Projektmanagement-Plan

Hochschule Luzern
Technik & Architektur

Applikationsentwicklung – FS14

Gruppe 20 | Horw, 29.05.2014
Bontekoe Christian | Estermann Michael | Rohrer Felix

Autoren

Bontekoe Christian	
Studiengang	Informatik - Software Systems (Berufsbegleitend)
Adresse	
Telefon	
E-Mail	

Estermann Michael	
Studiengang	Informatik - Software Systems (Berufsbegleitend)
Adresse	
Telefon	
E-Mail	

Rohrer Felix	
Studiengang	Informatik - Software Systems (Berufsbegleitend)
Adresse	
Telefon	
E-Mail	

Änderungskontrolle

Version	Datum	Autor	Beschreibung
1.0	28.03.2014	Felix Rohrer	Vorlage erstellen, erste Texte schreiben
1.1	03.04.2014	Felix Rohrer	Ergänzungen für Review 1
1.2	10.04.2014	Felix Rohrer	Ergänzungen für Review 2: Sprint 2 Planung
1.3	24.04.2014	Felix Rohrer	Ergänzungen für Review 2
1.4	24.04.2014	Christian Bontekoe	Risikomanagement erstellt
1.5	24.04.2014	Christian Bontekoe	Testplan/ Testfälle erstellt
1.6	08.05.2014	Felix Rohrer	Ergänzungen für Review 3
1.7	22.05.2014	Felix Rohrer	Ergänzungen für Review 4
1.8	29.05.2014	Felix Rohrer	Finalisieren für Abgabe

Inhalt

1 Projektorganisation

1.1 Organisationsplan

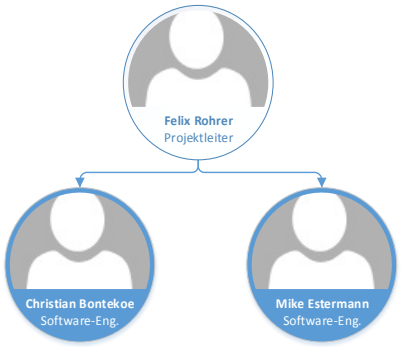


Abbildung 1: Organisationsstruktur

1.2 Rollen & Zuständigkeiten

1.2.1 Projektsicht

Rolle	Verantwortung	Beschreibung
Projektmanagement	Felix	Ist für das Projektmanagement verantwortlich.
Requirement-Eng.	Christian	Klärt die funktionalen und nicht funktionalen Anforderungen ab.
SW-Architektur	Mike	Plant die SW-Architektur und erstellt das Grob-Konzept dazu. Er überwacht, dass dieses umgesetzt wird.
SW-Entwicklung	Mike	Ist für die SW-Entwicklung verantwortlich. Die einzelnen Entwicklungssichten wurden noch einzeln aufgeteilt.
Test-Management	Christian	Definiert das Test-Vorgehen und stellt sicher, dass dieses angewendet wird.
Dokument-Mgmt.	Felix	Erstellt das Layout der Dokumente und überwacht, dass dieses eingehalten wird.

Tabelle 1: Zuständigkeiten - Projektsicht

1.2.2 Entwicklungssicht

Rolle	Verantwortung	Beschreibung
Presentation-Tier	Christian	Die Präsentationsschicht ist verantwortlich für die Darstellung der Funktionsblöcke sowie der generierten Resultate. Konkret handelt es sich um das GUI der Anwendung.
Logic-Tier	Mike	Die Logikschicht implementiert die Geschäftsprozesse. Sie ist auch Vermittlungsschicht zwischen Präsentations- und Datenschicht.
Data-Tier	Felix	Die Datenschicht stellt die Datengrundlage des Systems dar. Dazu wird ein relationales Datenbanksystem eingesetzt.

Tabelle 2: Zuständigkeiten - Entwicklungssicht

2 Projektführung

2.1 Rahmenplan

Nr.	Vorga	Vorgangsname	Dauer	Anfang	Ende	Vorgän
1		Projektstart	0 Tage	Don 20.02.14	Don 20.02.14	
2		Input	22 Tage	Don 20.02.14	Fre 21.03.14	
3		**Sprint 1**	**15 Tage**	**Fre 14.03.14**	**Don 03.04.14**	
5		Sprint Review 1	0 Tage	Don 03.04.14	Don 03.04.14	
6		**Sprint 2**	**15 Tage**	**Fre 04.04.14**	**Don 24.04.14**	5
8		Sprint Review 2	0 Tage	Don 24.04.14	Don 24.04.14	
9		**Sprint 3**	**10 Tage**	**Fre 25.04.14**	**Don 08.05.14**	8
11		Sprint Review 3	0 Tage	Don 08.05.14	Don 08.05.14	
12		**Sprint 4**	**10 Tage**	**Fre 09.05.14**	**Don 22.05.14**	11
14		Sprint Review 4	0 Tage	Don 22.05.14	Don 22.05.14	
15		Projektabschluss	0 Tage	Mit 28.05.14	Mit 28.05.14	

