

issue manager

Mängelverwaltung für die W&W Architekten GmbH

Projektdokumentation¹ im Rahmen des Moduls Softwarekomponenten – INM21

Dozenten: Jordan Šućur und Erwin Mathis

Frühling 2015, Hochschule Luzern

Projektgruppe C: Patrick Elsener (PEL), Sandro Klarer (SKL), Reno Meyer (RME), Aathavan Theivendram (ATH), Janick von Rotz (JVR), Erwin Willi (EWI)

¹ We acknowledge that this document uses material from the arc 42 architecture template, <http://www.arc42.de>. Created by Dr. Peter Hruschka & Dr. Gernot Starke. For additional contributors see [arc42.de/about/contributors.html](http://www.arc42.de/about/contributors.html)

Inhalt

1	Einführung und Ziele	6
1.1	Aufgabenstellung	6
1.2	Qualitätsziele	6
1.3	Stakeholder	7
2	Randbedingungen	9
2.1	Technische Randbedingungen	9
2.2	Organisatorische Randbedingungen	9
2.3	Konventionen	10
3	Kontextabgrenzung	13
3.1	Fachlicher Kontext	13
3.2	Technischer- oder Verteilungskontext	14
4	Lösungsstrategie	15
5	Bausteinsicht	16
5.1	Ebene 1	16
5.2	Ebene 2	18
5.3	Ebene 3 - GUI	25
6	Laufzeitsicht	30
6.1	Laufzeitszenario – Mangel erstellen	30
7	Verteilungssicht	31
7.1	Infrastruktur	31
8	Konzepte	33
8.1	Persistenz	33
8.2	Benutzungsoberfläche	33
8.3	Ergonomie	33
8.4	Ablaufsteuerung	33
8.5	Transaktionsbehandlung	33
8.6	Sessionbehandlung	33
8.7	Sicherheit	33
8.8	Plausibilisierung und Validierung	34

8.9	Ausnahme-/Fehlerbehandlung	34
8.10	Logging, Protokollierung, Tracing	34
8.11	Geschäftsregeln	34
8.12	Konfigurierbarkeit	34
8.13	Codegenerierung	34
8.14	Buildmanagement	34
9	Entwurfsentscheidungen	35
9.1	Entwurfsentscheidung 1: GUI	35
9.2	Entwurfsentscheidung 2: Datenbank	35
9.3	Entwurfsentscheidung 3: Zugriff des Clients	36
9.4	Entwurfsentscheidung 4: Clientabgrenzung	36
10	Qualitätsszenarien	38
10.1	Qualitätsbaum	38
10.2	Bewertungsszenarien	39
11	Risiken	40
12	Glossar	41
13	Weitere Arc42 Dokumentationen	42
14	Requirements Dokumentation	43
14.1	Funktionale Requirements	46
14.2	Nicht-funktionale Requirements	54
15	UseCase Dokumentation	56
16	Klassendiagramme	87
17	Deployment-Infos	98
17.1	Systemanforderungen	98
17.2	Ausführung	98
17.3	Konfiguration	98
17.4	Entwicklungsumgebung	98
18	TDD und Junit	104
18.1	Test-Driven-Development (TDD)	104
18.2	JUnit	104
18.3	Server / Client (End-to-End testing)	104
18.4	Testfälle	106

18.5	Erfahrungen mit TDD	108
18.6	Erfahrungen mit Unit-Testing	108
18.7	Probleme und Lösungen	108
19	Funktionale Tests	109
19.1	Testprotokoll	109
19.2	Abnahmeprotokoll	109
19.3	Testfälle	110
19.4	Zielüberprüfung	112
20	DB-Dokumentation	114
21	Beiträge pro Projektmitglied	115
21.1	Elsener Patrick	115
21.2	Klarer Sandro	116
21.3	Meyer Reno	120
21.4	Theivendram Aathavan	121
21.5	von Rotz Janik	123
21.6	Willi Erwin	128
22	Source-Code	131
23	Designkonzept	132
23.1	Farbkonzept	132
23.2	Logo	132
23.3	GUI01 – Login	132
23.4	GUI02 – Mängelübersicht	133
23.5	GUI03 – Mangel	134
23.6	GUI04 – Projekt	135
23.7	GUI05 – Projekt (KA)	136
23.8	GUI06 – Projektleitung	137
23.9	GUI07 – Zugewiesene Subunternehmen	138
23.10	GUI08 – Projekteübersicht	139
23.11	GUI09 – Subunternehmenübersicht	140
23.12	GUI10 – Subunternehmen	141
23.13	GUI11 – Personenübersicht	142
23.14	GUI12 – Bauherr	143
23.15	GUI13 – Kontaktadmin	144

23.16	GUI14 – Passwort	145
23.17	Verworfenene Designs	146
24	Logging	148
25	Beispielapplikation	149

1 Einführung und Ziele

In diesem Abschnitt wird in die Aufgabenstellung und die Ziele, welche mit dem Issue manager erreicht werden sollen, eingeführt.

1.1 Aufgabenstellung

Was ist der issue manager?

- issue manager ist eine Applikation für die Verwaltung von Bauprojekten der W&W Architektur GmbH.
- Sie dient der Übersicht und der Kommunikation zwischen dem Generalunternehmen, den Bauleitern und den Subunternehmen.
- Alle Bauprojekte, Bauleiter, Subunternehmen und Bauherren der W&W Architektur werden in der Applikation erfasst und verwaltet.
- Alle durch den Bauleiter entdeckten Mängel können vor Ort erfasst und zugewiesen werden
- Subunternehmen können Nachbesserungsarbeiten bestätigen und gegebenenfalls kommentieren.

1.2 Qualitätsziele

In der folgenden Tabelle sind die Qualitätsziele, geordnet nach der Wichtigkeit, vom issue manager zu entnehmen.

Nr./ Priorität	Qualitätsmerkmal	Ziel
1.	Änderbarkeit/ Erweiterbarkeit	Erweiterungen von zusätzlichen Felder oder Views können, aufgrund des synchronen Aufbaues, jederzeit implementiert und in die Applikation integriert werden.
2.	Interoperabilität	Auf eine spezifisch DBMS (Datenbank Management System) Unterstützung wird verzichtet. Das DBMS muss jederzeit ausgetauscht werden können und die Integration von Hersteller unabhängigen Produkten erlauben.
3.	Attraktivität	Die Client- und Server-Komponenten sind unabhängig voneinander. So können sie in heterogenen Systemumgebungen installiert und konfiguriert werden.
4.	Transparenz	Zur Nachvollziehbarkeit (Auditing) und Fehlerbehandlung (Exception Handling) werden alle Events und Benutzer Aktionen geloggt.
5	Analysierbarkeit	Der issue Manager basiert auf einer 3 tier Architektur und ist logisch aufgebaut. Methoden, Klassen, Interfaces und Exceptions sind dokumentiert und gut nachvollziehbar

In den Qualitätsszenarien in Abschnitt 10 werden die Ziele bezüglich der Umsetzung beurteilt.

1.3 Stakeholder

Die unten folgende Auflistung bietet einen Überblick über die verschiedenen Stakeholder und ihre entsprechende Anforderungen an die Applikation.

Name/ Rolle	Ziel/ Berührungspunkt	Notwendige Beteiligung
Sachbearbeiter (SB)	<ul style="list-style-type: none"> - erfassen und verwalten die Stammdaten - besitzen die Admin Rolle um bei Problemen einzugreifen - nutzen die Applikation um sich über den aktuellen Stand der Bauprojekt zu informieren 	<ul style="list-style-type: none"> - Interesse und Wünsche an die Applikation sind durch die Aufgabenstellung abgedeckt. Deshalb ist keine Beteiligung notwendig.
Bauleiter (BL)	<ul style="list-style-type: none"> - erfassen direkt auf der Baustelle die festgestellten Mängel und ordnen sie dem zuständigen Subunternehmen zu. - kommentieren erfasste Mängel um gewisse Sachverhalte zu erklären - geben auf einen allfälligen Kommentar von einem Subunternehmen Antwort. - tragen sich als Verantwortliche für die zugeteilten Bauprojekte ein. 	<ul style="list-style-type: none"> - Interesse und Wünsche an die Applikation sind durch die Aufgabenstellung abgedeckt. Deshalb ist keine Beteiligung notwendig.
Subunternehmen (SU)²	<ul style="list-style-type: none"> - informieren sich über den Stand der erledigten Arbeiten. - exportieren Listen auf welchen die zu erledigen Aufträge notiert sind. - kommentieren zugeordnete Mängel bei der Kenntnisname oder bei Unklarheiten. 	<ul style="list-style-type: none"> - Interesse und Wünsche an die Applikation sind durch die Aufgabenstellung abgedeckt. Deshalb ist keine Beteiligung notwendig.
Bauherr (BH)	<ul style="list-style-type: none"> - informiert sich über den aktuellen Stand seines beauftragten Bauprojekts, indem er mit den Sachbearbeiter Kontakt aufnimmt. 	<ul style="list-style-type: none"> - Interesse und Wünsche an die Applikation sind durch die Aufgabenstellung abgedeckt. Deshalb ist keine Beteiligung notwendig.
Projektleitung	<ul style="list-style-type: none"> - Applikation, welche die Wünsche des Auftraggebers abdeckt und Fristgerecht 	<ul style="list-style-type: none"> - Organisation und Koordination des

² Der Kontaktadmin und die Kontaktperson werden in der Dokumentation als Subunternehmen SU zusammengefasst.

	<ul style="list-style-type: none"> - geliefert werden kann. - Erfahrungen für zukünftige Projekte 	<ul style="list-style-type: none"> - Projektteams - Ansprechperson für den Auftraggeber
Architekt	<ul style="list-style-type: none"> - Solide Architektur, welche eine gewünschte Applikation ermöglicht. - Umsetzung der Qualitätsziele (Kapitel 1.2) um die langfristige Nutzung zu garantieren. - Erfahrungen für zukünftige Projekte 	<ul style="list-style-type: none"> - Erstellt die Architektur und koordiniert die Aufgaben, nach der gewünschten Reihenfolge
Development Team	<ul style="list-style-type: none"> - Neugierig auf konkrete Anwendung und Umsetzung - Möglichst grossen Lernfortschritt erzielen - Erfahrungen für zukünftige Projekte 	<ul style="list-style-type: none"> - Führen die anstehenden Arbeiten aus - Helfen beim Lösen von Problemstellungen - Versuchen die Projektleitung und den Architekt zu entlasten.

2 Randbedingungen

2.1 Technische Randbedingungen

ID	Randbedingungen	Erläuterungen
TRB01	Hardware	Die Applikation muss auf einem beliebigen Tablet, Laptop und Computer in Betrieb genommen werden können. Dies aufgrund der unterschiedlichen Interaktion der Stakeholder mit dem issue-manager.
TRB02	Software	Die Ausstattung der Hardware macht es erforderlich, dass die Lösung auf der Windows Version 7 und höher betreiben werden kann. Die Unterstützung anderer Betriebssysteme ist wünschenswert, aber nicht zwingend erforderlich.
TRB03	Betriebliche Randbedingungen	<p>Client:</p> <p>Für die Graphical User Interfaces (GUI) darf auf den JavaFX Scene Builder 2.0 zurückgegriffen werden.</p> <p>Middleware (Webservice):</p> <p>Die Implementierung der Applikation soll mit Java erfolgen. Dabei ist die Version JDK 1.8 und das Programmierwerkzeug Eclipse zu verwenden.</p> <p>Datenbank:</p> <p>Auf ein spezifisches DBMS wird verzichtet. Die Datenbank soll unabhängig wählbar sein.</p>

2.2 Organisatorische Randbedingungen

ID	Randbedingun- gen	Erläuterungen
OGR01	Team	Reno Meyer führte das Projektteam mit dem Architekt Janik von Rotz. Zum Entwicklungsteam gehörten Aathavan Theivendram, Erwin Willi, Patrick Elsener und Sandro Klarer. Das ganze Team besteht aus Studenten der Wirtschaftsinformatik der Hochschule Luzern.
ORG02	Zeitplan	Mitte Februar 2015 kontaktierte uns der Auftraggeber. Die Applikation muss am 21. Mai 2015 mit der zugehörigen Dokumentation abgegeben werden.
ORG03	Vorgehensmode	Anhand der Aufgabenstellung wurden die Requirements definiert und

	II	Use-Cases erstellt. Während dessen entwickelte der Architekt Janik von Rotz eine Beispiel Applikation. Auf Basis dieser wurde Woche für Woche die anstehenden Aufgaben verteilt und umgesetzt. Um die Arbeitsergebnisse zu dokumentieren wurde arc42 verwendet.
ORG04	Entwicklungswerkzeuge	Die Requirements und die daraus folgenden Use-Cases sind mit Microsoft Visio erstellt worden. Die Architektur reifte von ersten Skizzen von Hand über komplexer Entwürfe mit Microsoft Visio. Der Java-Quellcode wurde mit Hilfe von Eclipse und JavaFx Scene-Builder 2.0 erstellt.
ORG05	Konfigurations- und Versionsverwaltung	Es besteht aktuell nur die erste Version.
ORG06	Testwerkzeuge und -prozesse	Die Zugriffe auf die Datenbank von der Client Seite aus, wurde mit JUnit 4 getestet.
ORG07	Review und Berichtsprozesse	Der GUI Vorschlag wurde, wie vom Auftraggeber gefordert, präsentiert und diskutiert
ORG08	Veröffentlichung als Open Source	Der ganze Quelltext, sowie die Milestones und die Tasks der Applikation sind auf Git-Hub und waffle.io öffentlich zugänglich.

2.3 Konventionen

Konvention	Erläuterungen
Architekturdokumentation	Gemäss dem arc42-Template und den Ergänzungen der Hochschule für Wirtschaft Luzern im Jahre 2015.
Kodier Richtlinien für Java	Java Coding Conventions von Sun/Oracle
Sprache (Deutsch vs. Englisch)	Die Dokumentation werden die Texte, Beschriebe von Modellen und Diagramme auf Deutsch verfasst. Die Java-Doc Kommentare sind in Englischer Sprache. Dies Aufgrund dessen, das der Quellcode auch in Englischer Sprache ist.
Tags	<p>Für jede Klasse und Interface werden folgende Tags verwendet:</p> <ol style="list-style-type: none"> 1. @author (classes and interfaces only, required) 2. @version (classes and interfaces only, required. See footnote 1) 3. @since first Minor release version. <p>Für Methode entsprechend verwenden:</p> <ul style="list-style-type: none"> • @param (methods and constructors only) • @return (methods only)

- @exception (@throws is a synonym added in Javadoc 1.2)
- @see auto generated link

Namenskonventionen

FXML Komponenten:

Name der FXML Datei, die zur Übersicht von Models dienen:

`<Model>.fxml`
`Subunternehmen.fxml`

Name der FXML Datei, die als Bearbeitungsformular von Models dienen:

`<Model>Detail.fxml`
`SubunternehmenDetail.fxml`

Name des FXML Controllers:

`<Name>View.java`
`HomeView.java`

Name der Top FXML Datei.

`Top<Rolle>.fxml`
`TopDefault.fxml`

Name der Top FXML Controller.

`Top<Rolle>View.java`
`TopDefaultView.java`

Name der Left FXML Datei.

`Left<Top-Auswahl>.fxml`
`LeftMangel.fxml`

Name der Left FXML Controller.

`Left<Top-Auswahl>View.java`
`LeftMangelView.java`

Name der Dialog FXML Datei.

`<Typ>Dialog.fxml`
`ErrorDialog.fxml`

Variablenname der FXML Elemente (Buttons, Panes, TableViews, etc.)

`<short code><ElementName>`
`btDelete`

Methodenname der Element-Aktionen (Klick auf Delete-Button)

`<Aktion><ElementName>`
`clickDelete`

JavaFX Scene Builder:

TableView	tv
ListView	lw
TextField	tx
TableColumn	tc
ImageView	iv
PasswordField	pf
Button	bt
Pane	pn
Label	lb
ComboBox	cb
DatePicker	dp
TextArea	ta
TitlePane	tb
CheckBox	ch

Dokumenten Namen

Allgemein:

IM_<Bezeichnung>_<Name>_<Datum>

IM = Issue Manager

Bezeichnung = Sprechender Name für das Dokument

Datum = Format: YYYY-MM-DD

Name = Nachname Vorname

Use Case Visio

IM_UAC<Nummer>_<System>.vsdx

Optionen für <System>

- Webservice
- Client
- Applikation

Task Workflow

<#Waffel-ID><Task Name><

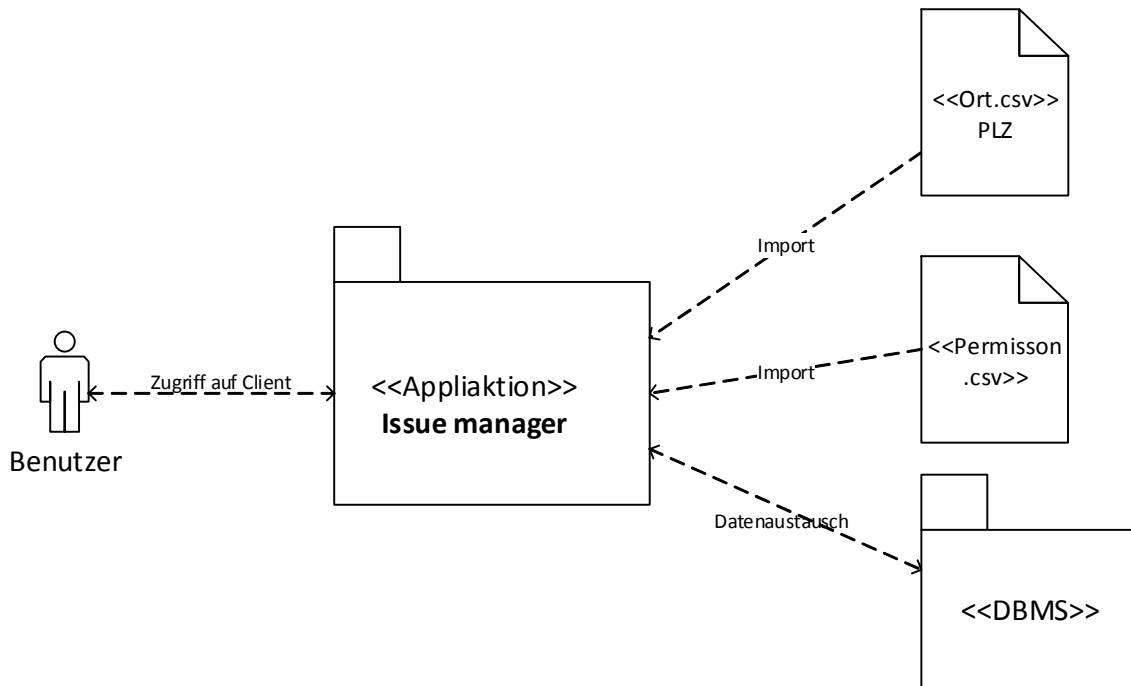
**Semantische
Versionierung**

1. Hauptversionsnummer Major:
Änderungen die zu wahrscheinlichen Inkompatibilitäten führen
2. Nebenversionsnummer Minor:
Änderungen oder Erweiterungen, welche abwärtskompatibel bleiben
3. Patch:
Fehler Korrekturen ohne die Kompatibilität zu beeinträchtigen

3 Kontextabgrenzung

Im folgendem Abschnitt wird das Umfeld vom issue manager dargestellt. Dabei wird erläutert, für welche Benutzer die Applikation gedacht ist und mit welchen Fremdsystemen interagiert wird.

3.1 Fachlicher Kontext

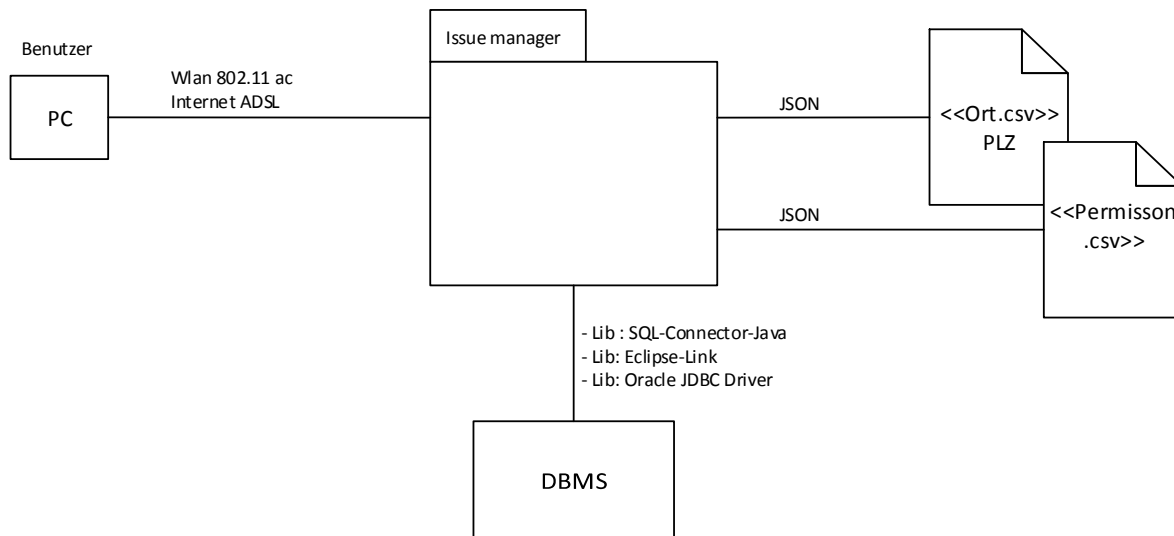


Schnittstelle/ Nachbarsystem	Ausgetauschte Daten (Datenformate, Medien)	Technologie/ Protokoll
Datenaustausch mit dem DBMS	<ul style="list-style-type: none"> - Stammdaten - Mangel Dataen 	<ul style="list-style-type: none"> - Lib : MySQL-Connector-Java - Lib: Eclipse-Link
Postleitzahlen	<ul style="list-style-type: none"> - CSV mit allen Ortschaften 	<ul style="list-style-type: none"> - JavaScript Object Notation (JSON)
Permission	<ul style="list-style-type: none"> - CSV mit vordefinierten Rollen 	<ul style="list-style-type: none"> - JavaScript Object Notation (JSON)

Unter dem Benutzer werden sämtliche Stakeholder zusammengefasst, welche gemäss der Aufgabenstellungen mit der Applikation arbeiten.

Detailliertere Informationen werden im Kapitel 5, Bausteinsicht erläutert.

3.2 Technischer- oder Verteilungskontext

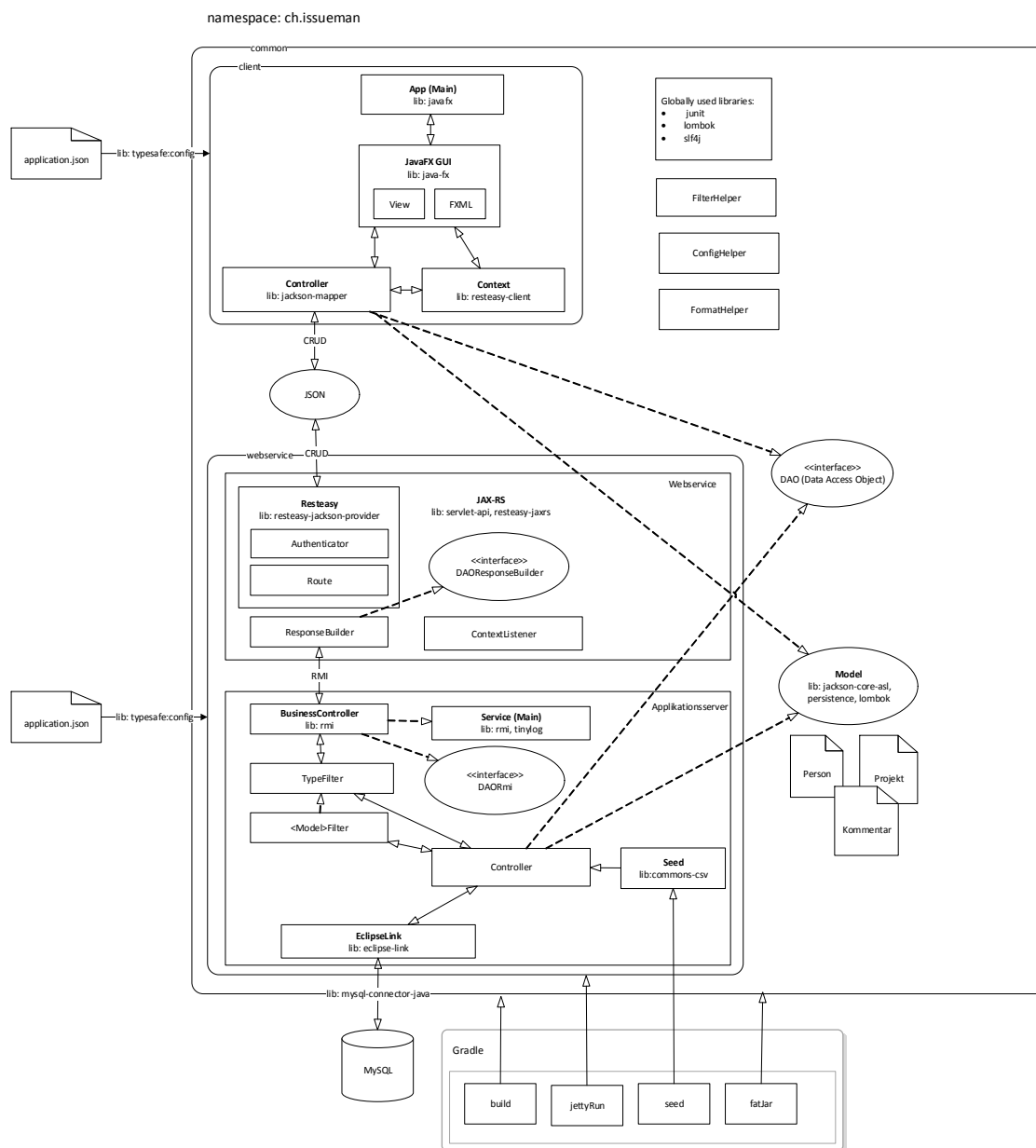


Fachliche Schnittstelle	Technischer Kanal
Zugriff auf Client	<ul style="list-style-type: none"> - WLAN 802.11ac - Internet ADSL
Issue manager/ DMBS	<ul style="list-style-type: none"> - Lib : SQL-Connector-Java - Lib: Eclipse-Link - Lib: Oracle JDBC Driver
Permission.csv	<ul style="list-style-type: none"> - JavaScript Object Notation (JSON)
Ort.csv	<ul style="list-style-type: none"> - JavaScript Object Notation (JSON)

Detailliertere Beschreibungen sind im Kapitel 5 aufgelistet.

4 Lösungsstrategie

Um die Anforderung nach einer verteilten Applikation mit mobiler Zugriffsmöglichkeit gerecht zu werden, haben wir uns für eine 3-Tier-Architektur entschieden. Dabei bildet das „common“ die erste Schicht, wo hauptsächlich die Models hinterlegt werden. Der „client“ ist die grafische Benutzeroberfläche. Alle Benutzer greifen auf das selbe GUI zurück, werden jedoch mit einer Abfrage ihrer Login-Informationen gewissen Einschränkungen unterworfen. Der „webservice“ stellt die Kommunikation zwischen den ersten beiden Schichten, sowie der Datenbank dar. Der Aufbau ist extrem flexibel, womit die Datenbank und der Webserver sehr einfach ausgetauscht werden können. Details werden in den folgenden 3 Kapitel näher erläutert.



5 Bausteinsicht

5.1 Ebene 1

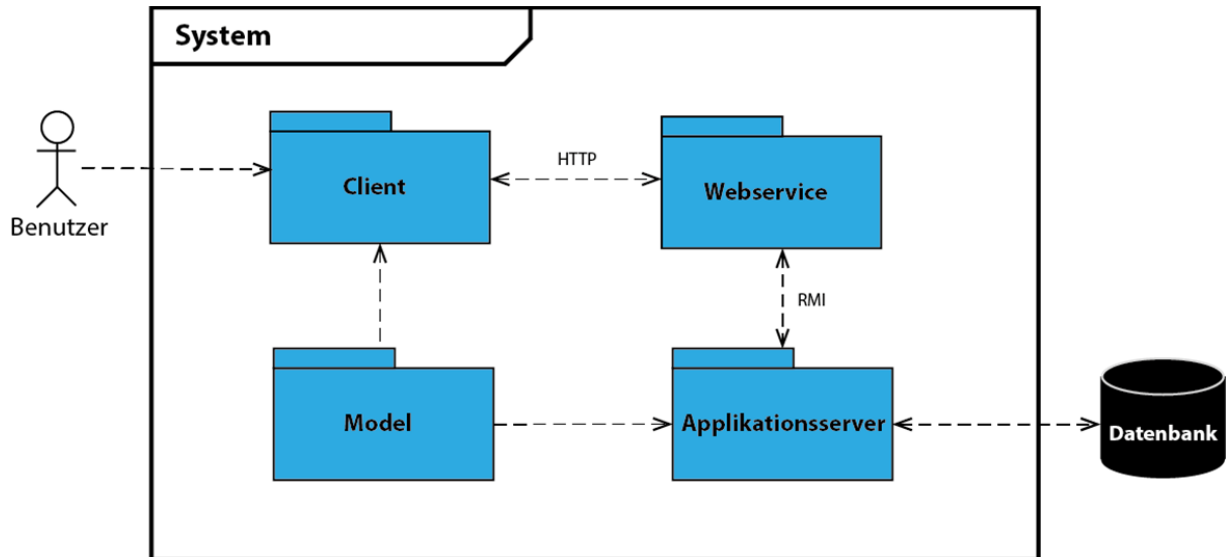


Abbildung 1: Übersicht über das gesamte System

5.1.1 Client

Bezeichnung	Bedeutung
Name	Client
Zweck & Verantwortlichkeit	GUI bereitstellen
Schnittstellen	HTTP
Erfüllte Anforderungen	JavaFX client
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.1.2 Webservice

Bezeichnung	Bedeutung
Name	Webservice
Zweck & Verantwortlichkeit	Restful API
Schnittstellen	HTTP REST
Erfüllte Anforderungen	Webservice
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.1.3 Applikationsserver

Bezeichnung	Bedeutung
Name	Applikationsserver
Zweck & Verantwortlichkeit	Bussiness Logik und Persistierung
Schnittstellen	RMI
Erfüllte Anforderungen	RMI Abstrahierung
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.1.4 Model

Bezeichnung	Bedeutung
Name	Model
Zweck & Verantwortlichkeit	Hier werden die gemeinsamen Datentypen für das ganze System zur Verfügung gestellt.
Schnittstellen	Controller
Erfüllte Anforderungen	Datenmodell
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.common
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2 Ebene 2

5.2.1 Client

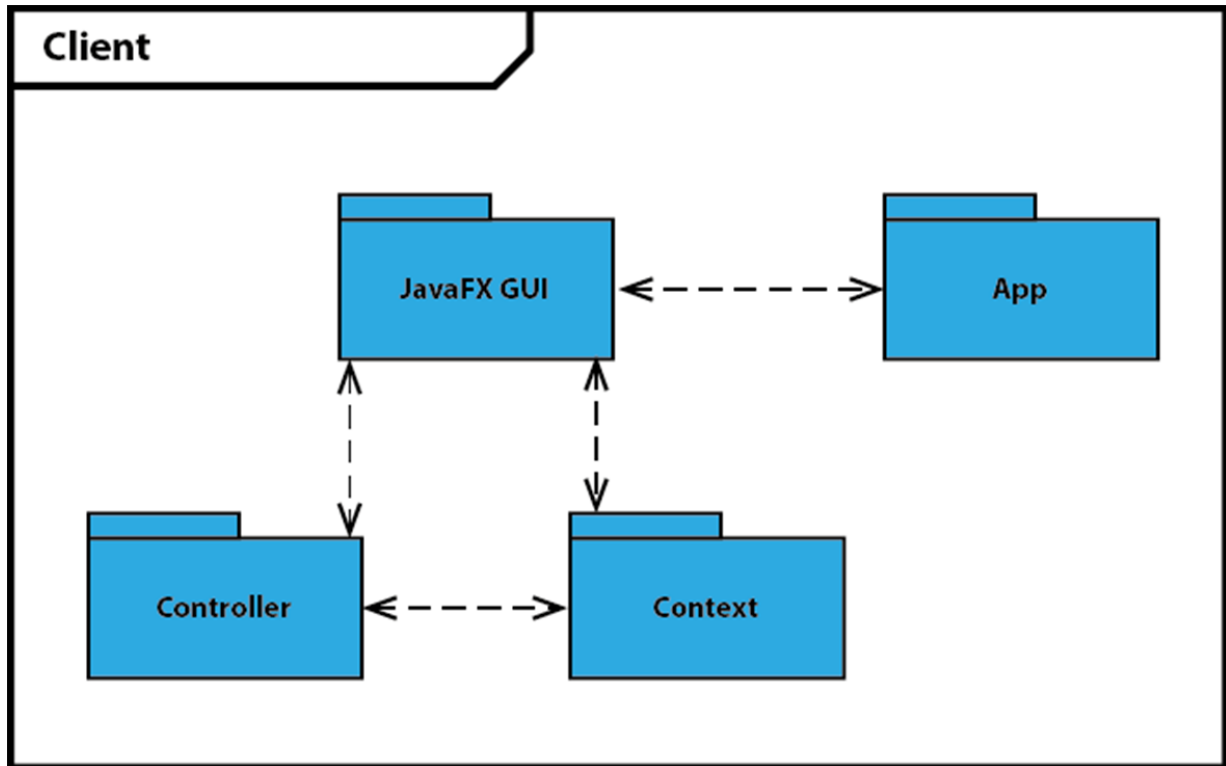


Abbildung 2: Client Komponenten

5.2.1.1 Controller

Bezeichnung	Bedeutung
Name	Controller
Zweck & Verantwortlichkeit	Basis client controller
Schnittstellen	Context, JavaFX GUI
Erfüllte Anforderungen	Webclient
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.1.2 JavaFX GUI

Bezeichnung	Bedeutung
Name	JavaFX GUI
Zweck & Verantwortlichkeit	Hier kann der Benutzer die eigentlichen Daten verwalten, da das GUI für den Benutzer die einzige Schnittstelle darstellt.
Schnittstellen	HTTP Client
Erfüllte Anforderungen	Das GUI ist einfach aufgebaut, verständlich und intuitiv zu bedienen
Variabilität	-
Leistungsmerkmale	Keine Verzögerungen bei Datenaufrufen und – eingaben
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.1.3 Context

