Università
Ca'Foscari
Venezia

INGEGNERIA DEL SOFTWARE

Piano di Testing

BeeSafe

Buffer Overflow

SERANI Hernest 877028

BALDASSO Enrico 874885

CAPPON Stefano 876895

Indice

1. Introduzione

1.1 - Fine

Il documento di piano di Testing serve per definire:

- Lo scopo dei test, le aree dove concentrarsi e gli obiettivi
- Le responsabilità legate ai test
- Le strategie di test legate ai modelli e ai tipi utilizzati
- Criteri di entrata ed uscita
- Le basi delle stime dei test
- Rischi, problemi, assunzioni e test delle dipendenze
- Il programma dei test con le relative tappe principali
- I risultati dei test

1.2 - Scopo

Il documento di piano di Testing ha lo scopo di definire le linee guida e una vista globale di ciò che riguarda test e parametri di testing, del progetto BeeSafe. Indicativamente:

- Definire cosa verrà testato
- Definire i parametri sul come eseguire i test
- Risorse necessarie e quando(ex. database realtime)

2. Riepilogo dei Test

2.1 - Scopo dei Test

Lo scopo dei test è quello di individuare possibili malfunzionamenti dell'applicazione. Noi del team creeremo dei test per sondare le caratteristiche globali del sistema adottando diverse strategie in base alla fase del processo di testing.

3. Analisi dell’Ambito e Definizione Area di Test

3.1 - Rilascio Contenuto

Si fa riferimento al documento di analisi dei requisiti.

3.2 - Test di Regressione

Quando verranno effettuate modifiche per l’applicazione BeeSafe e quindi verrà rilasciata una nuova versione dell’app, verranno effettuati test sulle nuove funzionalità, ma soprattutto andremo a verificare che le funzionalità preesistenti abbiano mantenuto le loro caratteristiche qualitative dopo l’introduzione di queste ultime

3.3 - Test della Piattaforma

Per i test relativi all’applicazione verrà richiesto uno smartphone con sistema operativo Android(Versione minima Oreo 8.0). Lo smartphone deve essere dotato di tecnologia bluetooth, servizio GPS e deve essere possibile connettersi ad internet.

Per i test relativi al funzionamento del real-time database viene richiesto l’utilizzo del medesimo smartphone che tramite l’applicazione effettuerà una comunicazione con il db.

4. Obiettivi del test di Progressione

Ref	Funzione	Obiettivo del test	Criterio di valutazione	X-Ref	P
R1	Primo accesso	Dare il permesso all'applicazione di utilizzare il bluetooth	Positivo se l'utente acconsente l'utilizzo del bluetooth e quindi verrà salvato il permesso	R2	4
R2	Primo accesso	Dare il permesso all'applicazione di utilizzare il GPS	Positivo se l'utente acconsente l'utilizzo del GPS e quindi verrà salvato il permesso	R1,R3	4
R3	Visualizzazione mappa	Visualizzare la mappa circostante all'utente	Positiva se l'applicazione tramite GPS riesce a centrare la posizione esatta dell'utente e visualizzare la mappa con un'approssimazione degli affollamenti	R2	3
R4	Scansione dei dispositivi Bluetooth	Fare una scansione dei dispositivi vicini in background ogni tot minuti	Positivo se il thread in background riesce a fare una scansione ogni tot minuti	R1	4
R5	Correttezza dell'algoritmo di tracing	Vedere se l'algoritmo riesce ad approssimare l'affollamento di un posto	Positivo nel caso in cui l'algoritmo riesce a ritornare il numero dei dispositivi nelle vicinanze e fare un approssimazione se un certo luogo è affollato	R4	4
R6	Caricamento dei dati	Caricare i dati riguardo a un luogo	Positivo nel caso in cui l'applicazione riesce a caricare nel realtime database il luogo e le rispettive informazioni	-	4
R7	Aggiunta luogo ai preferiti	Poter aggiungere un luogo ai preferiti	Positiva se l'utente effettuerà quest'azione: tap luogo della mappa->tap aggiungi ai preferiti. Una volta aggiunto il luogo deve essere possibile ricevere le notifiche	R3	3
R8	Rimozione luogo dai preferiti	Poter rimuovere un luogo dai preferiti	Positiva se l'utente effettuerà quest'azione: tap sul luogo della mappa(luogo presente tra i preferiti)->tap sull'icona a forma di X. Una volta rimosso il luogo l'utente non riceverà più le notifiche	R3, R7	3
R9	Notifiche per i luoghi preferiti	Ricevere notifiche riguardanti i luoghi preferiti	Positiva se una volta aggiunto un luogo tra i preferiti, in caso di affollamenti, verrà inviata una notifica all'utente	R6, R7	1
R10	Stima futura di un luogo	Vedere un'approssimazione, basata su predizione, del numero di persone presenti su un luogo	Positiva se per un luogo del quale si hanno sufficienti dati dalle settimane precedenti verrà effettuata una previsione approssimativa del numero di persone presenti in quel luogo in una certa fascia oraria	R3, R6	2

