

ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA  
CAMPUS DI CESENA  
SCUOLA DI SCIENZE

CORSO DI LAUREA IN SCIENZE E TECNOLOGIE INFORMATICHE

# GEOFRIENDS

## APP PER LA GEOLOCALIZZAZIONE DI AMICI

Relazione finale in  
MOBILE WEB DESIGN

Relatore  
Dott. Mirko Ravaioli

Presentata da  
Giacomo Ceccaroni

Sessione Seconda  
Anno Accademico 2013/2014



# SOMMARIO

---

1	Introduzione .....	7
1.1	Il mondo dei dispositivi mobili.....	8
1.1.1	Android.....	9
1.1.2	iOS.....	10
1.1.3	Windows Phone.....	10
1.2	Servizi di geolocalizzazione .....	11
1.2.1	Foursquare.....	11
1.2.2	Runtastic.....	12
1.2.3	Trova i miei amici .....	12
2	Progetto .....	15
2.1	Scopo e nascita dell'app .....	15
2.2	Introduzione agli argomenti trattati .....	16
2.3	Progettazione .....	17
2.3.1	Analisi dal punto di vista dell'utente.....	18
2.3.2	Stesura e analisi delle schermate.....	18
2.3.3	Progettazione concettuale della base di dati.....	19
2.3.4	Operazioni principali eseguibili sulla base di dati. ....	23
2.3.5	Scelta di software e tecnologie.....	23
2.3.6	Progettazione di schermate e attività ad esse associate.....	28
2.3.7	Riprogettazione .....	28
2.4	Problemi affrontati.....	29
2.4.1	Logica della modalità offline .....	29
2.4.2	Gestione dei crash .....	30

2.4.3	Gestione dell'accesso singolo.....	30
2.4.4	Migrazione da localhost a web.....	30
2.4.5	Scambio dell'immagine profilo client-server.....	31
3	Implementazione .....	33
3.1	Lato server .....	33
3.1.1	Script PHP.....	34
3.1.2	Pagine web .....	36
3.1.3	Eventi .....	36
3.2	Lato Client: l'app.....	36
3.2.1	MainActivity .....	38
3.2.2	SignUpActivity.....	39
3.2.3	MapActivity .....	41
3.2.4	FriendListActivity .....	45
3.2.5	SearchActivity.....	46
3.2.6	SettingsActivity.....	47
3.2.7	AdvancedSettingsActivity.....	48
3.2.8	CustomDialogBuilder, CustomNotificationBuilder e CustomToastBuilder.....	49
3.2.9	CustomTaskBuilder.....	50
3.2.10	GCMIntentService .....	51
3.2.11	GPSTracker .....	51
3.2.12	User e UserAdapter .....	51
3.2.13	UsersMapFragment .....	51
3.2.14	Classi di appoggio .....	52
3.3	Grafica .....	52
4	Conclusione.....	55

4.1 Sviluppi futuri.....56

5 Sitografia.....57



# 1 INTRODUZIONE

---

L'affascinante mondo delle tecnologie mobili ha spinto molti pionieri dell'informatica allo sviluppo di software orientati a risolvere i problemi quotidiani delle persone. Il boom di questo fenomeno si è verificato nell'ultimo decennio ed è in continua evoluzione; basti pensare che nel 2011 il numero di smartphone venduti ha superato quello dei personal computer, facendo segnare un incremento del 57% rispetto all'anno precedente ed arrivando a consegnare 158.500 cellulari in tutto il mondo.

Alla base degli smartphone stanno le app: software leggeri, flessibili e facilmente installabili appositamente progettati allo scopo di essere ottimizzati per il mondo mobile. Queste piccole applicazioni sono suddivisibili in app native e web app; le prime necessitano di un'installazione sulla periferica e pertanto non sono cross-platform mentre le seconde sono collegamenti verso un applicativo remoto e, seppur utilizzabili da una qualsiasi piattaforma mobile, necessitano di una connessione internet.

Il cambiamento portato dall'espansione del mondo mobile ha scosso ogni ambito tecnologico, non solo il mondo dei computer; basti pensare all'impatto avuto sul mercato delle fotocamere digitali, ad oggi divenute quasi inutili visto il ruolo centrale giocato dalle fotocamere integrate negli smartphone che permettono inoltre di condividere gli scatti in tempo reale.

Un mercato che più di altri è stato rivoluzionato da questo avvento, è stato però quello della telefonia cellulare e fissa. Se prima il punto di forza degli operatori stava nella vendita di piani tariffari basati su chiamate e sms, ora è virato su promozioni con come punto cardine le connessioni dati. La crescita di capacità di elaborazione e l'aumento costante di memoria digitale a disposizione di smartphone e tablet ha infatti spinto gli utenti di tutto il mondo a spostare le operazioni informatizzate di ogni giorno su piattaforma mobile.

Ecco dunque la ragione del crescente accantonamento di chiamate ed sms: l'utilizzo spropositato di internet in ogni sua forma, rende di secondaria importanza il ruolo delle telefonate in quanto esistono servizi complementari facenti uso delle connessioni web. Anche in questo caso la rivoluzione è avvenuta prepotentemente e negli ultimi anni è in perenne progresso. Audiweb è una grande azienda che si occupa di rilevazioni statistiche su internet in Italia, ha constatato che nel 2014 gli italiani che si collegano alla rete utilizzando solo piattaforme mobili sono in maggioranza sia rispetto a coloro che invece usano soltanto PC, che su quelli che usano sia PC che piattaforme mobili.

È quindi innegabile l'enorme impatto sul mondo tecnologico causato dall'esplosione del mercato mobile. Sempre stando a quanto rilevato da Audiweb, gli italiani passano la maggior parte del loro tempo online su social network.

Quando si parla di social network si intendono quei servizi che permettono, partendo dalla creazione di un proprio profilo personale, di creare interazioni con altri utenti registrati al servizio. Anche in questo caso si parla di un concetto fortemente sfruttato negli ultimi anni il cui boom si ha negli anni 2005-2006. Basti pensare dal 2005 al 2010, l'uso dei social network tra i giovani americani è passato dal 7% all'86%, per poi stabilizzarsi negli anni a seguire.

Lo sviluppo di smartphone e social network è strettamente collegato: la possibilità di essere online in ogni luogo e in ogni momento porta gli utenti ad una maggiore condivisione di eventi e di pensieri personali tramite la propria rete sociale.

## 1.1 Il mondo dei dispositivi mobili

Quando si parla di dispositivo mobile si intende una periferica dalle dimensioni compatte in grado di svolgere diverse funzioni di computazione. Generalmente questi strumenti sono dotati di un display (touchscreen o non) e in certi casi di una tastiera fisica.

Le due categorie al momento predominanti di dispositivi mobili risultano essere smartphone e tablet. La differenza sostanziale tra le due risulta essere la dimensione dello schermo: mentre gli smartphone sono orientati a mantenere un modello adatto alla telefonia (mantenendo quindi la grandezza dei monitor attorno ai 3 – 4 pollici), i tablet nascono per esigenze di navigazione e multimedialità, con display che si aggirano attorno ai 9 pollici di dimensione. Dando uno sguardo al presente e al futuro, si possono aggiungere altri generi di dispositivi mobili in via di sviluppo come ad esempio smartwatch e smartglass. Questo genere di tecnologie dette *wearable device* offrono all'utente servizi strettamente legati a quanto potrebbe offrire uno smartphone, con la sostanziale differenza di essere dispositivi indossabili. I più in voga risultano essere gli orologi dal polso, spesso connessi allo smartphone per una migliore sincronizzazione con le attività dell'utente.

A livello software, i dispositivi mobili contengono un sistema operativo appositamente modellato sulle esigenze della periferica. Un sistema operativo di questo genere è paragonabile a un sistema operativo che opera su di un computer (ad esempio: Windows su PC), ma appositamente adattato alle problematiche di cui i dispositivi mobili soffrono quali l'alimentazione limitata a una batteria, le risorse fisiche limitate seppur in continuo sviluppo, la difficoltà di lettura e visualizzazione dei



contenuti legata alla ridotta dimensione del monitor e gli impedimenti legati all'accesso (spesso continuo) alle risorse di rete. Ad oggi, i principali produttori di sistemi operativi per dispositivi mobili risultano essere Google, Apple e Microsoft.

### 1.1.1 Android

Android è un sistema operativo open source per dispositivi mobili basato su kernel Linux, creato nel 2003 da un'omonima azienda e acquistato da Google nel 2005.