Bezeichnung	Bedeutung
Name	Context
Zweck & Verantwortlichkeit	Client user context
Schnittstellen	Controller, JavaFX GUI, HTTP Client
Erfüllte Anforderungen	Benutzerlandschaft Client
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.1.4 App

Bezeichnung	Bedeutung
Name	App
Zweck & Verantwortlichkeit	Hauptklasse für die Client Applikation
Schnittstellen	JavaFX GUI
Erfüllte Anforderungen	Login
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.2 Webservice

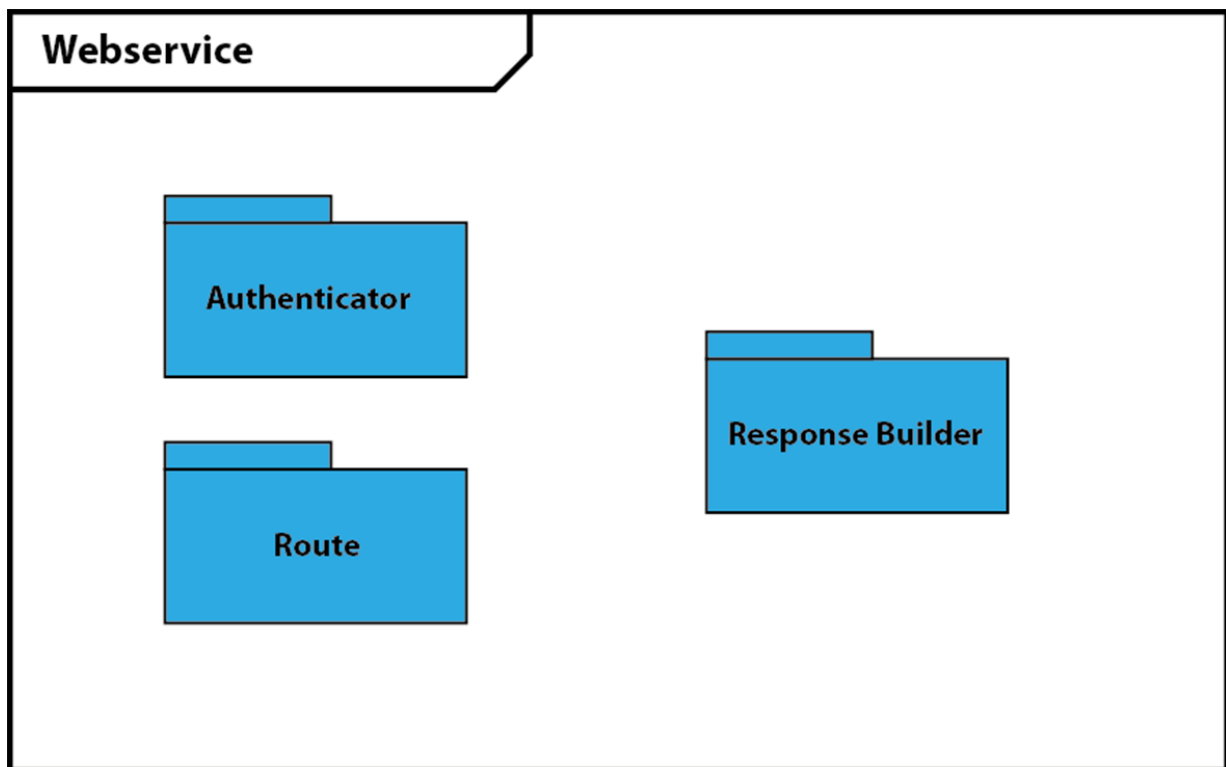


Abbildung 3: Webservice Komponenten

5.2.2.1 Authenticator

Bezeichnung	Bedeutung
Name	Authenticator
Zweck & Verantwortlichkeit	Für jede eingehende Anfrage für diese Klasse prüft den Header auf Authentifizierungsinformation
Schnittstellen	HTTP Session
Erfüllte Anforderungen	Authentifizierte Verbindung
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.2.2 Route

Bezeichnung	Bedeutung
Name	Route
Zweck & Verantwortlichkeit	Wird geladen durch JAX-RS Servlet und definiert die übrigen http routes und methoden
Schnittstellen	HTTP REST
Erfüllte Anforderungen	Model Schnittstellen Webservice
Variabilität	-
Leistungsmerkmale	HTTP REST
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.2.3 Response Builder

Bezeichnung	Bedeutung
Name	Response Builder
Zweck & Verantwortlichkeit	Erstellt http kompatible antworten für Route Anfragen
Schnittstellen	RMI Client
Erfüllte Anforderungen	RMI Verbindung
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.3 Applikationsserver

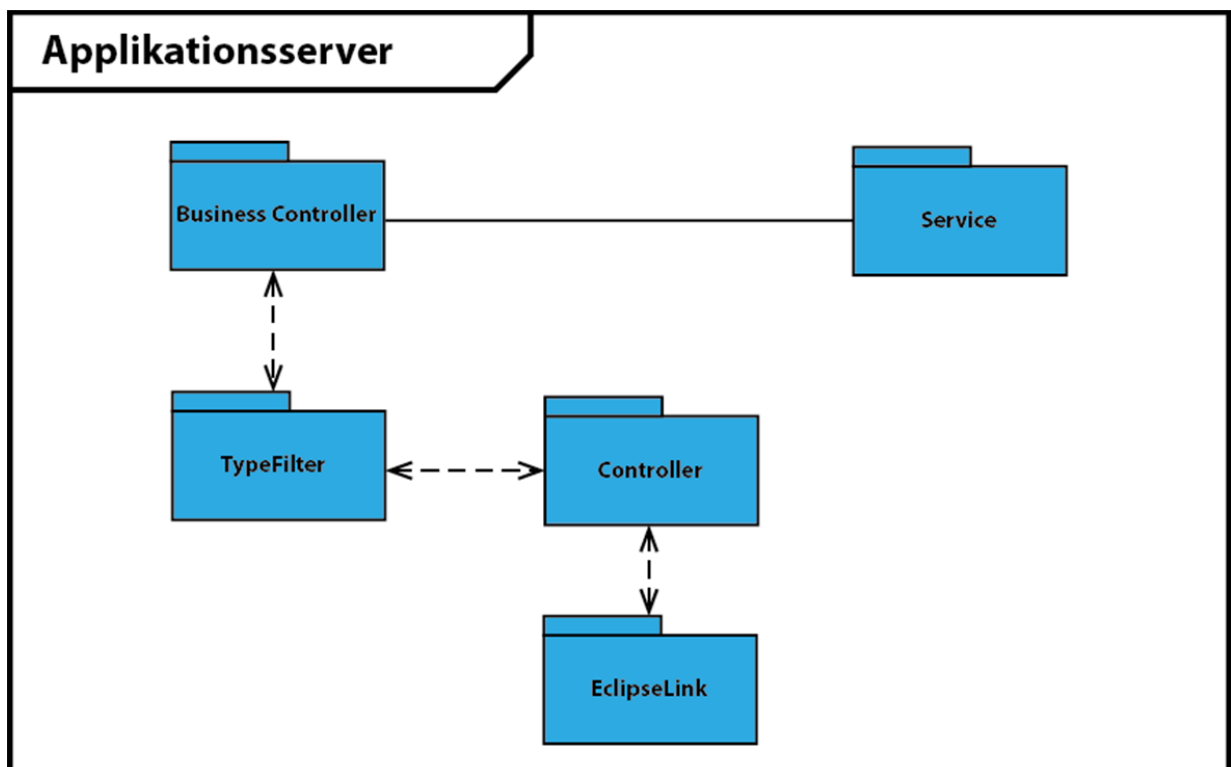


Abbildung 4: Applikationsserver Komponenten

5.2.3.1 Business Controller

Bezeichnung	Bedeutung
Name	Business Controller
Zweck & Verantwortlichkeit	Regeln und Logik für Basis Controller
Schnittstellen	RMI
Erfüllte Anforderungen	RMI Server
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.3.2 TypeFilter

Bezeichnung	Bedeutung
Name	TypeFilter
Zweck & Verantwortlichkeit	Filtert Entitäten und prüft Zugangsrechte auf Rollen und http Methoden
Schnittstellen	Controller
Erfüllte Anforderungen	Business Logik
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.3.3 Controller

Bezeichnung	Bedeutung
Name	Controller
Zweck & Verantwortlichkeit	Basis Server Controller Klasse
Schnittstellen	EclipseLink
Erfüllte Anforderungen	Persistierung
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.3.4 EclipseLink

Bezeichnung	Bedeutung
Name	EclipseLink
Zweck & Verantwortlichkeit	Konfiguriert mit den Eigenschaften des Config Files
Schnittstellen	Datenbank
Erfüllte Anforderungen	ORM
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.2.3.5 Service

Bezeichnung	Bedeutung
Name	Service
Zweck & Verantwortlichkeit	Applikationsserver Service
Schnittstellen	-
Erfüllte Anforderungen	Server starten
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.webservice
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.3 Ebene 3 - GUI

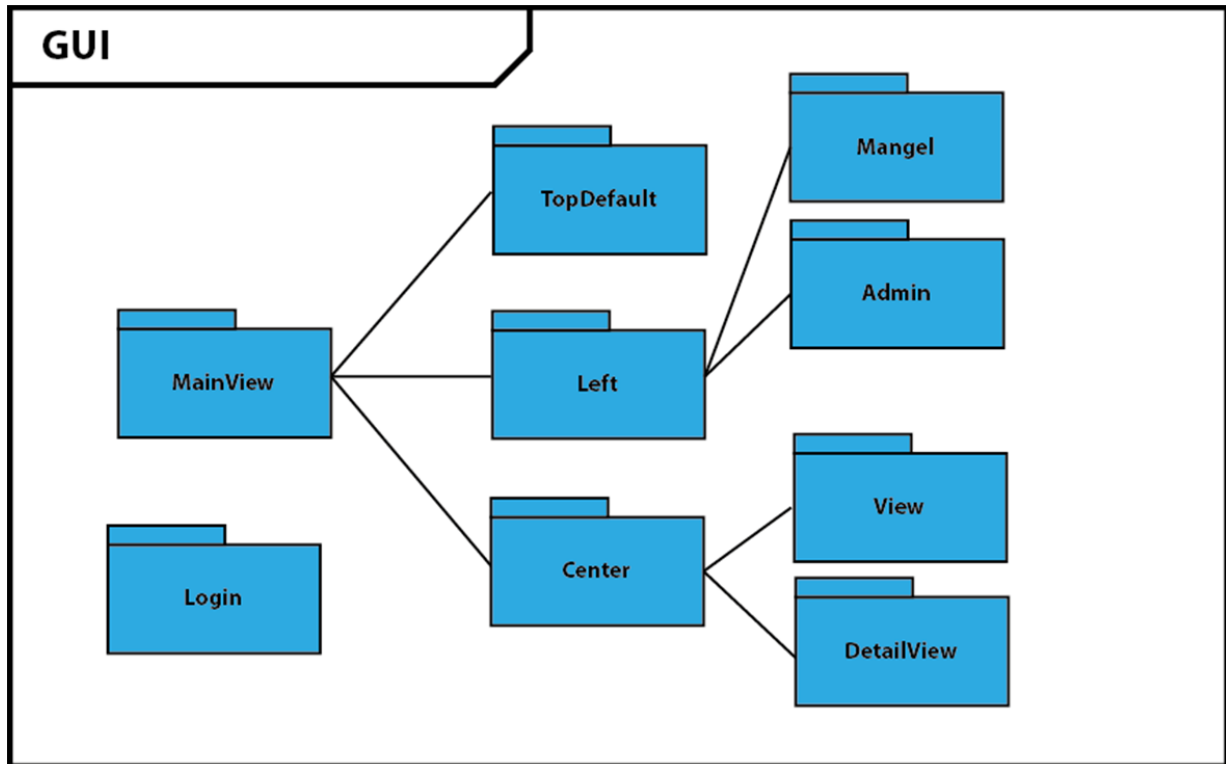


Abbildung 5: Hierarchie der GUI Komponenten

5.3.1 MainView

Bezeichnung	Bedeutung
Name	MainView
Zweck & Verantwortlichkeit	Die MainView beinhaltet die BorderPane und regelt die Navigation
Schnittstellen	Controller, JavaFX
Erfüllte Anforderungen	Multiple Windows
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.3.2 Login

Bezeichnung	Bedeutung
Name	Login
Zweck & Verantwortlichkeit	Das Login gewährleistet, dass der Benutzer mit der richtigen Rolle angemeldet wird
Schnittstellen	Context
Erfüllte Anforderungen	Benutzerlandschaft Client
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.3.3 TopDefault

Bezeichnung	Bedeutung
Name	TopDefault
Zweck & Verantwortlichkeit	TopDefault regelt die Top Navigation und ermöglicht verschiedene LeftViews und CenterViews
Schnittstellen	MainView
Erfüllte Anforderungen	Top Navigation
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.3.4 Left

Bezeichnung	Bedeutung
Name	Left
Zweck & Verantwortlichkeit	In Left wird je nachdem, was in Top ausgewählt wurde, eine Unternavigation dargestellt
Schnittstellen	MainView
Erfüllte Anforderungen	Left Navigation
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.3.5 Center

Bezeichnung	Bedeutung
Name	Center
Zweck & Verantwortlichkeit	Das Center zeigt die eigentliche Datenverwaltung
Schnittstellen	MainView
Erfüllte Anforderungen	Center Navigation
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.3.6 Mangel

Bezeichnung	Bedeutung
Name	Mangel
Zweck & Verantwortlichkeit	Ist eine Left Navigation, welche die aktuellen Projekte anzeigt
Schnittstellen	MainView
Erfüllte Anforderungen	Filterung Mangel
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.3.7 Admin

Bezeichnung	Bedeutung
Name	Admin
Zweck & Verantwortlichkeit	Die Admin Navigation zeigt in ausklappbaren Listen die verschiedenen Projekte, Subunternehmen und Personen an
Schnittstellen	MainView
Erfüllte Anforderungen	Filterung Stammdaten
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.3.8 View

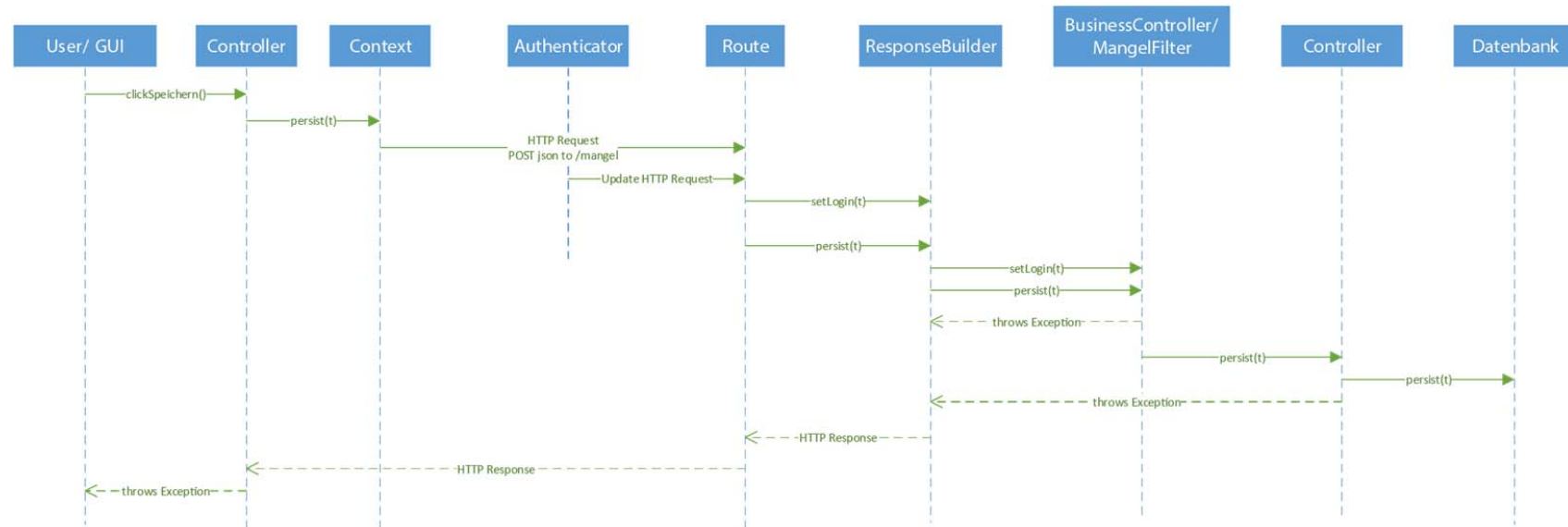
Bezeichnung	Bedeutung
Name	View
Zweck & Verantwortlichkeit	Diese CenterView zeigt eine Übersicht der Projekte, Subunternehmen, Personen oder Mängel in tabellarischer Form
Schnittstellen	MainView
Erfüllte Anforderungen	Diese View zeigt alle Projekte, Subunternehmen, Personen und Mängel an, welche auch sortiert oder durchsucht werden können. Mit einem Doppelklick kann auch jeweils direkt etwas davon aufgerufen werden
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

5.3.9 DetailView

Bezeichnung	Bedeutung
Name	DetailView
Zweck & Verantwortlichkeit	Die DetailView zeigt ein bestimmtes Projekt, Subunternehmen, Person oder Mangel
Schnittstellen	MainView
Erfüllte Anforderungen	DetailView zeigt alle Daten eines bestimmten Projektes, Subunternehmens, Person oder Mangel an und ermöglicht auch die Veränderung von Einträgen
Variabilität	-
Leistungsmerkmale	-
Ablageort / Datei	ch.issueman.client
Sonstige Verwaltungsinformationen	Kann dem Quellcode entnommen werden
Offene Punkte	Keine

6 Laufzeitsicht

6.1 Laufzeitszenario – Mangel erstellen



Hier wird ein Mangel erstellt. Der Bauleiter erfasst die Daten im GUI und klickt auf Speichern. Der Controller macht dann ein Persist und benutzt den Context um einen Benutzerspezifisierten Request abzusetzen. Dabei wird ein HTTP Request generiert. Der Authenticator überprüft diesen Request und schreibt die Login Daten in eine HTTP Session Variable. Im Route wird die Session Login Variable und der Request an den ResponseBuilder weitergereicht. Der ResponseBuilder initiiert die Serververarbeitung und reagiert entsprechend auf Fehler von den unterliegenden Layern. Über RMI gelangt das Login und der Request zum BusinessController. Dieser überprüft Berechtigungen und gibt den Request zur Persistierung an den Server Controller weiter. Der Controller persistiert das Objekt in die Datenbank. Wurde auf keiner Ebene ein Fehler ausgelöst, gibt der Server eine Response mit Status-Code 200 (OK) zurück.

7 Verteilungssicht

Diese Sicht beschreibt, in welcher Umgebung das System abläuft. Sie beschreiben die geographische Verteilung Ihres Systems oder die Struktur der Hardwarekomponenten, auf denen die Software abläuft. Sie dokumentiert Rechner, Prozessoren, Netztopologien und Kanäle, sowie sonstige Bestandteile der physischen Systemumgebung. Die Verteilungssicht zeigt das System aus Betreibersicht.

7.1 Infrastruktur

Knoten

Element	Beschreibung								
Name	PC, Laptop, Tablet								
Beschreibung	Diese Knoten repräsentieren zum Beispiel den Desktop PC des Sachbearbeiters oder der Kontaktperson, den Laptop oder das Tablet des Bauleiters, der sich gerade auf der Baustelle vor Ort aufhält								
Leistungsmerkmale	<p>Die Mindestanforderungen für diese Computer um Java und den Issue-Manager installieren und verwenden zu können sind wie folgt:</p> <table> <tr> <td>Betriebssystem:</td><td>Windows 7 oder höher</td></tr> <tr> <td>Prozessor:</td><td>Pentium</td></tr> <tr> <td>Arbeitsspeicher:</td><td>1000 MB-RAM</td></tr> <tr> <td>Speicherplatz:</td><td>20 GB</td></tr> </table>	Betriebssystem:	Windows 7 oder höher	Prozessor:	Pentium	Arbeitsspeicher:	1000 MB-RAM	Speicherplatz:	20 GB
Betriebssystem:	Windows 7 oder höher								
Prozessor:	Pentium								
Arbeitsspeicher:	1000 MB-RAM								
Speicherplatz:	20 GB								
Zugeordnete Software-Bausteine	<p>Um den Issue-Manager benutzen zu können, müssen wird folgende Software vorausgesetzt:</p> <ul style="list-style-type: none"> • Java Runtime Environment SE7 oder höher • .jar-File und .bat-File müssen sich im selben Verzeichnis befinden 								
Sonstige Verwaltungsinformationen	NA								
Offene Punkte	NA								

Element	Beschreibung
Name	Datenbank Server
Beschreibung	Die Zentrale Datenablage um alle Daten des Issue- Managers zu persistieren MySQL
Leistungsmerkmale	NA
Zugeordnete Software-Bausteine	NA
Sonstige Verwaltungsinformationen	NA
Offene Punkte	NA

8 Konzepte

8.1 Persistenz

Der ORM EclipseLink dient als Grundlage des Persistierungsvorgang und kommuniziert mit der Datenbank. Um den Gebrauch dieses Frameworks weiter zu erleichtern, wurden Controller-Klassen implementiert, welche mittels Generics flexibel auf die verschiedenen Modelle zugreifen kann.

8.2 Benutzungsoberfläche

Die Benutzeroberfläche besteht grundsätzlich aus einem Navigationsbereich (Auswahl zwischen Mangel- und Stammdatensicht), Filterbereich (weitere Aufspaltung der Navigationsauswahl) und einem Zentralbereich, wo die gewählten Informationen präsentiert werden. Diese drei Bereiche werden von einer Vielzahl von „View-Controller“ gesteuert, welche zugehörige FXML-Dateien lädt und mit Informationen versieht. Benutzereingaben werden von den gleichen Controller wieder persistiert.

8.3 Ergonomie

Trotz hoher Funktionalität besitzt die Anwendung ein bestechend einfaches Design (siehe auch Annex 1 - Designkonzept). Geschickte Filterungen ermöglichen das ein- und ausblenden sowohl von Informationen, wie auch von Schaltflächen, um die Benutzeroberfläche zu entlasten. Das Farbkonzept enthält wenige moderne, schlichte Farben, welche durch die ganze Benutzeroberfläche hindurch einheitlich verwendet werden. Dies soll helfen, die bereits intuitive Anwendung noch benutzerfreundlicher zu gestalten.

8.4 Ablaufsteuerung

Das Herzstück der Applikation ist die Mangelübersicht, wo die Mängel eines (angewählten) Projektes angezeigt werden. Angelehnt an das Kanban-Prinzip werden die Mängel bereits in ihre verschiedenen Status gruppiert und in einer prozessorientierten Reihenfolge abgebildet.

8.5 Transaktionsbehandlung

Die Transaktionen werden von EclipseLink durchgeführt und behandelt. Das ACID-Prinzip wird dabei beachtet.

8.6 Sessionbehandlung

Durch den Einsatz von REST werden Session irrelevant. Die ganzen Login-Informationen können jedoch jederzeit durch eine „Context“-Abfrage abgerufen werden.

8.7 Sicherheit

Die Authentifizierung beeinflusst limitiert die berechtigten Routes und ruft verschiedene Custom- und Type-Filter auf, welche weitere Einschränkungen vornimmt. Weitere Sicherheitsmassnahmen wurden im Rahmen dieses Projektes nicht vorgenommen.

8.8 Plausibilisierung und Validierung

Eingabevalidierung auf der Client-Seite konnte aus Zeitgründen nicht umgesetzt werden. Alle Transaktionen die von EclipseLink Server-seitig zugelassen werden, werden somit auf der Datenbank persistiert.

8.9 Ausnahme-/Fehlerbehandlung

Ausnahmen und Fehler, welche zur Laufzeit eintreffen, werden abgefangen und Mittels DialogBox angezeigt. Desweiteren werden sie in einer Log-Datei erfasst, womit Bugs identifiziert und eliminiert werden können.

8.10 Logging, Protokollierung, Tracing

Das Logging wird mit einem schlanken Tinylog-Framework gelöst. Mit dem Einsatz einer Fassade mittels slf4j, können wir problemlos ein anderes Framework wie log4j oder logback mit Tinylog austauschen. Aus Gründen wie einfache Konfiguration, Performance und insbesondere die Verwendung ohne Instanziierung haben uns dazu bewogen das Tinylog-Framework einzusetzen. Durch seine Schlichtheit bietet es nicht die grosse Anpassungsfähigkeit von log4j, aber für unsere Bedürfnisse ist dieses Features nicht von Bedeutung.

Das Logging geschieht nur auf dem Webservice, weil für uns nur diese Logs relevant sind. Der Client wird mittels Error Messages benachrichtigt und diese Nachrichten würden das Log-File unnötig aufblähen und die Fehlersuche erschweren. Die Konfiguration von Tinylog ist genauer unter Kapitel 24 beschrieben.

8.11 Geschäftsregeln

Die Berechtigungen für die verschiedenen Rollen sind in einer Excel-Datei erfasst, welche in ein JSON-Format convertiert werden kann.

8.12 Konfigurierbarkeit

Die Applikation wurde mit einer MySQL Datenbank und einem Gradle Jetty Server entwickelt. Diese Elemente können jedoch sehr einfach ausgetauscht und konfiguriert werden.

8.13 Codegenerierung

Die @Data Annotation (Project Lombok) generiert automatisch die Getters-, Setters, und weitere Methoden. Desweiteren wurden alle FXML-Dateien mit JavaFX Scene Builder 2.0 umgesetzt.

8.14 Buildmanagement

Der gesamte Sourcecode wurde mit GitHub versioniert. Ein Gradle „Seed-Task“ greift auf ein Seed-File im Sourcecode zurück und füllt Dummy-Data in die Datenbank. Somit haben wir sichergestellt, dass alle Entwickler die exakt selben Voraussetzungen haben.

9 Entwurfsentscheidungen

9.1 Entwurfsentscheidung 1: GUI

Fragestellung

Für eine Benutzerfreundliche Bedienung war ein Graphical User Interface (GUI) zwingend. Um dies zu gewährleisten mussten wir uns zwischen Swing und JavaFX entscheiden.

entschieden wir uns für JavaFX, zum Einen, weil wir dies bereits in INM11 ein wenig kennenlernten und zum anderen, weil es angenehm umzusetzen war.

Rahmenbedingungen

Die benutzte Komponente musste aufgrund der gestellten Anforderungen in der Aufgabenstellung mit Java und Eclipse kompatibel sein.

Annahmen

Wir gingen davon aus, dass die GUI Programmierung möglichst einfach sein soll, damit auch Projektmitglieder mit geringeren Programmierer Erfahrungen sich daran beteiligen konnten. Aufgrund von Internet Recherchen klang JavaFX ansprechend, da mit einem zusätzlichen Tool, dem SceneBuilder, GUI's verhältnismässig einfach umsetzen kann.

Betrachtete Alternativen

Wie bereits in der Fragestellung erwähnt, kamen für uns nur zwei Möglichkeiten in Frage, nämlich Swing und JavaFX. Beide erfüllten die Möglichkeit einer visuellen Darstellung einer Anwendung.

Entscheidung

Nach einigen Recherchen und dem Vergleichen der beiden Komponenten haben wir uns als Gruppe für JavaFX entschieden, um das GUI zu implementieren. Die Argumente, welche uns überzeugten waren die Folgenden:

- Mit JavaFX kann man FXML-Dateien erstellen, welche auf relative einfache Art und Weise angepasst werden können
- Ein starkes Tool, der SceneBuilder, basiert auf der Verwendung von JavaFX

9.2 Entwurfsentscheidung 2: Datenbank

Fragestellung

Wie speichern wir die benötigten Objekte in eine relationale Datenbank? Dafür standen uns zahlreiche Varianten zur Verfügung.

Rahmenbedingungen

Eigentlich gab es kaum Rahmenbedingungen. Als einzige Anforderung wurde die SQL-Server Kompatibilität gestellt.

Annahmen

Keine

Betrachtete Alternativen

Keine

Entscheidung

9.3 Entwurfsentscheidung 3: Zugriff des Clients

Fragestellung

Für den Zugriff des Clients auf das System benötigen wir eine Komponente, welche es dem Client ermöglicht mit einem Login sich sicher anmelden zu können.

Rahmenbedingungen

- Sichere Anmeldung des Clients auf das System
- Muss mit Java und Eclipse realisiert werden

Annahmen

Betrachtete Alternativen

- Webservice
- Kombination RMI und Webservice

Entscheidung

9.4 Entwurfsentscheidung 4: Clientabgrenzung

Fragestellung

Da nicht jeder Benutzer die gleichen Rechte hat, musste irgendwie gewährleistet werden, dass der jeweilige Benutzer nur Aktionen zur Verfügung hat, welche innerhalb seiner Rolle sind. So darf zum Beispiel der Sachbearbeiter praktisch alles machen, der Bauleiter aber nur das eigene Projekt verwalten.

Rahmenbedingungen

Muss mit Java und Eclipse realisiert werden.

Annahmen

Betrachtete Alternativen

- Rollenspezifische Interfaces
- Versteckte und sichtbare Funktionen, abhängig von der Rolle des Benutzers

Entscheidung

Wir entschieden uns für ein einheitliches Interface für alle Rollen. Jedoch stehen nicht alle Daten und Methoden allen Benutzern gleichermassen zur Verfügung. Je nach Rolle, welcher der eingeloggte Benutzer hat, werden nur diejenigen Daten und Methoden angezeigt, welche er auch sehen und ausführen darf.

10 Qualitätsszenarien

Dieses Kapitel fasst alles zusammen, was Sie zur systematischen Bewertung Ihrer Architektur gegen vorgegebene Qualitätsziele benötigen.

10.1 Qualitätsbaum



10.2 Bewertungsszenarien

Qualitätsziel	• Erfüllung/ Einhaltung
Bedienbarkeit	<ul style="list-style-type: none"> • Das Programm ist einfach aufgebaut • Jede Ansicht ist leicht verständlich und benötigt keine Erklärungen. Auch die Navigation ist selbsterklärend. • Einzigsten Zugangsbedingungen sind ein gültiges Login und eine Internetverbindung.
Verfügbarkeit	<ul style="list-style-type: none"> • Der issue manager erfüllt alle Anforderungen innerhalb eines angemessenen Zeitrahmens
Performanz	<ul style="list-style-type: none"> • Die Eingaben werden auf Mausklick bearbeitet
Änderbarkeit/ Erweiterbarkeit	<ul style="list-style-type: none"> • Alle Benutzeroberflächen und ihre Funktionen sind einfach aufgebaut und zu verstehen • Durch den ähnlichen Aufbau aller Views, kann der issue manager jederzeit durch neue Views oder Felder ergänzt werden. • Die Anpassungen sind durch einen überschaubaren Aufwand umsetzbar.
Interoperabilität	<ul style="list-style-type: none"> • Die Applikation ist nicht auf ein spezifisches DBMS angewiesen. • Der Besitzer vom issue manger kann frei wählen, mit welchem DBMS er arbeiten möchte. • Der Aufwand für einen DBMS Wechsel ist sehr gering und kann jederzeit vollzogen werden.
Attraktivität	<ul style="list-style-type: none"> • Der Client ist unabhängig von der Server Komponenten. • Dies ermöglicht eine Installation in heterogenen Systemumgebungen. • Die Unabhängigkeit ist somit gewährleistet.
Transparenz	<ul style="list-style-type: none"> • Durch das Open-Source Produkt tinylog, sind die Aktionen der Users nachvollziehbar • Dies ermöglicht bei einem Konflikt oder falschen Daten, die Aufklärung der Ursache. • Die damit geschaffene Transparenz ist sicherlich einen Gewinn.
Analysierbarkeit	<ul style="list-style-type: none"> • Der Aufbau ist anhand der Dokumentation gut nachvollziehbar. • Mit fundierten Java Kenntnisse ist der Code, anhand der Kommentare und dem zum Teil ähnlichen Aufbau, verständlich. • Dies ermöglicht eine Erweiterung, falls nötig, in der Zukunft.

11 Risiken

Risiko	Eintrittswahrscheinlichkeit	Schadenshöhe	Risikovermeidung/-minderung
Server Ausfall	>5%	hoch	<ul style="list-style-type: none">• Ersatz Server
Java Version wird nicht mehr unterstützt	gering	mittel	<ul style="list-style-type: none">• keine
DBMS wird nicht mehr unterstützt	gering	gering	<ul style="list-style-type: none">• keine

12 Glossar

Abkürzung	Bedeutung
GUI	Graphical User Interface
DBMS	Database Management System
TDD	Test Driven Development
JDK	Java Development Kit
RMI	Remote Method Invocation
REST	Representational State Transfer
ORM	Object-relational mapping
JDK	Java Development Kit

13 Weitere Arc42 Dokumentationen

Es werden keine weiteren Arc42 Dokumentationen gebraucht.

14 Requirements Dokumentation

Aus dem Projektauftrag wurden die Anforderungen als Requirements in das vorgegebene Excel-File übertragen.

Projekt Mängel-Manager - Requirements		Gruppe: C	
Nr.	Anforderung	Stakeholder	Funktional Testen mit
Req001	Verwalten der notwendigen Daten aller Mitarbeiter des Generalunternehmung (CRUD)	Sachbearbeiter	5
Req002	Verwalten der notwendigen Daten aller Bauherren (CRUD)	Sachbearbeiter	4
Req003	Verwalten der notwendigen Daten aller Subunternehmen (CRUD)	Sachbearbeiter	
Req004	Verwalten der notwendigen Daten aller Mitarbeiter des Subunternehmens (CRUD)	Sachbearbeiter	
Req005	Verwalten der notwendigen Daten aller Projekte (CRUD)	Sachbearbeiter, Bauleiter	9
Req006	Verwalten der notwendigen Daten aller Mängel (CRUD)	Sachbearbeiter, Bauleiter	7
Req007	Verwalten der notwendigen Daten aller Kommentare (CRUD)	Sachbearbeiter, Bauleiter, Subunternehmen	7
Req008	Liste aller Mängel exportieren	Sachbearbeiter, Bauleiter, Subunternehmen	
Req009	Applikation steht als Download zur Verfügung	Subunternehmen	
Req010	Zugriff auf rollenspezifische Daten mittels Login schützen		
Req011	Nachvollziehbarkeit von Änderungen (insbesondere bei Projekt, Mangel und Kommentaren)		
Req012	Logging der Applikation		
Req013	Applikation als verteilte Anwendung erstellen		
Req014	DBMS muss austauschbar sein		
Req015	Applikation nach Multi-Tier Architektur erstellen		
Req016	Unterschiedliche GUIs für Backoffice (GU), Bauleiter und Subunternehmen		
Req017	JUnit-Test nach dem TDD-Verfahren durchführen		
Req018	Abnahme durch funktionale Tests		

Die oben aufgelisteten Requirments bilden das Lastenheft, aus welchem ein Pflichtenheft ausformuliert wird. Die Requirements aus dem Pflichtenheft werden weiter unten genauer beschrieben und in funktionalen und nicht-funktionalen Requirements aufgeteilt. Die Tabelle dient lediglich zur Gesamtübersicht.

