Projekt: Typo

System - Testdokumentation

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

1 Dokumentinformationen

1.1 Änderungsgeschichte

Datum	Version	Änderung	Autor
13.05.17	1.0	Erstellung System-Testdokumentation	Nguyen, Frank
29.05.	2.0	Änderungsarbeiten an der Testdokumentation	Nguyen, Frank

1.2 Inhalt

2 Einführung (Introduction)

2.1 Definitionen und Abkürzungen (Definitions, Acronyms, Abbreviations)

<Erklärung der Abkürzungen und Definitionen oder Verweis auf separates Glossar-Dokument>
WPM = Wörter pro Minute
FPM = Fehler pro Minute

2.2 Referenzen (References)

<Liste aller verwendeten Dokumente, Bücher, etc…>

-

2.3 Übersicht (Overview)

<Übersicht über den restlichen Teil dieses Dokuments geben und dessen Aufbau erläutern>
Zuerst werden die grundsätzlichen Tests erklärt, mit denen wir regelmäßig unser Programm testen. Danach werden die Modul- und Unittests dargelegt und wer diese durchführt. Es werden verschiedene Testarten (Integrationstest, Systemtest) dargestellt. Daraufhin wird die Testart der verschiedenen Programmeigenschaften erläutert. Eine Übersicht über die Testpläne wird später hinzugefügt, da noch keine konkreten Testpläne festgelegt wurden. Die Kriterien für Testerfolg werden exemplarisch verläutet.

3 Testvorgehen

3.1 Funktionale Tests

3.1.1 Grundtests (Smoke Tests)

<Beschreiben Sie Tests, mit denen offensichtliche Fehler schnell gefunden werden können, z.B. Tests, die vor jedem Commit in das Repository durchzuführen sind (ggf. Verweis auf externes Dokument).>

- Lauffähigkeit des Programms durch Ausführen im Qt Creator.
- Verbindung mit der Datenbank wird über eine Abfrage zur Verbindung (ja oder nein) überprüft.
- Erkennung von Tippfehlern, indem wir absichtlich falsche Zeichen bei einem Übungsfall eingeben
- Anmelden mit Benutzername und Passwort
- Texte aus der Datenbank in das Programm ziehen/einlesen durch Programmablauf testen
- Serveranbindung wird versucht über den Multiplayermodus aufzubauen
- Verbindung mit der Webseite wird über den Button im Mainwindow getestet, danach im Internetbrowser des Testers wird ein Zugriff geprobt.
- Kann auf die verschiedenen Links auf der Website zugegriffen werden und die Funktionen von diesen genutzt werden
- Aktualisierung der Lernstatistik, durch Ausführen einer Übung mit einem Dummy-Account werden die Werte aktiv verändert und danach abgefragt.

3.1.2 Modul- und Unittests

<Wie (mit welchen Werkzeugen, in welcher Umgebung, in wessen Verantwortung) sollen Modul- und Unittests durchgeführt werden? Wie erfolgt die Dokumentation der Testergebnisse? Gibt es Vorgaben zur Testüberdeckung?
Hier werden nicht einzelne Tests beschrieben, sondern nur die Vorgehensweise!>
Alexander Krügel verwendet die Entwicklungsumgebung Qt Creator, um die Funktionalität der Datenbank zu testen. Zusätzlich führt er nach dem Zusammensetzen der Programmfiles (.cpp, .ui, .h, .txt, .db) eine regelmäßige Kontrolle im Zusammenhang mit der Datenbank Verbindungstests durch.
Die Programmlogik (Erkennung von Fehlern) wird regelmäßig von Lennart Paulu Frank mit offline .txt-Dateien und Texte/Wörter aus der Datenbank getestet.
Für einen Test im Multiplayer-Modus werden zwei Tester gebraucht. Dabei werden sowohl alle Entwickler im Team als auch Außenstehende den Modus testen. Der Fokus liegt darin, ob eine Verbindung

aufgebaut wird, Datenübertragung wie Texte, WPM und FPM stattfindet und gespeichert wird und ob der Multiplayer-Modus terminiert. Dieser Test liegt in der Verantwortung von Alexander Krügel, da er administrativ für den Root Server zuständig ist.

3.1.3 Integrationstests

<Sind gesonderte Tests notwendig, während / nachdem verschiedene Module zusammengeführt werden?>

Nachdem der Multiplayer-Modus hinzugefügt wurde, muss der Zugriff auf die Datenbank von verschiedenen Zugriffsstellen gleichzeitig getestet werden. Die Verteilung von Texten auf zwei Benutzer steht ebenfalls unter Beobachtung. Auch müssen wir testen, dass die Nutzerdaten auf der Datenbank und im Programm stets aktuell angezeigt werden.

