



Abschlussprüfung Sommer 2016

Fachinformatiker für Anwendungsentwicklung
Dokumentation zur betrieblichen Projektarbeit

Entwicklung einer Statistik-App

Webbasierte App für den ePages-App-Store zur statistischen
Analyse von KPIs

Abgabetermin: Gera, den 30.11.2016

Prüfungsbewerber:

Steven Hergt
Georg-Büchner-Str. 9
07749 Jena



Ausbildungsbetrieb:

ePages GmbH
Heinrich-Heine-Str. 1
07749 Jena

Dieses Werk einschließlich seiner Teile ist **urheberrechtlich geschützt**. Jede Verwertung außerhalb der engen Grenzen des Urheberrechtsgesetzes ist ohne Zustimmung des Autors unzulässig und strafbar. Das gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen sowie die Einspeicherung und Verarbeitung in elektronischen Systemen.

Inhaltsverzeichnis

Abbildungsverzeichnis	III
------------------------------	------------

Tabellenverzeichnis	IV
----------------------------	-----------

Listings	V
-----------------	----------

Abkürzungsverzeichnis	VI
------------------------------	-----------

1	Einleitung	1
1.1	Projektumfeld	1
1.2	Projektziel	1
1.3	Projektbegründung	1
1.4	Projektschnittstellen	2
1.5	Projektabgrenzung	3
2	Projektplanung	3
2.1	Projektphasen	3
2.2	Abweichungen vom Projektantrag	3
2.3	Ressourcenplanung	4
2.4	Entwicklungsprozess	4
3	Analysephase	4
3.1	Ist-Analyse	4
3.2	Wirtschaftlichkeitsanalyse	6
3.2.1	„Make or Buy“-Entscheidung	6
3.2.2	Projektkosten	6
3.2.3	Amortisationsdauer	7
3.3	Anwendungsfälle	8
3.4	Qualitätsanforderungen	8
3.5	Lastenheft/Fachkonzept	9
4	Entwurfsphase	11
4.1	Zielplattform	11
4.2	Architekturdesign	12
4.3	Entwurf der Benutzeroberfläche	12
4.4	Datenmodell	12
4.5	Geschäftslogik	13
4.6	Maßnahmen zur Qualitätssicherung	13
4.7	Pflichtenheft/Datenverarbeitungskonzept	14
4.8	Zwischenstand	14

Inhaltsverzeichnis

5	Implementierungsphase	14
5.1	Implementierung der Datenstrukturen	14
5.2	Implementierung der Benutzeroberfläche	14
5.3	Implementierung der Geschäftslogik	15
5.4	Zwischenstand	15
6	Abnahmephase	15
6.1	Zwischenstand	16
7	Einführungsphase	16
7.1	Zwischenstand	16
8	Dokumentation	16
8.1	Zwischenstand	17
9	Fazit	17
9.1	Soll-/Ist-Vergleich	17
9.2	Lessons Learned	17
9.3	Ausblick	18
	Literaturverzeichnis	19
A	Anhang	i
A.1	Detaillierte Zeitplanung	i
A.2	Lastenheft (Auszug)	i
A.3	Use Case-Diagramm	iii
A.4	Pflichtenheft (Auszug)	iii
A.5	Datenbankmodell	v
A.6	Oberflächenentwürfe	vi
A.7	Screenshots der Anwendung	viii
A.8	Entwicklerdokumentation	x
A.9	Testfall und sein Aufruf auf der Konsole	xii
A.10	Klasse: ComparedNaturalModuleInformation	xiii
A.11	Klassendiagramm	xvi
A.12	Benutzerdokumentation	xvii

Abbildungsverzeichnis

1	Widgetauswahl im Dashboard eines Merchant-Backoffice eines ePages Onlineshops . .	5
2	Graphische Darstellung von Bestellstatistiken	5
3	Auflistung von Artikelstatistiken	6
4	Amortisationszeit pro monatlicher Miete	8
5	Anwendungsfalldiagramm für die Statistik-App	9
6	Vereinfachtes ER-Modell	13
7	Prozess des Einlesens eines Moduls	13
8	Use Case-Diagramm	iii
9	Datenbankmodell	v
10	Liste der Module mit Filtermöglichkeiten	vi
11	Anzeige der Übersichtsseite einzelner Module	vii
12	Anzeige und Filterung der Module nach Tags	vii
13	Anzeige und Filterung der Module nach Tags	viii
14	Liste der Module mit Filtermöglichkeiten	ix
15	Aufruf des Testfalls auf der Konsole	xiii
16	Klassendiagramm	xvi

Tabellenverzeichnis

1	Zeitplanung	3
2	Kostenaufstellung	7
3	Entscheidungsmatrix	12
4	Zwischenstand nach der Entwurfsphase	14
5	Zwischenstand nach der Implementierungsphase	15
6	Zwischenstand nach der Abnahmephase	16
7	Zwischenstand nach der Einführungsphase	16
8	Zwischenstand nach der Dokumentation	17
9	Soll-/Ist-Vergleich	18

Listings

Listings/tests.php	xii
Listings/cnmi.php	xiii

Abkürzungsverzeichnis

API	Application Programming Interface
App	Applikation
CSV	Comma Separated Value
DOM	Document Object Model
EPK	Ereignisgesteuerte Prozesskette
ERM	Entity-Relationship-Modell
FTP	File Transfer Protocol
Git	Versionskontrollsystem
GitHub	Online-Dienst zur Verwaltung quelloffener Software
HTML	Hypertext Markup Language
MVC	Model View Controller
NatInfo	Natural Information System
PHP	Hypertext Preprocessor
QA	Quality Assurance
REST	Representational state transfer
REST-API	REST-Schnittstelle
SCP	Secure Copy
SFTP	SSH File Transfer Protocol
SQL	Structured Query Language
SSH	Secure Shell
UML	Unified Modeling Language
XML	Extensible Markup Language

1 Einleitung

1.1 Projektumfeld

Die ePages GmbH ist ein deutsches Software- und Dienstleistungsunternehmen, das Produkte zur Ermöglichung eines elektronischen Handels (E-Commerce) bereitstellt, d.h. Kunden können mit der Produktsoftware, die über Hosting-Provider wie Strato AG, 1 & 1, T-Online etc. vertrieben wird, einen individualisierten Onlineshop aufsetzen und ihn gegen eine monatliche Gebühr betreiben.

Die Firma wurde 1983 als “Beeck & Dahms GbR” von dem jetzigen Geschäftsführer Wilfried Beeck in Kiel gegründet und war später Teil der Intershop AG bis zur Absplittierung im Jahr 2002. Momentan arbeiten in der Firma insgesamt rund 180 Mitarbeiter. Der Hauptsitz der Firma ist in Hamburg, danach kommt Jena als Firmensitz mit rund 40 Mitarbeitern. Der Auftrag zur Erstellung der App kommt vom Produktmanagement und ist aus Kundenrückmeldungen entstanden. Innerhalb der Firma gibt es verschiedene mehr oder minder unabhängige Entwicklungsteams. Ich bin dabei Teil des ePages6-Core-Teams als Frontend-/Javascriptentwickler, das wiederum Teil der R&D-Abteilung ist. Die Projekterstellung findet halbtags während der Sprints statt, d.h. ich stehe daneben noch dem Team halbtägig zur Unterstützung zur Verfügung und gehe in den Dailys auch immer auf den Status meines Projektes ein.

1.2 Projektziel

Ziel des Projektes ist die Erstellung einer externen WebApp für Endkunden eines ePages Onlineshops. Mit dieser App soll es für den Kunden möglich sein spezielle KPIs für deren Onlineshop berichtsmäßig dokumentiert und deren zeitliche Entwicklung angezeigt zu bekommen. Daraus sollen Hinweise zum Anpreisen bestimmter Artikel resultieren. Auf alle relevanten Bestelldaten des Onlineshops soll per REST-API zugegriffen werden. Die wichtigsten KPIs sind hierbei der Umsatz und die meistverkauften Produkte. Die Berichte sollen anpassbar an frei wählbare Zeiträume und den Bezahlstatus sein.

1.3 Projektbegründung

Für Endkunden eines ePages-Onlineshops besteht standardmäßig die Möglichkeit über das Erstellmenü ihres Shops verschiedene Analysewidgets auszuwählen, die jedoch nur die wesentlichen KPIs als Umsatz- und Artikelverkaufsstatistiken bereitstellen ohne daraus spezielleren Handlungsbedarf des Händlers abzuleiten. Durch eine Nutzerumfrage wurde festgestellt, dass von den Händlern genauere Analysen des Käuferverhaltens (Kaufabbruchrate, Herkunft, Bestellzeiten etc.) und mehr Anpassungsmöglichkeiten (KPIs pro Kunde) gewünscht werden. Die Kunden haben zwar die Möglichkeit sich bei externen Firmen wie etracker für umfangreiche Statistikauswertungen für ihren Onlineshop anzumelden, jedoch besitzt das ePages System auch einen eigenen App-Store, der sich als Verkaufsplattform für eine eigens dafür programmierte Statistik-App ebenfalls anbietet, welche auf die Nutzerwünsche zugeschnitten ist und damit höhere Nutzerbindung und Nutzerzufriedenheit als Zielsetzung hat. Das

1 Einleitung

ist auch hinsichtlich der Vermarktung des neuesten Softwareproduktes von ePages sinnvoll, welches im nächsten Jahr ausgerollt wird und eine moderne Variante des Bestandsproduktes darstellt, wodurch neue Kunden gewonnen werden sollen und die Konkurrenzfähigkeit auf dem bestehenden Markt gewährleistet werden soll. Nicht zuletzt verdient ePages auch aktiv an der Vermarktung der App pro Nutzer.

1.4 Projektschnittstellen

Das Projekt wurde von meinem Ausbilder (Markus Höllein) und meinem direkten Disziplinarvorgesetzten (Mario Rieß) genehmigt, welcher stellvertretend für den Ausbildungsbetrieb spricht.

Die Projektmittel umfassen an Hardware einen leistungsstarken Laptop, zwei Monitore, ein externes Keyboard und eine Maus. An Software wird ein externer Zugang zu einem ePages-Developer-Shop benötigt, der exemplarisch als Anbindungsstelle für die zu entwickelnde App fungiert. Hard- und Software werden dabei vom Ausbildungsbetrieb bereitgestellt. Die App wird auf einem externen Server von “uberspace.de” gehostet, auf dem sich auch die MySQL-Datenbank der App befindet. Die monatlichen Kosten dafür übernehme ich selbst. Des Weiteren wird Git als Versionierungstool verwendet, wobei der firmeninterne GitHub-Zugang verwendet wird, um das Projekt auch extern auf dem GitHub-Server speichern zu können. Die Anbindung der App an die ePages-Software geschieht über die REST-API von ePages unter Verwendung einer speziellen PHP-Klasse¹. Dies geschieht in Zusammenarbeit mit erfahrenen Programmierern. Die finanziellen Mittel bzw. die Ausbildungsvergütung stellt die HR-Abteilung zur Verfügung.

Nutzer der App können alle ePages-Endkunden sein, die sich für die kostenbehaftete Nutzung der App für ihren Onlineshop entschließen. Das Ergebnis des Projekts muss zuallererst der teaminternen QA-Abteilung zum Testen präsentiert werden. Bestehen hier keine Bedenken mehr, wird die App dem Produktmanagement vorgelegt, welches letztendlich darüber entscheidet die App in dem ePages-App-Store zur Nutzung anzubieten oder ob weitere Verbesserungen notwendig sind.

Für die Programmierung der App werden das PHP-Mikroframework Slim² und verschiedene Javascript-Frameworks und Bibliotheken verwendet wie JQuery, React.js³ und D3.js⁴ bzw. die davon abgeleitete C3.js-Bibliothek⁵, welche allesamt kostenlos sind.

- Mit welchen anderen Systemen interagiert die Anwendung (technische Schnittstellen)?
- Wer genehmigt das Projekt bzw. stellt Mittel zur Verfügung?
- Wer sind die Benutzer der Anwendung?
- Wem muss das Ergebnis präsentiert werden?

