**Projekt: Mosti**

System - Testdokumentation

[Dokumentstruktur basiert auf RUP „Dokument Test Evaluation Summary“]

# Dokumentinformationen

## Änderungsgeschichte

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Version | Änderung | Autor |
| 7.04.2016 | 1.0 | Erstellung der Testdokumentation | Team 11 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Inhalt

[1 Dokumentinformationen 2](#_Toc450515341)

[1.1 Änderungsgeschichte 2](#_Toc450515342)

[1.2 Inhalt 2](#_Toc450515343)

[2 Einführung (Introduction) 3](#_Toc450515344)

[2.1 Definitionen und Abkürzungen (Definitions, Acronyms, Abbreviations) 3](#_Toc450515345)

[2.2 Referenzen (References) 3](#_Toc450515346)

[2.3 Übersicht (Overview) 3](#_Toc450515347)

[3 Testvorgehen 4](#_Toc450515348)

[3.1 Funktionale Tests 4](#_Toc450515349)

[3.1.1 Grundtests (Smoke Tests) 4](#_Toc450515350)

[3.1.2 Modul- und Unittests 4](#_Toc450515351)

[3.1.3 Integrationstests 4](#_Toc450515352)

[3.1.4 System Acceptance Test 4](#_Toc450515353)

[3.2 Bedienbarkeit und Nutzerinterface (Usability) 5](#_Toc450515354)

[3.3 Datenschutz, Datensicherheit (Security) 5](#_Toc450515355)

[3.4 Leistungsanforderungen (Performance) 5](#_Toc450515356)

[3.5 Zuverlässigkeit 5](#_Toc450515357)

[3.6 Schnittstellen 5](#_Toc450515358)

[3.7 Wartung und Servicefunktionen 5](#_Toc450515359)

[3.8 Installation 6](#_Toc450515360)

[3.9 Internationalisierung / Lokalisierung 6](#_Toc450515361)

[3.10 Testautomatisierung 6](#_Toc450515362)

[3.11 Verfolgbarkeit (Traceability) 6](#_Toc450515363)

[4 Übersicht der Testpläne 6](#_Toc450515364)

[5 Freigabe von Testergebnissen 6](#_Toc450515365)

# Einführung (Introduction)

## Definitionen und Abkürzungen (Definitions, Acronyms, Abbreviations)

Vgl. separates Glossary-Dokument im Repository, Dateiname: Glossary.docx

## Referenzen (References)

Buch: Craig Larman: UML 2 und Patterns angewendet (2005)

## Übersicht (Overview)

In diesem Dokument werden verschieden Testverfahren erläutert die wir für unsere Software entwickelt haben. Wir gehen darauf ein wie unsere Tests durchgeführt und diese dokumentiert werden.

# Testvorgehen

## Funktionale Tests

### Grundtests (Smoke Tests)

Schon während der Programmierung werden die einzelnen Codestücke immer von den programmierenden Mitgliedern geprüft. Dafür werden z.B. in der Datenbank einige Testkunde angelegt, um die Verbindung zur Datenbank oder die richtige Anzeige zu überprüfen. Des Weiteren werden die Codes vor dem Commit ins Repository auf ihre Lauffähigkeit getestet, d.h. es wird überprüft, ob das Codesegment unabhängig von eventuellen Nutzereingaben überhaupt kompilieren oder zur Laufzeit irgendwelche Exceptions werfen. Da die diese Tests innerhalb der ersten Programmierung ausgeführt werden, können sie nicht dokumentiert werden.

### Modul- und Unittests

Nach der Erstellung einer vorläufigen Version der einzelnen Module werden diese durch ein Teammitglied, das nicht an der Programmierung desselben Moduls beteiligt war, in den Unittests geprüft.

Jede Testfunktion prüft genau eine andere Funktion und ruft diese mit unterschiedlichen Parametern auf, berechnet aber gelichzeitig den richtigen Rückgabewert und vergleicht diesen dann mit der Rückgabe der zu testenden Funktion. Eventuelle Fehler und Unstimmigkeiten werden vom Tester behoben und die Ergebnisse an den verantwortlichen Programmierer weitergegeben.

