

Beuth Hochschule für Technik Berlin

Medieninformatik Online

Bachelorstudiengang am Fachbereich VI

Dokumentation Praxis-Projekt

Sprachspezifische Produktpflege (Rosetta-App)

Salim Oussayfi
3. Juli 2017

Gutachter / Betreuer
Prof. Dr. Stefan Edlich
Beuth Hochschule für Technik Berlin
FB VI / Labor Online Learning
Luxemburger Str. 10
13353 Berlin

EINLEITUNG

Die vorliegenden Seiten dienen dazu, mein Praxis-Projekt aus dem Sommersemester 2017 zu dokumentieren und dem Leser einen Einblick in alle Bereiche des bis heute abgeschlossenen Software-Entwicklungsprozesses zu geben.

Es galt, ein System zu entwickeln, das dazu dient, einen großen Pool an produktspezifischen Bezeichnungen in mehreren Sprachen zu katalogisieren und einen dynamischen Zugriff auf die Daten zu gewährleisten.

Es handelt sich demzufolge um einen digitalen Glossar, in dem große Mengen an Daten bezüglich Schreibweise und der jeweiligen Übersetzungen abgelegt sind.

Mithilfe des Systems wird der tägliche workflow des Auftraggebers optimiert und zeitgleich die Produktionskosten gesenkt.

Die Software sehe ich in seiner Funktion und Nutzen als Analogie zu dem „Stein von Rosette“. Prof. Dr. phil. Wolfhang Schenkel formulierte in „Die Entzifferung der Hieroglyphen und Karl Richard Lepsius“ **folgende Passage:**

„Im Juli 1799 wurde bei Schanzarbeiten der französischen Armee nahe der Mündung eines der Nilarme in das Mittelmeer, bei dem Dorf ar Rashid, der berühmte »Stein von Rosette« gefunden. Auf diesem »Stein« steht in der Form eines Dekrets der Beschluss einer Priestersynode des Jahres 196 v. Chr., den jugendlichen König Ptolemäus V. Epiphanes für seine dem Volk gewährten Vergünstigungen in verschiedenster Weise zu ehren. ... Niedergeschrieben worden ist der Text dann tatsächlich auf Ägyptisch in zwei Versionen, zuerst in traditionellem Ägyptisch mit Monumental-Hieroglyphen, danach in einem jüngeren, demotischen Ägyptisch mit demotischer Kursive, schließlich in griechischer Sprache mit griechischen Buchstaben.“ Originalveröffentlichung in: Verena M. Lepper und Ingelore Hafemann (Hg.), Karl Richard Lepsius. Der Begründer der deutschen Ägyptologie (Kaleidogramme 90), Berlin 2012, S. 37-78

Mithilfe des Steins von Rosetta bestand nun die Möglichkeit, die Hieroglyphen zu entschlüsseln, da sinngemäße Texte sowohl in Hieroglyphen als auch in bekannten griechischen Buchstaben auf eben diesen Stein geschrieben standen.

Aus diesem Grund habe ich die Anwendung „Rosetta-App“ genannt. Der Zusatz „App“ ist dadurch begründet, dass dieser „neuzeitige Stein“ ausschließlich in einer digitalen Form besteht und dynamisch genutzt und beliebig erweitert werden kann.

Diese Dokumentation, die die Entwicklung eben dieser Rosetta-App veranschaulicht, ist chronologisch aufgebaut, d.h. ich beginne mit den allgemeinen Voraussetzungen für das akademische Praxis-Projekt und gebe mich anschließend Schritt für Schritt durch die einzelnen Phasen des Entwicklungsprozesses – angefangen bei der Definition der Unternehmensziele bis hin zur Implementierung der fertigen Anwendung.

Im Anschluss daran werde ich die Dokumentation mit einem Fazit und möglicher Ausblicke zur Weiterentwicklung der Software beenden.

Zum Abschluss dieser Einleitung möchte ich noch auf die beiden ausschlaggebenden Parteien eingehen, die in diesem Projekt wesentlich involviert sind:

Das ist zum einen die TYPODROM WERBEAGENTUR GmbH, Radilostraße 43, 60489 Frankfurt am Main, im weiteren Verlauf dieser Dokumentation als TYPODROM bezeichnet. TYPODROM ist der Auftraggeber des Projekts und gleichzeitig mein Arbeitgeber.

Die zweite Partei wird gebildet durch die Adam Opel GmbH, Bahnhofsplatz, 65423 Rüsselsheim am Main – im weiteren Verlauf dieser Dokumentation als Opel bezeichnet.

Opel ist hier nicht der direkter Auftraggeber des Projekts sondern das System dient zur Optimierung des Workflows auf Seiten von TYPODROM bei der Bearbeitung von Aufträgen durch Opel.

GLIEDERUNG/INHALT

1. Aufgabenstellung	S. xx
1.1 Ziele und Anforderungen an das Praxis-Projekt	S. xx
1.2 Benefit Studenten	S. xx
2. Zielsetzung/Unternehmensziele	S. xx
2.1 Zielsetzung	S. xx
2.2 Benefit Unternehmen	S. xx
3. Analyse-Phase	S. xx
3.1 Beschreibung der Ausgangssituation/Ist-Analyse	S. xx
3.2 Vision und Systemidee/Soll-Konzept	S. xx
4. Vorstudie und Marktanalyse	S. xx
4.1 Umweltanalyse	S. xx
4.2 Akteure identifizieren	S. xx
4.3 Umfeld	S. xx
5. Konzept der Qualitätssicherung	S. xx
5.1 Requirements-Engineering-Tool	S. xx
5.2 Iterative Herangehensweise	S. xx
5.3 Retrospektive mit Stakeholdern	S. xx
6. Systemstruktur	S. xx
6.1 Infrastruktur	S. xx
6.2 Sprache	S. xx
6.3 Frameworks/Libraries	S. xx
6.4 Tools	S. xx
7. Design	S. xx
7.1 UML-Diagramme	S. xx
7.2 Prototyp	S. xx
7.3 Sidemap	S. xx
7.4 Datenstruktur	S. xx
8. Anwendung	S. xx
8.1 Screendesign/fertige Anwendung	S. xx
8.2 Features	S. xx
8.3 ClickDummy InVision	S. xx
9. Ergebnisse und Fazit	S. xx
9.1 Wissensmanagement-Tool	S. xx
9.2 Herausforderungen	S. xx
9.3 Gelerntes	S. xx
10. Ausblick	S. xx
10.1 was kann man machen	S. xx
10.2 Aussicht	S. xx
11. Literatur, Quellen, Bildnachweise	S. xx

1. AUFGABENSTELLUNG

– Ziele und Anforderungen an das Praxis-Projekt

In diesem ersten Abschnitt gehe ich auf die Ziele und Anforderungen des Praxis-Projekts ein.

Im Allgemeinen dient das Praxis-Projekt dazu, das im Studium erlernte sehr praxisnah anzuwenden. Die Technische Hochschule Köln definiert die Ziele des Praxis-Projekts wie folgt: „Im Praxisprojekt sollen die Studierenden Methoden und Techniken, die sie im Studium erlernt haben, in einem realitätsnahen Projekt weitgehend selbstständig anwenden.“ <http://www.f10.th-koeln.de/campus/institute/informatik/studium/praxisprojekte/>

Das Verfolgen dieser Ziele ist sicherlich von Studiengang zu Studiengang auf unterschiedliche Weise zu erreichen. Speziell im Studium der Medieninformatik liegt es nahe, ein Projekt zu wählen, in dem eine digitale Anwendung implementiert wird, semzufolge ein neues Software-System entsteht.

Nur mit dem Programmieren irgend einer Anwendung ist es allerdings nicht getan, es sollte ein weites Spektrum bei der Entwicklung angewendet werden, d. h. es sollten Bereiche aus der Analyse, Entwicklung, Implementierung und der Medien berücksichtigt werden, wie es Prof. Dr. Edlich der Beuth-Hochschule für Technik im „Leitfaden für das Praxisprojekt (Online)“ erläutert. <https://prof.beuth-hochschule.de/edlich/praxisprojekt-online/>

Neben der genannten Ziele und der Anforderungen wird auch ein zeitlicher Aufwand für die Bearbeitung des Projektes vorausgesetzt. Ebenfalls im zuvor zitierten „Leitfaden für das Praxisprojekt (Online)“ werden folgende Anforderungen zum Aufwand genannt: „Das Praxisprojekt im 5. Semester wird mit 15 Credits bewertet. Pro Credit werden als durchschnittlicher Arbeitsaufwand für die Studierenden 30 Zeitstunden berechnet. Sie müssen also 450 Stunden für die Bearbeitung des Projektes inkl. Bericht und Präsentation einplanen. Daraus ergibt sich renerisch natürlich auch eine Mindestanzahl von Tagen / Monaten für das Praktikum.“

<https://prof.beuth-hochschule.de/edlich/praxisprojekt-online/>

– Benefit Student

Bei der Beschreibung des Benefits für die/den Studierenden möchte ich meine Ausführung einzig auf mich und meine Eindrücke zum Praxis-Semester beziehen und ich bin mir sicher, dass sich meine Ansichten mit denen vieler Kommilitonen decken.

Ich habe das Praxis-Projekt so erlebt, dass ich viele Erfahrungen gewonnen habe, sei es in der Projekt-Arbeit im Allgemeinen oder in der Programmierung im Speziellen.