REQ #	Requirement	Stakeholder	UCA #	ADI #	TCA #
Authentifizierung			10		
11	Anmelden mit Benutzername und Passwort	SB, BL, SU	10		13
12	Abmelden und Beenden der Session	SB, BL, SU	10		14
13	Rollenbasierte Berechtigungen	SB, BL, SU	10, 50		15
14	Passwort ändern	SB, BL, SU	10		16
15	Passwort zurücksetzen	SB, BL, SU	10		17
Stammdatenverwaltung			20		
21	Neue Stammdaten erfassen	SB	21-1, 22-1, 23-1, 24-1		18
22	Daten der Bauleiter verwalten	SB	21-2, 21-3		19
23	Daten der Bauherren verwalten	SB	22-2, 22-3		19
24	Daten der Subunternehmen verwalten	SB, BL, SU	23-2, 23-3		19
25	Daten der Bauprojekte verwalten	SB, BL	24-2, 24-3	24	19

Mängelerfassung			30	
31	Mangel erfassen	SB, BL	31	20
32	Betreffendes Projekt zuordnen	SB, BL	32	20
33	Mangel beschreiben (kurzer Beschrieb)	SB, BL	33	20
34	Betreffendes Subunternehmen zuordnen	SB, BL	34	20
35	Status setzen	SB, BL	35	20
36	Frist definieren	SB, BL	36	20
37	Erfassungsdatum wird notiert	SB, BL	38	20
Mängelbearbeitung			40	
41	Mangeldaten verändern	SB, BL, SU	41	20
42	Mangelstatus verändern	SB, BL, SU	42	42 20
43	Kommentar abgeben	SB, BL	43	20
44	Mängeldaten exportieren	SB, BH, BL, SU	44	21
Design				
51	GUI	KSS	51, 52, 53, 54, 55	
52	Sortierung von Tabellen	KSS		
53	Suchfunktion	KSS		
54	Datenvalidierung	KSS		
55	Aufforderung zum Speichern	KSS		
56	Datenintegrität	KSS		
57	Warnungen	KSS		
Technische Anforderungen			60	
61	Entwicklung mit Eclipse und JDK 1.8	KSS		
62	Client und Server sind verteilbare Komponenten	KSS		
63	Dokumentation mit JavaDoc	KSS		
64	Abstraktion des DBMS Layer mit ORM	KSS		
65	GUI Entwicklung mit JavaFX	KSS		
66	Event Logging	KSS		
67	DBMS muss austauschbar sein	KSS		
68	Datenverarbeitung auf Applikationsserver	KSS		
69	Applikation als Download für das Subunternehmen	SU		
Testing				
71	JUnit-Test Tests nach TDD-Verfahren	KSS	01-12	

72	Abnahme mittels funktionalen Tests	KSS	13-21
REQ = Requirement, UCA = UseCase, ADI = Aktivitätsdiagramm, TCA = TestCase Stakeholder: SB = Sachbearbeiter, BL = Bauleiter, SU = Subunternehmen, BH = Bauherr, KSS = Kein spezifischer Stakeholder			

14.1 Funktionale Requirements

Authentifizierung

REQ11 - Anmelden mit Benutzername und Passwort

Beschreibung	Die Anmeldung muss mit der E-Mail-Adresse sowie einem Passwort möglich sein.
Stakeholder	<ul style="list-style-type: none">- Sachbearbeiter- Bauleiter- Subunternehmen
Erfolgskriterium	Die Anmeldung muss ohne zusätzlichen Aufwand möglich sein.
Referenzen	UCA10

REQ12 – Abmelden mit Beenden der Session

Beschreibung	Bei der Abmeldung wird die Session beendet.
Stakeholder	<ul style="list-style-type: none">- Sachbearbeiter- Bauleiter- Subunternehmen
Erfolgskriterium	Bei der Abmeldung wird die aktive Session beendet.
Referenzen	UCA10

REQ13 – Rollenbasierte Berechtigungen

Beschreibung	Nach der Anmeldung muss dem User die entsprechende User Rolle zugeteilt werden.
Stakeholder	<ul style="list-style-type: none">- Sachbearbeiter- Bauleiter- Subunternehmen
Erfolgskriterium	Nach der Anmeldung sind für den entsprechenden User nur die Daten sichtbar, auf welche er auch Zugriff hat bzw. auf die Projekte, in welche er Einsicht hat.
Referenzen	UCA10, UCA50

REQ14 – Passwort ändern

Beschreibung	Änderung des persönlichen Passwortes.
Stakeholder	<ul style="list-style-type: none">- Sachbearbeiter- Bauleiter- Subunternehmen

Erfolgskriterium	Das Passwort kann von der Client-Applikation aus geändert werden.
Referenzen	UCA10

REQ15 – Passwort zurücksetzen

Beschreibung	Bei vergessenem Passwort kann der Sachbearbeiter über die Client-Applikation das Passwort zurückgesetzt werden.
Stakeholder	<ul style="list-style-type: none"> - Sachbearbeiter - Bauleiter (kann sich beim SB melden) - Subunternehmen (kann sich beim SB melden)
Erfolgskriterium	User kann in der Client-Applikation das Passwort zurücksetzen.
Referenzen	UCA10

Stammdatenverwaltung

REQ21 - Neue Stammdaten erfassen

Beschreibung	<p>Es müssen Stammdaten erfasst werden können. Zu den Stammdaten gehören folgende Daten:</p> <ul style="list-style-type: none"> - Subunternehmen <ul style="list-style-type: none"> o Administrative Kontaktperson aus dem Subunternehmen o Projektspezifische Kontaktperson aus dem Subunternehmen o Adresse - Bauherr (als Privatperson oder Firma) <ul style="list-style-type: none"> o Adresse - Bauleiter - Bauprojekt <ul style="list-style-type: none"> o Projekttyp (Einfamilienhaus, Mehrfamilienhaus, Wohnung, Garage, etc.) o Bauvorhaben (Neu- oder Umbau, Renovation, Teil-Renovation) o Zeitperiode (Start und Ende) o Adresse o Bauleiter o Administrative Kontaktperson aus dem Subunternehmen <p>Die Rechte für die Erfassung dieser Daten obliegt den Sachbearbeitern des Generalunternehmens.</p>
Stakeholder	<ul style="list-style-type: none"> - Sachbearbeiter
Erfolgskriterium	Sachbearbeiter können die oben erwähnten Stammdaten erfolgreich erfassen.
Referenzen	UCA20, UCA22, UCA23, UCA24, UCA25

REQ22 - Daten der Bauleiter verwalten

Beschreibung	Die Daten der Bauleiter müssen gelesen und aktualisiert werden können. Die Rechte für die Verwaltung obliegt den Sachbearbeitern. Es muss möglich sein andere Bauleiter für ein Projekt zu definieren und diese Veränderung muss nachvollziehbar sein.
Stakeholder	- Sachbearbeiter
Erfolgskriterium	Sachbearbeiter können erfolgreich die Daten der Bauleiter lesen und aktualisieren.
Referenzen	UCA22

REQ23 - Daten der Bauherren verwalten

Beschreibung	Die Daten der Bauherren müssen gelesen und aktualisiert werden können. Die Rechte für die Verwaltung obliegt den Sachbearbeitern.
Stakeholder	- Sachbearbeiter
Erfolgskriterium	Sachbearbeiter können erfolgreich die Daten der Bauherren lesen und aktualisieren.
Referenzen	UCA23

REQ24 - Daten der Subunternehmen verwalten

Beschreibung	Die Daten der Subunternehmen müssen gelesen und aktualisiert werden können. Die Rechte für die Verwaltung über alle Subunternehmen haben nur die Sachbearbeiter des GU. Die Bauleiter und administrativen Kontaktpersonen der Subunternehmen können lediglich ihre eigenen Daten verwalten.
Stakeholder	<ul style="list-style-type: none"> - Sachbearbeiter - Bauleiter - Subunternehmen
Erfolgskriterium	<ul style="list-style-type: none"> - Sachbearbeiter können erfolgreich die Daten der Subunternehmen lesen und aktualisieren. - Bauleiter können erfolgreich die Daten der Subunternehmen von seinen Projekten lesen und aktualisieren. - Subunternehmen (Administrative Kontaktperson) können erfolgreich ihre eigenen Daten lesen und aktualisieren.
Referenzen	UCA24

REQ25 - Daten der Bauprojekte verwalten

Beschreibung	Die Daten der Bauprojekte müssen gelesen und aktualisiert werden können. Die Rechte für die Verwaltung aller Bauprojekte obliegt den Sachbearbeitern des GU und die Bauleiter können ihre eigenen Projekte verwalten.
Stakeholder	<ul style="list-style-type: none">- Sachbearbeiter- Bauleiter
Erfolgskriterium	<ul style="list-style-type: none">- Sachbearbeiter können erfolgreich die Daten aller Bauprojekte lesen und aktualisieren.- Bauleiter können erfolgreich die Daten ihrer Bauprojekte lesen und aktualisieren.
Referenzen	UCA25

Mängelerfassung

REQ31 – Mangel erfassen

Beschreibung	Der Bauleiter hat nach einer Besichtigung die Möglichkeit, vor Ort einen Mangel zu erfassen.
Stakeholder	<ul style="list-style-type: none">- Bauleiter- Sachbearbeiter (muss nicht direkt Mängel erfassen können, besitzt aber Adminrechte)
Erfolgskriterium	Der Bauleiter kann vor Ort einen Mangel erfassen.
Referenzen	UCA31

REQ32 – Betreffendes Projekt zuordnen

Beschreibung	Der Bauleiter kann den Mangel dem entsprechenden Projekt zuordnen.
Stakeholder	<ul style="list-style-type: none">- Bauleiter- Sachbearbeiter (muss nicht direkt Mängel erfassen können, besitzt aber Adminrechte)
Erfolgskriterium	Der Bauleiter kann jeden erfassten Mangel dem entsprechenden Projekt zuordnen.
Referenzen	UCA32

REQ33 – Mangel beschreiben (kurzer Beschrieb)

Beschreibung	Der Bauleiter kann den festgestellten Mangel (kurz) beschreiben und die fehlerhaften Punkte auflisten.
Stakeholder	<ul style="list-style-type: none"> - Bauleiter - Sachbearbeiter (muss nicht direkt Mängel erfassen können, besitzt aber Adminrechte)
Erfolgskriterium	Der Bauleiter muss den erfassten Mangel kurz beschreiben und die wichtigsten Punkte auflisten können.
Referenzen	UCA33

REQ34 – Betreffendes Subunternehmen zuordnen

Beschreibung	Der Bauleiter kann den erfassten Mangel dem zuständigen Subunternehmen zuordnen.
Stakeholder	<ul style="list-style-type: none"> - Bauleiter - Sachbearbeiter (muss nicht direkt Mängel erfassen können, besitzt aber Adminrechte)
Erfolgskriterium	Der Bauleiter muss jedem erfassten Mangel das betreffende Projekt zuordnen können.
Referenzen	UCA34

REQ35 – Status setzen

Beschreibung	Wenn der Bauleiter einen Mangel erfasst, kann er einen Status setzen.
Stakeholder	<ul style="list-style-type: none"> - Bauleiter - Sachbearbeiter (muss nicht direkt Mängel erfassen können, besitzt aber Adminrechte)
Erfolgskriterium	Der Bauleiter muss bei jedem Mangel ein Status setzen können.
Referenzen	UCA35

REQ36 – Frist definieren

Beschreibung	Der Bauleiter kann die Frist definieren, bis wann der Mangel behoben sein muss.
Stakeholder	<ul style="list-style-type: none"> - Bauleiter - Sachbearbeiter (muss nicht direkt Mängel erfassen können, besitzt aber Adminrechte)
Erfolgskriterium	Der Bauleiter muss eine Frist definieren können.
Referenzen	UCA36

REQ37 – Erfassungsdatum wird notiert

Beschreibung	Der Bauleiter kann das Erfassungsdatum der Kommentare notieren.
Stakeholder	<ul style="list-style-type: none">- Bauleiter- Sachbearbeiter (muss nicht direkt Mängel erfassen können, besitzt aber Adminrechte)
Erfolgskriterium	Der Bauleiter muss das Erfassungsdatum der Kommentare notieren können.
Referenzen	UCA38

Mängelbearbeitung

REQ41 – Mangeldaten verändern

Beschreibung	Der Bauleiter kann von sich aus oder auf Anfrage des Subunternehmers (Status „abzuklären“ und Kommentar), die Bezeichnung, das Subunternehmen und die Frist des Mangels anpassen.
Stakeholder	<ul style="list-style-type: none">- Bauleiter- Sachbearbeiter
Erfolgskriterium	<ul style="list-style-type: none">- Bauleiter muss die Bezeichnung und die Frist ändern können- Sachbearbeiter muss die Bezeichnung und die Frist ändern können
Referenzen	UCA41

REQ42 – Mangelstatus verändern

Beschreibung	<p>Nachdem ein neuer Mangel erhoben wird, kann er die folgenden Status durchlaufen:</p> <ol style="list-style-type: none"> 1) beauftragt: ursprünglicher Status, der direkt nach Erhebung zugewiesen wird. Der Bauleiter kann den Mangel zu jedem Zeitpunkt auf diesen Status zurücksetzen um den Prozess erneut zu starten. 2) angenommen: der Subunternehmer bestätigt mit diesem Status, dass er den Mangel zur Kenntnis genommen hat und ihn bearbeitet. 3) behoben: der Subunternehmer informiert mit diesem Status, dass der Mangel erledigt wurde und somit vom Bauleiter auf dessen Zufriedenheit überprüft werden kann. 4) abgeschlossen: gilt der Mangel für den Bauleiter als erledigt, kann er ihn mit diesem Status vom aktiven Raster nehmen. <p>Nebst diesem normalen Ablauf kann der Subunternehmer zu jedem Zeitpunkt (üblicherweise vor Annahme des Mangels) den Status „abzuklären“ wählen. Damit signalisiert er, dass er den Mangel ohne Rücksprache mit dem Bauleiter nicht annehmen und/oder bearbeiten kann.</p>
Stakeholder	<ul style="list-style-type: none"> - Bauleiter (hat folgende Status zur Auswahl: beauftragt, abgeschlossen) - Subunternehmer (hat folgende Status zur Auswahl: angenommen, behoben, abzuklären) - Sachbearbeiter (ist nicht direkt mit dem Auftrag der Mängelverwaltung betraut, geniesst die gleichen Rechte wie der Bauleiter)
Erfolgskriterium	Stakeholder können den Status von Mängeln gemäss ihrer Statusauswahl ändern.
Referenzen	UCA42

REQ43 – Kommentar abgeben

Beschreibung	<p>Die Kommentarfunktion dient dazu, zusätzliche Informationen zwischen Bauleiter und Subunternehmer auszutauschen. Dies ist hauptsächlich der Fall, um den Status „abzuklären“ zu begründen und zu entgegnen. Es kann aber auch zu jedem anderen Status als ergänzendes Detail dienen.</p> <p>Die Kommentare können aus Gründen der Nachvollziehbarkeit nach Abgabe nicht verändert werden.</p>
Stakeholder	<ul style="list-style-type: none"> - Bauleiter - Subunternehmer - Sachbearbeiter (ist nicht direkt mit dem Auftrag der Mängelverwaltung betraut, kann jedoch ebenfalls Kommentare verfassen)
Erfolgskriterium	Stakeholder können Kommentare zu einem Mangel abgeben.
Referenzen	UCA43

REQ44 – Mängeldaten exportieren

Beschreibung	Alle relevanten Informationen bezüglich Mängel können pro Projekt in Listenform exportiert und ausgedruckt werden. Es gibt zwei mögliche Ansichten, einerseits nach Erledigungsdatum, anderseits aggregiert nach Status.
Stakeholder	<ul style="list-style-type: none">- Bauleiter- Subunternehmer (nur die ihm zugewiesenen Mängel)- Sachbearbeiter (ist nicht direkt mit dem Auftrag der Mängelverwaltung betraut, kann jedoch Mängeldaten exportieren)- Bauherr (ist nicht direkter Stakeholder der Applikation, stellt aber eine wichtige Zielgruppe für die exportierte Mängelliste dar)
Erfolgskriterium	Stakeholder können Mängeldaten exportieren.
Referenzen	UCA44

14.2 Nicht-funktionale Requirements

Design

REQ51 – GUI

Beschreibung Die Benutzer können mittels einem GUI auf Daten zugreifen und sie verwalten.

REQ52 – Sortierung von Tabellen

Beschreibung In allen GUIs müssen die einzelnen Tabellen sortiert und gefiltert werden können. Auch sollen verschiedene Ansichten möglich sein, wo man einzelne Spalten nach Bedarf anzeigen lassen kann oder nicht.

REQ53 – Suchfunktion

Beschreibung Das User Interface soll die Möglichkeit bieten, nach einem Begriff zu suchen. Hierfür soll eine Fuzzy Search zur Anwendung kommen. D.H. Suchergebnisse sollen nicht den exakten Begriff finden, sondern Elemente in denen die eingegebene Zeichenkette unter anderem vorkommt.

REQ54 – Datenvalidierung

Beschreibung Nach der Eingabe von Daten im GUI soll der Benutzer um Bestätigung gefragt werden, ob alle Daten korrekt eingetragen sind.

REQ55 – Aufforderung zum Speichern

Beschreibung Wenn der Benutzer im GUI die Eingabeseite verlassen will, soll er aufgefordert werden, die bisherigen Eingaben zu speichern. Dem Benutzer soll ins Bewusstsein gerufen werden, dass bei einem Verlassen der Seite, alle Daten verloren gehen, wenn keine Speicherung erfolgt.

REQ56 – Datenintegrität

Beschreibung Werden Daten im GUI nicht korrekt eingegeben, soll eine Warnung erfolgen (Bsp. Kein @ in einer Email-Adresse)

REQ57 – Warnungen bei Lösch- oder Änderungsvorgängen

Beschreibung Werden im GUI Datensätze gelöscht oder verändert, so soll der Benutzer gewarnt werden. Lösch- und Änderungsvorgänge müssen explizit vom Benutzer bestätigt werden.

Technische Anforderungen

REQ61 - Entwicklung mit Eclipse und JDK 1.8

Beschreibung Die Java Applikation muss mithilfe der Eclipse IDE realisiert werden und soll JDK 1.8 kompatibel sein.

REQ62 - Client und Server sind verteilbare Komponenten

Beschreibung Die Client- und Server-Komponenten können unabhängig von einander in heterogenen Systemumgebungen installiert und konfiguriert werden.

REQ63 - Dokumentation mit JavaDoc

Beschreibung	Methoden, Klassen, Interfaces und Exceptions müssen dokumentiert werden. Die Dokumentation wird mit JavaDoc innerhalb von Eclipse geführt.
--------------	--

REQ64 - Abstraktion des DBMS Layer mit ORM

Beschreibung	Auf spezifische DBMS Unterstützung muss verzichtet werden. Dies hat den Einsatz eines ORM (object-relational mapping) zur Folge.
--------------	--

REQ65 - GUI Entwicklung mit JavaFX

Beschreibung	Die GUI Komponente muss mit dem JavaFX-Framework realisiert werden.
--------------	---

REQ66 - Event Logging

Beschreibung	Zur Nachvollziehbarkeit (Auditing) und Fehlerbehandlung (Exception Handling) müssen alle Events und Benutzer Aktionen geloggt werden.
--------------	---

REQ67 - DBMS muss austauschbar sein

Beschreibung	Das DBMS soll jederzeit ausgetauscht werden können und die Integration von Hersteller unabhängigen Produkten erlauben.
--------------	--

REQ68 - Datenverarbeitung auf Applikationsserver

Beschreibung	Daten, die auf der Client Applikation bearbeitet wurden, müssen von einer Server Komponente in Empfang genommen und weiter verarbeitet werden. Es empfiehlt sich die Gliederung der Applikation in eine 3-Tier Architektur.
--------------	---

REQ69 - Applikation als Download für das Subunternehmen

Beschreibung	Das Subunternehmen muss die Möglichkeit haben, die Applikation als Download zu beziehen und bei sich im Unternehmen in Betrieb zu nehmen.
--------------	---

Testing

REQ71 - JUnit-Test Tests nach TDD-Verfahren

Beschreibung	Für die relevanten Methoden müssen JUnit-Tests nach dem TDD-Verfahren erstellt werden.
--------------	--

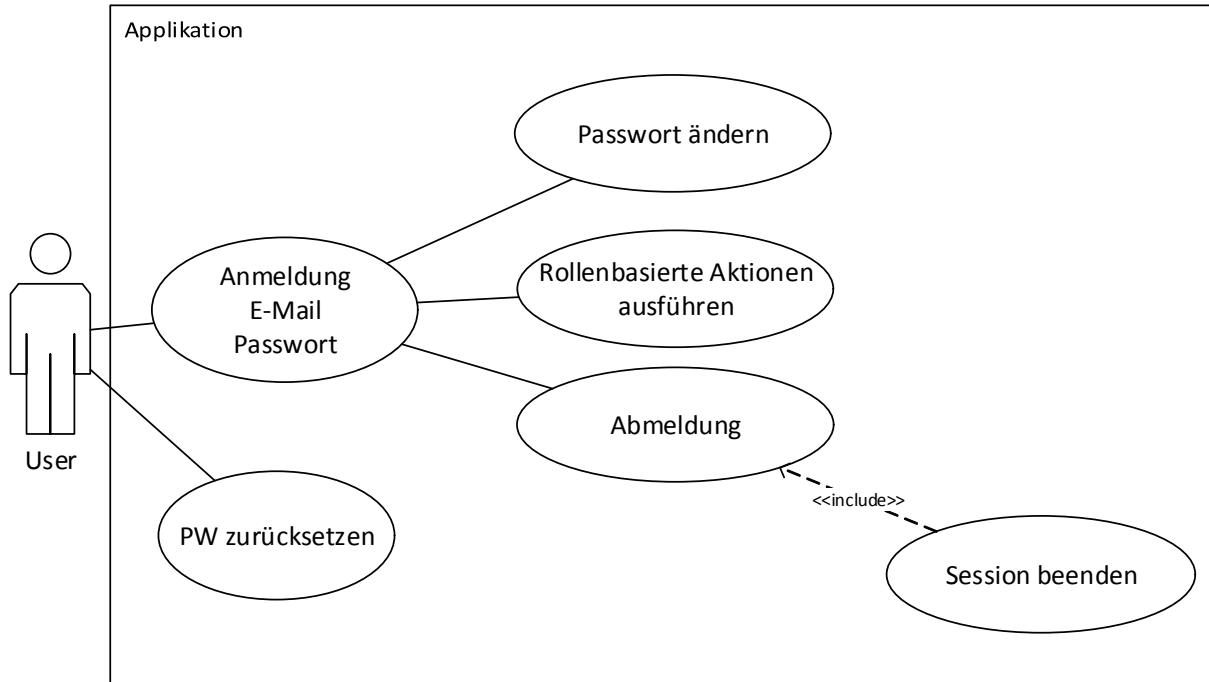
REQ72 - Abnahme mittels funktionalen Tests

Beschreibung	Anhand der definierten Testfälle werden die funktionalen Requirements getestet.
--------------	---

15 UseCase Dokumentation

Die Kategorie für die unten aufgeführten UseCases ist Primär. Um eine Redundanz zu vermeiden, wird auf die Auflistung der Kategorie in den einzelnen Beschreibungen verzichtet.

UCA10 – Authentifizierung



UCA11 – Anmeldung

Autor	Erwin Willi
Ziel	Anmeldung mit E-Mail und Passwort
Vorbedingungen	<ul style="list-style-type: none"> - User muss erfasst sein und einer entsprechenden Rolle zugeteilt sein - Projekt muss bereits im System erfasst worden sein durch BO/GU welchem der User zugeordnet ist - Passwort möchte geändert werden - Passwort wurde vergessen
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Der User konnte sich erfolgreich Anmelden und Änderungen im entsprechendem Projekt durchführen - Passwort konnte geändert werden - Abmeldung war erfolgreich - Passwort konnte zurückgesetzt werden.
Nachbedingungen Fehlslag	<ul style="list-style-type: none"> - Die Authentifizierung ist fehlgeschlagen - Der User hat Zugriff auf ein von Ihm nicht verwaltetes Projekt - Das Passwort konnte nicht geändert werden - Abmeldung war nicht erfolgreich

	- Das Passwort konnte nicht zurückgesetzt werden
Akteure	User (Bauleiter, Subunternehmer, Sachbearbeiter) Bauherr
Auslösendes Ereignis	An/Abmeldung an Client-Applikation
Beschreibung	<ol style="list-style-type: none"> 1. Der Benutzer muss die E-Mail Adresse und sein persönliches Passwort eingeben. 2. User wird authentifiziert 3. Bei Erfolg hat er Zugriff auf seine Projekte und Mängel
Erweiterung	Keine
Alternativen	Keine

UCA12 – Passwort ändern

Autor	Erwin Willi
Ziel	Persönliches Passwort ändern
Vorbedingungen	<ul style="list-style-type: none"> - User muss erfolgreich authentifiziert sein. - User möchte sein persönliches Passwort ändern.
Akteure	User (Bauleiter, Subunternehmer, Sachbearbeiter) Bauherr
Auslösendes Ereignis	Persönliches Passwort ändern
Beschreibung	<ol style="list-style-type: none"> 5. Passwortwechsel anwählen 6. Bestehendes Passwort bestätigen 7. Neues Passwort eingeben 8. Neues Passwort bestätigen 9. Neues Passwort ist gesetzt.
Erweiterung	Keine
Alternativen	Keine

UCA13 – Rollenbasierte Aktionen ausführen

Autor	Erwin Willi
Ziel	Änderungen an bestehendem Projekt durchführen.
Vorbedingungen	<ol style="list-style-type: none"> 1. User muss erfolgreich authentifiziert sein. 2. Wunsch auf Änderungen in bestehendem Projekt
Akteure	User (Bauleiter, Subunternehmer, Sachbearbeiter) Bauherr
Auslösendes Ereignis	Wunsch auf Änderungen in bestehendem Projekt
Beschreibung	<ol style="list-style-type: none"> 3. Gewünschte Änderungen am Projekt durchführen 4. Änderungen speichern
Erweiterung	Keine
Alternativen	Keine

UCA14 – Abmeldung

Autor	Erwin Willi
Ziel	Abmeldung von der Applikation
Vorbedingungen	<ol style="list-style-type: none"> 4. User muss erfolgreich authentifiziert sein. 5. Gewünschte Änderungen am Projekt wurden durchgeführt und gespeichert.
Akteure	User (Bauleiter, Subunternehmer, Sachbearbeiter) Bauherr
Auslösendes Ereignis	Abmeldung von der Applikation
Beschreibung	<ol style="list-style-type: none"> 5. User meldet sich über Logout ab
Erweiterung	Keine
Alternativen	Keine

UCA15 – Session beenden

Autor	Erwin Willi
Ziel	Session beenden
Vorbedingungen	<ol style="list-style-type: none"> 6. User Abmeldung muss erfolgreich gewesen sein.
Akteure	User (Bauleiter, Subunternehmer, Sachbearbeiter)

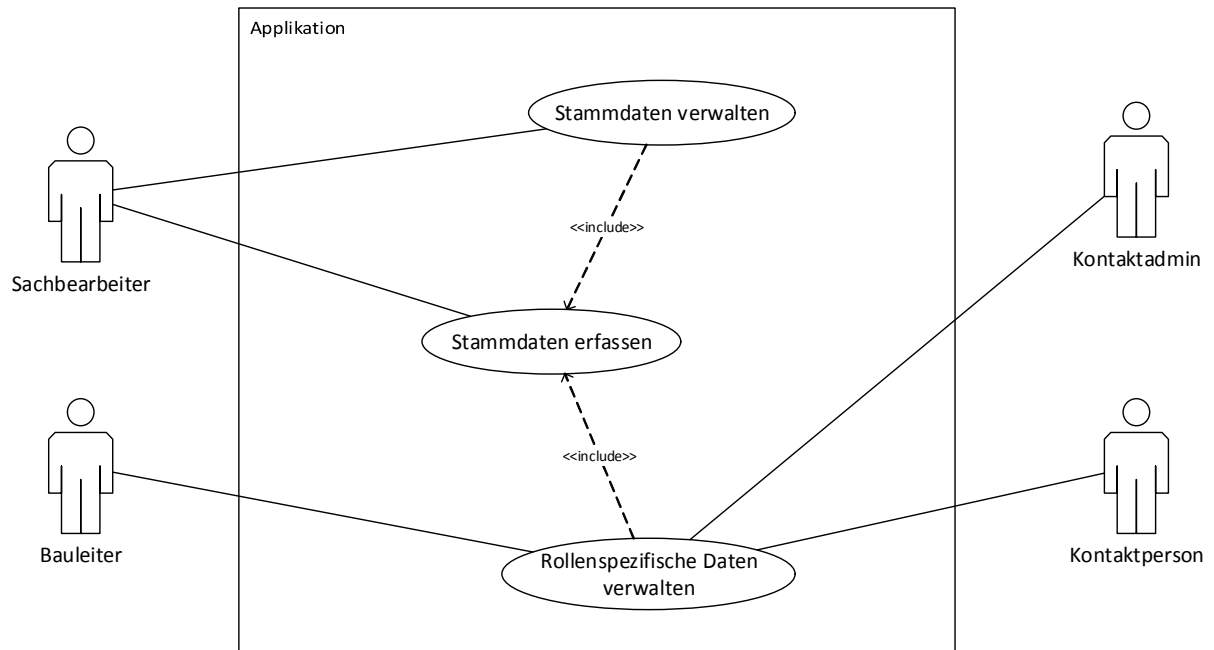
	Bauherr
Auslösendes Ereignis	Session beenden nach Abmeldung
Beschreibung	7. Session wird nach klicken auf Logout beendet.
Erweiterung	Keine
Alternativen	Keine

UCA16 – Passwort zurücksetzen

Autor	Erwin Willi
Ziel	Passwort zurücksetzen
Vorbedingungen	6. User hat Passwort vergessen
Akteure	User (Bauleiter, Subunternehmer, Sachbearbeiter) Bauherr
Auslösendes Ereignis	User hat sein persönliches Passwort vergessen und möchte es zurücksetzen lassen.
Beschreibung	8. User muss sich bei Sachbearbeiter melden 9. Sachbearbeiter setzt das Passwort zurück.
Erweiterung	Keine
Alternativen	Keine

UCA20 – Stammdatenverwaltung (Übersicht)

Auf das Löschen von Daten via GUI haben wir ganz verzichtet um die Nachvollziehbarkeit und Datenintegrität zu gewährleisten. Aus diesem Grund wird das CRUD jeweils ohne das Delete aufgeführt.



