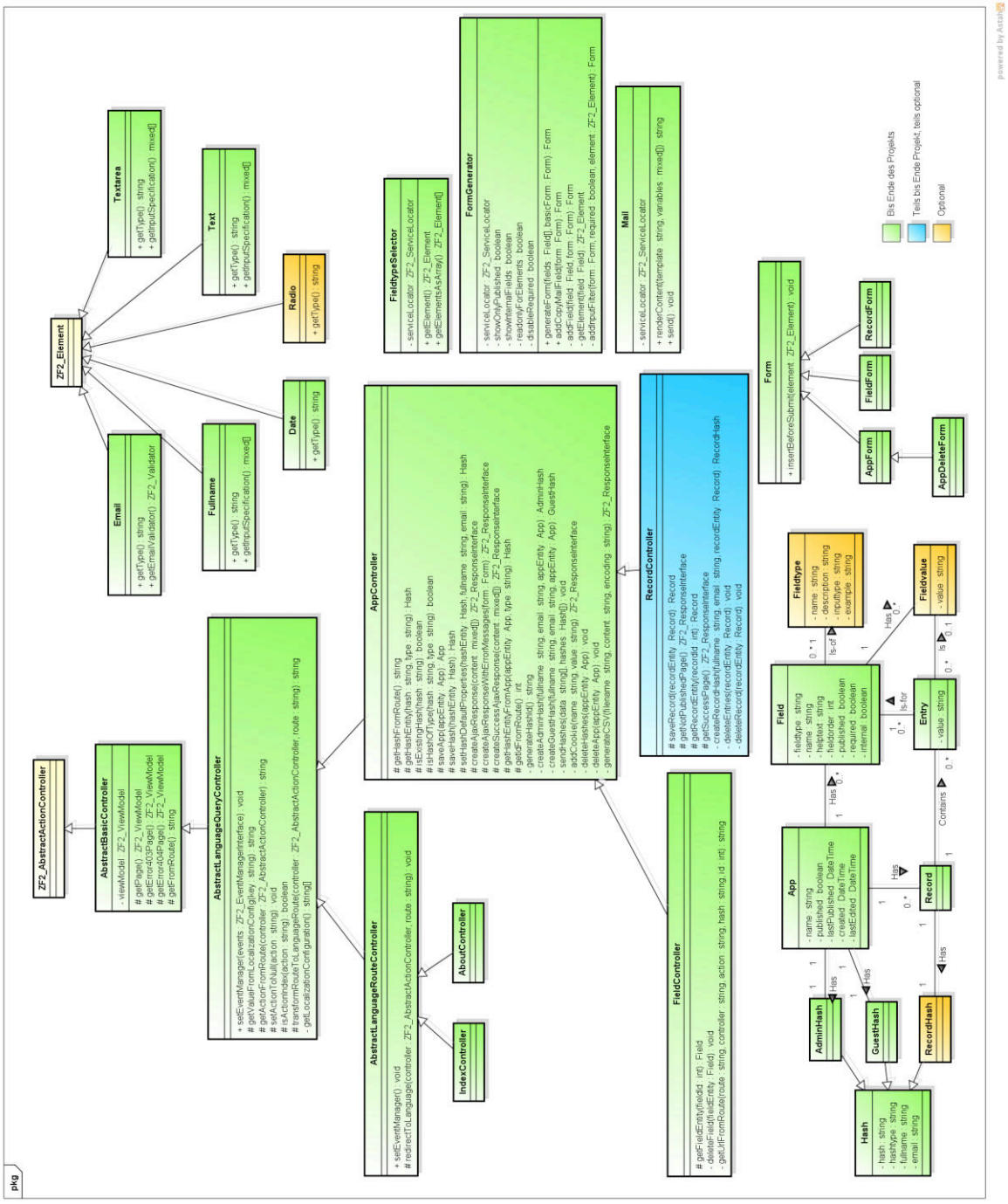


Abbildung 7: Klassendiagramm Business Logic Layer

10 Verzeichnisse

10.1 Abbildungsverzeichnis

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

Testspezifikation

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren: Fabian Schweizer, Manuel Schweizer
Betreuer: Dr. Daniel Keller
Projektpartner: foryouandyourcustomers AG, Pfäffikon ZH
Experte: Dr. Rudolf Mattmann
Gegenleser: Prof. Hansjörg Huser

Dokumentinformationen

Änderungsgeschichte

Datum	Version	Änderung	Autor
25.02.2013	0.1	Erstellen des Templates	ms
07.06.2013	1.0	Erste Fassung	fs
12.06.2013	1.1	Überarbeitung	ms
14.06.2013	1.2	Last Review	ms

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument beschreibt die Grundlagen für die einzelnen Testläufe.

1.2 Gültigkeitsbereich

Das Dokument ist während der gesamten Projektdauer gültig und wird periodisch aktualisiert.

1.3 Referenzen

Die folgenden Dokumente werden in dieser Testspezifikation referenziert:

- 01-Aufgabenstellung/Aufgabenstellung.pdf
- 03-Projektplan/Projektplan.pdf
- 04-Anforderungsspezifikation/Anforderungsspezifikation.pdf
- 10-Glossar/Glossar.pdf

1.4 Aufteilung

Das vorliegende Dokument beschreibt das angewendete Verfahren zum Testen der Software.

2 Testverfahren

2.1 Server

Serverseitig werden für Entitäten, Mapper, Services und spezielle Klassen (wie das Languagerouting) phpUnit Tests geschrieben. Die Tests werden jeweils manuell vor einem Commit ausgeführt. Vereinbart ist, dass kein Code eingecheckt werden darf, der einen der Unit Tests nicht besteht.
Dadurch wird sichergestellt, dass der jeweilige Entwickler direkt über den Fehler informiert wird und diesen vorgängig behebt.

2.2 Client

Clientseitig werden Benutzertests durchgeführt. Diese beinhalten GUI-Tests, explorative Funktionstests sowie Usability-Tests. Aufgetretenes Fehlverhalten wird festgehalten. Verbesserungen der Usability werden sofern möglich in der weiteren Entwicklung berücksichtigt.
Zusätzlich werden Systemtests durchgeführt, welche Teile von Use Cases prüfen.

2.3 Testinfrastruktur

Die Unit Tests finden direkt in der Entwicklungsumgebung, welche bei jedem Teammitglied vorhanden ist, statt. Die einzelnen Tests werden mit phpUnit geschrieben und getestet.
Für die Client-Tests wird jeweils die aktuellste Produktiv-Version verwendet.

Die Benutzertests werden auf einer Entwicklungsmaschine mit geöffnetem Browser durchgeführt.

3 Testfälle

3.1 Unit Tests

Die Testfälle der Unit Tests decken die jeweiligen Funktionen ab und werden deshalb nicht weiter erläutert.

3.2 Systemtest

In folgender Tabelle sind die einzelnen Testfälle und deren Beschreibungen zu finden. Die Testfälle wurden aus den Use Cases abgeleitet und prüfen die korrekte Funktion des Pflichtumfangs.

Test-Nr.	Beschreibung
T1 - Formular	
T1.1	Ein neues Formular kann angelegt werden
T1.1.1	Der Titel des Formulars kann gesetzt werden
T1.1.2	Die Beschreibung des Formulars kann gesetzt werden
T1.1.3	Der Name kann gesetzt werden
T1.1.4	Die E-Mailadresse kann gesetzt werden
T1.2	Ein Formular kann geändert werden
T1.2.1	Der Titel des Formulars kann geändert werden
T1.2.2	Die Beschreibung des Formulars kann geändert werden
T1.3	Ein Formular kann gelöscht werden
T1.4	Bei der Erstellung eines Formulars wird ein eindeutiger Link für die Verwaltung des Formulars generiert
T1.5	Bei der Erstellung eines Formulars wird ein eindeutiger Link für das Ausfüllen des Formulars generiert
T1.6	Der Manager erhält nach dem Anlegen des Formulars den Verwaltungslink per E-Mail
T1.7	Der Manager wird nach dem Anlegen des Formulars zur Feldverwaltung weitergeleitet
T2 - Feld	
T2.1	In der Feldverwaltung wird dem Manager der Link zur Verwaltung des Formulars angezeigt
T2.2	Ein neues Feld kann angelegt werden

T2.2.1	Der Name des Feldes kann gesetzt werden
T2.2.2	Der Typ des Feldes kann ausgewählt werden
T2.2.3	Der Hilfetext des Feldes kann gesetzt werden
T2.2.4	Die Option "Pflichtfeld" kann aktiviert/deaktiviert werden
T2.2.5	Die Option "Internes Feld" kann aktiviert/deaktiviert werden
T2.3	Ein angelegtes Feld wird in der Feldansicht dargestellt
T2.4	Ein Feld kann bearbeitet werden
T2.4.1	Der Name des Feldes kann bearbeitet werden
T2.4.2	Der Typ des Feldes kann nicht bearbeitet werden
T2.4.3	Der Hilfetext des Feldes kann bearbeitet werden
T2.4.4	Die Option "Pflichtfeld" kann bearbeitet werden
T2.4.5	Die Option "Internes Feld" kann bearbeitet werden
T2.5	Ein Feld kann gelöscht werden
T2.5.1	Alle Einträge zu einem Feld werden gelöscht wenn das Feld gelöscht wird
T3 - Vorschau	
T3.1	Die Vorschau kann nach dem Hinzufügen von Feldern angezeigt werden
T3.2	Der Manager kann die Vorschau abbrechen
T3.3	Der Manager wird bei Abbruch der Vorschau auf die Feldverwaltung weitergeleitet
T4 - Publikation	
T4.1	Der Manager kann das Formular publizieren
T4.2	Der Manager wird nach der Publikation zur Formularverwaltung weitergeleitet
T4.3	Alle Felder sind nach der Publikation des Formulars publiziert
T5 - Formularverwaltung	
T5.1	In der Formularverwaltung wird dem Manager der Gastlink angezeigt
T5.2	In der Formularverwaltung wird dem Manager der Verwaltungslink angezeigt
T6 - Routing	
T6.1	Der Manager gelangt mit dem Verwaltungslink zur Feldverwaltung, falls das Formular noch nie publiziert wurde

T6.2	Der Manager gelangt mit dem Verwaltungslink zur Formularverwaltung, falls das Formular schon einmal publiziert wurde
T7 - Gastansicht	
T7.1	Der Gast hat mit dem Gastlink Zugriff auf das ausfüllbare Formular
T7.2	Dem Gast wird eine Fehlermeldung angezeigt, wenn das Formular (noch) nicht publiziert ist
T7.3	In der Gästeansicht werden keine internen oder nicht publizierten Felder angezeigt
T7.4	Ein Datensatz kann angelegt werden
T7.5	Die Felder in der Gästeansicht werden validiert
T8 - Exportfunktion	
T8.1	Der Manager kann alle Datensätze exportieren
T8.2	Im Export sind sämtliche Datensätze enthalten

3.3 Benutzertests

Mit nachstehendem Auftrag werden Benutzer angewiesen die Applikation zu verwenden.
Die Erkenntnisse sollen zu einem besseren Applikationsfluss und einer intuitiveren Bedienung führen.

Dem Benutzer werden bei Testbeginn folgende Anweisungen gegeben:
„Du möchtest Adgenero nutzen, um ein Mitglieder-Formular für einen Verein zu erstellen. Von den Vereinsmitgliedern benötigst du folgende Angaben: Name, Wohnort, T-Shirtgrösse, E-Mailadresse. Der Vorname wäre ebenfalls interessant, ist aber keine zwingende Angabe.
Bitte schick das Formular an 5 verschiedene Personen zur Erfassung als Mitglied.“

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

Testprotokoll

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren:	Fabian Schweizer, Manuel Schweizer
Betreuer:	Dr. Daniel Keller
Projektpartner:	foryouandyourcustomers AG, Pfäffikon ZH
Experte:	Dr. Rudolf Mattmann
Gegenleser:	Prof. Hansjörg Huser

Dokumentinformationen

Änderungsgeschichte

Datum	Version	Änderung	Autor
25.02.2013	0.1	Template erstellt	ms
30.05.2013	0.2	Benutzertests	ms
07.06.2013	1.0	Unit Tests	fs
12.06.2013	1.1	Überarbeitung und Review	ms
13.06.2013	1.2	Testverlauf Systemtest	fs
14.06.2013	1.3	Last Review	ms

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument beschreibt die Resultate der Testaktivitäten.

