

Testdokumentation en**Courage**

Cole Bailey - Dominik Doerner - René Brandel - Tobias Röddiger

Inhaltsverzeichnis

1.	Einleitung.....	3
2.	Testvorbereitung	4
2.1.	Zu testende Komponenten	4
2.2.	Testvorgaben aus dem Pflichtenheft	5
2.3.	Benutzte Frameworks	7
3.	Automatische Tests	9
3.1.	Unit Tests	9
3.2.	Integration Tests	12
3.3.	Performance Tests	15
4.	Manuelle Tests	17
4.1.	UI Beta-Testing.....	17
4.2.	Testszenarios	18
5.	Verbleibende Bugs.....	22
6.	Anhang.....	23
6.1.	Statistiken.....	23
6.1.	Glossar.....	24
6.2.	UI Testprotokolle.....	25

1. Einleitung

„enCourage“ ist eine moderne Notfallbenachrichtigungs-Applikation, mit der Benutzern die Möglichkeit gegeben wird, schnell und lautlos alle Personen in der näheren Umgebung über einen Notfall zu informieren.

Dieses Dokument beschreibt dabei die Durchführung der Testphase dieser Applikation und baut dabei in großen Teilen auf denen im Pflichtenheft festgelegten Testvorgaben auf.

Die Modultests wurden mit Hilfe des Windows Unit Test Frameworks durchgeführt. Für die Integrations- und Performancetests wurde eine eigenes Test Framework in Form einer Windows App entwickelt.

Die Testszenarien folgen den Vorgaben des Pflichtenhefts. Diese und alle anderen manuellen Tests wurden mit Hilfe von Testprotokollen ausführlich dokumentiert.

Legende Testergebnisse:

Der Test wurde erfolgreich bestanden	✓
Es gab Probleme mit dem Test	(✓)
Der Test ist fehlgeschlagen	X
Test benötigt keinen Serverkontakt	-

2. Testvorbereitung

2.1. Zu testende Komponenten

Durch den hohen Kommunikationsaufwand der Applikation besteht jeder Aufruf aus einer großen Anzahl verschachtelter Aufrufe und damit vielen Abhängigkeiten. Aus diesem Grund sollen **ALLE FASSADENSCHNITTSTELLEN** der verschiedenen Subsysteme linear getestet und integriert werden.

Testziel ist hierbei eine komplette Abdeckung aller Schnittstellenmethoden mit Testfällen (*Integration Tests*), eine besondere Überprüfung von leeren und illegalen Eingaben (*Invalid Input Tests*) sowie die Überprüfung von Eingaben unter falschen Bedingungen/Zuständen (*Stress Tests*).

Ein ausgiebiges Überprüfen einzelner Klassen ist aufgrund der oben genannten Gründe nicht für alle Klassen sinnvoll, da viele Methoden der Kommunikation dienen und entsprechend kurz sind. Wir haben uns deshalb dazu entschieden, nur jene Klassen separat zu testen, die eine interne Logik kapseln und/oder (durch Komplexität oder Abhängigkeiten) besonders fehleranfällig sind. Darunter fallen **EMERGENCYCONTAINER**, **BOUNCER**, **UNIVERSALSERIALIZER**, **PUSHPROCESSOR** und **SETTINGS**.

Aufgrund des Zwecks unserer Applikation ist natürlich auch eine umfangreiche Prüfung der Intuitivität und Bedienbarkeit der **UI** erforderlich, was allerdings nur mit manuellen Tests möglich und entsprechend ausführlich im zugehörigen Kapitel beschrieben ist.

Nach längerer Analyse kam zudem heraus, dass die größten möglichen Quellen für Fehler die **ROLLEN** eines Benutzers bei einem Notfall sowie die **PUSH-BENACHRICHTIGUNGEN** sind, da diese maßgeblich den Zustand der Applikation steuern. Aus diesem Grund wurden alle möglichen Kombinationen aus Zuständen und Benachrichtigungen sowie alle (erlaubten sowie nicht erlaubten) Rollenübergänge in den Integrationstests überprüft.

2.2. Testvorgaben aus dem Pflichtenheft

Hinweis: Braun markierte Testfälle waren für Funktionalitäten vorgesehen, die aus verschiedenen Gründen bewusst nicht implementiert wurden und werden dementsprechend in dieser Phase (und diesem Dokument) nicht weiter beachtet.

1 - Melden eines Notfalls	
[TB1010]	Notfall melden
[TB1020]	Position des Melders erkennen und verfolgen
[TB1030]	Notfall-Meldung abbrechen
[TB1040]	Notfall beenden
[TB1050]	Notfall über LiveTile melden
[TB1060]	Automatisches Verstummen des Alarms nach festgelegter Zeit
[TB1070]	Verstummen Notfall automatisch vom Server beenden lassen
[TB1080]	Alarm wegen ausreichender Anzahl von Helfern verstummen lassen
[TB1100]	Alarm-Zeitlimit-Warnung ignorieren
[TB1110]	Zeitlimit des Alarms verlängern
[TE1010]	Verknüpften Personen stetig die aktualisierte Position mitteilen
[TE1020]	Informierten stetig die aktualisierte Position mitteilen
[TE1030]	Behörden informieren
[TE1040]	Benachrichtigungsradius automatisch vergrößern
[TE1050]	Ähnlichen Notfall als identisch melden
[TE1060]	Ähnlichen Notfall separat melden

2 – Notfalldetails	
[TB2010]	Notfall vor dem Melden spezifizieren
[TB2020]	Gemeldeten Notfall spezifizieren
[TB2030]	Anzahl der Helfenden überprüfen
[TB2040]	Details anzeigen lassen
[TB2050]	Position überprüfen
[TB2060]	Position in externen Navigations-App anzeigen lassen
[TE2010]	Anzahl der helfenden Experten anzeigen
[TE2020]	Überprüfen, dass die richtigen Experten angezeigt werden
[TE2030]	Kartenansicht für die Position des Notfalls prüfen

3 – Notfall-Benachrichtigungen erhalten	
[TB3010]	Aktive Personen in der näheren Umgebung benachrichtigen
[TB3020]	Benachrichtigung als Helfender beantworten
[TB3030]	Benachrichtigung ignorieren
[TB3040]	Auf Benachrichtigung klicken, um die Notfalldetails einzusehen
[TB3050]	Alarm als Missbrauch melden

4 – App-Einstellungen	
[TB4010]	Ruhemodus anschalten
[TB4020]	Ruhemodus ausschalten
[TB4030]	Sicherheitsfrage setzen
[TB4040]	Sicherheitsfrage ändern
[TB4050]	Sicherheitsfrage entfernen
[TE4010]	Qualifikationen eingeben

[TE4020]	Qualifikationen ändern
[TE4040]	Aktualisierungsrate neu einstellen
[TE4050]	Bestimmte Zeitspanne für den Ruhemodus einstellen

5 – Verknüpfte Personen	
[TB5010]	Neue verknüpfte Person hinzufügen
[TB5020]	Verknüpfte Person entfernen
[TB5040]	Verknüpfte Personen benachrichtigen
[TE5020]	Als verknüpfte Person das Zeitlimit des Alarms verlängern
[TE5030]	Als verknüpfte Person das Zeitlimit des Alarms verstreichen lassen