Autor	Janik von Rotz
Ziel	Ersterfassung und Bearbeitung von Stammdaten (Grobübersicht)
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Ersterfassung müssen die zu erfassenden Stammdaten bekannt sein - Für die Verwaltung müssen die Stammdaten bereits erfasst sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Stammdaten sind erfolgreich im System erfasst
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - Stammdaten, welche für die Verwendung der Hauptfunktion nötig wären, sind im System nicht vorhanden
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter - Bauleiter - Kontaktadmin - Kontaktperson
Auslösendes Ereignis	<ul style="list-style-type: none"> - Stammdaten müssen erfasst oder bearbeitet werden - Eingabemaske für Stammdaten wird ausgewählt
Beschreibung	<ol style="list-style-type: none"> 1. Akteur öffnet Eingabemaske 2. Erfasst oder verändert die gewünschten Stammdaten

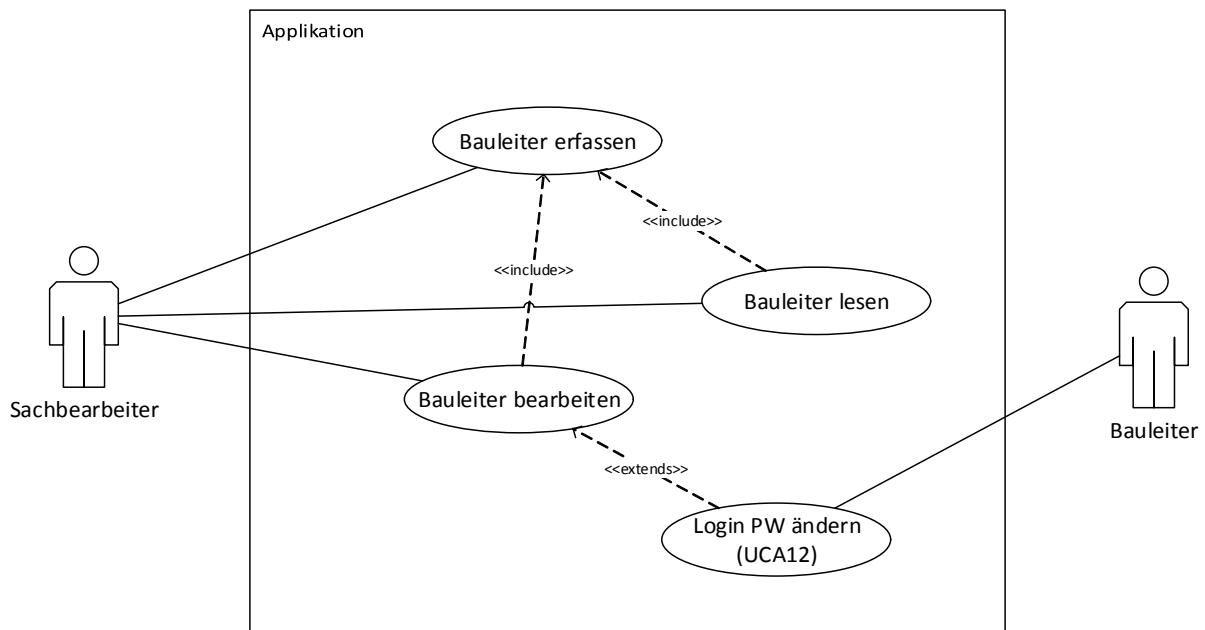
Erweiterung

UCA21-24

Alternativen

Keine

UCA21 – Bauleiter (CRU)



UCA21-1 – Bauleiter erfassen

Autor	Aathavan Theivendram
Ziel	Bauleiter erfassen
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Erfassung muss der Bauleiter bekannt sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Bauleiter ist erfolgreich im System erfasst
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - Bauleiter kann die für ihn bestimmten Aufgaben nicht wahrnehmen bzw. das Tool nicht benutzen.
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Bauleiter muss erfasst werden - Maske für die Erfassung von Bauleiter wird ausgewählt
Beschreibung	<ol style="list-style-type: none"> 1. Akteur öffnet Maske für Bauleiter 2. Erfasst einen neuen Bauleiter im System

Erweiterung	Keine
-------------	-------

Alternativen	Keine
--------------	-------

UCA21-2 – Bauleiter lesen

Autor	Aathavan Theivendram
-------	----------------------

Ziel	Bauleiter lesen
------	-----------------

Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für das Lesen müssen die Bauleiter bereits erfasst sein
----------------	---

Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Bauleiter kann erfolgreich gelesen werden
---------------------------	---

Nachbedingungen Fehl Schlag	<ul style="list-style-type: none"> - Bauleiter stehen für die korrekte Benutzung der Applikation nicht zur Verfügung
--------------------------------	---

Akteure	<ul style="list-style-type: none"> - Sachbearbeiter
---------	--

Auslösendes Ereignis	<ul style="list-style-type: none"> - Bauleiter muss gelesen werden
-------------------------	---

Beschreibung	<ol style="list-style-type: none"> 1. Akteur öffnet eine entsprechende Maske, in welchem der Bauleiter angezeigt wird 2. Liest einen Bauleiter
--------------	--

Erweiterung	Keine
-------------	-------

Alternativen	Keine
--------------	-------

UCA21-3 – Bauleiter bearbeiten

Autor	Aathavan Theivendram
-------	----------------------

Ziel	Bauleiter bearbeiten
------	----------------------

Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Bearbeitung müssen die Bauleiter bereits erfasst sein
----------------	---

Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Bauleiter ist erfolgreich aktualisiert
---------------------------	--

Nachbedingungen Fehl Schlag	<ul style="list-style-type: none"> - Bauleiter stehen für die korrekte Benutzung der Applikation nicht zur Verfügung.
--------------------------------	--

Akteure	<ul style="list-style-type: none"> - Sachbearbeiter
---------	--

Auslösendes Ereignis	<ul style="list-style-type: none"> - Bauleiter muss bearbeitet werden - Maske für die Bearbeitung von Bauleiter wird ausgewählt
-------------------------	---

Beschreibung	<ol style="list-style-type: none"> 1. Akteur öffnet Maske für Bauleiter 2. Bearbeitet einen Bauleiter
--------------	---

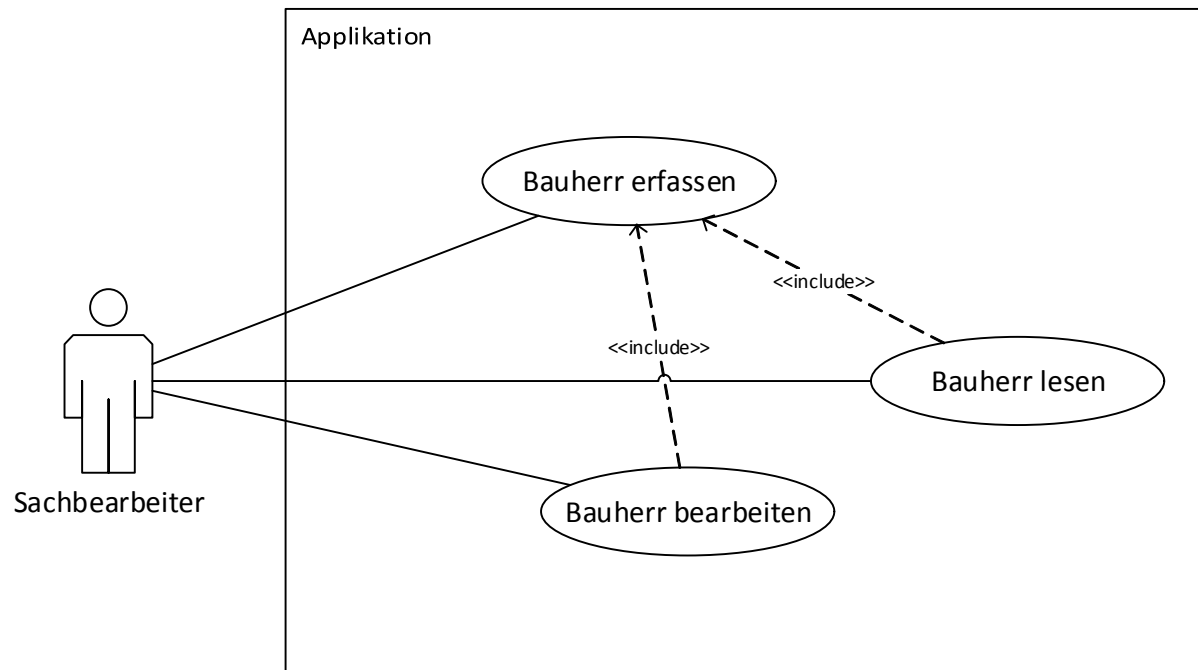
Erweiterung

UCA12

Alternativen

Keine

UCA22 – Bauherr (CRU)



UCA22-1 – Bauherr erfassen

Autor	Aathavan Theivendram
Ziel	Bauherr erfassen
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Ersterfassung muss der Bauherr bekannt sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Bauherr ist erfolgreich im System erfasst.
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - Projekte können dem Bauherren nicht zugewiesen werden.
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Bauherr muss erfasst werden - Maske für die Erfassung von Bauherren wird ausgewählt
Beschreibung	<ol style="list-style-type: none"> 1. Akteur öffnet Maske für Bauherren 2. Erfasst einen neuen Bauherr im System

Erweiterung	Keine
Alternativen	Keine

UCA22-2 – Bauherr lesen

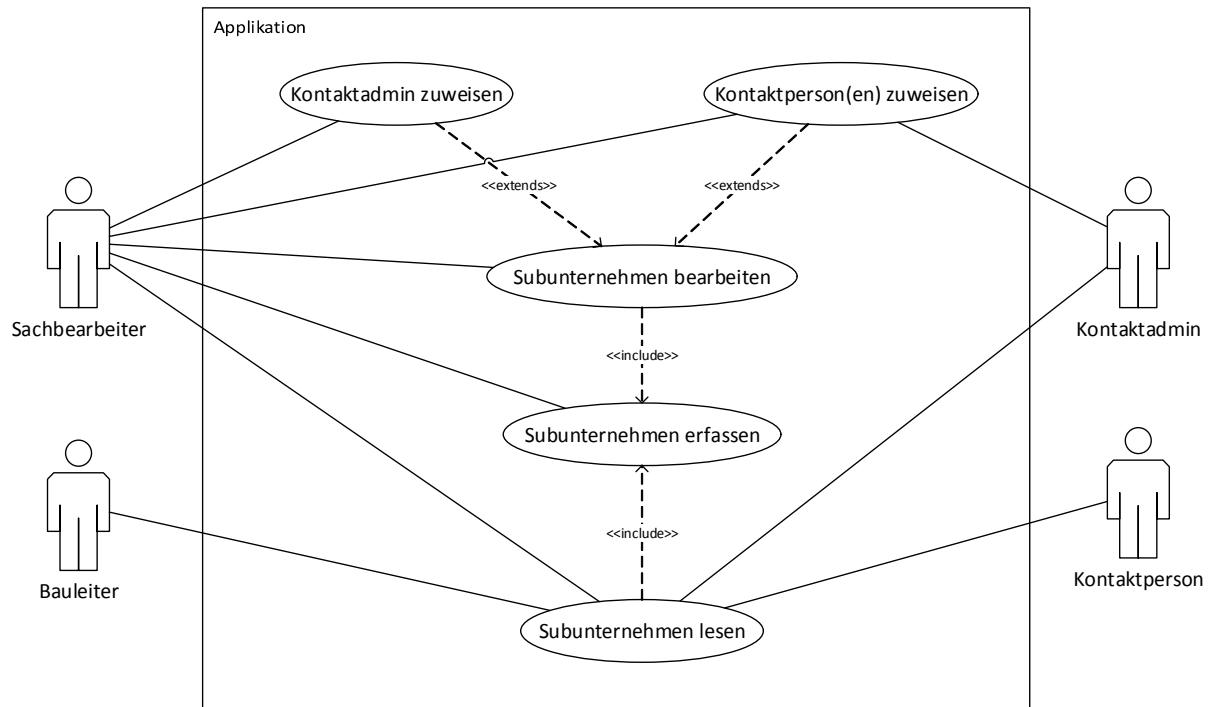
Autor	Aathavan Theivendram
Ziel	Bauherr lesen
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für das Lesen müssen die Bauherren bereits erfasst sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Bauherr kann erfolgreich gelesen werden
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - Im Projekt ist der zuständige Bauherr nicht sichtbar
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Bauherr muss gelesen werden
Beschreibung	<ol style="list-style-type: none"> 1. Akteur öffnet eine entsprechende Maske, in welchem der Bauherr angezeigt wird 2. Liest einen Bauherr
Erweiterung	Keine
Alternativen	Keine

UCA22-3 – Bauherr bearbeiten

Autor	Aathavan Theivendram
Ziel	Bauherr bearbeiten
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Bearbeitung müssen die Bauherren bereits erfasst sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Bauherr ist erfolgreich aktualisiert
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - Bauherr eines Projektes wird nicht korrekt oder gar nicht angezeigt.
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Bauherr muss bearbeitet werden - Maske für die Bearbeitung von Bauherren wird ausgewählt

Beschreibung	1. Akteur öffnet Maske für Bauherren 2. Bearbeitet einen Bauherr
Erweiterung	Keine
Alternativen	Keine

UCA23 – Subunternehmen (CRU)



UCA23-1 – Subunternehmen erfassen

Autor	Aathavan Theivendram
Ziel	Subunternehmen erfassen
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Ersterfassung muss das Subunternehmen bekannt sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Subunternehmen ist erfolgreich im System erfasst
Nachbedingungen Fehlslag	<ul style="list-style-type: none"> - Projekte und Mängel können nicht mehr einem Subunternehmen zugewiesen werden
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Subunternehmen muss erfasst werden - Maske für die Erfassung von Subunternehmen wird ausgewählt

Beschreibung	1. Akteur öffnet Maske für Subunternehmen 2. Erfasst ein neues Subunternehmen im System
Erweiterung	Keine
Alternativen	Keine

UCA23-2 – Subunternehmen lesen

Autor	Aathavan Theivendram
Ziel	Subunternehmen lesen
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für das Lesen müssen die Subunternehmen bereits erfasst sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Subunternehmen kann erfolgreich gelesen werden
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - Es ist nicht sichtbar welches Subunternehmen mit einem Projekt oder Mangel verknüpft ist.
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter - Bauleiter - Kontaktadmin - Kontaktperson
Auslösendes Ereignis	<ul style="list-style-type: none"> - Subunternehmen muss gelesen werden
Beschreibung	<ol style="list-style-type: none"> 1. Akteur öffnet eine entsprechende Maske, in welchem das Subunternehmen angezeigt wird. 2. Liest ein Subunternehmen
Erweiterung	Keine
Alternativen	Keine

UCA23-3 – Subunternehmen bearbeiten

Autor	Aathavan Theivendram
Ziel	Subunternehmen bearbeiten
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Bearbeitung müssen die Subunternehmen bereits erfasst sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Subunternehmen ist erfolgreich aktualisiert
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - Projekte und Mängel können nicht mehr einem Subunternehmen korrekt zugewiesen werden

Akteure	- Sachbearbeiter
Auslösendes Ereignis	- Subunternehmen muss bearbeitet werden - Maske für die Bearbeitung von Subunternehmen wird ausgewählt
Beschreibung	1. Akteur öffnet Maske für Subunternehmen 2. Bearbeitet ein Subunternehmen
Erweiterung	UCA23-4, UCA23-5
Alternativen	Keine

UCA23-4 – Kontaktadmin zuweisen

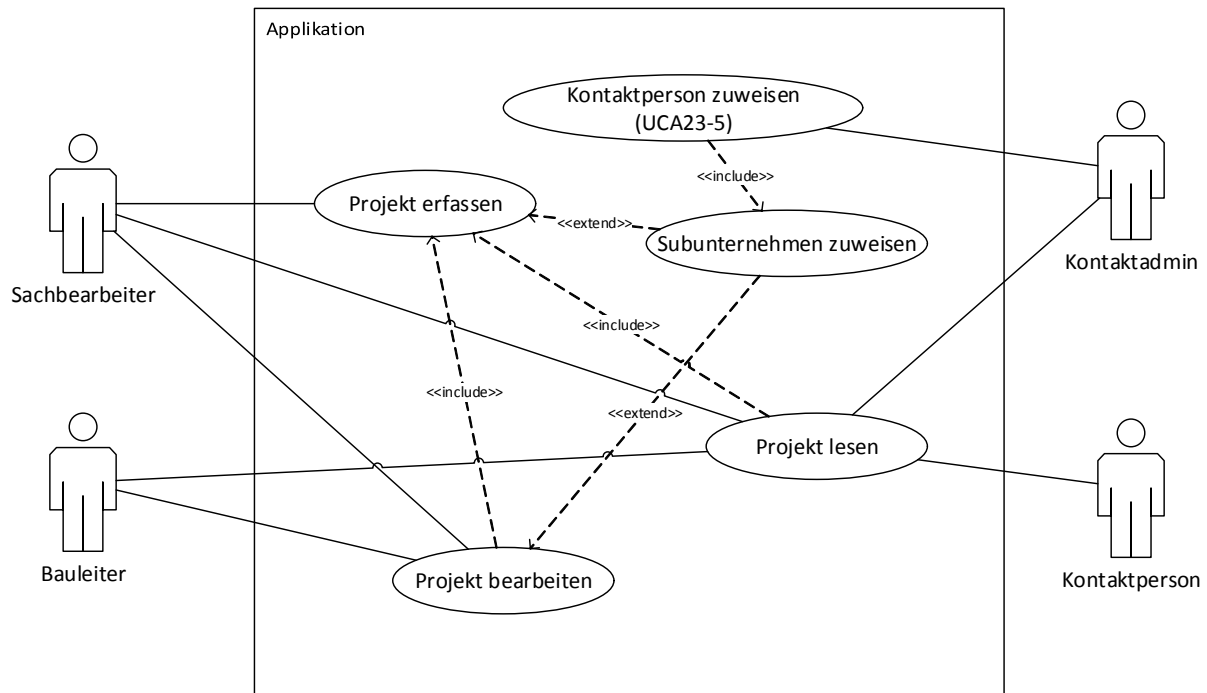
Autor	Aathavan Theivendram
Ziel	Kontaktadmin einem Subunternehmen zuweisen
Vorbedingungen	- Akteur ist authentifiziert (UCA10) - Für die Zuweisung müssen die Kontakte und das Subunternehmen bereits erfasst sein
Nachbedingungen Erfolg	- Kontaktadmin ist erfolgreich einem Subunternehmen zugewiesen
Nachbedingungen Fehlschlag	- Subunternehmen besitzt keinen Kontaktadmin
Akteure	- Sachbearbeiter
Auslösendes Ereignis	- Kontaktadmin muss einem Subunternehmen zugewiesen werden - Maske für die Zuweisung von Kontaktadmin auf Subunternehmen wird ausgewählt
Beschreibung	1. Akteur öffnet Maske für Subunternehmen 2. Fügt einen Kontaktadmin hinzu
Erweiterung	Keine
Alternativen	Keine

UCA23-5 – Kontaktperson zuweisen

Autor	Aathavan Theivendram
Ziel	Einen oder mehrere Kontaktpersonen einem Subunternehmen zuweisen
Vorbedingungen	- Akteur ist authentifiziert (UCA10) - Für die Zuweisung müssen die Kontakte und das Subunternehmen bereits erfasst sein
Nachbedingungen	- Kontaktperson ist erfolgreich einem Subunternehmen zugewiesen

Erfolg	
Nachbedingungen Fehlschlag	- Subunternehmen besitzt keinen Kontaktperson(en)
Akteure	- Sachbearbeiter - Kontaktadmin
Auslösendes Ereignis	- Kontaktperson muss einem Subunternehmen zugewiesen werden - Maske für die Zuweisung von Kontaktperson(en) auf Subunternehmen wird ausgewählt
Beschreibung	1. Akteur öffnet Maske für Subunternehmen 2. Fügt einen oder mehrere Kontaktpersonen hinzu
Erweiterung	Keine
Alternativen	Keine

UCA24 – Bauprojekt (CRU)



UCA24-1 – Bauprojekt erfassen

Autor	Aathavan Theivendram
Ziel	Bauprojekt erfassen
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Ersterfassung muss das Bauprojekt bekannt sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Bauprojekt ist erfolgreich im System erfasst
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - Mängel können nicht mehr einem Projekt zugewiesen werden
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Bauprojekt muss erfasst - Maske für die Erfassung von Bauprojekten wird ausgewählt
Beschreibung	<ol style="list-style-type: none"> 1. Akteur öffnet Maske für Bauprojekt 2. Erfasst ein neues Bauprojekt im System
Erweiterung	UCA24-4
Alternativen	Keine

UCA24-2 – Bauprojekt lesen

Autor	Aathavan Theivendram
Ziel	Bauprojekt lesen
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für das Lesen müssen die Bauprojekte bereits erfasst sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Bauprojekt kann erfolgreich gelesen werden
Nachbedingungen Fehl Schlag	<ul style="list-style-type: none"> - Projekte können nicht mehr gelesen werden - Mängel können nicht mehr einem Projekt zugewiesen werden
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter - Bauleiter - Kontaktadmin - Kontaktperson
Auslösendes Ereignis	<ul style="list-style-type: none"> - Bauprojekt muss gelesen werden - Maske für Bauprojekt wird ausgewählt
Beschreibung	<ol style="list-style-type: none"> 1. Akteur öffnet Maske für Bauprojekt 2. Liest ein Bauprojekt
Erweiterung	Keine
Alternativen	Keine

UCA24-3 – Bauprojekt bearbeiten

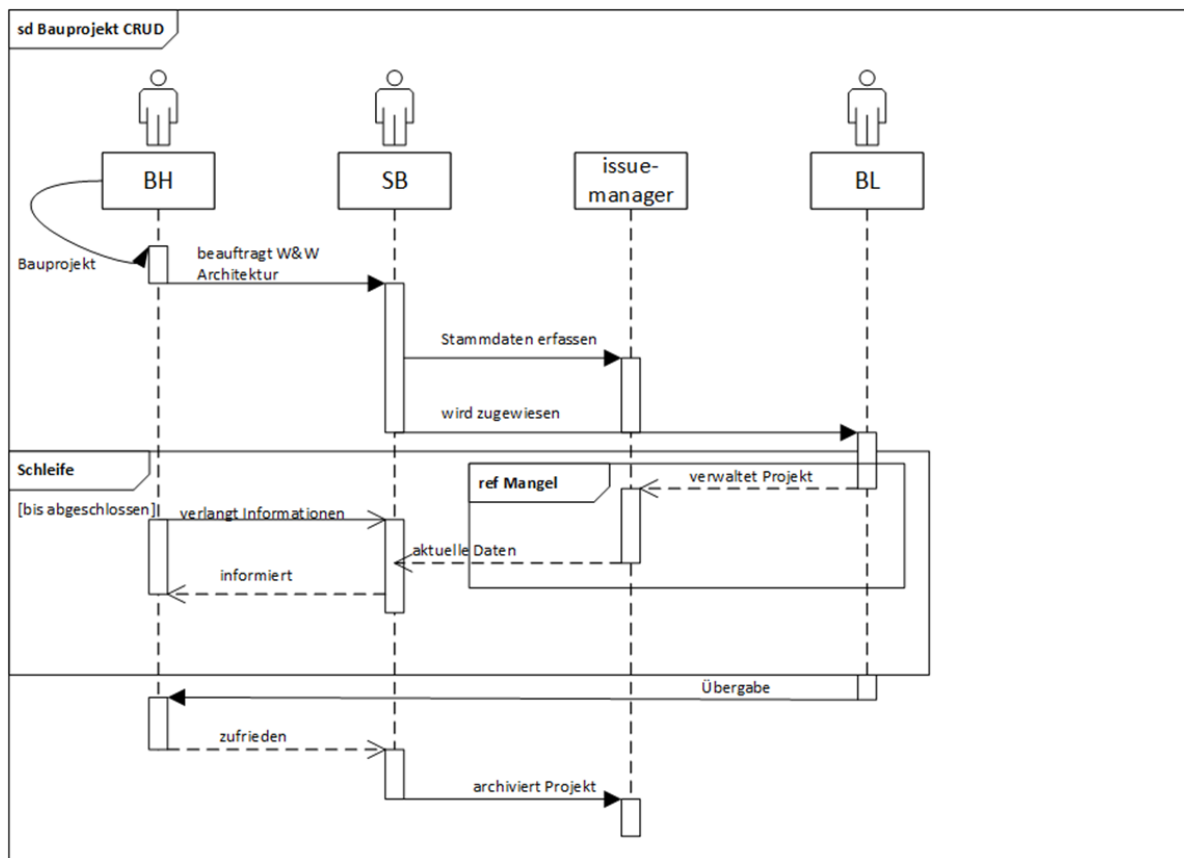
Autor	Aathavan Theivendram
Ziel	Bauprojekt bearbeiten
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Bearbeitung müssen die Bauprojekte bereits erfasst sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Bauprojekt ist erfolgreich aktualisiert
Nachbedingungen Fehl Schlag	<ul style="list-style-type: none"> - Projekte stehen für die korrekte Benutzung der Applikation nicht zur Verfügung.
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter - Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Bauprojekt muss bearbeitet werden - Maske für die Bearbeitung von Bauprojekten wird ausgewählt

Beschreibung	1. Akteur öffnet Maske für Bauprojekt 2. Bearbeitet ein Bauprojekt
Erweiterung	Keine
Alternativen	Keine

UCA24-4 – Subunternehmen zuweisen

Autor	Aathavan Theivendram
Ziel	Subunternehmen für ein Bauprojekt zuweisen
Vorbedingungen	<ul style="list-style-type: none"> - Akteur ist authentifiziert (UCA10) - Für die Zuweisung muss Subunternehmen und Bauprojekt bereits erfasst sein
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - Subunternehmen ist erfolgreich einem Bauprojekt zugewiesen
Nachbedingungen Fehl Schlag	<ul style="list-style-type: none"> - Bauprojekt besitzt keine involvierte Subunternehmen
Akteure	<ul style="list-style-type: none"> - Sachbearbeiter - Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Subunternehmen muss einem Bauprojekt zugewiesen werden - Maske für die Zuweisung von Subunternehmen auf Bauprojekte wird ausgewählt
Beschreibung	3. Akteur öffnet Maske für die Zuweisung 4. Fügt ein Subunternehmen hinzu
Erweiterung	Keine
Alternativen	Keine

ADI24 – Bauprojekt



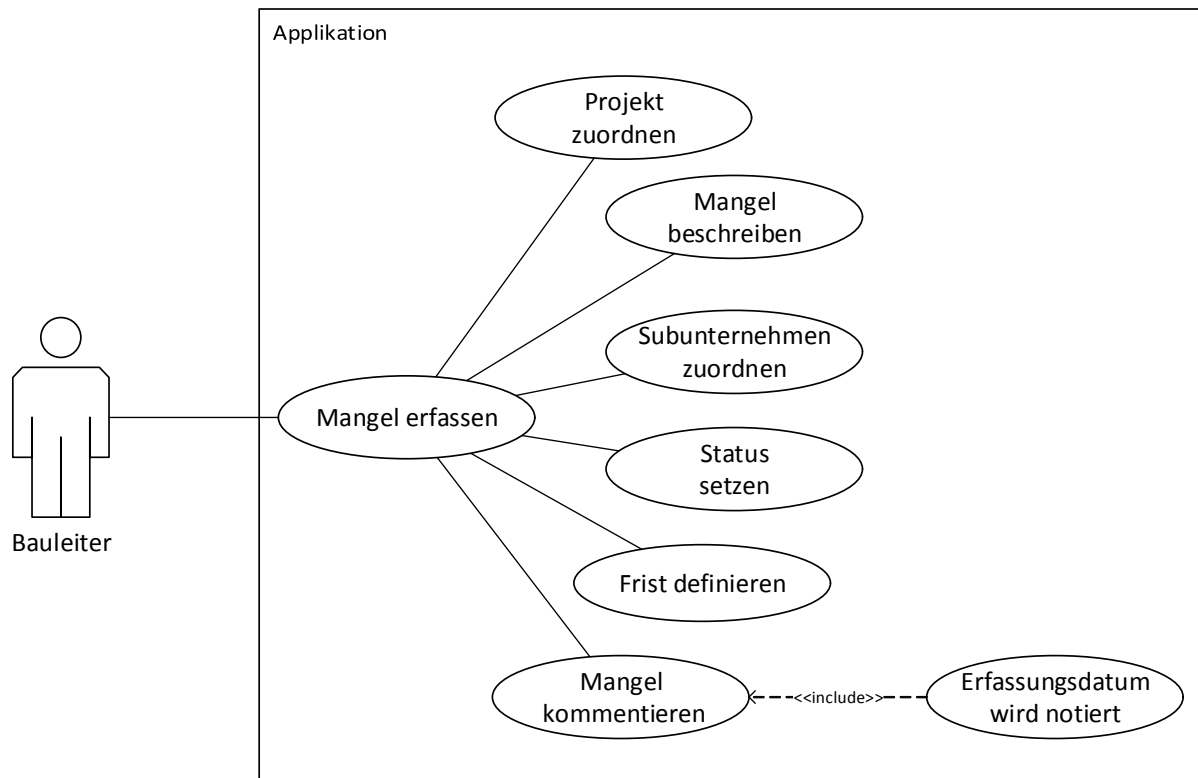
Autor

Sandro Klarer

Ziel

Erfassung und Verwaltung eines Bauprojekts

UCA30 – Mängelerfassung



UCA31 – Mangel erfassen

Autor	Sandro Klarer
Ziel	Erfassen eines Mangels auf einem Mobile Device vor Ort
Vorbedingungen	<ul style="list-style-type: none"> - User ist authentifiziert (UCA10) - User hat die notwendige Rollen Rechte um einen Mangel zu erfassen - Notwendige Stammdaten sind erfasst worden (UCA 20)
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - BL kann Mangel erfassen - BL kann den erfassten Mangel abspeichern - Mangel wird erfolgreich in der Datenbank gespeichert und steht für die Bearbeitung zur Verfügung (UAC40)
Nachbedingungen Fehlslag	<ul style="list-style-type: none"> - BL kann keinen Mangel erfassen - BL kann den Mangel nicht in der Datenbank speichern, die Informationen gehen verloren und die Daten stehen für die weitere Bearbeitung nicht zur Verfügung.

Akteure	- Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Systemtechnisch keines - Workflow: BL inspiziert die beauftragen Arbeiten, welche durch Subunternehmen durchgeführt wurden, auf die Qualität und gestellten Anforderungen
Beschreibung	<ol style="list-style-type: none"> 1. Der Bauleiter meldet sich in der Applikation an 2. Der Bauleiter wählt den entsprechenden Menüpunkt an 3. Der Mangel View wird geladen und steht zur Bearbeitung zur Verfügung
Erweiterung	Keine
Alternativen	Keine

UCA32 – Projekt zuordnen

Autor	Sandro Klarer
Ziel	Den Mangel betreffendes Projekt zuordnen
Vorbedingungen	<ul style="list-style-type: none"> - User ist authentifiziert (UCA10) - User hat die notwendige Rollen Rechte um einen Mangel zu erfassen und zu bearbeiten - Notwendige Stammdaten sind erfasst worden (UCA 20)
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - BL kann Mangel einem Projekt zuordnen - BL kann den erfassten Zuordnung abspeichern - Änderungen werden erfolgreich in der Datenbank gespeichert und stehen für die Bearbeitung zur Verfügung (UAC40)
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - BL kann den Mangel kein Projekt zuordnen - BL kann die Zuordnung nicht in der Datenbank speichern, die Informationen gehen verloren und die Daten stehen für die weitere Bearbeitung nicht zur Verfügung
Akteure	- Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Mangel wird erfasst - Workflow: BL entdeckt einen Mangel, welchen er dem entsprechenden Projekt zuordnet, damit er eine Übersicht über die zuständigen Subunternehmen erhält.
Beschreibung	<ol style="list-style-type: none"> 1. Der Bauleiter meldet sich in der Applikation an. 2. Der Bauleiter wählt den entsprechenden Menüpunkt an. 3. Der Mangel View wird geladen und steht zur Bearbeitung zur Verfügung. 4. Entsprechendes Feld wird ausgefüllt, geändert.
Erweiterung	Keine

Alternativen	Keine
--------------	-------

UCA33 – Mangel beschreiben

Autor	Sandro Klarer
Ziel	Den festgestellten Mangel kurz beschreiben
Vorbedingungen	<ul style="list-style-type: none"> - User ist authentifiziert (UCA10) - User hat die notwendige Rollen Rechte um einen Mangel zu erfassen und zu bearbeiten - Notwendige Stammdaten sind erfasst worden (UCA 20)
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - BL kann Mangel beschreiben - BL kann den erfassten Beschrieb abspeichern - Änderungen werden erfolgreich in der Datenbank gespeichert und stehen für die Bearbeitung zur Verfügung (UAC40)
Nachbedingungen Fehlslag	<ul style="list-style-type: none"> - BL kann den Mangel nicht beschreiben - BL kann die Ergänzungen nicht in der Datenbank speichern, die Informationen gehen verloren und die Daten stehen für die weitere Bearbeitung nicht zur Verfügung
Akteure	<ul style="list-style-type: none"> - Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Mangel wird erfasst - Workflow: BL entdeckt einen Mangel, welchen er kurz beschreibt, sodass das zuständige Subunternehmen weiss, was zu korrigieren ist
Beschreibung	<ol style="list-style-type: none"> 1. Der Bauleiter meldet sich in der Applikation an. 2. Der Bauleiter wählt den entsprechenden Menüpunkt an. 3. Der Mangel View wird geladen und steht zur Bearbeitung zur Verfügung. 4. Entsprechendes Feld wird ausgefüllt, geändert.
Erweiterung	Keine
Alternativen	Keine

UCA34 – Betreffendes Subunternehmen zuordnen

Autor	Sandro Klarer
Ziel	Den Mangel betreffendes Subunternehmen zuordnen
Vorbedingungen	<ul style="list-style-type: none"> - User ist authentifiziert (UCA10) - User hat die notwendige Rollen Rechte um einen Mangel zu erfassen und zu bearbeiten - Notwendige Stammdaten sind erfasst worden (UCA 20)

Nachbedingungen Erfolg	<ul style="list-style-type: none"> - BL kann Mangel dem zuständigen Subunternehmen zuordnen - BL kann den erfassten Zuordnung abspeichern - Änderungen werden erfolgreich in der Datenbank gespeichert und stehen für die Bearbeitung zur Verfügung (UAC40)
Nachbedingungen Fehlslag	<ul style="list-style-type: none"> - BL kann den Mangel kein Subunternehmen zuordnen - BL kann die Zuordnung nicht in der Datenbank speichern, die Informationen gehen verloren und die Daten stehen für die weitere Bearbeitung nicht zur Verfügung
Akteure	<ul style="list-style-type: none"> - Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Mangel wird erfasst - Workflow: BL entdeckt einen Mangel, welchen er dem zuständigen Subunternehmen zuordnet, damit dieses den Mangel behebt.
Beschreibung	<ol style="list-style-type: none"> 1. Der Bauleiter meldet sich in der Applikation an. 2. Der Bauleiter wählt den entsprechenden Menüpunkt an. 3. Der Mangel View wird geladen und steht zur Bearbeitung zur Verfügung. 4. Entsprechendes Feld wird ausgefüllt, geändert.
Erweiterung	Keine
Alternativen	Keine

UCA35 – Status setzen

Autor	Sandro Klarer
Ziel	Status setzen
Vorbedingungen	<ul style="list-style-type: none"> - User ist authentifiziert (UCA10) - User hat die notwendige Rollen Rechte um einen Mangel zu erfassen und zu bearbeiten - Notwendige Stammdaten sind erfasst worden (UCA 20)
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - BL kann den Mangel einen Status setzen. - BL kann die erfasste Änderung abspeichern - Änderungen werden erfolgreich in der Datenbank gespeichert und stehen für die Bearbeitung zur Verfügung (UAC40)
Nachbedingungen Fehlslag	<ul style="list-style-type: none"> - BL kann keinen Status vergeben - BL kann Änderung nicht in der Datenbank speichern, die Informationen gehen verloren und die Daten stehen für die weitere Bearbeitung nicht zur Verfügung
Akteure	<ul style="list-style-type: none"> - Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Mangel wird erfasst - Workflow: BL entdeckt einen Mangel, erfasst diesen, ordnet ihn einem

	Projekt und Subunternehmen zu und setzt einen Status, sodass das Subunternehmen diesen erkennt und bearbeitet
Beschreibung	<ol style="list-style-type: none"> 1. Der Bauleiter meldet sich in der Applikation an. 2. Der Bauleiter wählt den entsprechenden Menüpunkt an. 3. Der Mangel View wird geladen und steht zur Bearbeitung zur Verfügung. 4. Entsprechendes Feld wird ausgefüllt, geändert.
Erweiterung	Keine
Alternativen	Keine

UCA36 – Frist definieren

Autor	Sandro Klarer
Ziel	Den Mangel betreffendes Projekt zuordnen
Vorbedingungen	<ul style="list-style-type: none"> - User ist authentifiziert (UCA10) - User hat die notwendige Rollen Rechte um einen Mangel zu erfassen und zu bearbeiten - Notwendige Stammdaten sind erfasst worden (UCA 20)
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - BL kann eine Frist definieren - BL kann die erfasste Frist abspeichern - Änderungen werden erfolgreich in der Datenbank gespeichert und stehen für die Bearbeitung zur Verfügung (UAC40)
Nachbedingungen Fehlslag	<ul style="list-style-type: none"> - BL kann keine Frist definieren - BL kann die Änderung nicht in der Datenbank speichern, die Informationen gehen verloren und die Daten stehen für die weitere Bearbeitung nicht zur Verfügung
Akteure	<ul style="list-style-type: none"> - Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Mangel erfassen - Workflow: BL entdeckt einen Mangel, welchen er beschreibt, zuordnet und definiert, bis wann er behoben sein muss
Beschreibung	<ol style="list-style-type: none"> 1. Der Bauleiter meldet sich in der Applikation an. 2. Der Bauleiter wählt den entsprechenden Menüpunkt an. 3. Der Mangel View wird geladen und steht zur Bearbeitung zur Verfügung. 4. Entsprechendes Feld wird ausgefüllt, geändert.
Erweiterung	Keine
Alternativen	Keine