Zur Überprüfung der Testfunktionen verwenden wir das Tool EclEmma (EclEmma Java Code Coverage, Version 2.3.3.102602231923, ID: com.mountanminds.eclemma.feature.feature.group, Provider: Mountainminds GmbH & Co. KG), welches überprüft, ob alle möglichen Testfälle der geschriebenen Testfunktion durchlaufen wurden. Ziel ist Hierbei eine Codecoverage von mindestens 90%.

### Integrationstests

Um die Mosti-Software möglichst sicher und konsistent zu halten, werden alle Methoden im Rahmen ihrer Nutzung von anderen Klassen mit der niedrigsten Sichtbarkeit versehen. Hierbei soll sichergestellt werden, dass interne Änderungen möglichst wenig Auswirkung nach außen und zu anderen Klassen bzw. Packages hat.

Beim Zusammenfügen von mehreren Modulen muss hierbei allerdings getestet werden, ob die Packages untereinander auf alle Funktionen, die sie von anderen Packages brauchen, auch zugreifen können, oder aber sich einzelne Funktionsaufrufe gegenseitig negativ beeinflussen.

### System Acceptance Test

Zur Überprüfung der Anwenderfreundlichkeit und Bedienbarkeit sind regelmäßige User Acceptance Tests mit Mitarbeitern der Mosterei Hemau geplant. Hierbei müssen die Probanden nach einer kurzen Einführung in das System selbst einige Funktionen ausführen, oder sich ohne Einweisung in der Software zurechtfinden. Die gewonnenen Erkenntnisse werden nach dem Template im Test Plan festgehalten und dokumentiert. Grundlegende Änderungen der Anforderungen werden im kompletten Termin besprochen und bei Bedarf kontrolliert durchgeführt.

## Bedienbarkeit und Nutzerinterface (Usability)

Um die Bedienbarkeit und Nutzerfreundlichkeit des Systems noch vor dem Release der Software zu testen, stehen uns verschieden Mitarbeiter der Mosterei Hemau zur Verfügung. Nach einem vorgefertigten Template werden regelmäßig User Acceptance Tests mit verschiedenen Probanden (unterschiedliches Alter, unterschiedliche Erfahrungen mit Computern usw.) durchgeführt, die Ergebnisse ausgewertet und bei Bedarf die Software geändert bzw. angepasst. Geänderte oder neu entstandene Anforderungen werden laut dem Requirement Management Plan behandelt.

## Datenschutz, Datensicherheit (Security)

Um die Sicherheit der gespeicherten Daten zu gewährleisten, können nur berechtigte Personen mit Benutzername und Passwort auf die Software zugreifen. Bestimmte Ansichten (z.B. die Übersicht über die täglichen Verkaufseinnahmen) sind auch nur dem Mosterei-Inhaber vorbehalten. Die Speicherung der Passwörter soll vom System verschlüsselt erfolgen und kann somit von außen nicht gelesen werden.

## Leistungsanforderungen (Performance)

Die Leistungsanforderungen der Mosti-Software werden einerseits in den Unittest von den einzelnen Mitgliedern nach den Festlegungen in der Anforderungsspezifikation geprüft. Des Weiteren können die Probanden des User Acceptance Tests die Leistung der Software in ihren Testbögen bewerten. Ein Härtefalltest, für den in der Datenbank bis zu 1000 Datensätze angelegt werden, ist geplant. Bei Nichterfüllen der Leistungsanforderungen müssen im Rahmen von Codereviews ineffiziente Codestrukturen gefunden und beseitigt werden.

## Zuverlässigkeit

Die Zuverlässigkeit der Mosti-Software wird innerhalb der Unittests geprüft. Dabei wird jede nach außen hin sichtbare Methode durch eine Testfunktion abgedeckt, die der zu testenden Methode jeweils verschiedene Parameter übergibt und das errechnete Ergebnis überprüft. Die Unittests werden sowohl schon während der Programmierung einzelner Module als auch zum Abschluss desselben durchgeführt. Alle Änderungen im Code, die eine Schnittstelle nach außen besitzen, müssen vor dem Release getestet worden sein und die Testdokumentationen den Mitgliedern zugänglich sein.

