GymTrackerX

Version 1.0

Table of Contents

1. Einleitung 3

1.1 Ziel 3

2. Testabdeckung 3

3. Überblick über die geplanten Tests 4

3.1 Überblick über geplante Tests 4

3.2 Überblick über andere mögliche Tests 4

3.3 Überblick über exkludierte Tests 4

4. Überblick über die Testwerkzeuge 4

5. Verwaltung der Testfälle 4

# Einleitung

## Ziel

* Ziel dieses Testplans ist es, alle notwendigen Informationen zu sammeln, um den Testaufwand für die aktuelle Iteration der *GymTrackerX App* zu planen und zu steuern. Der Plan beschreibt die Teststrategie und dient als Steuerungsinstrument für die Qualitätssicherung im Projekt.
* Dieser Testplan für das Projekt *GymTrackerX* verfolgt folgende Ziele*:*
* Identifikation der zu testenden Komponenten der App
* Überblick über die geplanten Tests
* Beschreibung des Testvorgehens und der Testwerkzeuge
* Verwaltung und Rückverfolgbarkeit der Testfälle

**1.2 Anwendungsbereich**

Testlevel:

* Unit-Tests
* Widget-Tests
* Integrationstests (Basisumfang)

Testarten:

* Funktionalitätstests (Kernfunktionen: Übung erfassen, speichern, anzeigen)
* Usability (nur manuell und grob, keine strukturiere UX-Tests)
* Stabilität (über automatisiere Testwiederholungen geprüft)

Nicht im Anwendungsbereich:

* Sicherheitstests (keine sensiblen Daten vorhanden)
* Performancetests
* Plattformkompatibilitätstests für Altgeräte

# Testabdeckung

Testziele / Komponenten:

* Logik zur Übungs- und Satzerfassung
* Speicher- und Abruffunktionen (lokale Datenbank)
* Benutzeroberfläche für Übungseinträge und Historie
* Navigation zwischen den Hauptbildschirmen
* Externe Komponenten:
* Flutter SDK (vertraut auf Stabilität durch Flutter selbst)
* SQLite (nur über Schnittstellen getestet)

Relative Wichtigkeit:

* Hoch: Übungstracking, Speichern, Anzeigen
* Mittel: UI-Reaktionen
* Niedrig: UI-Layout auf älteren Geräten

# Überblick über die geplanten Tests

* [This section provides a high-level outline of the testing that will be performed. The outline in this section represents a high level overview of both the tests that will be performed and those that will not.]

## Überblick über geplante Tests

* Unit-Tests: Testen von Methoden zur Berechnung, Validierung und Datenspeicherung
* Widget-Tests: Prüfung der Interaktion einzelner UI-Komponenten wie Eingabefelder und Buttons
* Integrationstests: Kernabläufe wie “Trainingsplan erstellen“
* Automatisierte Testausführung über GitHub Actions: alle Tests bei jedem Commit automatisch ausführen

## Überblick über andere mögliche Tests

* Usability-Tests mit Testpersonen: könnten später zur UX-Verbesserung eingesetzt werden
* Performance-/Ladetests bei großen Datenmengen: denkbar, falls bei Nutzung Performanceprobleme auftreten

## Überblick über exkludierte Tests

* Sicherheitstests werden nicht durchgeführt, da keine sensiblen Daten gespeichert oder verarbeitet werden. Begründung: Diese Tests sind in der aktuellen Version der App nicht notwendig.
* Cross-Plattform-Tests auf älteren OS-Versionen (< Android 10 / iOS 13) entfallen. Begründung: Es bestehen nicht genügend Ressourcen zur gezielten Unterstützung von Altgeräten.
* Systematische explorative Tests durch ein QA-Team werden nicht durchgeführt. Begründung: Es gibt kein separates QA-Team im Projekt.

# Überblick über die Testwerkzeuge

* flutter test – für Unit- und Widget-Tests
* flutter driver – für einfache Integrationstests
* GitHub Acxtions – für automatisierte Testläufe bei jedem Commit
* Lcov /genhtml – perspektivisch zur Generierung von Testabdeckungsreports (optional)

# Verwaltung der Testfälle

* Tests sind im Code gespeichert und versioniert über Git
* Jeder Commit oder Pull Request trigger Tests automatisch via GitHub Actions
* Rückverfolgbarkeit: Durch Verknüpfung von Testlogs mit Commits ist nachvollziehbar, welche Tests wann bestanden oder fehlgeschlagen sind
* Manuelle Verwaltung von Testplänen ist nicht notwendig, da durch Automatisierung und saubere Struktur im Testcode die Wartung minimal bleibt