**Projekt: Mosti**

System - Testdokumentation

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

# Dokumentinformationen

## Änderungsgeschichte

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Version | Änderung | Autor |
| 04.05.2016 | 1.0 | Erstellung der Testdokumentation | Team 11 |
| 06.07.2016 | 2.0 | Überarbeitung für Endabgabe | Team 11 |
|  |  |  |  |
|  |  |  |  |

## Inhalt

[1 Dokumentinformationen 2](#_Toc450551481)

[1.1 Änderungsgeschichte 2](#_Toc450551482)

[1.2 Inhalt 2](#_Toc450551483)

[2 Einführung (Introduction) 3](#_Toc450551484)

[2.1 Definitionen und Abkürzungen (Definitions, Acronyms, Abbreviations) 3](#_Toc450551485)

[2.2 Referenzen (References) 3](#_Toc450551486)

[2.3 Übersicht (Overview) 3](#_Toc450551487)

[3 Testvorgehen 4](#_Toc450551488)

[3.1 Funktionale Tests 4](#_Toc450551489)

[3.1.1 Grundtests (Smoke Tests) 4](#_Toc450551490)

[3.1.2 Modul- und Unittests 4](#_Toc450551491)

[3.1.3 Integrationstests 4](#_Toc450551492)

[3.1.4 System Acceptance Test 4](#_Toc450551493)

[3.2 Bedienbarkeit und Nutzerinterface (Usability) 5](#_Toc450551494)

[3.3 Datenschutz, Datensicherheit (Security) 5](#_Toc450551495)

[3.4 Leistungsanforderungen (Performance) 5](#_Toc450551496)

[3.5 Zuverlässigkeit 5](#_Toc450551497)

[3.6 Schnittstellen 5](#_Toc450551498)

[3.7 Wartung und Servicefunktionen 5](#_Toc450551499)

[3.8 Installation 6](#_Toc450551500)

[3.9 Internationalisierung / Lokalisierung 6](#_Toc450551501)

[3.10 Testautomatisierung 6](#_Toc450551502)

[3.11 Verfolgbarkeit (Traceability) 6](#_Toc450551503)

[4 Übersicht der Testpläne 7](#_Toc450551504)

[5 Freigabe von Testergebnissen 7](#_Toc450551505)

# 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)

Dieses Dokument bietet eine Übersicht über die verschiedenen Testverfahren, mit welchen die Mosti-Software getestet wird. Die Tests werden einzeln vorgestellt und das jeweilige Textvorgehen mit ihren Werkzeugen erläutert.

# 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 Testkunden 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 die Codesegmente unabhängig von eventuellen Nutzereingaben überhaupt kompilieren oder zur Laufzeit irgendwelche Exceptions werfen. Da 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 innerhalb von Unittests geprüft.

Jede Testfunktion prüft genau eine andere Funktion und ruft diese mit unterschiedlichen Parametern auf, berechnet aber gleichzeitig den richtigen Rückgabewert und vergleicht diesen dann mit der Rückgabe der zu testenden Funktion. Eventuelle Fehler und Unstimmigkeiten werden vom Tester behoben und bei Problemen auf der Seite LeanTesting (https://leantesting.com/en/projects/mosti/14011) festgehalten und so dem gesamten Team zur Diskussion und Bearbeitung freigestellt.

### 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 haben.

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.

Durch das Testen der Methoden außerhalb ihres „normalen“ Sichtbereiches wurden alle Methoden public gesetzt und nicht mehr auf private zurückgesetzt, da diese Sichtbarkeit nicht getestet werden konnte.

### 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. Ausgewertet wird bei den Tests lediglich, ob der Proband im Stande war, die geforderten Aufgaben in einer gewissen Zeit zu erfüllen oder nicht. Grundlegende Änderungen der Anforderungen, die sich dadurch evtl. ergeben könnten, werden im kompletten Team 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 u. a. Mitarbeiter der Mosterei Hemau zur Verfügung. Nach verschiedenen vorgefertigten Templates 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 Benutzernamen 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 Unittests 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. Bei Nichterfüllen der Leistungsanforderungen müssen im Rahmen von Codereviews ineffiziente Codestrukturen gefunden und beseitigt werden.

## Zuverlässigkeit

Die Zuverlässigkeit der Mosti-Software wird innerhalb der JUnittests 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.

Aufgrund mangelnder Zeit konnten nicht mehr alle Funktionalitäten in kompletten Umfang getestet werden.

## 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 dieser 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 liefert. 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 kann 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.

User Acceptance Tests orientieren sich an den im Testplan spezifizierten Templates 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.

# Übersicht der Testpläne

**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*

# Freigabe von Testergebnissen

Für 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.