Ich habe das komplettes Projekt von Anfang an geplant, begleitet und geleitet.

In meinem beruflichen Alltag habe ich nicht viele Schnittmengen mit den Themengebieten aus dem Studium, gerade deswegen hat sich die Projektarbeit als eine wertvolle Erfahrung herausgestellt, wenn es darum geht, das im Studium gelernte in einem konkreten Projekt anwenden zu können.

Ich habe viele Techniken und Methoden aus meinem Studium anwenden können, was das im Einzelnen ist, möchte ich anhand dieser Dokumentation präsentieren.

Sicherlich gab es bis jetzt viele interessante und spannende Projekte im Studium, die sowohl im Team als auch als Einzelarbeit zu bearbeiten waren, aber im Umfang an für sich und Anspruch an der nebenläufigen Organisation des Projekts stellt sich die hier beschriebene Arbeit als eine bedeutende Erfahrung für mich heraus.

2. ZIELSETZUNG/UNTERNEHMENSZIELE

– Zielsetzung

In diesem Abschnitt beschreibe ich die Zielsetzung und die Motivation seitens des Auftraggebers für das in der vorliegenden Dokumentation beschriebene System.

Das primäre Ziel war es, vorhandenes Wissen dynamisch an nur einem zentralen Ort zu speichern und unternehmensweit zur Verfügung zu stellen.

Ähnlich wie bei einem webbasierten Wissensmanagement-System wie z.B. Confluence der Firma Atlassian sollte auch bei Rosetta-App das kollaborative Arbeiten ermöglicht und gefördert werden.

Jeder Mitarbeiter soll bei Bedarf die zentral abgelegten Daten für seine Zwecke nutzen und gleichzeitig den Datenbestand erweitern, bzw. aktualisieren können.

Es galt, durch genügend Transparenz immer ersichtlich und nachvollziehbar sein zu lassen, welcher Anwender einzelne Datensätze kreiert, manipuliert, kommentiert oder gelöscht hat.

Zu jedem Beitrag soll demzufolge sowohl Urheber des Eintrags angezeigt werden als auch die Nutzer, die den Daten geändert oder gelöscht haben.

Durch eine Kommentarfunktion soll es möglich sein, die Relevanz mancher Einträge von Anderen zu differenzieren, bzw. weiterführende Informationen zu jedem Eintrag hinterlegen zu können.

Zusätzlich soll jede Manipulation der Daten mit dem jeweiligen Datum und Uhrzeit gespeichert und im System angezeigt werden.

Die zentrale Datenspeicherung hat zur Folge, dass Redundanzen in der Datenbeschaffung und Speicherung reduziert bzw. vermieden werden.

– Benefit Unternehmen

Für TYPODROM entsteht mithilfe des Systems ein Mehrwert unter anderem dadurch, dass der Workflow wesentlich optimiert wird.

Arbeits-Unterbrechungen werden minimiert, da alle nötigen Informationen zur Verfügung stehen und prompt genutzt werden können.

Jeder Mitarbeiter kann in einem ersten Schritt prüfen, ob eine gewünschte Übersetzung bereits existiert, bevor er sie in Auftrag gibt.

Somit werden unnötige Kosten vermieden, da die zu übersetzenden Produktbeschreibungen nur einmal übersetzt werden müssen bzw. übersetzt worden sind.

Durch die Kommentar-Funktion können zudem sich ähnelnde Übersetzungen leicht differenziert, bzw. priorisiert werden.

3. ANALYSE-PHASE

– Beschreibung der Ausgangssituation/Ist-Analyse

TYPODROM betreut in der Funktion einer Tagesgeschäftsagentur seinen Kunden Opel.

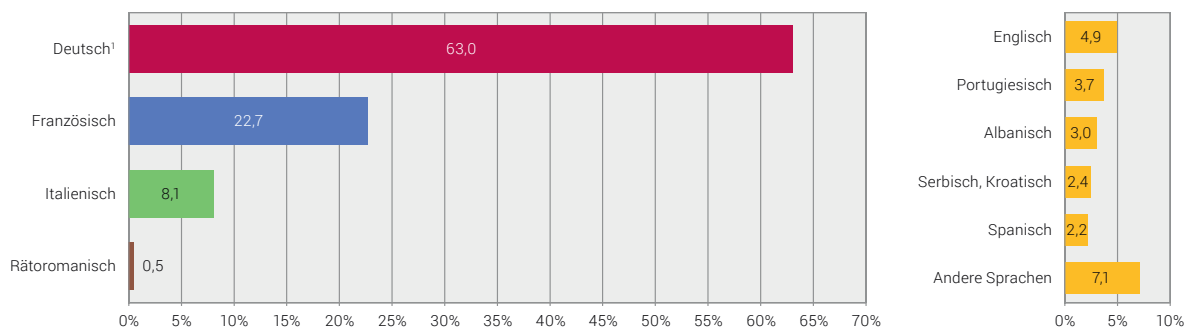
TYPODROM generiert für Opel Printmedien wie z. B. Kataloge, Anzeigen, POS-Materialien und weitere – auf eine Auflistung des gesamten Spektrums der verkaufsfördernden Publikationen, die TYPODROM im Auftrag von Opel entwickelt, sehe ich an dieser Stelle ab, da diese Information nicht zur Beschreibung meines Projekts erforderlich ist – es sei hier nur erwähnt, dass die Werbematerialien für den gesamten europäischen Raum kreiert und produziert werden.

Für das Praxis-Projekt habe ich mich zunächst auf den Schweizer Markt konzentriert – im Kapitel Ausblicke gehe ich auf eine mögliche Skalierbarkeit ein.

In der Schweiz sind neben Deutsch weitere Landessprachen vertreten. Dies sind die Sprachen Französisch, Italienisch und Rätoromanisch.

Die folgende Grafik (Veröffentlicht am 27.03.2017) des Bundesamt für Statistik stellt aus der Strukturerhebung des Jahres 2015 die Gewichtung der gesprochenen Sprachen für den Zeitraum 2015 in der Schweiz dar.

Als Hauptsprachen genannte Sprachen, 2015



¹ oder Schweizerdeutsch

Ständige Wohnbevölkerung, die in Privathaushalten lebt. Die Befragten konnten mehrere Sprachen angeben.

Quelle: BFS – Strukturerhebung (SE)

© BFS 2017

<https://www.bfs.admin.ch/bfs/de/home/statistiken/bevoelkerung/sprachen-religionen/sprachen.html>

Opel führt in der Schweiz alle Publikationen in den drei Sprachen Deutsch, Französisch und Italienisch.

Bei der initialen Entwicklung des Systems habe ich zunächst die Opel Preislisten berücksichtigt. Auch hier ist man zukünftig nicht auf das Medium „Preisliste“ beschränkt und kann analog zur Wahl der Sprachen auch im Bereich des Mediums und unabhängig eines solchen das System beliebig skalieren.

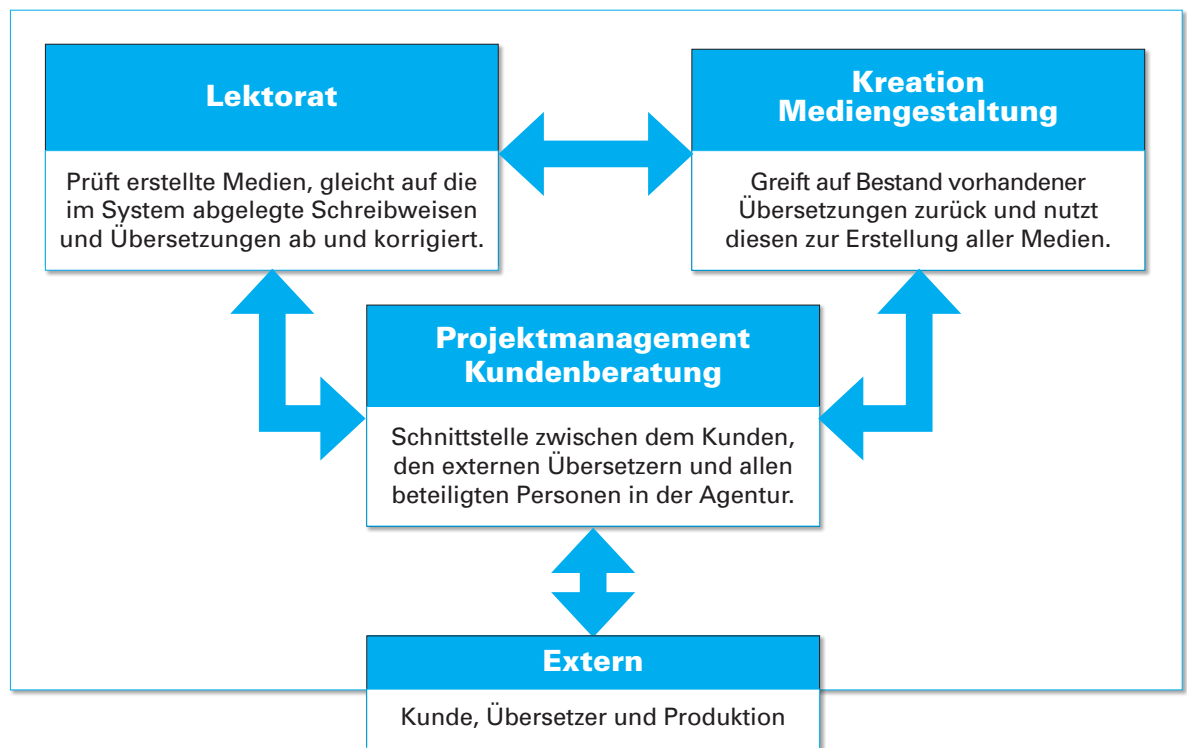
Da ich, wie erwähnt, das Projekt unter der Premisse der Preislisten gestartet habe, werde ich diese Dokumentation auch an diesen Preislisten ausgerichtet weiterführen.