Inizialmente Android Inc, azienda fondata da Andy Rubin, Rich Miner, Nick Sears e Chris White, doveva creare Android perché servisse da sistema operativo per fotocamere digitali, in modo che gli utenti potessero installare nuove app e aggiungere nuove funzionalità ai loro dispositivi. Ben presto, però, i quattro fondatori decisero di cambiare rotta e virare verso il mercato degli smartphone, convinti che ben presto i “cellulari intelligenti” avrebbero sopraffatto le fotocamere. In poco meno di 12 mesi i fondi stanziati inizialmente dai fondatori finirono e il progetto rischiò di essere archiviato. In soccorso del robottino verde arrivò Steve Perlman, amico di Rubin ed ex ingegnere Apple, con una donazione di 10.000 dollari che permise al progetto Android di andare avanti.

Nel 2005, sul palcoscenico Android fa la sua apparizione anche Google. Quello che si avviava a diventare un gigante del web e dell'hi-tech in generale inizia a finanziare il progetto di Rubin e degli altri e, dopo pochi mesi, ne prende il controllo. Nel 2007, lo stesso anno della presentazione del primo iPhone, Google annuncia la costituzione dell'Open Handset Alliance (OHA), un consorzio che riuniva in un'unica sigla una lunga lista di produttori di smartphone e telefonini. Lo scopo era quello di gettare le basi per lo sviluppo di standard aperti (sia in ambito software sia in ambito hardware) in ambito mobile. E l'occasione, naturalmente, di presentare al mondo il suo sistema operativo mobile: Android.

A livello tecnico la piattaforma Android è un set di sottoinsiemi software composto da applicazioni Java le quali vengono eseguite su un framework object oriented, fondato anch'esso su Java. Le librerie Java vengono a loro volta eseguite tramite Dalvik, ovvero la macchina virtuale specifica per dispositivi mobili dotata di compilatore just-in-time. Al contrario di altri sistemi operativi per dispositivi mobile, Android è adatto a funzionare su smartphone e tablet di diversi produttori.

Una delle caratteristiche più risaltanti di Android sta nella denominazione delle diverse versioni, le quali sono indicate a livello non ufficiale, tramite un *codename* tradizionalmente ispirato alla pasticceria (come ad esempio “*Cupcake*” equivale alla versione 1.5 di Android). Nel giugno 2014,

Android lancia quella che ad oggi risulta essere la versione più recente, ovvero Kit Kat (Android 4.4.4).

Il negozio virtuale da cui è possibile scaricare app dedicate ai dispositivi Android (oltre a brani musicali, film e altro materiale multimediale) è il Google Play Store. A metà dell'anno 2013 è stata raggiunta la soglia dei 50 miliardi di download sul milione di app disponibili.

### *1.1.2 iOS*

iOS è un sistema operativo proprietario per dispositivi mobili appartenenti all'azienda informatica Apple. Il software nasce nel 2007, in occasione del lancio del primo iPhone e la sua versione corrente è la 8. Allo stesso modo del sistema operativo Mac, anche iOS è una derivazione di UNIX e usa un microkernel XNU Mach basato su Darwin OS a quattro livelli di astrazione. La dimensione del sistema operativo è pressappoco mezzo Gigabyte. Nel 2013 Apple risultava essere il secondo sistema operativo mobile con il 23% del mercato e quasi 400 milioni di dispositivi venduti.

Il negozio virtuale da cui è possibile scaricare app destinate ai dispositivi Apple è l'App Store, il quale vede la luce nel 2008 e permette lo scaricamento di applicazioni disponibili all'interno dell'iTunes Store. I software possono essere gratuiti o a pagamento e riservati ai proprietari di dispositivi mobili Apple quali iPhone, iPod touch e iPad. Una stima risalente a giugno 2014 rileva che i download dallo Store risultano essere 75 miliardi mentre il numero di app disponibili si avvicina a 1.200.000.

### *1.1.3 Windows Phone*

Windows Phone è il sistema operativo sviluppato da Microsoft presentato nel 2010. Il suo impatto nel mondo delle device mobile è stato perlopiù negativo; il sistema operativo, infatti, non ha mai goduto di grande popolarità e ciò è dovuto principalmente alla mancanza di applicazioni presenti nello Store e all'incapacità di sopperire ai gap tecnologici con la concorrenza. La versione attualmente in circolazione è la 8.1, con la quale Microsoft cerca di unificare lo sviluppo di app per PC e lo sviluppo di app per piattaforma mobile.

Il negozio virtuale dal quale scaricare software per Windows Phone è il Windows Phone Store il quale, a fine 2013, contava “appena” 190.000 app.

## 1.2 Servizi di geolocalizzazione

L'idea di un oggetto in grado di offrire tra gli altri servizi una tracciabilità geografica quasi sempre attendibile ha suscitato tra le persone diversi tipi di reazioni, che possono essere raccolte in due principali correnti di pensiero. Da un lato troviamo gli scettici, ovvero gli utenti che mettono in primo piano la protezione della loro privacy e che non vogliono dar modo ad altri di essere in grado di trovare la loro posizione corrente, per motivazioni più o meno frivole. Dall'altro lato vi sono invece coloro che trovano nella geolocalizzazione varie opportunità di condivisione e sviluppo. Molti sono i fattori che si possono sfruttare partendo dalle coordinate reperibili da uno smartphone associabile a un utente; dalla più basilare creazione di un navigatore, ad un videogioco che si mischia alla realtà del luogo in cui ci si trova.

Partendo proprio dalla possibilità di avere un GPS su cui lavorare, sono nate diverse app, molte delle quali di successo. Di seguito verranno elencati alcuni esempi di software per dispositivi mobili i quali si fondano sulla geolocalizzazione.

### 1.2.1 Foursquare

Foursquare è un'applicazione mobile e web disponibile per iOS, Android, Windows Phone e BlackBerry, che permette agli utenti registrati di condividere la propria posizione con i propri contatti. Foursquare è probabilmente l'app più utilizzata tra quelle basate sui servizi di geolocalizzazione ed ha avuto successo grazie ad una sorta di competizione tra gli utilizzatori.

Il check-in nei luoghi permette di ottenere punti necessari a scalare una classifica settimanale, basata sugli ultimi 7 giorni, della quale fanno parte i contatti della propria stessa città. I check-in possono inoltre essere condivisi, insieme ad un breve status, collegando Foursquare ai propri profili Facebook e Twitter.

Gli utenti ricevono inoltre dei "badge", dei riconoscimenti per aver raggiunto certi obiettivi, eseguendo il check-in in certi luoghi, a una certa frequenza o trovandosi in una certa categoria di luoghi. Se un utente esegue il check-in in uno stesso luogo più giorni di seguito e visita un luogo più di qualsiasi altro utente nei precedenti 60 giorni ne diventa sindaco e il suo avatar è inserito nella pagina relativa al luogo fino a quando un nuovo utente non esegue più check-in del sindaco in carica. I proprietari di un'attività hanno la facoltà, qualora la pagina relativa al luogo non sia stata creata da loro, di reclamarla e di offrire sconti e offerte al sindaco o agli utenti che vi fanno il

check-in. Gli utenti possono inoltre creare una lista pubblica di cose da fare e scrivere dei brevi suggerimenti per gli utenti che seguono check-in nel luogo stesso o in quelli vicini.

Al suo lancio nel 2009 Foursquare era disponibile in maniera limitata in sole 100 aree metropolitane in tutto il mondo. Nel gennaio 2010 il social network ha cambiato il modello alla base dei check-in permettendo di fatto di effettuarlo in qualsiasi parte del mondo. A marzo del 2010 il servizio ha superato i 500000 utenti attivi mentre a luglio 2010, in soli 4 mesi, è riuscito a superare la quota di 2 milioni.

### 1.2.2 Runtastic

Runtastic è una società mobile internazionale di fitness che combina lo sport tradizionale con le app, con il social networking ed elementi di serious game.

L'idea iniziale nacque durante un progetto all'università di scienze applicate di Hagenberg, in Austria, per tracciare e seguire il percorso delle barche durante le gare di navigazione. Dal momento che tale target di riferimento era troppo ristretto, si decise di focalizzare il progetto su sport più popolari, come la corsa, il ciclismo o il footing al punto da fondare un ramo della società dedicato alla creazione di dispositivi originali Runtastic, così da raccogliere dati direttamente tramite l'app. In seguito si possono sincronizzare i dati con il sito Runtastic allo scopo di condividere, studiare i percorsi e effettuare ulteriori analisi anche fisiche.

Runtastic offre inoltre un sito come parte di una più ampia piattaforma dedicata al fitness, affiancando le app mobili. Su questo, gli utenti possono trovare una ancora più completa e dettagliata analisi delle loro attività sportive le cui informazioni possono essere condivise con amici.