¹<https://github.com/ePages-de/epages-rest-php>

²<http://www.slimframework.com/>

³<https://facebook.github.io/react/>

⁴<https://d3js.org/>

⁵<http://c3js.org/>

1.5 Projektabgrenzung

Das Projekt ist nicht Teil der ePages-Bestandssoftware oder irgendeines anderen ePages-Softwareproduktes, sondern stellt als externe App eine eigenständige Software dar, die mittels geeigneter API nicht nur an ePages, sondern auch an andere Onlineshopsoftware angebunden werden könnte. Vorrangig wird jedoch die Anbindung an ePages sein. Durch die Eigenständigkeit der App wird zudem gewährleistet, dass die Bestandssoftware bei Entwicklungsfehlern/Bugs in der App nicht in Mitleidenschaft gezogen wird.

Das Projekt wird nur soweit entwickelt, dass die Grundfunktionalitäten vorhanden sind, womit alle geplanten Anwendungsfälle realisiert werden können. Der Feinschliff durch das Feedback aus der QA-Abteilung wird aus Zeitgründen in seiner Gänze nicht mehr Teil dieses Projektes sein können genauso wie die Einbeziehung des Feedbacks vom Produktmanagement oder die Integration in den ePages-App-Store.

2 Projektplanung

2.1 Projektphasen

Die Gesamtprojektbearbeitungszeit ist auf 70 Stunden festgelegt. Der Start des Projekts ist der 10.10.2016 und der Abschluss ist der 30.11.2016. Die 70 Stunden werden auf den Zeitraum relativ gleichmäßig verteilt, sodass mindestens halbtäglich noch den Tagesgeschäften nachgegangen und das Team unterstützt werden kann. Außerdem muss gewährleistet werden, dass ich trotz des Projektes an den wichtigsten Scrum- und Team-Meetings teilnehmen kann, die die Arbeitszeit regelmäßig und unregelmäßig unterbrechen.

Tabelle 1 zeigt die grobe Zeitplanung.

Projektphase	Geplante Zeit
Analysephase	8 h
Entwurfsphase	6 h
Implementierungsphase	50 h
Erstellen der Dokumentation	6 h
Gesamt	70 h

Tabelle 1: Zeitplanung

Eine detailliertere Zeitplanung findet sich im Anhang [A.1: Detaillierte Zeitplanung](#) auf Seite [i](#).

2.2 Abweichungen vom Projektantrag

Eine wesentliches Ziel der App sollte die Darstellung statistischer Analysen verkaufter Artikel des Onlineshops sein, der die App nutzt. Während der Projektdurchführung wurde jedoch festgestellt,

3 Analysephase

dass die genutzte PHP-Order-Klasse für die REST-Calls noch keine Möglichkeit bietet Artikel bzw. Produkte aus den Shopbestellungen zu extrahieren, obwohl im Vorfeld fälschlicherweise davon ausgegangen wurde, dass diese Möglichkeit besteht⁶. Aus Zeitgründen war es bisher weder mir noch dem verantwortlichen Entwickler möglich die Klasse dahingehend zu erweitern, sodass auf Artikelstatistiken gänzlich verzichtet wurde. Des Weiteren wurde die Registrierungsmöglichkeit für Neukunden der App bisher nur soweit aufbereitet, dass alle Kunden automatisch den ePages-Developer-Test-Shop zugewiesen bekommen, da die Anbindung an echte Onlineshops noch gar nicht getestet werden kann. Dazu ist erst das Einverständnis des Produktmanagements nach Sichtung der App notwendig.

2.3 Ressourcenplanung

An Ressourcen ist ein PC-Arbeitsplatz (Schreibtisch, Rollcontainer, Schreibtischstuhl, Schreibtischlampe) in einem Firmenbüro vonnöten. Dazu kommt an Hardware ein LAN- und WLAN-fähiger COREi7-Laptop mit 16 GByte Arbeitsspeicher und mindestens 100 GByte großer Festplatte sowie zwei Monitore, eine Maus und ein Headset. An Software wird ein Windows 10 Betriebssystem, Sublime Text 3 als Text-Editor, ein Internetbrowser (Google Chrom und Firefox), Putty als [SSH-Client](#), WinSCP als [SFTP](#) und [FTP-Client-Software](#), ein externer Server, auf dem die App „gehostet“ wird bereitgestellt von „uberspace.de“, phpMyAdmin zur Verwaltung der Datenbank, HipChat als internes Chattool der Firma, JIRA von Atlassian als Verwaltungssoftware der agilen Entwicklungsprozesse unter Scrum sowie Confluence von Atlassian als interne Kollaborationssoftware für Teams. Dazu kommt die Verwendung von [Git](#) und [GitHub](#) zur Versionierungskontrolle. Darüberhinaus werden auch personelle Ressourcen beansprucht, die einen Backendentwickler zum Einrichten des epages Developer-Shops und des Slim-Frameworks sowie eine [QA-Kollegin](#) zum Testen der App-Funktionen und meinen Ausbilder zur generellen Überwachung meines Projektes umfassen.

2.4 Entwicklungsprozess

Für die Entwicklung wird das Wasserfallmodell verwendet, in dem bestimmte Meilensteine gesetzt und abgeschlossene Phasen definiert werden können. Dadurch werden insbesondere eine gut definierte und ausgereifte Planungs- und Entwurfsphase ermöglicht, die zur eigentlichen Implementierung notwendig sind. enü

3 Analysephase

3.1 Ist-Analyse

Derzeit haben Mieter eines ePages-Onlineshops die Möglichkeit im Backoffice ihres Onlineshop, in dem alle wichtigen Shopeinstellungen zu treffen sind, zwei Analysewidgets auszuwählen, siehe [Abbildung 1](#)

⁶<https://github.com/ePages-de/epages-rest-php/blob/master/src/shopobjects/order/Order.class.php>

3 Analysephase

Das erste zeigt eine Top-3-Liste der meistverkauften Artikel der letzten 30 Tage an und das zweite den Nettoumsatz sowie die Anzahl der Bestellungen für heute, gestern, diese Woche, letzte Woche, diesen und letzten Monat an. Des Weiteren wurde bereits auf einem zweitägigen firmeninternen Hackathon

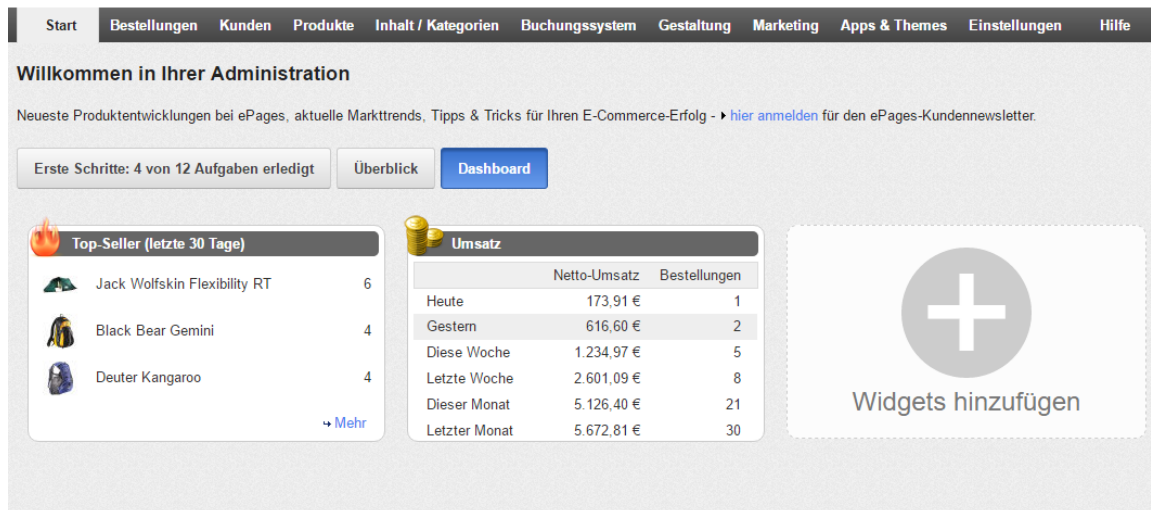


Abbildung 1: Widgetauswahl im Dashboard eines Merchant-Backoffice eines ePages Onlineshops

im Frühjahr 2016 der Versuch der Programmierung einer Statistik-App unternommen, wobei dabei nur mit „Mock-Daten“ hantiert wurde und keine echte Shopanbindung realisiert wurde. Die Ergebnisse zu dieser App sind auf einer internen Wikiseite veröffentlicht und liefern einen ersten Eindruck wie so eine App aussehen könnte und welche Funktionalitäten und Use-Cases ePages vorrangig für solch eine Statistik-App vorsieht. Zu sehen ist ein Menü, über das Bestellungs- und Artikelstatistiken anwählbar sind. In [Abbildung 2](#) sieht man wie eine graphische Darstellung von Umsätzen pro Bestellung aussehen könnte.

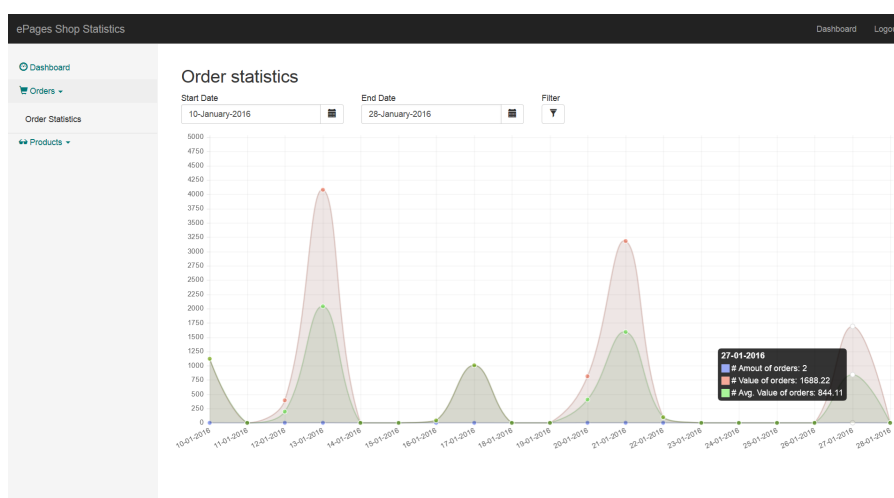


Abbildung 2: Graphische Darstellung von Bestellstatistiken

3 Analysephase

In [Abbildung 3](#) werden Artikel mit Gesamtumsatz und Bestellanzahl aufgelistet. Diese Beispiel-App verwendet für das Frontend das AngularJS-Framework und die Backendskripte sind in Ruby on Rails programmiert. Datenbankabfragen oder angelegte Datenbanken gibt es keine, genauso wenig eine Registrierungs- oder Loginfunktionalität.

The screenshot shows the 'ePages Shop Statistics' dashboard. On the left is a sidebar with navigation links: Dashboard, Orders, Products, Product Statistics, Most Sold Products, Most Ordered Products, and Highest Revenue Products. The main area displays a table with the following columns: Product Name, Purchases, Containing Orders, Total Revenue, and a link for 'Bundel suggestions'.