UCA37 – Mangel kommentieren

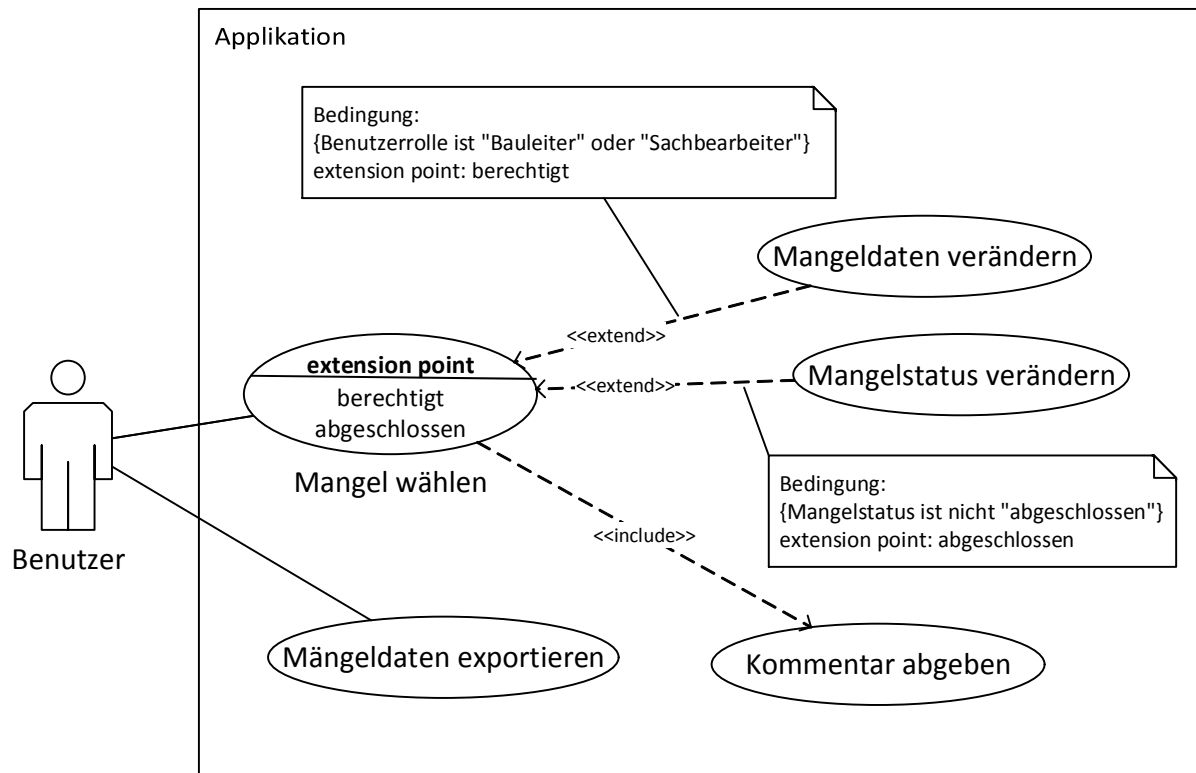
Autor	Sandro Klarer
Ziel	Den Mangel kommentieren könne, falls dies notwendig ist und der Beschrieb nicht ausreicht.
Vorbedingungen	<ul style="list-style-type: none"> - User ist authentifiziert (UCA10) - User hat die notwendige Rollen Rechte um einen Mangel zu erfassen und zu bearbeiten - Notwendige Stammdaten sind erfasst worden (UCA 20)
Nachbedingungen Erfolg	<ul style="list-style-type: none"> - BL kann Mangel kommentieren - BL kann den Kommentar abspeichern - Änderungen werden erfolgreich in der Datenbank gespeichert und stehen für die Bearbeitung zur Verfügung (UAC40)
Nachbedingungen Fehlslag	<ul style="list-style-type: none"> - BL kann keine zusätzlichen Kommentare erfassen - BL kann die Kommentare nicht in der Datenbank speichern, die Informationen gehen verloren und die Daten stehen für die weitere Bearbeitung nicht zur Verfügung
Akteure	<ul style="list-style-type: none"> - Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Mangel erfassen - Workflow: BL entdeckt einen Mangel, welcher behoben werden muss. Der Beschrieb reicht nicht aus und/ oder es gibt Fragen, welche geklärt werden müssen
Beschreibung	<ol style="list-style-type: none"> 1. Der Bauleiter meldet sich in der Applikation an. 2. Der Bauleiter wählt den entsprechenden Menüpunkt an. 3. Der Mangel View wird geladen und steht zur Bearbeitung zur Verfügung. 4. Entsprechendes Feld wird ausgefüllt, geändert
Erweiterung	Keine
Alternativen	Keine

UCA38 – Erfassungsdatum wird notiert

Autor	Sandro Klarer
Ziel	Dem Kommentar muss automatisch ein Erfassungsdatum beigefügt werden.
Vorbedingungen	<ul style="list-style-type: none"> - User ist authentifiziert (UCA10) - User hat die notwendige Rollen Rechte um einen Mangel zu erfassen und zu bearbeiten - Notwendige Stammdaten sind erfasst worden (UCA 20)

Nachbedingungen Erfolg	<ul style="list-style-type: none"> - BL erfasst einen Kommentar und das Erfassungsdatum wird automatisch notiert - BL kann die Ergänzungen abspeichern - Änderungen werden erfolgreich in der Datenbank gespeichert und stehen für die Bearbeitung zur Verfügung (UAC40)
Nachbedingungen Fehlschlag	<ul style="list-style-type: none"> - Das Erfassungsdatum, wird bei dem Erfassen eines Kommentars, nicht automatisch generiert - BL kann die Ergänzungen nicht speichern, die Informationen gehen verloren und die Daten stehen für die weitere Bearbeitung nicht zur Verfügung
Akteure	<ul style="list-style-type: none"> - Bauleiter
Auslösendes Ereignis	<ul style="list-style-type: none"> - Mangel erfassen - Kommentar erfassen - Workflow: BL entdeckt einen Mangel, welchen er beschreibt. Die Beschreibung reicht nicht aus und/oder es gibt Fragen, welche zu klären sind. Diese werden durch die Kommentar Funktion festgehalten und ein Erfassungsdatum wird, für die Nachvollziehbarkeit, notiert.
Beschreibung	<ol style="list-style-type: none"> 1. Der Bauleiter meldet sich in der Applikation an. 2. Der Bauleiter wählt den entsprechenden Menüpunkt an. 3. Der Mangel View wird geladen und steht zur Bearbeitung zur Verfügung. 4. Entsprechendes Feld wird ausgefüllt, geändert
Erweiterung	Keine
Alternativen	Keine

UCA40 – Mangel wählen



Autor	Reno Meyer
Ziel	Wählen eines existierenden Mangels
Vorbedingungen	<ul style="list-style-type: none"> - KP ist authentifiziert (UCA10) - Mangel wurden erfasst (UCA30) - Mangel wird angezeigt (nach Projekt)
Akteure	<ul style="list-style-type: none"> - alle Benutzer
Auslösendes Ereignis	<ul style="list-style-type: none"> - Die Mangelübersicht für ein Projekt wird angewählt
Beschreibung	<p>10. Der Benutzer wird gemäss seiner Rolle auf seine Berechtigung überprüft</p> <p>11. Mängel werden angezeigt:</p> <ol style="list-style-type: none"> Bauleiter und Sachbearbeiter: alle Kontaktperson und Kontaktadmin: alle die dem ihrem Subunternehmen zugeteilt sind <p>12. Der Benutzer wählt einen Mangel an um ihn zu bearbeiten</p> <p>13. Mangelstatus wird überprüft</p>

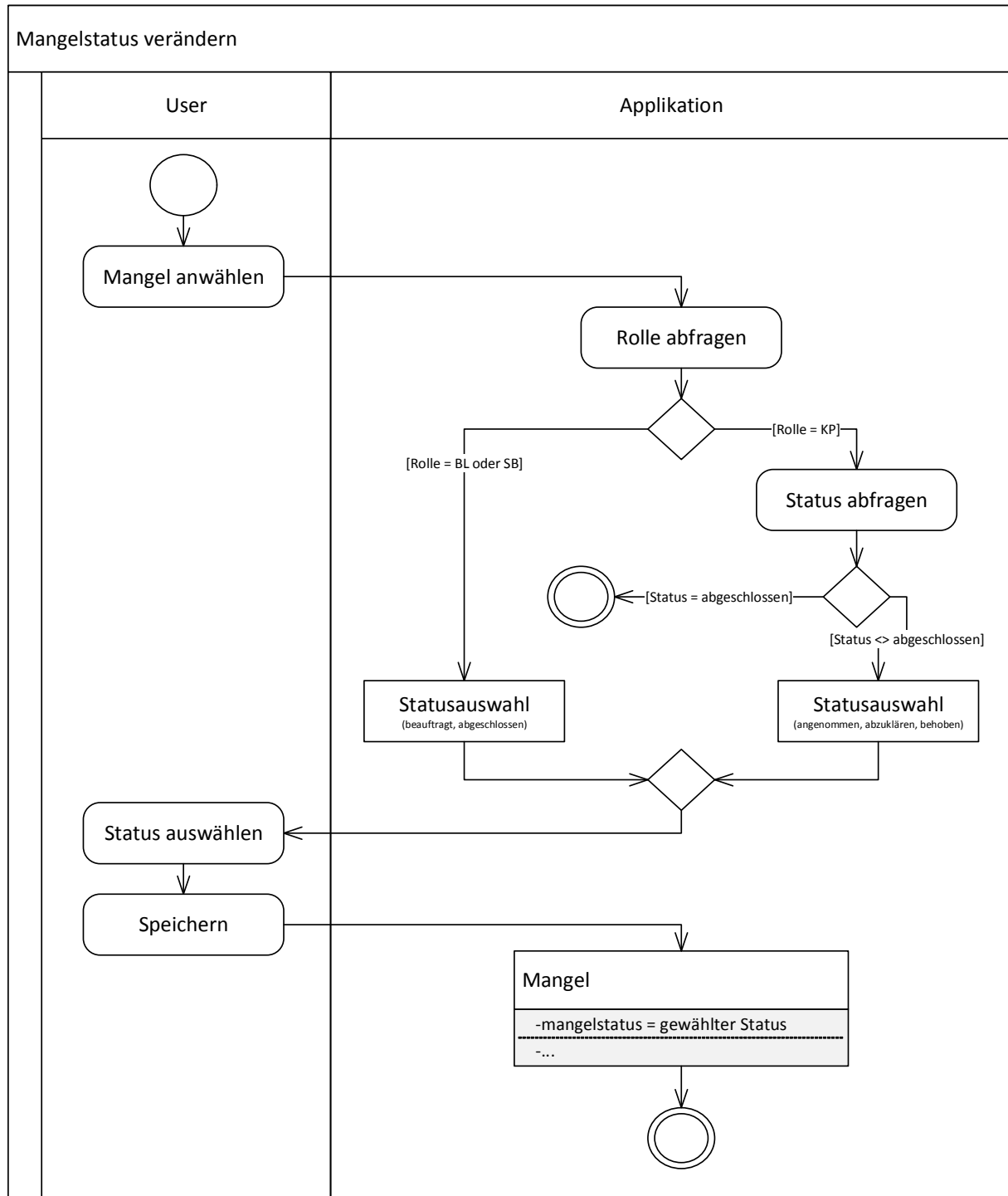
UCA41 – Mangeldaten verändern

Autor	Reno Meyer
Ziel	Verändern von Mangeldaten
Vorbedingungen	- Benutzerrolle wurde überprüft (UCA41)
Akteure	- Bauleiter - Sachbearbeiter
Auslösendes Ereignis	- Mangel wird angewählt
Beschreibung	14. Mangeldaten (Beschreibung, Subunternehmen, Frist) werden veränderbar 15. Der Benutzer kann Änderungen vornehmen 16. Der Benutzer kann die Änderungen speichern oder verwerfen 17. Der Benutzer wird zur Mängelübersicht zurückgeführt

UCA42 – Mangelstatus verändern

Autor	Reno Meyer
Ziel	Verändern des Mangelstatus
Vorbedingungen	- Benutzerrolle wurde überprüft (UCA41) - Mangelstatus wurde überprüft (UCA41)
Akteure	- alle Benutzer
Auslösendes Ereignis	- Mangel wird angewählt
Beschreibung	Für Bauleiter und Sachbearbeiter: 1) Mangelstatus veränderbar 2) Der Benutzer hat folgende Status zur Auswahl: a. „beauftragt“ b. „abgeschlossen“ 3) Der Benutzer kann die Änderungen speichern oder verwerfen 4) Der Benutzer wird zur Mängelübersicht zurückgeführt

ADI42 – Mangelstatus verändern



Autor

Reno Meyer

Ziel

Verändern des Mangelstatus

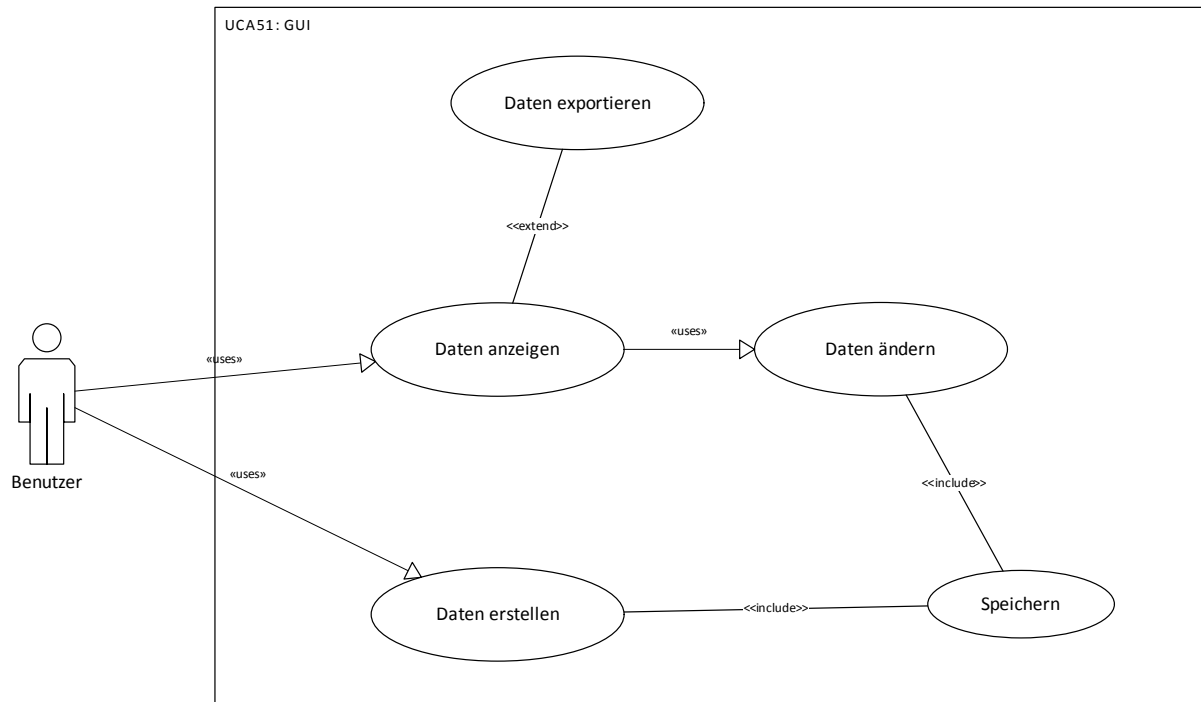
UCA43 – Kommentar abgeben

Autor	Reno Meyer
Ziel	Abgeben eines Kommentares
Vorbedingungen	- Mangel ausgewählt (UCA41)
Akteure	- alle Benutzer
Auslösendes Ereignis	- Mangel wird angewählt
Beschreibung	5) Der Benutzer kann beliebigen Text eingeben 6) Der Benutzer kann den Kommentar abschicken 7) Kommentarliste wird sofort aufgefrischt und zeigt (alle alten und) den neuen Kommentar

UCA44 – Mängeldaten exportieren

Autor	Reno Meyer
Ziel	Exportieren von Mängeldaten
Vorbedingungen	- Benutzer ist in der Mangelübersicht eines spezifischen Projektes
Akteure	- alle Benutzer
Auslösendes Ereignis	- Export Button wird angewählt
Beschreibung	1) Die Mängeldaten des angewählten Projektes werden in eine .csv Datei geschrieben 2) Der Benutzer wird aufgefordert die Datei zu benennen und zu speichern

UCA51 – GUI



Autor	Patrick Elsener
Ziel	Datenverwaltung mittels GUI
Vorbedingungen	- Applikation wurde korrekt gestartet
Nachbedingungen Erfolg	- Benutzeroberfläche wird angezeigt
Nachbedingungen Fehl Schlag	- Benutzeroberfläche wird nicht angezeigt
Akteure	- Sachbearbeiter, Bauleiter, Subunternehmen
Auslösendes Ereignis	- Systemtechnisch keines
Beschreibung	<ol style="list-style-type: none"> 1. Der Benutzer startet den Client 2. Das Login Fenster wird angezeigt 3. Die Hauptbenutzeroberfläche wird angezeigt
Erweiterung	Keine
Alternativen	Keine

UCA52 – Daten anzeigen

Autor	Patrick Elsener
Ziel	Daten im GUI anzeigen
Vorbedingungen	- Benutzer befindet sich im GUI
Nachbedingungen Erfolg	- Benutzer kann Projekte, Subunternehmen, Personen und Mängel im GUI anschauen
Nachbedingungen Fehlschlag	- Daten werden im GUI nicht angezeigt
Akteure	- Sachbearbeiter, Bauleiter, Subunternehmen
Auslösendes Ereignis	- Benutzer wählt ein Projekt, Subunternehmen, Person oder Mangel, welcher er angezeigt haben möchte
Beschreibung	1. Der Benutzer wählt die Daten, welcher er angezeigt haben möchte. 2. Die gewählten Daten werden angezeigt
Erweiterung	Keine
Alternativen	Keine

UCA53 – Daten exportieren

Autor	Patrick Elsener
Ziel	Daten mittels Export-Button exportieren
Vorbedingungen	- Benutzer befindet sich im GUI
Nachbedingungen Erfolg	- Benutzer kann Projekte, Subunternehmen, Personen und Mängel im GUI exportieren
Nachbedingungen Fehlschlag	- Export schlägt fehl
Akteure	- Sachbearbeiter, Bauleiter, Subunternehmen
Auslösendes Ereignis	- Benutzer wählt ein Projekt, Subunternehmen, Person oder Mangel, welcher er exportiert haben möchte
Beschreibung	1. Der Benutzer wählt die Daten, welcher er exportiert haben möchte. 2. Die gewählten Daten werden exportiert
Erweiterung	Keine
Alternativen	Keine

UCA54 – Daten ändern

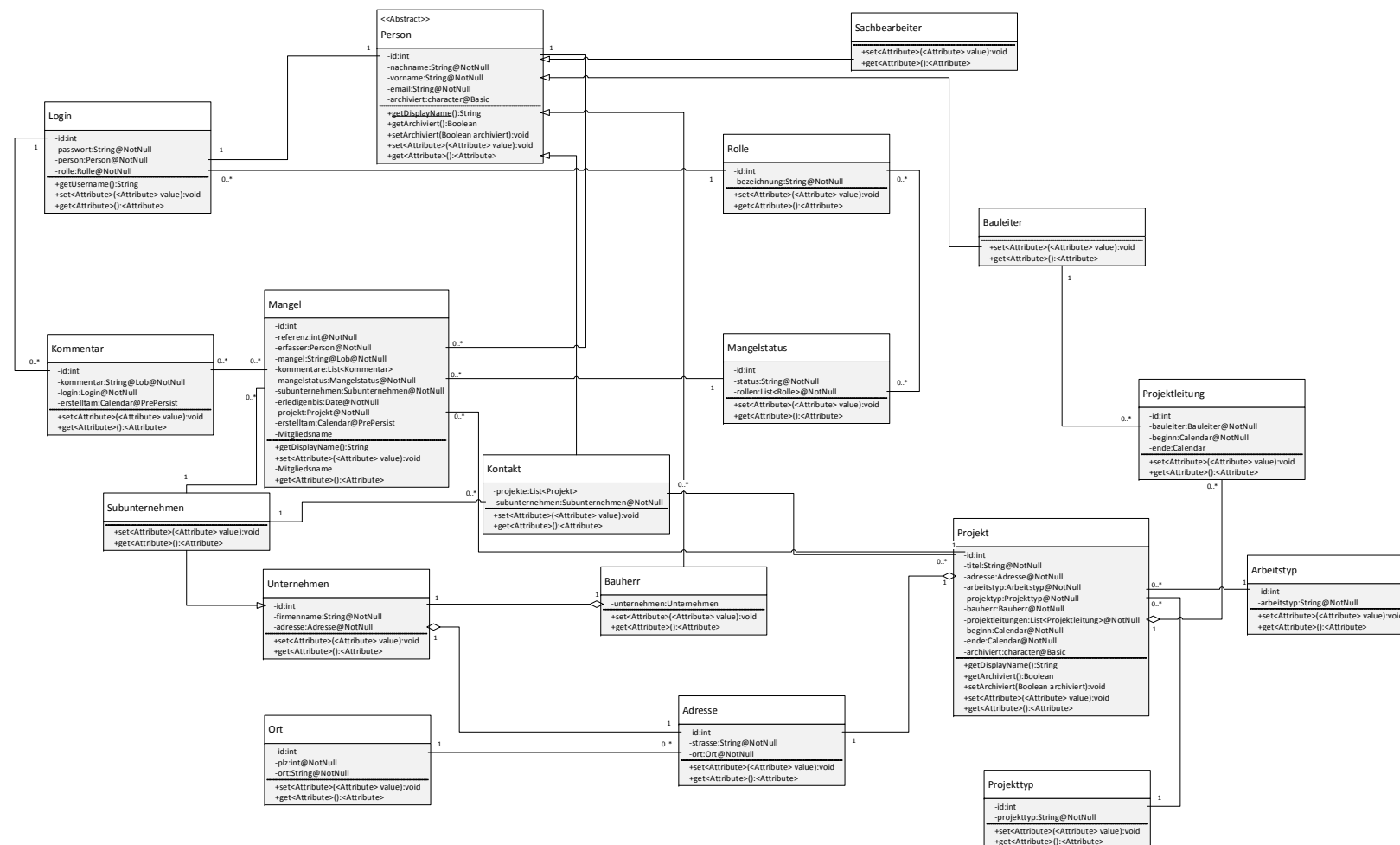
Autor	Patrick Elsener
Ziel	Daten ändern
Vorbedingungen	- Benutzer befindet sich im GUI
Nachbedingungen Erfolg	- Benutzer kann Projekte, Subunternehmen, Personen und Mängel im GUI ändern
Nachbedingungen Fehlslag	- Änderungen können nicht vorgenommen werden
Akteure	- Sachbearbeiter, Bauleiter, Subunternehmen
Auslösendes Ereignis	- Benutzer wählt ein Projekt, Subunternehmen, Person oder Mangel, welcher er geändert haben möchte
Beschreibung	1. Der Benutzer wählt die Daten, welcher er geändert haben möchte. 2. Die Änderungen werden vorgenommen
Erweiterung	Keine
Alternativen	Keine

UCA55 – Daten speichern

Autor	Patrick Elsener
Ziel	Neu erstellte oder geänderte Daten speichern
Vorbedingungen	- Benutzer befindet sich im GUI
Nachbedingungen Erfolg	- Benutzer kann neue oder geänderte Projekte, Subunternehmen, Personen und Mängel speichern
Nachbedingungen Fehlslag	- Daten werden nicht gespeichert
Akteure	- Sachbearbeiter, Bauleiter, Subunternehmen
Auslösendes Ereignis	- Benutzer wählt ein Projekt, Subunternehmen, Person oder Mangel, welches er speichern möchte
Beschreibung	1. Der Benutzer wählt die Daten, welche er geändert haben möchte oder erstellte neue Daten 2. Die Eingabemaske zur entsprechenden Auswahl wird aufgerufen 3. Die Daten werden eingegeben und erfasst/verändert
Erweiterung	Keine
Alternativen	Keine

Nachfolgend ein gesamthaftes Klassendiagramm der Entitätsklassen und die Beschreibungen der einzelnen verwendeten Klassen.

Gesamtklassendiagramm der Entität-Klassen (Elektronisch: auf CD Pfad: Dokumentation\IM_Klassenmodell.pdf)



Klassen im Detail:

Getter- und Setter-Methoden werden auf Grund ihrer Einfachheit nicht näher beschrieben. Bzw. nicht aufgeführt.

Klasse	Person
Diagramm	<pre> <<Abstract>> Person -id:int -nachname:String@NotNull -vorname:String@NotNull -email:String@NotNull -archiviert:character@Basic +getDisplayName():String +getArchiviert():Boolean +setArchiviert(Boolean archiviert):void +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	-
Zuständigkeit	Stellt die Entität Person dar. Ist eine Abstrakte Klasse
Wichtige Methoden	<ul style="list-style-type: none"> - getDisplayName - getArchiviert - setArchiviert
Autor	Janik Von Rotz

Klasse	Login
Diagramm	<pre> Login -id:int -passwort:String@NotNull -person:Person@NotNull -rolle:Rolle@NotNull +getUsername():String +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Login dar. Wickelt die Useranmeldungen ab
Wichtige Methoden	<ul style="list-style-type: none"> - getUsername
Autor	Sandro Klarer

Klasse	Sachbearbeiter
Diagramm	<p>Sachbearbeiter</p> <p>+set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute></p>
Erbt von	Person
Implementiert	-
Zuständigkeit	Stellt die Entität Sachbearbeiter dar. Wird als Zwischentabelle genutzt
Wichtige Methoden	
Autor	Erwin Willi

Klasse	Rolle
Diagramm	<p>Rolle</p> <p>-id:int -bezeichnung:String@NotNull +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute></p>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Rolle dar. Erstellt die Rollen in der Datenbank
Wichtige Methoden	
Autor	Erwin Willi

Klasse	Bauleiter
Diagramm	<p>Bauleiter</p> <pre> +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	Person
Implementiert	-
Zuständigkeit	Stellt die Entität Bauleiter dar. Speichert neue Bauleiter in der Datenbank
Wichtige Methoden	
Autor	Aathavan Theivendram

Klasse	Projekt
Diagramm	<p>Projekt</p> <pre> -id:int -titel:String@NotNull -adresse:Adresse@NotNull -arbeitstyp:Arbeitstyp@NotNull -projekttyp:Projekttyp@NotNull -bauherr:Bauherr@NotNull -projektleitungen:List<Projektleitung>@NotNull -beginn:Calendar@NotNull -ende:Calendar@NotNull -archiviert:character@Basic +getDisplayName():String +getArchiviert():Boolean +setArchiviert(Boolean archiviert):void +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Projekt dar. Erstellt neue Projekte in der Datenbank und setzt Attribut Archiviert auf false
Wichtige Methoden	<ul style="list-style-type: none"> - getDisplayName - getArchiviert - setArchiviert <ul style="list-style-type: none"> o Projekte werden nicht gelöscht sondern nur auf archiviert gesetzt.
Autor	Reno Meyer

Klasse	Projektleitung
Diagramm	<p>Projektleitung</p> <pre> -id:int -bauleiter:Bauleiter@NotNull -beginn:Calendar@NotNull -ende:Calendar +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Projektleitung dar. Weisst einem Projekt einen Projektleiter bzw. Bauleiter zu
Wichtige Methoden	
Autor	Aathavan Theivendram

Klasse	Projekttyp
Diagramm	<p>Projekttyp</p> <pre> -id:int -projekttyp:String@NotNull +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Projekttyp dar. Weisst einem Projekt einen Projekttyp zu (Einfamilienhaus etc.)
Wichtige Methoden	
Autor	Reno Meyer

Klasse	Kontakt
Diagramm	<p>Kontakt</p> <pre> -projekte:List<Projekt> -subunternehmen:Subunternehmen@NotNull +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	Person
Implementiert	-
Zuständigkeit	Stellt die Entität Kontakt dar. Weisst einem Projekt die Kontaktperson zu
Wichtige Methoden	
Autor	Patrick Elsener

Klasse	Adresse
Diagramm	<p>Adresse</p> <pre> -id:int -strasse:String@NotNull -ort:Ort@NotNull +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Adresse dar. Weisst einem Projekt oder Subunternehmen eine Adresse zu.
Wichtige Methoden	
Autor	Sandro Klarer

Klasse	Ort
Diagramm	<p>Ort</p> <pre> -id:int -plz:int@NotNull -ort:String@NotNull +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Ort dar. Erstellt eine Liste mit Orten aus csv-File und weist wird benötigt um der Adresse einen Ort zuzuweisen.
Wichtige Methoden	
Autor	Janik Von Rotz

Klasse	Unternehmen
Diagramm	<p>Unternehmen</p> <pre> -id:int -firmenname:String@NotNull -adresse:Adresse@NotNull +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Unternehmen dar. Wird benötigt um einem Unternehmen einen Namen sowie eine Adresse zuzuweisen.
Wichtige Methoden	
Autor	Patrick Elsener

Klasse	Bauherr
Diagramm	<p>Bauherr</p> <p>-unternehmen:Unternehmen +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute></p>
Erbt von	Person
Implementiert	-
Zuständigkeit	Stellt die Entität Bauherr dar. Einem Bauherr wird ein Unternehmen zugewiesen
Wichtige Methoden	
Autor	Aathavan Theivendram

Klasse	Mangel
Diagramm	<p>Mangel</p> <p>-id:int -referenz:int@NotNull -erfasser:Person@NotNull -mangel:String@Lob@NotNull -kommentare:List<Kommentar> -mangelstatus:Mangelstatus@NotNull -subunternehmen:Subunternehmen@NotNull -erledigenbis:Date@NotNull -projekt:Projekt@NotNull -erstelltam:Calendar@PrePersist -Mitgliedsname +getDisplayName():String +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute></p>
Erbt von	Person
Implementiert	-
Zuständigkeit	Stellt die Entität Mangel dar. Erstllt einen Mangel in der Datenbank mit allen benötigten Attributen.
Wichtige Methoden	- getDisplayName
Autor	Janik Von Rotz

Klasse	Mangelstatus
Diagramm	<p>Mangelstatus</p> <pre> -id:int -status:String@NotNull -rollen:List<Rolle>@NotNull +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Mangelstatus dar. Weisst einem Mangel einen Status zu (erledigt etc.) Sowie einer entsprechenden UserRolle, welche den Mangel geschrieben hat.
Wichtige Methoden	
Autor	Erwin Willi

Klasse	Subunternehmen
Diagramm	<p>Subunternehmen</p> <pre> +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	Unternehmen
Implementiert	-
Zuständigkeit	Stellt die Entität Subunternehmen dar. Wird zur Speicherung eines Subunternehmens benötigt.
Wichtige Methoden	
Autor	Patrick Elsener

Klasse	Kommentar
Diagramm	<p>Kommentar</p> <pre> -id:int -kommentar:String@Lob@NotNull -login:Login@NotNull -erstelltam:Calendar@PrePersist +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Kommentar dar. Wird für das erfassen eines Datums sowie dem Erstelldatum verwendet.
Wichtige Methoden	
Autor	Sandro Klarer

Klasse	Arbeitstyp
Diagramm	<p>Arbeitstyp</p> <pre> -id:int -arbeitstyp:String@NotNull +set<Attribute>(<Attribute> value):void +get<Attribute>():<Attribute> </pre>
Erbt von	-
Implementiert	Model
Zuständigkeit	Stellt die Entität Arbeitstyp dar. Wird für die Zuweisung eines Arbeitstyps verwendet (Renovation etc.)
Wichtige Methoden	
Autor	Reno Meyer

Klasse	Model
Diagramm	<<Interface>> Model +getId():String
Erbt von	Serializable
Implementiert	-
Zuständigkeit	Stellt die Entität Arbeitstyp dar. Wird für die Zuweisung eines Arbeitstyps verwendet (Renovation etc.)
Wichtige Methoden	getId
Autor	Janik Von Rotz

Die Model-Klasse wird hier nur der Vollständigkeit aufgeführt da Sie von diversen Klassen implementiert wird.

17 Deployment-Infos

Der Deployment-Prozess ist in diesem Fall sehr trivial.

17.1 Systemanforderungen

- MySQL oder PostgreSQL
 - Datenbank: issuemanager
 - Benutzer: issuemanager
 - Passwort: issuemanager
- Beliebiger Applikationsserver
 - Jetty
 - Tomcat
 - JBoss
 - etc.