6 – LiveFeed	
[TB6010]	Notfälle in der größeren Umgebung anzeigen
[TB6020]	Aktuelle Notfällen in einer Liste anzeigen lassen
[TB6030]	Auf einen Notfall klicken, um die Details einzusehen
[TE6010]	Kartenansicht für einen einzelnen Notfall prüfen
[TE6020]	Kartenansicht für mehrere gleichzeitige Notfälle prüfen

7 – Serververbindung	
[TB7010]	Position übertragen
[TB7020]	Spezifikationen übertragen
[TB7030]	Positionen aktualisieren
[TE7010]	Die Position des Melders stetig aktualisieren

2.3. Benutzte Frameworks

Microsoft Unit Test Framework

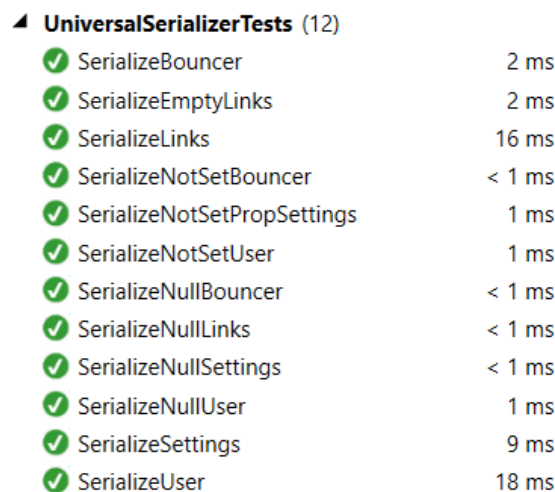
Das Microsoft Unit Test Framework ermöglicht es, Unit Tests in Visual Studio integriert durchzuführen. Die Modultests (Unit-Tests) dienen dazu, funktionale Einzelteile des Programms (und damit insbesondere einzelne Klassen) zu testen, d.h. sie auf korrekte Funktionalität zu prüfen.

Bei der Programmierung der Tests nutzt man den `UnitTestFramework` Namespace. Eine Testklasse wird durch `[TestClass]` gekennzeichnet. In dieser Testklasse sind einzelne Testmethoden enthalten, gekennzeichnet durch `[TestMethod]` welche jeweils einen einzelnen Unit-Test repräsentieren. Eine solche Testmethode lässt sich wiederum in drei Abschnitte unterteilen:

ARRANGE: Hier werden alle Vorbedingungen für den Unit Test geschaffen.

ACT: Hier wird die zu überprüfende Funktionalität ausgeführt.

ASSERT: Hier wird überprüft, ob die in ACT ausgeführte Funktionalität korrekt abgelaufen ist.



▲ UniversalSerializerTests (12)	
✓ SerializeBouncer	2 ms
✓ SerializeEmptyLinks	2 ms
✓ SerializeLinks	16 ms
✓ SerializeNotSetBouncer	< 1 ms
✓ SerializeNotSetPropSettings	1 ms
✓ SerializeNotSetUser	1 ms
✓ SerializeNullBouncer	< 1 ms
✓ SerializeNullLinks	< 1 ms
✓ SerializeNullSettings	< 1 ms
✓ SerializeNullUser	1 ms
✓ SerializeSettings	9 ms
✓ SerializeUser	18 ms

Abbildung 2.1 – Ausschnitt der Unit Test Ergebnisse

Visual Studio bietet die Möglichkeit, diese Tests dann automatisch ausführen zu lassen und visualisiert die Ergebnisse in einem eigenen Reiter. Die Integration in die Entwicklungsumgebung ist vor allem dazu hilfreich, die automatische Ausführung vor jedem Build anzustoßen.

Eigenentwickeltes Integration Test Framework

Für die Integration Tests war es sehr wichtig, dass die Testfälle in einer festen Reihenfolge durchgeführt werden, um aufeinander aufbauen zu können und Datenduplizierung zu vermeiden. Eine Folge von Testfällen kann dabei einer typischen Anwendungsfolge ähneln um auch Abhängigkeiten besser zu untersuchen.

Um dies zu ermöglichen, entschlossen wir uns, ein eigenes Test Framework zu entwickeln. Ganz ähnlich des Unit Test Frameworks sollte dabei eine Klasse oder Schnittstelle separat getestet werden. Die Testklasse selbst erbt von einer abstrakten Test-Superklasse, die Assert-Methoden, Fehlererwartungen sowie Testausführung und Protokollierung kapselt und herauszieht und definiert selbst nur die eigentlichen Testmethoden sowie Testobjekte.

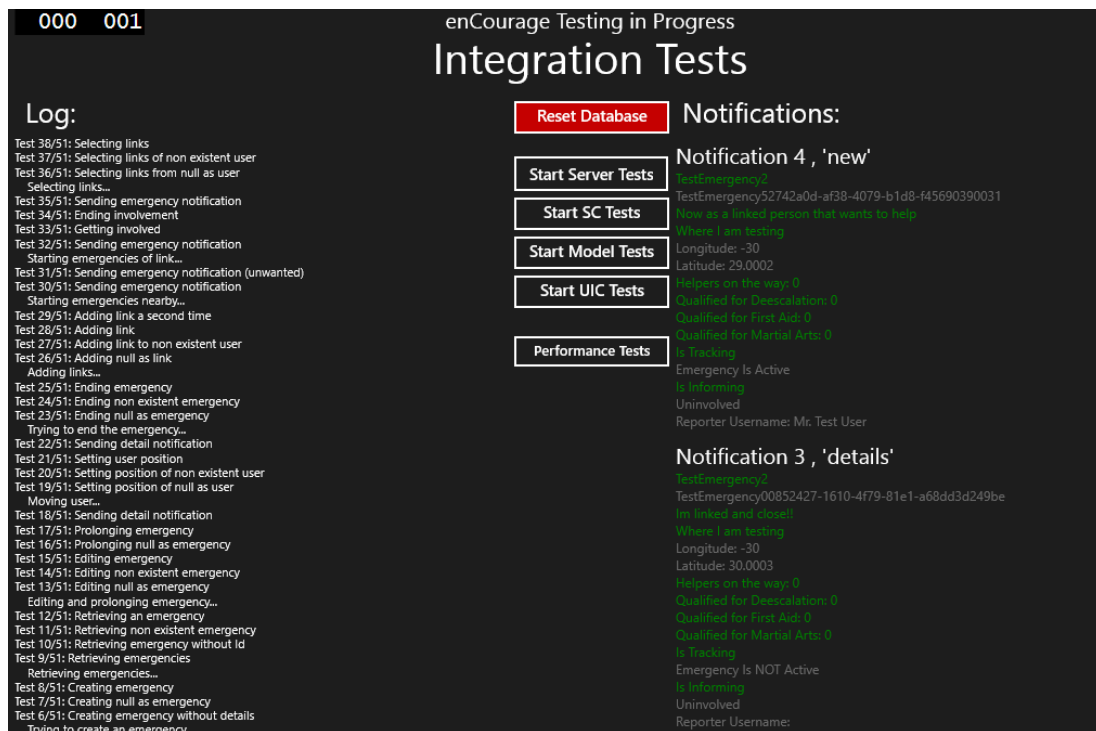


Abbildung 2.2 – UI des eigenen Test-Frameworks

Es viel uns damit auch leichter, während den Tests mit der Datenbank zu interagieren, um die ankommenden Daten zu prüfen, asynchrone Methoden auszuführen, sowie ankommende Push-Benachrichtigungen besser zu überprüfen und anzuzeigen.