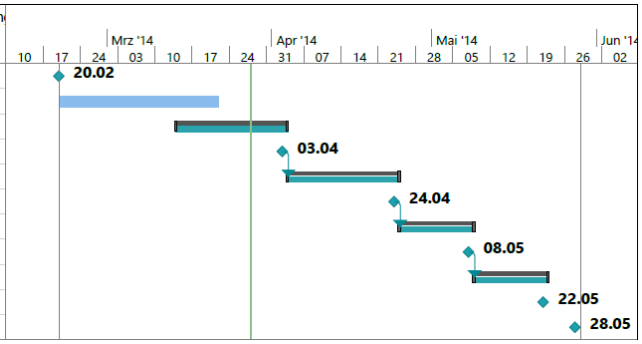


Abbildung 2: Rahmenplan

2.2 Risikomanagement

2.2.1 Stand 22.05.2014

Nr.	Risiko	EW	S/A	Vermeidungsstrategie
N2	Dokumentation ist zu gegebenem Termin nicht fertiggestellt	1	1	Laufende Arbeiten wöchentlich überprüfen Wöchentlich dokumentieren Aufgaben für das Schreiben von Themenbereiche definieren
N3	Zu viele Definitionen, zu wenig Realisierung	2	2	Nicht nur Programmieraufgaben verteilen, sondern auch Themenbereiche für die Dokumentation verteilen Wöchentlich dokumentieren
T2	Software ist zu gegebenem Termin nicht fertiggestellt	2	1	Kunden frühzeitig informieren Wenn möglich Termin verschieben
T3	Software erfüllt Anforderungen nicht	1	2	Anforderungen regelmässig überprüfen Bei Unklarheiten sofort nachfragen

Nr.: Tx: Technisches / Nx: Nicht-technisches Risiko

EW: Eintrittswahrscheinlichkeit: 1 tiefe, 3 hohe EW

S/A: Schaden/Auswirkung: 1 kleine, 3 grosse Auswirkungen

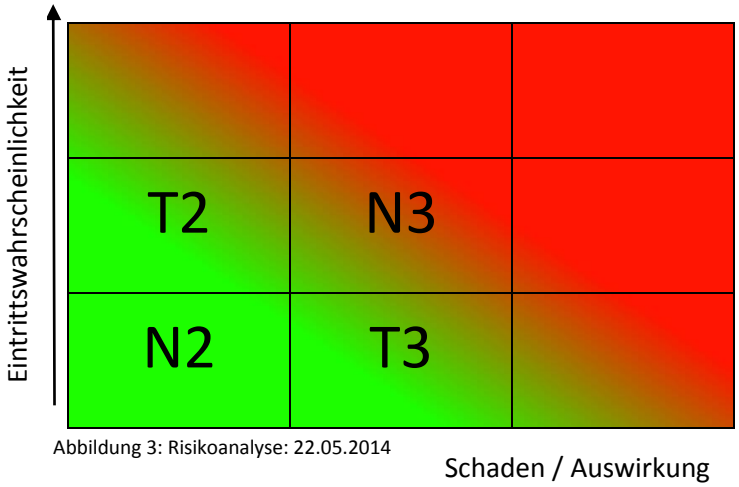


Abbildung 3: Risikoanalyse: 22.05.2014

3 Projektunterstützung

3.1 Tools für Entwicklung, Test & Abnahme

Bereich	Tool
Scrum-Tool	ScrumDo (Online)
ERM- / ERD-Designer	MySQL Workbench 6.1 / Microsoft Visio 2013
UML-Designer	Microsoft Visio 2013
Entwicklungsumgebung	Eclipse, Kepler Service Release 1
Source-Code Verwaltung	SVN, Subversion 1.6.17
Source-Code Style	CheckStyle / PMD / JaCoCo (Coverage)
Build-Server	ANT, Jenkins ver. 1.558
Test	JUnit / manuelle Integrations-Test
Dokumentation	Microsoft Word 2013

Tabelle 3: Projektunterstützung- Tool Übersicht

3.2 Konfigurationsmanagement

Das Konfigurationsmanagement richtet sich nach der definierten Infrastruktur. Die Software Artefakte werden auf dem Repository abgelegt. Die Version basiert auf dem Majorrelease, dem Minorrelease und der Revisionnummer.

Final Release ist die Version 1.0.0

Die Dokumentenverwaltung wird über Dropbox geregelt. Zusätzlich zum Titelblatt mit der jeweiligen Version wird diese auch im Dateiname verwaltet. Format _YYYYMMDDv#.#

3.3 QS & Prozessoptimierung

Die Qualitätssicherung wird durch die Team Mitglieder gewährleistet. Der erstellte Source-Code, das Programm wird mittels JUnit Tests geprüft. Zusätzlich kommt CheckStyle und PMD zum Einsatz. Entsprechend wird dann der Code optimiert. Die Dokumentation wird jeweils von einer Zweitperson beurteilt.

Prozessverbesserungen werden durch teaminterne Diskussionen vorgenommen.

4 Testplan

Es wird nach Möglichkeit das Prinzip Test-First umgesetzt. Unit-Tests werden innerhalb der Entwicklungsumgebung erstellt. Unit-Tests werden jeweils nach Bearbeitung resp. Änderungen im Code getestet. Unit Tests sollen auf Stufe Klasse unabhängig von den anderen Komponenten durchgeführt werden.

Die Entwickler testen das Zusammenspiel der Komponenten zusammen in System- und Integrationstest. Ein Testprotokoll soll auf Programmier- oder Überlegungsfehler aufmerksam machen.