Tramite la geolocalizzazione, è inoltre possibile condividere in tempo reale la propria posizione con gli amici, i quali possono inviare incitamenti dalla propria postazione. Citando quanto detto dal New York Times, *“Abbastanza inquietante, ma interessante”*.

Nell'ottobre 2013, Runtastic ha stimato che l'app conta 95 milioni di download e 45 milioni di utenti registrati al sito.

### 1.2.3 Trova i miei amici

Trova i miei amici (conosciuto anche come Find my Friends, dal suo nome originale in lingua inglese) è un'applicazione sviluppata da Apple Inc. per tutti i dispositivi che montano come sistema operativo iOS.

Il software una volta installato permette all'utente di condividere la posizione del suo dispositivo in tempo reale ad un certo gruppo di persone, e di visualizzare su di una mappa la posizione dei dispositivi collegati a persone che l'utente ha aggiunto, le quali hanno accettato di pubblicare la loro posizione.

Il software mette a disposizione dell'utente diverse opzioni per permettergli di garantire la sua privacy: oltre alla possibilità di aggiungere e condividere la posizione con degli amici temporanei, i quali non potranno più ricevere aggiornamenti dopo un certo periodo di tempo, è possibile interrompere per un certo periodo di tempo il servizio, in modo che anche i propri amici non siano in grado di localizzare il proprio dispositivo. In quest'ultimo caso, gli altri utenti non vedranno altro che lo stesso messaggio che viene visualizzato quando è impossibile ottenere la posizione del dispositivo (come se questo fosse disconnesso da Internet). I problemi relativi all'utilizzo di questo tipo di applicativi, in termini di privacy, restano comunque molti e le discussioni in proposito sono ancora molte e aperte.



## 2 PROGETTO

---

Di seguito verrà analizzato il software sviluppato per la tesi, il quale ha richiesto pressappoco cinque mesi di lavoro. Il progetto coinvolge non solo la materia sul quale è basato, ovvero Mobile Web Design, ma anche gran parte delle altre materie svolte nel corso del triennio e in particolar modo Basi Di Dati.

È importante sottolineare che il progetto è nato come esame per la materia Mobile Web Design e che le prime fasi di esso sono state svolte in collaborazione con una compagna di corso mentre la tesi in sé è stata portata avanti dal sottoscritto.

L'app trattata si chiama GeoFriends.



*Figura 2.1 – Logo dell'app GeoFriends*

### 2.1 Scopo e nascita dell'app

L'app in questione si pone l'obiettivo di risolvere una delle comuni domande quotidiane cui ogni persona si può trovare a far fronte: "Dove ti trovi?". Può capitare infatti in svariati casi di voler essere in grado di conoscere la posizione di una persona in un dato momento, volerne seguire gli spostamenti o poter far tenere traccia dei propri cambi di locazione.

L'idea è nata da una giornata come le altre, in cui dovevo incontrarmi con alcuni amici per vedere una partita di calcio in un locale in cui nessuno di noi era mai stato. Per far capire a tutti gli invitati dove si trovasse il posto, sono servite una decina di chiamate e parecchi messaggi in cui contenuto

era spesso una descrizione rudimentale della zona circostante, come ad esempio “mi trovo davanti al benzinaio all’angolo; dove devo andare?”.

Lo scopo del software è dunque quello di evitare lunghe ed imprecise descrizioni ricche di ambiguità e incomprensioni che, inevitabilmente, risultano sfociare in ritardi e inutili litigi.

## 2.2 Introduzione agli argomenti trattati

Il software descritto nelle pagine successive, sarà utilizzato per dare un po’ più di precisione alla definizione di “per strada”. Utilizzando la geolocalizzazione, verrà infatti finalmente definito il punto della strada cui una persona si trovi. Niente più “Sono lì tra cinque minuti” che poi diventano un’ora; grazie ad una mappa saremo in grado di sbugiardare i nostri amici ritardatari tenendo traccia dei loro spostamenti in tempo reale.

Un ruolo fondamentale all’interno dell’app sarà quindi svolto dalla mappa, la quale verrà visualizzata come primissima schermata ad accesso avvenuto. Non sempre è necessario rieseguire il login: all’interno del dispositivo verrà tenuto conto dell’identità dell’utente proprietario e si cercherà di evitare più volte possibile la connessione e disconnessione al servizio. È ovvio che l’accesso di un utente potrà avvenire da una e una solta periferica; sarebbe paradossale la situazione in cui vi fosse una persona posizionata nello stesso istante in due luoghi distinti della Terra.

Un occhio di riguardo in tutto ciò naturalmente va alla privacy. L’utente avrà assoluta libertà di essere svincolato dall’invio della propria posizione mantenendosi in una modalità invisibile e interrompendo di fatto l’invio e la ricezione degli aggiornamenti relativi agli eventi di mappa.

C’è da tenere inoltre presente che non tutti gli utenti vedranno le informazioni di tutti gli altri utenti registrati all’app, ma soltanto quelle degli amici. Gli amici vengono aggiunti nella maniera più classica di ogni social network attuale: un utente cerca un secondo utente sulla rete per poi inviargli una richiesta d’aggiunta alla lista, richiesta che può essere accettata come anche no a discrezione del destinatario.

La registrazione al servizio è obbligatoria, ma non è necessario che avvenga tramite credenziali interne all’app. È stata infatti implementata l’opportunità di eseguire l’accesso tramite social network già esistenti, allo scopo di alleggerire le operazioni che l’utente dovrà svolgere.

Sempre grazie ai social network di terze parti è possibile condividere online la propria posizione. La maggior parte delle app include questa opportunità con lo scopo di ottenere pubblicità gratuita,



incoraggiando altre persone non ancora iscritte al servizio ad installare l'app sul proprio smartphone ed entrare a far parte della relativa rete sociale.

Le informazioni relative all'utente iscritto sono ridotte all'osso allo scopo mantenere leggero il database e non infastidire l'utilizzatore dell'app facendogli perdere troppo tempo con l'iscrizione. Tra gli altri campi, troviamo l'immagine profilo, la quale può essere acquisita via fotocamera o galleria e che potrà poi essere modificata in ogni momento.

Un altro aspetto che GeoFriends mette a disposizione è un servizio di notifiche il quale avverte l'utente in caso di necessità che verranno esaminate nel corso della documentazione. La ricezione delle notifiche non dipende dal rilevamento o meno della posizione ma è una scelta a parte che si basa sulla volontà dell'utente di essere o meno tenuto informato di quanto accade sulla rete dei suoi contatti (come, ad esempio, un amico che passa online).

In sintesi, quanto verrà descritto è la costruzione di un social network la cui funzione è quella di mantenere e condividere la posizione real-time degli utenti in linea.

## 2.3 Progettazione

La fase di progettazione è avvenuta in partendo da un'analisi delle situazioni che in linea di massima l'app avrebbe dovuto coprire adottando un approccio "a schermate", ovvero gettando in prima istanza una bozza di disegno di ciò che il software avrebbe dovuto mostrare all'utente. Ciò ha portato ad avere un'idea più chiara della situazione da affrontare e di separare processi prioritari da quelli secondari e da quelli facoltativi.

La sequenza dei passi di progettazione è stata dunque la seguente:

- Analisi dal punto di vista dell'utente.
- Stesura e analisi delle schermate.
- Progettazione concettuale della base di dati.
- Operazioni principali eseguibili sulla base di dati.
- Progettazione di schermate e attività ad esse associate.
- Scelta di software e tecnologie.
- Riprogettazione.

Qui di seguito verrà proposto un approfondimento dei passi sopra citati.

### *2.3.1 Analisi dal punto di vista dell'utente*

Calarsi nella parte dell'utente può essere complicato in realtà differenti o sconosciute da quelle a noi familiari; fortunatamente in questo caso il contesto all'interno del quale lavorare è piuttosto comprensibile e non è stato quindi difficile immaginare cosa un'utilizzatore si aspettasse dal software e in quale ordine le operazioni sarebbero dovute avvenire.

Le seguenti righe rappresentano ciò che un potenziale utente dovrebbe pensare, secondo l'analisi svolta: "Il servizio offerto è di base un social network, quindi il primo passo è quello della registrazione o, nel caso in cui questa fosse già stata effettuata in precedenza, il login. Solo una volta effettuato l'accesso posso sfruttare quanto messo a servizio da GeoFriends e mi aspetto di trovare all'apertura una mappa, avendo installato un'app che mostra le posizioni dei miei amici. Al primo accesso è consigliabile aggiungere dei contatti di persone a me familiari, altrimenti l'installazione dell'app ha ben poco senso. Per fare ciò, ho bisogno di eseguire una ricerca tra gli utenti già registrati in maniera da poter inviare la richiesta di amicizia. In alternativa, posso condividere il mio utente su social network come Facebook o Twitter cosicché altri utenti già iscritti possano aggiungermi alla loro lista di amici. La lista di amici dovrà essere in una sua pagina e tenere conto sia delle richieste accettate che di quelle ancora in fase di stallo. Mi aspetto inoltre di avere un minimo di possibilità per personalizzare sul mio utente".