Eine Ausführung aller Tests vor jedem Build ist damit leider nicht möglich, weswegen wir uns auch darauf beschränken, nur die Integration oder Performance Tests mit diesem Framework umzusetzen.

3. Automatische Tests

3.1. Unit Tests

Bouncer

Testmethode	Testbeschreibung	Ergebnis
NullInputTests	Dieser Test checkt, ob die Sicherheitsantwort automatisch auf einen leeren String gesetzt wird (ohne sie zu hashen), wenn die Antwort auf null gesetzt wird.	✓
EmptyInputTest	Dieser Test stellt sicher, dass der Bouncer einen leeren String nicht hasht.	✓
HashTest	Hier wird die Sicherheitsantwort gesetzt und überprüft, ob der SHA-512 Hash richtig berechnet wurde.	✓
CheckerTrueTest	In diesem Test wird überprüft, ob die CheckSecurityAnswer Funktion bei einer richtigen Eingabe wahr zurückgibt.	✓
CheckerFalseTest	In diesem Test wird überprüft, ob die CheckSecurityAnswer Funktion bei einer falschen Eingabe falsch zurückgibt.	✓
SecurityIsEnabledTest	Hier wird überprüft, ob die Sicherheitsabfrage aktiv ist, wenn ein nicht leerer String als Sicherheitsantwort gesetzt wurde. In diesem Fall sollte das Feld IsSecurityEnabled wahr zurückgeben.	✓
SecurityIsNotEnabled	Hier wird überprüft, ob die Sicherheitsabfrage deaktiviert ist, wenn ein leerer String als Sicherheitsantwort gesetzt wurde. In diesem Fall sollte das Feld IsSecurityEnabled falsch zurückgeben.	✓

EmergencyContainer

Testmethode	Testbeschreibung	Ergebnis
AddEmergencyObserverTest	Überprüft, ob dem EmergencyContainer erfolgreich ein Beobachter hinzugefügt werden kann.	✓
UpdateCurrentEmergencyDetailsTest	Überprüft, ob die Details des aktuellen Notfalls korrekt aktualisiert werden.	✓
UpdateEmergencyPositionTest	Überprüft, ob die Position eines Notfalls korrekt aktualisiert wird und diese Änderung an die Beobachter weitergegeben wird.	✓
UpdateEmergenciesTest	Überprüft, ob eine Liste von Notfällen korrekt vom EmergencyContainer verarbeitet wird und bei den Beobachtern ankommt.	✓
ReinitializeActiveEmergency	Überprüft ob ein Notfall richtig reinitialisiert wird und der Aufruf an die Beobachter korrekt ist.	✓
StopCurrentActiveEmergencyTest	Überprüft, ob der aktuell aktive Notfall erfolgreich im EmergencyContainer beendet wird und die Beobachter darüber benachrichtigt werden.	✓
UpdateEmergencyDetailsTest	Überprüft, ob die Details eines einzelnen Notfalls korrekt aktualisiert werden und diese Änderung bei den Beobachtern ankommt.	✓
UpdateEmergencyRoleToReporterTest	Überprüft, ob die Rolle in einem Notfall zu Reporter geändert werden kann (entspricht dem Melden eines Notfalls) und der zugehörige Beobachter Aufruf korrekt ist.	✓
UpdateEmergencyRoleToHelperOrInvolvedTest	Überprüft, ob die Rolle in einem Notfall zu Helfer oder Involved geändert werden kann.	✓

UpdateEmergencyRoleIgnorerFromHelperTest	Überprüft, ob die Rolle in einem Notfall von Helper zu Ignorer geändert werden kann.	✓
InformTimeoutWarningTestTest	Überprüft, ob eine TimeoutWarning korrekt vom EmergencyContainer verarbeitet wird.	✓
UpdateEmergencyRoleTolgnorerFromInvolvedTest	Überprüft, ob die Rolle in einem Notfall von Involved auf Involved geändert werden kann.	✓
UpdateEmergencyRoleUninvolvedFromInvolvedTest	Überprüft, ob die Rolle von Involved zu Uninvolved geändert werden kann.	✓
AddEmergencyNotificationTest	Überprüft, ob ein Notfall der über eine Benachrichtigung erhalten wurde vom EmergencyContainer korrekt verarbeitet und an die Beobachter weitergeleitet wird.	✓

Emergency

Testmethode	Testbeschreibung	Ergebnis
UpdateTestValidInputs	Überprüft, ob ein Notfall korrekt aktualisiert wird.	✓
UpdateTestInvalidId	Überprüft, ob ein Notfall nicht aktualisiert wird, wenn die ID des Notfalls nicht mit der ID des aktualisierten Notfalls übereinstimmt.	✓

Settings

Testmethode	Testbeschreibung	Ergebnis
DNDActiveTest	Überprüft die Einstellungen mit einem aktivierten Ruhemodus.	✓
DNDInactiveTest	Überprüft die Einstellungen mit einem deaktivierten Ruhemodus.	✓
DNDTimerActiveTest	Überprüft die Einstellungen mit einem aktivierten Ruhemodus-Timer, der sich über die aktuelle Zeit erstreckt	✓
DNDTimerInactiveTest	Überprüft die Einstellungen mit einem deaktivierten Ruhemodus-Timer, der sich über die aktuelle Zeit erstreckt.	✓
DNDTimerInvalidTimeSpansTest	Überprüft die Einstellungen mit einem aktivierten Ruhemodus-Timer, der sich nicht über die aktuelle Zeit erstreckt.	✓
DNDTimerDifferentDatesInvalidTimeSpansTest	Überprüft die Einstellungen mit einem aktivierten Ruhemodus-Timer, der sich nicht über die aktuelle Zeit erstreckt, aber nicht über den heutigen Tag.	✓
DNDTimerDifferentDatesValidTimeSpansTest	Überprüft die Einstellungen mit einem aktivierten Ruhemodus-Timer, der sich über die aktuelle Zeit erstreckt, aber nicht über den heutigen Tag.	✓

UniversalSerializer

Testmethode	Testbeschreibung	Ergebnis
SerializeUser	In diesem Test wird eine User Instanz serialisiert und direkt danach deserialisiert und überprüft, ob es sich um den selben Benutzer handelt.	✓
SerializeNullUser	In diesem Test wird geprüft, ob die SerializeUser Methode von UniversalSerializer null Eingaben abfängt.	✓