1.2 Gültigkeitsbereich

Das Dokument ist während der gesamten Projektdauer gültig und wird periodisch aktualisiert.

1.3 Referenzen

Die folgenden Dokumente werden in diesem Testprotokoll referenziert:

- 01-Aufgabenstellung/Aufgabenstellung.pdf
- 03-Projektplan/Projektplan.pdf
- 04-Anforderungsspezifikation/Anforderungsspezifikation.pdf
- 07-Tests/Testspezifikation.pdf
- 10-Glossar/Glossar.pdf

1.4 Aufteilung

Das vorliegende Dokument geht zuerst auf die Unit Tests ein, geht dann über zu den Integrationstests und befasst sich danach mit den Resultaten der Benutzertests.

2 Unit Tests

Unit Tests werden nach Einsatzgebiet in Controller, Entitäten, Formulare, Mapper und Services unterteilt. Dabei hat jede Kategorie eigene Anforderungen an die Testfälle.

2.1 Controller

Im Bereich der Controller werden die Methoden des BasicControllers und das Routing der Sprache getestet.

2.2 Entitäten

Bei jeder Entität wird geprüft ob diese korrekt initialisiert wird. Zusätzlich wird geprüft ob Setter und Getter mit gültigen Werten funktionieren und ob beim Setzen von ungültigen Werten der entsprechende Fehler geworfen wird. Für sämtliche weiteren Funktionen wird jeweils ein eigener Testfall geschrieben.

2.3 Formulare

Da Formulare von Zend Framework 2 bereits getestet wurden werden diese lediglich in kleinem Rahmen geprüft. Es wird überprüft ob nach der Initialisierung die Anzahl der Elemente im Formular korrekt ist sowie die grundlegende Funktion getType von Elementen.

2.4 Mapper

Mapper werden komplett überprüft indem geprüft wird ob die Funktionen insert, update und delete korrekt funktionieren. Für den erfolgreichen Test der Mapper wird eine Datenbankanbindung benötigt. Es wird kein Mocking eingesetzt.

2.5 Services

Services sind wichtige Bestandteile der Applikation und werden daher geprüft. Es wird insbesondere Wert auf die Funktionen des FormGenerators und FieldtypeSelectors gelegt.

3 Systemtests

3.1 Angaben zur Durchführung

Mit der im Rahmen dieser Bachelorarbeit erstellten finalen Version wurde ein kompletter Systemtest durchgeführt. Die Ergebnisse dieses Tests sind nachfolgend dokumentiert.

3.2 Protokoll

Test-Nr.	Implementiert	Fehler / Unschönheit	Status
T1.1	Ja		✔
T1.1.1	Ja		✔
T1.1.2	Ja		✔
T1.1.3	Ja		✔
T1.1.4	Ja		✔
T1.2	Ja		✔
T1.2.1	Ja		✔
T1.2.2	Ja		✔
T1.3	Ja	Beim Löschen wird ein Fehler geworfen, da der Guest-Hash nicht gelöscht werden konnte. Problem wird vor der Präsentation noch behoben.	✗
T1.4	Ja		✔
T1.5	Ja		✔
T1.6	Ja		✔
T1.7	Ja		✔
T2.1	Ja		✔
T2.2	Ja		✔
T2.2.1	Ja		✔
T2.2.2	Ja		✔
T2.2.3	Ja		✔
T2.2.4	Ja		✔
T2.2.5	Ja		✔

T2.3	Ja		✔
T2.4	Ja		✔
T2.4.1	Ja		✔
T2.4.2	Ja		✔
T2.4.3	Ja		✔
T2.4.4	Ja		✔
T2.4.5	Ja		✔
T2.5	Ja		✔
T2.5.1	Ja		✔
T3.1	Ja		✔
T3.2	Ja		✔
T3.3	Ja		✔
T4.1	Ja		✔
T4.2	Ja		✔
T4.3	Ja		✔
T5.1	Ja		✔
T5.2	Ja		✔
T6.1	Ja		✔
T6.2	Ja		✔
T7.1	Ja		✔
T7.2	Ja		✔
T7.3	Ja		✔
T7.4	Ja		✔
T7.5	Ja	Fehlermeldungen sind nicht übersetzt	✔
T8.1	Ja		✔
T8.2	Ja		✔

4 Benutzertests

Gegen Ende der Arbeit wurden diverse Benutzertests durchgeführt. Die Erkenntnisse aus diesen Tests sollen zu einem besseren Applikationsfluss und einer intuitiveren Bedienung führen.

4.1 Testpersonen

Es wurden insgesamt 5 Benutzertests durchgeführt:

Name (*)	J.J.	S.M.	R.G.	M.G.	H.M.
Altersgruppe	15-20	20-30	30-40	20-30	60-70
Officekenntnisse	gut	gut	sehr gut	sehr gut	sehr gut
Computerkenntnisse	Anwenderin	Anwenderin	Anwender	Experte	Experte
Testdatum	20.05.13	20.05.13	25.05.13	01.06.13	08.06.13
Testdauer	15 Min.	20 Min.	10 Min.	15 Min.	10 Min.

(*) Vollständiger Name bekannt

4.2 Insights

Feld hinzufügen
Obwohl der „add“-Button ziemlich prominent platziert wurde, wünschten sich die Testpersonen einen speziellen Hinweis darauf (z.B. ein Pfeil mit Erklärung), dass man damit Formularfelder erstellt. Sobald das erste Feld erstellt war, gab es keine Unklarheiten mehr, aber das initiale Feld bereitete Mühe.

Sortierung Feldauswahl
Bei der Auswahl möglicher Felder wurde die vorgegebene Sortierung bemängelt. Da diese alphabetisch ist, berücksichtigt sie nicht, dass gewisse Felder, z.B. ein normales Textfeld, deutlich öfter verwendet wird als ein Spezialfeld wie z.B. E-Mail.

Pflichtfeld automatisch
Die Benutzer ärgerten sich darüber, dass bei der Erstellung der Felder die Funktion „Pflichtfeld“ jeweils manuell ausgewählt werden muss. Es wurde argumentiert, dass bei einem Formular meistens mehr Felder Pflichtangaben sind als freiwillige Angaben.

Massen-Hinzufügen
Gerade im Vergleich zur Erstellung einer Excel-Tabelle hinkt Adgenero beim Hinzufügen weiterer Felder hinterher. Eine Ursache dafür ist die fehlende Möglichkeit, mehrere Felder nacheinander hinzuzufügen anstatt jeweils in der Übersichtsmaske zu landen.
Ein entsprechender Featurerequest „Hinzufügen und weiter“ existierte bereits vor dem Benutzertest, konnte aus zeitlichen Gründen aber nicht mehr implementiert werden.

Vorschau überflüssig

Die Benutzer haben nicht verstanden, wozu der Schritt „Vorschau“ gut sein soll. Auf Rückfrage konstatierten die meisten, dass die WYSIWYG-Ansicht der Formularmaske ja bereits eine detaillierte Vorschau bietet und deshalb der zusätzliche Schritt über die 1:1 Vorschau unnötig sei.
Ein guter Vorschlag war, dem Benutzer die Wahl zwischen Live-Vorschau und direkter Publikation zu geben.

Feldtyp anpassen
Die fehlende Möglichkeit, den Feldtyp nachträglich anzupassen, sorgte ebenfalls für Verwirrung.
Der Grund dafür ist technischer Natur: Sollten zu einem bestimmten Feld bereits Einträge in der Datenbank existieren, müsste eine entsprechende Typkonversion vorgenommen werden. Womöglich wäre ein zuvor valider Eintrag nach der Anpassung nicht mehr ein gültiger Wert für dieses Feld.

4.3 Konsequenzen

Ein Teil der Änderungswünsche wäre mit vertretbarem Aufwand umsetzbar, sprengt aber den zeitlichen Rahmen dieser Bachelorarbeit. Selbstverständlich werden wir Kritikpunkte, die während der Arbeit nicht mehr berücksichtigt werden können, bei einer allfälligen Fortsetzung oder Neuentwicklung nach dem Studium beherzigen.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

Entwicklungsumgebung

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren: Fabian Schweizer, Manuel Schweizer
Betreuer: Dr. Daniel Keller
Projektpartner: foryouandyourcustomers AG, Pfäffikon ZH
Experte: Dr. Rudolf Mattmann
Gegenleser: Prof. Hansjörg Huser

Dokumentinformationen

Änderungsgeschichte

Datum	Version	Änderung	Autor
01.05.2013	0.1	Erste Fassung	ms
14.06.2013	1.0	Release	fs
14.06.2013	1.1	Last Review	ms

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument beschreibt die nötigen Schritte zum Aufbau einer Entwicklungsumgebung für das Projekt Adgenero.

1.2 Gültigkeitsbereich

Das Dokument ist während der gesamten Projektdauer gültig und wird periodisch aktualisiert.

1.3 Aufteilung

Das vorliegende Dokument geht zuerst auf die Anforderungen an die Entwicklungsumgebung ein und bietet danach eine Schritt-für-Schritt Anleitung zur Installation.

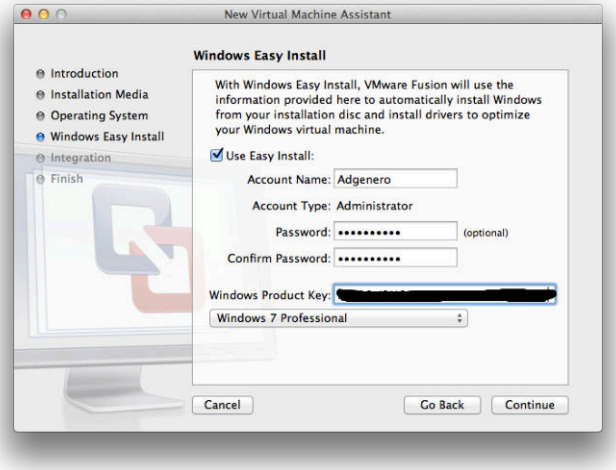
2 Grundlage

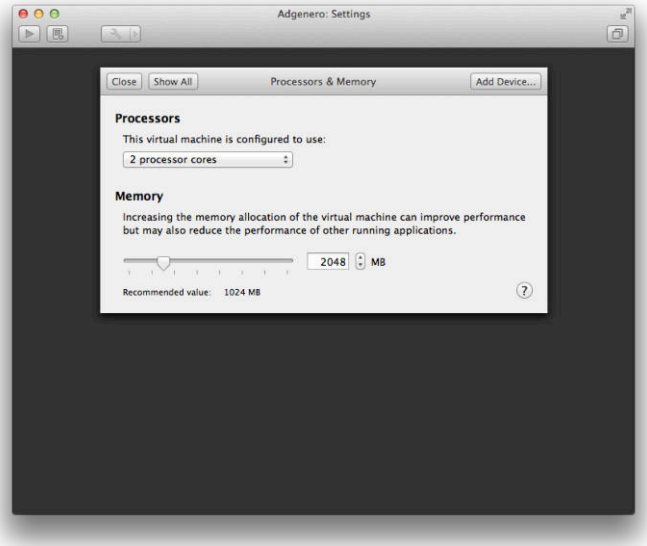
Grundlage für das Adgenero SDE (Software Development Environment) ist folgendes Setup:

- Windows 7 Professional 32-bit
- Eclipse 4.2.2 Classic 32-bit
- Java Runtime Environment 1.7.0_21
- XAMPP for Windows 1.8.1
- Google Chrome
- Mozilla Firefox
- Opera