4.1 Testdesign & Abläufe

4.1.1 Unit-Tests

Zu möglichst jeder Klasse soll ein Unit-Test erstellt werden. Die Tests werden vor der jeweiligen Entwicklung erstellt und werden laufend ergänzt und angepasst. Zu einem späteren Zeitpunkt werden die erfolgreich getesteten Softwareteile zusammengefügt und mit einem Integrations- und Systemtest auf Fehler über-prüft.

4.1.2 Test First Methode

Die geplanten Tests sollen im Rahmen dieses Projektes nach der Test First Methode durchgeführt werden. Diese beschreibt, dass die Testfälle vor der Implementation geschrieben werden sollen. Indem man sich zuerst die Testfälle wie z.B. Spezialfälle oder Extremwerte überlegt, kann die Implementation des Programmcodes bereits bei der Produktion optimiert werden. Der Programmcode reift durch dieses Verfahren förmlich heran.

4.1.3 Integrationstest

Beim Integrationstest ist es wichtig, dass die einzelnen Funktionen und Komponenten bereits im Voraus mit Unit Tests geprüft wurden. Denn beim Integrationstest werden die einzelnen Komponenten zusammengeführt und auf das korrekte Zusammenspiel geprüft. Die Integrationstests prüfen somit die von Beginn weg definierten Schnittstellen und deren Kompatibilität.

4.1.4 Systemtest

Wenn das Testen des Zusammenspiels der Teilkomponenten erfolgreich war, folgt schlussendlich der Systemtest. Beim Systemtest wird das System als Ganzes zusammengeführt und getestet. Beim Systemtest wird nach der Spezifikation getestet, somit kann die Korrektheit der Funktionen sichergestellt werden.

4.2 Testfälle

Testfall	**#01 | Start GUI**
Beschreibung	Die Applikation kann fehlerfrei gestartet werden. Die Verbindung zum Backend funktioniert und der User kann sich ein loggen.
Vorgehen	- Applikation starten - Mit gültigem Username / Passwort anmelden
Voraussetzung	Backend muss mit der DB kommunizieren können.
Erwartetes Resultat	Die Applikation lässt sich ohne Fehlermeldungen starten und das Login wird akzeptiert.

Testfall	**#02 | Show Customer**
Beschreibung	Bestehende Kunden können im GUI dargestellt werden.
Vorgehen	- GUI starten inkl. Anmelden - „Kunde“ auswählen - „Kunde suchen“ anklicken - Adressen werden im oberen Bereich dargestellt
Voraussetzung	#01
Erwartetes Resultat	Vorhandene Kundenadressen werden korrekt dargestellt.

Testfall	**#03 | Add Customer**
Beschreibung	Es kann ein neuer Kunde im GUI erfasst werden.
Vorgehen	- GUI starten inkl. Anmelden - „Kunde“ auswählen - „Kunde suchen“ anklicken - „+“ anklicken - Angaben zum Kunden eintragen - <<grüner Häkchen>> Button anklicken um zu speichern
Voraussetzung	#01
Erwartetes Resultat	Der neu eingetragene Kunde muss korrekt in der DB gespeichert werden und wird anschliessend im GUI dargestellt.

Testfall	**#04 | View Product**
Beschreibung	Produkte können angezeigt werden
Vorgehen	- GUI starten - Unter „Produktbestellung“ werden alle aktiven Produkte angezeigt
Voraussetzung	#01
Erwartetes Resultat	Vorhandene aktive Produkte in der DB werden korrekt im GUI dargestellt

Testfall	**#05 | Add Order**
Beschreibung	Es kann eine Bestellung ausgelöst werden.
Vorgehen	- GUI starten - „Produktbestellung“ auswählen - Produkte und Menge auswählen und mittels „Bestellung eintragen“ hinzufügen - Es können mehrere Produkte hinzugefügt werden (siehe Punkt vorhin) - „Kund“ resp. „Weiter zum Kunde“ anklicken - „Kunde suchen“ anklicken - Gewünschter Kunde auswählen und mittels „OK“ bestätigen - Falls gewünscht „Rechnungsadresse“ und „Lieferadresse“ auswählen - „Bestellung“ resp. „Zu den Bestellungen“ klicken - In der Übersicht werden die ausgewählten Produkte nochmals angezeigt - Mittels „x“ kann eine Position wieder entfernt werden - Durch Anklicken von „Bestellung bestätigen“ wird die Bestellung ausgelöst.
Voraussetzung	#01
Erwartetes Resultat	Es kann eine neue Bestellung erstellt werden. Einzelne Produkte können der Bestellung hinzugefügt werden sowie der Kunde wird angegeben. Die ganze Bestellung mit allen Angaben wird korrekt in der DB gespeichert und im GUI dargestellt.