### *2.3.2 Stesura e analisi delle schermate*

In prima istanza, quando ancora l'idea era solo tale, abbiamo cercato di fare un po' di chiarezza mettendo su carta una bozza rudimentale di ciò che sarebbe poi dovuta diventare GeoFriends, pensando con mentalità da utente. Lo scopo di questo passaggio, è stato avere qualcosa sotto mano su cui poter effettivamente prendere appunti e che tenesse traccia dei nostri pensieri.

Le pagine che abbiamo pensato potessero essere fondamentali sono:

- Login
- Registrazione
- Mappa
- Cerca
- Lista amici

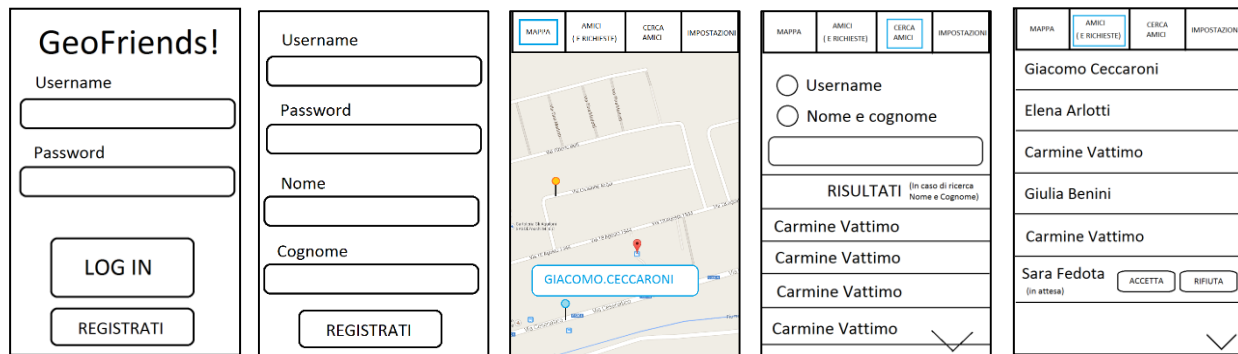


Figura 2.2 – Schermate

Il livello di astrazione a questo punto è ancora molto alto, pertanto sono state rimandate la grafica e tutto ciò che abbiamo ritenuto non fosse di primaria importanza.

### 2.3.3 Progettazione concettuale della base di dati

Il primo punto di progettazione vero e proprio affrontato riguarda la progettazione concettuale della base di dati, ovvero la rappresentazione su carta di un database disegnata tramite linee e figure geometriche.

Un database è un contenitore di dati in cui le informazioni in esso contenute sono strutturate e collegate tra loro secondo un particolare modello logico (relazionale, gerarchico, reticolare o a oggetti) e in modo tale da consentire la gestione e organizzazione efficiente dei dati stessi e l'interfacciamento con le richieste dell'utente attraverso operazioni (dette query) di interrogazione, inserimento, cancellazione e aggiornamento le quali funzionano grazie a particolari applicazioni software dedicati (DBMS). Il modello più utilizzato per la rappresentazione dell'architettura di un database è il modello E/R.

Il modello *entity-relationship* è un modello per la rappresentazione concettuale dei dati ad un alto livello di astrazione, formalizzato dal prof. Peter Chen nel 1976 i cui principali costrutti sono:

- Entità: rappresentano classi di oggetti che hanno caratteristiche in comune. L'istanza di un'entità rappresenta un singolo oggetto appartenente a quell'entità.
- Associazione: rappresenta un legame tra due o più entità.
- Attributo: rappresenta una proprietà che descrive un'entità o un'associazione.
- Identificatore: costituisce un sottoinsieme di attributi di un'entità che identifica in maniera univoca ogni istanza della stessa entità, è indicato graficamente come un cancelletto prima del nome degli attributi che compongono l'identificatore.

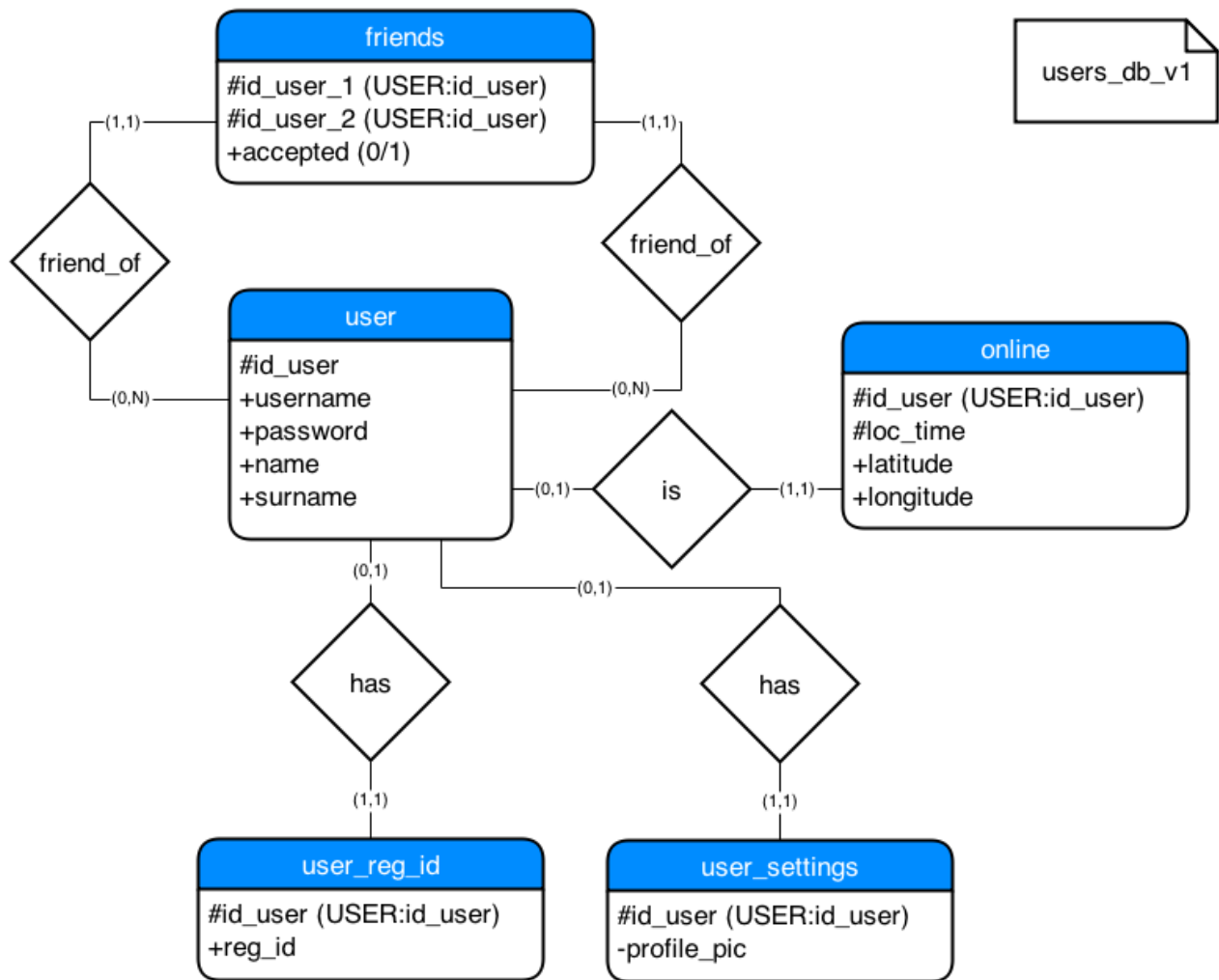


Figura 2.3 – ER del database di supporto all'applicativo

Analizzando la struttura del database si può notare come questo sia semplice e compatto. Questo perché i concetti da modellare sono pochi e con poche informazioni da mantenere.

L'entità più importante è quella che modella il concetto di utente, ovvero "user", tabella nella quale sono salvate le informazioni richieste ad ogni utilizzatore dell'app. Al momento della registrazione, viene inserito un record relativo all'utente ed è permanente. Non sono previste modifiche frequenti su tali record. Gli attributi dell'entità sono:

- id\_user: rappresenta la chiave primaria della tabella e non è altro che un numero intero incrementale.
- username: rappresenta un nome simbolico utilizzato all'interno dell'app per riconoscere univocamente un utente da parte di un altro utente. È utilizzato nel momento del login per

eseguire l'accesso ed è testuale e obbligatorio. Non può essere modificato una volta eseguita la registrazione.