3 Vorgang

3.1 Betriebssystem installieren





3.2 Eclipse installieren

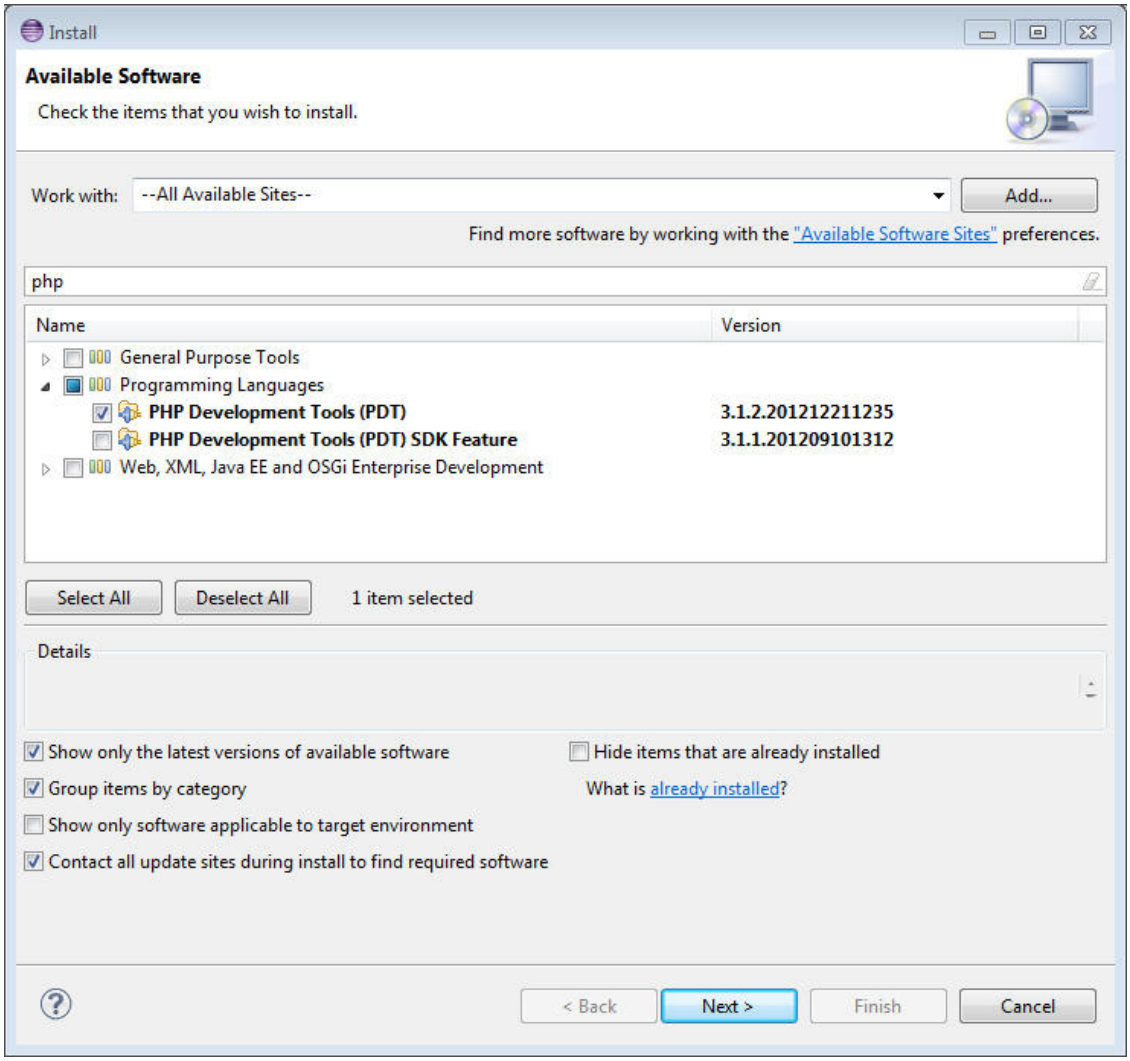
Eclipse muss zuerst heruntergeladen werden. Das heruntergeladene Archiv kann extrahiert und eclipse ohne weitere Installation gestartet werden.

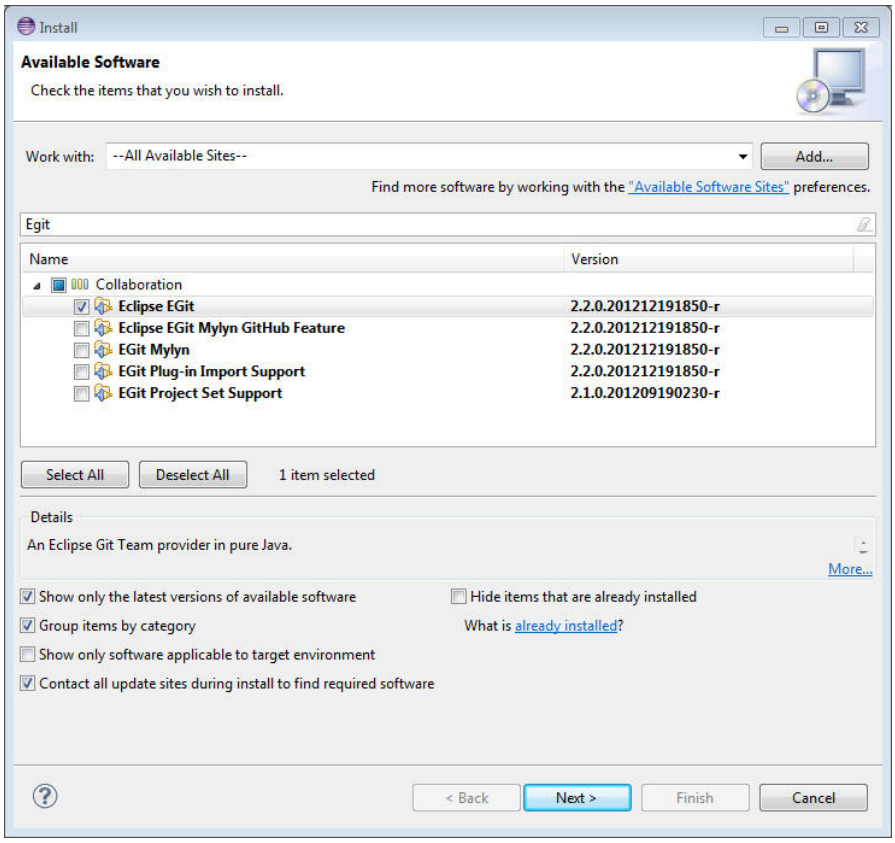
Es müssen die Plugins PDT, EGit, MakeGood und FileSync installiert werden. Diese sind wie nachstehend angegeben zu finden:

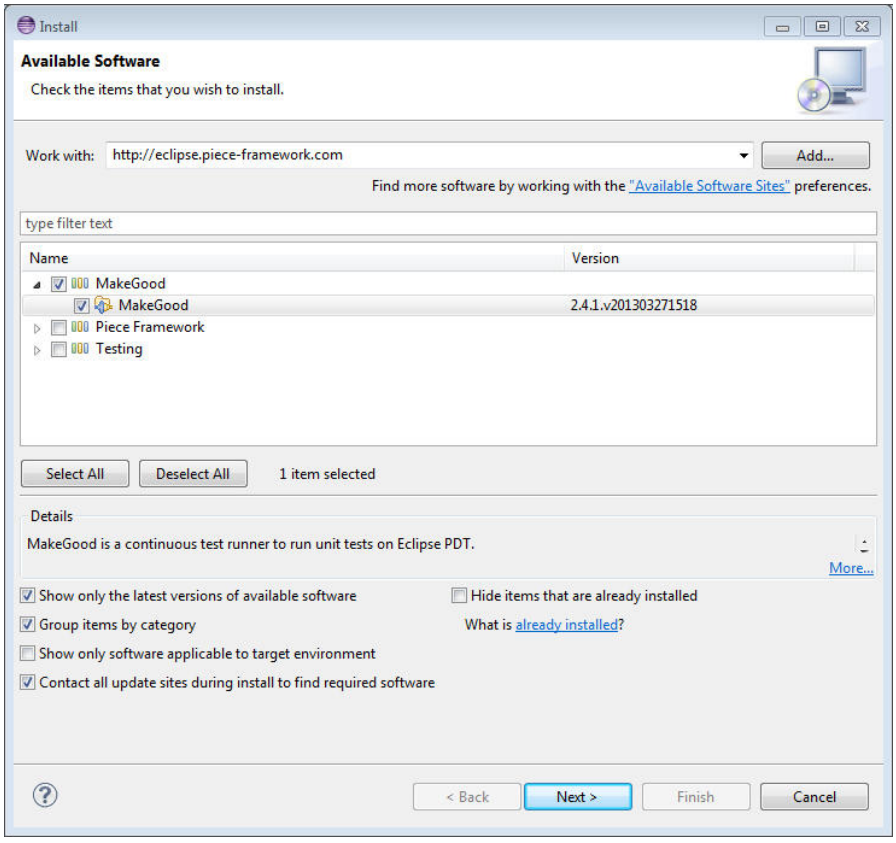
Help -> Install new Software -> All Sites
- Programming Languages => PHP Development Tools (PDT)
- Collaboration => Eclipse EGit

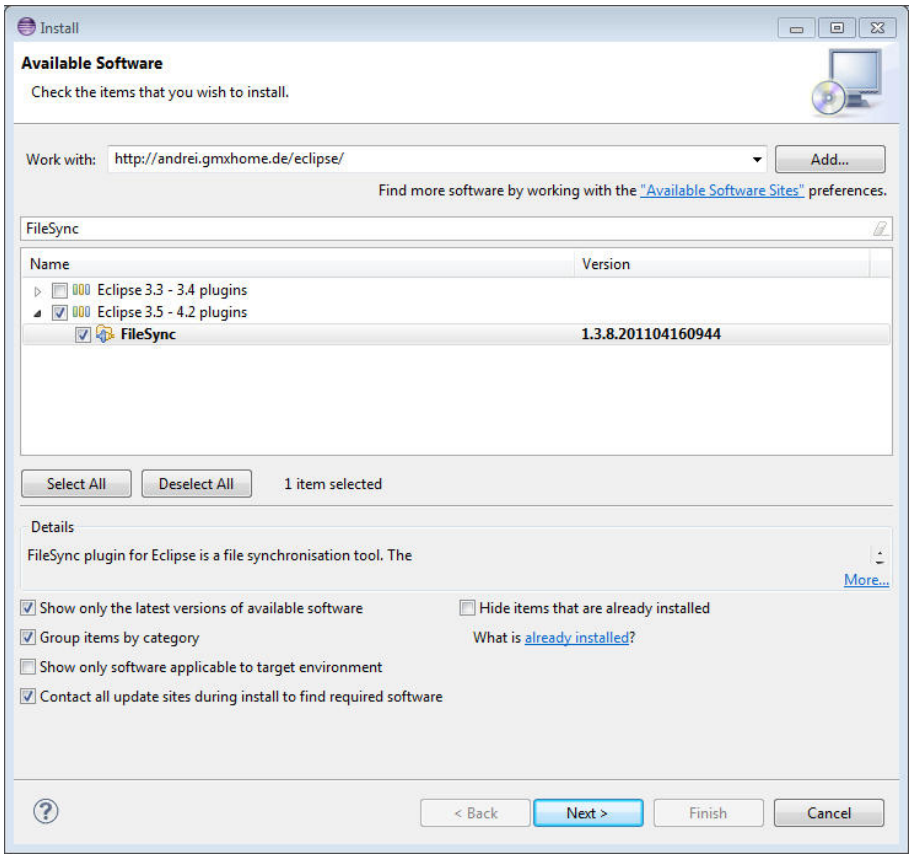
Help -> Install new Software -> http://eclipse.piece-framework.com
- MakeGood => MakeGood

Help -> Install new Software -> http://andrei.gmxhome.de/eclipse/
- Eclipse 3.5-4.2 Plugins => FileSync



