Programmazione Dispositivi Mobili

Università degli Studi di Torino

Dipartimento di Informatica

Relazione NewsVoF

Anno Accademico 2017/2018

Studenti:
Matteo Marsala
Francesco Ferraris

1 Introduzione:

Da svariati anni, il web è afflitto da un fenomeno spaventoso: le **Fake News**.

Gli studi sperimentali nell'ambito delle bufale online sono numerosi, ma sono presentati agli utenti con eventi organizzati, dentro laboratori in cui le persone forniscono i propri dati al fine di uno studio campionato. Da qui nasce l'idea: creare un'applicazione apposita che permetta di acquisire dati da studiare e allo stesso tempo far sì che l'utente si diverta e sia invogliato a usare l'applicazione, nonchè ad informarsi sulle notizie che popolano il web.

I dati che si riceveranno potranno poi essere usati per costruire vari esperimenti, su argomenti noti come la *Social Influence* e di come sia capace di generare un pensiero su un fatto di cui non si abbia una conoscenza completa, oppure argomenti come il *Backfire Effect*, che non ci permette di cambiare le nostre convizioni, anche se errate.
Tali esperimenti potranno poi essere utilizzati in futuro per un obbiettivo sociale più ampio, come la rimozione delle bufale dal web.

Ovviamente, oltre al fine principale dell'applicazione e a quello di rispettare le specifiche accademiche, si vuole costruire qualcosa che abbia come scopo finale quello di presentare un'app interessante, utile, divertente e che non occupi spazio inutile sulla memoria del proprio device.

2 Specifiche generali dell'applicazione:

L'applicazione avrà come funzionalità principale quella di permettere agli utenti di valutare le notizie da noi manipolate tramite i **lettori RSS**. Per far questo, bisognerà effettuare il log-in.

Nel caso in cui non si voglia creare il proprio account, si potranno comunque leggere le notizie ma, in ogni caso, non si potranno valutare.

Saranno presenti anche altre funzionalità, non completamente utili per il nostro scopo, ma formative dal punto di vista della **Programmazione Mobile**. Sarà quindi possibile cambiare i propri dati del profilo, accedere alla classifica globale, contattare con una mail i creatori dell'applicazione, leggere

una guida introduttiva e impostare le opzioni preferenziali dell'applicazione. Gli amministratori potranno anche accedere ad una pagina privata che riassume i dati degli utenti che hanno votato una determinata notizia, ricercare utenti o notizie a seconda di una parola chiave o di una caratteristica, e così via.

2.1 Situazione del PlayStore:

Attualmente sul Play Store non esiste nulla di simile, in quanto NewsVoF è un'applicazione di nicchia usata per gli studi sperimentali in ambito accademico, non commerciale.

Tuttavia, è presente un'applicazione (chiamata **Fighting Fake News**) che mostra liste di bufale dando la possibilità agli utenti di affermare di aver letto tale notizia. Oltre a ciò, esiste anche **Hoaxy**, un motore di ricerca per le fake news.

Inoltre, esisteva un sito creato per lo stesso fine, ma non collegato ad applicazioni Android. Per questo motivo abbiamo deciso di ampliare i canali con cui raggiungere l'utente, in modo da aver una maggior mole di dati su cui lavorare.

2.2 Benefici dell'applicazione:

L'esistenza di un'applicazione scaricabile da Play Store è già un vantaggio in sè, in quanto non esistono piattaforme accademiche mobile simili. Questo rende possibile la reperibilità di più dati, in quanto le persone potranno effettuare le proprie scelte da dispositivo mobile e non solamente da PC.
Inoltre, tale applicazione andrà a colmare alcune lacune presenti nel sito: grazie al login utente, sarà possibile catalogare i voti anche secondo le caratteristiche degli utenti, quali età, sesso, stato di residenza, ecc. Si potranno inserire sempre più caratteristiche a seconda del tipo di dati che si vogliono studiare.

Gli utenti saranno invogliati a usare l'applicazione grazie alla classifica delle risposte: chi avrà giudicato le notizie nel modo corretto, avrà degli effetti grafici sul suo nome, visibile nella classifica pubblica e la gloria della vittoria.

In futuro, sarà possibile poi implementare un processo di gamification più ampio, in modo da invogliare sempre di più gli utenti a usare l'applicazione.