5. Altri Test

5.1 - Stress & Volume Test(S&V)

Verificare comportamento app :

- Con invio dati per molti luoghi preferiti, possibile decisione di un limite di preferiti settabili
- Con invio di dati per aree molto estese
- Con invio di dati per molte aree
- Chiudere e riaprire l'app diverse volte velocemente
- Molte richieste di scansioni

5.2 - Test Connettività (CT)

Verificare connettività con il database.

5.3 - Ripristino di Emergenza/Backup

Controllare che i dati vengano mantenuti nel database in vista del loro riutilizzo per avere la possibilità di prevedere l'affollamento in determinati luoghi, ad esempio preferiti.

5.4 - Unit Test

Test riguardanti Firebase, ovvero spostamento, selezione e cancellazione dei dati.

6. Strategia di Test

6.1 - Responsabilità del livello di Test

Livello di test	Parte esterna	Team di progetto	Business
Stress & Volume Test	S	P	
Test di connettività		P	
Ripristino di Emergenza/Backup		P	
Unit Test		P	P
Test di accettazione dell'utente		S	P
Test di verifica della produzione		S	P

6.2 - Tipo di Test e Approccio

- **Incremental Testing** - Procederemo con i test dell'applicazione man mano che le funzionalità verranno implementate. I test verranno effettuati sia sulle nuove funzionalità che sulle funzionalità già esistenti.

- **Black-box testing** - Al completamento dell'implementazione dell'applicazione verranno effettuati dei test di tipo "scatola nera", ovvero utilizzando degli input definiti e confrontando i risultati ottenuti con quelli previsti.

6.3 - pianificazione dell'Esecuzione dei Test

La seguente tabella fa riferimento alla tabella definita negli obiettivi di testing (Tabella - Punto 4). Ad ogni testing si associa un membro del team. Il gestore di testing avrà la responsabilità di monitorare ed elaborare gli esiti.
L'esito sarà Positivo(P) oppure Negativo (N). Ad ogni testing si ha la possibilità di aggiungere dei consigli o miglioramenti con dei feedback.
I membri sono da stabilire.

Ref	Membro	Esito	Consigli/Miglioramenti
R1			
R2			
R3			
R4			
R5			
R6			
R7			
R8			
R9			
R10			

6.4 - Strutture, Dati e Piano di Gestione Risorse

6.4.1 - Environment

- Android Oreo versione 8.0
- Firebase Android BoM 26.0.0
- In-App Messaging 19.1.2
- buildToolsVersion "30.0.1"
- Bluetooth library : 'com.clj.fastble:FastBleLib:2.3.4'
- androidx
- MAPS : 'com.google.android.gms:play-services-maps:17.0.0'
- MAPS : 'com.google.maps.android:android-maps-utils:2.0.3'
- HEATMAPS:'com.google.maps.android:android-maps-utils:1.0.2'
- GEOHASHING : 'ch.hsr:geohash:1.4.0'
- FIREBASE : 'com.google.firebase:firebase-database:19.5.0'
- GSON : 'com.google.code.gson:gson:2.8.6'
- org.jetbrains:annotations-java5:15.0
- junit:junit:4.12
- com.google.firebase:firebase-messaging:21.0.0
- com.google.firebase:firebase-analytics:18.0.0