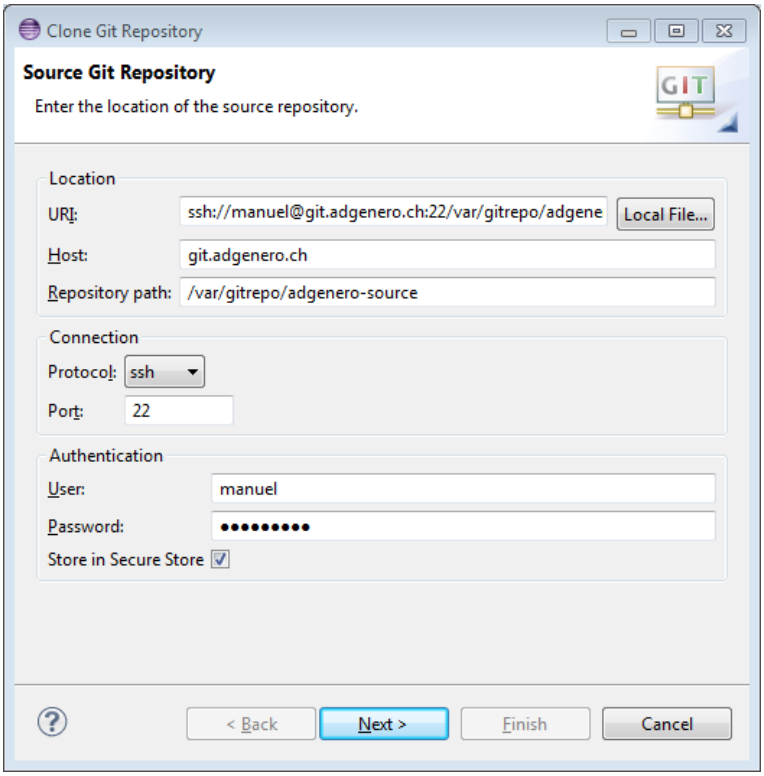

Install
Available Software
Check the items that you wish to install.
Work with: --All Available Sites-- Add...
Find more software by working with the "Available Software Sites" preferences.
Egit
Name Version
Collaboration
Eclipse EGit 2.2.0.201212191850-r
Eclipse EGit Mylyn GitHub Feature 2.2.0.201212191850-r
EGit Mylyn 2.2.0.201212191850-r
EGit Plug-in Import Support 2.2.0.201212191850-r
EGit Project Set Support 2.1.0.201209190230-r
Select All Deselect All 1 item selected
Details
An Eclipse Git Team provider in pure Java.
More...
Show only the latest versions of available software
Hide items that are already installed
Group items by category
What is already installed?
Show only software applicable to target environment
Contact all update sites during install to find required software
< Back Next > Finish Cancel

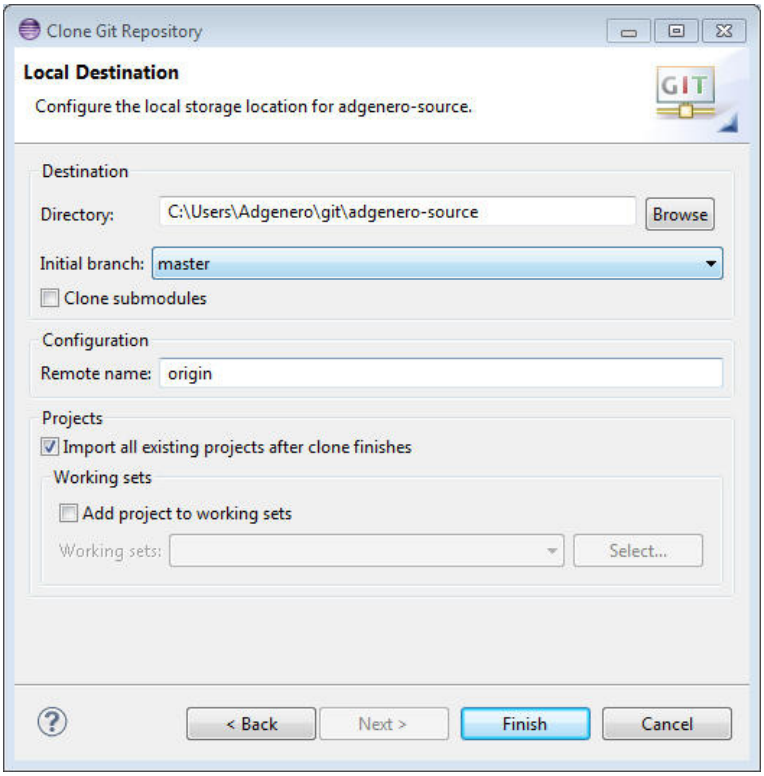

Install
Available Software
Check the items that you wish to install.
Work with: http://eclipse.piece-framework.com Add...
Find more software by working with the "Available Software Sites" preferences.
type filter text
Name Version
MakeGood
MakeGood 2.4.1.v201303271518
Piece Framework
Testing
Select All Deselect All 1 item selected
Details
MakeGood is a continuous test runner to run unit tests on Eclipse PDT.
More...
Show only the latest versions of available software
Hide items that are already installed
Group items by category
What is already installed?
Show only software applicable to target environment
Contact all update sites during install to find required software
< Back Next > Finish Cancel

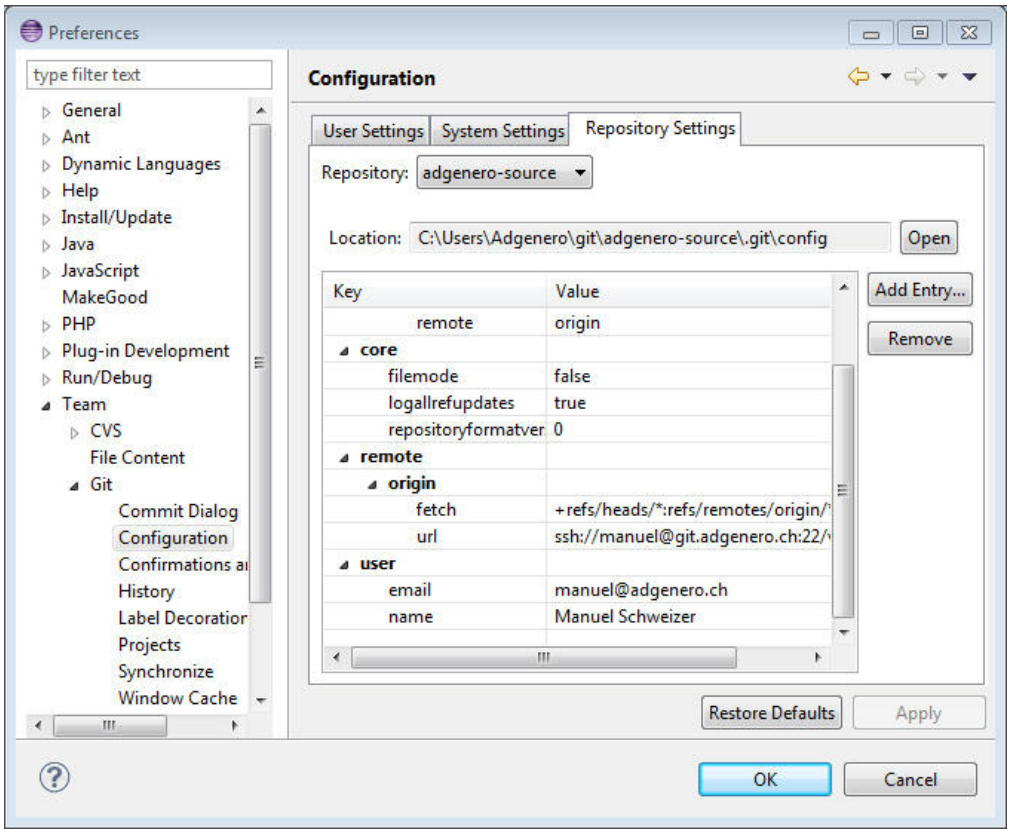

Install
Available Software
Check the items that you wish to install.
Work with:
http://andrei.gmxhome.de/eclipse/
Add...
Find more software by working with the "Available Software Sites" preferences.
FileSync
Name
Version
Eclipse 3.3 - 3.4 plugins
Eclipse 3.5 - 4.2 plugins
FileSync
1.3.8.201104160944
Select All
Deselect All
1 item selected
Details
FileSync plugin for Eclipse is a file synchronisation tool. The
More...
Show only the latest versions of available software
Hide items that are already installed
Group items by category
What is already installed?
Show only software applicable to target environment
Contact all update sites during install to find required software
< Back
Next >
Finish
Cancel

3.3 Git einrichten

Das Git-Repository muss eingerichtet werden. Dazu gilt es dieses zu klonen und die entsprechenden Commit-Einstellungen zu tätigen.