4.3 Testprotokoll

1 Referenzen

- Testplan (APPE_G20_PMP_20140529v1.8)

2 Test

2.1 Angaben zum Test

Build Version	Tester	Datum der Testdurchführung
FBSData: 1.0.0 FBSService: 1.0.0 FBSGuiClient: 1.0.0	Felix Rohrer	29.05.2014

2.2 Zusammenfassung Ergebnis

# Test durchgeführt?	# Tests erfolgreich?	# Tests fehlgeschlagen?
5	5	0

2.3 Ergebnisse Tests

Was	OK / nicht OK	Aufgetretene Fehler / Bemerkung
Unit-Tests	OK	FBSData: 54 JUnit-Tests erfolgreich FBSService: 36 JUnit-Tests erfolgreich
Systemtest 1: **Start GUI**	OK	
Systemtest 2: **Show Customer**	OK	
Systemtest 3: **Add Customer**	OK	
Systemtest 4: **View Product**	OK	
Systemtest 5: **Add Order**	OK	

2.3.1 JUnit Tests - FBSData

ch.hslu.appe.usermanagement.filter.PlaceFilterTest [Runner: JUnit 4] (0.000 s)
- testFilterCanton (0.000 s)
- testFilterIdFail (0.000 s)
- testFilterCantonFail (0.000 s)
- testFilterAll (0.000 s)
- testFilterId (0.000 s)
- testFilterAllFail (0.000 s)
- testFilterNameFail (0.000 s)
- testFilterZipCodeFail (0.000 s)
- testFilterZipCode (0.000 s)
- testFilterName (0.000 s)

ch.hslu.appe.order.persistency.OrderPersistencyImplTest [Runner: JUnit 4] (10.389 s)
- testGetBills (6.593 s)
- testGetOrder (1.891 s)
- testGetNextOrderNumber (1.905 s)

ch.hslu.appe.usermanagement.model.UserTest [Runner: JUnit 4] (0.006 s)
- testLoginIllegal (0.000 s)
- testEqualsOtherProperties (0.001 s)
- testEqualsFailId (0.000 s)
- testEqualsFailLogin (0.001 s)
- testHashCodeOtherProperties (0.000 s)
- testHashCode (0.001 s)
- testEquals (0.000 s)
- testHashCodesFailId (0.000 s)
- testLoginLegal (0.001 s)
- testHashCodeFailLogin (0.002 s)

ch.hslu.appe.usermanagement.persistency.PlacesPersistencyImplTest [Runner: JUnit 4] (0.439 s)
- testGetPlacesCompleteness (0.383 s)
- testGetPlacesByZipCode (0.000 s)
- testPlaceExistsByZipCode (0.001 s)
- testPlaceNotExists (0.000 s)
- testPlaceExists (0.001 s)
- testGetPlacesByNonExistentId (0.000 s)
- testPlaceNotExistsByZipCode (0.001 s)
- testGetPlacesWithinRadius (0.011 s)
- testGetPlacesByNonExistentZipCode (0.000 s)
- testFilteredLocations (0.033 s)
- testGetPlacesById (0.001 s)
- testFilteredLocationsNotMatching (0.008 s)

ch.hslu.appe.usermanagement.persistency.UserPersistencyImplTest [Runner: JUnit 4] (6.347 s)
- testAddUser (0.824 s)
- testUserExistsByLogin (0.902 s)
- testUserExistsById (0.867 s)
- testUserExists (0.864 s)
- testGetUserById (0.921 s)
- testGetUserEmptyLogin (0.526 s)
- testGetUserInvalidId (0.535 s)
- testAddUserDuplicate (0.908 s)

ch.hslu.appe.usermanagement.persistency.AddressPersistencyImplTest [Runner: JUnit 4] (4.693 s)
- testGetNonExistentUserAddresses (0.239 s)
- testAddressExistsTrue (0.425 s)
- testAddSecondHomeAddress (0.530 s)
- testAddAddress (0.361 s)
- testDeleteAddress (0.549 s)
- testAddressUserNotExists (0.235 s)
- testDeleteHomeAddress (0.528 s)
- testAddressExistsFalse (0.232 s)
- updateHomeAddress (0.507 s)
- updateAddress (0.558 s)
- testGetAddresses (0.529 s)

2.3.2 JUnit Tests - FBSService

ch.hslu.appe.usermanagement.session.SessionTest [Runner: JUnit 4] (2.159 s)
- testGetUser (0.358 s)
- testExtend (0.001 s)
- testGetKey (1.800 s)

ch.hslu.appe.inventory.InventoryTest [Runner: JUnit 4] (10.940 s)
- testProductActive (0.858 s)
- testProductExists (0.757 s)
- testCommitProductReservation (1.221 s)
- testReservateProductFail (0.754 s)
- testProductNotActive (0.769 s)
- testGetProduct (0.748 s)
- testProductNotExists (0.760 s)
- testForceDoStockOrder (1.353 s)
- testGetProducts (1.054 s)
- testProductReservationMethods (1.010 s)
- testGenerateStockProductNumber (0.696 s)
- testDoStockOrder (0.959 s)

ch.hslu.appe.order.logic.OrderSystemTest [Runner: JUnit 4] (27.974 s)
- testInvalidUserCreateOrder (1.907 s)
- testAnnulateNonExistentOrder (1.425 s)
- testOrderNumberGeneration (2.208 s)
- testCreateOrder (2.056 s)
- testAnnulateLockedOrder (2.054 s)
- testCreateNullOrder (1.412 s)
- testCreateTwoOrdersAtOnce (2.038 s)
- testGetActiveOrder (2.040 s)
- testAnnulateOrder (2.051 s)
- testAnnulateOrderResetReservations (2.571 s)
- testDeliveryDate (3.436 s)
- testDeletePosition (1.426 s)
- testAnnulatePersistedOrder (3.350 s)