- password: rappresenta la password necessaria, insieme allo username, per effettuare il login. È ovviamente obbligatoria e testuale e può essere modificata una volta eseguito l'accesso.
- name: rappresenta il nome dell'utilizzatore. È obbligatorio e può essere modificato una volta eseguito l'accesso.
- surname: rappresenta il cognome dell'utilizzatore. È obbligatorio e può essere modificato una volta eseguito l'accesso.

Associata ad ogni utente vi sono due tabelle opzionali, ovvero “user\_settings” e “user\_reg\_id”.

La prima rappresenta i record contenenti dati aggiuntivi collegati ad un utente, i quali necessitano di una tabella a parte in quanto più volatili o troppo poco snelli per una tabella statica come “user”.

I campi della tabella sono:

- id\_user: rappresenta la chiave primaria esterna che collega le impostazioni dell'utente al record corrispondente della tabella “user”.
- profile\_pic: rappresenta il campo contenente l'immagine profilo dell'utente. Il formato di salvataggio è *mediumblob*, ovvero un oggetto binario con una lunghezza massima di  $2^{24}-1$  caratteri.

Al momento non vi sono altri dati personali particolari da aggiungere a questa entità, ma la struttura viene mantenuta tale per eventuali sviluppi futuri.

La tabella “user\_reg\_id” è invece utile per mantenere il codice di registrazione al servizio Google Cloud Messaging. Onde evitare che un eventuale problema nella ricezione dell'id sia causa di mancato accesso all'app, l'esistenza dell'entità non è obbligatoria nonostante in genere venga creata al momento del login, salvo eccezioni.

I campi della tabella sono:

- id\_user: rappresenta la chiave primaria esterna che collega la tabella corrente al record corrispondente della tabella “user”.
- reg\_id rappresenta il campo contenente il codice di registrazione fornito dal server di Google Cloud Messaging nel momento dell'aggiunta al servizio. Il codice di registrazione del dispositivo è rappresentato come stringa alfanumerica che può contenere alcuni simboli, di lunghezza media di 163 caratteri. Il campo può contenere valore nullo.

La tabella “friends” mantiene salvate le informazioni relative allo stato di relazione tra due utenti. Come già spiegato infatti, alla base del progetto sta il concetto di rete sociale ed è pertanto importante tenere traccia delle liste contatti di ogni singolo utente. Per comporre le liste contatti vengono salvati dei record all’interno della tabella “friends” che collega un utente ad un altro tramite id. La tabella viene creata solo nel caso in cui vi fosse una richiesta di amicizia da parte di una persona nei confronti di un’altra.

Nello specifico, gli attributi della tabella sono:

- `id_user_1`: rappresenta una parte di chiave primaria esterna che collega la tabella corrente a un record corrispondente della tabella “user”.
- `id_user_2`: rappresenta la seconda parte della chiave primaria esterna che collega la tabella corrente a un altro record della tabella “user”, in maniera da conoscere univocamente lo stato di amicizia tra due utenti.
- `accepted`: rappresenta lo stato della richiesta di amicizia utilizzando un valore booleano che risulta essere *true* qualora l’amicizia fosse stata accettata o *false* nel caso invece in cui la richiesta fosse in fase d’attesa.

Infine è presente la tabella “online”, la quale modella un set di dati aggiuntivi che possono essere associati all’utente qualora questo avesse attivato la condivisione della propria posizione. È importante far presente che quando si parla della tabella “online” all’interno del database “users\_db\_v1”, si intende quell’entità che tiene conto non dell’accesso dell’utente all’app, ma del fatto che sta registrando periodicamente la propria posizione.

I campi della tabella sono:

- `id_user`: rappresenta la prima parte della chiave primaria, e collega la tabella corrente al record corrispondente della tabella esterna “user”.
- `loc_time`: rappresenta la seconda parte di cui è composta la chiave primaria ed è un campo utilizzato per contenere l’orario di accesso. Il tipo di questo campo è *datetime*.
- `latitude`: rappresenta la latitudine inviata dal dispositivo dell’utente in un dato istante. Il formato di salvataggio dell’informazione è decimale, con 11 posizioni intere e 8 posizioni per la mantissa.
- `longitude`: rappresenta la longitudine inviata dal dispositivo dell’utente in un dato istante. Il formato di salvataggio dell’informazione è decimale, con 11 posizioni intere e 8 posizioni per la mantissa.

#### *2.3.4 Operazioni principali eseguibili sulla base di dati.*

La fase successiva alla progettazione del database è stato lo studio delle operazioni con cui avremmo probabilmente dovuto aver a che fare. Stando all'analisi precedente:

- Al momento della registrazione deve avvenire l'inserimento di un nuovo utente, con tutte le informazioni necessarie per la sua descrizione.
- Una volta registrato, l'utente deve effettuare il login. Pertanto dovrà avvenire un controllo sui dati necessari all'autenticazione.
- L'utente loggato deve aver la possibilità di modificare i propri dati personali, tra i quali immagine profilo e password. È da tenere presente che quest'ultima necessita di un controllo ulteriore sulla corrispondenza della vecchia password.

Altre operazioni consentite all'utente loggato dovranno essere:

- Ricerca di amici, la quale può avvenire conoscendo lo username o, in alternativa, inserendo nome e cognome.
- Invio richiesta di amicizia all'utente desiderato.
- Gestione della lista contatti: ovvero visualizzazione gli utenti aggiunti, possibilità di accettare e rifiutare richieste d'amicizia e possibilità di rimuovere contatti in lista.
- Aggiornamento dei dati relativi alla mappa.
- Passaggio da offline (ovvero non visualizzabile sulla mappa) a online e viceversa.
- Gestione delle funzioni collegate al servizio Google Cloud Messaging.
- Logout dall'applicazione.

#### *2.3.5 Scelta di software e tecnologie*

La prima decisione affrontata è stata la scelta del sistema operativo più adatto per lo sviluppo dell'app. I sistemi operativi per dispositivi mobili studiati nel corso del triennio sono i tre principali presenti sul mercato, ovvero Android, iOS e Windows Phone. La scelta è ricaduta sul primo di questi perché, oltre ad essere quello il cui linguaggio risulta essere più familiare, è anche ad oggi il più diffuso al mondo.

## TOP US SMARTPHONE OPERATING SYSTEMS BY MARKET SHARE

DURING Q2 2013

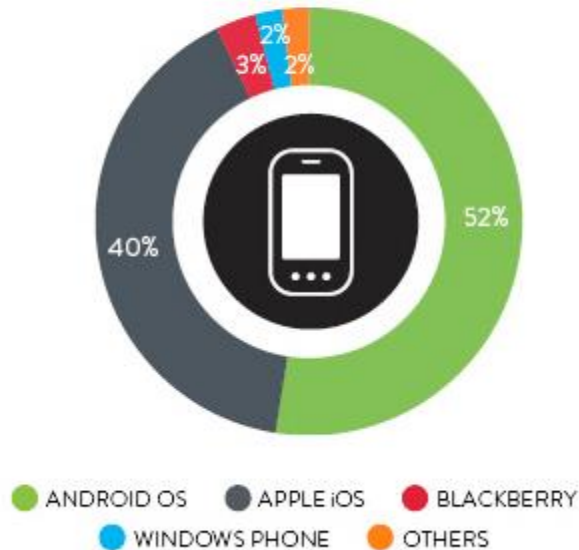


Figura 2.4 - Grafico sull'utilizzo dei sistemi operativi mobile

Il linguaggio di programmazione è Java, un linguaggio ad oggetti molto utilizzato in diversi ambiti. La sua diffusa popolarità è stato un fattore molto positivo per lo sviluppo in quanto sulla rete è stato molto facile trovare documentazione articolata e risoluzioni di problemi noti.

Se avessimo optato per lo sviluppo dell'app per Windows Phone, saremmo stati nella situazione opposta di quanto appena detto per Android: la documentazione sul web è più che scarsa essendo un sistema operativo ancora poco sviluppato rispetto alla concorrenza. Come si può inoltre valutare dai dati, il software di casa Microsoft è ad oggi molto snobbato dai clienti che ad esso preferiscono la “vecchia guardia”.