SerializeNotSetUser	Hier wird eine User Instanz über den Standard Konstruktor erzeugt und serialisiert ohne ihre Felder zu setzen. Danach wird sie umgehend deserialisiert, um zu schauen, ob es sich immer noch um selben User handelt.	✓
SerializeSettings	In diesem Test wird eine Settings Instanz serialisiert und direkt danach deserialisiert und überprüft, ob die Einstellungen gleich sind.	✓
SerializeNullSettings	In diesem Test wird geprüft, ob die SerializeSettings Methode von UniversalSerializer null Eingaben abfängt.	✓
SerializeNotSetPropSettings	Hier wird eine Settings Instanz über den Standard Konstruktor erzeugt und serialisiert, ohne ihre Felder zu setzen. Danach wird sie umgehend deserialisiert, um zu überprüfen, ob es sich immer noch um die selben Einstellungen handelt.	✓
SerializeBouncer	In diesem Test wird eine Bouncer Instanz serialisiert und direkt danach deserialisiert und überprüft, ob es sich um den selben Bouncer handelt.	✓
SerializeNullBouncer	In diesem Test wird geprüft, ob die SerializeBouncer Methode von UniversalSerializer null Eingaben abfängt.	✓
SerializeNotSetBouncer	Hier wird eine Bouncer Instanz über den Standard Konstruktor erzeugt und serialisiert ohne ihre Felder zu setzen. Danach wird sie umgehend deserialisiert, um zu schauen, ob es sich immer noch um den selben Bouncer handelt.	✓
SerializeLinks	In diesem Test wird ein Dictionary von Links Instanzen serialisiert und direkt danach deserialisiert und überprüft, ob es sich um die gleichen Verknüpfungen handelt.	✓
SerializeEmptyLinks	Hier wird ein leeres Dictionary von Links serialisiert und danach überprüft, ob das deserialisierte Dictionary ebenfalls leer ist.	✓
SerializeNullLinks	In diesem Test wird geprüft, ob die SerializeLinks Methode von UniversalSerializer null Eingaben abfängt.	✓

PushProcessor

Testmethode	Testbeschreibung	Ergebnis
AddObserverTest	Fügt einen gültigen IPushDataObserver hinzu.	✓
AddNullObserverTest	Fügt null als Observer hinzu.	✓
ValidNewEmergencyTest	Interpretiert eine gültige Enkodierung eines Notfalls.	✓
ValidDetailsTest	Interpretiert eine gültige Enkodierung einer Detailänderung.	✓
ValidPositionTest	Interpretiert eine gültige Enkodierung einer Positionsänderung.	✓
ValidTimeoutTest	Interpretiert eine gültige Enkodierung eines Zeitlimits.	✓
ClickedNotificationTest	Interpretiert eine gültige Enkodierung eines ausgewählten Notfalls.	✓
DetailsValidTimeStampsTest	Interpretiert zwei Detailsänderungen in der validen Reihenfolge.	✓
DetailsInvalidTimeStampsTest	Interpretiert zwei Detailsänderungen in der umgekehrten Reihenfolge.	✓
MissingMsgtypeTest	Interpretiert eine Enkodierung ohne Nachrichtszweck.	✓
MissingIDTest	Interpretiert eine Enkodierung ohne Notfall-ID.	✓
MissingLatitudeTest	Interpretiert eine Enkodierung ohne geographische Breite.	✓
MissingLongitudeTest	Interpretiert eine Enkodierung ohne geographische Länge.	✓
MissingUnnecessaryFieldsTest	Interpretiert eine Enkodierung mit allen notwendigen Daten (Zweck, ID, Position) und sonst nichts.	✓

3.2. Integration Tests

Hinweis: Als Integrationstest des Views dienen die manuellen Testszenarien im nächsten Kapitel. Ebenso sind hier aufgrund der schieren Menge der Testfälle mehrere Methoden zu einem logischen Block zusammengefasst.

Server

Testblock	Testbeschreibung	Ergebnis
Einfügen von Benutzern	Fügt alle Benutzer für die Tests in die Datenbank ein und überprüft die Registrierung dieser.	✓
Einfügen von Notfällen	Überprüft die Meldung von Notfälle mit gültigen, ungültigen und leeren Daten.	✓
Notfälle abrufen	Überprüft die Abfrage von existierenden und nicht-existierenden Notfällen in der Datenbank.	✓
Notfall melden	Setzt einen Benutzer als Melder eines Notfalls und überprüft die Rolle.	✓
Position aktualisieren	Aktualisiert die Benutzer-Position sowie die automatische Aktualisierung der Notfall-Position bei aktivierter Verfolgung.	✓
Notfälle aktualisieren	Aktualisiert einen gemeldeten Notfall mit gültigen und ungültigen Daten.	✓
Bei einem Notfall helfen	Fügt Helfer und Experten hinzu und überprüft die Aktualisierung der jeweiligen Zähler im Notfall.	✓
Helfer eines Notfalls entfernen	Entfernt Helfer und Experten und überprüft die Aktualisierung der jeweiligen Zähler.	✓
Notfälle in der Umgebung anfordern	Fordert eine Liste aller Notfälle der Umgebung an, sowohl mit einigen als auch mit keinen im jeweiligen Radius.	✓
Notfall löschen	Überprüft das Beenden eines Notfalls	✓
Verknüpfung hinzufügen	Fügt verknüpfte Personen hinzu und fragt diese ab.	✓
Verknüpfung löschen	Entfernt verknüpfte Personen.	✓

ServerController

Testblock	Testbeschreibung	Ergebnis
PushDataObserver hinzufügen	Fügt einen PushDataObserver-Mock als Observer hinzu, um damit die Tests besser überprüfen zu können.	✓
Aktuelle Version anfordern	Frägt die aktuelle Version aus dem Server ab und kontrolliert diese.	✓
Benutzer registrieren	Überprüft die Registrierung von Benutzern am Server.	✓
Notfall erstellen	Überprüft die Erstellung von Notfällen vom Client aus.	✓
Notfälle anfordern	Frägt einzelne, mehrere oder ungültige Notfälle vom Server ab.	✓
Notfall bearbeiten	Bearbeitet und verlängert gültige und ungültige Notfälle und überprüft die dadurch resultierenden Benachrichtigungen.	✓
Position aktualisieren	Aktualisiert die Benutzer-Position und überprüft die automatische Aktualisierung der Position des von dieser Person gemeldeten Notfalls.	✓
Notfall beenden	Überprüft, ob ein Notfall vom Melder und nur vom Melder beendet werden kann.	✓
Verknüpfung hinzufügen	Überprüft, dass Verknüpfungen zwischen Benutzers nur einmal erstellt werden können, und nur, wenn beide Benutzer existieren.	✓

Notfälle in der Nähe empfangen	Erstellt mehrere Notfälle in der Nähe, um den Benachrichtigungsradius sowie das korrekte Empfangen der Benachrichtigungen zu prüfen.	✓
Notfälle der Verknüpfung empfangen	Erstellt mehrere Notfälle von verknüpften Personen und überprüft Benachrichtigungen sowie Interaktionen mit diesen.	✓
Verknüpfungen anfordern	Überprüft die Abfrage aller Verknüpfungen vom Server.	✓
Helferrollen durchtesten	Überprüft die Rollenübergänge sowie korrekten Benachrichtigungen für Involvierte und Helfer.	✓
Notfall ignorieren	Überprüft das Ignorieren und damit automatische Abmelden eines Notfalls.	✓
Verknüpfung entfernen	Entfernt Verknüpfungen und kontrolliert, dass danach keine Benachrichtigungen mehr erhalten werden.	✓
Benachrichtigungen überprüfen	Vergleicht die Benachrichtigungen, die empfangen wurden, mit denen, die erwartet wurden.	(✓)