Für jeden Fahrzeugtyp gibt es eine 16 – 28 seitige Preisliste. Jedes dieser Dokumente gibt es in zwei Sprach-Versionen, d.h. es gibt die Preislisten in Deutsch/Französisch und analog dazu in Deutsch/Italienisch. Je nach Modell-Jahr führt Opel 18 – 22 Fahrzeugtypen.

In den Preislisten werden Preise und Daten zu den verfügbaren Motoren, der Serien- und Sonderausstattung und zu jeglichem erwerbbaaren Zubehör kommuniziert. Kurz gefasst lässt sich sagen, die Preislisten befassen sich mit allen Fahrzeugteilen, die nicht dem Auto als Ganzes entsprechen.

Bei TYPODROM werden die Preislisten abteilungsübergreifend im Tagesgeschäft erstellt. Diese Abteilungen sind im Einzelnen das Projektmanagement bzw. die Kundenberatung, die Kreation zusammen mit der Mediengestaltung und das Lektorat.

Die folgende Grafik gibt einen Überblick darüber, inwiefern die Abteilungen im Workflow involviert und miteinander verknüpft sind und beschreibt die jeweiligen Station.



Durch die Zusammenarbeit dieser Entitäten entstehen periodisch neue Preislisten und mit jeder neuen Preisliste werden auch neue Produkte vermarktet bzw. wird eine Fahrzeugausstattung mit dem Zubehör aus einer anderen Flotte erweitert. Diese Produkte sind sehr fachspezifisch bzw. beruhen teilweise auf sprachgebundenen Eigennamen und sprachspezifischen Wortschöpfungen.

Bei der Erstellung neuer Publikationen in den drei Sprachen wird regelmäßig auf externe Ressourcen zugegriffen, um die Produktbeschreibungen in die jeweils benötigte Sprache zu übersetzen. Dadurch entstehen unnötige Kosten und Redundanzen, zudem ergibt sich hieraus ein erheblicher zeitlicher Mehraufwand.

- Vision und Systemidee/Soll-Konzept

Es sollte ein Software-System entwickelt werden, das alle Produktbezeichnungen von Opel beinhaltet und jeweils die korrekten Übersetzungen und Schreibweisen enthält.

Einzelne Übersetzungen können durch eine Kommentar-Funktion leicht verifiziert und somit von ähnlichen, obsoleten Schreibweisen differenziert werden.

Zudem kann eine Referenz auf die verwendete Publikation gegeben werden mit all ihren Attributen wie Erscheinungszeitpunkt und einer Verlinkung zu dem eigentlichen Dokument. Bei dem hier behandelten Praxis-Projekt wird zu jedem Eintrag automatisch ein Link generiert, über den sich die jeweilige Preisliste als PDF aufrufen lässt.

Alle Einträge sind im View editierbar und können auf einfache Art und Weise in die Zwischenablage kopiert bzw. kommentiert werden.

Durch die Möglichkeit, den Eintrag in die Zwischenablage kopieren zu können, wird das System auf komfortable Art und Weise in den Workflow integriert.

Das System kann von jedem registrierten Mitarbeiter für seine Arbeit genutzt und je nach Berechtigung auch gepflegt werden.

Das System ist skalierbar, d. h., es ist möglich, beliebig viele Sprachen zu integrieren, desweiteren ist das System autark, was das zu behandelnde Medium betrifft.

4. VORSTUDIE UND MARKTANALYSE

– Umweltanalyse

Sicherlich besteht heutzutage die Möglichkeit, sich mithilfe von online zur Verfügung stehenden Übersetzungstools ganze Textabschnitte in jede beliebige Sprache übersetzen zu lassen und damit seine Anliegen dem Empfänger verständlich zu machen.

Als Beispiel sei hier der Google-Übersetzer (<https://translate.google.com>), Linguee (<http://www.linguee.de/>) oder Leo (<http://www.leo.org/>) zu nennen.

Das gilt allerdings nur in einem Bereich, in dem die Anforderungen ausschließlich der Weitergabe von Informationen dienen und eine zuverlässige Interpretation vorausgesetzt werden darf. Sobald auf professionelle Weise Produkte vertrieben werden, ist diese Herangehensweise aus unternehmerischer Sicht nicht mehr empfehlenswert.

Es sollten ausgebildete Übersetzer, zudem meist „Muttersprachler“ die Texte in die jeweilige Sprache transformieren.

Im Fall des hier behandelten Projektes kommt erschwerend hinzu, dass es sich zum Teil um fachspezifische **Bezeichnungen/Wortschöpfungen (Beispiel/Achim fragen)** handelt, die seitens des Kunden entwickelt und verwendet werden.

Dies sind beispielsweise:





- Beispiel
- Beispiel
- Beispiel

Es bietet sich also an, diese Produktbezeichnungen, inklusive aller atomaren Spracheinheiten und ganzer zusammengehöriger Textabschnitte zu speichern und zentral zur Verfügung zu stellen.

– Akteure identifizieren

Die Nutzer des Systems sind unternehmensweit vertreten, d.h. alle Mitarbeiter, die für den Kunden Opel arbeiten. Im Einzelnen sind dies Mitarbeiter aus der Kreation/Mediengestaltung, des Projekt-Managements/der Kundenberatung und aus dem Lektorat.

Da die Nutzer des Systems in heterogenen Bereichen tätig sind und unterschiedliche Affinitäten aufweisen, ist dementsprechend für die Bedienung des Systems keinerlei oder nur wenig Schulungsbedarf erforderlich, was durch eine intuitiv bedienbare Nutzeroberfläche gewährleistet ist.

Front- und Seitenairbag, Beifahrer, manuell deaktivierbar  Post, sum enimis quissum quae rem iustisit voluipi acerchil et optur. Max Mustermann 08.05.2017	airbag frontal et airbag latéral du passager avant désactivables manuellement  Imolendi dia consequisci ipsam, essit late volienesti di dolorem poreper fererum laboressunte. Max Mustermann 08.05.2017	airbag frontale e laterale passeggero disattivabile manualmente  Kaboritam rem qui quaecum hantasilati officipsa. Max Mustermann 08.05.2017	Serienausstattung Sicherheit  Enthalten in: ADAM (df) (di), AstraST (df) (di), MokkaX (df) (di), InsigniaST (df) (di), InsigniaGS (df) (di), Zafira (df) (di)
Erstellt von Martina Musterfrau am 03.05.2017 16:31 Uhr		Geändert von Max Mustermann am 05.05.2017 15:13 Uhr	

Darstellung eines Datensatzes

– Umfeld

Das System ist primär als browserseitige Anwendung konzipiert, die Nutzung auf einem mobilen Gerät ist auch möglich, dient allerdings nur dem Zweck, gegebenenfalls Vorort beim Kunden auf den Datenbestand zugreifen zu können, was in der Regel auch am Notebook funktioniert, aber hier zur Abdeckung aller Eventualitäten auf dem Mobiltelefon gewährleistet ist.

5. KONZEPT DER QUALITÄTSSICHERUNG

Zur Unterstützung in der Planung habe ich drei Kollegen gewählt, welche für den Kunden Opel Schweiz arbeiten und somit die spätere Nutzergruppe repräsentieren.

Folgend nenne ich kurz deren jeweilige disjunkten Funktionen im Unternehmen und erläutere, in welcher Relation sie mit dem System stehen werden:

Thomas S./Mediengestaltung: kann auf den Bestand vorhandener Übersetzungen zugreifen und diese ad hoc zur Erstellung aller Medien nutzen

Claudia F./Lektorat: prüft die erstellten Medien, gleicht auf die im System abgelegten Schreibweisen und Übersetzungen ab und korrigiert diese ggf. mithilfe des vorhandenen Bestands

Nicole R./Kundenberatung: Schnittstelle zwischen dem Kunden, den externen Übersetzern, der Produktion und allen beteiligten Personen in der Agentur

In regelmäßigen Besprechungen habe ich diesen drei Kollegen den aktuellen Stand des Projekts präsentiert, konnte gesammelte Fragen stellen und habe gleichzeitig Feedback und Input/Ideen zur weiteren Herangehensweise erhalten.

Auf diese Art und Weise habe ich gleichzeitig eine Regel des Vorgehensmodells XP befolgt, indem ich die späteren Nutzer, hier als Auftraggeber zu sehen, eng in den Entwicklungsprozess eingebunden habe.

Auf <http://www.extremeprogramming.org/> wird unter dem Punkt „rules/coding“ folgendes genannt:

„The customer is always Available

One of the few requirements of extreme programming (XP) is to have the customer available. Not only to help the development team, but to be a part of it as well.“

<http://www.extremeprogramming.org/rules/customer.html>

Im weiteren Verlauf der Dokumentation werde ich die drei zuvor genannten Kollegen als die **Stakeholder** des Systems bezeichnen.