Testprotokoll

Hochschule Luzern
Technik & Architektur

ch.hslu.appe.usermanagement.session.SessionManagerTest [Runner: JUnit 4] (0.010 s)
- testCreateEmptySession (0.001 s)
- testIsInvalidByKey (0.001 s)
- testClearSession (0.001 s)
- testCount (0.004 s)
- testIsInvalidBySession (0.001 s)
- testIsInvalidByExpiredSession (0.001 s)
- testeCreateSessionTwice (0.001 s)
- testCreateSession (0.000 s)

Abbildungsverzeichnis

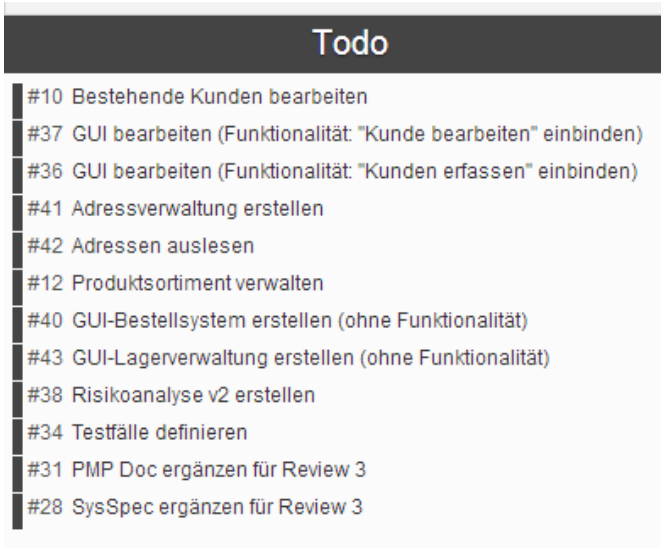
Tabellenverzeichnis

Anhang

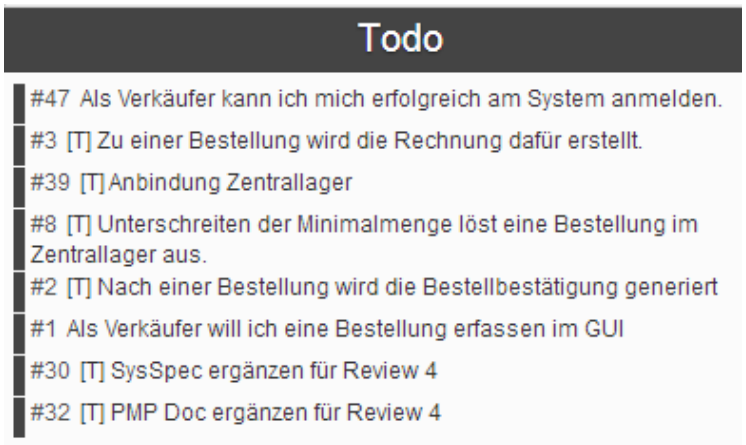
Sprintpläne:

Sprint 2:

Sprint 3:

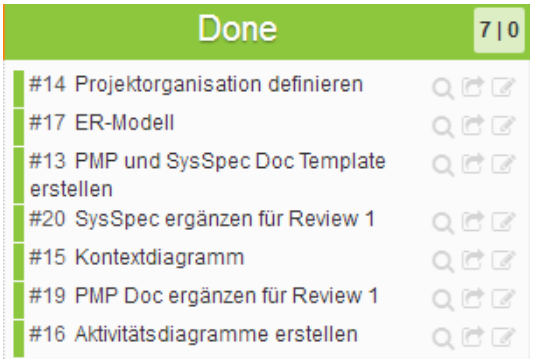


Sprint 4:

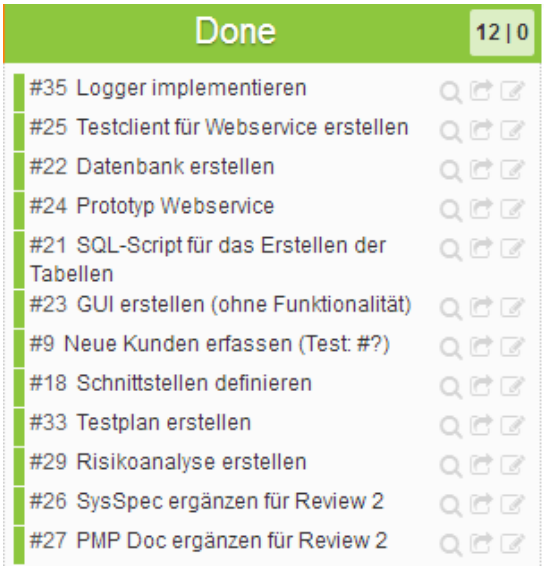


Sprintreview-Protokolle:

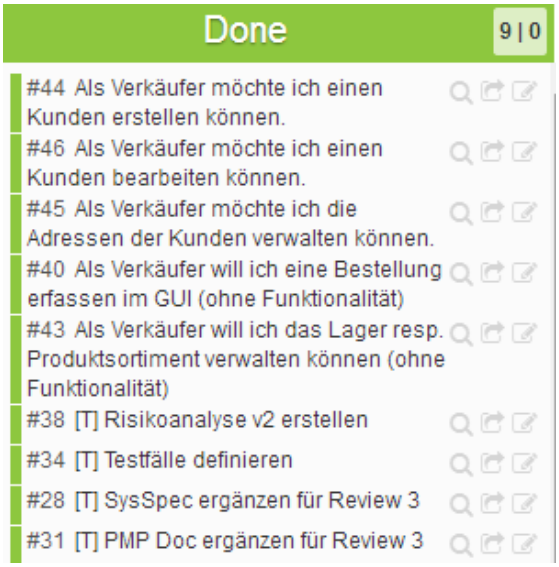
Sprint-Review 1:



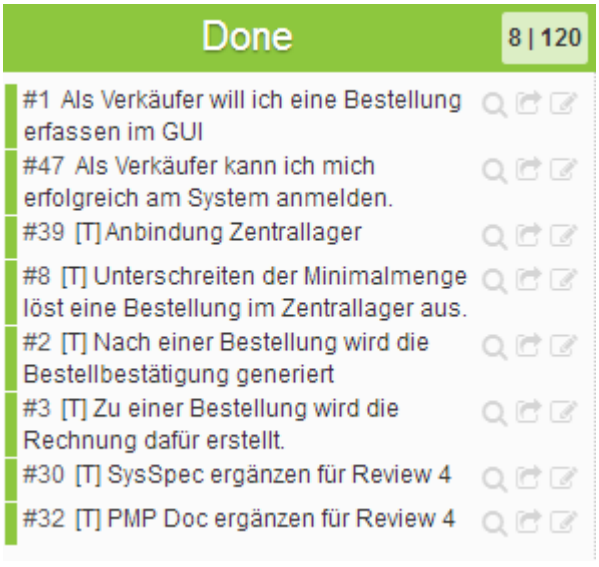
Sprint-Review 2:



Sprint-Review 3:



Sprint-Review 4:



Meilensteinberichte:

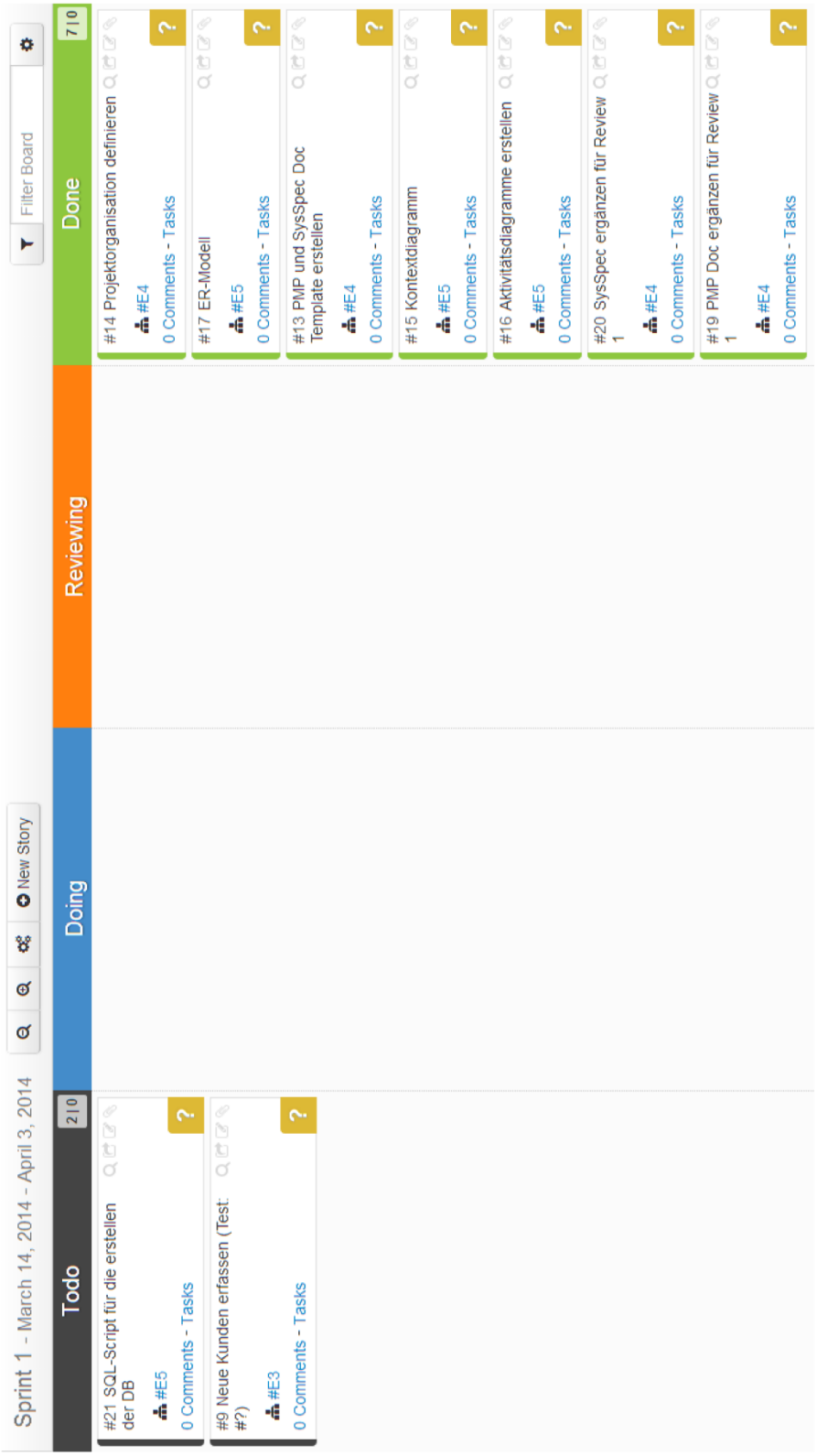
1 Protokoll

Besprechung	Sprint-Review 1
Datum	03.04.2014
Teilnehmer	Bontekoe Christian Estermann Michael Rohrer Felix
Dozenten	Hofstetter Jörg
Autor	Gruppe 20, Felix Rohrer