Model

Testblock	Testbeschreibung	Ergebnis
EmergencyObserver hinzufügen	Fügt einen EmergencyObserver-Mock als Observer hinzu, um damit die Tests besser überprüfen zu können.	✓
Lokale Daten anfordern	Überprüft die Anforderung und den korrekten Erhalt der lokalen Daten aus dem Model (User, Settings, Security)	✓
Verknüpfungen verwalten	Überprüft das korrekte Hinzufügen, Abfragen und Löschen von Verknüpfungen.	✓
Einstellungen und Profil verwalten	Ändert und kontrolliert App- sowie Profil-Einstellungen.	✓
Sicherheitseinstellungen verwalten	Bearbeitet die Sicherheitseinstellungen und überprüft die korrekte Verwendung der Sicherheitsfunktionalitäten.	✓
Sicherheitsfunktionalität ausschalten	Überprüft die Eingabe einer falschen Sicherheitsantwort sowie die Deaktivierung der Sicherheitsfunktion.	✓
Notfälle anfordern	Fordert die Notfälle in der Umgebung an und kontrolliert diese.	✓
Involviert werden	Registriert den Benutzer bei einem Notfall und überprüft, dass andere Rollen nicht zuvor angenommen werden können.	✓
In einem Notfall helfen	Hilft bei einem Notfall und überprüft die Stabilität des aktuellen Zustands.	✓
Rollenübergänge testen	Überprüft alle bisher nicht durchgeführten Übergänge zwischen Rollen, erlaubt sowie nicht erlaubt, auf die entwurfstreue Reaktion des Systems.	✓
Eigenen Notfall melden	Meldet einen eigenen Notfall, bearbeitet diesen und überprüft die Stabilität des aktuellen Zustands.	✓
Illegale Rollenübergänge testen	Stellt sicher, dass der Melder nicht zu einer anderen Rolle übergehen kann.	✓
Notfall beenden	Beendet den eigenen Notfall.	✓

UIController

Testblock	Testbeschreibung	Ergebnis
Lokale Daten anfordern	Lokal gespeicherte Daten (Profil, Einstellungen, Verknüpfungen, UserID) werden angefordert und überprüft.	✓
Verknüpfungen verwalten	Überprüft das Hinzufügen, Umbenennen und Entfernen von verknüpften Personen.	✓
Lokale Daten ändern	Ändert die lokalen Einstellungen und das Profil des Benutzers ab.	✓

Sicherheitseinstellungen ändern	Aktiviert und deaktiviert die Sicherheitsfrage und überprüft deren korrekte Verwendung bei den privilegierten Aktionen.	✓
Notfall öffnen	Öffnet den LiveFeed und meldet sich bei einem Notfall an.	✓
Notfallan-/abmeldung	Überprüft das stabile An- und Abmelden von Notfällen.	✓
Notfallinteraktionen	Meldet den Benutzer als Helfer an und ignoriert dann den Notfall.	✓
Notfall melden	Meldet, bearbeitet und verlängert einen eigenen Notfall.	✓
Notfall beenden	Beendet den eigenen Notfall.	✓

3.3. Performance Tests

Hinweis: Die Performance Tests wurden mit dem selbst entwickelten Integration Test Framework erstellt und ausgeführt.

Emergency Creation Speed Test

Testbeschreibung: Misst die Zeit, die zwischen dem Aufruf zum Melden eines Notfalls und der danach eingehenden Bestätigung des Servers vergeht.

Testumfang: 1 neuer Notfall

Zeitlimit: 4000 ms

Testergebnis: 1177 ms

Notification Speed Test

Testbeschreibung: Misst die Zeit, die zwischen dem Versenden einer Benachrichtigung am Server und dem Eingehen der Benachrichtigung am Client vergeht.

Testumfang: 50 versendete Benachrichtigungen

Zeitlimit: 4000 ms

Testergebnis:

- Minimum: 270 ms
- Maximum: 3813 ms
- Durchschnitt: 606 ms

Notification Reliability Test

Testbeschreibung: Überprüft, ob eine versendete Benachrichtigung innerhalb des Zeitlimits den Client erreicht.

Testumfang: 100 versendete Benachrichtigungen

Zeitlimit: 4000 ms

Testergebnis: 96 / 100

Position Update Concurrency Test

Testbeschreibung: Überprüft, wie viele nahezu gleichzeitig versendete Aufrufe zum Aktualisieren der Position desselben Benutzers ohne Schreib-Konflikte im Server ausgeführt werden. Ein missglückter Aufruf wird dabei im Client später ignoriert und hat somit keinen Einfluss auf die Applikation.

Testumfang: 100 Update Aufrufe

Testergebnis: 68 / 100

Helper Registration Concurrency Test

Testbeschreibung: Überprüft, wie viele Helfer, die sich nahezu gleichzeitig anmelden, ohne Schreib-Konflikte im Server registriert werden.

Ein missglückter Aufruf kann dabei einen großen Einfluss auf die Applikation haben, da es zu Inkonsistenzen mit diesem Helfer und der Helferanzahl eines Notfalls kommen kann.

Testumfang: 20 neue Helfer

Testergebnis: 20 / 20

4. Manuelle Tests

4.1. UI Beta-Testing

Mit Intuitivität und Bedienbarkeit als primären Qualitätszielen war es sehr wichtig, die Verwendung der App von Außenstehenden testen und bewerten zu lassen.

Mehrere Testpersonen, die sich nicht mit der Applikation auskannten, wurden dabei gefragt, bestimmte Aktionen auszuführen, ohne dass dies anhand einer Anleitung erklärt wurde. Dabei wurde genau protokolliert, wie schwer es ihnen fiel, die jeweiligen Seiten und Funktionalitäten zu finden und wie gut sie die Verwendung im Allgemeinen fanden.

Diese entstandenen Protokolle, die nach Personen und Aktionen getrennt waren, sind im Anhang zu finden und im Folgenden der Übersicht wegen zusammengefasst. Ein Test ist dann als problematisch oder fehlgeschlagen markiert, wenn das Ergebnis bei mindestens einer Person war.

Testpersonenbeschreibung: Ohne Kenntnis über Windows Phone, Altersgruppe 20-25, Student

Zusammenfassung der Tests:

Testbeschreibung	Ergebnis
App starten	✓
AGB akzeptieren	✓
Notfall auslösen	(✓)
Details editieren	✓
Positionsverfolgung deaktivieren	✓
Alarm beenden	✓
Profil Name festlegen	✓
Verknüpfte Person hinzufügen	X
Ruhemodus aktivieren	X
Notfälle aus der Umgebung anzeigen	X
Notification erhalten	✓
Bei diesem Notfall helfen	✓
Notfall über Navigation im Notfall Radar ignorieren	✓
Live Tile zum Melden eines Alarms aktivieren	(✓)
Verknüpfte Person umbenennen	✓
Sicherheitsfrage festlegen	✓
Qualifikation "Erste-Hilfe" hinzufügen	✓
Verknüpfte Person löschen	(✓)
Eigenen Profilcode per SMS teilen	(✓)
Zeitgesteuerten Ruhemodus von 12:00 bis 16:00 Uhr aktivieren	✓
Notfall-Auslösung abbrechen	✓
Notfall vor dem Auslösend des Alarms spezifizieren und Links nicht benachrichtigen	✓
Missbrauch bei einem Notfall melden	✓

4.2. Testszenarien

Hinweis: Die Testszenarien [TS01] – [TS06] wurden aus dem Pflichtenheft übernommen und bündeln einzelne Testfälle. Darüber hinaus wurden diese noch um die weiteren Szenarien [TS07] – [TS08] ergänzt, um weitere mögliche Arten der Benutzung zu überprüfen.