6.4.2 - Requisiti dell'ambiente

- Connessione Internet per interagire con i servizi FireBase e di Google Maps
- Dispositivi dotati di Bluetooth e GPS
- Accesso al real-time database
- Browser per usare il GUI di FireBase e le cloud-machine Functions
- Android Studio
- GitHub per la condivisione dei release e codice

6.4.3 - Risorse e Requisiti Conoscitivi

- Conoscenza di Java, Typescript, File XML
- Conoscenza del lifecycle delle Activity nei Sistemi Operativi Android e gli servizi background
- Sapere usare l'API di Google Maps e FireBase

6.5 - Strumenti di Test

Processo	Strumento
Test dell'applicazione	Smartphone con sistema operativo minimo Android 8.0
Test connettività	Database su server Firebase + Smartphone
Test di Backup	Database su Firebase
Unit Test	Server Firebase

6.6 - Procedura di Consegna del Test

Report strutturato come segue:

- Sommario e dettaglio dei risultati di esecuzione delle sessioni di test.
- Risultati dei test (superati, falliti) per requisito non funzionale.
- Statistiche risultati test (superati, falliti).
- Grafico e lista dei difetti per loro priorità e stato.

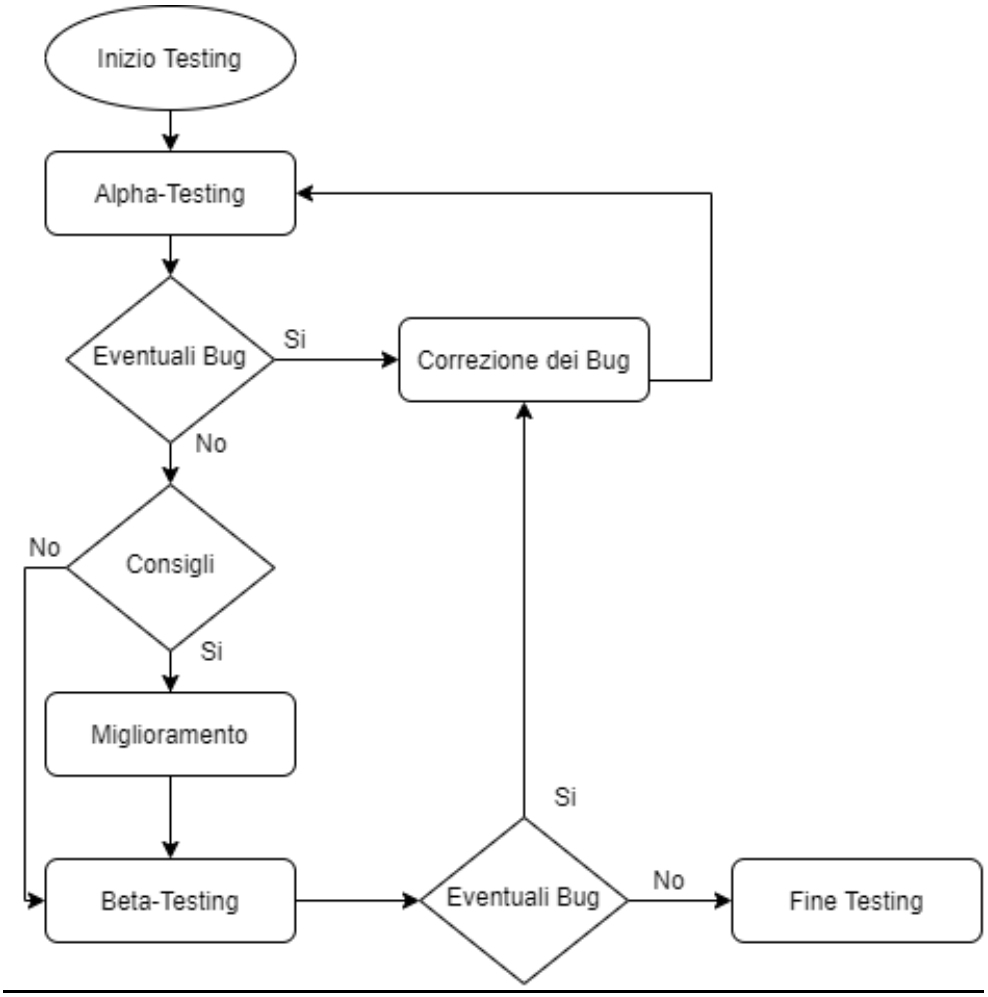
6.7 - Metriche di Test

Durante l'esecuzione, vengono misurate delle metriche relative alle caratteristiche di performance dell'applicazione oggetto di test:

- Informazioni sull'efficienza del real-time database
- Informazioni sulla piattaforma sulla quale vengono eseguiti i test
- Informazioni di performance relative all'applicazione
- Informazioni API utilizzate
- Feedback riguardo la User Experience (UX) e usabilità dell'interfaccia grafica.
- Informazioni sullo spreco delle risorse, sia del dispositivo Android che effettua i test, che le risorse del cloud-machine di FireBase

7. Piano Ambiente di Test

7.1 - Diagramma dell’ambiente di Testing



Per avere più sicurezza nei test dell'applicazione, nel caso di eventuali bug anche nei prossimi beta-release, si effettuerà il testing dall’inizio, perché una feature aggiunta o la risoluzione di bug nella seconda fase può creare bug risolti prima.

I passaggi del flow chart per il testing sono:

- Alpha Testing - L'implementazione base dell’applicazione, gestione dei permessi, l’interfaccia utente dove si mostra una mappa all’utente, la sua posizione e i posti affollati usando dei dati statici presenti nel database. La creazione di un semplice algoritmo di tracing per i posti affollati.

- Primo inter-frame - Analisi dei risultati ottenuti.
- Consigli - Dopo aver risolto gli eventuali bug, otteniamo dai tester dei consigli riguardo l'usabilità dell'applicazione e le feature consigliate da aggiungere nel prossimo testing.
- Miglioramento - Valutare i consigli dati dai testers, miglioramento interfaccia grafica e prestazioni. Migliorare l'algoritmo di tracing, basato sui test fatti e ottenuti.
- Beta Testing - Creare la possibilità di prevedere l'affollamento di un posto basandosi sui dati presenti nel database. In questa fase è importante tenere a mente i consigli dati dai tester.
- Terzo inter-frame - Testare le feature aggiunte e l'accuratezza degli algoritmi sviluppati sia per tracing che per la previsione dei posti affollati.

7.2 - Dettagli dell'Ambiente di Test

I testers saranno delle persone, alle quali forniremo l'APK dell'applicazione. Saranno circa 20 persone includente il cerchio di amicizia e familiari. Per effettuare il testing dell'applicazione serve solo l'APK, devono installarla, accettare i permessi per il GPS e Bluetooth e andare in giro nelle seguenti città: Mestre (Italia), Treviso (Italia), Padova(Italia), Tirana (Albania).

Il dispositivo Android che devono avere, deve essere dotato di Bluetooth, GPS, e sistema operativo Android maggiore o uguale a Android Oreo. Si deve avere accesso ad Internet durante tutto il testing. La memoria di archiviazione richiesta per usare l'applicazione deve essere almeno 30 Mb. L'accesso al real-time database, Google Maps e gli altri servizi di FireBase sara' fornito da noi sviluppatori usando un API Key di debugging.

L'interfaccia esterna utilizzata durante il testing sarà la console di Firebase, che tramite la sua GUI la useremo per investigare i dati nel database real-time, che mostra una struttura JSON del database.

7.3 - Stabilire l’ambiente

Il piano per stabilire l’ambiente di testing e le rispettive responsabilità sarà completato in futuro. Da stabilire in base agli impegni di ciascun membro del team rispetto al tempo libero, però alcuni task importanti da fare sono listati.