2 Pendenzen

ID	Beschreibung	Wer / Wann
---	Story 9 soweit OK (abgesehen davon das sie noch nicht begonnen wurde).	---
---	Wir sollten mit der Programmierung beginnen. Zum Beispiel mit dem GUI beginnen.	---
1	Stories für nachfolgenden Sprint sollten bereits definiert sein.	Felix Review 2
2	Stories überprüfen und ggf. in kleinere Stories aufteilen.	Alle asap
---	Nach Möglichkeit „Test First“ umsetzen. Dazu braucht es keine einzelne Story.	---
3	WebService Prototyp erstellen	Mike 11.04.2014

3 ScrumDo


Sprint 1 - March 14, 2014 - April 3, 2014
New Story
Filter Board
Todo 2 | 0
Doing
Reviewing
Done 7 | 0
#21 SQL-Script für die erstellen der DB
#E5
0 Comments - Tasks
#9 Neue Kunden erfassen (Test: #7)
#E3
0 Comments - Tasks
#14 Projektorganisation definieren
#E4
0 Comments - Tasks
#17 ER-Modell
#E5
0 Comments - Tasks
#13 PMP und SysSpec Doc Template erstellen
#E4
0 Comments - Tasks
#15 Kontextdiagramm
#E5
0 Comments - Tasks
#16 Aktivitätsdiagramme erstellen
#E5
0 Comments - Tasks
#20 SysSpec ergänzen für Review 1
#E4
0 Comments - Tasks
#19 PMP Doc ergänzen für Review 1
#E4
0 Comments - Tasks

1 Protokoll

Besprechung	Architektur Review
Datum	04.04.2014
Teilnehmer	Bontekoe Christian Estermann Michael Rohrer Felix
Dozenten	Gisler Roland
Autor	Gruppe 20, Felix Rohrer

2 Pendenzen

ID	Beschreibung	Wer / Wann
---	Architektur wird mittels WebService umgesetzt. 3 Tier-Architektur 4 Webservices: - Accounting - Location - Inventory - Order Persistence: Hibernate Framework (JPA)	---
---	Diagramm: „Gemisch“ aus Klassen, Objekt, „Abstrakt“ (DTO) → grenzwertig - Tiers sind nicht sichtbar	---
---	Nächste Schritte: - Fachliche Schnittstellen definieren - Zuerst den „Durchstich“ und erst danach den Webservice einbauen →Dadurch Vereinfachung des ersten Durchstiches und danach erweitern.	---
1	Klassendiagramm (inkl. Interfaces) Komponenten Diagramm (Interfaces !) Deployment Diagramm	Christian

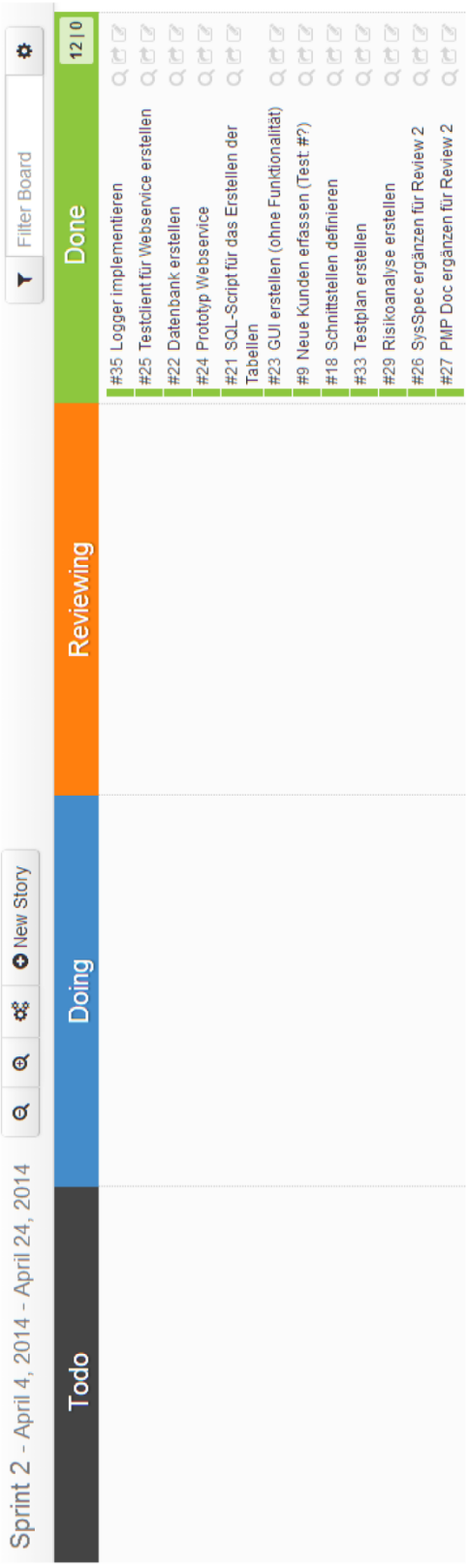
1 Protokoll

Besprechung	Sprint-Review 2
Datum	24.04.2014
Teilnehmer	Bontekoe Christian Estermann Michael Rohrer Felix (entschuldigt, Microsoft Kurs in Wallisellen)
Dozenten	Hofstetter Jörg
Autor	Gruppe 20, Felix Rohrer