Product Name	Purchases	Containing Orders	Total Revenue	
Campingaz Twister 270	35	10	114.75	Bundel suggestions
Berghaus Pacille Jacket - Women	27	8	999.75	Bundel suggestions
Camping AZ CV470 valve Gas canister	33	12	15.9	Bundel suggestions
Eureka El Capitan IV	34	11	1699.75	Bundel suggestions
Camping AZ CV270 valve gas canister	9	4	3.95	Bundel suggestions
Meindl Air Revolution 2.0	27	9	169.95	Bundel suggestions
North Face Tadpole 23	26	8	1139.8	Bundel suggestions
Berghaus Pacille Jacket - Men	31	9	999.75	Bundel suggestions
Deuter Kangaroo	11	3	199.9	Bundel suggestions
Grangers Extreme Waterproof	24	8	74.75	Bundel suggestions
Deuter Teddy Bear	32	10	53.9	Bundel suggestions
Meindl RFS Tibet	24	10	189.95	Bundel suggestions
Grangers Extreme Cleaner	23	6	29.85	Bundel suggestions
Patrfinder ZG	26	6	779.85	Bundel suggestions
Deuter Hydro 2.0	28	8	149.9	Bundel suggestions
Jack Wolfskin Blizzard Jacket	13	4	1079.75	Bundel suggestions
Tatonka Vento	29	8	139.9	Bundel suggestions
Ederid Black Bear Rope	12	4	1.45	Bundel suggestions
Black Bear Gemini	14	5	59.9	Bundel suggestions
Mag-Lite Mini	18	5	50.85	Bundel suggestions
Jack Wolfskin Flexibility RT	20	5	1499.85	Bundel suggestions
Leatherman Tool Survival	36	9	291.8	Bundel suggestions

Abbildung 3: Auflistung von Artikelstatistiken

3.2 Wirtschaftlichkeitsanalyse

3.2.1 „Make or Buy“-Entscheidung

Abgesehen von den im MBO hinzufügbaren Statistik-Wigdetts fehlen tiefergehende statistische Analysen und grafische Darstellungen der wichtigsten und interessantesten KPIs, die sich dynamisch über REST-Calls aktualisieren lassen. Darunter fallen die Möglichkeit der Berichterstattung von KPIs pro Kunde und genauere Analysen des Käuferverhaltens (Kaufabbruchrate, Herkunft, Bestellzeiten etc.). Der Wunsch nach einer auf das ePages-System zugeschnittenen App für den ePages-App-Store wurde direkt vom Produktmanagement geäußert und sollte bereits schon mal während eines zweitägigen Hackathons entwickelt werden, was jedoch aufgrund des zu hohem Zeitaufwands für die Entwicklung bei weitem nicht fertig gestellt werden konnte. Durch diese App verspricht sich das Produktmanagement eine höhere Kundenzufriedenheit und damit eine bessere Kundenbindung an die ePages Softwareprodukte sowie längerfristig eine Anwerbung neuer Kunden, die sich durch die Existenz dieser speziellen App vorzugsweise für ePages anstatt für eine andere Onlineshopsoftware entscheiden würden.

3.2.2 Projektkosten

Die Kosten für die Durchführung des Projektes setzen sich für die 70 Stunden Bearbeitungszeit sowie aus Personal- als auch Ressourcenkosten zusammen.

3 Analysephase

Berechnung der Entwicklungskosten für ePages Laut Arbeitsvertrag liegt meine Ausbildungsvergütung im aktuellen Lehrjahr bei 680 € pro Monat. Damit verursache ich der Firma jährliche Kosten in Höhe von

$$680 \text{ €/Monat} \cdot 12 \text{ Monate/Jahr} = 8160 \text{ €/Jahr.} \quad (1)$$

Die Anzahl der Arbeitstage 2016 belaufen sich auf 252 Tage in Thüringen (Jena). Davon stehen mir 25 Urlaubstage zu. Es verbleiben $(252 - 25) \text{ Tage} = 227 \text{ Tage}$ vollwertige achtstündige Arbeitstage. Meine Stundenlohn berechnet sich damit zu

$$8 \text{ h/Tag} \cdot 227 \text{ Tage/Jahr} = 1816 \text{ h/Jahr.} \quad (2)$$

$$\frac{8160 \text{ €/Jahr}}{1816 \text{ h/Jahr}} \approx 4,49 \text{ €/h.} \quad (3)$$

Für die Nutzung der Ressourcen⁷ wird ein pauschaler Stundensatz von 15 € angenommen. Für die anderen Mitarbeiter wird pauschal ein Stundenlohn von 25 € angenommen. Eine Aufstellung der Kosten befindet sich in Tabelle 2 und sie betragen insgesamt 1844,30 €.

Vorgang	Zeit	Kosten pro Stunde	Kosten
Entwicklungskosten	70 h	4,49 € + 15 € = 19,49 €	1364,30 €
Unterstützung durch Mitarbeiter	10 h	25 € + 15 € = 40 €	400,00 €
Abnahmetest durch QA	2 h	25 € + 15 € = 40 €	80,00 €
			1844,30 €

Tabelle 2: Kostenaufstellung

3.2.3 Amortisationsdauer

Geplant ist die fertige App den Nutzern der ePages-Software zur monatlichen Miete zur Verfügung zu stellen. Der Mietpreis wird in dem Bereich [5 €, 50 €] liegen. Die angenommene Zahl an Nutzern liegt in dem Bereich [1, 20000].

Berechnung der Amortisationszeit Für den Umsatz ($=U$), die diese App abhängig von der Zeit in Monaten ($=t_m$), den Nutzern ($=N$) und von dem Mietpreis pro Monat ($=p_m$) erwirtschaften soll wird die Formel $U = p_m \cdot N \cdot t_m$ zugrunde gelegt. Es wird angenommen, dass die App nach einem Monat 10 Nutzer hat. Dieser Wert wird modellhaft als linear ansteigend mit der Zeit t_m angenommen, d.h. $N = 10 \cdot t_m$. Daraus lässt eine Formel zur Berechnung der Amortisationszeit (t_m^A) ableiten:

⁷Laptop, Monitore, Servernutzung, Büro- und Firmenräumlichkeiten, Stromverbrauch, Heizkosten etc.

$$t_m^A = \frac{U}{p_m \cdot N}, \quad \text{wobei } U = 1844,30 \text{ €, } N = 10 \cdot t_m^A \quad (4)$$

$$\Rightarrow t_m^A = \sqrt{\frac{1844,30 \text{ €}}{10 \cdot p_m}} = 13,58 \cdot p_m^{-\frac{1}{2}}. \quad (5)$$

Aus dem Funktionsgraphen aus [Abbildung 4](#) lässt sich die Amortisationszeit t_m^A für jeden möglichen Mietpreis pro Monat ablesen, wobei die Unsicherheit über das Ergebnis bei kleinen Monatsmietpreisen am geringsten ist, so liegt beispielsweise die Amortisationszeit für 5 € Monatsmiete bei ca. 6 Monaten, was durchaus im akzeptablen Budget- bzw. Investitionsrahmen der Firma ist.

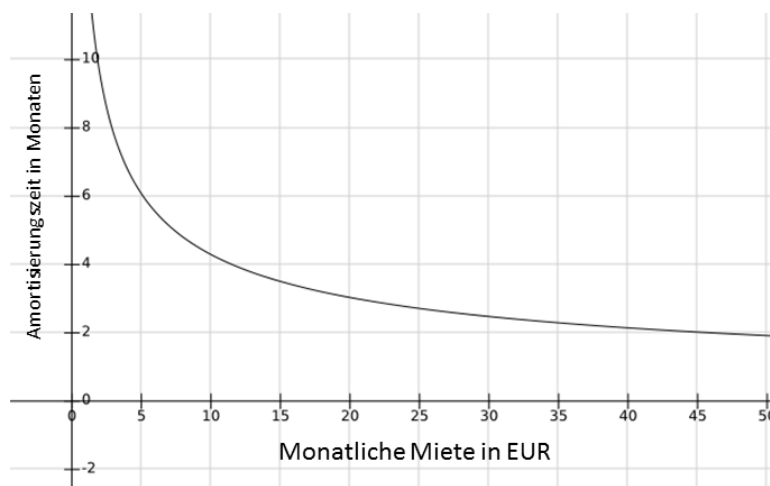


Abbildung 4: Amortisationszeit pro monatlicher Miete

3.3 Anwendungsfälle

Die zu entwickelnde App soll die folgenden Anwendungsfälle für den Nutzer/Merchant dargestellt in einem Anwendungsfalldiagramm mindestens realisiert haben.

3.4 Qualitätsanforderungen

Die App soll intuitiv vom Benutzer bedien- bzw. navigierbar sein. Alle dargestellten Statistiken und Berichte sollen selbsterklärend sein. Die App wird als Single-Page-Anwendung programmiert, d.h. bei keiner Aktion des Benutzer soll ein Seitenneuladen ausgelöst werden. Bei Aktualisierung der Daten durch Datenbankabfragen, wenn beispielsweise die Währung oder der Zeitraum geändert wird, sollen alle betroffenen Diagramme und Berichte nahezu gleichzeitig mit den neuen Daten aktualisiert werden. Bei der Datensynchronisation soll der Nutzer eine Rückmeldung darüber erhalten, ob die Synchronisation noch im Gange ist oder nicht oder gar fehlgeschlagen ist. Zusätzlich soll für jeden Nutzer beim Login eine einstündig gültige Session gestartet werden, sodass die URL für diesen Client einzigartig

3 Analysephase

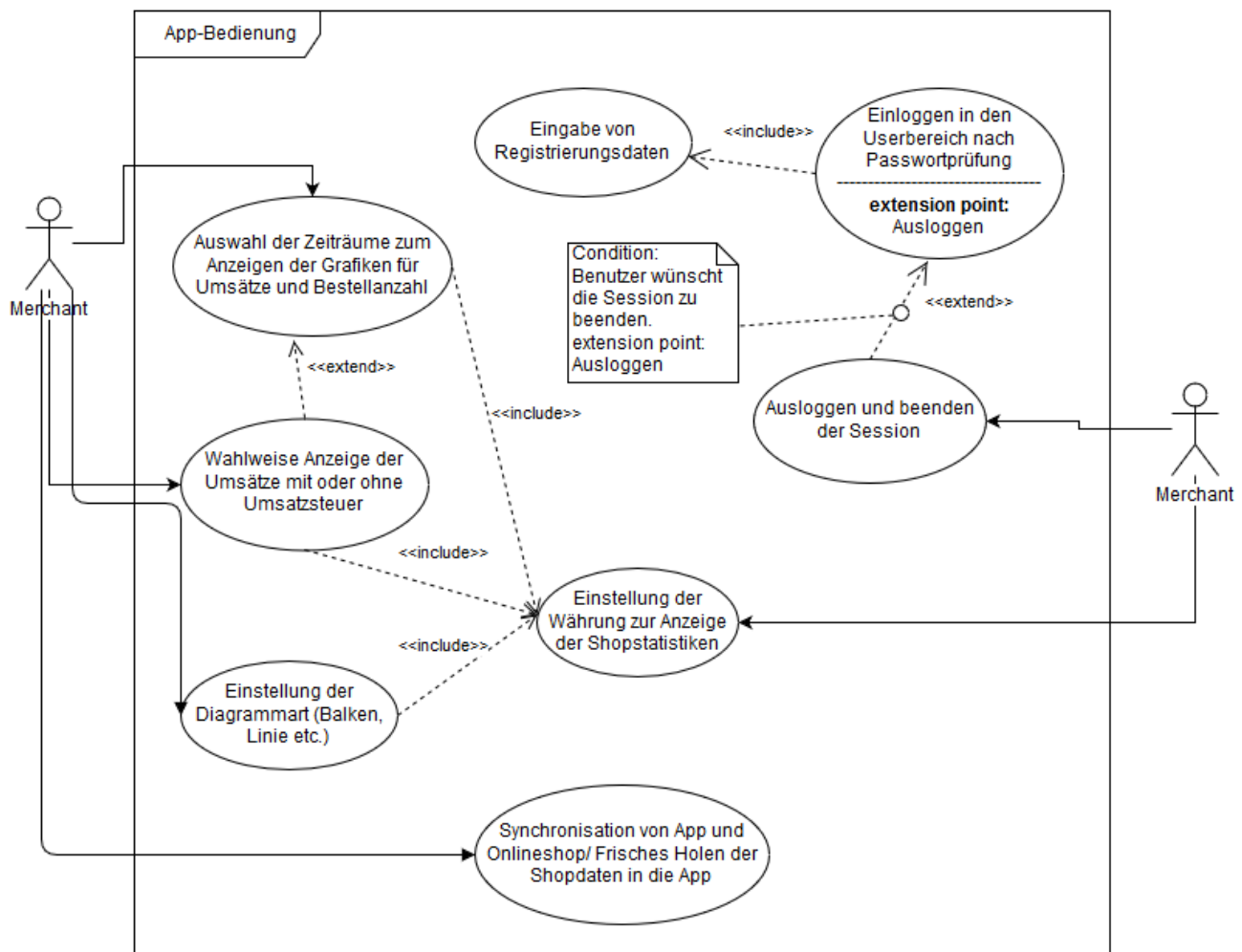


Abbildung 5: Anwendungsfalldiagramm für die Statistik-App

ist und bei keinem anderen Clientcomputer oder Browser gleichzeitig verwendet werden kann. Das Design sollte für den Nutzer ansprechend sein und die App sollte in Ansätzen „responsive“ sein, um auch auf mobilen Geräten benutzbar sein zu können. Die App soll frontend- und backendseitig modular aufgebaut sein, so dass Fehlerbeseitigung oder Modifizierungen schnell und übersichtlich möglich sind und sich konsistent in das Gesamtprogramm einfügen. Der Programmcode sollte strukturiert nach gängigen oder firminternen Code-Conventions sein und an allen wichtigen Stellen Kommentare für Entwickler enthalten.