17.2 Ausführung

- Entpacken der mysql.zip oder postgres.zip Datei.
- Umbenennen der webservice-x.x.x.war Datei zu webservice.war und kopieren ins webapps Verzeichnis des Webservice Applikationsserver
- Ausführen der start-webservice.bat Datei auf dem Applikationsserver.
 - Zum einmaligen Seeden der Daten kann auch start-seed.bat ausgeführt werden.
- Ausführen der start-client.bat Datei.
- Login:

Rolle	Username	Passwort
Sachbearbeiter	sb@im.ch	1
Bauleiter	bl@im.ch	1
Kontaktadmin	ka@im.ch	1
Kontaktperson	kp@im.ch	1

17.3 Konfiguration

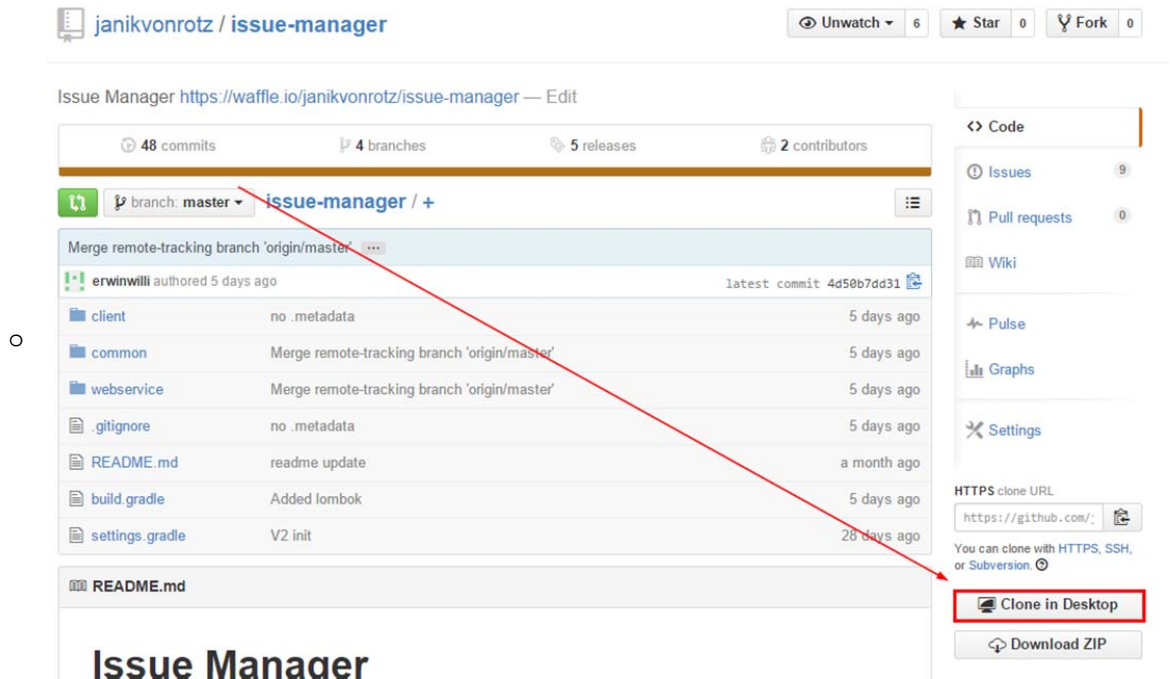
Alle Jar- und War-Dateien können über enthaltene application.json Datei konfiguriert werden.

17.4 Entwicklungsumgebung

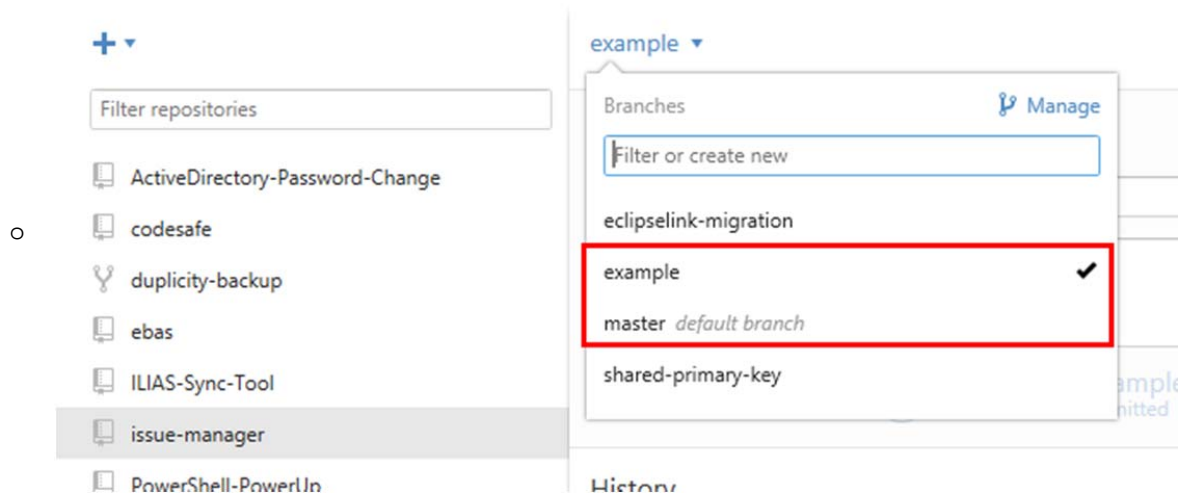
Die Entwicklungsumgebung bereitzustellen ist um einiges aufwendiger. Deshalb wird hier nur gezeigt, wie die Umgebung für die Beispielapplikation eingerichtet werden kann.

- Github for Windows installieren (GitHub Client)
 - <https://windows.github.com/>
- Einloggen in GitHub Client
- Einloggen auf GitHub Website

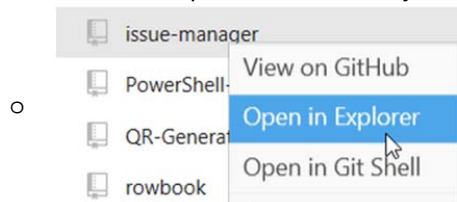
- Issue Manager Repository klonen
 - <https://github.com/janikvonrotz/issue-manager>



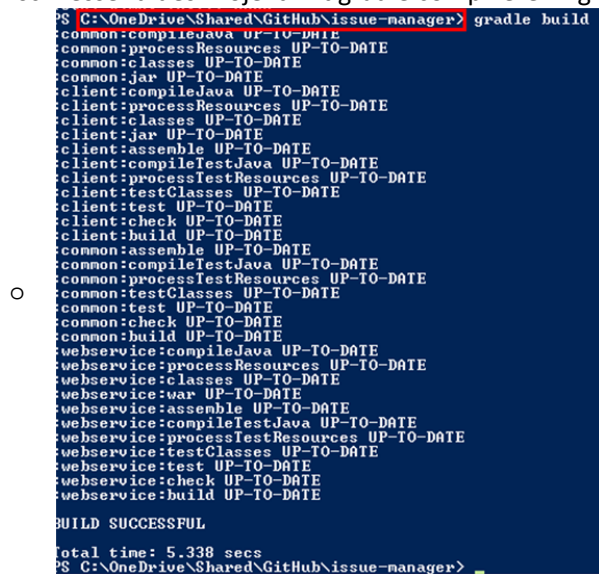
- Bitnami Webstack installieren
 - <https://bitnami.com/stack/nginx>
- Chocolatey installieren
 - <https://chocolatey.org/>
- Gradle installieren
 - Auf der Kommandozeile (PowerShell) folgendes eingeben: `choco install gradle`
- In GitHub Client den example Branch auswählen
 - **!** In einem GitHub Repository können mehrer Branches (Code Baum Äste) gespeichert werden. Diese Branches können weiter geteilt oder zusammengeführt werden. Das Beispielprojekt, dass in dieser Anleitung beschrieben wird findet man im "example"-Branche. Unser Projekt ist im Master gespeichert.
 - **!** **Also immer aufpassen auf welchem Branch man sich befindet und seine commits macht.**



- In GitHub Client Speicherort des Projekts öffnen



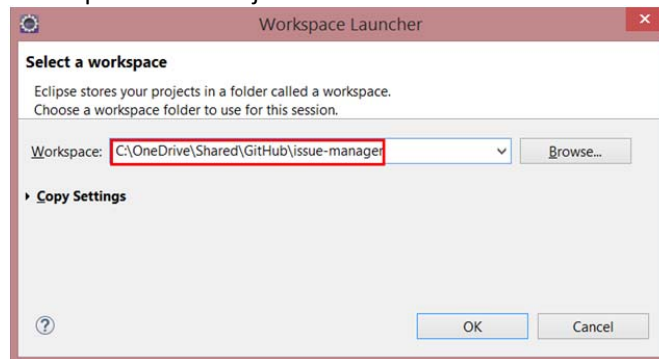
- PowerShell öffnen und in dieses Verzeichnis navigieren.
 - Man kann auch 'Open in Git Shell' anklicken, einfach sicher sein, dass auch PowerShell und nicht das CMD geöffnet wird.
- Neuster Release auswählen
 - Innerhalb eines Branch gibt es mehrer Release
 - Release anzeigen: `git tag`
 - Den letzten Release abfragen: `git checkout tags/v2.2`
- Anschliessend des Projekt mit gradle compilieren: `gradle build`



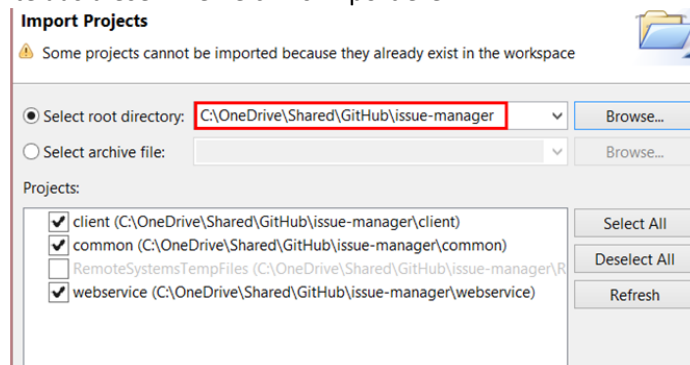
- Bei der ersten Installation produziert dieser Prozess wesentlich mehr Output.
- Das Projekt für eclipse aufbereiten: `gradle eclipse`

```
PS C:\OneDrive\Shared\GitHub\issue-manager> gradle eclipse
:client:eclipseClasspath
:client:eclipseJdt
:client:installLombok
Lombok installed to: C:\Program Files (x86)\eclipse\eclipse.exe
:client:eclipseProject
:client:eclipse
:common:eclipseClasspath
:common:eclipseJdt
:common:installLombok
Lombok installed to: C:\Program Files (x86)\eclipse\eclipse.exe
:common:eclipseProject
:common:eclipse
:webservice:eclipseClasspath
:webservice:eclipseJdt
:webservice:installLombok
Lombok installed to: C:\Program Files (x86)\eclipse\eclipse.exe
:webservice:eclipseProject
:webservice:eclipse
BUILD SUCCESSFUL
Total time: 7.14 secs
PS C:\OneDrive\Shared\GitHub\issue-manager>
```

- Eclipse öffnen und neuer Workspace auswählen.
 - Workspace muss Projekt-Verzeichnis sein.



- Projekte aus diesem Verzeichnis importieren.



- In phpMyAdmin einloggen und user mit Datenbank erstellen
 - <http://localhost/phpmyadmin/index.php>

Benutzer hinzufügen

Anmelde-Informationen

Benutzername:

Host:

Passwort:

Wiederholen:

Passwort generieren:

Datenbank für Benutzer

☒ Erstelle eine Datenbank mit gleichem Namen und gewähre alle Rechte.

☐ Gewähre alle Rechte auf Datenbanken die mit dem Benutzernamen beginnen (username_%).

- Datenbank mit Daten füllen: `gradle seed`

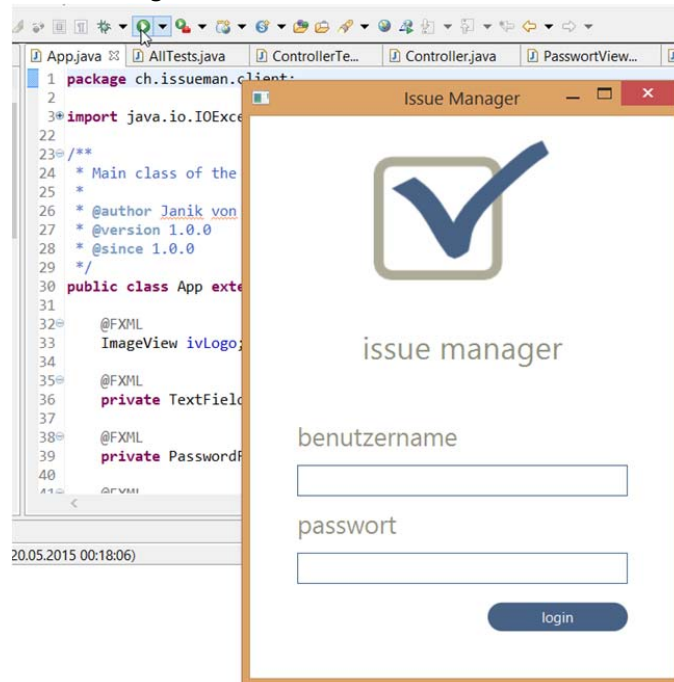
```
C:\OneDrive\Shared\GitHub\issue-manager [master +0 ~1 -0] > gradle seed
:common:compileJava UP-TO-DATE
:common:processResources UP-TO-DATE
:common:classes UP-TO-DATE
:common:jar UP-TO-DATE
:webservice:compileJava
:webservice:processResources UP-TO-DATE
:webservice:classes
:webservice:seed
[main] INFO com.github.javafaker.service.FakeValuesService - Using locale en
[EL Infol: server: 2015-04-06 21:52:08.402--ServerSession(23079252)--Detected
latform.server.NoServerPlatform.
Seeded: 63 people
Seeded: 6 projects
Seeded: 66 comments

BUILD SUCCESSFUL
Total time: 8.944 secs
C:\OneDrive\Shared\GitHub\issue-manager [master +0 ~1 -0] >
```

- Webservice mit Gradle starten: `gradle jettyRun`

```
C:\OneDrive\Shared\GitHub\issue-manager [master] > gradle jettyRun
:common:compileJava UP-TO-DATE
:common:processResources UP-TO-DATE
:common:classes UP-TO-DATE
:common:jar UP-TO-DATE
:webservice:compileJava UP-TO-DATE
:webservice:processResources UP-TO-DATE
:webservice:classes UP-TO-DATE
> Building 87% > :webservice:jettyRun > Running at http://localhost:8080/webservice_
```

- Client Anwendung starten



- In Client Anwendung einloggen

Rolle	Username	Passwort
Sachbearbeiter	sb@im.ch	1
Bauleiter	bl@im.ch	1
Kontaktadmin	ka@im.ch	1
Kontaktperson	kp@im.ch	1

18 TDD und JUnit

18.1 Test-Driven-Development (TDD)

Um Vertrauen in den laufend erstellten Code zu erhalten, wurden JUnit –Testfälle geschrieben. Diese Testfälle wurden nach dem Test-Driven-Development-(TDD)-Prinzip erstellt. Dabei bilden die Testfälle ein gewünschtes Zielverhalten der Applikation ab, ohne dass diese bereits entwickelt ist. Die Tests laufen also bei der erstmaligen Ausführung auf Fehler. Die Applikation wird nun Schritt für Schritt entwickelt, mit dem Ziel die Vorgaben der Testfälle zu erfüllen.

Das Berechtigungs-technische Verhalten der Applikation wird durch Jason geregelt.

18.2 JUnit

Unsere Unit-Test sind automatisiert und wiederholbar. Alle Unit-Tests sind auf Knopfdruck ausführbar. Für die Unit-Tests in dieser Applikation wurde JUnit 4 verwendet. Bei jedem Test wird ein Sachbearbeiter eingeloggt und auch wieder ausgeloggt.

Voraussetzung für die Tests ist, dass das mithilfe des Seed-Files die Datenbank gefüllt wird. Damit kann verhindert werden, dass NullPointerExceptions geworfen werden da wir so viel Datensätze erstellen, dass es für mindesten zwei Durchläufe aller Test genügt.

Da wir alle Datenbankzugriffe über Controller steuern lag unser Fokus auf den Grundfunktionen der jeweiligen Controllern.

18.3 Server / Client (End-to-End testing)

Da wir alle Fehler, welche vom Server her ruhen an den Client weiterleiten, war es nicht notwendig explizite Tests auf der Server-Seite zu implementieren.

Damit die Client-Tests funktionieren muss zuerst die Datenbank mit Test-Datensätzen gefüllt werden. Da wir nicht zentral auf eine Datenbank Zugriff hatten, haben wir uns entschieden aussagekräftige Datensätze von „Hand“ zu erstellen. Grund dafür war, dass wir alle immer mit den gleichen Daten arbeiten und somit die gleichen Ausgaben bekamen.

Die Befüllung der Datenbank wird mittels der Seed.java Klasse realisiert.

Wie die Client-Tests genau aufgebaut sind, wird im Detail anschliessend an diesen Absatz erklärt.

Folgende Testklassen haben wir erstellt:

Test Nr.	Test	Autor
01	ContextTest	Janik Von Rotz
02	ControllerTestAdresse	Aathavan Theivendram
03	ControllerTestArbeitstyp	Sandro Klarer
04	ControllerTestBauherr	Erwin Willi
05	ControllerTestBauleiter	Erwin Willi
06	ControllerTestKontakt	Erwin Willi
07	ControllerTestMangel	Erwin Willi
08	ControllerTestModel	Janik Von Rotz
09	ControllerTestProjekt	Erwin Willi
10	ControllerTestRolle	Reno Meyer
11	ControllerTestSachbearbeiter	Erwin Willi
12	ControllerTestLogin	Erwin Willi

Da alle Zugriffe die auf die gleiche Weise ablaufen, haben wir alle unsere Tests auf die gleiche Art aufgebaut:

Funktionsnamen	Aufgabe
setUp()	Login für Datenbankzugriff setzen und einloggen
tearDown()	ausloggen
testPersist()	Datensatz des jeweiligen Controllers in DB schreiben
testGetAll()	letzter Datensatz aus der DB laden (komplettes Objekt)
testGetById()	Datensatz aus der DB laden mit entsprechender ID
testUpdate()	passendes Feld vom letzten Datensatz ändern
testDelete()	Letzter Datensatz löschen

Einzige Ausnahme zu den oben genannten Funktionen bildet der ContextTest (Nr 01). In diesem Test wurde lediglich der Zugriff zur Datenbank bzw. zum Webservice getestet.

Beim AllTest() handelt es sich um einen Test, welcher alle anderen Tests aufruft bzw. ausführt.

18.4 Testfälle

Folgende Punkte müssen die TDD-Tests, ausgenommen dem **ContextTest**(#1), erfüllen:

- Datensatz schreiben
- Letzten Datensatz laden
- Bestimmten Datensatz mit entsprechender ID kann geladen werden
- Aussagekräftiges Feld aus letztem Datensatz laden und ändern
- Letzten Datensatz löschen → muss eine Exception werfen bzw. eine passende Fehlermeldung ausgeben

Die weiteren Testfällen wurden vor der Abnahme durch das Projektteam in der Applikation getestet.

Test-Nr.	Req-Nr	Testsituation	Zu testende Punkte	Soll-Resultat	Ist-Resultat	Status
01		Verbindung zur Datenbank herstellen und wider ausloggen	Kann eine Verbindung mit Sachbearbeiter zur DB hergestellt werden und wider ausgeloggt werden	Verbindung zur DB herstellen und wieder ausloggen	Verbindung zur DB hergestellt und wider ausgeloggt.	OK
02		Testen des ControllerTestAdress	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Alle oben beschriebenen Punkte erfüllt	OK
03		Testen des ControllerTestArbeitstyp	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Alle oben beschriebenen Punkte erfüllt	OK
04	23	Testen des ControllerTestBauherr	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Alle oben beschriebenen Punkte erfüllt	OK
05	22	Testen des ControllerTestBauleiter	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Alle oben beschriebenen Punkte erfüllt	OK
06		Testen des ControllerTestKontakt	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Alle oben beschriebenen Punkte erfüllt	OK
Test-Nr.	Reg-Nr	Testsituation	Zu testende Punkte	Soll-Resultat	Ist-Resultat	Status
07	31	Testen des	Alle oben beschriebenen	Die oben beschriebenen	Alle oben	OK

		ControllerTestMangel	Punkte erfüllen	Punkte erfüllen	beschriebenen Punkte erfüllt	
08		Testen des ControllerTestModel	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Alle oben beschriebenen Punkte erfüllt	OK
09	25	Testen des ControllerTestProjekt	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Bis auf den PersistTest funktioniert alles tadellos	NOK
10		Testen des ControllerTestRolle	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Alle oben beschriebenen Punkte erfüllt	OK
11		Testen des ControllerTestSachbearbeiter	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Alle oben beschriebenen Punkte erfüllt	OK
12	11	Teste des ControllerTestLogin	Alle oben beschriebenen Punkte erfüllen	Die oben beschriebenen Punkte erfüllen	Alle oben beschriebenen Punkte erfüllt	OK

Gründe wieso nicht alle Tests funktionieren

Da das erfassen eines Projektes aus der Applikation heraus funktioniert, haben wir uns infolge Zeitmangels entschieden diesen Fehler zu einem späteren Zeitpunkt zu behandeln und zuerst die schwerwiegenden Probleme zu lösen.

18.5 Erfahrungen mit TDD

Beim TDD ergibt sich die Herausforderung bereits beim Erstellen der Unit-Test zu wissen, was die Klasse alles für Funktionalitäten abdecken sollte. Zudem können sich aus den Funktionstest am Ende neue Anforderungen ergeben, wodurch der TDD-Prozess in seiner Reinheit schwierig einzuhalten ist.

Trotzdem zeigte es sich, dass das TDD Prinzip einen enormen Beitrag an die Effektivität und die Effizienz der Code-Erstellung bringt. Dass bedeutet, es wird der Code geschrieben, der auch benötigt wird. (Effektivität) Zudem wird gezielt auf diesen Code hingearbeitet und somit die Zeit zur Codegewinnung verkürzt, da die Gedankengänge sofort Zielorientiert geleitet werden.

18.6 Erfahrungen mit Unit-Testing

Die Nutzung von JUnit 4 war eine grosse Hilfe bei der Erstellung der Test. Vor allem konnten durch die Tests Probleme der Datenintegrität in unserer Datenbank ausgemerzt werden. Dank den Test konnten wir auch feststellen ob Berechtigungstechnisch alles funktioniert.

Ein weiterer Vorteil war, dass nach jeder grösseren Änderung am Code die Tests ausgeführt werden konnten, um zu prüfen ob noch alles reibungslos abläuft.

Ebenfalls hilfreich waren die Tests bei der Erstellung der Seed-Klasse (Datenbank füllen) da uns mit Hilfe der Tests immer wieder aufgefallen ist, was wir bezüglich dem schreiben der Datensätze noch anpassen mussten.

18.7 Probleme und Lösungen

Nach dem Erstellen der ersten Tests ist uns aufgefallen, dass wir die Tests welche eine Exception werfen müssen, umschreiben müssen. Grund für das werfen der Exceptions sind die Berechtigungen, welche wir vorgängig auf der Datenbank bzw. per Permission.csv gesetzt haben.

Ein weiteres Problem war, dass die einzelnen Funktionen in den Test willkürlich ausgeführt werden (wird von JUnit 4 bestimmt). Doch nach kleinen Anpassungen der Test, war auch dieses Problem behoben.

Doch nach Anfänglichen Problemen bekamen wir das Ganze sehr gut in den Griff.

19 Funktionale Tests

19.1 Testprotokoll

Applikation	Issue-Manager
Version 1.0	<input checked="" type="checkbox"/> Funktionstest
	<input type="checkbox"/> Integrationstest
	<input type="checkbox"/> Abnahmetest
Testverantwortlicher	Erwin Willi

Resultat des Funktionstests:

Testresultat	<input checked="" type="checkbox"/> bestanden <input type="checkbox"/> nicht bestanden
Bemerkungen	Die Funktionstests waren alle erfolgreich.
Datum	19.05.2015
Testverantwortlicher	Erwin Willi

19.2 Abnahmeprotokoll

Applikation	Issue-Manager
Version	1.0
Fachverantwortlicher	

Resultat der Abnahme:

Applikation	Issue-Manager
Version 1.0	<input type="checkbox"/> Funktionstest
	<input type="checkbox"/> Integrationstest
	<input checked="" type="checkbox"/> Abnahmetest
Testverantwortlicher	Erwin Willi

Zustand	<input checked="" type="checkbox"/> Release in Ordnung <input type="checkbox"/> Release nicht in Ordnung
Bemerkungen	Das Deployment auf dem vorgegebenen Rechner hat Problemlos funktioniert. Die Applikation arbeitet einwandfrei
Datum	19.05.2015
Abnahme durch	Erwin Willi

Ort, Datum: Luzern, 19.05.2015

Verantwortlicher: Erwin Willi

19.3 Testfälle

Folgende Tests wurden mit Hilfe des Tools durchgeführt. Dabei war die Ausgangssituation die gleiche wie bei den nicht funktionalen Tests. Also die Datenbank ist mit Daten gefüllt.

TCA	REQ	Testsituation	Zu testende Punkte	Soll-Resultat	Ist-Resultat	Status
13	13	Anmeldung mit E-Mail und Passwort	Funktioniert Login	Login funktioniert	Login ist erfolgreich	OK
14	12	Abmeldung an der Applikation	Nach Abmeldung wird die Session geschlossen	Abmeldung wird durchgeführt und Session beendet.	Abmeldung ist erfolgreich. Session wird beendet und neu gestartet.	OK
15	13	Einloggen als BL	BL hat nur Zugriff auf die ihm zugewiesenen Projekte und Mängel	Es werden nur Mängel und Projekte des entsprechenden BL angezeigt	BL sieht nur die ihm zugeordneten Mängel und Projekte.	OK
16	14	Persönliches Passwort ändern	Einloggen und das persönliche Passwort ändern	User kann sein persönliches Passwort wechseln	Passwort kann nicht gewechselt werden	OK
17	15	Passwort durch SB ändern lassen	Kann der SB sich anmelden und das Passwort ändern	Passwort kann durch SB geändert werden	Passwort kann nicht geändert werden	OK
18	21	Erfassen der Stammdaten mit allen Objekten (C)	Erfassen von Stammdaten für alle Objekte	Stammdaten können erstellt werden	Stammdaten könne erstellt werden	OK

TCA	REQ	Testsituation	Zu testende Punkte	- Soll-Resultat	- Ist-Resultat	Status
19	22/23/24/25	Datenverwaltung aller möglichen Kombinationen (RUD)	<ul style="list-style-type: none"> - Lesen - Ändern - Löschen - 	<ul style="list-style-type: none"> - Daten können nach ID oder Propertie angezeigt werden - Daten können angepasst werden Daten können nicht gelöscht werden 	<ul style="list-style-type: none"> - Daten können gelesen werden - Daten können angepasst werden Daten können nicht gelöscht werden 	OK
20	31-37/41-43	Mangelmanipulationen durchführen	<ul style="list-style-type: none"> - Mangel erfassen - Projekt zuordnen - Mangel Beschreibung - Subunternehmen zuordnen - Status setzen - Frist definieren Erfassung Datum	Alle Punkte müssen erfüllt werden	Alle Punkte wurden erfüllt	OK
21	44	Export von allen Mängeln	Angezeigte Mängel exportieren	Angezeigte Mängel können in csv-File exportiert werden	Csv-File wird erstellt.	OK

19.4 Zielüberprüfung

REQ #	Requirement	Ziel erreicht?	Begründung/Kommentar
Authentifizierung			
11	Anmelden mit Benutzername und Passwort	Ja	
12	Abmelden und Beenden der Session	Ja	
13	Rollenbasierte Berechtigungen	Ja	
14	Passwort ändern	Ja	
15	Passwort zurücksetzen	Ja	
Stammdatenverwaltung			
21	Neue Stammdaten erfassen	Ja	
22	Daten der Bauleiter verwalten	Ja	
23	Daten der Bauherren verwalten	Ja	
24	Daten der Subunternehmen verwalten	Ja	
25	Daten der Bauprojekte verwalten	Ja	
Mängelerfassung			
31	Mangel erfassen	Ja	
32	Betreffendes Projekt zuordnen	Ja	
33	Mangel beschreiben (kurzer Beschrieb)	Ja	
34	Betreffendes Subunternehmen zuordnen	Ja	
35	Status setzen	Ja	
36	Frist definieren	Ja	
37	Erfassungsdatum wird notiert	Ja	
Mängelbearbeitung			
41	Mangeldaten verändern	Ja	
42	Mangelstatus verändern	Ja	
43	Kommentar abgeben	Ja	
44	Mangeldaten exportieren	Ja	
Design			
51	GUI	Ja	
52	Sortierung von Tabellen	Ja	
53	Suchfunktion	Ja	
54	Datenvalidierung	Ja	
55	Aufforderung zum Speichern	Nein	Aus zeitlichen Gründen auf dieses Feature verzichtet.

56	Datenintegrität	Ja
57	Warnungen	Ja
Technische Anforderungen		
61	Entwicklung mit Eclipse und JDK 1.8	Ja
62	Client und Server sind verteilbare Komponenten	Ja
63	Dokumentation mit JavaDoc	Ja
64	Abstraktion des DBMS Layer mit ORM	Ja
65	GUI Entwicklung mit JavaFX	Ja
66	Event Logging	Ja
67	DBMS muss austauschbar sein	Ja
68	Datenverarbeitung auf Applikationsserver	Ja
69	Applikation als Download für das Subunternehmen	Ja
Testing		
71	JUnit-Test Tests nach TDD-Verfahren	Ja
72	Abnahme mittels funktionalen Tests	Ja

20 DB-Dokumentation

Die Sachverhalte dieses Kapitels wurden bereits im Kapitel 16 beschrieben.

21 Beiträge pro Projektmitglied

21.1 Elsener Patrick

Datum Phase	eventuell Aufwand	Tätigkeit / Aufgabe	Bemerkung/Erkenntnis
Anfang	2.0 Std.	Requirements ausarbeiten	
Späte Anfangsphase	16.0 Std.	Client Prototyp erstellen	Erkenntnis: Früher Janik fragen, hätte vermutlich einige Zeit und Frust erspart
Mittlere Phase und Schlussphase	8.0 Std.	TopView und LeftView erstellen	Im Nachhinein grosse Freude, dass in der LeftView die verschiedenen Projekte, Subunternehmen und Personen in einer ausklappbaren Liste angezeigt werden. Diese habe ich auch bis auf ein paar wenige Inputs von den Teammitgliedern selber realisiert
Mittlere Phase	4.0 Std.	ErrorBox Method	Erkenntnis: Früher Janik fragen, hätte vermutlich einige Zeit und Frust erspart
Mittlere Phase und Schlussphase	3.0 Std.	Views implementieren	
Schlussphase	1.0 Std.	Logos einfügen	
Schlussphase	1.0 Std.	DoubleClick Method	
Anfangsphase	2.0 Std.	Use Case schreiben	
Schlussphase	3.0 Std.	Dokumentation Kapitel 5	
Schlussphase	0.5 Std.	Dokumentation Kapitel 6	
Schlussphase	0.6 Std.	Dokumentation Kapitel 7	
Schlussphase	1.0 Std.	Dokumentation Kapitel 9	
Schlussphase	0.3 Std.	Dokumentation Kapitel 10	
Schlussphase	0.3 Std.	Dokumentation Kapitel 11	

Schlussphase	0.6 Std.	Scrollbare Views	
Ende	5.0 Std.	Schlussformatierung, Drucken	

21.2 Klarer Sandro

Datum Phase	eventuell	Dauer Aufwand	Tätigkeit / Aufgabe	Bemerkung/Erkenntnis
14.03.2015		1 Std.	Skizzierung der Ausgangslage: Ich habe mich mit der Aufgabenstellung befasst und dabei Skizzen und Notizen bezgl. des aktuellen Prozesses und der gewünschten Lösung gemacht. Diese Skizzen dienen der persönlichen Orientierung und sind in der Dokumentation nicht vorhanden.	Sehr sehr interessante Technologien, ganz besonders das ORM Hibernate.
17.03.2015		1.5 Std.	#2 Definition Requirements: Ich habe anhand meiner Notizen und Skizzen die Kundenanforderungen mit Patrick erfasst.	Mit Gradle kann man sehr einfach Java Entwicklungs-Prozess bezogene Aufgaben automatisieren.
23.03.2015		1 Std.	#19 Use Case schreiben: Ich habe für mich persönlich und evtl. als Grundstein für die Gruppe, den UseCase "Mängel erfassen" erstellt. Meine Vorlage werden wir im kommenden Meeting besprechen und sie kann, wenn man noch einige kleine Anpassungen/ Korrekturen vornimmt als Template für die Gruppe verwendet werden.	Mit Jax-RS können ganz einfach Restful webservices erstellt werden, genial!
26.03.2015		1 Std.	# 10 Bereitstellung IDE: Anhand der Anleitung von Janik von Rotz habe ich meine Entwicklungsumgebung eingerichtet.	Kaum zu glauben aber mit Jackson kann man nested POJOs hin und her konvertieren.
29.03.2015		0.5 Std.	#2 Definition Requirements: Zugeteiltes Requirement "Mängel erfassen" ausgearbeitet und in einzelne Anforderungen aufgesplittet. Diese ist in der Dokumentation Kapitel 14, Requirement 30 zu finden.	Juhu, die Applikation funktioniert. Links habe ich dem Task hinzugefügt.
29.03.2015		1.0 Std.	#14 Requirements Dokumentation anpassen: Aufgrund der aufgesplitteten Anforderungen des Requirement "Mängel erfassen" habe ich die notwendige Beschreibung der einzelnen Anforderungen erfasst.	Gut das wir alle einmal zusammen gekommen sind. Das Meeting konnte sehr effizient durchgeführt werden.

		Requiermentsbeschreibung ist im K14 aufgelistet.	
29.03.2015	0.3 Std.	# 16 Dokumentation Kapitel 1-3	Wir merkten gleich dass dies nicht wirklich nötig war, da die Datenbanks vom ORM erstellt werden kann.
04.04.2015	1 Std.	#14 Requirements Dokumentation anpassen: Ich habe die Requiermentsbeschreibung von dem Requirement 30, "Mängel erfassen", angepasst und finalisiert. Dies ist ist in der Dokumentation K14 zu finden.	Es ist schwierig anhand des Textes die Definitionen zu erkennen und zu adaptieren.
04.04.2015	2 Std.	#19 Use Case schreiben: Ich habe einen Use Case "Mängel erfassen" finalisiert und die Beschreibung dem Gruppentemplate angepasst. Da ich den Use-Case in erster Instanz als Entwurf für ein Template entworfe habe, ohne dabei die detaillierten Requirements ausgearbeitet zu haben, waren einige Anpassungen notwendig. Der Use Case und die dazugehörige Beschreibung ist im K15 abgelegt.	In einem Zweiterteam kommt man bei solchen Arbeiten besser voran.
09.04.2015	0.5 Std.	# 16 Dokumentation Kapitel 1-3: Ich habe mir stichwortartig für die Kapitel relevanten Punkte notiert.	
16.04.2015	0.8 Std.	# 24 Model Login und Kommentar: Gemäss Anleitung von Yanik von Rotz habe ich die Klassen Login und Kommentar erstellt.	Das Modell musste entsprechend neu aufgebaut werden.
16.04.2015	0.1 Std.	# 28 Model Ort und Adresse: Janik hat als Demonstrationszweck die Klasse Ort implementiert. Anhand dieser Vorlage habe ich die Klasse Adresse erfasst.	Ein Verantwortlicher muss Definition zu den Aufträgen erarbeiten.
22.04.2015	0.6 Std.	# 33 Routes implementieren: Im Route.java habe ich anhand der Vorlage für jede Entity die Update, Post und Delete methode festgelegt.	Bis jetzt war dies die grösste Herausforderung. Auf dem Client kann nun ganz einfach ein Controller für die ganze Laufzeit authentifiziert werden.
23.04.2015	0.5 Std.	# 39 GUI Skelett mit SceneBuilder: Reno Meyer und Aathavan Theivendram haben mit einem Mock-Up Tool die verschiedenen GUI-Elemente ausgearbeitet. Meine Aufgabe ist es nun, diese Vorlagen, so gut es geht, mit dem Scene Builder nachzustellen.	