Stakeholder sind Personen bzw. Gruppen von Personen mit einem besonderen Interesse am Ergebnis eines Prozesses, wie es auf „agiles-projektmanagement.org“ definiert ist. <http://agiles-projektmanagement.org/scrum-stakeholder/>

Wie zuvor erwähnt, wies meine im vorherigen Abschnitt beschriebene Vorgehensweise Parallelen zu der eines **Vorgehensmodells auf – respektive zum Vorgehensmodell Extreme Programming**. <https://blogs.itemis.com/de/scrum-kompakt-extreme-programming-xp>

Im überwiegenden Verlauf der Projektentwicklung habe ich mich allerdings an dem **agilen Rahmenwerk Scrum** orientiert. <https://www.it-agile.de/wissen/einstieg-und-ueberblick/scrum/>

In diesem Vorgehensmodell geht man davon aus, dass Softwareprojekte nicht im Voraus detailliert planbar sind. Aus diesem Grund erfolgt die Planung nach dem Prinzip der schrittweisen Verfeinerung, wobei die Entwicklung des Systems durch das Team nahezu gleichberechtigt erfolgt. <http://wirtschaftslexikon.gabler.de/Definition/scrum.html>

Auf drei Schnittmengen dieses Vorgehensmodells und der hier dokumentierten Software-Implementierung gehe ich im Folgenden ein.

Es handelt sich hierbei um die Definition der **Backlogs**, die **iterative Vorgehensweise** innerhalb zeitlich definierter **Sprints** und die regelmäßigen **Retrospektiven** mit den Stakeholdern.

– Requirements–Engineering–Tool

Ein Requirements-Engineering dient zur Ermittlung, Beschreibung, Analyse und Gewichtung der Anforderungen. Die Festlegung der Requirements erfolgt in einer möglichst exakten und operationalen Form, um eine qualitative Verbesserung der Anforderungsdefinition und eine Reduktion der Fehler zu erreichen, wie es im Gabler Wirtschaftslexikon definiert ist. <http://m.wirtschaftslexikon.gabler.de/Definition/requirements-engineering.html>

Zur Festlegung eben dieser Vorgaben habe ich ein eigenes Tool geschrieben, mithilfe dessen die Stakeholder und ich einzelne Vorgaben definieren konnten.

Bleibt man bei dem Vergleich zum Vorgehensmodell SCRUM, sind die jeweiligen Requirements gleichzusetzen mit einem **Product Backlog**, in welchem die zu entwickelnden Features des Systems priorisiert werden und in gemeinsamer Absprache und unter Berücksichtigung des Nutzens und der Notwendigkeit in einem folgenden Sprint abgearbeitet und implementiert wurden.

Das Tool erzwingt eine möglichst einfache Definition der Vorgaben. Es sollen neben der Priorisierung durch den Teilnehmer lediglich der Titel und eine Kurzbeschreibung der jeweiligen Vorgabe eingetragen werden.

Der Nutzerkreis des Requirements–Engineering–Tools war auf vier Personen beschränkt, dies waren die drei Stakeholder und ich.

Formular ausblenden

Author

bitte wählen

Titel

drag&drop

Beschreibung

Verbindlichkeit

☐ muss (Pflicht)
 ☐ soll (Wunsch)
 ☐ kann (Absicht)
 ☐ wird (Vorschlag)

Submit

Requirements Rosetta-App

Verbindlichkeit	ID	Datum/Uhrzeit	Author	Titel	Beschreibung
kann	64	21.02.17 19:13:40	Salim	Zwischentablage	Einträge sollen per Klick auf Schaltfläche in die Zwischentablage gespeichert werden können
muss	65	21.02.17 19:59:36	Salim	Suche	Es soll nach Einträgen gesucht werden können
muss	69	26.02.17 11:43:27	Salim	Benutzerverwaltung	Alle Nutzer des Systems haben eigenen Account
muss	77	01.05.17 07:47:22	Thomas	Transparenz	es muss ersichtlich sein, wer Änderungen zu welchem Zeitpunkt vorgenommen hat
muss	81	22.03.17 09:51:40	Nicole	Datenspeicherung	Daten müssen automatisch eingepflegt werden, nicht händisch
muss	89	20.05.17 08:05:29	Nicole	Passwort ändern	Der Nutzer muss sich sein Passwort ändern können
muss	91	20.05.17 07:48:16	Thomas	Transparenz	es muss ersichtlich sein, wer Änderungen zu welchem Zeitpunkt vorgenommen hat
muss	92	20.05.17 07:49:11	Thomas	Transparenz	es muss ersichtlich sein, wer Änderungen zu welchem Zeitpunkt vorgenommen hat
soll	67	21.02.17 19:40:16	Salim	Carline/Produktcode	zu jedem Eintrag soll die Carline und der Produktcode ergänzt werden
soll	68	26.02.17 11:46:17	Salim	Autovervollständigung	Bei Eingabe in das Suchformular sollen (ab 3 Zeichen) Suchvorschläge angezeigt werden
soll	71	26.02.17 15:09:36	Salim	Einträge formatieren	evtl. zwei verschiedene Felder pro Sprache -> 1. Headline (bold), 2. Content (auch mit Bullets)
soll	73	21.03.17 09:55:07	Claudia	Häufigkeit des Suchbegriffes	es soll angezeigt werden, wie oft ein Begriff in einem Dokument vorkommt
soll	74	20.05.17 07:43:48	Claudia	Hervorhebung durch Übersetzer	Bei zwei oder mehreren Übersetzungsvarianten Hervorhebungsmöglichkeit der bevorzugten Übersetzung durch den Übersetzer (Kommentarfunktion)
soll	75	01.06.17 21:17:37	Thomas	Nach Carline suchen?	Man soll die Begrifflichkeiten einer Carline zuweisen können
soll	76	27.02.17 14:40:12	Thomas	hervorgehobenes Suchwort	gesuchtes Wort soll in Ergebnisanzeige hervorgehoben angezeigt werden
soll	80	20.05.17 07:45:03	Claudia	Kommentarfunktion	Einträge sollen kommentiert werden können
soll	82	20.05.17 07:45:32	Salim	Daten nicht endgültig löschen	gelöschte Daten sollen nur von der Anzeige ausgeschlossen sein, nicht endgültig gelöscht werden
soll	84	04.04.17 10:55:14	Salim	Interface zur Datenspeicherung	- Nutzer soll XML-Daten über eine GUI in die Datenbank einpflegen können - Nutzer soll aktualisierte PDF in das System laden können
soll	87	19.04.17 15:48:48	Salim	Dashboard	ein Dashboard mit den wichtigsten Funktionen soll auf der Startseite verfügbar sein
soll	88	19.04.17 15:50:06	Thomas	Filterfunktion	Es soll nach jedem gefundenen Begriff gefiltert werden können
soll	96	21.05.17 09:58:37	Claudia	Häufigkeit des Suchbegriffes	es soll angezeigt werden, wie oft ein Begriff in einem Dokument vorkommt
soll	97	01.06.17 14:16:09	Thomas	drag&drop	beim Date Upload sollen mehrere Dateien per drag&drop hochgeladen werden können
soll	98	01.06.17 21:17:09	Thomas	drag&drop	beim Date Upload sollen mehrere Dateien per drag&drop hochgeladen werden können
soll	99	01.06.17 21:17:41	Thomas	drag&drop	beim Date Upload sollen mehrere Dateien per drag&drop hochgeladen werden können
wird	66	19.02.17 15:50:45	Salim	Sprachen	deutsch, italienisch, französisch
wird	78	20.05.17 07:59:45	Salim	Nutzerverwaltung Admin	Der Admin hat vollen Zugriff auf alle Nutzerdaten
wird	94	20.05.17 07:59:52	Salim	Nutzerverwaltung Admin	Der Admin hat vollen Zugriff auf alle Nutzerdaten

<http://requirements.rosetta-app.de/index.php>

Dieses Werkzeug habe ich in PHP geschrieben, die Einträge liegen in einer MariaDB-Datenbank ab. Einzelne Funktionen wie das farbige Hinterlegen der Verbindlichkeiten oder das animierte Ein- und Ausblenden des Formulars sind mithilfe von JavaScript implementiert.

10

Da es sich bei diesem Tool um einen erforderlichen Nebenschauplatz handelt, möchte ich hier nicht weiter auf die Programmierung eingehen sondern verweise auf den für die Projekt-Entwicklung entscheidenden Inhalt.