3.5 Lastenheft/Fachkonzept

Die Anwendung muss folgende Anforderungen erfüllen:

1. Registrierungsmechanismus

- 1.1. Der Nutzer muss Vorname, Name, Email-Adresse, Passwort und seine ShopUrl zur Registrierung eingeben können.

3 Analysephase

- 1.2. Die Passworteingabe sollte blicksicher sein und verifiziert werden.
- 1.3. Nach der Registrierung bekommt man eine Erfolgsmeldung und wird auf die Loginseite weitergeleitet.
2. Login-Mechanismus
 - 2.1. Die Logindaten sollen die Email-Adresse und das Passwort sein
 - 2.2. Bei Falscheingabe von Email-Adresse oder Passwort soll eine entsprechende Rückmeldung erfolgen.
3. Logout-Mechanismus
 - 3.1. Im Nutzerbereich der App soll eine Möglichkeit zum Beenden der Session/Logout geschaffen werden.
 - 3.2. Beim Ausloggen soll die Session beendet und eine Weiterleitung auf den Loginbereich erfolgen.
4. Navigation im Nutzerbereich
 - 4.1. Alle wesentlichen Benutzerinteraktionen der App sollen nur über ein einziges Navigationspanel entweder an der Seite oder oben als Leiste durchführbar sein
 - 4.2. Die Navigation sollte einen logischen und intuitiven Aufbau haben und sollte vom Design her einheitlich sein
5. Graphische Darstellungen statistischer Auswertungen
 - 5.1. Die Umsätze und Bestellungen im Onlineshop sollen wahlweise als Balken- oder Liniendiagramme darstellbar sein.
 - 5.2. Für die Diagramme soll der Zeitraum auswählbar über die Navigationsleiste sein. Die auswählbaren Zeiträume sollen mindestens die einzelnen Kalenderwochen, Monate und Jahre umfassen.
 - 5.3. Es soll möglich sein, die Diagramme wahlweise aufgrund von Brutto - oder Nettoumsätzen anzuzeigen.
6. WährungsfILTER
 - 6.1. Die Anzeige der Statistiken und Berichte sollte abhängig von der ausgewählten Währung sein, da im Onlineshop mehrere Währungen eingestellt sein können.
 - 6.2. Hierzu muss eine Auswahlmöglichkeit über einen WährungsfILTER geschaffen werden.
7. Datensynchronisation

4 Entwurfsphase

- 7.1. Es soll die Möglichkeit für den Nutzer geschaffen werden über einen „Button“ die Datensynchronisation zwischen seinem Shop und der App auszulösen, falls der Nutzer eine neuerliche Synchronisation für notwendig hält.
 - 7.2. Der Nutzer soll eine Rückmeldung über den Prozess der Synchronisation bekommen und eine entsprechende Mitteilung falls die Synchronisation fehlgeschlagen ist.
 - 7.3. Die Daten sollen über REST-Calls unter Verwendung der ePages REST-API vom Onlineshop angefordert und in einer MySQL-Datenbank abgespeichert werden.
8. Datenbank
- 8.1. Zum Aktualisieren der App-Statistiken durch Benutzerinteraktionen abseits der Datensynchronisation sollen keine REST-Calls mehr ausgelöst werden, um die ePages-REST-API nicht zu sehr zu belasten. Vielmehr sollen hier die benötigten Daten aus der Datenbank abgefragt werden.
 - 8.2. Es soll ein Datenbankmodell in der 3. Normalform entworfen werden.
9. Umsatzstatistik nach Bundesland
- 9.1. Es soll über ein Tortendiagramm möglich sein, die Umsatzverteilung pro Bundesland angezeigt zu bekommen.
 - 9.2. Außerdem sollen Shopkunden dabei berücksichtigt werden, denen kein Bundesland zugewiesen ist oder die im Ausland wohnen.
10. Darstellung umsatzstärkster Kunden
- 10.1. Die App soll in tabellarischen Stil die Kunden geordnet nach Umsatz auflisten mit Namen, Email-Adresse, Gesamtumsatz, Gesamtzahl an Bestellungen und letztem Bestelldatum.
 - 10.2. Die Erstellung der Tabelle sollte dynamisch erfolgen.
11. Design
- 11.1. Die gesamte App sollte zumindest ein grundlegendes responsives Verhalten aufgrund des 12er-Grid-Systems zeigen.

4 Entwurfsphase

4.1 Zielplattform

Die Statistik-App wird als Web-Applikation entwickelt und sollte damit lauffähig in allen gängigen Internetbrowsern eines jeden Heim-PCs oder Laptops sein, mindestens jedoch im Internet-Explorer,

4 Entwurfsphase

Firefox und Google-Chrome. Die App ist über eine feste URL auf dem „uberspace.de“-Server erreichbar. Die Wahl auf „uberspace“ als Host fiel aufgrund der geringen monatlichen Mietkosten für das Hosting und die Servernutzung und der Unterstützung von MySQL. Die verwendeten Programmiersprachen umfassen Javascript und PHP. Dies ist der Tatsache geschuldet, dass viele meiner Kollegen und mein Ausbilder ein großes Know-How in diesen Sprachen haben und mich so effektiv unterstützen können. Des Weiteren ist für mich als Auszubildender in der Frontendentwicklung Javascript die Programmiersprache, in der ich mich am besten auskenne. Die Wahl auf PHP als serverseitige Sprache fiel aufgrund der großen Community im Vergleich zu NodeJS und der einfachen Datenbankanbindung an eine MySQL-Datenbank. NodeJS wird zwar für Single-Page-Anwendungen empfohlen, jedoch sollte man für rechenintensive Anwendungen, wie die Statistik-App eine ist, dann doch eher auf PHP zurückgreifen. Zudem gibt es von ePages auf der App-Entwicklungs-Webseite Tutorials zur App-Entwicklung in PHP und auch bereits eine eigene PHP-Klasse zur Anforderung von REST-Daten. Prinzipiell hätte man für diese Anwendung auch MongoDB als Datenbanksystem nehmen können. Das Vorhaben wurde aber mangels Erfahrung im Umgang damit verworfen.

- Beschreibung der Kriterien zur Auswahl der Zielplattform (u. a. Programmiersprache, Datenbank, Client/Server, Hardware).

4.2 Architekturdesign

- Beschreibung und Begründung der gewählten Anwendungsarchitektur (z. B. [MVC](#)).
- Ggfs. Bewertung und Auswahl von verwendeten Frameworks sowie ggfs. eine kurze Einführung in die Funktionsweise des verwendeten Frameworks.

Beispiel Anhand der Entscheidungsmatrix in Tabelle 3 wurde für die Implementierung der Anwendung das [PHP](#)-Framework [Symfony](#)⁸ ausgewählt.

Eigenschaft	Gewichtung	Akelos	CakePHP	Symfony	Eigenentwicklung
Dokumentation	5	4	3	5	0
Reengineerung	3	4	2	5	3
Generierung	3	5	5	5	2
Testfälle	2	3	2	3	3
Standardaufgaben	4	3	3	3	0
Gesamt:	17	65	52	73	21
Nutzwert:		3,82	3,06	4,29	1,24

Tabelle 3: Entscheidungsmatrix

⁸Vgl. [SENSIO LABS](#) [2010].

4.3 Entwurf der Benutzeroberfläche

- Entscheidung für die gewählte Benutzeroberfläche (z. B. GUI, Webinterface).
- Beschreibung des visuellen Entwurfs der konkreten Oberfläche (z. B. Mockups, Menüführung).
- Ggfs. Erläuterung von angewendeten Richtlinien zur Usability und Verweis auf Corporate Design.

Beispiel Beispielentwürfe finden sich im Anhang [A.6: Oberflächenentwürfe](#) auf Seite [vi](#).

4.4 Datenmodell

- Entwurf/Beschreibung der Datenstrukturen (z. B. [ERM](#) und/oder Tabellenmodell, [XML](#)-Schemas) mit kurzer Beschreibung der wichtigsten (!) verwendeten Entitäten.

Beispiel In [Abbildung 6](#) wird ein Entity-Relationship-Modell ([ERM](#)) dargestellt, welches lediglich Entitäten, Relationen und die dazugehörigen Kardinalitäten enthält.

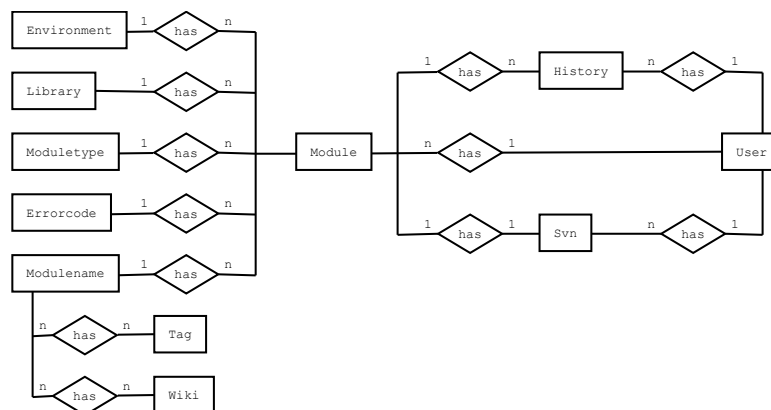


Abbildung 6: Vereinfachtes ER-Modell

4.5 Geschäftslogik

- Modellierung und Beschreibung der wichtigsten (!) Bereiche der Geschäftslogik (z. B. mit Komponenten-, Klassen-, Sequenz-, Datenflussdiagramm, Programmablaufplan, Struktogramm, Ereignisgesteuerte Prozesskette ([EPK](#))).
- Wie wird die erstellte Anwendung in den Arbeitsfluss des Unternehmens integriert?

4 Entwurfsphase

Beispiel Ein Klassendiagramm, welches die Klassen der Anwendung und deren Beziehungen untereinander darstellt kann im Anhang A.11: [Klassendiagramm](#) auf Seite xvi eingesehen werden.

Abbildung 7 zeigt den grundsätzlichen Programmablauf beim Einlesen eines Moduls als EPK.