25.04.2015	2 Std.	# 39 GUI Skelett mit SceneBuilder: Template + Ansichten	
26.04.2015	1.5 Std.	# 39 GUI Skelett mit SceneBuilder:	
27.04.2015	2 Std.	# 39 GUI Skelett mit SceneBuilder:	
30.04.2015	4 Std.	# 39 GUI Skelett mit SceneBuilder:	Bin mir noch nicht Sicher ob die Überführung des Users zur Login Klasse auch wirklich funktioniert.
02.05.2015	4 Std.	# 39 GUI Skelett mit SceneBuilder:	
04.05.2015	1.5 Std.	# 42 Views Implementieren: Ich habe gewisse Views: PasswortView, SubunternehmenZugewiesenView und ProjektleitungView zugeteilt bekommen. Meine Aufgabe ist es nun diese zu implementieren.	
06.05.2015	1 Std.	# 42 Views Implementieren:	Es gibt noch einige Probleme beim Mapping des Datemodells
06.05.2015	1 Std.	# 16 Dokumentation Kapitel 1-3: Ich habe mir stichwortartig für die Kapitel relevanten Punkte notiert.	Nach wie vor die anspruchvollste Aufgabe soweit. Methodisch konnten wir die Abhängigkeiten analysieren und entsprechend ausarbeiten.
07.05.2015	1 Std.	# 42 Views Implementieren: PasswortView und Passwort FXML file	
07.05.2015	0.25 Std	# 39 GUI Skelett mit SceneBuilder: Bauherr FXML File an neue Umstände anpassen.	
07.05.2015	0.25 Std.	# 44 Style sheet erstellen: Label Background-Color und Border-Color, TextField Border-Color, ComboBox Roll	Die Übersetzung von vererbten Klassen als JSON in ein POJO hat sich als sehr schwierig erwiesen. Aber zum Glück gab es genügen Ressourcen auf dem Web.
09.05.2015	0.5 Std.	# 44 Style sheet erstellen: TableView, Button und Stylesheet in Views laden	
09.05.2015	1.5 Std.	# 39 GUI Skelett mit SceneBuilder: Korrekturen anhand Anweisungen + Stylesheets laden	Testing lohnt sich!
11.05.2015	2 Std.	# 44 Style sheet erstellen: TableView, Button und Stylesheet in Views laden	Funktioniert noch nicht, da unklar wie RMI korrekt gestartet und initialisiert werden kann.

11.05.2015	0.5 Std.	# 39 GUI Skelett mit SceneBuilder: Korrekturen anhand Anweisungen + Stylesheets laden	Eine Neue Komponente muss dazwischen geschaltet werden.
11.05.2015	0.2 Std.	# 42 Views Implementieren: PasswortView und Passwort FXML file	
13.05.2015	0.75 Std.	# 16 Dokumentation Kapitel 1-3: Ich habe mein Entwurf indas Dokumentation Template eingefügt und kleine Verbesserungen vorgenommen.	
13.05.2015	0.75 Std.	# 39 GUI Skelett mit SceneBuilder & # 44 Style sheet erstellen: Aufgrund eines Misslungen Merge auf Git-Hub sind einige Änderungen rückgängig gemacht worden. Deshalb musste ich diese nochmals neu erfassen.	Junit Tests sind sehr abgeschottet voneinander, haben dadurch einige Denkfehler gemacht und entsprechend viele Anläufe gebraucht.
14.05.2015	0.3 Std.	# 44 Style sheet erstellen: Da einige FXML Files angepasst wurde, habe ich die Style Class angepasst.	
14.05.2015	2 Std.	# 16 Dokumentation Kapitel 1-3: Da ich mit dem bisherigen Ergebniss nicht zufrieden war, habe ich mich nochmals intensiv mit dem entsprechenden Kapitel befasst. Dabei habe ich festgestellt, dass ich einige Sachverhalte anpassen muss. Es ist alles zu "oberflächlich" geschrieben.	
15.05.2015	4 Std.	# 42 Views Implementieren: ProjektleitungView und SubunternehmenZugewiesenView	
16.05.2015	2.5 Std.	# 16 Dokumentation Kapitel 1-3: Ich habe diese Kapitel fertiggestellt.	
16.05.2015	4 Std.	# 44 Style sheet erstellen: Feinschließ der letzten Details.	
17.05.2015	4 Std.	# 44 Style sheet erstellen: Feinschließ der letzten Details.	
17.05.2015	0.1 Std.	# 62 Dokumentation Kapitel 20:	
18.05.2015	1 Std.	# 64 Dokumentation Kapitel 22: Wie kann man den Source Code aus Eclipse exportieren und drucken.?	
18.05.2015	1 Std.	# 44 Style sheet erstellen: Feinschließ der letzten Details.	
19.05.2015	2 Std.	# 44 Style sheet erstellen: Feinschließ der letzten Details.	

21.3 Meyer Reno

Datum Phase	eventuell	Dauer Aufwand	Tätigkeit / Aufgabe	Bemerkung/Erkenntnis
Design		6h	Designkonzept: Entwicklung des GUI-Designs, inklusive Prozesslogik und Farbkonzept (--> siehe Dokumentation Kapitel 23)	Die User-Experience steht bei modernen Applikationen im Mittelpunkt und war mir ein besonderes Anliegen. Das GUI-Design setzt zahlreiche ergonomische Merkmale um und versucht den logischen Prozess für die Bedienung abzubilden. Nebst der angegebenen Zeit ist sehr viel Denkarbeit, sowie Aufwand zum (externen) Feedback einholen entstanden.
Client		>50h	Views: Umsetzung der DetailView-Controller, sowie Mithilfe und Koordination bei den anderen View-Controller, CSS-styling, und FXML generierung mit Scene Builder (--> siehe Sourcecode und Benutzeroberfläche der Applikation).	Die Applikation hat eine nicht zu unterschätzende Komplexität. Kleine Änderungen, welche unvermeidbar sind, bringen oft gravierenden Anpassungsbedarf mit sich. Die Zusammenarbeit war trotz GitHub Repository und weiteren Management-Tools nicht immer einfach und gemeinsame Arbeitssessions waren in meiner Absicht essentiell für den erfolgreichen Abschluss des Projekts.
alle Phasen			Projektleitung: Vorbereitung von Teamsitzungen, Koordination, Prioritisierung und Zuteilung von Arbeiten, Motivation des Teams (--> gruppeninterne Kommunikation, welche nicht mit den Dozenten geteilt wird)	Meine Aufgabe als Projektleiter habe ich so verstanden, dass ich im Normalfall immer einen Schritt voraus sein sollte. Dies ist mir teilweise gut gelungen; bei technischen Aspekten, war ich jedoch auf die tatkräftige Mithilfe von Janik (Architekt) angewiesen. Des weiteren haben sich auch alle anderen Teammitglieder sehr aktiv im Planungs- und Entwicklungsprozess beteiligt.
Architektur/ Webservice				Mit der Architektur einer verteilten Anwendung war ich absolut überfordert. Ich finde, dass wir

			im Unterricht völlig ungenügen auf diese hochkomplexe Aufgabe vorbereitet wurden und sie eigentlich nur dank sehr tiefen Vorkenntnissen von Janik bewältigen konnten.
verteilt	15h	Dokumentation: Vorbereitung und Styling des Grundgerüst und Templates, Schreiben von verschiedenen Kapiteln, Editieren (--> siehe gesamt Dokumentation, speziell Kapitel 4, 8, 14, 15, 23)	Die Arbeit an der Dokumentation empfand ich als unnötig und mühsam. Der zusätzliche Aufwand zur eigentlichen Entwicklungsarbeit war relativ hoch, wobei ich den Nutzen hinsichtlich Lerneffekt als sehr gering betrachte. Auch hier war die Vorbereitung im Unterricht wieder ungenügen und die technischen Anforderungen zu hoch. In Bezug auf unseren Wissensstand, sowie den Umfang des Projektes, erachte ich das Template von Arc42 als unpassend.

21.4 Theivendram Aathavan

Datum Phase	eventuell	Dauer Aufwand	Tätigkeit / Aufgabe	Bemerkung/Erkenntnis
Vorbereitung		1,0 Std.	#8 Collaboration Tool evaluieren: Zur engeren Auswahl kam Taiga und waffle und ich habe mich enger mit den beiden Tools beschäftigt. Später habe ich einen Pro/Contra Vergleich erstellt, um auf dessen Basis beim folgenden Teammeeting eine Entscheidung zu fällen.	Ich war erstaunt wie etliche Project-Management-Tools sich um die Gunst der User buhlen. Sie werden (zum Glück) immer innovativer und benutzerfreundlicher (siehe http://www.capterra.com/project-management-software/)
Vorbereitung		1,0 Std.	#10 Bereitstellung IDE: Um mit einer soliden und vor allem homogenen Entwicklungsumgebung zu arbeiten, habe ich für Team C ein Image erstellt, welches als virtuelle Umgebung eingesetzt werden konnte.	Später (#4 2015-03-17 Teamsitzung) haben wir entschieden ohne virtuelle Maschine zu arbeiten, weil wir nicht so viele Tools (wie vermutet) einsetzen werden. Auf OneNote wurden die relevanten Tools mit Download-Links aufgelistet.

Requirements Engineering	3.0 Std.	#2 Definition Requirements: Nach einer gemeinsamen Session bezüglich den Requirements habe ich meinen Teil mit der Stammdatenverwaltung ausformuliert. Als Vorbereitung habe ich die Aufgabenbeschreibung im Detail angeschaut und die relevanten Inhalte zusammengefasst. Ausserdem ich das "Volere" Dokument angeschaut, um die Requirements richtig beschreiben zu können.	Es war wichtig, dass alle Teammitglieder das gleiche Verständnis bezüglich Projekt mitbringen. Deshalb war die gemeinsame Session zu Beginn sehr hilfreich.
Requirements Engineering	3.0 Std.	#18 Berechtigungskonzept erstellen: Zusammen mit Reno Meyer habe ich ein Excel-Sheet mit allen Rollen, Objekten mit deren Attributen und Rechten (CRUD) erfasst und daraus ein PivotTable erstellt. Dazu sind wir alle möglichen Szenarien der Interaktion mit der Applikation durchgegangen. Dieses File war später die Grundlage für die Handhabung der Permissions auf dem Webservice.	Die Aufgabe war sehr aufwändig, aber wir konnten uns dadurch Probleme, die wir sonst später hätten, schon vorgängig klären und lösen.
Development Model	1.0 Std.	#26 Model Bauleiter, Projektleitung und Bauherr: Implimentierung der erwähnten Model-Klassen.	Aufgrund der Erfahrungen aus dem 1. Semester, stellte es kein grosses Problem dar.
Development Webservice	6.0 Std.	#35 Logging Framework konfigurieren: Das Logging für das Webservice habe ich mit Tinylog umgesetzt. Anstatt direkt mit Tinylog zu arbeiten, habe ich mit Hilfe von Janik eine Lösung mit slf4j programmiert. Ausserdem habe ich den Configurator so definiert, dass die Logfiles nach dem Logrotate-Prinzip geschrieben werden.	Nach der anfänglichen Unsicherheiten, habe ich später die riesigen Möglichkeiten vom Tinylog-Framework entdeckt.
Development Client	4.0 Std.	#11 Client GUI skizzieren: Initial habe ich und Reno zusammen in einer Session unsere Ideen zum GUI per Hand skizziert. Einen Teil habe ich mit dem Moqup-Tool (moqups.com) nachgezeichnet und später in das Designkonzept mit einer Beschreibung, wie die entsprechende Seite genutzt wird, eingefügt.	Ich habe während dieser Tätigkeit immer mehr gemerkt, wie wichtig die GUI Skizzierung ist. Falls wir direkt im Scene Builder angefangen hätten, wären sicherlich einige Anläufe nötig gewesen. Mit unserem Ansatz konnten wir alle Probleme in der Benutzerführung schon vorgängig aufgreifen und lösen.

Development Client	38.0 Std.	#42 View implementieren: Nachdem ich den Admin-Bereich skizziert habe, implementierte ich nahtlos diese Ansichten als Controller und Views. Mit stetigem Aufbau von Know-how im Scene Builder, habe ich auch die restlichen Views auf das Designkonzept angepasst.	Wir wollten kein 0815-GUI und wie schwierig das wirklich ist, habe ich erst in dieser Phase gemerkt. Am Anfang lief alles träge, aber als das Wissen mal da war, ging es sehr schnell und das GUI nahm immer mehr das gewünschte Design an. Mit der Zeit hat es mir wirklich Spass gemacht mit den Möglichkeiten von Java FX und Scene Builder zu experimentieren.
Dokumentation	4.0 Std.	#19 UseCase schreiben: Alle UseCases zur Stammdatenverwaltung mit den entsprechenden Beschreibungen erstellt.	Ich habe gemerkt, dass das Thema UseCase unterschiedlich interpretiert werden kann. Ich habe die erstellten UseCases mit verschiedenen Mitstudenten angeschaut und jeder hatte eine andere Meinung, wie es richtig wäre.
Dokumentation	4.0 Std.	#56/57 Dokumentation Kapitel 14, 15: In der Dokumentation habe ich zusätzlich zu den UseCases die Requirements überarbeitet und finalisiert. Ausserdem die Überprüfung der Requirements definiert und bei den funktionalen Tests mitgeholfen.	Mit ist aufgefallen, dass in der Arc42 nur die Junit-Tests und funktionalen Requirements getestet werden. Aber eine Zielüberprüfung aller Requirements hat in der Dokumentation gefehlt und habe es deshalb hinzugefügt.

21.5 von Rotz Janik

Datum eventuell Phase	Dauer Aufwand	Tätigkeit / Aufgabe	Bemerkung/Erkenntnis
01.03.2015	3.0 Std.	#5 Entwicklung Beispielapplikation: Entwicklung und Recherche zu einfacher MVC Applikation mit Hibernate, Gradle, Eclipse, JavaFX und Flyway	Sehr sehr interessante Technologien, ganz besonders das ORM Hibernate.
02.02.2015	4.0 Std.	#5 Entwicklung Beispielapplikation: Entwicklung und Recherche zu einfacher MVC Applikation mit Hibernate, Gradle, Eclipse, JavaFX und Flyway	Mit Gradle kann man sehr einfach Java Entwicklungs-Prozess bezogene Aufgaben automatisieren.
09.03.2015	2.0 Std.	#6 Entwicklung 3-tier Beispielapplikation: Entwicklung und	Mit Jax-RS können ganz einfach Restful

		Recherche JAX-RS, Jackson Object Mapping	webservices erstellt werden, genial!
13.03.2015	4.0 Std.	#6 Entwicklung 3-tier Beispielapplikation: Implementation Mutli Model Architektur mit Webservice und Client	Kaum zu glauben aber mit Jackson kann man nested POJOs hin und her konvertieren.
15.03.2015	2.0 Std.	#6 Entwicklung 3-tier Beispielapplikation: Implementation Mutli Model Architektur mit Webservice und Client	Juhu, die Applikation funktioniert. Links habe ich dem Task hinzugefügt.
17.03.2015	2.0 Std.	#4 2015-03-17 Teamsitzung: Unsere erste Besprechung, siehe Traktanden im Meeting Protokol	Gut das wir alle einmal zusammen gekommen sind. Das Meeting konnte sehr effizient durchgeführt werden.
17.03.2015	1.0 Std.	#9 SQL definieren: Nebst dem Datenmodell haben wir auch gleich den SQL code definiert.	Wir merkten gleich dass dies nicht wirklich nötig war, da die Datenbanks vom ORM erstellt werden kann.
17.03.2015	1.5 Std.	#3 Datenmodell modellieren: Anhand der Auftragsdeklaration haben wir in einer ersten Phase versucht die Entities des Modells zu erkennen und schon einmal die Beziehungen untereinander zu definieren.	Es ist schwierig anhand des Textes die Definitionen zu erkennen und zu adaptieren.
18.03.2015	2.0 Std.	#3 Datenmodell modellieren: Das ERM konnte mit noch wenigen offenen Punkten fertiggestellt werden. Zusätzlich haben wir noch die SQL scripts dazu erstellt. Natürlich soweit als möglich.	In einem Zweiterteam kommt man bei solchen Arbeiten besser voran.
24.03.2015	3.0 Std.	#14 2013-02-24 Teamsitzung: Besprechen der Requirements und des Datenmodells	
25.03.2015	1.0 Std.	#3 Datenmodell modellieren: Wir haben bestimmen, dass sich Subunternehmen nur über deren Kontaktperson anmelden können.	Das Modell musste entsprechend neu aufgebaut werden.
25.03.2015	1.0 Std.	#2 Definition Requirements: Da unsere zweite Besprechung keine brauchbaren Resultate für das Requirement Engineering hervor brachte, habe ich eine Vorlage mit Beschrieb erstellt.	Ein Verantwortlicher muss Definition zu den Aufträgen erarbeiten.
29.03.2015	4.0 Std.	#6 Entwicklung 3-tier Beispielapplikation: Implementierung eines Authentifizierungsmechanismus mit Rollenverteilung	Bis jetzt war dies die grösste Herausforderung. Auf dem Client kann nun ganz einfach ein Controller für die ganze Laufzeit authentifiziert werden.

30.03.2015	1.0 Std.	#2 Definition Requirements: Erstellen der Requirements für die Technische Anforderungen	
31.03.2014	4.0 Std.	#20 2015-03-31 Teamsitzung: Besprechen der Requirements und weitere Aufträge	
10.04.2015	2.0 Std.	#22 2015-04-10 Teamsitzung: Besprechen der Requirements und Use Cases	
12.04.2015	1.0 Std.	#23 Model Person: Abstrakte Klasse Person implementieren.	
20.04.2015	3.0 Std.	#32 Webservice Komponenten aus Beispielprojekt bereitstellen: Die Komponenten wie z.B. der Authenticator habe ich aus dem Beispielprojekt ins IM Projekt überführt. Tests und Kommentare sind noch ausstehen.	Bin mir noch nicht Sicher ob die Überführung des Users zur Login Klasse auch wirklich funktioniert.
20.04.2015	1.0 Std.	#31 Seed Task implementieren: Klasse für Implementierung vorbereiten	
20.04.2015	0.5 Std.	#32 Routes implementieren: Routing Klasse zur einfachen Implementierung bereitstellen.	
23.04.2015	2.0 Std.	#31 Seed Task implementieren: Refactoring der gesamten Seed Klasse, Methoden zur Filterung und zum Logging ergänzt	Es gibt noch einige Probleme beim Mapping des Datemodells
24.03.2015	3.0 Std.	#31 Seed Task implementieren: Update Projekt Seed mit Abhängigkeiten.	Nach wie vor die anspruchvollste Aufgabe soweit. Methodisch konnten wir die Abhängigkeiten analysieren und entsprechend ausarbeiten.
25.04.2015	1.0 Std.	#31 Seed Task implementieren: Persist aufrufe in Methode ausgelagert.	
25.04.2015	1.0 Std.	#32 Routes implementieren: Login Route umbenennen	
25.04.2015	3.0 Std.	#32 Webservice Komponenten aus Beispielprojekt bereitstellen: Kommentieren der TypeFilters, ResponseBuilder und Route Klassen. Ergänzen Config mit Default Werten für unbestimmte entities. Den ResponseFilter schickt nun ganze Exception instance zurück, wenn etwas schief geht.	Die Übersetzung von vererbten Klassen als JSON in ein POJO hat sich als sehr schwierig erwiesen. Aber zum Glück gab es genügend Ressourcen auf dem Web.

26.04.2015	1.0 Std.	#38 end to end testing: Erstellen von Junit tests auf dem Client. Die Idee von end to end testing ist dass man alle Komponenten eines Services auf deren Funktionalität als Ganzes prüft anstatt nur einzelne Funktionstests.	
27.04.2015	3.0 Std.	#38 end to end testing: Zusammen mit Erwin und mit der Hilfe von Tim konnten einige Fehler anhand der Junit Tests gefunden werden.	Testing lohnt sich!
28.04.2015	3.0 Std.	#40 RMI implementieren: Abkopplung der DAO Response Builder Komponente mit RMI.	Funktioniert noch nicht, da unklar wie RMI korrekt gestartet und initialisiert werden kann.
29.04.2015	3.0 Std.	#40 RMI implementieren: Die Abkopplung konnte nicht wie geplant umgesetzt werden. Die http-Response lässt sich nicht serialisieren.	Eine Neue Komponente muss dazwischen geschaltet werden.
30.04.2015	1.0 Std.	#35 Logging Framework konfigurieren: Besprechen Idee von SLF4J und Integration Logging Framework in Java Servlet mit Aathavan. Erstellen ServletContextListener und vorbereiten für Integration Tinylog Konfiguration.	
30.04.2015	1.0 Std.	#17 Models implementieren: ManyToOne und OneToMany Beziehungen mussten mit dem EAGER fetch type ergänzt werden, da ansonsten nach einer Serialisierung der Entities die Beziehungen verloren gehen.	
01.05.2015	2.0 Std.	#38 end to end testing: Weiterarbeit and Controller Test für Mangel	Junit Tests sind sehr abgeschottet voneinander, haben dadurch einige Denkfehler gemacht und entsprechend viele Anläufe gebraucht.
01.05.2015	1.0 Std.	#34 Client Prototyp erstellen: Coding Conventions für GUI Komponenten definieren und Besprechen mit Projektteam	
02.05.2015	6.0 Std.	#38 end to end testing: Ein kritischer Bug wurde gefunden. Wenn man Listen mit Jackson konvertiert geht der Type bei vererbten Personen verloren, dadurch kann die Liste nicht mehr zurückkonvertiert werden. Die Generik in unserer Lösung macht diese Problematik extrem komplex.	
04.05.2015	2.0 Std.	#34 Client Prototyp erstellen: Erstellen view controller	

		interfaces, damit die anderen korrekte Views implementieren können. Implementieren der List und Detail View für Subunternehmen	
05.05.2015	1.0 Std.	#41 Views Implementieren: Viewable und ViewableDetail interfaces in den entsprechenden View handler implementieren.	
06.05.2015	1.0 Std.	#17 Models implementieren: Die Datumskonvertierung war nicht möglich. Der Fehler war in den Models. Bei Calendar Felder musste mit einer Temporal Annoation ergänzt werden.	
07.05.2015	0.5 Std.	#43 Error Box method: Vorbereiten Aufgabe, Beispiele zu Error Dialog mit Stack Trace anzeigen suchen.	
07.05.2015	1.0 Std.	#32 Webservice Komponenten aus Beispielprojekt bereitstellen: Der Authenticator benutzte immer noch den basic controller und nicht den business controller. Die Anpassung hatte zur Folge, dass die Signin Methode ebenfalls auf der RMI Schnittstelle realisiert werden musste.	
07.05.2015	2.0 Std.	#32 Webservice Komponenten aus Beispielprojekt bereitstellen: Die get by property methode und delete all waren noch nicht implementiert. Eine Realisierung findet nur serverseitig statt.	
11.05.2015	0.5 Std.	#67 Webservice UTF8 Codieren: Umlaute wurden nicht korrekt angezeigt. Fehler lag in der Codierung der JSON Daten. UTF musste explizit im Route Java angegeben werden.	
12.05.2015	1.0 Std.	#66 Top und Left View erstellen: Controller für die Top und Left view erstellen und für Patrick zur Implementation vorbereiten.	
15.05.2015	1.0 Std.	#19 Use Case schreiben: UAC 25 und 26 anpassen. Formatieren für Dokumentation.	
15.05.2015	2.0 Std.	#17 Models implementieren: Bei vererbten Objekten wurde die equals hash methode mit Lombok falsch deklariert. Spricht vererbte Objekte konnten untereinander nicht	

		verglichen werden.	
17.05.2015	2.0 Std.	#68 Architektur Komponenten bereitstellen: Bereitstellen der RMI, Webservice und Client Komponente zur verteilten Distribution.	
18.05.2015	1.0 Std.	#47 Dokumentatin Kapitel 5: Erstellen Sequenzdigramm und Whitebox und BlackBox Diagramme mit Patrick	Unser Sequenzdiagramm ist sehr komplex, macht das ganze aber dennoch verständlich, ein überraschend hilfreiches Tool.
18.05.2015	5.0 Std.	#68 Architektur Komponenten bereitstellen: Die client jars mit gradle zu erstellen, war um einiges schwerer als gedacht. Ich und Stefan Beehler haben uns ausgetauscht und konnten den Fehler nicht finden. Unklar ist auch ob die anderen Komponenten sich korrekt verhalten.	Entgegen der Erwartung ist das Deployment sehr aufwendig. Sehr frustrierend.
18.05.2015	1.0 Std.	#21 Dokumentationstemplate einrichten: Damit Sandro ganz einfach den Source Code ausdrucken kann, habe ich ihm ein PowerShell Script geschrieben, das alle Java Files lädt und in ein Word schreibt.	Wenn Java nur so mächtig wie PowerShell wäre. :D
19.05.2015	4.0 Std.	#68 Architektur Komponenten bereitstellen: Stefan war eine grosse Hilfe. Er hat mir gezeigt, wie man mithilfe des Gradle shadowJar plugins doch noch ein runnable Jar erstellen kann. Das War File für den Webservice konnte ohne Probleme verteilt werden.	

21.6 Willi Erwin

Datum Phase	eventuell	Dauer Aufwand	Tätigkeit / Aufgabe	Bemerkung/Erkenntnis
17.03.2014		1.5 Std.	#3 Datenmodell erstellen: Datenbank-Modell erstellen und Tabellenattribute definieren	Es ist nicht klar ersichtlich wie die Beziehungen zu den jeweiligen Instanzen realisiert werden sollen.
18.03.2014		2.0 Std.	#3 Datenmodell erstellen: Datenbank-Modell erstellen und Tabellenattribute definieren	Kardinalitäten definieren und fehlende Tabellen hinzufügen Da Visio nicht geeignet war fürs

			Datenbankmodell übertrag in Workbench-Projekt
26.03.2014	2.0 Std.	#3 Datenmodell erstellen: erstellen des Klassenmodells	
31.03.2014	1 Std	#3 Datenmodell erstellen: fertigstellen des Klassenmodells	
02.04.2014	1 Std.	#14 Requirements Dokumentation anpassen	Schreiben des Authentication Requirements
03.04.2014	1 Std.	#19 Use Case schreiben	Use Case für das Authentication Requirements schreiben
07.04.2014	0.5 Std.	Activity Diagramm erstellen	Activity Diagramm für Authentication erstellen
19.04.2015	1 Std.	#25 Model Sachbearbeiter, Rolle und Mangelstatus	Da ich noch nicht alle Zusammenhänge genau verstehe, was das programmieren angeht, ist es sehr hilfreich anhand von Vorgaben das Ganze selber auszuprobieren
20.04.2015	3 Std.	#31 Seed Task implementieren	Trotz grossem Mehraufwand haben wir uns entschieden nicht mit Fake Daten zu arbeiten, sondern selber Adressen etc. zu erstellen
21.04.2015	1 Std.	#31 Seed Task implementieren	Anpassen der Daten
22.04.2015	1 Std.	#31 Seed Task implementieren	Anpassen der Daten
25.04.2015	2 Std.	#31 Seed Task implementieren	Anpassen der Daten
27.04.2015	3 Std.	#38 end to end testing	Test müssen unabhängig voneinander ausgeführt werden (obwohl Sie sich im gleichen Java File befinden)
30.04.2015	2 Std.	#38 end to end testing	Anpassen der Tests
02.04.2015	6 Std	#38 end to end testing	Problem mit Rückgabotyp der getAll Funktion sobald von Klasse Person geerbt wird
02.04.2015	6 Std	#38 end to end testing	Test Anpassungen und Kontrolle damit auch alle Daten richtig geschrieben werden
04.05.2015	1 Std.	#38 end to end testing	Anpassen der Tests
05.05.2015	2 Std.	#42 Views Implementieren	TimeConvertierung von GregorianCalendar in einen String ist nicht so einfach wie man denkt
07.05.2015	2 Std.	#41 CRUD Filterung auf Webservice	Filterung für Webservice implementieren







08.05.2015	1 Std.	#45 Mangel Referenz Subunternehmen	Anpassen des Datenmodells sowie der dazugehörigen Klassen
10.05.2015	0.5 Std.	#38 end to end testing	Anpassen der Tests
10.05.2015	1 Std.	#41 CRUD Filterung auf Webservice	Filterung für Mangel Implementieren
11.05.2015	3 Std.	#41 CRUD Filterung auf Webservice	Filterung für Projekt Implementieren
14.05.2015	2 Std.	#70 Listen Export CSV	Export Funktion Implementieren
15.05.2015	2 Std.	#70 Listen Export CSV	Quelltext Optimierung: Viele Wege führen nach Rom
15.05.2015	2 Std.	#58 Dokumentation Kapitel 16	
15.05.2015	2 Std.	#60 Dokumentation Kapitel 18	
15.05.2015	1 Std.	#61 Dokumentation Kapitel 19	
16.05.2015	3 Std.	#70 Listen Export CSV	Einfügen der Exportfunktion bei allen Views
16.05.2015	1 Std.	#38 end to end testing	Optimieren bzw. anpassen der Testfunktionen
16.05.2015	2 Std.	#42 Views implementieren	Unterstützung der anderen Teammitgliedern
17.05.2015	5. Std.	#42 Views Implementieren	Unterstützung der anderen Teammitgliedern
18.05.2015	1. Std.	#61 Dokumentation Kapitel 19	Durcheinander aufräumen zwischen funktionellen und nicht funktionellen Tests
18.05.2015	4. Std.	#31 Seed Task implementieren	Anpassen des SeedFiles da Problemem beim erstellen der Datensätze entstanden sind
18.05.2015	3. Std.	#70 Listen Export CSV	Anpassen der verschiedenen Exporte da die Umlaute wider nicht richtig angezeigt wurden Anpassen der Exportierenden Felder
18.05.2015	6. Std.	#31 Seed Task implementieren	Anpassen des SeedFiles da Problemem beim erstellen der Datensätze entstanden sind
19.05.2015	3. Std.	#42 Views Implementieren	Passwort ändern und zurücksetzen Funktion, Fehler suchen

22 Source-Code

Der komplette Source Code ist in einem separaten Ordner abgelegt.

23 Designkonzept

23.1 Farbkonzept

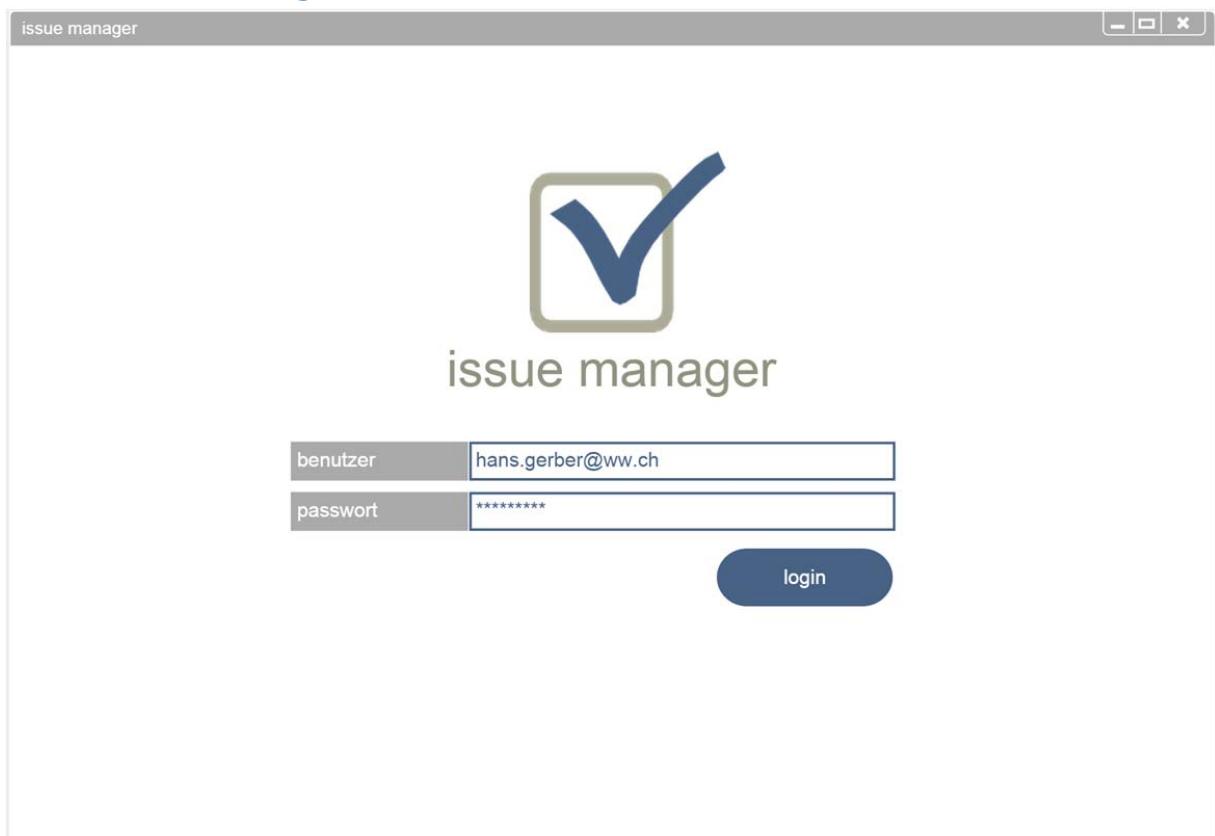
	#FFFFFF	Hintergrund
	#AAAAAA	System, Labels, deaktiviert, Status "abzuklären"
	#939382	Titel, gewählt, Status "beauftragt" "angenommen" "behoben"
	#486285	Aktion, aktiviert, wählbar
	#FF6B68	Überfällig, abbrechen, fehlende/falsche Eingabe
	#719AD1	Status "abgeschlossen", Kommentare

23.2 Logo



Das Logo – eine abgehakte Checkbox – soll das erledigen von Aufgaben symbolisieren.