[TS01] – Typischer Notfallablauf

Ein Benutzer meldet einen Notfall, wodurch alle aktiven Personen in der näheren Umgebung benachrichtigt werden. Der Melder gibt weitere Spezifikationen ein, die automatisch auch bei den Informierten aktualisiert werden. Von drei Personen meldet sich eine als Helfer, eine ignoriert die Benachrichtigung und eine sieht sich die Notfalldetails an. Nachdem der Notfall beseitigt wurde, beendet der Melder den Alarm.

Test-ID	Testbeschreibung	Client	Server	Kommentar
[TB1010]	Notfall melden	✓	✓	
[TB1020]	Melder verfolgen	✓	✓	5 Positions Updates
[TB3010]	Aktive Personen in der näheren Umgebung benachrichtigen lassen	✓	-	
[TB2020]	Gemeldeten Notfall spezifizieren	✓	✓	
[TB3020]	Benachrichtigung als Helfender beantworten	✓	✓	
[TB3030]	Benachrichtigung ignorieren	✓	-	
[TB3040]	Auf Benachrichtigung klicken, um die Notfalldetails anzusehen	✓	-	Bei geöffneter App über Dialog sichtbar
[TB2040]	Details anzeigen lassen	✓	✓	
[TB2030]	Anzahl der Helfenden überprüfen	✓	✓	
[TB2050]	Position überprüfen	✓	✓	
[TB2060]	Position in externer Navigations-App überprüfen	✓	-	
[TB1040]	Notfall beenden	✓	✓	

[TS02] – Ablauf eines über das LiveTile gemeldeten Alarms

Eine Person meldet einen Notfall über das LiveTile auf seinem Gerät, woraufhin der Server die Personen in der Nähe benachrichtigt. Ein eingetroffener Informierter hingegen kann keine Ursache für das Melden finden und markiert deshalb den Notfall als einen Missbrauch. Da der Melder die App nicht weiter benutzt, verstummt der Alarm automatisch nach dem verstrichenen Zeitlimit und wird etwas später vom Server beendet.

Test-ID	Testbeschreibung	Client	Server	Kommentar
[TB1050]	Notfall über LiveTile melden	✓	✓	
[TB1020]	Melder verfolgen	✓	✓	
[TB7010]	Position übertragen	✓	✓	
[TB7030]	Positionen aktualisieren	✓	✓	
[TB3010]	Aktive Personen in der näheren Umgebung benachrichtigen lassen	✓	-	
[TB3050]	Notfall als Missbrauch melden	✓	✓	
[TB1060]	Alarm automatisch nach Zeitlimit verstummen lassen	(✓)	✓	Gelegentlich Absturz bei Helfern oder Links
[TB1070]	Verstummten Notfall automatisch beenden	✓	✓	

[TS03] – Einstellungen und verknüpfte Personen konfigurieren

Ein Benutzer kümmert sich um die Einstellungen seiner frisch installierten App. Er schaltet dabei den Ruhemodus an, setzt seine persönliche Sicherheitsfrage, und fügt verknüpfte Personen hinzu. Etwas später ändert er seine Meinung und setzt alles Eingestellte wieder zurück.

Test-ID	Testbeschreibung	Client	Server	Kommentar
[TB4010]	Ruhemodus anschalten	✓	-	
[TB4030]	Sicherheitsfrage setzen	✓	-	
[TB5010]	Neue verknüpfte Person hinzufügen	✓	✓	
[TB5020]	Verknüpfte Person entfernen	✓	✓	
[TB4040]	Sicherheitsfrage ändern	✓	-	
[TB4050]	Sicherheitsfrage entfernen	✓	-	
[TB4020]	Ruhemodus ausschalten	✓	-	

[TS04] – Den Melder stetig verfolgen

Eine Person will zuerst einen Notfall unspezifiziert melden, entscheidet sich dann aber anders und bricht die Meldung ab, um die Spezifikation zuerst vorzunehmen. Zusätzlich gibt er an, dass seine Position nicht nur für die verknüpften Personen, sondern alle Informierten stetig verfolgt werden soll. Es melden sich auch in kurzer Zeit viele Personen als Helfer, sodass der Alarm durch das Personenlimit automatisch verstummt. Der Notfall kann schnell gelöst werden und der Melder beendet seinen Alarm.

Test-ID	Testbeschreibung	Client	Server	Kommentar
[TB1030]	Notfall-Meldung abbrechen	✓	-	
[TB2010]	Notfall vor dem Melden spezifizieren	✓	-	
[TB1010]	Notfall melden	✓	✓	
[TB5040]	Verknüpfte Personen benachrichtigen	✓	-	
[TB3010]	Aktive Personen in der näheren Umgebung benachrichtigen lassen	✓	-	
[TE7010]	Die Position des Melders stetig aktualisieren	✓	✓	
[TE1010]	Verknüpften die stetig aktualisierte Position mitteilen	✓	-	
[TE1020]	Informierten die stetig aktualisierte Position mitteilen	✓	-	
[TB1080]	Alarm automatisch aufgrund der Anzahl der Helfenden verstummen lassen	✓	✓	
[TB1040]	Notfall beenden	✓	✓	

[TS05] – LiveFeed

Kurz hintereinander werden in einer kleineren Umgebung drei Notfälle gemeldet, weswegen die letzten beiden Melder gefragt werden, inwieweit ihre Notfälle identisch sind. Der zweite Melder sieht ein, dass er einen Notfall melden will, der bereits aktiv ist und bricht deshalb seine Meldung ab. Der Dritte hingegen ist Zeuge eines separaten Notfalls und gibt deshalb eine neue Meldung aus.

Ein anderer Benutzer sieht die beiden gemeldeten Notfälle auf seinem LiveFeed. Die Positionen und Entfernungen entnimmt er der Karte und der Liste und sieht sich bei einem der Notfälle die weiteren Notfalldetails an.

Test-ID	Testbeschreibung	Client	Server	Kommentar
[TB1010]	Notfall melden	✓	✓	
[TB6010]	Notfälle in der größeren Umgebung anzeigen	✓	-	
[TB6020]	Aktuelle Notfälle in einer Liste anzeigen	✓	-	
[TB6030]	Auf einen Notfall klicken, um die Details anzusehen	✓	-	
[TE2030]	Position des Notfalls in der Kartenansicht überprüfen	✓	-	
[TE6020]	Kartenansicht für mehrere Notfälle überprüfen	✓	-	

(Hinweis: Zwei Testfälle wurden entfernt, weil die entsprechenden Funktionalitäten nicht umgesetzt wurden)

[TS06] – Experten und Qualifikationen

Ein Benutzer trägt „Erste-Hilfe“ als besondere Qualifikation in seinem Profil ein. Bei einem späteren Notfall gibt er an, zu helfen, wodurch andere Informierte sehen können, dass ein Helfer mit medizinischer Kenntnis bereits unterwegs ist.