Mithilfe der folgenden Tabelle liste ich die einzelnen Vorgaben tabellarisch auf:

Requirements Rosetta-App		
Titel	Beschreibung	erledigt
Sprachen	deutsch, italienisch, französisch	
Zwischenablage	Einträge sollen per Klick auf Schaltfläche in die Zwischenablage gespeichert werden können	
Suche	Es soll nach Einträgen gesucht werden können	
Benutzerverwaltung	Alle Nutzer des Systems haben eigenen Account	
Datenpflege	Daten müssen automatisiert eingepflegt werden, nicht händisch	
Transparenz	es muss ersichtlich sein, wer Änderungen zu welchem Zeitpunkt vorgenommen hat	
Nutzerverwaltung Admin	Der Admin hat vollen Zugriff auf alle Nutzerdaten	
Passwort ändern	Der Nutzer muss sich sein Passwort ändern können	
Carline/Produktcode	zu jedem Eintrag soll die Carline und der Produktcode ergänzt werden	
Autovervollstaendigung	Bei Eingabe in das Suchformular sollen (ab 3 Zeichen) Suchvorschläge angezeigt werden	
Eintraege formatieren	evtl. zwei verschiedene Felder pro Sprache => 1. Headline (bold), 2. Content (auch mit Bullets)	
Häufigkeit des Suchbegriffes	es soll angezeigt werden, wie oft ein Begriff in einem Dokument vorkommt	
Hervorhebung durch Übersetzer	Bei zwei oder mehreren Übersetzungsvarianten Hervorhebungsmöglichkeit der bevorzugten Übersetzung durch den Übersetzer (Kommentarfunktion)	
Nach Carline suchen?	Man soll die Begrifflichkeiten einer Carline zuweisen können.	
hervorgehobenes Suchwort	gesuchtes Wort soll in Ergebnisanzeige hervorgehoben angezeigt werden	
Kommentarfunktion	Einträge sollen kommentiert werden können	
Daten nicht endgültig löschen	gelöschte Daten sollen nur von der Anzeige ausgeschlossen sein, nicht endgültig gelöscht werden	
Interface zur Dateneinpfege	- Nutzer soll XML-Dateien über eine GUI in die Datenbank einpflegen können - Nutzer soll aktualisierte PDF in das System laden können	
Dashboard	ein Dashboard mit den wichtigsten Funktionen soll auf der Startseite verfügbar sein	
drag&drop	beim Datei-Upload sollen mehrere Dateien per drag&drop hochgeladen werden können	
Filterfunktion	Es soll nach jedem gefundenen Begriff gefiltert werden können	

Tabelle Stand 01.06.2017

- **iterative Herangehensweise**

Es wurde im Team festgelegt, welche dieser Anforderungen im einzelnen in einem nächsten Schritt bearbeitet werden sollten.

Diese iterative herangehensweise lässt sich gut vergleichen mit einem Sprint aus dem Vorhenesmodell Scrum.

In einem Sprint wird innerhalb einer festgelegten Dauer von typischerweise zwei bis vier Wochen ein funktionsfähiges Inkrement des Software-Produkts implementiert.

- **Retrospektive Stakeholdern**

In regelmäßigen Meetings mit den Stakeholdern wurde besprochen, was passiert ist. Die implementierten Features wurden getestet, beurteilt und ggf. eine Nachgebesserung in die Wege geleitet.

Diese Methode möchte ich im Ansatz mit der einer Retrospektive vergleichen,

...

6. SYSTEMSTRUKTUR

– Infrastruktur

Das System habe ich mithilfe der integrierten Entwicklungsumgebung PHP Storm des Unternehmens JET BRAINS entwickelt.

Ich habe mich für diese IDE entschieden, weil PHP Storm eine gute Autovervollständigung, ein intelligentes Refactoring und vor allem eine komfortable Anbindung an mein für diese Arbeit erstelltes Repository auf github.com bietet.

<https://www.jetbrains.com/phpstorm/>

Als Serverumgebung habe ich die freie Software **XAMPP des Entwicklers Apache Friends** mit dem integrierten phpMyAdmin-Interface genutzt.

Dadurch konnte ich das System auf einem lokalen Server entwickeln und testen.

<https://www.apachefriends.org/de/index.html>

Als relationales Datenbank-System habe ich Open Source-Datenbank MariaDB genutzt.

<https://mariadb.org/>

„MariaDB ist eine Abspaltung (Fork) von MySQL. Diese wurde entwickelt, nachdem Oracle Sun Microsystems im Jahre 2010 übernommen hatte.“

<https://wiki.ubuntuusers.de/MariaDB/>

Die Kommunikation mit der Datenbank habe ich mittels **PHP Data Objects realisiert. Wie von PHP.net beschrieben**, stellt diese PHP Data Object-Erweiterung (PDO) eine leichte, konsistente Schnittstelle bereit, um mit PHP auf Datenbanken zuzugreifen. Dadurch, dass der Datenzugriff über eine Abstraktionsschicht läuft, ist es möglich, mittels PDO auf jedes beliebige Datenbanksystem mit den selben Funktionen zuzugreifen zu können.

<http://php.net/manual/de/intro.pdo.php>

Daraus ergibt sich, dass man nicht an die jetzige Datenbank gebunden ist und der aktuelle Code bereits für alternative Datenbanken gültig konfiguriert wurde.

Doch der entscheidende Grund für die Arbeit mit PDO war für mich die Aktualität dieser Technik im Vergleich zu MySQL und MySQLi.

Zudem bietet PDO eine einfache Nutzung von Prepared Statements.

Online läuft das System auf einem Server der STRATO AG.

Analog zur lokalen Umgebung ist auch dieser Server mit phpMyAdmin und MariaDB ausgestattet wodurch ein Umzug von lokal nach online relativ unkompliziert ist, es muss lediglich die Datenbankanbindung konfiguriert werden,

– Sprache

– PHP

Da das System eine typische Web 2.0-Anwendung ist, der Nutzer über den Browser auf den dynamisch erzeugten Content zugreift und bidirektional den Datenbestand nutzt, habe ich für die Erstellung des Systems PHP verwenden.

Ich habe bei der Programmierung teilweise auf objektorientierte Features von PHP zurückgegriffen, um dadurch einen schlankeren Code erzeugen zu können.

Als Beispiel möchte ich die Verwendung von Klassen nennen, mithilfe derer ich z.B.: das Gerüst von Formular-Feldern einmal in einer Klasse definiert habe und im Produktions-Code über nur ein kurzen Aufruf mit Parameterübergabe darstellen konnte. Dies kommt der Modellierung des Views in einem MVC-Pattern sehr nahe.

Das Model in diesem Pattern stelle ich in Form der verschiedenen Datenbank-Anbindungen dar, über die der Datenbestand manipuliert, bzw. ausgegeben wird.

Der eigentliche Controller, der die Kommunikation zwischen View und Model regelt, ist zum aktuellen Stand der Dokumentation noch nicht implementiert, vielmehr sind zur Zeit Funktionen des Controllers teilweise sowohl im View als auch im Model integriert.

Um ein echtes MVC-Pattern zu implementieren, müsste man diese Abhängigkeiten dementsprechend anpassen.

– JavaScript

Neben der Programmierung mit PHP habe ich teilweise auf JavaScript zurückgegriffen und auch das Framework jQuery genutzt.

Mithilfe dieser Skriptsprache habe ich Funktionen implementiert wie z. B. das automatische Skalieren der Textfelder, sobald ein Übertrag an Inhalt entsteht. Dies ermöglicht es, eine feste Größe des Textfeldes darzustellen, welche dennoch flexibel ist. Die Folge ist eine ausgeglichene Darstellung für den User und gleichzeitig bleibt keinerlei Input im Verborgenen.

Eine weiteres nützliches Feature ist die Funktion „copyToClipboard()“, welche den generischen Parameter „e“ entgegen nimmt, dem Body-Element im DOM ein temporäres Input-Field hinzufügt, dieses Feld mit dem zu kopierenden Wert belegt, sodass man den gewünschten Text mithilfe des Aufrufs „document.execCommand(„copy“)“ in die Zwischenablage kopieren kann.

Hierzu ist für jede Textstelle, die kopiert werden kann, eine eindeutige ID nötig, dies habe ich wiederum mittels von PHP realisiert.

Diese Feature erweist sich als besonders praktisch, sobald man bestimmte Daten aus dem System für die aktuelle Arbeit benötigt.

Zum hinzufügen der Carlines habe ich ein Multiselect-Formular genutzt, welches sich in einem eleganten Dropdown-Menü verbirgt. Hierzu habe ich auf das Script „Bootstrap-Multiselect“ zurückgegriffen.

<https://github.com/davidstutz/bootstrap-multiselect/blob/master/dist/js/bootstrap-multiselect.js>

Die Autocomplete Funktion habe ich mithilfe der unter der MIT-Lizenz stehenden freien jQuery JavaScript Library verwirklicht.

<http://code.jquery.com/jquery-1.10.2.js>

Copyright 2005, 2013 jQuery Foundation, Inc. and other contributors

Released under the MIT license

<http://jquery.org/license>

– Frameworks/Libraries

– Bootstrap

Ich habe das freie CSS-Framework Bootstrap referenziert um so die in diesem Framework vordefinierte Klassen-Hierarchie nutzen zu können. Überwiegend habe ich den Aufbau der Formulare, Buttons und der Navigation genutzt und diese teilweise individualisiert.

Im Besonderen habe ich mich für dieses Framework entschieden, weil es ein 12-Grid-System unterstützt, welches ich beim Screendesign angewandt habe.

Ein weiterer ausschlaggebender Grund für die Nutzung von Bootstrap ist die bereits implementierte responsive Darstellung, d. h. ich musste nicht eigene MediaQueries für die unterschiedlichsten Endgeräte definieren.