Task	Requisiti	Responsabilità	Data Inizio	Data Fine
Fornire un APK di installazione funzionante ai testers	La compilazione del codice e il funzionamento dei servizi offerti da FireBase			
Assicurare dai testers la loro partecipazione e verificare i parametri dei loro dispositivi	Avere già in anticipo una lista dei testers possibili			
Essere aggiornato sulle nuove norme di misura COVID nelle città dove si vuole effettuare il testing	-			

7.4 - Controllo dell’ambiente

L’ambiente di testing sara’ controllato frequentemente, investigando e tenendo in considerazione i feedback ricevuti dai testers. Nel caso di un errore di programmazione, lanciando delle eccezioni dall’applicazione, i testers devono fornirci uno screenshot delle eccezioni lanciate in runtime. In compile time non devono essere presenti errori, come già descritto nello stabilimento dell'ambiente di testing, che sarà un requisito per fornire l’APK funzionale in fase di installazione. Durante il testing si deve monitorare anche il uptime dei servizi FireBase e usando i suoi strumenti monitoriamo i log nel caso di un errore, oppure controllare che i dati siano corretti quando non si verificano errori. L’ultima cosa che si deve monitorare è lo spreco delle risorse delle macchine cloud di FireBase, perché un errore di programmazione può portare a un loop e quindi consumare le risorse, come lo spazio di archiviazione del real-time database, la CPU del cloud-machine quando si usa FireBase per alcuni servizi di backend per la parte del server.

7.5 - Ruoli e Responsabilità dell’ambiente

Ruolo	Membro	Responsabilità
Gestione del progetto e dei release	Hernest Serani	Responsabile per il coordinamento dei lavori, schedulazione, il controllo e la distribuzione del software.
Monitoraggio delle funzionalità backend del Server	Enrico Baldasso	Responsabile per il monitoraggio del funzionamento dei servizi backend e investigazione dei dati presenti nel db.
Gestione dei Testing	Stefano Cappon	Responsabile per avvisare il gestore di release per i problemi/bug/miglioramenti durante il testing.

8. Presupposti e Dipendenze

8.1 - Presupposti

- Il presupposto più importante durante la fase di testing dell'applicazione sarà’ quello che tutte le persone nelle vicinanze abbiano il Bluetooth acceso, ed a ciascuna persona e’ associato un unico dispositivo Bluetooth.
- L’uptime dei servizi FireBase e di Google Maps deve essere una percentuale grande. Nei siti rispettivi si indica che questa percentuale e’ 99.95% che e’ una percentuale abbastanza buona e indica che è molto improbabile che durante it testing possiamo avere delle interruzioni dalla parte del Server

8.2 - Dipendenze

L’unica configurazione prevista sarà quella dei servizi back-end del Server. Forniamo ai testers un APK configurato per il testing per rispettare i seguenti accessi:

- Per dare accesso ai servizi di FireBase (Realtime database, cloud-machine functions) forniamo un API Key di debugging, perche’ come gia’ descritto in precedenza, lo spreco delle risorse durante il testing può essere elevato.
- Con la stessa logica useremmo un API Key per debugging anche per le API di Google Maps.

9. Criteri di Ingresso ed Uscita

Dettagli sui criteri di inizio e terminazione delle fasi di testing, raccolta degli obbiettivi principali da esaurire per considerare la fase completata

- **Alpha-testing**: L'alpha-testing inizierà concorrentemente alla fase di sviluppo, finirà quando le seguenti funzionalità saranno state testate:
    - Primo Accesso e Autorizzazioni
    - Visualizzazione della mappa
    - Scansione in background tramite Bluetooth
    - Implementazione dell'algoritmo di tracing
    - Caricamento ai dati sul realtime database
    - Aggiungere o rimuovere un luogo dai preferiti
    - Notifiche per i luoghi preferiti
- **Beta-Testing**: Il beta-testing inizierà una volta finito l'alpha-testing e aver valutato i suoi risultati. Inoltre dopo aver preso in considerazione i suggerimenti dei tester per il miglioramento delle funzionalità. Il Beta-Testing finirà dopo aver risolto gli eventuali bug e implementato le seguente features:
    - Previsioni di un posto affollato

Nel caso in cui siano presenti dei bug dopo aver fatto il Beta-Testing, si ritornerà all'Alpha-Testing, perché possono ripresentarsi dei bug già risolti in precedenza.

10. Piano Amministrativo

10.1 - Responsabilità

Task	Persona Responsabile	Approvata
Systems Integration	Hernest Serani	
User Acceptance Testing	Enrico Baldasso	
Production Verification Testing	Stefano Cappon	

10.2 - Milestones e Programma del Test

Sono state identificate le seguenti milestones per il programma di test, si noti che alpha-test e beta-test hanno 2 cicli dato che potrebbero essere ripetuti più volte(Se saranno necessari altri cicli verranno aggiunti in seguito).