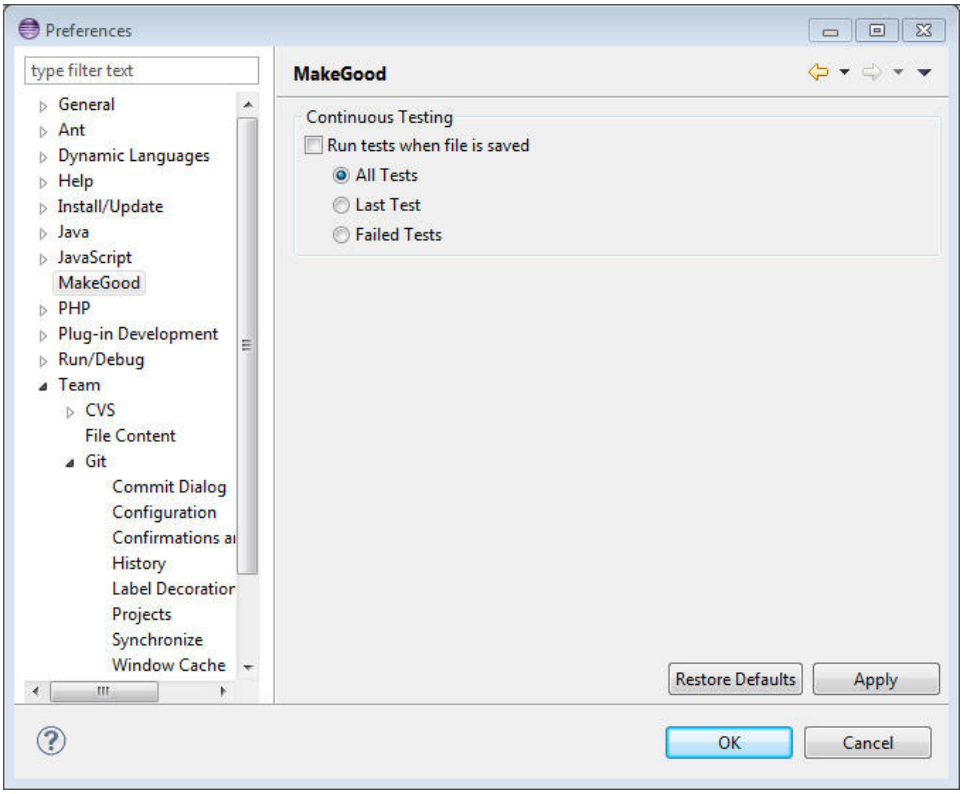


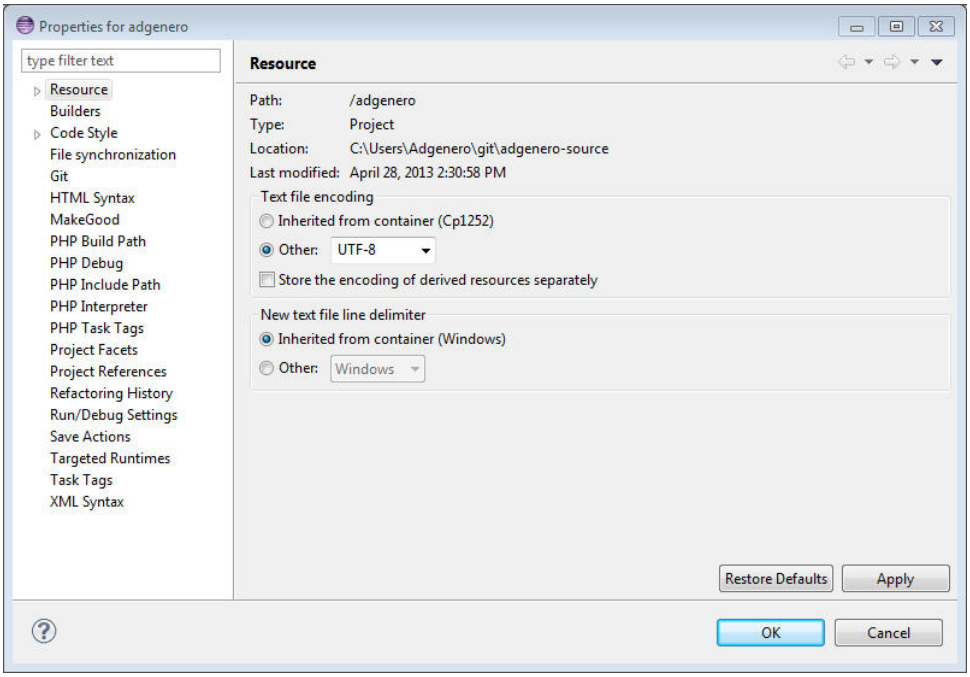


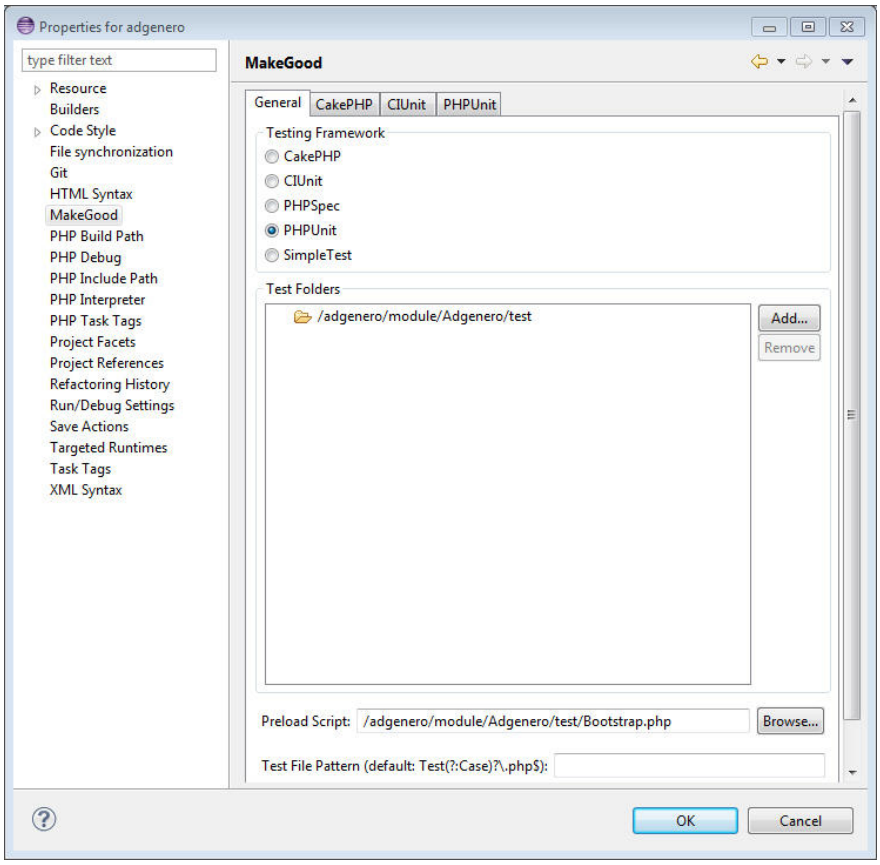

Preferences
type filter text
General
Ant
Dynamic Languages
Help
Install/Update
Java
JavaScript
MakeGood
PHP
Plug-in Development
Run/Debug
Team
CVS
File Content
Git
Commit Dialog
Configuration
Confirmations a
History
Label Decoratior
Projects
Synchronize
Window Cache
Configuration
User Settings
System Settings
Repository Settings
Repository: adgenero-source
Location: C:\Users\Adgenero\git\adgenero-source\.git\config
Open
Key
Value
remote
origin
core
filemode
false
logallrefupdates
true
repositoryformatver
0
remote
origin
fetch
+refs/heads/*:refs/remotes/origin/*
url
ssh://manuel@git.adgenero.ch:22/
user
email
manuel@adgenero.ch
name
Manuel Schweizer
Add Entry...
Remove
Restore Defaults
Apply
OK
Cancel

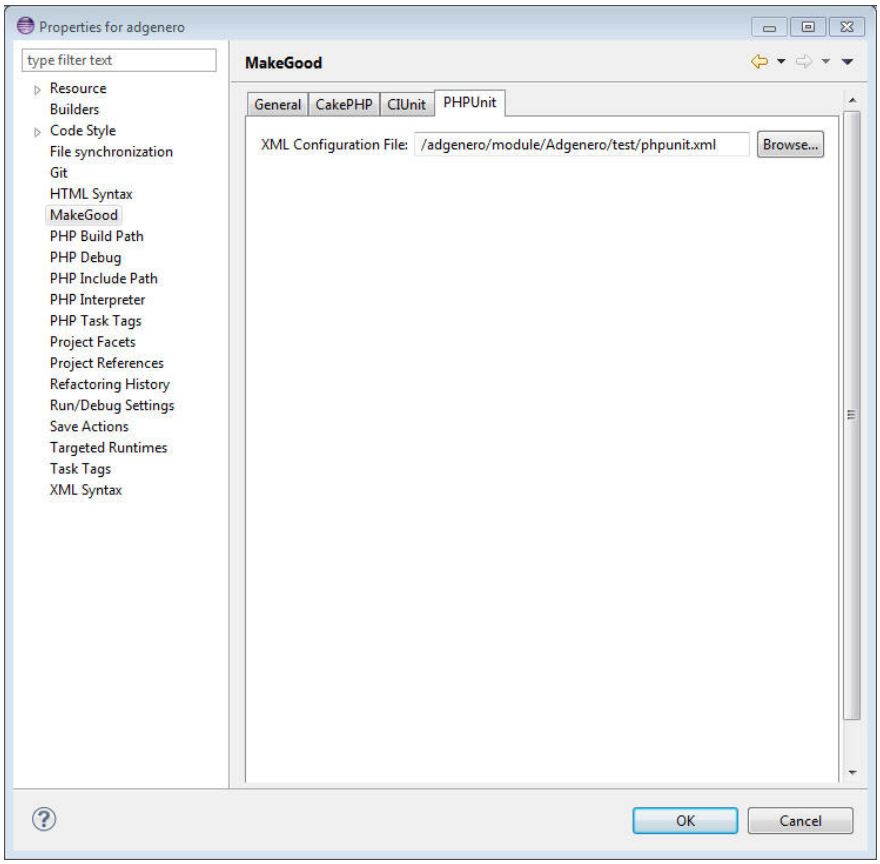
3.4 MakeGood konfigurieren

MakeGood wird verwendet für Unit Testing und muss entsprechend konfiguriert werden wie nachstehend augezeigt.





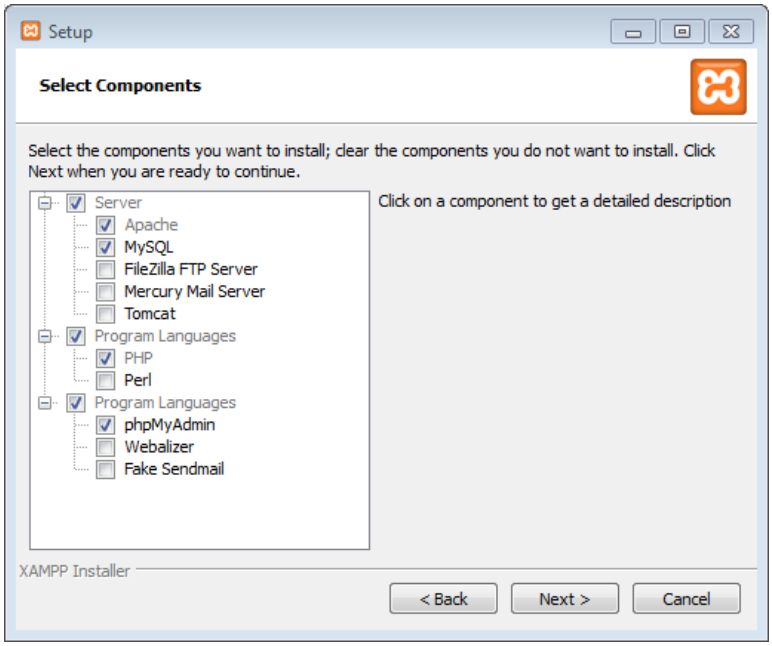

Properties for adgenero
type filter text
Resource
Builders
Code Style
File synchronization
Git
HTML Syntax
MakeGood
PHP Build Path
PHP Debug
PHP Include Path
PHP Interpreter
PHP Task Tags
Project Facets
Project References
Refactoring History
Run/Debug Settings
Save Actions
Targeted Runtimes
Task Tags
XML Syntax
MakeGood
General
CakePHP
CIUnit
PHPUnit
Testing Framework
CakePHP
CIUnit
PHPSpec
PHPUnit
SimpleTest
Test Folders
/adgenero/module/Adgenero/test
Add...
Remove
Preload Script:
/adgenero/module/Adgenero/test/Bootstrap.php
Browse...
Test File Pattern (default: Test(?:Case)?\.php$):
OK
Cancel

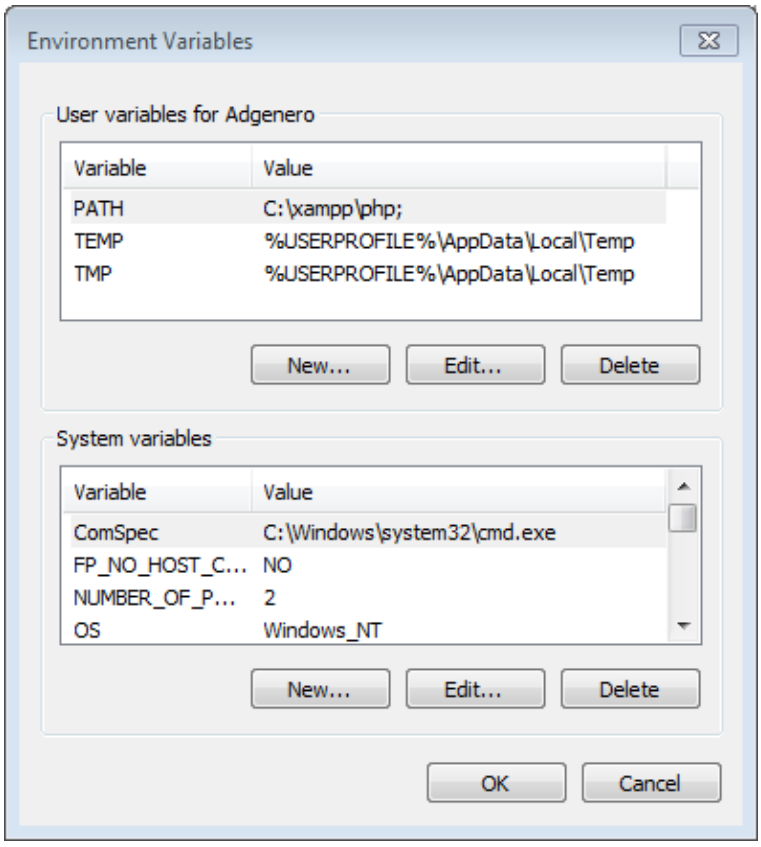

Properties for adgenero
type filter text
Resource
Builders
Code Style
File synchronization
Git
HTML Syntax
MakeGood
PHP Build Path
PHP Debug
PHP Include Path
PHP Interpreter
PHP Task Tags
Project Facets
Project References
Refactoring History
Run/Debug Settings
Save Actions
Targeted Runtimes
Task Tags
XML Syntax
MakeGood
General
CakePHP
CIUnit
PHPUnit
XML Configuration File:
/adgenero/module/Adgenero/test/phpunit.xml
Browse...
OK
Cancel

3.5 XAMPP installieren

XAMPP bietet einen AMP-Stack für diverse Plattformen. Dieses Programm kann heruntergeladen und muss wie beschrieben installiert werden.