<http://getbootstrap.com/>

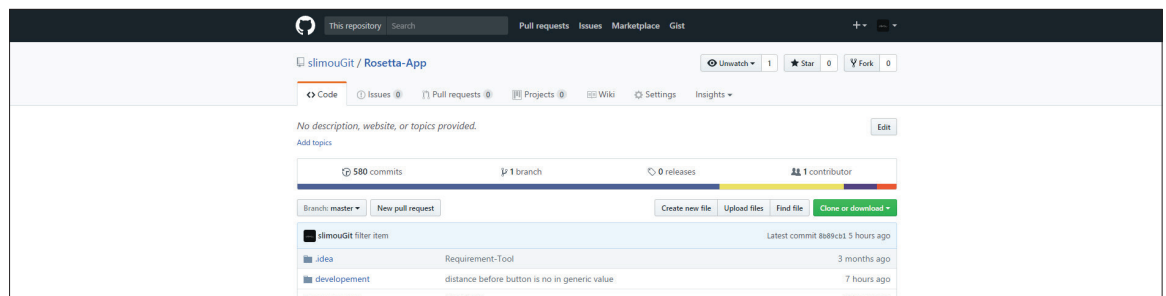
– Tools

Neben der eigentlichen Entwicklungsumgebung PHP Storm und der Serverumgebung XAMPP von Apache Friends habe ich die folgenden Tools genutzt, die mich bei der Projekt-Arbeit unterstützen.

– Github <https://github.com/>

Zur Versionsverwaltung und damit ich nicht auf die Verwendung eines einzigen Arbeitsplatzes während der Projektarbeit beschränkt bin, habe ich Git verwendet, d. h. Github in Verbindung mit PHP-Storm.

Das dortige Repository dient zudem zum einfachen Austausch mit dem Betreuer der Hochschule und als Beleg des erbrachten Aufwands, der sich für die Projekt-Arbeit ergeben hat. <https://github.com/slimouGit/Rosetta-App.git>

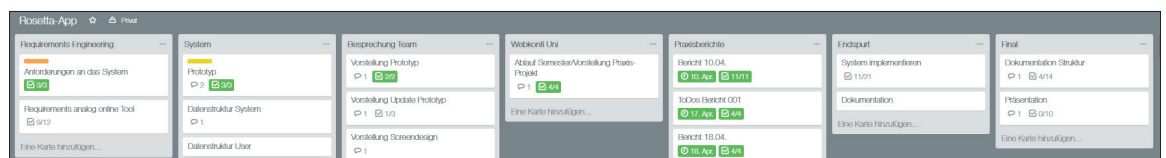


– Trello <https://trello.com/>

Ich habe mit Trello gearbeitet, um den gesamten Workflow zu organisieren.

Ich habe mir Listen zu verschiedenen Teilbereichen der Projekt-Arbeit erstellt und diese jeweils mit einzelnen Karten und Checklisten befüllt.

Dadurch hatte ich stets den vollen Umfang im Blick und konnte meine Arbeit optimal organisieren.



Ausschnitt Screenshot Stand 03.06.2017

– Requirements Engineering

Wie bereits unter dem Punkt „Konzept der Qualitätssicherung“ erwähnt habe ich ein einfaches, unabhängig laufendes Requirements-Engineering-Tool aufgesetzt, mithilfe dessen die Anforderungen an das System definiert wurden.

<http://requirements.rosetta-app.de/>

- UML

Zur Modellierung der UML-Diagramme habe ich das für akademische (nicht kommerzielle) Zwecke freie Tool Astah Community. <http://astah.net/editions/community>

7. DESIGN

- UML-Diagramme

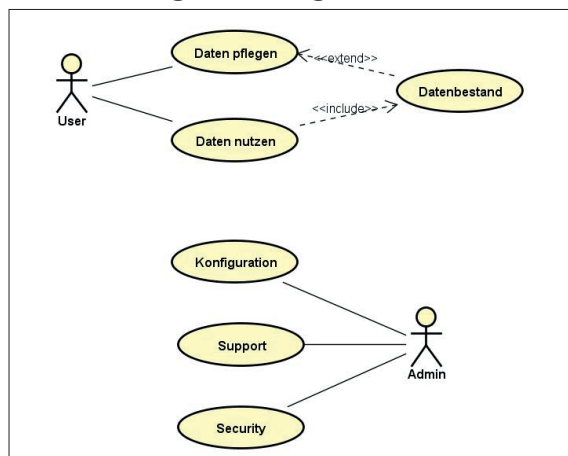
Zur Modellierung und Dokumentation bestimmter Abläufe in der Software habe ich Unified Modeling Language (UML) verwendet.

Auf der Internetpräsenz der Hochschule Darmstadt wird der Begriff UML wie folgt definiert: „Die UML ist eine Modellierungssprache, also eine Sprache zur Beschreibung von Software-Systemen. Sie bietet eine einheitliche Notation, die für viele Anwendungsgebiete nutzbar ist. Sie enthält Diagramme und Prosa-Beschreibungsformen. Mit Hilfe der UML können statische, dynamische und Implementierungsaspekte von Softwaresystemen beschrieben werden. „Die UML ist damit zur Zeit die umfassendste Sprache und Notation zur Spezifikation, Konstruktion Visualisierung und Dokumentation von Modellen für die Softwareentwicklung.“ <https://www.fbi.h-da.de/labore/case/uml.html>

Originalquelle: Gabriele Bannert – Objektorientierter Softwareentwurf mit UML

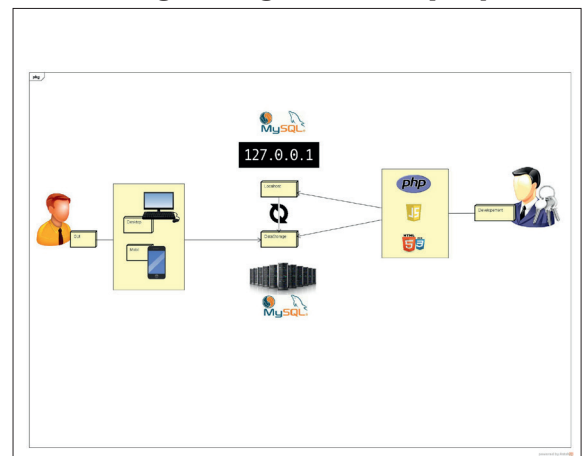
Ich habe vier Diagramme modelliert, dies sind im Einzelnen die beiden Verhaltensdiagramme „Use-Case“ und „Activity“ und die beiden Diagramme Deployment und Component aus der Gruppe der Architekturdiagramme.

Anwendungsfall-Diagramm (Use-Case)



<https://github.com/slimouGit/Rosetta-App/blob/master/development/documentary/UML/UseCase.jpg>

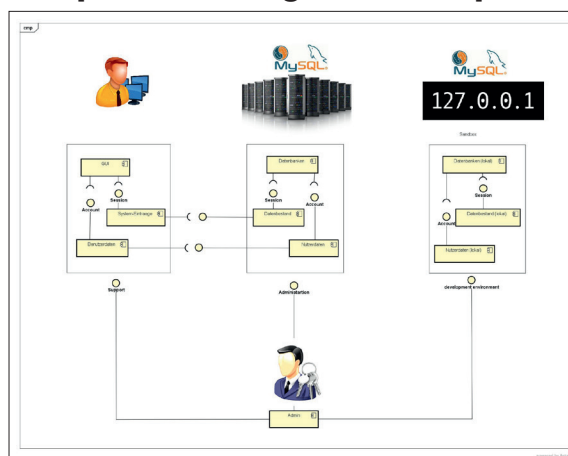
Verteilungs-Diagramm (Deployment)



<https://github.com/slimouGit/Rosetta-App/blob/master/development/documentary/UML/Deployment.jpg>

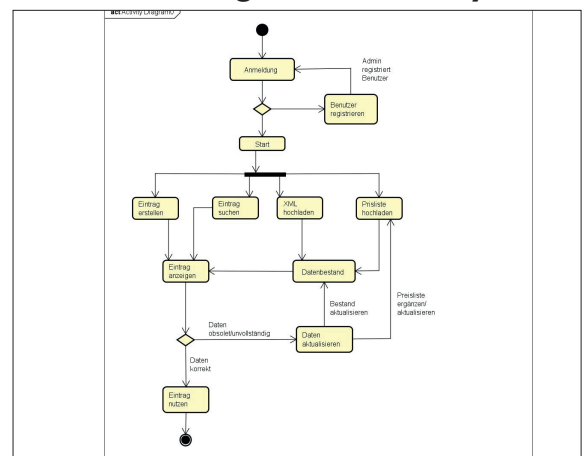
Bildquellen, alternative Bilder für User und Admin

Komponenten-Diagramm (Component)



<https://github.com/slimouGit/Rosetta-App/blob/master/development/documentary/UML/Component.jpg>

Aktivitäts-Diagramm (Activity)



<https://github.com/slimouGit/Rosetta-App/blob/master/development/documentary/UML/Activity.jpg>

– Prototyp

Mit der initialen Idee für mein Projekt und auf Basis des Exposés habe ich meine Arbeit damit begonnen, einen Prototypen zu entwickeln.

Während dieses Prozesses entstanden laufend neue Ideen, auch durch die beteiligten Stakeholder.

Gleichzeitig tauchten Probleme und Fragen auf, sei es in der Programmierung, der Infrastruktur oder darin, die bestmögliche Nutzung des Systems zu ermöglichen.

Meine Vorgehensweise, die iterative Entwicklung eines Prototyps vorab, ermöglichte es mir, eben diese Problematiken früh zu erkennen und zu behandeln/beheben.

Bevor der Prototyp in seinen Funktionen weitestgehend abgeschlossen und funktionsfähig war, habe ich nicht mit der Implementierung des eigentlichen Systems begonnen.