## Schnittstellen

Die geplante Mosti-Software besitzt verschiedene Schnittstellen nach außen. Die wichtigste hierbei ist die Verbindung zu einer Datenbank, die z.B. Kunden- oder Mitarbeiterdaten speichert. Die Verbindung zu diese Datenbank kann und wird innerhalb den Unittests abgedeckt. Eine weitere Schnittstelle ist die Verbindung zu einem Drucker, der eine Übersicht über die gekauften Produkte und Dienstleistungen eines Kunden druckt. Auch diese Schnittstelle kann im Rahmen von Unit- oder Funktionalitätstests geprüft werden.

## Wartung und Servicefunktionen

Schon beim Zusammenfügen der einzelnen Module wird die geringe Kopplung und hohe Kohäsion der Software geprüft. Dies spiegelt die Erfüllung der Wartungsanforderungen wieder und können vor dem ersten Release nur durch Hinzufügen von neuen Codesegmenten geprüft werden. Ineffiziente Strukturen müssen bei Codereviews analysiert und beseitigt werden.

## Installation

Um die Installationsanforderungen der Software zu testen, ist ein Funktionalitätstest vorgesehen, innerhalb welchem die Software auf einem Computer neu installiert wird. Für die Kunden ist eine Bedienungsanleitung vorgesehen, die auch eine Installationsanleitung enthält.

## Internationalisierung / Lokalisierung

Unsere Software soll nach ihrer Fertigstellung zum Download auf einer entsprechenden Website zur Verfügung stehen. Auch hier sind Tests mit Anwendern geplant, um zu gewährleisten, dass die Software schnell und einfach heruntergeladen werden kann. Wenn die Testergebnisse es implizieren, wird eine Anleitung zum Download erstellt. Möglichkeiten zur Internationalisierung (z.B. verschiedene Sprachen, verschiedene Währungen innerhalb der Mosti-Software) sind nicht geplant.

## Testautomatisierung

Die geplanten Unittests werden innerhalb von Eclipse ausgeführt. Dafür wird von Mavenprojekten ein eigener Testordner vorgegeben. Unittests werden mithilfe des Eclipsetools EclEmma durchgeführt.

User Acceptance Tests orientieren sich an den im Testplan spezifizierten Template und werden ohne zusätzliche Werkzeuge ausgeführt.

## Verfolgbarkeit (Traceability)

In der Mitte der Construction-Phase ist eine Sitzung mit allen Teammitgliedern geplant, in welcher über die geplanten und die bisher tatsächlich umgesetzten Anforderungen gesprochen und diskutiert wird. Diese wird zum Ende der Construction-Phase noch einmal wiederholt.

*Wie wird sichergestellt, dass alle erfassten Anforderungen verifiziert werden?>*

# Übersicht der Testpläne

Bisher dokumentierte Tests:

**Funktionalitätstests:**

* Kassenfunktion: vgl. *Funktionalitätstest Kasse-1.odt*
* Kundenverwaltung: vgl. *Funktionalitätstest KundenVer-1.odt*

**User Acceptance Tests (mit diversen getesteten Modulen):**

* Kassenfunktion, Terminplanung, Dienstleistung, Lagerverwaltung, Kundenverwaltung:  
  vgl*. User Acceptance Test 1.docx*
* Kassenfunktion: vgl. *User Acceptance Test 2.docx*
* Kassenfunktion, Terminplanung, Dienstleistung, Lagerverwaltung, Kundenverwaltung:  
  vgl. *User Acceptance Test 3.docx*

Weitere Tests viola

# Freigabe von Testergebnissen

Die Freigabe der Testergebnisse sind alle Mitglieder des Teams verantwortlich. Diese werden im Rahmen der wöchentlichen Sitzungen besprochen und diskutiert.

Zur Fehlerbeschreibung wird folgendes Klassifikationsschema verwendet:

*minor:* Kleine Fehler, wie z.B. unübersichtliche Grafikaufbereitung oder Rechtschreibfehler

*moderate:* Mittlere Probleme, wie z.B. kleine Rundungsfehler, umständliche Handhabung der Software

*major:* Schwere Fehler, wie z.B. falsche oder keine Speicherung von Datenbanksätzen

Für den Hauptrelease werden nur Fehler von höchstens minor akzeptiert.