3 Implementazione e funzionalità:

3.1 Schema di navigazione di base:

L'activity principale (MainActivity) fornirà da subito l'elenco delle notizie da leggere. Come spiegato dalla guida iniziale (visibile solo al primo accesso all'applicazione), i titoli delle notizie saranno rossi o verdi, per indicare se la maggior parte degli utenti pensa che la notizia sia vera o falsa. Questo ci servirà, in futuro, per analizzare i dati relativi alla social influence.

Nel caso in cui l'utente decida di effettuare il login, le notizie mostreranno l'ora in cui sono state caricate e potranno essere votate, in modo che il voto sia salvato nei database. L'Actionbar presenta i pulsati per tornare alla home, per effettuare il login o per accedere al menu di navigazione. Da qui si potranno raggiungere altri Fragment, con contenuto diverso e appropriato:

- **Profilo**: Questa pagina sarà disponibile solo per gli utenti loggati, che avranno quindi fornito i propri dati. Tutti avranno la possibilità di essere modificati tramite un semplice click. Sarà inoltre possibile cambiare la foto personale, salvata tramite SharedPreferences e non su database;

- **Classifica**: Questa activity mostra una tabella con i giocatori che si sono posizionati in classifica votando almeno una notizia. Vi sarà anche mostrata la propria percentuale, in modo da incentivare gli utenti a valutare più news per farla salire. Il numero di voti generali totali sarà presente su ogni record.

  Inoltre, questo Fragment dispone di uno spinner, utile per cercare i votanti tramite una loro caratteristica specifica (in questa release sono disponibili ricerca per città e lavoro);

- **Contatti**: Questo Fragment è stato creato per implementare l'invio di mail tramite applicazione esterna.

- **Login/Logout**: Fragment necessario per effettuare il log-in. Una volta effettuata quest'azione, verrano sbloccate alcune funzionalità, come la pagina profilo o la possibilità di valutare le notizie. Una volta loggati sarà possibile effettuare il Logout.

  Inoltre, dalla pagina di login sarà possibile accedere alla pagina di creazione del proprio account. Saranno richieste una *Email*, un *Username*, e una *Password*.

  Sarà anche presente un checkbox, per il consenso alla condivisione dei dati personali e per le norme di utilizzo.

- **Guida**: Questo fragment non è altro che un riassunto di ciò che l'utente ha letto al suo primo login. Ciò è stato pensato poichè solitamente le applicazioni rimangono inusate per parecchio tempo, e in questo caso le funzionalità principali vengono dimenticate.

- **Amministrazione**: l'idea di questa pagina è puramente tecnica: solo l'amministratore la vedrà e potrà accedervi, usufruendo di alcune funzionalità, come la possibilità di vedere gli utenti iscritti, quella di promuoverli ad amministratore o visualizzare gli utenti che sono già di massimo grado e quindi che potranno vedere quella pagina.

  Vi è inoltre una **Search Bar**, con cui sarà possibile ricercare istantaneamente le notizie per parole chiave, gli utenti per caratteristica oppure selezionare una delle azioni disponibili nello spinner.

  Questa pagina potrebbe servire per un infinità di cose che però non sono state implementate, ma sicuramente per un'applicazione richiesta da un cliente potrebbero essere aggiunte: da questo pensiero è nata l'idea di quest'activity.

3.2 Motivazione delle scelte:

La scelta di utilizzare un Drawable Layout per costruire il menu è stata fatta per permettere all'utente di non dover faticare per fare l'azione desiderata, prevedendo in anticipo quale sarà il risultato di ogni sua mossa. Tutto

questo è stato ottimizzato dai **Fragment**, che migliorano l'user experience e diminuiscono la complessità temporale e spaziale dell'applicazione. Inoltre, anche la parte grafica sarà semplice ed elegante, per non disorientare l'utente con aspetti inutili e non necessari.

Passando allo storage, Il nostro Database è formato da 3 tabelle finali: *Users, News e Vote*. La tabella *users* ha come attributi l'username, la password, il ruolo (Amministratore o utente, alla creazione sempre utente), la percentuale e il nome. La tabella *news* ha come attributi il titolo e il valore di verità. La tabella *vote* ha come attributi il voto, l'username e il titolo. Username e titolo sono le chiavi private delle precedenti tabelle, formando così una relazione ternaria. La chiave primaria della tabella *vote* è quindi formata dalla chiave primaria della tabella degli utenti, più la chiave primaria della tabella delle notizie.