Der Prototyp ist online abrufbar: <http://prototype.rosetta-app.de>.

Es sind zwei Beispiel-User registriert, welche verschiedene Rechte habe:

Administrator: admin@rosetta-app.de Passwort: admin
User: user@rosetta-app.de Passwort: user

Screenshots des initialen Prototyps

initiale Dtaenanzeige

initiale Dateneingabe

Screenshots des abgeschlossenen Prototyps

Daeshboard

Suche und Datenausgabe

Anzeige aller registrierter Nutzer

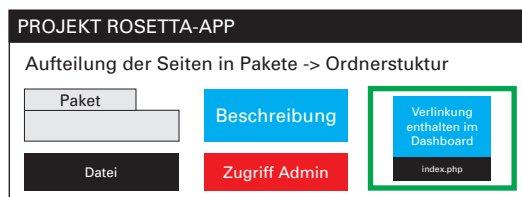
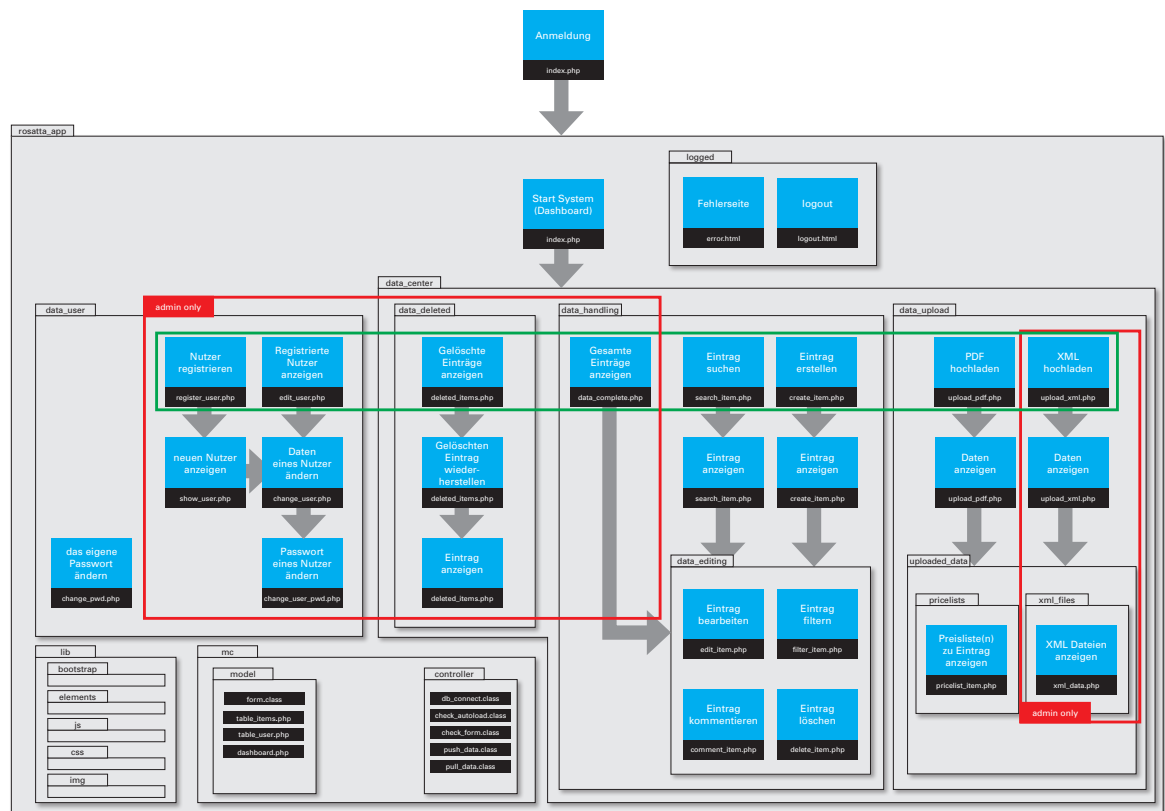
Daten ändern

- Sidemap

Nachdem die Implementierung des Prototyps abgeschlossen war, habe ich zu meiner Orientierung und besseren Übersicht und für eine sinnvollen Gliederung des Codes eine Sidemap entwickelt.

Ich habe eine Trennung von Code vorgenommen, d.h. der Code ist gegliedert in Kommunikation mit der Datenbank, dem Gerüst für die darzustellenden Daten und den Seiten, die die eigentlichen Daten anzeigen. Diese Herangehensweise entspricht der eines MVC-Patterns, wobei der Controller, wie zuvor erwähnt aktuell noch in den anderen Skripten implementiert ist.

In der folgenden Grafik sind alle Seiten des Systems mit einer jeweiligen Kurzbeschreibung in eine Ordnerstruktur unterteilt.



https://github.com/slimouGit/Rosetta-App/blob/master/development/sidemap/sidemap_Rosetta-App.pdf

- Verzeichnisstruktur

Folgender Verzeichnisstruktur ...

VERZEICHNIS-STRUKTUR

```

root_disrectory
- rosetta_app
- logged
- data_center
  - data_handling
  - data_editing
- data_upload
  - uploaded_data
  - pricelists
  - xml_files
- data_deleted
- data_user
- mc
  - model
  - view
  - controller
- lib
  - bootstrap
  - elements
  - js
  - css
  - img

```

– Datenstruktur

In den folgenden beiden Tabellen stelle ich die Datenstruktur/das Datenbankschema der Anwendung vor.

Mithilfe der beiden Tabellen „rosetta_data“ und „rosetta_user“ ist es möglich, ein weitreichendes Spektrum an Relationen darzustellen und umfangreiche Operationen im System zu gewährleisten.

Die **Datenbank-Tabellen sind in einer normalisierten Form** gehalten.

Die Beziehung beider Tabellen wird über den jeweiligen Nutzer hergestellt, der einen Tupel generiert, manipuliert oder löscht. Als eindeutige Kennung wird der vollständige Nutzernamen verwendet.

Neben der Bezeichnung der jeweiligen Spalten beinhaltet die folgende Abbildung den Datentyp und eine Kurzbeschreibung. Die in der folgenden Tabelle nicht dargestellte Kollation ist bei Textbasierten Werten immer auf „utf8_general_mysql500_ci“ gestellt. Die jeweiligen Primär-Schlüssel sind in beiden Tabellen „item_id“, bzw. „user_id“.

Zur besseren Übersicht habe ich die Tabellen jeweils in vier sinnngemäße Kategorien eingeteilt. In der Tabelle „rosetta_data“ sind dies beispielhaft: „Kennung“, „Daten“, „Zusatzinformationen“ und „create/update/delete Operationen“.

rosetta_data		
Spaltenbezeichnung	Typ	Beschreibung
Kennung		
token	varchar	eindeutige Kennung erhält jeder Datensatz bei Initialisierung
data_id	int	ID des Datensatzes
state	varchar	initial ist das Feld auf „active“, wird der Datensatz gelöscht, wechselt der Wert auf „deleted“
Daten		
item_de	text	Text Deutsch
item_de_comment	varchar	Kommentar Deutsch
user_de_comment	varchar	von wem ist das Kommentar
date_de_comment	varchar	Datum des Kommentars (hier als String)
item_fr	text	Text Französisch
item_fr_comment	varchar	Kommentar Französisch
user_fr_comment	varchar	von wem ist das Kommentar
date_fr_comment	varchar	Datum des Kommentars (hier als String)
item_it	text	Text Italienisch
item_it_comment	varchar	Kommentar Italienisch
user_it_comment	varchar	von wem ist das Kommentar
date_it_comment	varchar	Datum des Kommentars (hier als String)
Zusatzinformation		
category	varchar	Rubrik des Objekts
info	varchar	Info bzw. Objekt-Code
carline	varchar	enthalten in welchen Carlines (= Verlinkung zu PDFs)
create/update/delete Operationen		
user_create	varchar	wer hat den Datensatz erstellt (forename surname)
date_create	timestamp	wann wurde der Datensatz erstellt
user_update	varchar	wer hat den Datensatz aktualisiert (forename surname)
date_update	timestamp	wann wurde der Datensatz aktualisiert
user_delete	varchar	wer hat den Datensatz gelöscht (forename surname)
date_delete	varchar	wann wurde der Datensatz gelöscht (hier als String)

rosetta_users		
Spaltenbezeichnung	Typ	Beschreibung
Kennung		
user_id	int	ID des Users
authorization	varchar	Rechte User -> Differenzierung user/admin
Daten		
forename	varchar	Vorname
surname	varchar	Nachname
email	varchar	Email des Users
password	varchar	Password (php password_hash)
create/update Benutzer-Operationen		
create_user	timestamp	Erstellzeitpunkt User
update_user	timestamp	Änderungszeitpunkt User
Passwort ändern		
password_code	varchar	Temporäre Kennung (Sicherheitsaspekt, wenn Passwort neu angefordert wird)
password_date	timestamp	Zeitpunkt der Passwortanfrage

8. ANWENDUNG

In diesem Bereich gehe ich auf die schrittweise Umsetzung des Systems ein. Ich starte mit dem Screendesign und zeige, wie ich einen interaktiven Clickdummy mit Invision umgesetzt habe.

Anschließend stelle ich die fertige Anwendung vor und schließe diesen Abschnitt mit ein paar Features des Tools ab.