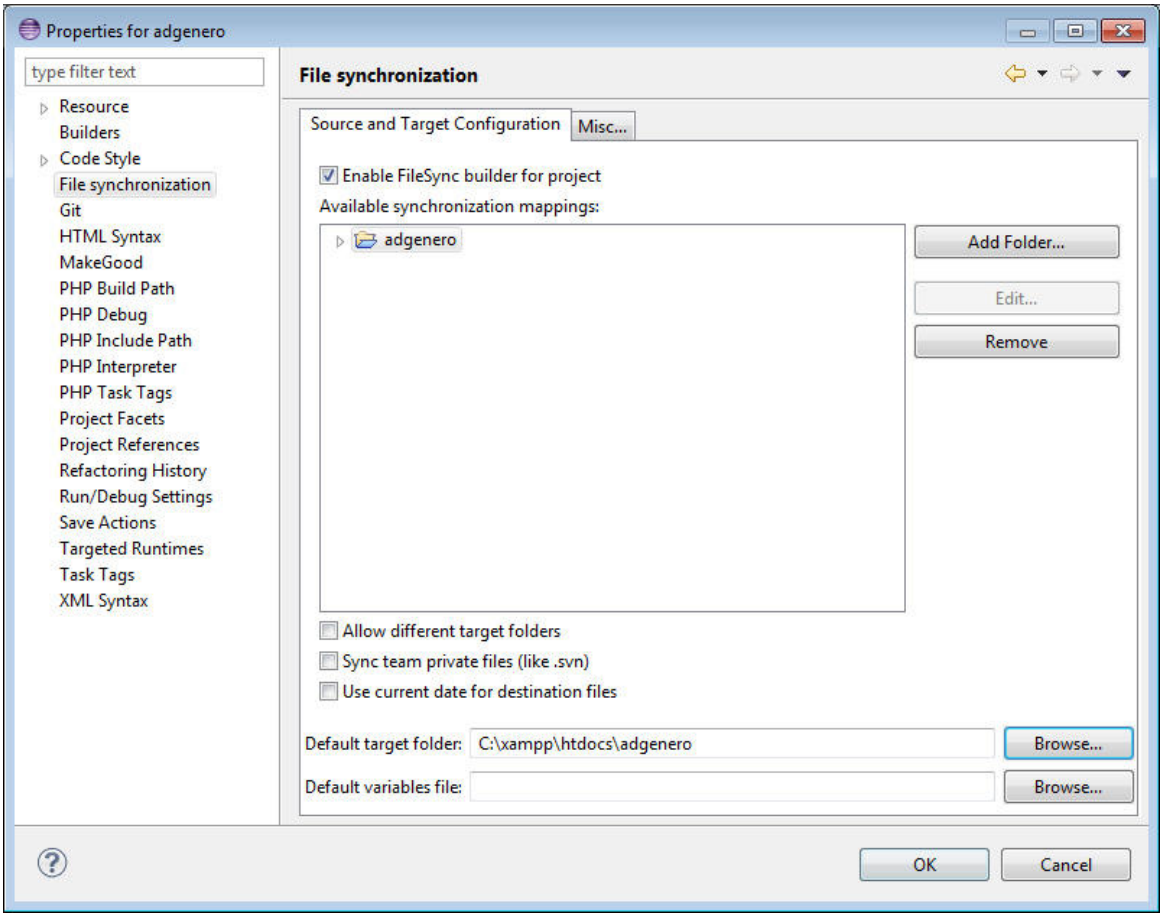
Des Weiteren muss die Umgebungsvariable PATH gesetzt werden wie folgt:
Rechtsklick auf System => Erweiterte Systemeinstellungen => Umgebungsvariablen





3.6 Filesynchronisation einrichten

Damit XAMPP genutzt werden kann sollen die Projektdateien in das Webverzeichnis synchronisiert werden. Dies wird wie folgt eingestellt:

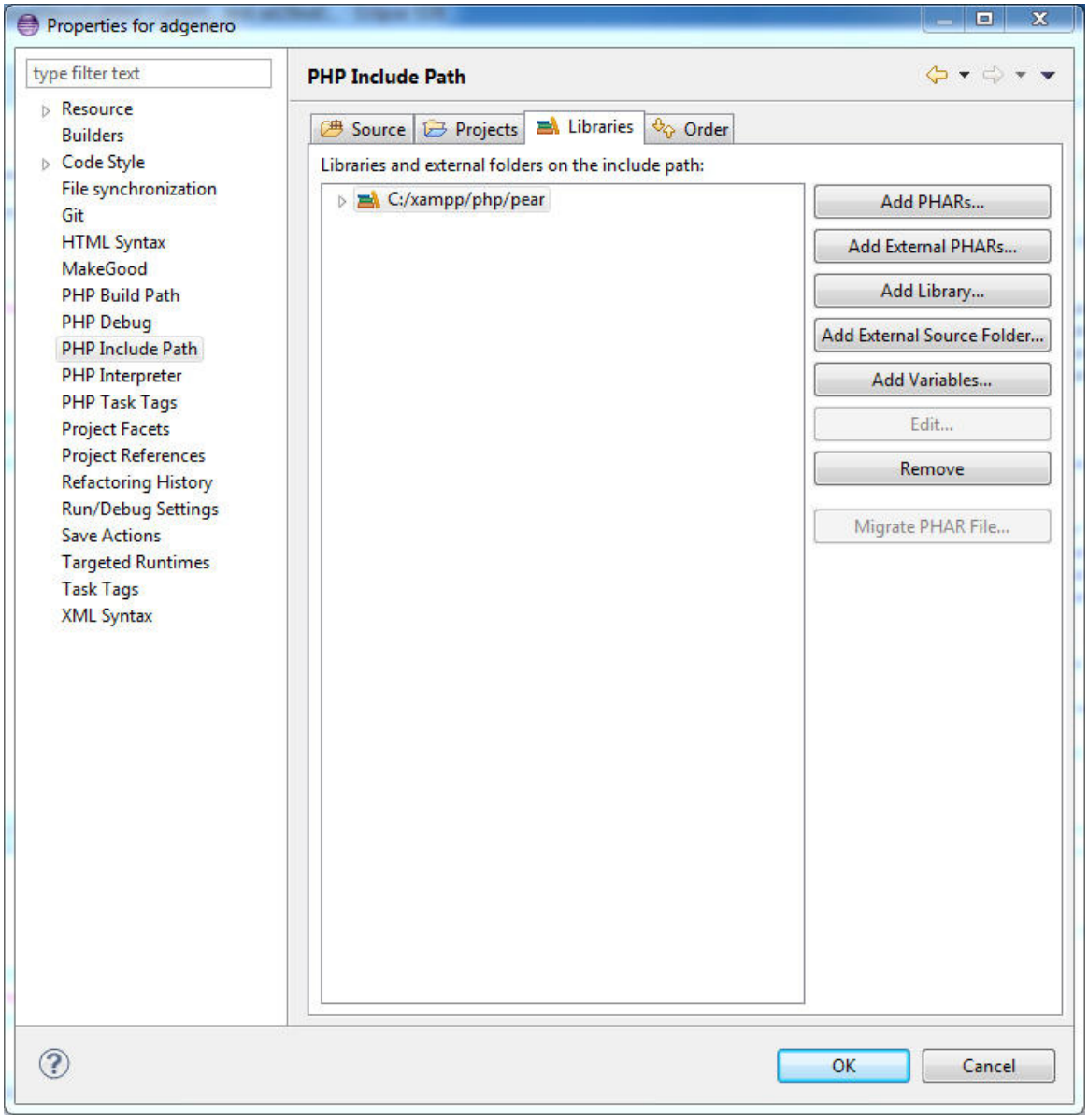


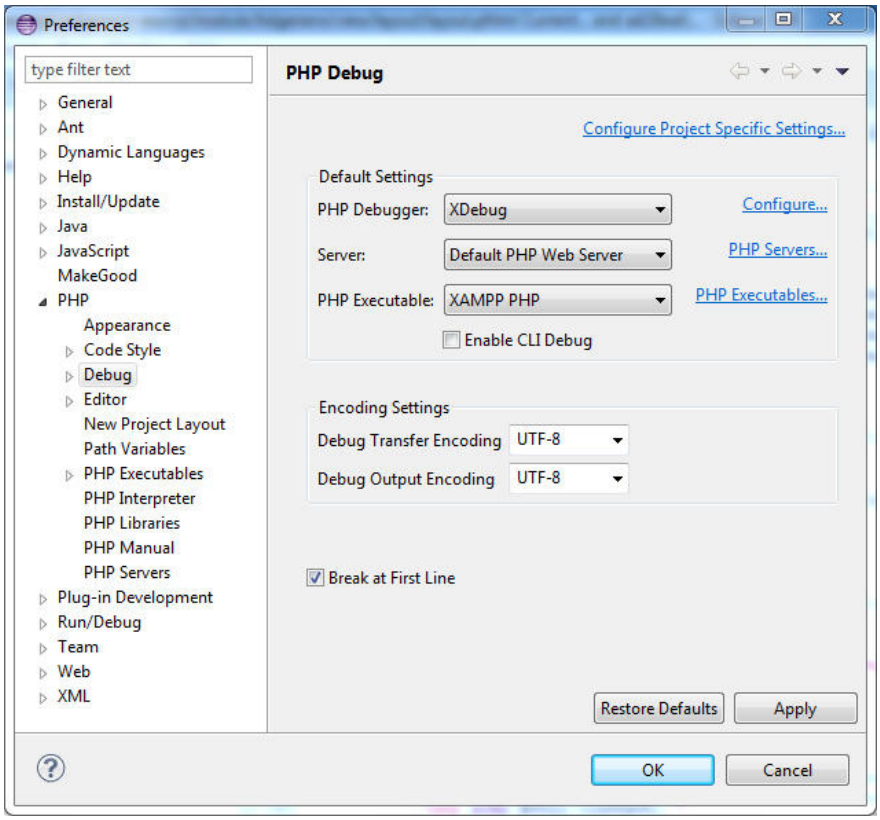
3.7 PHPUnit einrichten

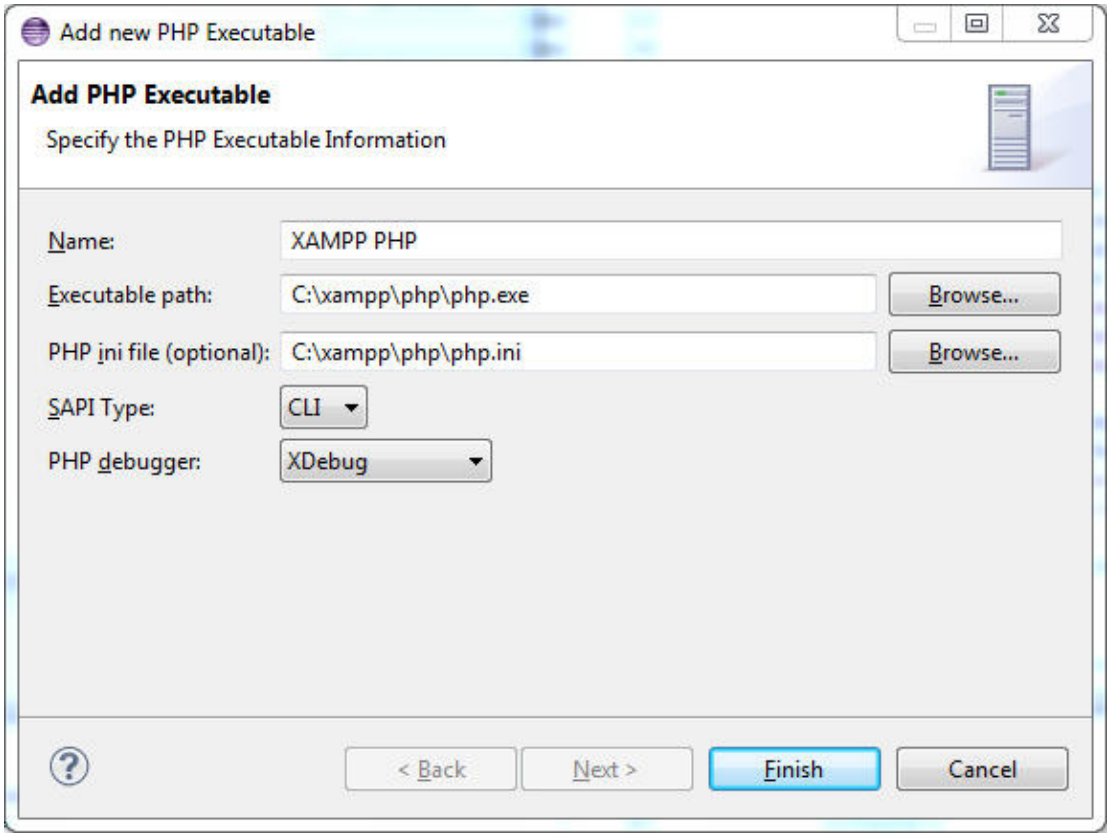
Um die Tests von MakeGood durchlaufen zu lassen muss PHPUnit entsprechend konfiguriert werden.