L'ultima alternativa era quella di sviluppare il software per iOS, ma per i dispositivi della “mela” esiste già un'applicazione molto simile a quanto da noi sviluppato, mentre per Android non esisteva a inizio progetto. Gli svantaggi dello sviluppo per iOS sono inoltre numerosi e non banali. In prima istanza, sarebbe stato caldamente consigliabile possedere un Mac e un iPhone, dispositivi che né io né la mia collega abbiamo; inoltre lo sviluppo sarebbe dovuto avvenire in Objective-C, un linguaggio esclusivo per la produzione di app per iOS e non popolare come è invece Java.

In sostanza, la scelta finale è ricaduta sul sistema operativo di casa Google, Android.



La versione minima di Android supportata è la 4.0 il cui nome in codice è "Ice Cream Sandwich" e che implementa le API 14. La scelta è ricaduta su tale versione allo scopo di ricoprire un'ampia fetta dei dispositivi più utilizzati dagli utenti, senza cadere troppo nell'obsoleto.

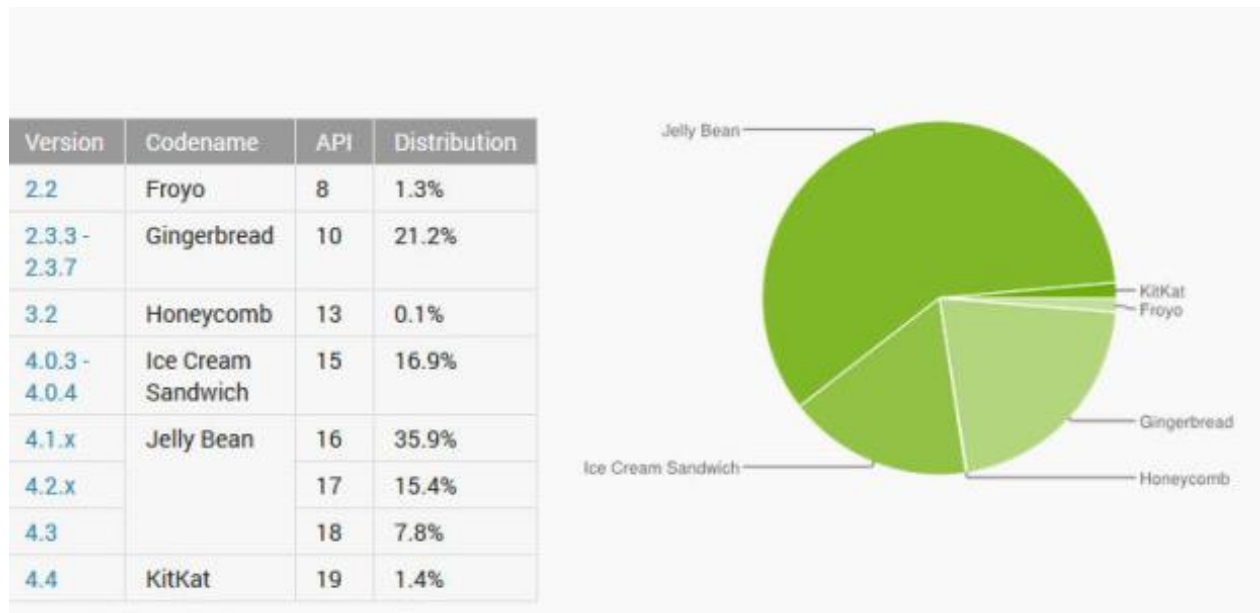


Figura 2.5 - Grafico sulla distribuzione delle principali versioni Android

Seguendo le specifiche precedentemente menzionate notiamo che l'applicazione richiede un continuo scambio di dati tra il dispositivo e uno spazio web propriamente configurato.

Lo spazio web selezionato per questa tesi mette a disposizione un server web Apache in grado di gestire pagine PHP e un database basato su MySQL.

Apache è il nome della piattaforma server Web sviluppata dalla Apache Software Foundation. È la piattaforma server Web modulare più diffusa, in grado di operare su una grande varietà di sistemi operativi, tra cui UNIX/Linux, Microsoft e OpenVMS, Apache è un software che realizza le funzioni di trasporto delle informazioni, di internetwork e di collegamento, ha il vantaggio di offrire anche funzioni di controllo per la sicurezza come quelli che compie il proxy. Gli amministratori del server possono usare il file httpd.conf, che è situato nella subdirectory conf della directory indicata durante la installazione. Questo file mette a disposizione tutta la libertà offerta dal server, quindi aggiungere moduli, estensioni, nuovi mime-type ed altro ancora. Il file .htaccess permette una ulteriore personalizzazione del webserver a livello di directory. È utilizzato in

particolare modo nell'hosting condiviso, per modificare le impostazioni standard fornite dal server stesso.

Oracle MySQL è un Relational database management system (RDBMS) composto da un client a riga di comando e un server. Entrambi i software sono disponibili sia per sistemi Unix che per Windows; le piattaforme principali di riferimento sono Linux e Oracle Solaris. MySQL è un software libero rilasciato a doppia licenza. I sistemi e i linguaggi di programmazione che supportano MySQL sono molto numerosi: ODBC, Java, Mono, .NET, PHP, Python e molti altri.

L'ambiente di lavoro è stato inizialmente installato in locale, scaricando un pacchetto preparato per gli sviluppatori allo scopo di facilitare le operazioni di interfacciamento del server web emulato sulla macchina. La piattaforma software è chiamata WAMPServer, acronimo che sta per Windows (sistema operativo installato sul PC), Apache, MySQL, PHP (in questo caso).

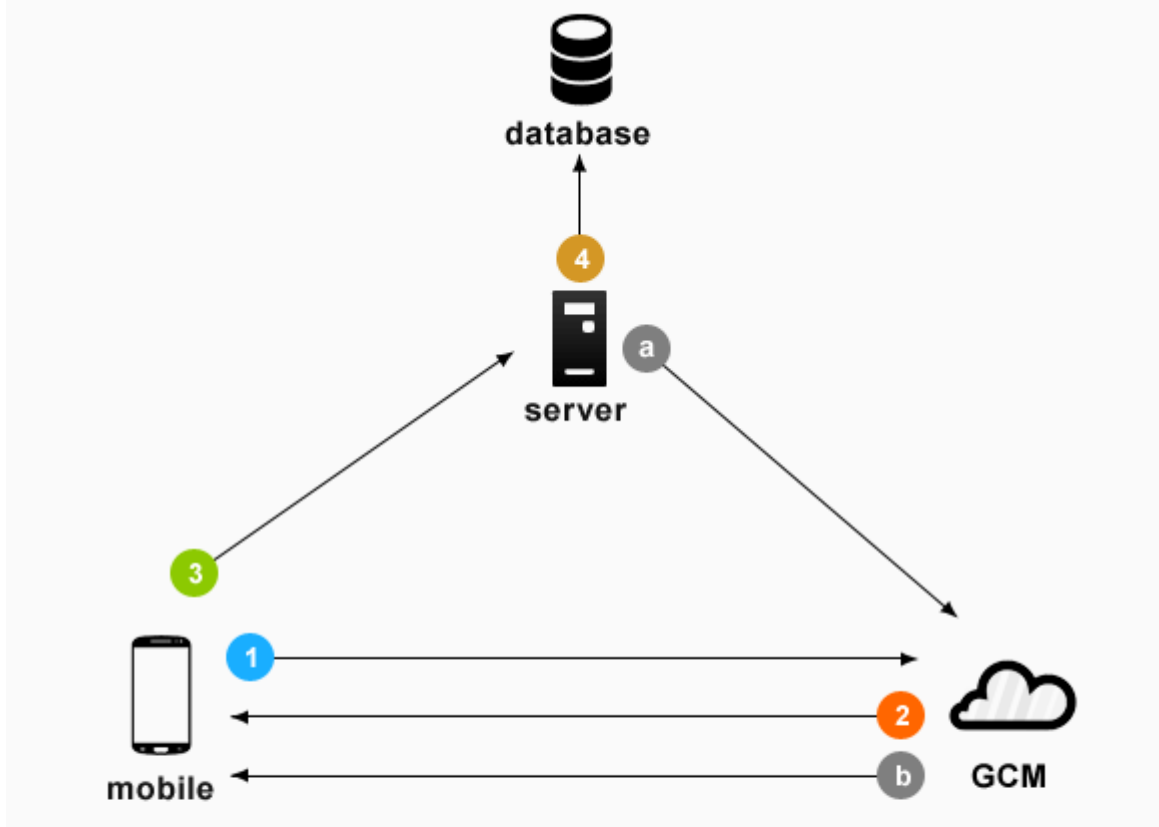
In linea con i servizi offerti la parte web che si occuperà di elaborare le richieste effettuate dagli utenti tramite l'applicazione saranno scritte in PHP. Saranno le stesse pagine PHP ad interfacciarsi col database permettendo quindi l'accesso a tutti i dati nelle modalità richieste.