– Screendesign

Für das Screendesign habe ich mit dem 960 Grid System gearbeitet. Dieses Raster ist insgesamt 960px breit, hat 12 Spalten mit jeweils 60px. Jede Spalte hat 10px Randabstand nach rechts und links, also 20px Stege. Damit stehen 940px für den eigentlichen Inhalt zur Verfügung.

Ich habe mich für dieses gängige und flexible Spaltenlayout entschieden, weil ich das System unter Anwendung des Bootstrap-Frameworks verwirklicht habe und auch Bootstrap mit dieser Spalteneinteilung arbeitet.

<http://getbootstrap.com/examples/grid/>

Rosetta-App			Suchen	Erstellen	Preisliste hochladen	Administration	Passwort ändern		logout	
					Rubrik	Info/Code				
<div>Front- und Seitenairbag, Befahrer, manuell deaktivierbar</div> <div></div> <div>Post, sum enim quassum quae rem iudicat volupci acerbil et optur.</div> <div>Max Mustermann</div> <div>08.05.2017</div>	<div>airbag frontal et airbag latéral du passager avant désactivables manuellement</div> <div></div> <div>Imolendi dia conestquis ipsam, essit late volenesti di dolorem poreper ererum laboressunt.</div> <div>Max Mustermann</div> <div>08.05.2017</div>	<div>airbag frontale e laterale passeggero disattivabile manualmente</div> <div></div> <div>Kaboritam rem qui duaecum haritasit officosa.</div> <div>Max Mustermann</div> <div>08.05.2017</div>	Serienausstattung	Sicherheit						
					Enthalten in:		ADAM (df) (di), AstraST (df) (di), MokkaX (df) (di), insigniaST (df) (di), insigniaGS (df) (di), Zafira (df) (di)			
Erstellt von Martina Musterfrau am 03.05.2017 16:31 Uhr			Geändert von Max Mustermann am 05.05.2017 15:13 Uhr				ID 3032			
<div>1.4 Turbo 4x4 ECOTEC® Start/Stop 103 kW/140 PS</div> <div></div> <div>Imolendi dia conestquis ipsam, essit late volenesti di dolorem poreper ererum laboressunt.</div> <div>Max Mustermann</div> <div>08.05.2017</div>	<div>1.4 Turbo 4x4 ECOTEC® Start/Stop 103 kW/140 ch</div> <div></div> <div>Kaboritam rem qui duaecum haritasit officosa.</div> <div>Max Mustermann</div> <div>08.05.2017</div>	<div>1.6 ECOTEC® Start/Stop 85 kW/115 CV</div> <div></div> <div></div>	Benzin Motor	ECOTEC						
					Enthalten in:		ADAM (df) (di), AstraST (df) (di), MokkaX (df) (di), insigniaST (df) (di), insigniaGS (df) (di), Zafira (df) (di), ADAM (df) (di), AstraST (df) (di), MokkaX (df) (di), insigniaST (df) (di), insigniaGS (df) (di), Zafira (df) (di)			
Erstellt von Max Mustermann am 03.05.2017 16:31 Uhr							ID 3451			
<div>Antiblockiersystem (ABS) mit Kurvenbremskontrolle und Bremsassistent</div> <div></div> <div></div>	<div>Système antiblocage des roues (ABS) avec contrôle du freinage en courbe et assistant de freinage</div> <div></div> <div></div>	<div>Sistema antibloccaggio (ABS) con controllo della frenata in curva e assistenza alla frenata</div> <div></div> <div></div>	Benzin Motor	Benzin						
					Enthalten in:		General (df) (di)			
Erstellt von Max Mustermann am 03.05.2017 16:31 Uhr							ID 3142			

Das Screendesign erstellt habe ich mit den Programmen Photoshop, Illustrator und InDesign von Adobe.

Ich habe verschiedene Unterseiten entworfen, teilweise inklusive bestimmter Fallunterscheidungen wie zum Beispiel die Response des Systems nach dem Hochladen einer Datei.

Auf diese Weise konnte ich das fertige Screendesign zum Aufbau der späteren CSS-Datei nutzen, da ich alle nötigen Informationen im Bereich des Designs zur Verfügung hatte.

Die folgenden Screenshots stellen einen Ausschnitt aus dem Dokument dar. Das komplette Design ist in Form eines PDFs in meinem Repository auf Github verfügbar.

https://github.com/slimouGit/Rosetta-App/blob/master/development/screendesign/screendesign_Rosetta-App.pdf

Login Startseite

Daeshboards

Anzeige der Datensätze

Anzeige der im System registrierten Benutzer

neuen Eintrag manuel

bestehenden Eintrag bearbeiten

erfolgreicher Upload eines PDFs

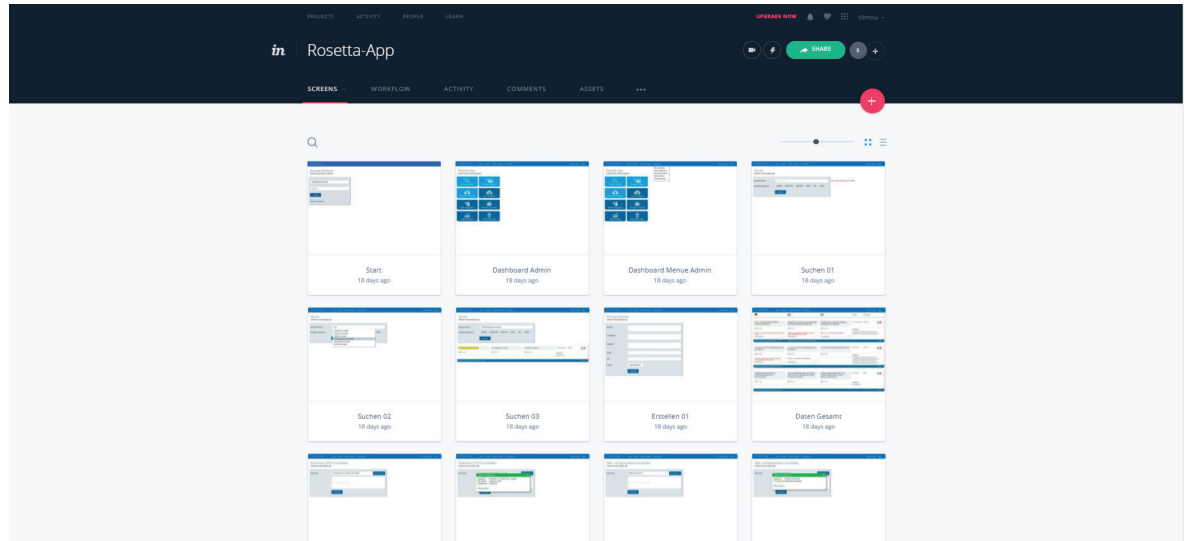
Autovervollständigung bei der Suche

– ClickDummy InVision

Sobald das Screendesign abgeschlossen war, konnte ich mithilfe der einzelnen Screens und InVision der InVision AG einen interaktiven Prototyp erstellt.

<https://projects.invisionapp.com/d/main#/console/10397535/232607158/preview>

Mithilfe dieses interaktiven Prototyps war es mir möglich, das Projekt in seinem späteren Erscheinungsbild zu Präsentieren.



<https://projects.invisionapp.com/d/main#/console/10397535/232607158/preview>

– fertige Anwendung

Orientierend am Aufbau des Prototyps und dem fertigen Screendesign konnte ich mit der Implementierung des Systems starten.

Hierbei galt es für mich vor allem einen sauberen Code zu generieren.

...

Screenshots

Die Anwendung läuft unter: <http://prototype.rosetta-app.de>

Es sind zwei Beispiel-User registriert, welche verschiedene Rechte haben:

Administrator:	admin@rosetta-app.de	Passwort: admin
User:	user@rosetta-app.de	Passwort: user

– Features

- Zwischenablage (JavaScript)
- Textarea scalable ...
- Autovervollständigung ...
- Div-Container statt table
warum?
https://wiki.selfhtml.org/wiki/HTML/Tutorials/Alternativen_zu_Tabellen
<http://t3n.de/news/css-tabellenlayout-ohne-table-element-349609/>
- Dateneinpflege
Dateneinpflege für Implementierung XML und manuell
- PDF hochladen
automatische Umbenennung
- Benutzerverwaltung
 - Admin kann Nutzerdaten ändern (inkl. Passwort)

- ein Nutzer kann sich nicht selber registrieren (aber Passwort ändern/vergessen)

9. ERGEBNISSE UND FAZIT

- **Wissensmanagement-Tool**

... ein spezialisiertes und vollwertiges Wissensmanagement-Tool aus ...

- **Probleme**

- UTF Codierung speziell bei der Suchabfrage
=> preg_replace
Unterschied lokal/Server

- **Gelerntes**

Projekt nach Vorgaben der Stakeholder/PHP-Kenntnisse erweitert/...
Praxis-Erfahrungen

10. AUSBLICK

- **was kann man machen**

Versionierung möglich ...

- **Aussicht**

System ist skalierbar (gesamter europäischer Markt)/generisch d.h. auf andere Projekte anpassbar ...

11. LITERATUR/QUELLEN/BILDNACHWEISE

- **Text**

Text ...

- **Bildquellen**

<http://www.freepik.com/free-icons> ...