23.3 GUI01 – Login

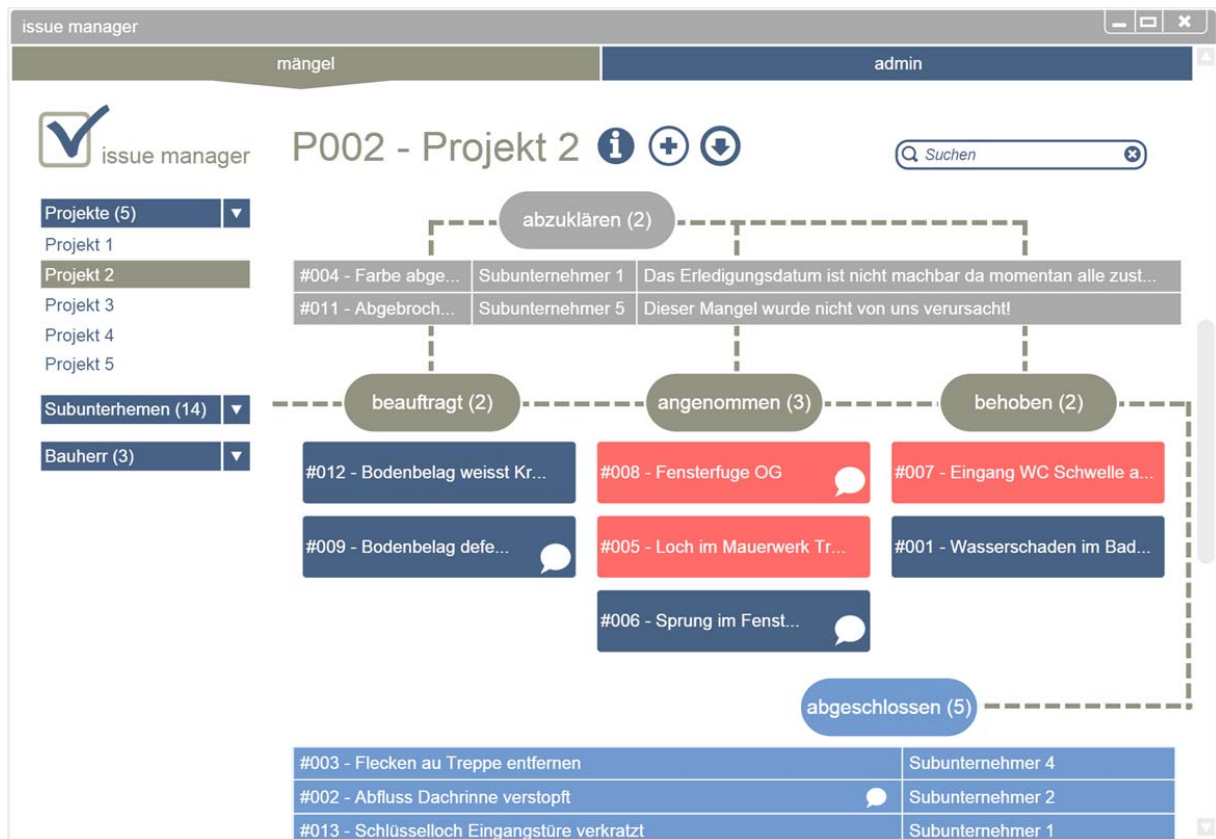


Beschreibung Startseite um die Benutzer zu authentifizieren.

Benutzer

Die Oberfläche ist für alle Benutzer gleich. Nach der Authentifizierung werden die Benutzer gemäss ihrer Rolle auf die Projekteübersicht (SB, KA) oder Mängelübersicht (BL, KP) geführt.

23.4 GUI02 – Mängelübersicht



Beschreibung

Verwaltung von Mängel. Das Design ist angelehnt an die Kanban-Oberfläche, wobei die Mängel verschiedene Status durchlaufen können. Die Änderung des Status verschiebt den Mangel von einer „Lane“ zur anderen.

Benutzer

Bauleiter (und Sachbearbeiter): Die Mängel können im linken Menu nach Projekt, Subunternehmen oder Bauherr aggregiert werden. Über die Icons neben dem Projekttitel kommt man zur Projekt-Seite dieses Projekts (GUI05), kann ein neues Projekt erfassen (ebenfalls GUI05) oder kann die Mängel in eine externe Datei exportieren. Die Mängel können angewählt (und/oder mit der Suchfunktion gesucht) werden, womit der Benutzer auf die Mangel-Seite (GUI03) geführt wird.

Kontaktperson (und Kontaktadmin): Hat die selbe Oberfläche, wobei die Aggregation nach Subunternehmen und Bauherr, sowie die Option ein neues Projekt zu erstellen, wegfallen.

23.5 GUI03 – Mangel

The screenshot shows the 'issue manager' application window. The title bar says 'issue manager'. Below the title bar, there's a navigation bar with 'mängel' and 'admin'. The main content area has a header with a checkmark icon, 'issue manager', and the issue ID 'P002-M006'. Below this, there's a user profile 'Bauleiter 5' and a timestamp '12.04.2015'. The form consists of several fields: 'projekt' (P002 - Neubau Altstätterstrasse 4), 'beschreibung' (Sprung im Fenster OG Zimmer rechts), 'subunternehmen' (Huber Fenster AG), 'status' (angenommen), and 'frist' (empty). There's a 'kommentar' section with a text area containing two comments: 'Das Fenster muss ersetzt werden.' by Gerber Hans and 'Wir werden das Fenster gerne ersetzen, Dies würde jedoch wieder in Rechnung gestellt, da der Schaden nicht von uns verursacht wurde!' by Sutter Stefan. At the bottom, there's a 'kommentar...' input field and two buttons: 'abbrechen' and 'speichern'.

Beschreibung Erfassen und Bearbeiten von Mängel.

Benutzer Bauleiter (und Sachbearbeiter): Dem erfassten Mangel wird vom System eine Referenz (P002-M006 → steht für Projekt 2, Mangel 6), sowie der Name des Erstellers und ein Timestamp gegeben. Das Feld Status hat die Optionen „beauftragt“ und „abgeschlossen“ als Auswahl. Obligatorische Felder, die nicht beschrieben werden, werden rot umrandet. Neue Kommentare werden mit dem hochlade Icon direkt gespeichert; alle anderen Eingaben müssen mit dem Speicher-Button bestätigt, bzw. mit abbrechen verworfen werden. In beiden Fällen wird man zur Mängelübersicht zurückgeführt (GUI02).

Kontaktperson (und Kontaktadmin): Die Felder Projekt, Beschreibung, Subunternehmen und Frist sind inaktive und werden grau eingefärbt. Das Feld Status hat die Optionen „angenommen“, „behoben“ und „abzuklären“ als Auswahl.

23.6 GUI04 – Projekt

The screenshot shows a web application window titled 'issue manager'. At the top, there is a navigation bar with 'mängel' and 'admin' tabs. The main content area displays a form for a project titled 'P002 - Neubau Altstätterstrasse 4'. The form includes the following fields:

- titel:** Neubau Altstätterstrasse 4
- strasse:** Altstätterstrasse 4
- plz:** 6004
- ort:** Luzern
- arbeitstyp:** Renovation (dropdown menu)
- projekttyp:** Mehrfamilienhaus (dropdown menu)
- beginn:** 04.02.2015 (calendar icon)
- ende:** 29.09.2015 (calendar icon)
- bauherr:** CSS Versicherungen - Müller Franz (dropdown menu with plus icon)
- projektleiter:** Gerber Hans
- subunternehmer:** Biregger Sanitär, Dali AG, Fischer Gartenbau, Honegger und Söhne, Huber Fenster AG

At the bottom of the form, there are three buttons: 'abbrechen' (red), 'archivieren' (blue), and 'speichern' (blue).

Beschreibung	Erfassen und Bearbeiten von Projekten.
Benutzer	<p>Sachbearbeiter: Nur der Sachbearbeiter kann neue Projekte erstellen. Die Referenz (hier P002) ergibt sich dabei aus dem ID-Attribut, welches aufsteigend vom System generiert wird. Das Feld Ort wird durch die Postleitzahl automatisch ermittelt. Bauherren können mit dem Dropdown-Menü zugewiesen werden, oder, falls noch nicht vorhanden, über das Plus-Icon neu erstellt werden (verweis auf GUI12). Das Feld Projektleiter und Subunternehmer kann hier nicht verändert werden, sondern führt bei Anwahl zum GUI06, bzw. GUI07. Projekte können archiviert werden.</p> <p>Bauleiter: Der Unterschied zum Sachbearbeiter ist, dass der Bauleiter keine neuen Projekte und Bauherren erstellen, sowie nicht archivieren kann.</p> <p>Kontaktadmin: Siehe GUI05.</p> <p>Kontaktperson: Siehe GUI05.</p>

23.7 GUI05 – Projekt (KA)

issue manager

mängel admin

issue manager

P002 - Neubau Altstätterstrasse 4

titel	Neubau Altstätterstrasse 4		
strasse	Altstätterstrasse 4		
plz	6004	ort	Luzern
arbeitstyp	Renovation ▼		
projekttyp	Mehrfamilienhaus ▼		
beginn	04.02.2015		
ende	29.09.2015		
bauherr	CSS Versicherungen - Müller Franz ▼		
projektmanager	Gerber Hans		
kontaktperson	Sutter Stefan ▼		

abbrechen speichern

Beschreibung Erfassen und Bearbeiten von Projekten.

Benutzer Sachbearbeiter: Siehe GUI04.

Bauleiter: Siehe GUI04.

Kontaktadmin: Alle Felder, bis auf Projektleiter und Kontaktperson, sind inaktiv und dienen nur der Information. Das Feld Projektleiter führt zum GUI06. Das Zuweisen einer Kontaktperson ist die Hauptaufgabe des Kontaktadmin und kann auf dieser Seite vorgenommen werden.

Kontaktperson: Das Feld Kontaktperson ist ebenfalls inaktiv. Die ganze Seite ist somit nur Information, womit auch die Buttons abbrechen und speichern wegfallen.

23.8 GUI06 – Projektleitung



bauleiter	beginn	ende
Gerber Hans	12.04.2015	
Hermann Thomas	01.03.2015	12.04.2015
Lutz Christian	04.02.2015	01.03.2015
neuer bauleiter...		

abbrechen speichern

Beschreibung Erfassen und Bearbeiten der Projektleitung.

Benutzer Bauleiter (und Sachbearbeiter): Hier werden die verantwortlichen Bauleiter eines Projektes und die Periode erfasst, in der sie verantwortlich sind. Einmal erfasst, können nur noch die Perioden angepasst werden. Durch anwählen der Bauleiter gelangt man auf die Kontaktdetails (siehe Beschreibung im GUI12).

Kontaktperson (und Kontaktadmin): Die ganze Seite ist nur Information, womit die Buttons abbrechen und speichern wegfallen. Die Perioden sind inaktiv.

23.9 GUI07 – Zugewiesene Subunternehmen

issue manager

mängel admin

issue manager

P002 - subunternehmen

subunternehmen	kontaktperson
Biregger Sanitär	Weiss Roman ▼
Dali AG	Durrer Sepp ▼
Fischer Gartenbau	Bieri Sandro ▼
Honegger und Söhne	Honegger Peter ▼
Huber Fenster AG	Sutter Stefan ▼
neues subunternehmen...	▼

abbrechen speichern

Beschreibung	Zuweisen der Subunternehmen.
Benutzer	<p>Bauleiter (und Sachbearbeiter): Hier werden die betroffenen Subunternehmen eines Projektes erfasst. Die Zuweisung der Kontaktperson wird generell vom verantwortlichen Kontaktadmin (in GUI05) vorgenommen. Der Bauleiter (und Sachbearbeiter) kann hier jedoch ebenfalls Zuweisungen vornehmen (falls unbedingt nötig). Einmal erfasst, können nur noch die Kontaktpersonen angepasst werden. Durch anwählen der Subunternehmen gelangt man auf Details (siehe Beschreibung im GUI10).</p> <p>Kontaktperson und Kontaktadmin: Haben keine Einsicht auf dieses GUI.</p>

23.10 GUI08 – Projekteübersicht

The screenshot shows the 'issue manager' application window. The title bar says 'issue manager'. Below it, there are two tabs: 'mängel' (selected) and 'admin'. The main content area has a header with a checkmark icon, 'issue manager', and 'Projekte' with a plus icon. To the right of 'Projekte' is a search bar with the text 'Suchen'. Below the header, there is a table with the following columns: 'Projektname', 'Projekttyp', 'Arbeitstyp', 'Bauleiter', 'Bauherr', 'Enddatum', and 'Offene Mängel'. The table contains 12 rows of data. On the left side of the table, there are three filter buttons: 'Projekte (5)', 'Subunterhemen (14)', and 'Person (14)', each with a dropdown arrow.

▼ Projektname	▼ Projekttyp	▼ Arbeitstyp	▼ Bauleiter	▼ Bauherr	▼ Enddatum	▼ Offene Mängel
Projekt xy	Wohung	Neubau	Muster Max	Müller Thomas	21.03.2017	15
Projekt ab	Einfamilienhaus	Rennovation	Meier Andreas	Firma xy	15.06.2015	3
Projekt xy	Wohung	Neubau	Muster Max	Müller Thomas	21.03.2017	15
Projekt ab	Einfamilienhaus	Rennovation	Meier Andreas	Firma xy	15.06.2015	3
Projekt xy	Wohung	Neubau	Muster Max	Müller Thomas	21.03.2017	15
Projekt ab	Einfamilienhaus	Rennovation	Meier Andreas	Firma xy	15.06.2015	3
Projekt xy	Wohung	Neubau	Muster Max	Müller Thomas	21.03.2017	15
Projekt ab	Einfamilienhaus	Rennovation	Meier Andreas	Firma xy	15.06.2015	3
Projekt xy	Wohung	Neubau	Muster Max	Müller Thomas	21.03.2017	15
Projekt ab	Einfamilienhaus	Rennovation	Meier Andreas	Firma xy	15.06.2015	3
Projekt xy	Wohung	Neubau	Muster Max	Müller Thomas	21.03.2017	15
Projekt ab	Einfamilienhaus	Rennovation	Meier Andreas	Firma xy	15.06.2015	3

Beschreibung Dient zur Stammdatenverwaltung von Projekten. Folgende Möglichkeiten bietet diese Seite an:

- neue Projekte erfassen
- nach Projekten suchen
- bestehende Projekte anschauen

Benutzer Sachbearbeiter: Seine Sicht ist genau wie oben dargestellt und hat die Rechte alles auszuführen.

Bauleiter: Hat nur auf seine Projekte Zugriff und kann links alle drei Auswahlfelder wählen. Einzig das Plus-Symbol zur Erstellung neuer Projekt fällt bei ihm weg.

Kontaktadmin: Hat nur Zugriff auf Projekte, welche seinem Subunternehmen zugewiesen sind. Bei der Bearbeitung eines Projekts kann der Kontaktadmin nur eine Zuweisung von Kontaktpersonen auf Projekt vornehmen (GUI05).

Kontaktperson: Hat nur Zugriff auf Projekte, welche seinem Subunternehmen zugewiesen sind.

23.11 GUI09 – Subunternehmenübersicht

issue manager

mängel admin

issue manager Subunternehmen

▼ Firma	▼ Strasse	▼ PLZ	▼ Ort	▼ Telefon
Huber Fenster AG	Zentralstrasse 1	6002	Luzern	041 250 00 00
Biregger Sanitär	Pilatusstrasse 3	6005	Luzern	031 260 55 55
Fischer Gartenbau	Luzernerstrasse 14	6014	Luzern	041 250 00 00
Huber Fenster AG	Zentralstrasse 1	6002	Luzern	031 260 55 55
Biregger Sanitär	Pilatusstrasse 3	6005	Luzern	041 250 00 00
Fischer Gartenbau	Luzernerstrasse 14	6014	Luzern	031 260 55 55
Huber Fenster AG	Zentralstrasse 1	6002	Luzern	041 250 00 00
Biregger Sanitär	Pilatusstrasse 3	6005	Luzern	031 260 55 55
Fischer Gartenbau	Luzernerstrasse 14	6014	Luzern	041 250 00 00
Huber Fenster AG	Zentralstrasse 1	6002	Luzern	031 260 55 55

Beschreibung Dient zur Stammdatenverwaltung von Subunternehmen. Folgende Möglichkeiten bietet diese Seite an:

- neue Subunternehmen erfassen
- nach Subunternehmen suchen
- bestehendes Subunternehmen anschauen

Benutzer

Sachbearbeiter: Seine Sicht ist genau wie oben dargestellt und hat die Rechte alles auszuführen.

Bauleiter: Hat nur Zugriff auf Subunternehmen, die mit seinen Projekten verknüpft sind und kann links alle drei Auswahlfelder wählen. Einzig das Plus-Symbol zur Erstellung von neuen Subunternehmen fällt bei ihm weg.

Kontaktadmin und Kontaktperson: Hat nur auf sein Subunternehmen Zugriff.

23.12 GUI10 – Subunternehmen

The screenshot shows a web application window titled 'issue manager'. The top navigation bar has 'mängel' and 'admin' tabs. The main content area displays the 'Huber Fenster AG' form with the following fields:

- firma: Huber Fenster AG
- strasse: Zentralstrasse 1
- plz: 6002, ort: Luzern
- telefon: 041 250 00 00

Below the form are two buttons: 'abbrechen' (red) and 'speichern' (blue). Underneath is a section titled 'Kontakte' with a plus icon, followed by a table of contacts.

▼ Nachname	▼ Vorname	▼ E-Mail	▼ Telefon
Muster	Peter	peter.muster@hufest.ch	041 250 00 01
Müller	Thomas	thomas.müller@hufest.ch	041 250 00 12
Meier	Andreas	meier.andreas@hufest.ch	041 250 00 16
Muster	Peter	peter.muster@hufest.ch	041 250 00 01
Müller	Thomas	thomas.müller@hufest.ch	041 250 00 12

Beschreibung	Zur Erstellung oder Bearbeitung von Subunternehmen. Ausserdem können hier die Kontakte eines Subunternehmens eingesehen werden, falls diese vorgängig erfasst und zugewiesen wurden. Hier besteht die Möglichkeit direkt neue Kontakte zu erfassen und dabei wird das Feld Subunternehmen vorgefüllt (siehe GUI13).
Benutzer	<p>Sachbearbeiter: Seine Sicht ist genau wie oben dargestellt und hat die Rechte alles auszuführen.</p> <p>Bauleiter: Kann Subunternehmen, die mit seinem Projekt verknüpft sind, ansehen und bearbeiten, aber er sieht das Plus-Symbol nicht und kann somit keine Kontakte selber hinzufügen.</p> <p>Kontaktadmin: Kann Subunternehmen und seine Kontakt nur ansehen. Die Zuweisung von Kontaktpersonen auf Subunternehmen erfolgt in einer anderen Seite (GUI05).</p> <p>Kontaktperson: Kann Subunternehmen und seine Kontakt nur ansehen.</p>

23.13 GUI11 – Personenübersicht

issue manager

mängel admin

issue manager Person +

Suchen

Nachname	Vorname	E-Mail	Rolle	Firma
Muster	Peter	peter.muster@wwarch.ch	Bauleiter	-
Müller	Thomas	thomas.müller@hufest.ch	Kontaktadmin	Huber Fenster AG
Meier	Andreas	meier.andreas@biregsan.ch	Kontaktperson	Biregger Sanitär
Muster	Peter	peter.muster@wwarch.ch	Bauleiter	-
Müller	Thomas	thomas.müller@hufest.ch	Kontaktadmin	Huber Fenster AG
Meier	Andreas	meier.andreas@biregsan.ch	Kontaktperson	Biregger Sanitär
Muster	Peter	peter.muster@wwarch.ch	Bauleiter	-
Müller	Thomas	thomas.müller@hufest.ch	Kontaktadmin	Huber Fenster AG
Meier	Andreas	meier.andreas@biregsan.ch	Kontaktperson	Biregger Sanitär
Muster	Peter	peter.muster@wwarch.ch	Bauleiter	-

Beschreibung	<p>Dient zur Stammdatenverwaltung von Personen. Folgende Möglichkeiten bietet diese Seite an:</p> <ul style="list-style-type: none"> - neue Person erfassen - nach Person suchen - bestehende Person anschauen
Benutzer	<p>Sachbearbeiter: Seine Sicht ist genau wie oben dargestellt und hat die Rechte alles auszuführen.</p> <p>Bauleiter: Kann Personen ansehen und bearbeiten, die mit seinem Projekt verknüpft sind, aber er sieht das Plus-Symbol nicht.</p> <p>Kontaktadmin: Kann Personen ansehen, die mit seinem Subunternehmen verknüpft sind. Aber er sieht das Plus-Symbol nicht und kann somit keine Kontakte selber hinzufügen.</p> <p>Kontaktperson: Kann nur seine eigene Person ansehen.</p>

23.14 GUI12 – Bauherr

The screenshot shows a web application window titled 'issue manager'. The window has a dark blue header bar with 'mängel' on the left and 'admin' on the right. Below the header, there is a logo with a checkmark and the text 'issue manager'. The main content area is titled 'Meier Andreas'. It contains a form with the following fields:

- nachname: Meier
- vorname: Andreas
- email: meier.andreas@biregsan.ch
- telefon: 031 430 22 88
- rolle: Bauherr (dropdown menu)
- firma: (empty)
- strasse: (empty)
- plz: (empty)
- ort: (empty)

At the bottom right of the form, there are two buttons: 'abbrechen' (red) and 'speichern' (blue).

Beschreibung	Verwaltung einer Person ist je nach Auswahl der Rolle unterschiedlich. Die Darstellung oben zeigt wie es aussieht, wenn Bauherr als Rolle ausgewählt wird. Falls Bauleiter angewählt wird, fallen die Felder unterhalb des Feldes Rolle weg.
Benutzer	<p>Sachbearbeiter: Seine Sicht ist genau wie oben dargestellt und hat die Rechte alles auszuführen.</p> <p>Bauleiter: Kann Bauherren nur ansehen.</p> <p>Kontaktadmin: Kommt nicht in diese Ansicht. Sieht den Bauherren lediglich über das Projekt.</p> <p>Kontaktperson: Kommt nicht in diese Ansicht. Sieht den Bauherren lediglich über das Projekt.</p>

23.15 GUI13 – Kontaktadmin

The screenshot shows a web application window titled 'issue manager'. At the top, there is a navigation bar with two tabs: 'mängel' (selected) and 'admin'. Below the navigation bar, the main content area displays a form for editing a contact. The form is titled 'Meier Andreas' and includes the following fields: 'nachname' (Meier), 'vorname' (Andreas), 'email' (meier.andreas@biregsan.ch), 'telefon' (031 430 22 88), 'rolle' (Kontaktadmin, selected from a dropdown), 'passwort' (masked with asterisks), and 'subunternehmen' (empty). Below the form, there are three buttons: 'abbrechen' (red), 'passwort ändern' (blue), and 'speichern' (blue). A small plus icon is visible next to the 'subunternehmen' field.

Beschreibung Verwaltung einer Person ist je nach Auswahl der Rolle unterschiedlich. Die Darstellung oben zeigt wie es aussieht, wenn Kontaktadmin als Rolle ausgewählt wird.

Benutzer

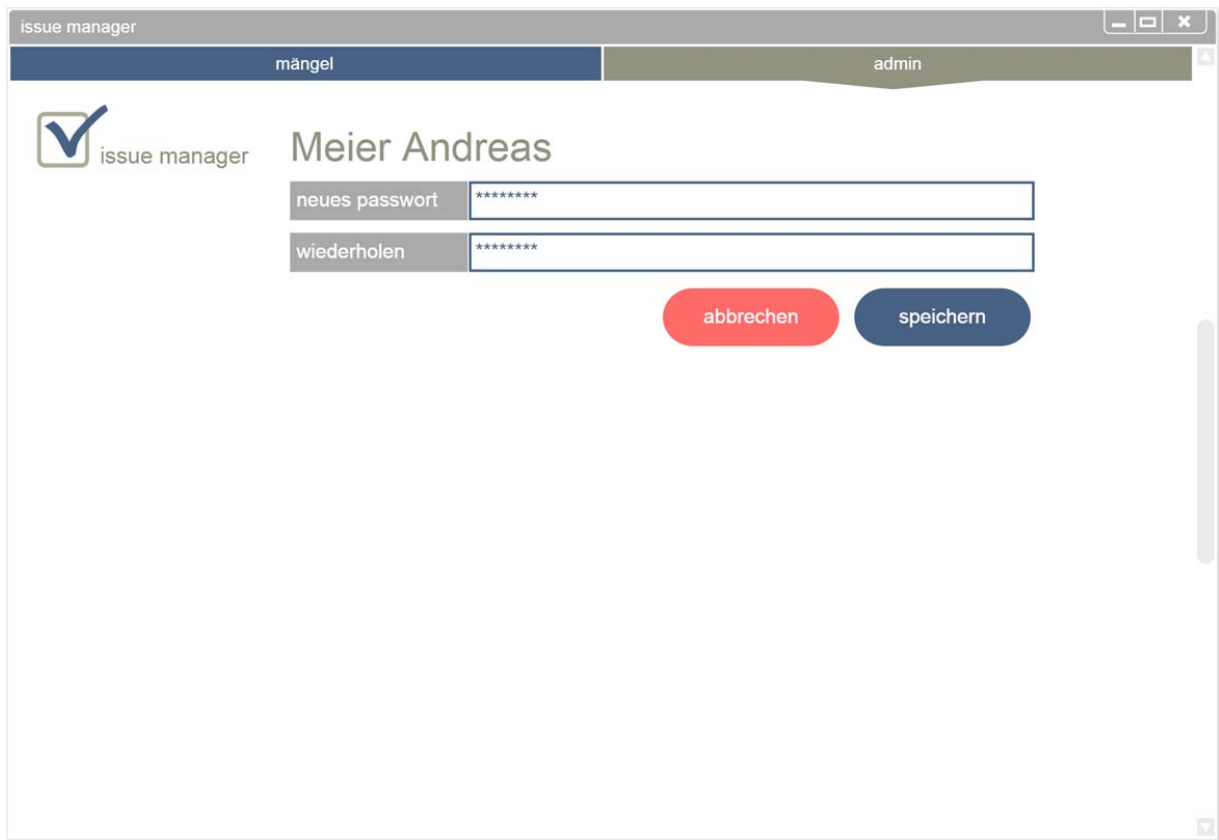
Sachbearbeiter:
Seine Sicht ist genau wie oben dargestellt und hat die Rechte alles auszuführen.

Bauleiter:
Kann Kontaktadmin nur ansehen.

Kontaktadmin:
Kann nur sich selber in dieser Sicht ansehen.

Kontaktperson:
Kann nur sich selber in dieser Sicht ansehen.

23.16 GUI14 – Passwort



issue manager

mängel admin

✓ issue manager

Meier Andreas

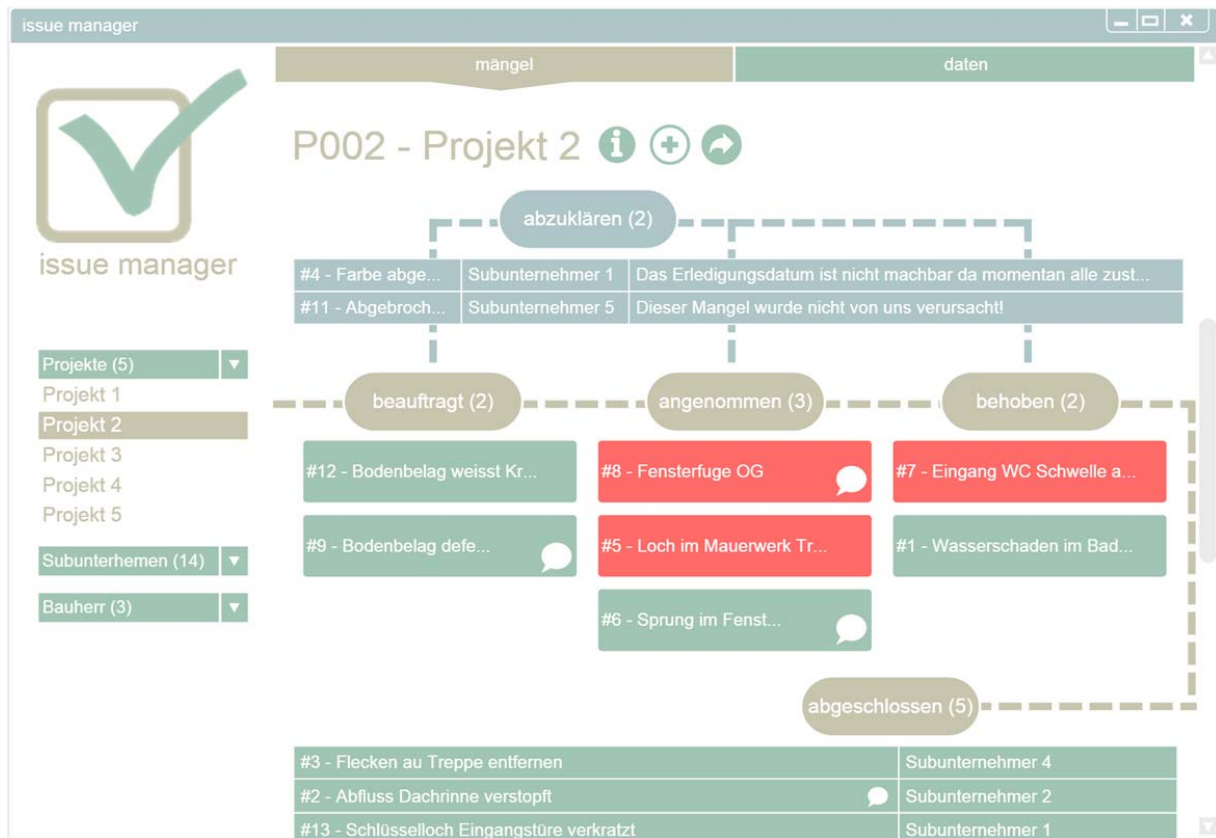
neues passwort *****

wiederholen *****

abbrechen speichern

Beschreibung	Passwort eines Benutzers ändern.
Benutzer	<p>Sachbearbeiter: Seine Sicht ist genau wie oben dargestellt und hat die Rechte alles auszuführen. Er kann Passwort aller Benutzer ändern.</p> <p>Bauleiter: Kann sein eigenes Passwort ändern.</p> <p>Kontaktadmin: Kann sein eigenes Passwort ändern.</p> <p>Kontaktperson: Kann sein eigenes Passwort ändern.</p>

23.17 Verworfenne Designs



Issue Manager

issue manager

Projekte (4)

Projekt 1

Projekt 2

Projekt 3

Projekt 4

Projekt 5

Subunterhemen (14)

Bauherr (3)

Issues

Daten

Projekt 2

abzuklären

#4 - Farbe abge...

Subunternehmer 1

K: Das Erledigungsdatum ist nicht machbar da momentan alle zust...

beauftragt

angenommen

behooben

#12 - Bodenbelag weisst Kr...

#9 - Bodenbelag defe...

#11 - Plattenfugen unsaube...

#8 - Fensterfuge OG

#5 - Loch im Mauerwerk Tr...

#6 - Sprung im Fenst...

#10 - Wasserschaden im Ba...

#7 - Eingang WC Schwelle a...

#1 - Wasserschaden im Bad...

abgeschlossen

#3 - Flecken au Treppe entfernen

#2 - Abfluss Dachrinne verstopft

#13 - Schlüsselloch Eingangstüre verkratzt

Subunternehmer 4

Subunternehmer 2

Subunternehmer 1

24 Logging

Die Konfiguration von Tinylog befindet sich auf der Webservice-Schicht unter der Klasse Service.java, damit es global auf der ganzen Schicht verwendet werden kann.

```

1. public static void runLogConfigurator(){
2.     Configurator.defaultConfig()
3.         .writer(new RollingFileWriter(ConfigHelper.getConfig("tinylog.location", "log.txt")
4.             ,
5.             ConfigHelper.getConfig("tinylog.files", 10),
6.             new TimestampLabeler(ConfigHelper.getConfig("tinylog.timestamp", "yyyy-MM-
7.                 dd_HH-mm-ss")),
8.             new SizePolicy(ConfigHelper.getConfig("tinylog.size", 10240))))
9.         .level(Level.valueOf(ConfigHelper.getConfig("tinylog.level", "INFO")))
10.        .formatPattern(ConfigHelper.getConfig("tinylog.format", "{level}:\t{date}\t{class}.
11.            {method}()\t{message}"))
12.        .activate();
13.    if(ConfigHelper.getConfig("tinylog.logtoconsole", "No").equals("Yes")){
14.        Configurator.currentConfig().addWriter(new ConsoleWriter()).activate();
15.    }
16. }
```

Zeile	Element	Beschreibung
2	defaultConfig()	Nimmt die Standard-Konfiguration als Grundlage
3	writer()	Ist für die Ausgabe der Log-Einträge zuständig. Der RollingFileWriter bietet die Verwaltung der Logs nach dem Logrotate-Prinzip an. Hier kann der Zielordner und Name, Angabe von maximalen Files von 10 und Grösse der Files angegeben werden. Standardmässig wird es in der Konsole ausgegeben, aber wir haben uns für eine Hybrid-Lösung entschieden. Die Ausgabe auf der Konsole (Zeile 12) erleichtert den Administratoren und auch den Entwicklern Probleme direkt beim Starten zu sehen.
7	level()	Wir haben Log-Level „INFO“ ausgewählt, um schnell Fehler finden zu können. Zu einem späteren Zeitpunkt kann das problemlos auf „ERROR“ gesetzt werden.
8	formatPatter()	Hier gebe ich an wie das Logfile aufgebaut sein soll.
9	activate()	Wenn eine Configuration definiert wurde, muss es immer mit dieser Methode aktiviert werden.
13	currentConfig()	Holt die aktuelle Konfiguration, um es zum Beispiel zu erweitern in speziellen Fällen. In unserem Fall wird die zusätzliche Ausgabe auf der Konsole hinzugefügt.

25 Beispielapplikation

Zur Vorbereitung und Erarbeitung von Programmierkenntnissen wurde im Vorfeld eine Beispielapplikation entwickelt. Die Erfahrungen und entwickelten Komponenten wurden anschliessend für dieses Projekt übernommen.

Die Dokumentation zu dieser Applikation ist online verfügbar:

<https://janikvonrotz.ch/2015/03/15/build-a-java-3-tier-application-from-scratch-part-1-introduction-and-project-setup/>