Projekt: U-Car

Testplan

# Dokumenteninformation

## Änderungsgeschichte

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Version | Änderungsgrund | Autorin |
| 07.04.23 | 1.0 | Erstellung | Wenzl |
| 03.05.23 | 1.1 | Kleinere Änderungen am Testvorgehen | Wenzl |
| 20.05.23 | 1.2 | Aktualisierung | Wenzl |
| 18.06.23 | 1.3 | Aktualisierung der Verfolgbarkeit | Wenzl |

## Inhalt

[1 Dokumenteninformation 2](#_Toc135510704)

[1.1 Änderungsgeschichte 2](#_Toc135510705)

[1.2 Inhalt 3](#_Toc135510706)

[2 Einführung 4](#_Toc135510707)

[2.1 Definitionen und Abkürzungen 4](#_Toc135510708)

[2.2 Referenzen 4](#_Toc135510709)

[3 Testvorgehen 4](#_Toc135510710)

[3.1 Funktionale Tests 4](#_Toc135510711)

[3.1.1 Grundtests (Smoke-Tests) 4](#_Toc135510712)

[3.1.2 Modul- und Unittests 4](#_Toc135510713)

[3.1.3 Integrationstests 4](#_Toc135510714)

[3.1.4 System Acceptance Test 4](#_Toc135510715)

[3.2 Bedienbarkeitstests 4](#_Toc135510716)

[3.3 Zuverlässigkeitstests 5](#_Toc135510717)

[3.4 Leistungstests 5](#_Toc135510718)

[3.5 Sicherheitstests 5](#_Toc135510719)

[3.6 Installationstests 5](#_Toc135510720)

[3.7 Testautomatisierung 5](#_Toc135510721)

[3.8 Verfolgbarkeit 5](#_Toc135510722)

[4 Übersicht der Testpläne 5](#_Toc135510723)

[5 Freigabe von Testergebnissen 5](#_Toc135510724)

# Einführung

## Definitionen und Abkürzungen

GUI: Graphic User Interface

## Referenzen

# Testvorgehen

## Funktionale Tests

### Grundtests (Smoke-Tests)

Offensichtliche oder grobe Fehler sollen von der jeweiligen Entwicklerin gefunden werden, bevor der Code in das Git-Repository aufgenommen wird. Hier soll exploratives Testen zum Einsatz kommen, um die grundsätzliche Funktionalität des Codes zu überprüfen. Das Kompilieren stellt dabei den ersten Smoke-Test dar, da anhand Warnungen oder Fehlermeldungen sofort erkennbar ist, ob der Code ausgeführt werden kann. Treten hier keine Fehler auf, kann mittels Unittests weiter getestet werden.

### Modul- und Unittests

Zur Durchführung von Unittests verwenden wir das Framework GoogleTest. Damit soll für die Methoden aller Klassen getestet werden, ob diese entsprechend verschiedenen Eingaben das erwartete Ergebnis produzieren. Wir verwenden White-Box-Tests und wollen eine vollständige Anweisungsüberdeckung, gute Zweig- und Pfadüberdeckung und mindestens einfache Bedingungsüberdeckung erreichen.

### Integrationstests

Sobald die Datenbank eingebunden sowie das GUI mit der Domain Logic zusammengeführt werden, wird die Integration dieser Komponenten ausführlich getestet.

### System Acceptance Test

Alle Tests sollen von der jeweiligen Entwicklerin selbst geschrieben, durchgeführt und dokumentiert werden. Dazu verwenden wir zusätzlich zum Testplan zwei Excel-Tabellen für Test-Case und Test-Suite, in denen die einzelnen Testfälle beschrieben sind.

## Bedienbarkeitstests

Die Bedienbarkeit der Anwendung wird im Umfang dieses Projekts durch einzelne Personen der Altersgruppe zwischen 18 und 40 Jahren mit der dafür typischen Technikaffinität getestet. Ihnen wird die Anwendung ohne weitere Instruktionen bereitgestellt, um zu überprüfen, ob sie diese innerhalb von fünf Minuten verstehen können.

## Zuverlässigkeitstests

Die Zuverlässigkeit wird im Rahmen der Unittests überprüft, da Abstürze oder Fehlfunktionen der Anwendung hier ohnehin ausgeschlossen werden.

## Leistungstests

Durch die Messung der Antwortgeschwindigkeit beim Wechsel zwischen den grafischen Ansichten soll die Leistungsfähigkeit überprüft werden.

## Sicherheitstests

Die Legitimität der Anmeldung wird mit richtigen und falschen Passwörtern getestet.

## Installationstests

Es soll getestet werden, ob die Anwendung auf einem Rechner mit Linux-Betriebssystem ohne Komplikationen oder weitere Voraussetzungen installiert werden kann.

## Testautomatisierung

Alle Tests sollen im Laufe des Projekts durch die Verwendung von GoogleTest in Kombination mit CMake automatisiert werden.

## Verfolgbarkeit

Für alle funktionalen und nichtfunktionalen Anforderungen sowie jede Funktion der Use Cases soll je mindestens ein Test erstellt und durchgeführt werden.

# Übersicht der Testpläne

Alle Dokumente zur Erfassung der Testfälle befinden sich in unserem GitLab-Repository unter folgenden Pfaden:

* Test-Suite: main/Documentation/TestSuite.xlsx
* Test-Case: main/Documentation/TestCase.xlsx

# Freigabe von Testergebnissen

Fehler und Erfolg von Testfällen sollen von der Person klassifiziert werden, die den jeweiligen Test durchführt. Grundsätzlich sollen Test jedoch als „pass“ eingeordnet werden, wenn sie in jedem Fall das erwartete Ergebnis liefern, und als „fail“, wenn dies nicht so ist.