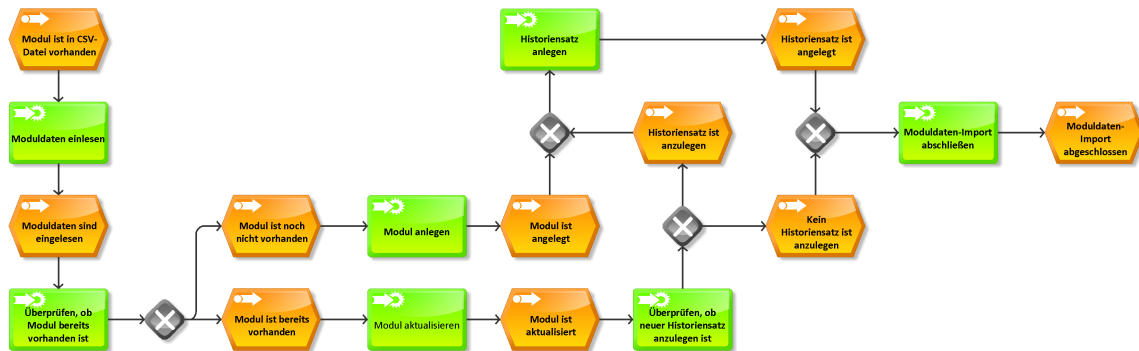


Abbildung 7: Prozess des Einlesens eines Moduls

4.6 Maßnahmen zur Qualitätssicherung

- Welche Maßnahmen werden ergriffen, um die Qualität des Projektergebnisses (siehe Kapitel 3.4: [Qualitätsanforderungen](#)) zu sichern (z. B. automatische Tests, Anwendertests)?
- Ggfs. Definition von Testfällen und deren Durchführung (durch Programme/Benutzer).

4.7 Pflichtenheft/Datenverarbeitungskonzept

- Auszüge aus dem Pflichtenheft/Datenverarbeitungskonzept, wenn es im Rahmen des Projekts erstellt wurde.

Beispiel Ein Beispiel für das auf dem Lastenheft (siehe Kapitel 3.5: [Lastenheft/Fachkonzept](#)) aufbauende Pflichtenheft ist im Anhang A.4: [Pflichtenheft \(Auszug\)](#) auf Seite iii zu finden.

4.8 Zwischenstand

Tabelle 4 zeigt den Zwischenstand nach der Entwurfsphase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Prozessentwurf	2 h	3 h	+1 h
2. Datenbankentwurf	3 h	5 h	+2 h
3. Erstellen von Datenverarbeitungskonzepten	4 h	4 h	
4. Benutzeroberflächen entwerfen und abstimmen	2 h	1 h	-1 h
5. Erstellen eines UML-Komponentendiagramms	4 h	2 h	-2 h
6. Erstellen des Pflichtenhefts	4 h	4 h	

Tabelle 4: Zwischenstand nach der Entwurfsphase

5 Implementierungsphase

5.1 Implementierung der Datenstrukturen

- Beschreibung der angelegten Datenbank (z. B. Generierung von [SQL](#) aus Modellierungswerkzeug oder händisches Anlegen), [XML](#)-Schemas usw..

5.2 Implementierung der Benutzeroberfläche

- Beschreibung der Implementierung der Benutzeroberfläche, falls dies separat zur Implementierung der Geschäftslogik erfolgt (z. B. bei [HTML](#)-Oberflächen und Stylesheets).
- Ggfs. Beschreibung des Corporate Designs und dessen Umsetzung in der Anwendung.
- Screenshots der Anwendung

Beispiel Screenshots der Anwendung in der Entwicklungsphase mit Dummy-Daten befinden sich im Anhang [A.7: Screenshots der Anwendung](#) auf Seite [viii](#).

5.3 Implementierung der Geschäftslogik

- Beschreibung des Vorgehens bei der Umsetzung/Programmierung der entworfenen Anwendung.
- Ggfs. interessante Funktionen/Algorithmen im Detail vorstellen, verwendete Entwurfsmuster zeigen.
- Quelltextbeispiele zeigen.
- Hinweis: Wie in Kapitel [1: Einleitung](#) zitiert, wird nicht ein lauffähiges Programm bewertet, sondern die Projektdurchführung. Dennoch würde ich immer Quelltextausschnitte zeigen, da sonst Zweifel an der tatsächlichen Leistung des Prüflings aufkommen können.

Beispiel Die Klasse `ComparedNaturalModuleInformation` findet sich im Anhang [A.10: Klasse: ComparedNaturalModuleInformation](#) auf Seite [xiii](#).

5.4 Zwischenstand

Tabelle 5 zeigt den Zwischenstand nach der Implementierungsphase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Anlegen der Datenbank	1 h	1 h	
2. Umsetzung der HTML-Oberflächen und Stylesheets	4 h	3 h	-1 h
3. Programmierung der PHP-Module für die Funktionen	23 h	23 h	
4. Nächtlichen Batchjob einrichten	1 h	1 h	

Tabelle 5: Zwischenstand nach der Implementierungsphase

6 Abnahmephase

- Welche Tests (z. B. Unit-, Integrations-, Systemtests) wurden durchgeführt und welche Ergebnisse haben sie geliefert (z. B. Logs von Unit Tests, Testprotokolle der Anwender)?
- Wurde die Anwendung offiziell abgenommen?

Beispiel Ein Auszug eines Unit Tests befindet sich im Anhang [A.9: Testfall und sein Aufruf auf der Konsole](#) auf Seite [xii](#). Dort ist auch der Aufruf des Tests auf der Konsole des Webserverns zu sehen.

6.1 Zwischenstand

Tabelle 6 zeigt den Zwischenstand nach der Abnahmephase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Abnahmetest der Fachabteilung	1 h	1 h	

Tabelle 6: Zwischenstand nach der Abnahmephase

7 Einführungsphase

- Welche Schritte waren zum Deployment der Anwendung nötig und wie wurden sie durchgeführt (automatisiert/manuell)?
- Wurden ggfs. Altdaten migriert und wenn ja, wie?
- Wurden Benutzerschulungen durchgeführt und wenn ja, Wie wurden sie vorbereitet?

7.1 Zwischenstand

Tabelle 7 zeigt den Zwischenstand nach der Einführungsphase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Einführung/Benutzerschulung	1 h	1 h	

Tabelle 7: Zwischenstand nach der Einführungsphase

8 Dokumentation

- Wie wurde die Anwendung für die Benutzer/Administratoren/Entwickler dokumentiert (z. B. Benutzerhandbuch, [API-Dokumentation](#))?
- Hinweis: Je nach Zielgruppe gelten bestimmte Anforderungen für die Dokumentation (z. B. keine IT-Fachbegriffe in einer Anwenderdokumentation verwenden, aber auf jeden Fall in einer Dokumentation für den IT-Bereich).

Beispiel Ein Ausschnitt aus der erstellten Benutzerdokumentation befindet sich im Anhang [A.12: Benutzerdokumentation](#) auf Seite [xvii](#). Die Entwicklerdokumentation wurde mittels PHPDoc⁹ automatisch generiert. Ein beispielhafter Auszug aus der Dokumentation einer Klasse findet sich im Anhang [A.8: Entwicklerdokumentation](#) auf Seite [x](#).

8.1 Zwischenstand

Tabelle 8 zeigt den Zwischenstand nach der Dokumentation.

⁹Vgl. PHPDOC.ORG [2010]

Vorgang	Geplant	Tatsächlich	Differenz
1. Erstellen der Benutzerdokumentation	2 h	2 h	
2. Erstellen der Projektdokumentation	6 h	8 h	+2 h
3. Programmdokumentation	1 h	1 h	

Tabelle 8: Zwischenstand nach der Dokumentation

9 Fazit

9.1 Soll-/Ist-Vergleich

- Wurde das Projektziel erreicht und wenn nein, warum nicht?
- Ist der Auftraggeber mit dem Projektergebnis zufrieden und wenn nein, warum nicht?
- Wurde die Projektplanung (Zeit, Kosten, Personal, Sachmittel) eingehalten oder haben sich Abweichungen ergeben und wenn ja, warum?
- Hinweis: Die Projektplanung muss nicht strikt eingehalten werden. Vielmehr sind Abweichungen sogar als normal anzusehen. Sie müssen nur vernünftig begründet werden (z. B. durch Änderungen an den Anforderungen, unter-/überschätzter Aufwand).

Beispiel (verkürzt) Wie in Tabelle 9 zu erkennen ist, konnte die Zeitplanung bis auf wenige Ausnahmen eingehalten werden.

Phase	Geplant	Tatsächlich	Differenz
Entwurfsphase	19 h	19 h	
Analysephase	9 h	10 h	+1 h
Implementierungsphase	29 h	28 h	-1 h
Abnahmetest der Fachabteilung	1 h	1 h	
Einführungsphase	1 h	1 h	
Erstellen der Dokumentation	9 h	11 h	+2 h
Pufferzeit	2 h	0 h	-2 h
Gesamt	70 h	70 h	

Tabelle 9: Soll-/Ist-Vergleich

9.2 Lessons Learned

- Was hat der Prüfling bei der Durchführung des Projekts gelernt (z. B. Zeitplanung, Vorteile der eingesetzten Frameworks, Änderungen der Anforderungen)?

9.3 Ausblick

- Wie wird sich das Projekt in Zukunft weiterentwickeln (z. B. geplante Erweiterungen)?

Literaturverzeichnis

ISO/IEC 9126-1 2001

ISO/IEC 9126-1: *Software-Engineering – Qualität von Software-Produkten – Teil 1: Qualitätsmodell*. Juni 2001

phpdoc.org 2010

PHPDOC.ORG: *phpDocumentor-Website*. Version: 2010. <http://www.phpdoc.org/>, Abruf: 20.04.2010

Sensio Labs 2010

SENSIO LABS: *Symfony - Open-Source PHP Web Framework*. Version: 2010. <http://www.symfony-project.org/>, Abruf: 20.04.2010

A Anhang

A.1 Detaillierte Zeitplanung

Analysephase	8 h
1. Analyse des Ist-Zustands	1 h
1.1. Studium des Blogeintrags zur gewünschten App und Verschriftlichung	1 h
2. Wirtschaftlichkeitsprüfung und Amortisationsrechnung	2 h
3. Erstellen eines „Use-Case“-Diagramms	2 h
4. Erstellung eines Lastenheftes	3 h
Entwurfsphase	6 h
1. Erstellung eines Pflichtenheftes	3 h
2. Auswahl eines geeigneten Designs	2 h
3. Erstellung eines UML-Klassendiagramms	1 h
Implementierungsphase	50 h
1. Einrichtung eines ePages-Developer-Shops für die App-Entwicklung	0,5 h
2. Einrichtung des Slim-Frameworks zur Anbindung an die ePages REST-API	1,5 h
3. Implementierung des DOM-Gerüsts mit React.js	16 h
3.1. Routing der REST-Calls von Slim zu React	1 h
4. Datenbankerstellung	4 h
4.1. Erstellung der MySQL-Datenbank	1 h
4.2. Datenbankmodellerstellung zur Speicherung von Kunden- und Shopdaten	2 h
4.3. Umsetzung des Datenbankmodells	1 h
5. Implementierung serverseitiger php-Skripte	6 h
6. Implementierung der Javascript UI-Interaktionen	18 h
6. Tests der App-Funktionalitäten	4 h
6.1. Anlegen von Kunden- und Shopdaten im epages Developershop	2 h
6.1. Durchführung der Test	2 h
Erstellen der Dokumentation	6 h
1. Erstellen der Projektdokumentation	4 h
2. Erstellen der Entwicklerdokumentation	2 h
Gesamt	70 h

A.2 Lastenheft (Auszug)

Es folgt ein Auszug aus dem Lastenheft mit Fokus auf die Anforderungen:

Die Anwendung muss folgende Anforderungen erfüllen:

1. Verarbeitung der Moduldaten
 - 1.1. Die Anwendung muss die von Subversion und einem externen Programm bereitgestellten Informationen (z.B. Source-Benutzer, -Datum, Hash) verarbeiten.
 - 1.2. Auslesen der Beschreibung und der Stichwörter aus dem Sourcecode.
2. Darstellung der Daten

- 2.1. Die Anwendung muss eine Liste aller Module erzeugen inkl. Source-Benutzer und -Datum, letztem Commit-Benutzer und -Datum für alle drei Umgebungen.
- 2.2. Verknüpfen der Module mit externen Tools wie z.B. Wiki-Einträgen zu den Modulen oder dem Sourcecode in Subversion.
- 2.3. Die Sourcen der Umgebungen müssen verglichen und eine schnelle Übersicht zur Einhaltung des allgemeinen Entwicklungsprozesses gegeben werden.
- 2.4. Dieser Vergleich muss auf die von einem bestimmten Benutzer bearbeiteten Module eingeschränkt werden können.
- 2.5. Die Anwendung muss in dieser Liste auch Module anzeigen, die nach einer Bearbeitung durch den gesuchten Benutzer durch jemand anderen bearbeitet wurden.
- 2.6. Abweichungen sollen kenntlich gemacht werden.
- 2.7. Anzeigen einer Übersichtsseite für ein Modul mit allen relevanten Informationen zu diesem.
3. Sonstige Anforderungen
 - 3.1. Die Anwendung muss ohne das Installieren einer zusätzlichen Software über einen Webbrowser im Intranet erreichbar sein.
 - 3.2. Die Daten der Anwendung müssen jede Nacht bzw. nach jedem **SVN!**-Commit automatisch aktualisiert werden.
 - 3.3. Es muss ermittelt werden, ob Änderungen auf der Produktionsumgebung vorgenommen wurden, die nicht von einer anderen Umgebung kopiert wurden. Diese Modulliste soll als Mahnung per E-Mail an alle Entwickler geschickt werden (Peer Pressure).
 - 3.4. Die Anwendung soll jederzeit erreichbar sein.
 - 3.5. Da sich die Entwickler auf die Anwendung verlassen, muss diese korrekte Daten liefern und darf keinen Interpretationsspielraum lassen.
 - 3.6. Die Anwendung muss so flexibel sein, dass sie bei Änderungen im Entwicklungsprozess einfach angepasst werden kann.