PEAR: PHP Include Path => Add External Source Folder
PHP => PHP Executables => Add

XDebug
XAMPP PHP
php.exe
php.ini




Preferences
type filter text
General
Ant
Dynamic Languages
Help
Install/Update
Java
JavaScript
MakeGood
PHP
Appearance
Code Style
Debug
Editor
New Project Layout
Path Variables
PHP Executables
PHP Interpreter
PHP Libraries
PHP Manual
PHP Servers
Plug-in Development
Run/Debug
Team
Web
XML
PHP Debug
Configure Project Specific Settings...
Default Settings
PHP Debugger: XDebug
Configure...
Server: Default PHP Web Server
PHP Servers...
PHP Executable: XAMPP PHP
PHP Executables...
Enable CLI Debug
Encoding Settings
Debug Transfer Encoding UTF-8
Debug Output Encoding UTF-8
Break at First Line
Restore Defaults
Apply
OK
Cancel


Add new PHP Executable
Add PHP Executable
Specify the PHP Executable Information
Name: XAMPP PHP
Executable path: C:\xampp\php\php.exe
Browse...
PHP ini file (optional): C:\xampp\php\php.ini
Browse...
SAPI Type: CLI
PHP debugger: XDebug
< Back
Next >
Finish
Cancel

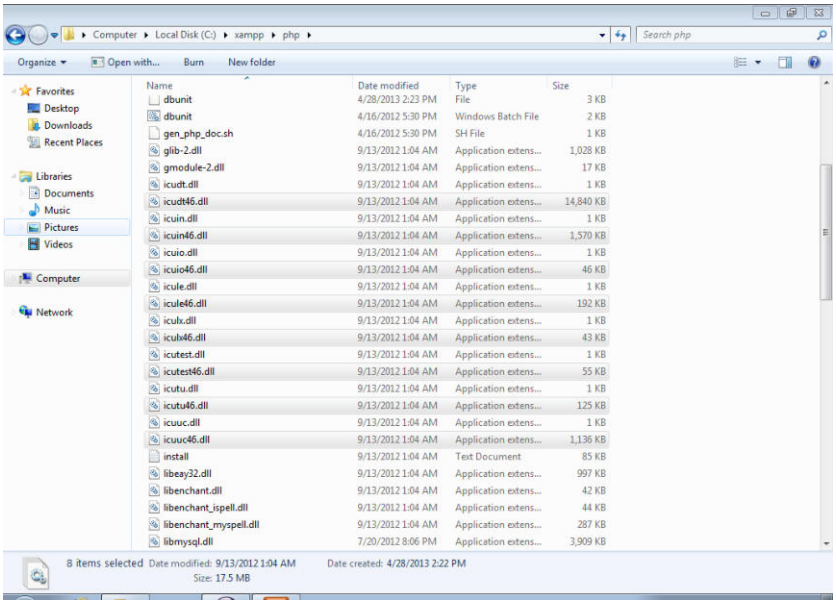
3.8 Apache konfigurieren

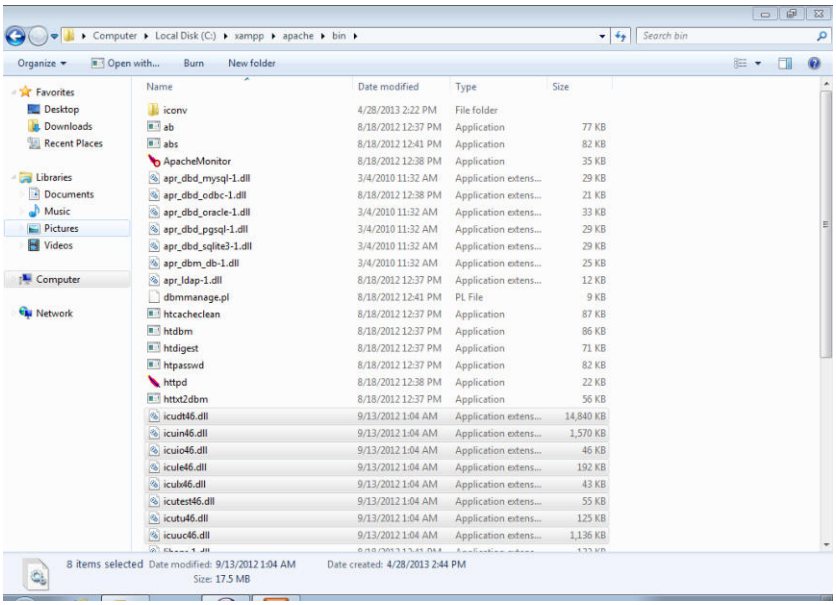
Der Webserver selbst benötigt ebenfalls noch eine Konfiguration.
Die Dateien und Werte wurden unten angegeben.
Zusätzlich müssen gewisse DLLs kopiert werden.

httpd.conf: DocumentRoot "C:/xampp/htdocs/adgenero/public"

apache.conf
DocumentRoot "C:/xampp/htdocs/adgenero/public"

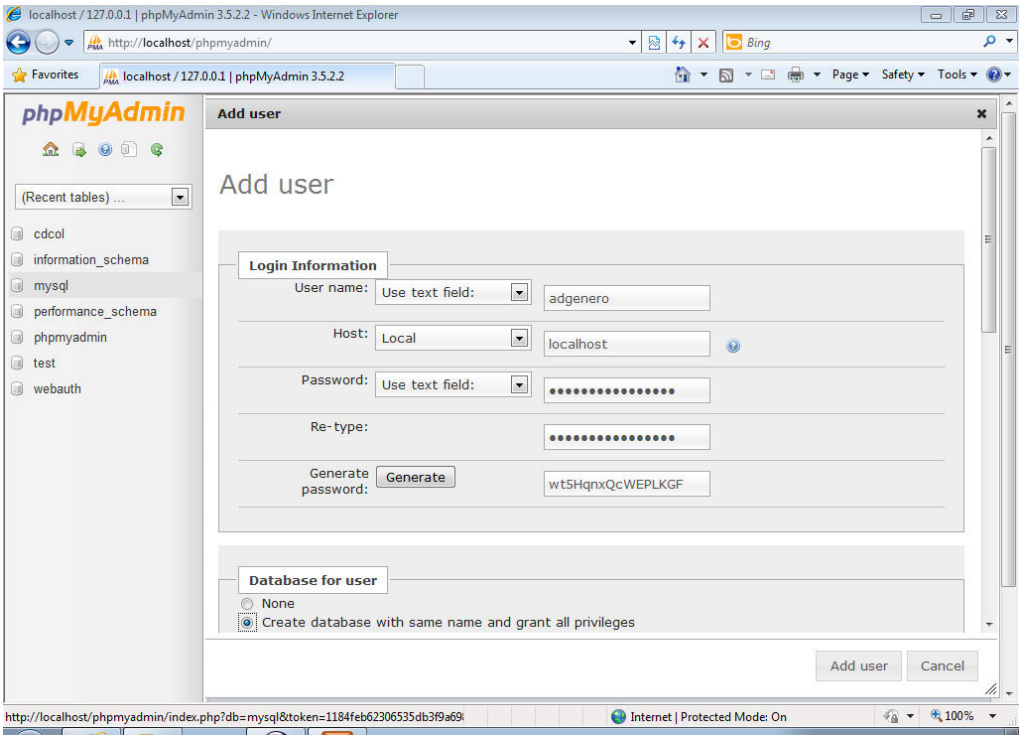
php.ini:
extension=php_intl.dll und
extension=php_openssl.dll aktivieren

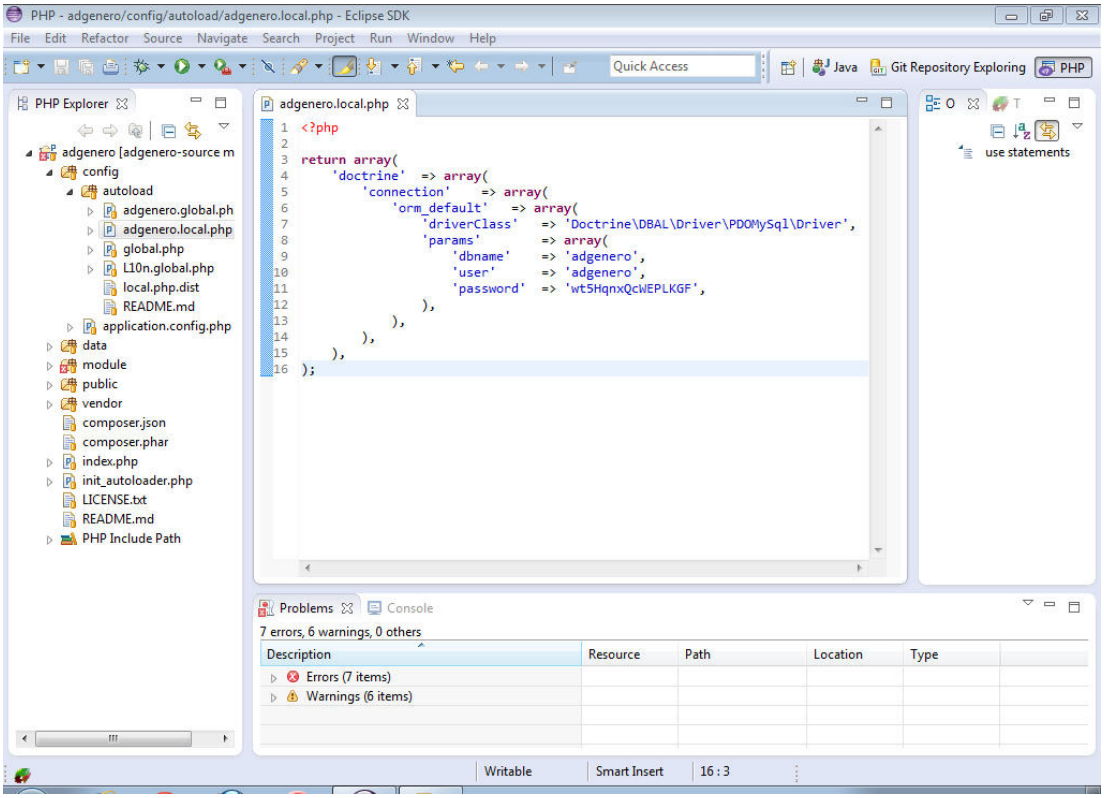




3.9 Datenbank erstellen

Die Datenbank muss eingerichtet werden. Dazu werden ein Benutzer und eine Datenbank in phpMyAdmin angelegt. Die Angaben müssen im Projekt hinterlegt werden. Zusätzlich muss noch die “database.sql”-Datei aus dem Projekt in die erstellte Datenbank importiert werden.