Questo database non è costruito con il classico modello Java ma è **Online**, in modo da avere i dati condivisi tra tutti gli utenti su ogni device. Le richieste HTTP al db vengono fatte con **Volley**, libreria che rende le connessioni con Android facili e soprattutto veloci.

Infine, è necessario specificare quali **pattern** sono stati utilizzati all'interno dell'applicazione, al fine di capire meglio l'architettura usata:

- **Singleton**: Serve a garantire che venga creata una e una sola istanza, fornendo un punto di accesso globale ad essa. In questo caso, l'istanza identifica un gestore delle richieste al web server che accede al database nel quale sono salvati i dati;
- **Observer-Observable**: Viene usato per segnalare la ricezione dei messaggi dal database. Infatti, gli Observers, verranno avvertiti quando ci sarà un cambiamento, in modo che possano gestire la propria interfaccia (es. classifica, amministrazione, ecc);

4 Autovalutazione e testing dell'applicazione:

La valutazione dell'applicazione è stata fatta con un introspezione: sono stati scelti svariati utenti, tra cui amici, familiari e colleghi per vedere cosa non capivano, cosa mancava o se ci fosse qualcosa di poco chiaro. Le tipologie di

utente sono state scelte appositamente: l'intero gruppo di testing era infatti composto da persone esperte nell'arte della programmazione mobile e da altri che erano in difficoltà a mettere il proprio cellulare in modalità silenziosa.

In molti casi, questo tipo di analisi è stato utile e ha dato alcune idee per l'applicazione (alcune implementate, altre no).

Inoltre, oltre ad un numero cospicuo di funzioni e ad un'efficacia perfetta, un'applicazione deve poter garantire una buona resistenza ai malfunzionamenti sotto ogni condizione di utilizzo. Per poter garantire questa affidabilità, è stato necessario pianificare un'attenta **strategia di testing**, mirata all'automatizzazione.

Per i motivi sopra elencati, sono stati effettuati anche test tecnici con le quali sono state controllate la maggior parte delle funzionalità principali.

Dato che la fase di testing deve poter trovare la propria collocazione nel processo che conduce dall'idea iniziale alla distribuzione al pubblico, è stato deciso di effettuarlo in modo parallelo allo sviluppo, utilizzando la filosofia di programmazione detta **Test Driven Development**, in modo da permettere al testing di guidare lo sviluppo, così che l'obbiettivo primario del codice sia quello di superare i test.

Tutto questo ha portato a dei grossi vantaggi, poichè il codice è sempre stato verificato su più valori (minimizzando quindi la possibilità di errore) e i test sono ripetibili via via che la progettazione avanza.
Passando alla praticità di quanto detto, i test sviluppati sono di due tipi:

- **I Test Unitari**, che si occupano di testare una singola funzionalità o classe, valutandone il lavoro isolatamente e sviluppati grazie alla libreria **JUnit4**.
- **Gli Integration Test**, che verificano l'azione congiunta di più elementi, sviluppati grazie ad **Espresso**. Questo framework creato da *Google*, è stato ideato per testare le interfacce grafiche, ed essendo molto elastico ha preso piede tra gli sviluppatori android, motivo per cui è diventato parte dell'*Android Testing Support Library*.

Inoltre, dopo un breve studio, Espresso offre delle potenzialità altissime: è possibile infatti simulare l'interazione con una view o oggetto, chiamando il metodo *onView()*, o utilizzare gli elementi integrati in un AdapterView, con il metodo *OnData()*.

Tuttavia, è necessario utilizzare il metodo *perform()* per effettuare l' azione vera e propria.

In conclusione, questo progetto ha portato ad imparare a lavorare con i dispositivi mobili, lavorando inoltre su un'applicazione che sarà utile per la nostra tesi di laurea e, con un po' di fortuna, anche dopo. Proprio per questo motivo, tutto il codice è strutturato in maniera modulare, in modo che in futuro si possa rispondere con facilità all'implementazione di funzionalità aggiuntive. Inoltre, i metodi più importanti presentano una documentazione scritta secondo gli standard dei **JavaDoc**, in modo da minimizzare il tempo perso a rileggere il codice per capirlo nella sulla interezza.