A.3 Use Case-Diagramm

Use Case-Diagramme und weitere UML-Diagramme kann man auch direkt mit \LaTeX zeichnen, siehe z. B. <http://metauml.sourceforge.net/old/usecase-diagram.html>.

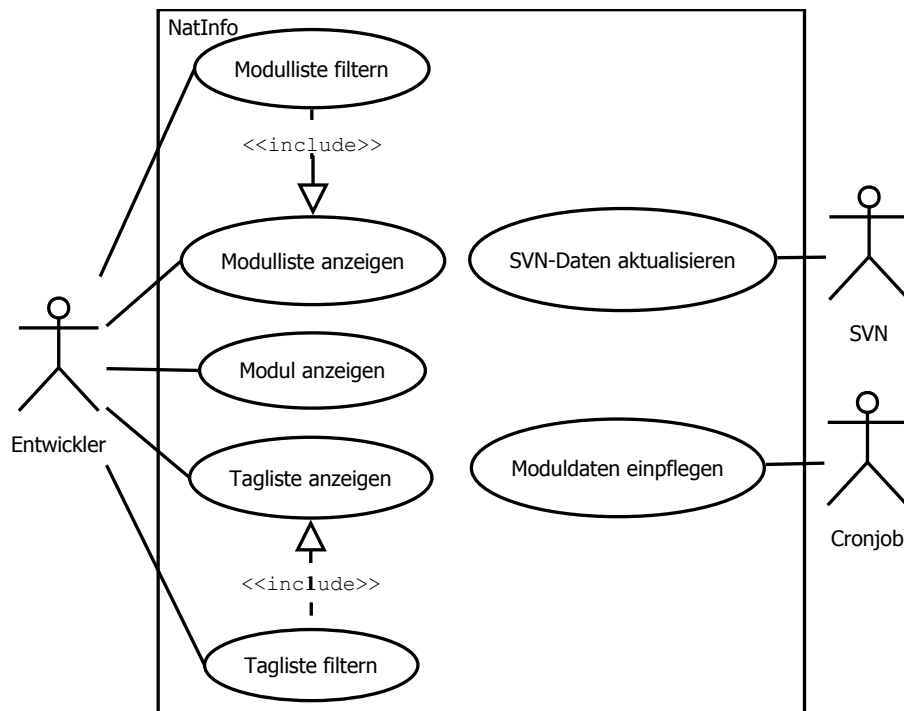


Abbildung 8: Use Case-Diagramm

A.4 Pflichtenheft (Auszug)

Zielbestimmung

1. Musskriterien

1.1. Modul-Liste: Zeigt eine filterbare Liste der Module mit den dazugehörigen Kerninformationen sowie Symbolen zur Einhaltung des Entwicklungsprozesses an

- In der Liste wird der Name, die Bibliothek und Daten zum Source und Kompilat eines Moduls angezeigt.
- Ebenfalls wird der Status des Moduls hinsichtlich Source und Kompilat angezeigt. Dazu gibt es unterschiedliche Status-Zeichen, welche symbolisieren in wie weit der Entwicklungsprozess eingehalten wurde bzw. welche Schritte als nächstes getan werden müssen. So gibt es z. B. Zeichen für das Einhalten oder Verletzen des Prozesses oder den Hinweis auf den nächsten zu tätigenden Schritt.
- Weiterhin werden die Benutzer und Zeitpunkte der aktuellen Version der Sourcen und Kompilate angezeigt. Dazu kann vorher ausgewählt werden, von welcher Umgebung diese Daten gelesen werden sollen.

- Es kann eine Filterung nach allen angezeigten Daten vorgenommen werden. Die Daten zu den Sourcen sind historisiert. Durch die Filterung ist es möglich, auch Module zu finden, die in der Zwischenzeit schon von einem anderen Benutzer editiert wurden.
- 1.2. Tag-Liste: Bietet die Möglichkeit die Module anhand von Tags zu filtern.
- Es sollen die Tags angezeigt werden, nach denen bereits gefiltert wird und die, die noch der Filterung hinzugefügt werden könnten, ohne dass die Ergebnisliste leer wird.
 - Zusätzlich sollen die Module angezeigt werden, die den Filterkriterien entsprechen. Sollten die Filterkriterien leer sein, werden nur die Module angezeigt, welche mit einem Tag versehen sind.
- 1.3. Import der Moduldaten aus einer bereitgestellten [CSV](#)-Datei
- Es wird täglich eine Datei mit den Daten der aktuellen Module erstellt. Diese Datei wird (durch einen Cronjob) automatisch nachts importiert.
 - Dabei wird für jedes importierte Modul ein Zeitstempel aktualisiert, damit festgestellt werden kann, wenn ein Modul gelöscht wurde.
 - Die Datei enthält die Namen der Umgebung, der Bibliothek und des Moduls, den Programmtyp, den Benutzer und Zeitpunkt des Sourcecodes sowie des Kompilats und den Hash des Sourcecodes.
 - Sollte sich ein Modul verändert haben, werden die entsprechenden Daten in der Datenbank aktualisiert. Die Veränderungen am Source werden dabei aber nicht ersetzt, sondern historisiert.
- 1.4. Import der Informationen aus **SVN!** (**SVN!**). Durch einen „post-commit-hook“ wird nach jedem Einchecken eines Moduls ein [PHP](#)-Script auf der Konsole aufgerufen, welches die Informationen, die vom **SVN!**-Kommandozeilentool geliefert werden, an [NATINFO](#) übergibt.
- 1.5. Parsen der Sourcen
- Die Sourcen der Entwicklungsumgebung werden nach Tags, Links zu Artikeln im Wiki und Programmbeschreibungen durchsucht.
 - Diese Daten werden dann entsprechend angelegt, aktualisiert oder nicht mehr gesetzte Tags/Wikiartikel entfernt.
- 1.6. Sonstiges
- Das Programm läuft als Webanwendung im Intranet.
 - Die Anwendung soll möglichst leicht erweiterbar sein und auch von anderen Entwicklungsprozessen ausgehen können.
 - Eine Konfiguration soll möglichst in zentralen Konfigurationsdateien erfolgen.

Produkteinsatz

1. Anwendungsbereiche

Die Webanwendung dient als Anlaufstelle für die Entwicklung. Dort sind alle Informationen

A Anhang

für die Module an einer Stelle gesammelt. Vorher getrennte Anwendungen werden ersetzt bzw. verlinkt.

2. Zielgruppen

wird lediglich von den in der EDV-Abteilung genutzt.

3. Betriebsbedingungen

Die nötigen Betriebsbedingungen, also der Webserver, die Datenbank, die Versionsverwaltung, das Wiki und der nächtliche Export sind bereits vorhanden und konfiguriert. Durch einen täglichen Cronjob werden entsprechende Daten aktualisiert, die Webanwendung ist jederzeit aus dem Intranet heraus erreichbar.

A.5 Datenbankmodell

ER-Modelle kann man auch direkt mit \LaTeX zeichnen, siehe z. B. <http://www.texample.net/tikz/examples/entity-relationship-diagram/>.

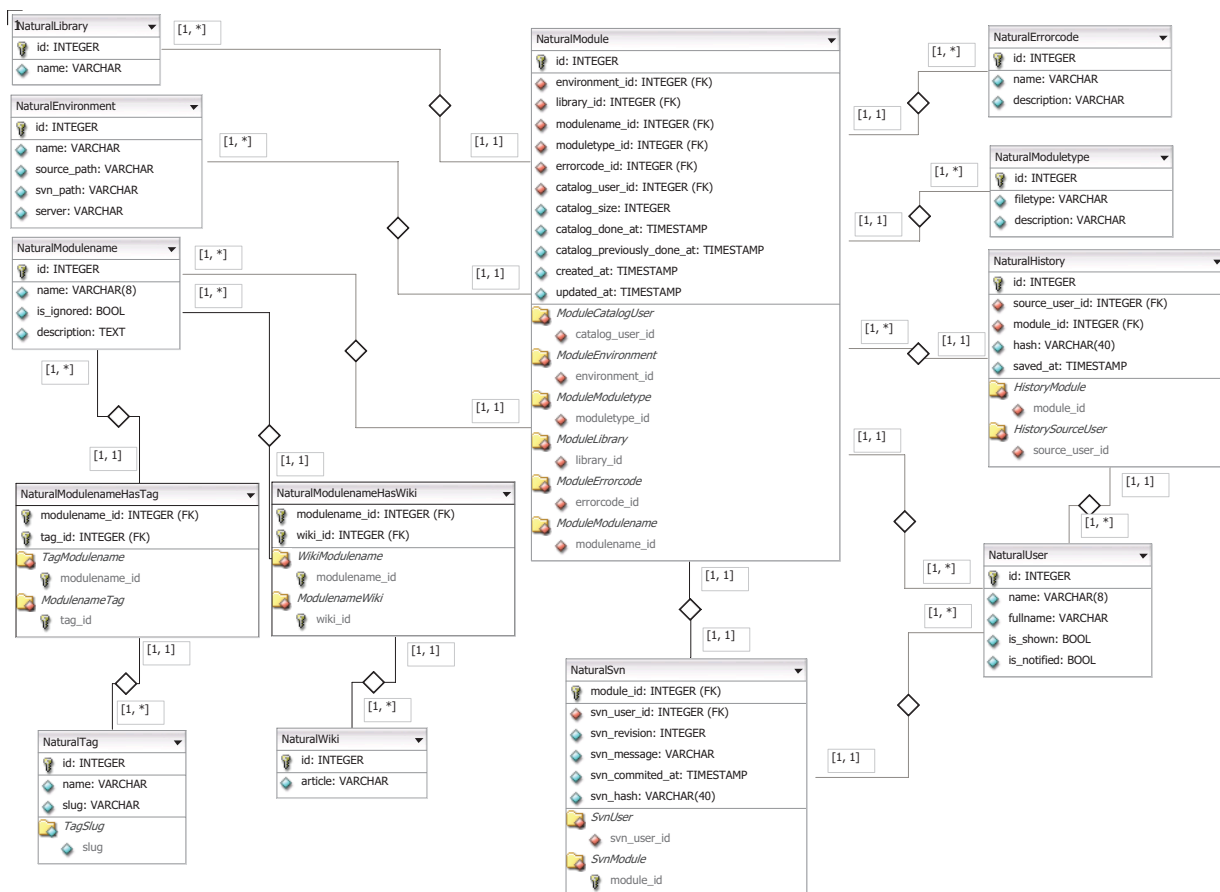


Abbildung 9: Datenbankmodell

A.6 Oberflächenentwürfe

Filter:

Source- and Catalog-Information from Environment:

Library:

Filter

Source:

Date:

Username:

Catalog:

Date:

Username:

Module	Library	Sourcen	Catalog	Source-User	Source-Date	Catalog-User	Catalog-Date
DGTEST	UTILITY	+	+	Grashorn	01.04.2010	Grashorn	02.04.2010
EINTEST	SYSTEM	-	+	Mustermann	01.04.2010	Grashorn	02.04.2010
NOCHEINS	UTILITY	o	-	Grashorn	05.04.2010	Grashorn	05.04.2010
MANUEL	SYSTEM	o	+	Grashorn	01.03.2010	Grashorn	01.03.2010
RESET	CON	-	+	Mustermann	02.03.2010	Mustermann	02.03.2010
TESTER	SYSTEM	+	-	Grashorn	01.02.2010	Grashorn	01.02.2010
...