3.1.4 System Acceptance Test

<Auflistung der Tests auf System- / Nutzerebene, die sich an den Use Cases orientieren. Ggf. Verweis auf separates Dokument. Beschreibung, wie / von wem diese Tests durchzuführen sind.>

3.2 Bedienbarkeit und Nutzerinterface (Usability)

Trang Nguyen wird anhand des aktualisierten Use Case und Domain Modell die Funktionalität und Richtigkeit des Programms testen. Dabei steht der Fokus, dass die Übungen die richtigen Texte beinhalten, das Programm terminiert, die Nutzerdaten gespeichert und aktualisiert angezeigt werden. Zur Bedienbarkeit gehört auch ein gutes Design des Programms und wird daher ständig überabeitet und angepasst.

3.3 Datenschutz, Datensicherheit (Security)

Ein Angriff auf unsere Datenbank wird simuliert um die Datensicherheit zu testen. Dieser wird in Brute-Force Variante und einer weiteren Variante ausgeführt. Es wird auch versucht, über Brute Force einen LogIn Vorgang zu erreichen.

3.4 Leistungsanforderungen (Performance)

Die Programmausführung wird regelmäßig auf flüssigen Ablauf getestet.

3.5 Zuverlässigkeit

Es wird versucht, mit vielen Spielern gleichzeitig einen Multiplayer-Wettkampf zu starten, um das System zu überlasten.

3.6 Schnittstellen

Lennart Paul Frank testet die backend-Schnittstellen zur UI. Dabei wird vor allem auf die Fehlererkennungslogik eingegangen. Auch wird durch Zugriffe auf die Datenbank die Verbindung von Programm und Datenbank getestet.

3.7 Wartung und Servicefunktionen

Die Wartbarkeit des Programms wird durch die Programmierer der einzelnen Module gewährleistet, diese wird durch einen Austausch von Modulen und Code-Review zwischen Team-Mitgliedern getestet. Die Website und die Implementation und der benutzerfreundliche Umgang mit dem Benutzerhandbuch wird durch Christian Weich getestet.

3.8 Installation

Die Installation des Programms wird durch Trang Nguyen durch einen Installationsversuch getestet. Jede Prototypversion erhält ihre eigene Versionsnummer. Ab Programmversion 1.0.3 wird jede aktualisierte Version in einer .exe Datei verpackt und in mehreren Windows-Systemen zur Installation getestet und auf Funktionalität und Richtigkeit geprüft.

3.9 Internationalisierung / Lokalisierung

Das Programm wird nur auf Deutsch getestet und in Deutschland vertrieben.

3.10 Testautomatisierung

<Welche Tests werden mit welchen Werkzeugen automatisiert? Wer erstellt / prüft die automatisierten Tests?>
Wir besitzen im Moment noch keine automatisierten Tests oder Testwerkzeuge.

3.11 Verfolgbarkeit (Traceability)

<Wie wird sichergestellt, dass alle erfassten Anforderungen verifiziert werden?>
Die Vollständigkeit der Anforderungen wird von mehreren Teammitgliedern individuell und unbeeinflusst über eine Liste geprüft. Diese Liste beinhaltet die vorher festgelegten Anforderungen aus dem 2. Meilenstein.

4 Übersicht der Testpläne

<Hier werden alle Dokumente, in denen Testfälle beschrieben werden, identifizierbar erfasst.>
Unser Testplan wird durch das Dokument: „Testplan vom Programm Typo“ erläutert (Repository\Meilensteine\MS 4\Dokumente\ Testplan vom Programm Typo.pdf)

5 Freigabe von Testergebnissen

<Wer entscheidet auf Grund welcher Kriterien darüber, ob Testergebnisse akzeptiert werden. (z.B.: Klassifikationsschema minor – moderate – major für Fehler. Für Alpha-Release werden moderate und minor Fehler, für Final Release nur minor Fehler akzeptiert.>
Der Projektadmin entscheidet aufgrund von teamseitig festgelegten Kriterien, ob Testergebnisse akzeptiert oder abgelehnt werden. Diese sind für die einzelnen Module angepasst. z.B. Werden bei der Fehlererkennung nicht erkannte Fehler als inakzeptabel klassifiziert, wenn mehr als 1 Wort je 1000 Wörter trotz fehlerbehaftung nicht als fehlerhaft erkannt wird. Weitere akzeptierte Fehler sind: Wenn eine Verbindung mit einem anderen Spieler im Multiplayer-modus bei 1 von 10 Versuchen nicht erfolgt. Bei dem Auslesen eines Textes aus der Offline Datenbank sollte sich nur bei 1nem von 30 Versuchen ein Fehler ereignen.
Testergebnisse werden nur dann vom Teamadmin freigegeben, wenn sie vom Team selbst als „nicht schwerwiegend“ klassifiziert werden.
Schwerwiegendere Testergebnisse werden nicht freigegeben.
Ab mehr als 3 schwachen Fehlern soll das Testergebnis auch nicht freigegeben werden.