Test-ID	Testbeschreibung	Client	Server	Kommentar
[TE4010]	Experten-Qualifikationen eingeben	✓	-	
[TE4020]	Experten-Qualifikationen ändern	✓	-	
[TB2040]	Notfall-Details anzeigen lassen	✓	✓	
[TE2010]	Anzahl der helfenden Experten überprüfen	✓	✓	
[TE2020]	Qualifikationen der helfende Experten überprüfen	✓	✓	

[TS07] – Verwendung ohne Internet

Ein Benutzer versucht die Applikation zu benutzen, hat aber leider keine Internetverbindung. Da er sich aber mit der App vertraut machen will, hält ihn das nicht auf und er testet durch, was er alles ohne Verbindung machen kann und was es alles gibt.

Test-ID	Testbeschreibung	Client	Server	Kommentar
[TB9010]	Applikation starten – ohne Internet	✓	-	
[TB9020]	Notfall melden – ohne Internet	✓	-	
[TB9030]	Gemeldeten Notfall spezifizieren – ohne Internet	✓	-	
[TB9040]	Notfall beenden – ohne Internet	✓	-	
[TB9050]	LiveFeed laden – ohne Internet	✓	-	
[TB9060]	Notfalldetails öffnen – ohne Internet	✓	-	
[TB9070]	Bei einem Notfall helfen – ohne Internet	✓	-	

[TB9080]	Notfall ignorieren – ohne Internet	✓	-	
[TB9090]	Notfall als Missbrauch Melden – ohne Internet	✓	-	
[TB9100]	DetailsPage verlassen – ohne Internet	✓	-	
[TB9110]	Link hinzufügen – ohne Internet	✓	-	
[TB9120]	Link ändern – ohne Internet	✓	-	
[TB9130]	Link löschen – ohne Internet	✓	-	

[TS08] – Verwendung nach Beenden/Entfernen eines Notfalls

Ein Benutzer nimmt die Applikation sehr ernst und versucht möglichst aktiv zu sein. Leider hat er mit Verbindungsproblemen zu kämpfen und schaltet sein Handy sehr häufig aus, sodass er die Benachrichtigungen über das Ende eines Notfalls öfters verpasst und viel mit veralteten Daten interagiert.

Test-ID	Testbeschreibung	Client	Server	Kommentar
[TB9210]	Gemeldeter Notfall von sich selbst beendet	✓	-	
[TB9220]	Dialog eines beendeten Notfalls folgen (als Involved)	✓	-	
[TB9230]	Dialog eines beendeten Notfalls folgen (als Helper)	✓	-	
[TB9240]	Dialog eines beendeten Notfalls ignorieren (als Involved)	✓	-	
[TB9250]	Dialog eines beendeten Notfalls ignorieren (als Helper)	✓	-	
[TB9260]	Zur DetailsPage eines beendeten Notfall über Push navigieren	✓	-	
[TB9270]	Zur DetailsPage eines beendeten Notfall über Dialog navigieren	✓	-	
[TB9280]	Zur DetailsPage eines beendeten Notfall über LiveFeed navigieren	✓	-	
[TB9290]	DetailsPage eines beendeten Notfalls verlassen	✓	-	
[TB9300]	Bei einem beendeten Notfall helfen	✓	✓	
[TB9310]	Beendeter Notfall ignorieren	✓	-	
[TB9320]	Beendeter Notfall als Missbrauch melden	✓	-	

5. Verbleibende Bugs

Nummer	Beschreibung	PRIORITÄT
[B003]	Einige Positionsupdates des Benutzers führen zu einem Datenbankkonflikt im Server.	<i>MITTEL</i>
[B009]	Ignorierte Notfälle werden nach Neustart der App resettet	<i>NIEDRIG</i>
[B010]	Fehlermeldungen werden nicht übersetzt	<i>NIEDRIG</i>
[B012]	Push-Benachrichtigungen werden nicht übersetzt	<i>NIEDRIG</i>
[B019]	Apostroph in den Spezifikationen wird in Push-Benachrichtigungen durch ein falsches Zeichen ersetzt.	<i>NIEDRIG</i>
[B026]	Navigieren zu DetailsPage eines nicht-existierenden Notfalls von der gepufferten Liste der LiveFeed (also vom Server vollständig gelöscht)	<i>NIEDRIG</i>
[B034]	Die LocalSettings können überlaufen (zu viele Links)	<i>MITTEL</i>
[B035]	Textboxen/StringInput ist ohne Limit möglich	<i>NIEDRIG</i>
[B037]	In manchem Sprachen fehlen einzelne Strings	<i>NIEDRIG</i>
[B039]	Gelöschte Links bleiben lokal auf den Geräten bestehen und können nicht gelöscht werden (Exception bei RemoveLinkAsync)	<i>NIEDRIG</i>
[B041]	Navigation zur DetailsPage mittels Push während die App im Suspend mit einer DetailsPage offen ist ist fehlerhaft	<i>MITTEL</i>

6. Anhang

6.1. Statistiken

Code Metrics

Metric:	Without Test classes	With Test classes
Classes/Interfaces:	103	129
Lines of Code:	13.356	21.222
Statements:	5.579	8.675
Percent Comment Lines:	4,2 %	6,1 %
Percent Documentation Lines:	28,1 %	23,9 %
Methods per Class:	7,50	9,69
Calls per Method:	2,13	2,59
Most complex class:	EmergencyContainer (13)	EmergencyContainer (13)
Longest class:	UIManager (676)	ServerControllerTests (1.153)

Ergebnisse der Phase

Elemente:	Anzahl:
Unit Test Klassen:	6
Unit Test Methoden:	63
Integration Test Klassen:	4
Integration Test Methoden:	176
Mock Klassen:	8
In der Phase gefixte Bugs:	30
Übrig gebliebene Bugs:	11