Abbildung 10: Liste der Module mit Filtermöglichkeiten

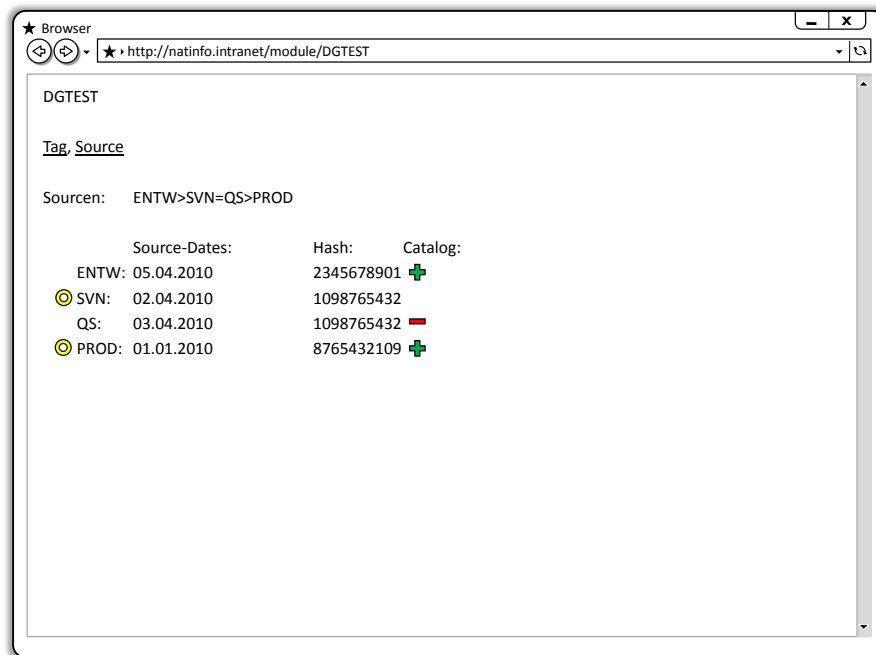


Abbildung 11: Anzeige der Übersichtsseite einzelner Module

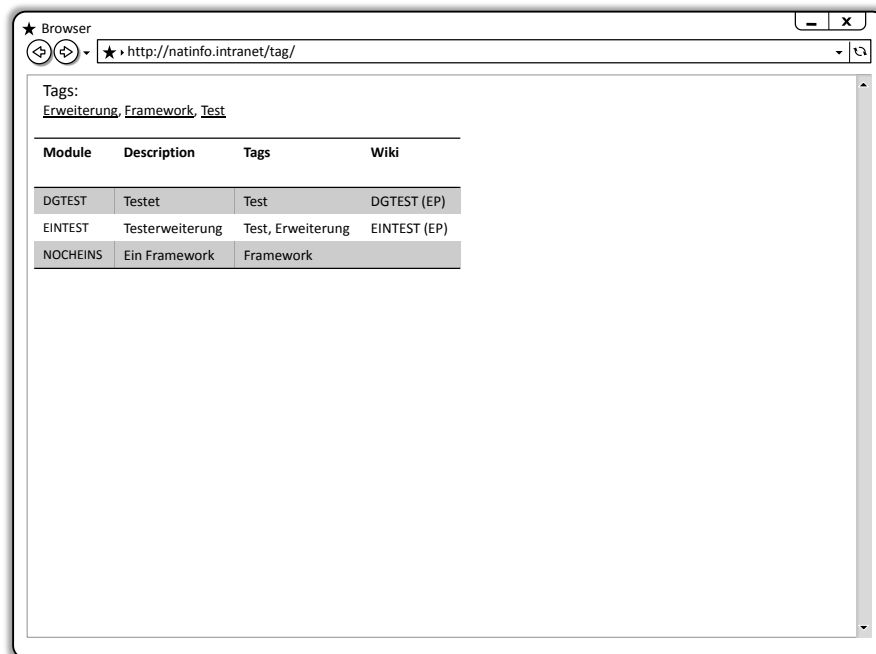
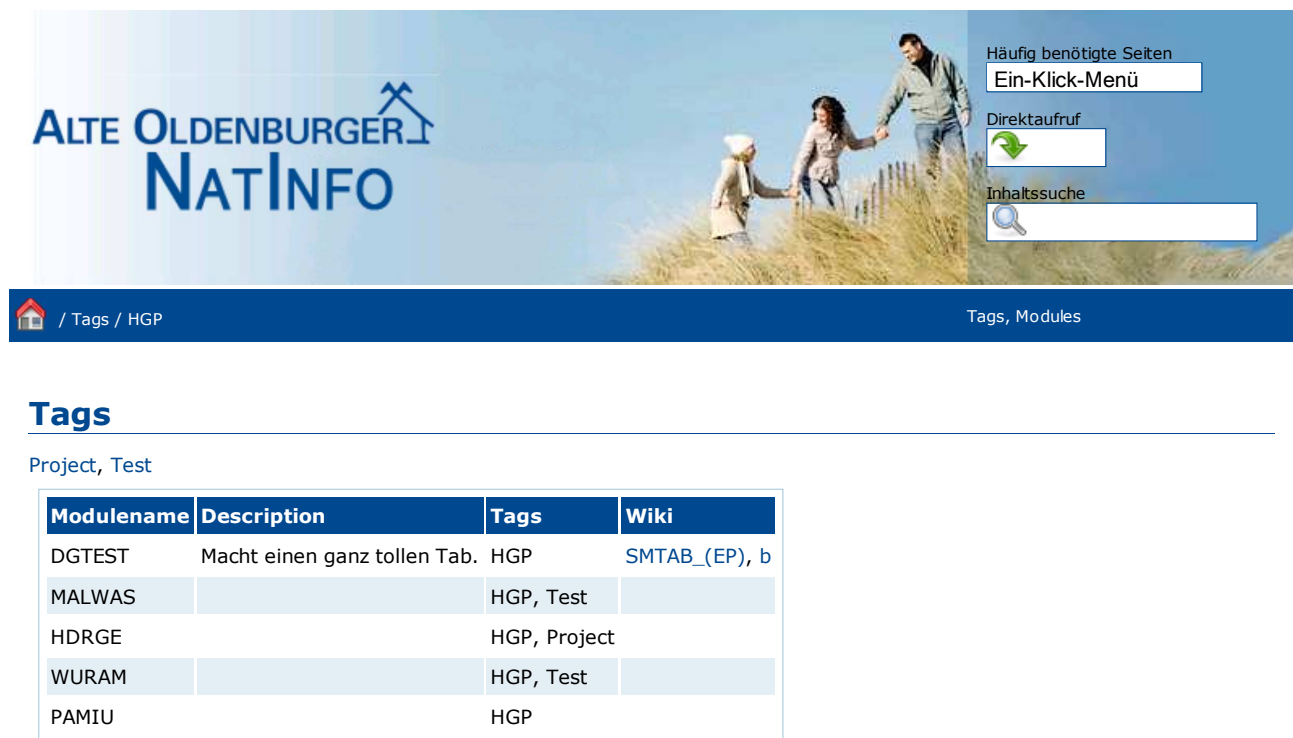


Abbildung 12: Anzeige und Filterung der Module nach Tags

A.7 Screenshots der Anwendung



ALTE OLDENBURGER NATINFO

Häufig benötigte Seiten
Ein-Klick-Menü
Direktaufruf
Inhaltssuche

/ Tags / HGP Tags, Modules

Tags

Project, Test

Modulename	Description	Tags	Wiki
DGTEST	Macht einen ganz tollen Tab.	HGP	SMTAB_(EP), b
MALWAS		HGP, Test	
HDRGE		HGP, Project	
WURAM		HGP, Test	
PAMIU		HGP	

Abbildung 13: Anzeige und Filterung der Module nach Tags



Modules

Environment	ENTW
Library	Select
Catalog user	Select
Catalog date	Select
Source user	Select
Source date	Select
Reset Filter	

Name	Library	Source	Catalog	Source-User	Source-Date	Catalog-User	Catalog-Date
SMTAB	UTILITY			MACKE	01.04.2010 13:00	MACKE	01.04.2010 13:00
DGTAB	CON			GRASHORN	01.04.2010 13:00	GRASHORN	01.04.2010 13:00
DGTEST	SUP			GRASHORN	05.04.2010 13:00	GRASHORN	05.04.2010 13:00
OHNETAG	CON			GRASHORN	05.04.2010 13:00	GRASHORN	01.04.2010 15:12
OHNEWIKI	CON			GRASHORN	05.04.2010 13:00	MACKE	01.04.2010 15:12

Abbildung 14: Liste der Module mit Filtermöglichkeiten

A.8 Entwicklerdokumentation

lib-model

[class tree: lib-model] [index: lib-model] [all elements]

Packages:
lib-model

Files:
Naturalmodulename.php

Classes:
Naturalmodulename

Class: Naturalmodulename

Source Location: /Naturalmodulename.php

Class Overview

```

BaseNaturalmodulename
|
--Naturalmodulename

```

Subclass for representing a row from the 'NaturalModulename' table.

Methods

- [__construct](#)
- [getNaturalTags](#)
- [getNaturalWikis](#)
- [loadNaturalModuleInformation](#)
- [__toString](#)

Class Details

[line 10]
Subclass for representing a row from the 'NaturalModulename' table.

Adds some business logic to the base.

[\[Top \]](#)

Class Methods

constructor [__construct](#) [line 56]

```
Naturalmodulename __construct( )
```

Initializes internal state of Naturalmodulename object.

Tags:

see: parent::__construct()
access: public

[\[Top \]](#)

method [getNaturalTags](#) [line 68]

```
array getNaturalTags( )
```

Returns an Array of NaturalTags connected with this Modulename.

Tags:

return: Array of NaturalTags
access: public

[\[Top \]](#)

method getNaturalWikis [line 83]

```
array getNaturalWikis( )
```

Returns an Array of NaturalWikis connected with this Modulename.

Tags:

return: Array of NaturalWikis
access: public

[\[Top \]](#)

method loadNaturalModuleInformation [line 17]

```
ComparedNaturalModuleInformation  
loadNaturalModuleInformation( )
```

Gets the ComparedNaturalModuleInformation for this NaturalModulename.

Tags:

access: public

[\[Top \]](#)

method __toString [line 47]

```
string __toString( )
```

Returns the name of this NaturalModulename.

