Software Verification Plan

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Autor: Ben Oeckl
Datum: 14.02.2025
Zuletzt geändert:
von: Ben Oeckl
am: 14.02.2025
Version: 1
Prüfer:
Letzte Freigabe:
durch:
am:

Changelog

Datum	Verfasser	Kurzbeschreibung
14.02.2025	Ben Oeckl	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Software Verification Plan

Einleitung und Zielsetzung

Die Verifikation der Software ist notwendig, um eine zuverlässige Funktion in allen Anforderungsbereichen Sicherzustellen und Fehler zu minimieren um Kosten (in Form von Aufwand) gering zu halten.

Im Kontext des Projektes existieren keine kritischen sicherheitsbedingten Anforderungen. Primär geht es beim Testen und Verifizieren dabei, eine reibungslose und zuverlässige Benutzerfahrung zu garantieren.

Anwendungsbereich

Getestet werden sollen alle Funktionen der Anwendung. Darunter fällt die Erstellung, Betrachtung und Verwaltung von Softwareprojekten anhand eigens dafür eingeführten Manifest-Dateien.

Verifikationsstrategie

- Umsetzung von Lasttests (Black-Box-Tests) mit Hilfe von Shellscripts
- Lasttest 1: Anlegen von vielen Projekten
- Lasttest 2: Gleichzeitiges Laden von vielen Projekten
- Oberflächen werden manuell getestet
- Unit-Tests (per CI-Pipeline) (White-Box-/ Grey-Box-Tests) (C1-Zweigüberdeckung)

Verifikationsumgebung

Hardware: x86_64 Rechner / GitHub Actions Umgebung Software:

- Betriebssysteme: Microsoft Windows 10 + 11, Linux (Debian, Ubuntu, Arch, Nix)
- Programme: Episko (Anwendung selber), Grafische Oberfläche
- Unit-Tests: Standard Rust

Testfall-Definition

- Lasttest 1: 10000 Projekte
- Lasttest 2: 100
- Unit-test-Abdeckung: 55 %

Rollen & Verantwortlichkeiten

- Unit-Tests werden von den Entwicklern erstellt.
- Oberflächentests werden von den Entwicklern getätigt

Zeitplan & Meilensteine

- Unit-Test werden während den Entwicklungsphasen erstellt.
- Unit-Tests werden beim Bauen automatisch ausgeführt (CI-Pipeline)
- Oberflächentests werden während der Entwicklung und an der fertigen Anwendung durchgeführt.
- Lasttests werden durchgeführt wenn alle, zu einem Ablauf gehörenden, Funktionen implementiert sind.

Dokumentation & Nachverfolgbarkeit

- Auffälligkeiten bei Oberflächentests werden bei Reviews dokumentiert.
- Unit-Tests werden durch die Implementierung selbst Dokumentiert.
- Die Ergebnisse von Lasttests werden Dokumentiert.