2 Pendenzen

ID	Beschreibung	Wer / Wann
--	Beim nächsten Review soll ein lauffähiges Programm demonstriert werden.	Alle
--	Nach Möglichkeit sollen Task definiert werden welche einen Durchstich beinhaltete und nicht nur einen einzelnen Layer.	Alle

3 ScrumDo


Filter Board
Done
12 | 0
#35 Logger implementieren
#25 Testclient für Webservice erstellen
#22 Datenbank erstellen
#24 Prototyp Webservice
#21 SQL-Script für das Erstellen der Tabellen
#23 GUI erstellen (ohne Funktionalität)
#9 Neue Kunden erfassen (Test #?)
#18 Schnittstellen definieren
#33 Testplan erstellen
#29 Risikoanalyse erstellen
#26 SysSpec ergänzen für Review 2
#27 PMP Doc ergänzen für Review 2
Reviewing
Doing
New Story
Sprint 2 - April 4, 2014 - April 24, 2014
Todo

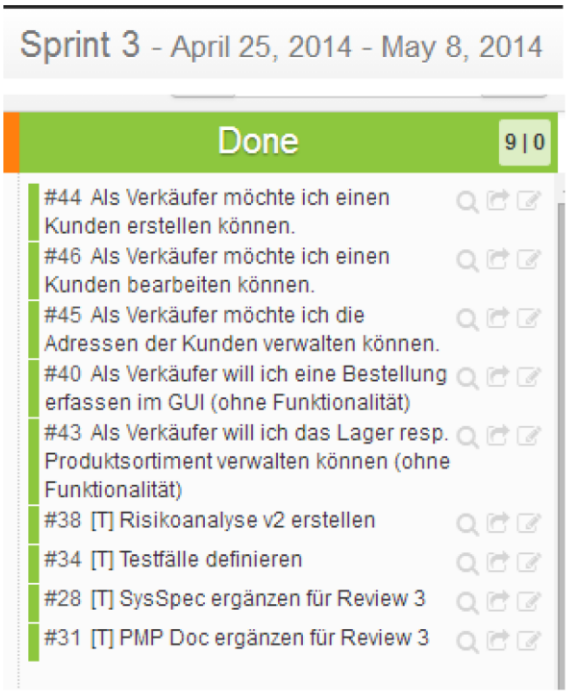
1 Protokoll

Besprechung	Sprint-Review 3
Datum	08.05.2014
Teilnehmer	Bontekoe Christian Estermann Michael Rohrer Felix
Dozenten	Hofstetter Jörg
Autor	Gruppe 20, Felix Rohrer

2 Pendenzen

ID	Beschreibung	Wer / Wann
--	Bestellung ist erst gemockt	
1	Bestellung im GUI erfassen und in DB eintragen inkl. Lagerbestand nachtragen	
--	Aufwandschätzung in ScrumDo eintragen	
2	Neue Story „Login mit Session Mgmt“ inkl. Testfall	

3 ScrumDo


Sprint 3 - April 25, 2014 - May 8, 2014
Done
9 | 0
#44 Als Verkäufer möchte ich einen Kunden erstellen können.
#46 Als Verkäufer möchte ich einen Kunden bearbeiten können.
#45 Als Verkäufer möchte ich die Adressen der Kunden verwalten können.
#40 Als Verkäufer will ich eine Bestellung erfassen im GUI (ohne Funktionalität)
#43 Als Verkäufer will ich das Lager resp. Produktsortiment verwalten können (ohne Funktionalität)
#38 [T] Risikoanalyse v2 erstellen
#34 [T] Testfälle definieren
#28 [T] SysSpec ergänzen für Review 3
#31 [T] PMP Doc ergänzen für Review 3

1 Protokoll

Besprechung	Sprint-Review 4
Datum	22.05.2014
Teilnehmer	Bontekoe Christian Estermann Michael Rohrer Felix
Dozenten	Gisler Roland Olnhoff Thomas
Autor	Gruppe 20, Felix Rohrer

2 Pendenzen

ID	Beschreibung	Wer / Wann
--	Komponenten Diagramm wurde besprochen, die grafische Darstellung (Linien die sich kreuzen) kann noch optimiert werden.	Mike / 29.05
--	Die Applikation konnte erfolgreich ohne Fehler demonstriert werden.	--

3 ScrumDo

Sprint 4 - May 9, 2014 - May 22, 2014

Done 8 | 120

- #1 Als Verkäufer will ich eine Bestellung erfassen im GUI
- #47 Als Verkäufer kann ich mich erfolgreich am System anmelden.
- #39 [T] Anbindung Zentrallager
- #8 [T] Unterschreiten der Minimalmenge löst eine Bestellung im Zentrallager aus.
- #2 [T] Nach einer Bestellung wird die Bestellbestätigung generiert
- #3 [T] Zu einer Bestellung wird die Rechnung dafür erstellt.
- #30 [T] SysSpec ergänzen für Review 4
- #32 [T] PMP Doc ergänzen für Review 4