Tags:

access: public

[\[Top \]](#)

A.9 Testfall und sein Aufruf auf der Konsole

```

1 <?php
2 include(dirname(__FILE__).'/../bootstrap/Propel.php');
3
4 $t = new lime_test(13);
5
6 $t->comment('Empty Information');
7 $emptyComparedInformation = new ComparedNaturalModuleInformation(array());
8 $t->is($emptyComparedInformation->getCatalogSign(), ComparedNaturalModuleInformation::EMPTY_SIGN, '
    Has no catalog sign');
9 $t->is($emptyComparedInformation->getSourceSign(), ComparedNaturalModuleInformation::SIGN_CREATE, '
    Source has to be created');
10
11 $t->comment('Perfect Module');
12 $criteria = new Criteria();
13 $criteria->add(NaturalmodulePeer::NAME, 'SMTAB');
14 $moduleName = NaturalmodulePeer::doSelectOne($criteria);
15 $t->is($moduleName->getName(), 'SMTAB', 'Right module name selected');
16 $comparedInformation = $moduleName->loadNaturalModuleInformation();
17 $t->is($comparedInformation->getSourceSign(), ComparedNaturalModuleInformation::SIGN_OK, 'Source sign
    shines global');
18 $t->is($comparedInformation->getCatalogSign(), ComparedNaturalModuleInformation::SIGN_OK, 'Catalog sign
    shines global');
19 $infos = $comparedInformation->getNaturalModuleInformations();
20 foreach($infos as $info)
21 {
22     $env = $info->getEnvironmentName();
23     $t->is($info->getSourceSign(), ComparedNaturalModuleInformation::SIGN_OK, 'Source sign shines at ' . $env);
24     if ($env != 'SVNENTW')
25     {
26         $t->is($info->getCatalogSign(), ComparedNaturalModuleInformation::SIGN_OK, 'Catalog sign shines at ' .
            $info->getEnvironmentName());
27     }
28     else
29     {
30         $t->is($info->getCatalogSign(), ComparedNaturalModuleInformation::EMPTY_SIGN, 'Catalog sign is empty
            at ' . $info->getEnvironmentName());
31     }
32 }
33 ?>

```



```

ao-suse-ws1.ao-dom.alte-oldenburger.de - PuTTY
ao-suse-ws1:/srv/www/symfony/natural # ./symfony test:unit ComparedNaturalModuleInformation
1..13
# Empty Information
ok 1 - Has no catalog sign
ok 2 - Source has to be created
# Perfect Module
ok 3 - Right modulename selected
ok 4 - Source sign shines global
ok 5 - Catalog sign shines global
ok 6 - Source sign shines at ENTW
ok 7 - Catalog sign shines at ENTW
ok 8 - Source sign shines at QS
ok 9 - Catalog sign shines at QS
ok 10 - Source sign shines at PROD
ok 11 - Catalog sign shines at PROD
ok 12 - Source sign shines at SVNENTW
ok 13 - Catalog sign is empty at SVNENTW
# Looks like everything went fine.
ao-suse-ws1:/srv/www/symfony/natural #

```

Abbildung 15: Aufruf des Testfalls auf der Konsole

A.10 Klasse: ComparedNaturalModuleInformation

Kommentare und simple Getter/Setter werden nicht angezeigt.

```

1 <?php
2 class ComparedNaturalModuleInformation
3 {
4     const EMPTY_SIGN = 0;
5     const SIGN_OK = 1;
6     const SIGN_NEXT_STEP = 2;
7     const SIGN_CREATE = 3;
8     const SIGN_CREATE_AND_NEXT_STEP = 4;
9     const SIGN_ERROR = 5;
10
11     private $naturalModuleInformations = array();
12
13     public static function environments()
14     {
15         return array("ENTW", "SVNENTW", "QS", "PROD");
16     }
17
18     public static function signOrder()
19     {
20         return array(self::SIGN_ERROR, self::SIGN_NEXT_STEP, self::SIGN_CREATE_AND_NEXT_STEP, self::SIGN_CREATE, self::SIGN_OK);
21     }
22
23     public function __construct(array $naturalInformations)
24     {
25         $this->allocateModulesToEnvironments($naturalInformations);

```


A Anhang

```

26     $this->allocateEmptyModulesToMissingEnvironments();
27     $this->determineSourceSignsForAllEnvironments();
28 }
29
30 private function allocateModulesToEnvironments(array $naturalInformations)
31 {
32     foreach ($naturalInformations as $naturalInformation)
33     {
34         $env = $naturalInformation->getEnvironmentName();
35         if (in_array($env, self::environments()))
36         {
37             $this->naturalModuleInformations[array_search($env, self::environments())] = $naturalInformation;
38         }
39     }
40 }
41
42 private function allocateEmptyModulesToMissingEnvironments()
43 {
44     if (array_key_exists(0, $this->naturalModuleInformations))
45     {
46         $this->naturalModuleInformations[0]->setSourceSign(self::SIGN_OK);
47     }
48
49     for ($i = 0; $i < count(self::environments()); $i++)
50     {
51         if (!array_key_exists($i, $this->naturalModuleInformations))
52         {
53             $environments = self::environments();
54             $this->naturalModuleInformations[$i] = new EmptyNaturalModuleInformation($environments[$i]);
55             $this->naturalModuleInformations[$i]->setSourceSign(self::SIGN_CREATE);
56         }
57     }
58 }
59
60 public function determineSourceSignsForAllEnvironments()
61 {
62     for ($i = 1; $i < count(self::environments()); $i++)
63     {
64         $currentInformation = $this->naturalModuleInformations[$i];
65         $previousInformation = $this->naturalModuleInformations[$i - 1];
66         if ($currentInformation->getSourceSign() <> self::SIGN_CREATE)
67         {
68             if ($previousInformation->getSourceSign() <> self::SIGN_CREATE)
69             {
70                 if ($currentInformation->getHash() <> $previousInformation->getHash())
71                 {
72                     if ($currentInformation->getSourceDate('YmdHis') > $previousInformation->getSourceDate('YmdHis'))
73                     {
74                         $currentInformation->setSourceSign(self::SIGN_ERROR);
75                     }
76                 }
77             }
78         }
79     }
80 }

```

A Anhang

```

76         else
77         {
78             $currentInformation->setSourceSign(self::SIGN_NEXT_STEP);
79         }
80     }
81     else
82     {
83         $currentInformation->setSourceSign(self::SIGN_OK);
84     }
85 }
86 else
87 {
88     $currentInformation->setSourceSign(self::SIGN_ERROR);
89 }
90 }
91 elseif ($previousInformation->getSourceSign() <> self::SIGN_CREATE && $previousInformation->
    getSourceSign() <> self::SIGN_CREATE_AND_NEXT_STEP)
92 {
93     $currentInformation->setSourceSign(self::SIGN_CREATE_AND_NEXT_STEP);
94 }
95 }
96 }
97
98 private function containsSourceSign($sign)
99 {
100     foreach($this->naturalModuleInformations as $information)
101     {
102         if ($information->getSourceSign() == $sign)
103         {
104             return true;
105         }
106     }
107     return false ;
108 }
109
110 private function containsCatalogSign($sign)
111 {
112     foreach($this->naturalModuleInformations as $information)
113     {
114         if ($information->getCatalogSign() == $sign)
115         {
116             return true;
117         }
118     }
119     return false ;
120 }
121 }
122 ?>

```

A.11 Klassendiagramm

Klassendiagramme und weitere UML-Diagramme kann man auch direkt mit \LaTeX zeichnen, siehe z. B. <http://metauml.sourceforge.net/old/class-diagram.html>.

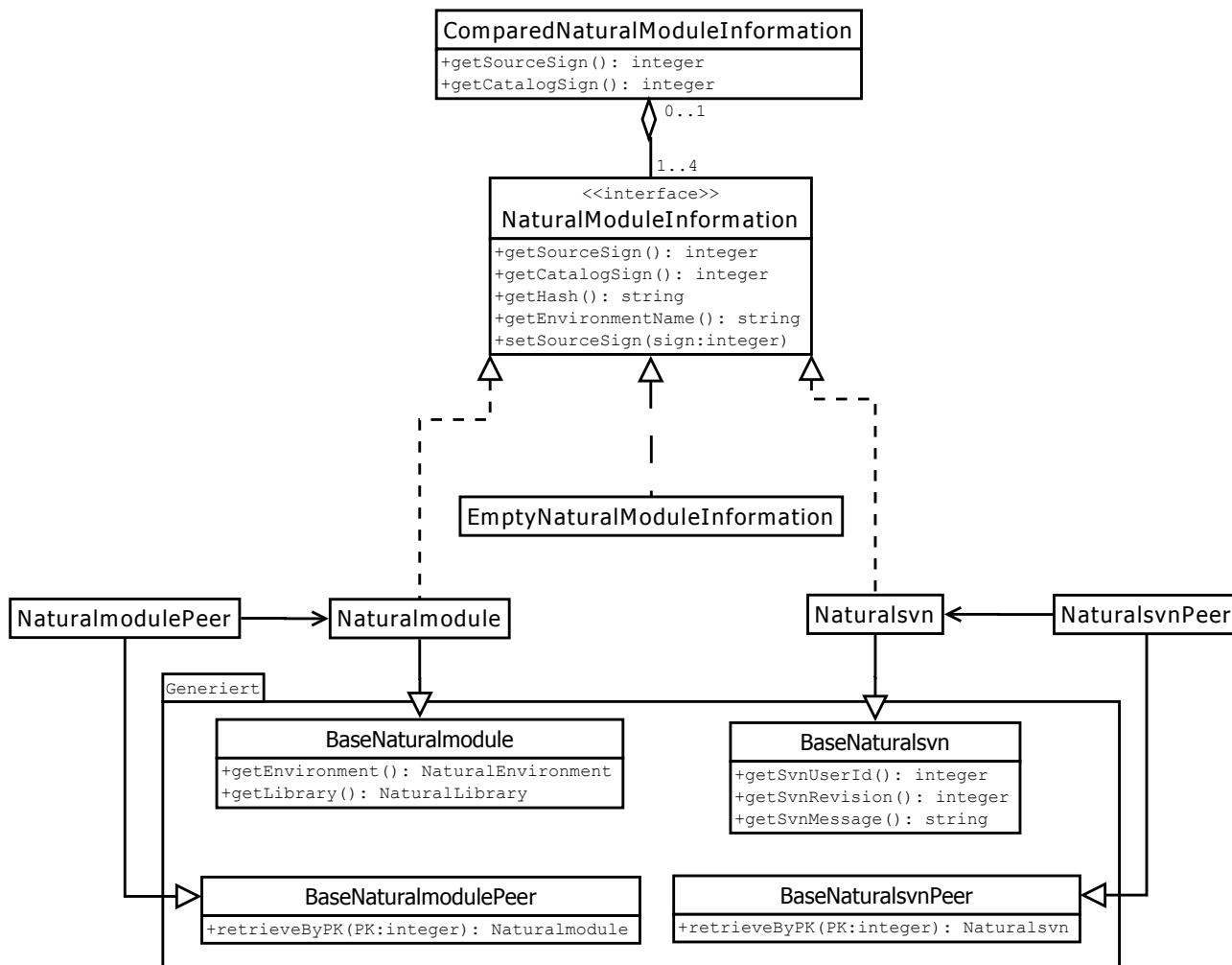







Abbildung 16: Klassendiagramm

A.12 Benutzerdokumentation

Ausschnitt aus der Benutzerdokumentation:

Symbol	Bedeutung global	Bedeutung einzeln
	Alle Module weisen den gleichen Stand auf.	Das Modul ist auf dem gleichen Stand wie das Modul auf der vorherigen Umgebung.
	Es existieren keine Module (fachlich nicht möglich).	Weder auf der aktuellen noch auf der vorherigen Umgebung sind Module angelegt. Es kann also auch nichts übertragen werden.
	Ein Modul muss durch das Übertragen von der vorherigen Umgebung erstellt werden.	Das Modul der vorherigen Umgebung kann übertragen werden, auf dieser Umgebung ist noch kein Modul vorhanden.
	Auf einer vorherigen Umgebung gibt es ein Modul, welches übertragen werden kann, um das nächste zu aktualisieren.	Das Modul der vorherigen Umgebung kann übertragen werden um dieses zu aktualisieren.
	Ein Modul auf einer Umgebung wurde entgegen des Entwicklungsprozesses gespeichert.	Das aktuelle Modul ist neuer als das Modul auf der vorherigen Umgebung oder die vorherige Umgebung wurde übersprungen.