La scelta di salvare tutti i dati online è obbligata: per far sì che ogni utente conosca le posizioni degli amici è infatti necessario un costante aggiornamento possibile soltanto utilizzando un servizio in linea, e lo stesso vale per il trattamento della rete sociale.

Ogni operazione effettuata dall'applicazione sarà una richiesta HTTP ad una pagina PHP specifica che produrrà un risultato testuale o in formato JSON. Con JSON si intende JavaScript Object Notation ed è un formato adatto per lo scambio dei dati in applicazioni client-server.

L'applicazione si occuperà poi di gestire la risposta nel modo più opportuno.

Per lo scambio di messaggi tra i dispositivi è stato comodo utilizzare il servizio fornito per i dispositivi Android, chiamato Google Cloud Notification o GCM. Questo servizio permette l'invio di dati da un server proprietario a tutti i dispositivi Android ad esso collegato, e anche di ricevere messaggi dai dispositivi grazie alla stessa connessione.



1. First android device sends **sender id, application id** to GCM server for registration.
2. Upon successful registration GCM server issues **registration id** to android device.
3. After receiving registration id, device will send **registration id** to our server
4. Our server will store **registration id** in the **database** for later usage

Figura 2.6 - Funzionamento del servizio Google Cloud Messaging

Il servizio GCM si occupa della gestione di tutti gli aspetti relativi alla coda di messaggi e della consegna di questi al dispositivo Android destinatario sul quale è installata l'app. Il servizio è gratuito (fino al raggiungimento di un certo numero di utenze).

### 2.3.6 Progettazione di schermate e attività ad esse associate

Dopo la progettazione del lato server descritta fino ad ora, siamo passati all'analisi orientata al front-end parlando delle varie schermate (dette anche *activity*) da implementare, al fine di gestire al meglio i processi precedentemente descritti.

Le *activity* identificate sono state:

- Pagina iniziale di login.
- Pagina di registrazione con i campi relativi all'utente.
- Pagina con la mappa, la quale diventerebbe la pagina principale a login avvenuto.
- Pagina con la lista dei contatti aggiunti e delle richieste d'amicizia.
- Pagina per la ricerca di utenti in rete.
- Pagina per la modifica di impostazioni personali.
- Pagina per la modifica di impostazioni relative all'app.

Le prime due schermate descritte sono utilizzabili da qualsiasi persona, mentre le successive sono riservate agli utenti loggati.

### 2.3.7 Riprogettazione

Come già accennato in precedenza, l'app è partita come progetto d'esame svolto in coppia, per diventare solo in un secondo momento il mio personale progetto di tesi. È dunque importante analizzare le fasi affrontate per portare la versione di GeoFriends dalla zero alla uno, ovvero ampliarne le funzionalità in modo da renderla più completa e performante.

La prima evoluzione avuta è stata nel database: inizialmente non era infatti prevista l'inclusione dell'immagine profilo, del servizio Google Cloud Notification e certi campi erano formattati in maniera differente.

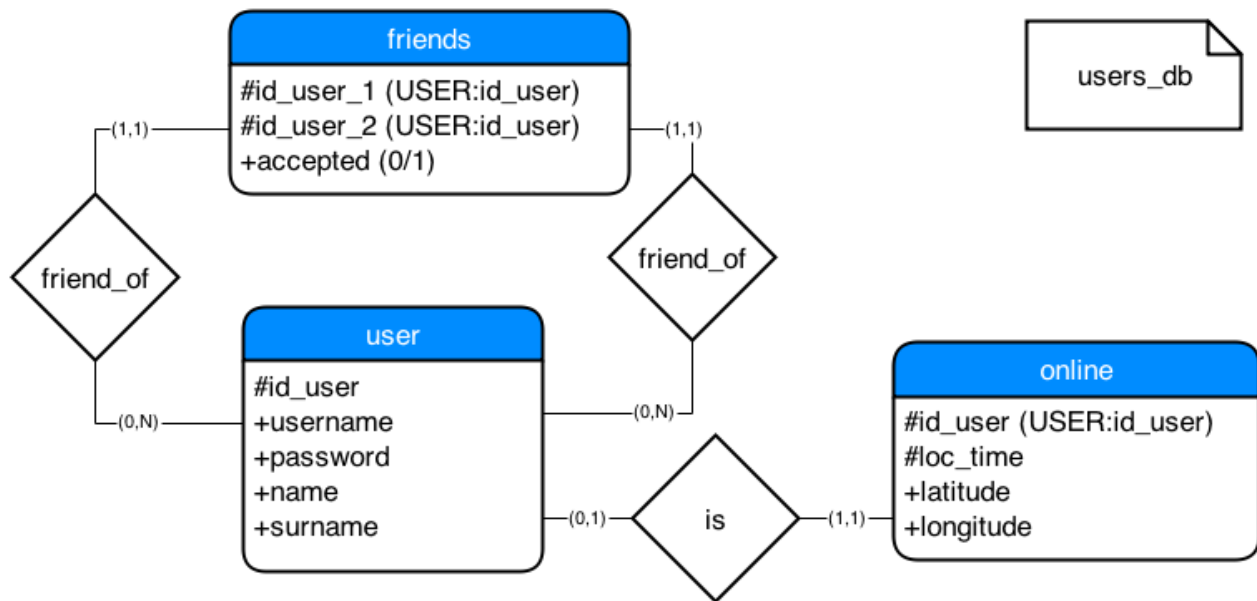


Figura 2.7 - Prima versione dell'ER relativo al database di supporto all'applicativo

In secondo luogo vi sono stati dei ritocchi pesanti nel campo delle operazioni supportate: la modifica dei dati dell'utente non era infatti inizialmente implementata, così come anche l'aggiornamento manuale della mappa, la modalità "offline" e la parte relativa alla personalizzazione delle impostazioni.

È stata inoltre aggiunta un supporto per l'integrazione con i social network già esistenti, quali Facebook e Twitter: entrambi sono utilizzabili per la condivisione della propria posizione in un dato momento e il primo permette persino il login senza bisogno di un'ulteriore registrazione interna a GeoFriends.

## 2.4 Problemi affrontati

I problemi affrontati durante la fase di analisi sono stati numerosi e hanno impiegato molto più tempo del previsto per essere risolti.

### 2.4.1 Logica della modalità offline

L'app inizialmente era stata costruita in maniera da consentire all'utente loggato di essere rintracciabile e in grado di rintracciare. Qualora però l'utente fosse risultato scollegato (anche solo, ad esempio, per una mancanza di copertura della rete di qualche secondo di durata), sarebbe avvenuto

un logout automatico che avrebbe riportato alla pagina iniziale. Questo evento è chiaramente sconveniente: in un ambiente reale in cui la connessione va e viene, è infatti scomodo dover reinserire le proprie credenziali ad ogni mancanza dovuta alla rete.

Il problema è stato risolto grazie ad una modalità “ponte” interposta tra lo stato di logout e quello di “online sulla mappa”, chiamato “modalità offline”. Durante tale configurazione si è infatti in grado di mantenere la sessione del proprio utente aperta all’interno dell’app, senza però poter eseguire alcuna operazione relativa alla mappa.

I casi in cui viene attivata la modalità offline sono 3:

- Mancata copertura di rete, quindi Wi-Fi e connessione dati a pacchetto entrambe irraggiungibili.
- Impossibilità di individuare la posizione corrente.
- Scelta da parte dell’utente di essere offline.

#### 2.4.2 Gestione dei crash

Un argomento da sempre ostico all’interno della programmazione è la gestione delle eccezioni runtime. Per quanto un’app possa infatti essere perfetta, potrebbe sempre capitare un improvviso errore non gestito che causi l’improvviso arresto del software durante il suo utilizzo.

Al fine di non causare alcuna perdita di dati all’interno della sessione in caso di crash, i dati dell’utente sono mantenuti in una zona di memoria riservata all’applicazione.

#### 2.4.3 Gestione dell’accesso singolo

All’interno di un’app basata sulla geolocalizzazione come quella descritta, appare evidente l’accesso ad un utente debba essere a singola istanza; non è pertanto ammissibile che una persona esegua il login da più periferiche e, pertanto, risulti essere in più posti contemporaneamente.

Onde evitare che venisse a crearsi tale situazione, la logica è stata quella di mantenere il più “giovane”. Ogni qualvolta un utente esegue l’accesso viene inserito nella base di dati e, nel caso in cui all’interno di essa vi fosse già una registrazione di tale sessione, questa verrebbe sostituita causando il logout dell’utilizzatore in seguito ad una notifica d’avvertimento.