Milestone	Data fine programmata	Data fine effettiva
Pianificazione dei test	10/11/2020	
Piano di test completo	19/11/2020	
Esecuzione dei test		
Alpha-test ciclo 1	01/12/2020	
Alpha-test ciclo 2		
Beta-test ciclo 1	10/12/2020	
Beta-test ciclo 2		
Fine esecuzione dei test	15/01/2020	

10.3 - Training

Requisito di Training	Staff	Data
Aggiornamento sulla situazione attuale della pandemia COVID-19		
Essere aggiornato sui servizi FireBase, e la disponibilità del server		

10.4 - Gestione dei Difetti

I principali difetti per questo progetto possono essere rilevati nel momento in cui viene eseguita una scansione:

- *Problema*: una scansione può portare risultati ambigui per quanto riguarda i dispositivi rilevati perché, oltre agli smartphone, la scansione potrebbe rilevare il segnale bluetooth di televisioni, PC, smartwatch e qualsiasi dispositivo dotato di bluetooth. Quindi la scansione potrebbe rilevare un numero maggiore di dispositivi rispetto alle persone realmente presenti in una certa zona.
- *Soluzione*: Per risolvere questo problema è stata ideata una blacklist in fase di programmazione, la quale è strutturata in modo tale da tenere conto solo dei dispositivi smartphone(assumendo che una persone ne abbia al massimo uno) per le scansioni.

Un altro difetto si potrebbe riscontrare con l'algoritmo di previsione di affollamento, se si hanno pochi dati relativi ad un luogo sarà più difficile fare delle previsioni sull'affollamento nelle diverse fasce orarie.

11. Definizioni

Fare riferimento ai documenti Piano di Progetto e Analisi Specifica
Nuove parole introdotte in questo documento:

API Key - Un identificatore che permette di usare i servizi di un API.

APK - formato di file che e si usa per la distribuzione e l'installazione di componenti sulla piattaforma di dispositivi Android.

Backend - In architettura client-server, back end definisce il livello di accesso ai dati che stanno nel server, la loro manipolazione e la logica dell'applicazione.

BlackList - è una lista utilizzata per controllare gli accessi a una certa risorsa.

Cloud Machine - una paradigma di erogazione di servizi offerti su richiesta da un fornitore a un cliente finale attraverso la rete internet, configurabili e disponibili in remoto sotto forma di architettura distribuita.

Cloud Functions - Servizio di Google Cloud machine Computing che permette di eseguire codice JS/TypeScript dentro le macchine cloud e di definire la logica dell'applicazione.

COVID - CO sta per Corona, VI sta per Virus, infine D sta per disease (malattia) è il tipo di virus che ha interessato il mondo tra il 2019 e il 2020.

CPU - Central Processing Unit, unità logica e fisica che gestisce l'elaborazione di processi e le funzionalità base di un computer

Debugging - Processo basato su analisi e testing che portano a una soluzione di problemi lato software.

Log di Errore - notifica di errore a runtime o compile time, contiene il tipo d'errore la fonte e la relativa propagazione.

UX - User Experience, è un termine utilizzato per definire la relazione tra una persona e un prodotto, un servizio, un sistema.

Thread - è una suddivisione di un processo in due o più filoni (istanze) o sottoprocessi che vengono eseguiti concorrentemente da un elaboratore.

XML - è un metalinguaggio per la definizione di linguaggi di markup, ovvero un linguaggio marcatore basato su un meccanismo sintattico che consente di definire e controllare il significato degli elementi contenuti in un documento o in un testo.

12. Riferimenti

Per la creazione di questo documento si sono utilizzati i seguenti riferimenti:

Nome Documento	Commento
Piano di Progetto	Documento interno al progetto
Requisiti	Documento interno al progetto
Test Plan Template	Template piano di testing
https://firebase.google.com/docs/android/setup	Firebase Android Documentation
Test Strategy and Environment di "Melodic big data cloud"	Info testing in particolare hardware