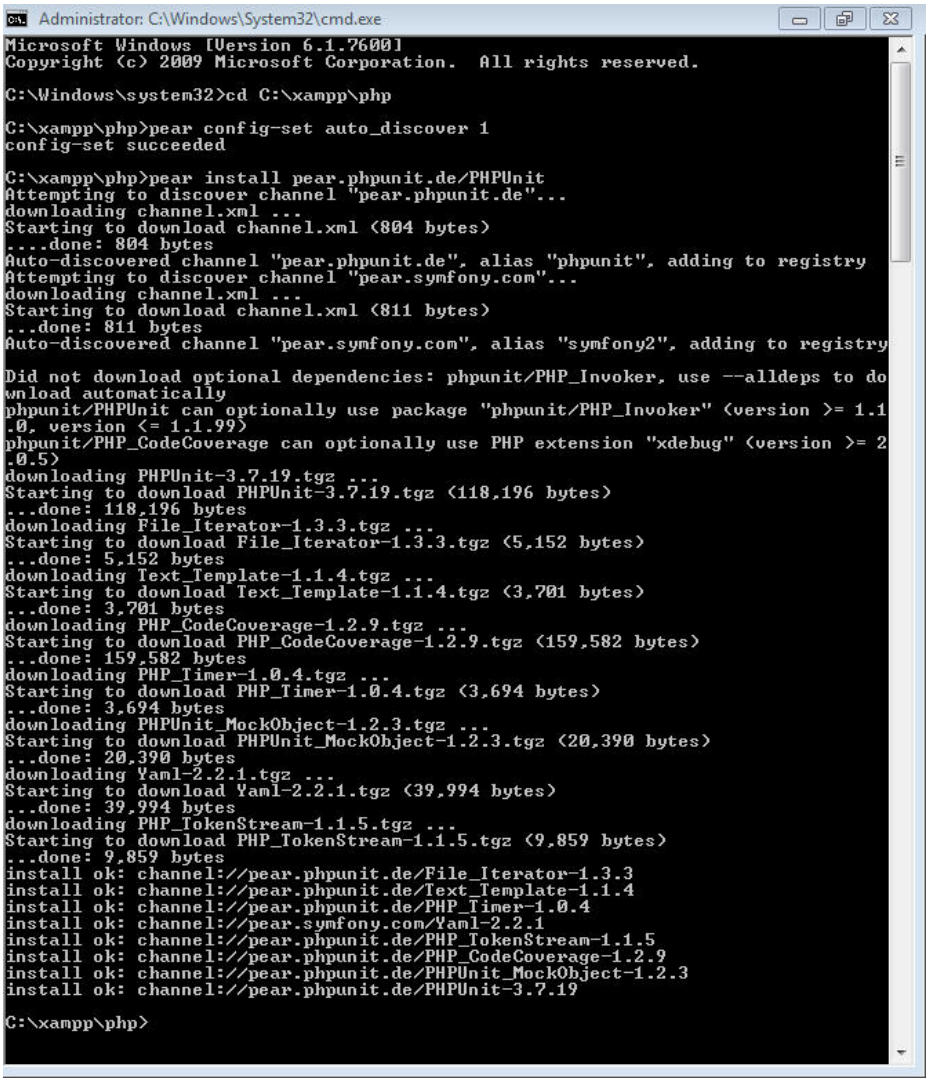




3.10 Composer & PEAR

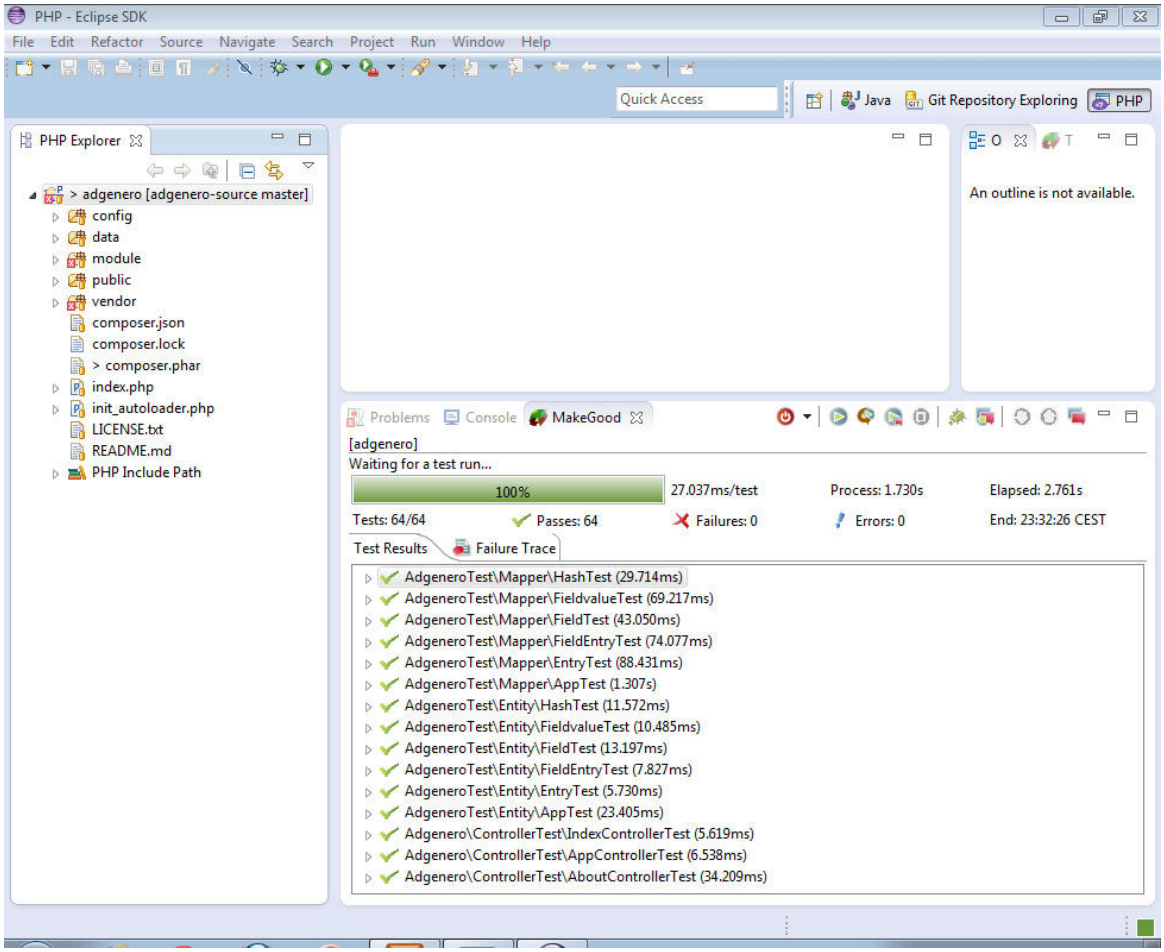
Es müssen noch alle Abhängigkeiten mit Composer geladen und PHPUnit über PEAR aktualisiert werden wie folgt beschrieben.

http://pear.php.net/go-pear.phar
pear config-set auto_discover 1
pear install pear.phpunit.de/PHPUnit


Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Windows\system32>cd C:\xampp\php
C:\xampp\php>pear config-set auto_discover 1
config-set succeeded
C:\xampp\php>pear install pear.phpunit.de/PHPUnit
Attempting to discover channel "pear.phpunit.de"...
downloading channel.xml ...
Starting to download channel.xml (804 bytes)
....done: 804 bytes
Auto-discovered channel "pear.phpunit.de", alias "phpunit", adding to registry
Attempting to discover channel "pear.symfony.com"...
downloading channel.xml ...
Starting to download channel.xml (811 bytes)
...done: 811 bytes
Auto-discovered channel "pear.symfony.com", alias "symfony2", adding to registry
Did not download optional dependencies: phpunit/PHP_Invoker, use --alldeps to do
wnload automatically
phpunit/PHPUnit can optionally use package "phpunit/PHP_Invoker" (version >= 1.1
.0, version <= 1.1.99)
phpunit/PHP_CodeCoverage can optionally use PHP extension "xdebug" (version >= 2
.0.5)
downloading PHPUnit-3.7.19.tgz ...
Starting to download PHPUnit-3.7.19.tgz (118,196 bytes)
...done: 118,196 bytes
downloading File_Iterator-1.3.3.tgz ...
Starting to download File_Iterator-1.3.3.tgz (5,152 bytes)
...done: 5,152 bytes
downloading Text_Template-1.1.4.tgz ...
Starting to download Text_Template-1.1.4.tgz (3,701 bytes)
...done: 3,701 bytes
downloading PHP_CodeCoverage-1.2.9.tgz ...
Starting to download PHP_CodeCoverage-1.2.9.tgz (159,582 bytes)
...done: 159,582 bytes
downloading PHP_Timer-1.0.4.tgz ...
Starting to download PHP_Timer-1.0.4.tgz (3,694 bytes)
...done: 3,694 bytes
downloading PHPUnit_MockObject-1.2.3.tgz ...
Starting to download PHPUnit_MockObject-1.2.3.tgz (20,390 bytes)
...done: 20,390 bytes
downloading Yaml-2.2.1.tgz ...
Starting to download Yaml-2.2.1.tgz (39,994 bytes)
...done: 39,994 bytes
downloading PHP_TokenStream-1.1.5.tgz ...
Starting to download PHP_TokenStream-1.1.5.tgz (9,859 bytes)
...done: 9,859 bytes
install ok: channel://pear.phpunit.de/File_Iterator-1.3.3
install ok: channel://pear.phpunit.de/Text_Template-1.1.4
install ok: channel://pear.phpunit.de/PHP_Timer-1.0.4
install ok: channel://pear.symfony.com/Yaml-2.2.1
install ok: channel://pear.phpunit.de/PHP_TokenStream-1.1.5
install ok: channel://pear.phpunit.de/PHP_CodeCoverage-1.2.9
install ok: channel://pear.phpunit.de/PHPUnit_MockObject-1.2.3
install ok: channel://pear.phpunit.de/PHPUnit-3.7.19
C:\xampp\php>

3.11 Abschluss

Wurde die Umgebung korrekt eingerichtet sollten alle Tests von MakeGood durchlaufen.



HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ADGENERO

einfach online verwalten

Bachelorarbeit

Glossar

Abteilung Informatik
Hochschule für Technik Rapperswil

14. Juni 2013

Autoren: Fabian Schweizer, Manuel Schweizer
Betreuer: Dr. Daniel Keller
Projektpartner: foryouandyourcustomers AG, Pfäffikon ZH
Experte: Dr. Rudolf Mattmann
Gegenleser: Prof. Hansjörg Huser

1 Begriffe und Erklärungen

Begriff	Erklärung
Adgenero	Webapplikation zur einfachen Erstellung von Formularen
App	Ein Formular
cross-browser	Für verschiedene Browser geeignet
cross-platform	Für verschiedene Betriebssysteme geeignet
Entry	Ein Eintrag
Field	Ein Formularelement (z.B. Textfeld, Radiobuttons)
Friendly URL	Eine URL, die sich selbst erklärt und in der Regel kein „?“ oder „&“ enthält
Gast	Ein Benutzer, der nur über den Gast-Link verfügt
Gast-Link	Link zu einem Formular
GIT	Versionsverwaltungs-Tool
Hash	Kryptischer, zufällig generierter Wert einer bestimmten Länge
Manager	Ein Benutzer, der über den Manager-Link verfügt
Manager-Link	Link zur Verwaltung eines Formulars
ORM	Object-Relational Mapping
Record	Mehrere zusammengehörende Einträge
Redmine	Projektverwaltungs-Tool
UI	User Interface, bzw. Benutzeroberfläche