6.2. Glossar

AKTIVE PERSON <i>Active User</i>	Person, die die Applikation momentan nicht im Ruhemodus hat.
ALARM <i>Alarm</i>	Funktion eines Notfalls, die Personen benachrichtigt und auf dem LiveFeed erscheint.
BENACHRICHTIGUNG <i>Notification</i>	Statusleistennachricht/Notification durch die Applikation.
BENUTZER <i>User</i>	Person, die die Applikation geöffnet hat und momentan verwendet.
CLIENT <i>Client</i>	Gerät des Benutzers, auf dem die Applikation installiert ist.
DETAILS <i>Details</i>	Alle Eigenschaften eines Notfalls, die ein Informierter einsehen kann. Gebündelt verfügbar in der Detailansicht eines Notfalls.
EXPERTE <i>Expert</i>	Person, die in ihrem Profil besondere Qualifikationen angegeben hat.
GRÖßERE UMGEBUNG <i>Larger Area</i>	Alle räumlichen Punkte in einem festgelegten, größeren Radius.
HELFENDE <i>Helper</i>	Informierte, die in der Applikation ausgewählt haben, dass sie auf dem Weg sind.
INVOLVIERTE <i>Involved User</i>	Melder oder benachrichtigte Personen, die den Notfall nicht ignoriert haben.
LIVEFEED <i>LiveFeed</i>	Karte/Liste mit allen aktuellen Notfällen
MELDER <i>Reporter</i>	Person, die den Notfall als Erste gemeldet hat.
NÄHERE UMGEBUNG <i>Nearby Area</i>	Alle räumlichen Punkte in einem festgelegten, kleineren Radius.
NOTFALL <i>Emergency</i>	Vorfall, bei dem Menschen, Tiere oder Eigentum ohne menschliches Eingreifen Schaden nehmen können.
PERSON <i>Person</i>	Mensch, der die Applikation auf seinem Smartphone installiert hat.
POP-UP <i>Pop-Up</i>	Dialog auf der grafischen Benutzeroberfläche, der sich (teilweise) über einen anderen Bildschirm legt und Informationen darstellt. Kann geschlossen werden.
PROFIL <i>Profile</i>	Auswahlbildschirm für gerätespezifische Informationen, die weitere Details über den Benutzer preisgeben.
RUHEMODUS <i>Do Not Disturb Mode</i>	Modus der Applikation, in der alle Benachrichtigungen und Standortdaten deaktiviert sind.
SPEZIFIKATION <i>Specification</i>	Die Angabe und Änderung der Details eines Notfalls durch den Melder.
VERKNÜPFT PERSONEN <i>Linked Person</i>	Personen, die freiwillig mit dem Gerät des Benutzers verknüpft sind, um priorisiert behandelt zu werden.

6.3. UI Testprotokolle

Hinweis: Diese Protokolle entstanden im Laufe der Phase. Die meisten präsentierten Mängel wurden dabei schon behoben, weswegen die Ergebnisse nicht dem aktuellen Stand entsprechen.

Geschlecht: weiblich	Datum: 22.02.2015
Alter: 21	
Beruf: Student, Informatik	
Betriebssystem des eigenen Smartphones: Android	

Auszuführende Aktion	Ergebnis	Kommentar
App starten	✓	
AGB akzeptieren	✓	
Notfall auslösen	(✓)	Während dem Runter halten nach unten gezogen, Editor wurden angezeigt, dann aber wieder zurück bewegt und ausgelöst.
Details editieren	✓	
Positionsverfolgung deaktivieren	✓	
Alarm beenden	✓	
Profil Name festlegen	✓	
Verknüpfte Person hinzufügen	X	(nicht gefunden) -> nach richtiger Navigation aber richtig hinzugefügt
Ruhemodus aktivieren	✓	
Notfälle aus der Umgebung anzeigen	(✓)	Hat lang gedauert. Viele Fehlritte.
Notification erhalten	✓	
Bei diesem Notfall helfen	✓	
Notfall über Navigation im Notfall Radar ignorieren	✓	
Live Tile zum Melden eines Alarms aktivieren	(✓)	Dachte da komm ich ganz normal in die App => Alarm über Live Tile ausgelöst.
Verknüpfte Person umbenennen	✓	Aber Anmerkung: Unten bleibt das Plus zum Hinzufügen, das sollte man ersetzen durch ein Häkchen zum Speichern der Änderung.
Sicherheitsfrage festlegen	✓	
Qualifikation Erste-Hilfe hinzufügen	✓	
Verknüpfte Person löschen	(✓)	Mülltonne und X sind nicht eindeutig! Nur Mülltonne und Änderungen über Häkchen.
Eigenen Profilcode per SMS teilen	✓	
Zeitgesteuerten Ruhemodus von 12:00 bis 16:00 Uhr aktivieren	✓	
Notfall Auslösung abbrechen	✓	
Notfall vor dem Auslösend des Alarms spezifizieren und Links nicht benachrichtigen	✓	Aber: zuerst neben den Button runtergewischt
Missbrauch bei einem Notfall melden	✓	

Geschlecht: männlich

Datum: 22.02.2015

Alter: 22

Beruf: Student, Informatik

Betriebssystem des eigenen Smartphones: Android

Auszuführende Aktion	Ergebnis	Kommentar
App starten	✓	
AGB akzeptieren	✓	
Notfall auslösen	✓	
Details editieren	✓	
Positionsverfolgung deaktivieren	✓	
Alarm beenden	✓	
Profil Name festlegen	✓	
Verknüpfte Person hinzufügen	(✓)	Falsch navigiert -> im Prinzip nur gefunden weil letzte Option
Ruhemodus aktivieren	✓	
Notfälle aus der Umgebung anzeigen	X	Nicht gefunden. Emergency Radar sieht aus wie Überschrift.
Notification erhalten	✓	
Bei diesem Notfall helfen	✓	
Notfall über Navigation im Notfall Radar ignorieren	✓	Wäre gut wenn man auf der Karte zur Notfällen navigieren kann.
Live Tile zum Melden eines Alarms aktivieren	✓	
Verknüpfte Person umbenennen	✓	
Sicherheitsfrage festlegen	✓	
Qualifikation Erste-Hilfe hinzufügen	✓	
Verknüpfte Person löschen	(✓)	Mülltonne und X sind nicht eindeutig! Nur Mülltonne und Änderungen über Häkchen!
Eigenen Profilcode per SMS teilen	(✓)	Sehr lange gedauert.
Zeitgesteuerten Ruhemodus von 12:00 bis 16:00 Uhr aktivieren	✓	
Notfall Auslösung abbrechen	✓	Texte größer schreiben
Notfall vor dem Auslösend des Alarms spezifizieren und Links nicht benachrichtigen	✓	
Missbrauch bei einem Notfall melden	✓	

Geschlecht: männlich

Datum: 24.02.2015

Alter: 23

Beruf: Student, Filmregie

Betriebssystem des eigenen Smartphones: iOS

Auszuführende Aktion	Ergebnis	Kommentar
App starten	✓	
AGB akzeptieren	✓	
Notfall auslösen	✓	
Details editieren	✓	
Positionsverfolgung deaktivieren	✓	
Alarm beenden	✓	
Profil Name festlegen	(✓)	Erst ins Settings, dann Profil gefunden, will Return
Verknüpfte Person hinzufügen	✓	
Ruhemodus aktivieren	X	Do Not Disturb statt Ruhemodus, Deactive Notification nicht verstanden
Notfälle aus der Umgebung anzeigen	X	Nicht gefunden. Emergency Radar sieht aus wie Überschrift, Will aber eine persönlichere Überschrift
Notification erhalten	✓	
Bei diesem Notfall helfen	✓	
Notfall über Navigation im Notfall Radar ignorieren	✓	
Live Tile zum Melden eines Alarms aktivieren	(✓)	Dachte es geht zur App, nicht, dass es ausgelöst hat. Nicht klar, dass Live Tile direkt den Alarm auslöst.
Verknüpfte Person umbenennen	✓	
Sicherheitsfrage festlegen	✓	
Qualifikation Erste-Hilfe hinzufügen	✓	
Verknüpfte Person löschen	✓	
Eigenen Profilcode per SMS teilen	✓	
Zeitgesteuerten Ruhemodus von 12:00 bis 16:00 Uhr aktivieren	✓	
Notfall Auslösung abbrechen	✓	
Notfall vor dem Auslösend des Alarms spezifizieren und Links nicht benachrichtigen	✓	
Missbrauch bei einem Notfall melden	✓	