#### 2.4.4 Migrazione da localhost a web

Il primo ambiente di crescita dell’app è stato il *localhost*, ovvero un server web emulato sullo stesso PC utilizzato per lo sviluppo. Esso è però limitato in quanto non è utilizzabile da dispositivo mobile

se non sotto una connessione wireless che comprenda anche il PC. È inoltre necessario cambiare frequentemente l'indirizzo al quale l'app punta per scaricare i dati dal database, specie in presenza di emulatori.

È stato così ritenuto opportuno virare su un servizio web gratuito piuttosto che continuare a sviluppare il locale. La scelta ha portato non pochi problemi; primi fra tutti, quelli di compatibilità tra le diverse versioni di server e database manager. Inoltre le latenze temporali tra richiesta e risposta hanno fatto emergere problemi che prima, lavorando in locale, non si erano presentati.

#### *2.4.5 Scambio dell'immagine profilo client-server*

Il download e l'upload di un'immagine è stato gestito convertendo questa in una stringa. Eppure per il passaggio di una stringa di così elevata lunghezza, sono state necessarie due classi dedicate appositamente a risolvere tale problema; a causa infatti di un bug negli *stream* di input, l'immagine veniva scaricata parzialmente e il metodo restituiva, pertanto, errore. Fortunatamente, altri programmatori erano incorsi nello stesso problema prima di me e avevano pubblicato online la soluzione prima di me.



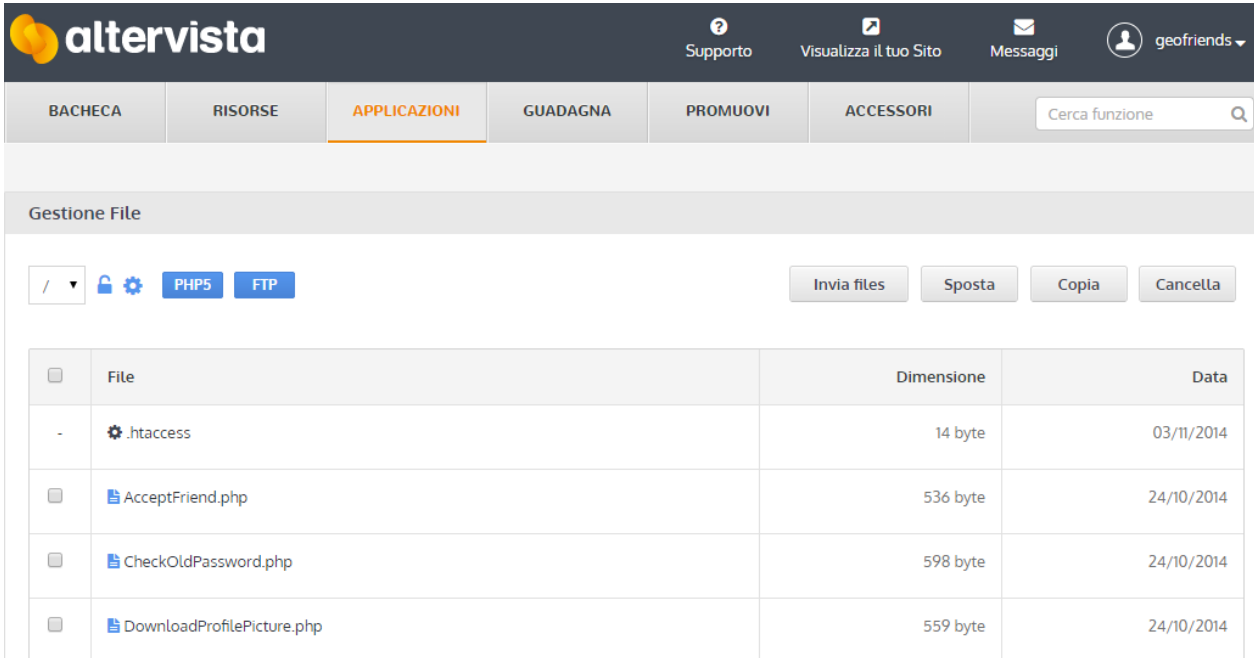


## 3 IMPLEMENTAZIONE

All'interno di questo capitolo verrà trattato più nel dettaglio lo sviluppo dell'app descrivendo quello che è il codice sorgente utilizzato per la creazione delle varie parti della stessa nonché l'impalcatura online a supporto del servizio. Verranno inoltre visualizzate le varie schermate di cui si compone l'app mettendone in luce i singoli campi e gli elementi compresi in ognuna di esse.

### 3.1 Lato server

Altervista è una piattaforma web italiana che offre la possibilità di aprire gratuitamente un sito web, un forum o un blog avendo sin dall'inizio a disposizione un dominio di terzo livello (nomesito.altervista.org) con delle caratteristiche base. L'idea è quella di offrire una casa sul web dove pubblicare i propri contenuti: a costo zero e con la possibilità di guadagnare, grazie alla pubblicità, con il proprio traffico web.



The screenshot shows the Altervista website management interface. At the top, there is a navigation bar with the Altervista logo and several utility links: Supporto, Visualizza il tuo Sito, Messaggi, and a user profile for 'geofriends'. Below this is a secondary navigation bar with tabs for BACHECA, RISORSE, APPLICAZIONI (which is highlighted), GUADAGNA, PROMUOVI, and ACCESSORI. A search bar labeled 'Cerca funzione' is also present. The main content area is titled 'Gestione File' and displays a file manager interface. It includes a toolbar with a dropdown menu, a lock icon, a settings gear, and buttons for 'PHP5' and 'FTP'. Action buttons for 'Invia files', 'Sposta', 'Copia', and 'Cancella' are also visible. The file list below has columns for checkboxes, file names, dimensions, and dates.

<input type="checkbox"/>	File	Dimensione	Data
<input type="checkbox"/>	-  .htaccess	14 byte	03/11/2014
<input type="checkbox"/>	AcceptFriend.php	536 byte	24/10/2014
<input type="checkbox"/>	CheckOldPassword.php	598 byte	24/10/2014
<input type="checkbox"/>	DownloadProfilePicture.php	559 byte	24/10/2014

Figura 3.1 - Pagina web del sito Altervista

Tra gli altri servizi offerti gratuitamente da Altervista, vi è un database online accessibile tramite l'interfaccia fornita da PhpMyAdmin; il Relational database management system (RBMS) utilizzato è MySQL. Il Web Server messo a disposizione dal servizio online è Apache 5.1.71 e la versione di PHP è la 5.3.

### 3.1.1 Script PHP

Le operazioni sul database vengono garantite da una serie di pagine PHP accessibili all'app che comunicano utilizzando il metodo POST. Le interrogazioni avvengono La maggior parte di essi restituisce un codice o una stringa come risultato. I file in questione sono:

- AcceptFriend.php: script utilizzato per accettare una richiesta d'amicizia
- CheckOldPasword.php: script utilizzato per verificare che la vecchia password inserita da un utente in fase di modifica della stessa, sia corretta.
- db\_connect.php: script utilizzato per le funzioni di connessione e disconnessione dal database.
- db\_functions.php: script utilizzato per contenere alcune funzioni chiave che coinvolgono da vicino il database tra le quali la messa online e offline di un utente, la registrazione di un utente su server e l'ottenimento di id in seguito a un'apposita richiesta. Queste funzioni non sono mai chiamate direttamente dall'app, ma possono essere accessibili solo da altri script o pagine PHP.
- DownloadProfilePicture.php: script utilizzato per lo scaricamento di un'immagine profilo corrispondente a un id dato.
- GCM.php: contiene le funzioni necessarie per il corretto utilizzo delle Google Cloud Notification. Al momento in realtà, all'interno della classe vi è soltanto una funzione utile per l'invio delle notifiche push ad altri dispositivi, e che riceve in ingresso l'id del client destinatario e il messaggio da inviare.
- GCMGetFriendRegIDFromUserID.php: script utilizzato per "convertire" l'id di un utente nell'id corrispondente alla registrazione su server GCM.
- GCMGetRegisteredFriends.php: script utilizzato per scaricare la lista degli amici registrati su server GCM in un dato momento, una volta ricevuti come input l'id dell'utente e il corrispondente id di registrazione sul server GCM.
- GetAllFriends.php: script utilizzato per estrarre tutti gli amici dato un id\_user. Quanto restituito è un oggetto JSON contenente tre JARRAY (ovvero vettori codificati), i quali sono rispettivamente composti da amici online, amici offline e richieste d'amicizia in attesa. Ciò facilita la decodifica dei valori da parte dell'app la quale riconoscerà già tre vettori distinti tra di loro.

- GoOffline.php: script utilizzato per rimuovere un utente dalla tabella “online” conoscendo l’id\_user.
- GoOnline.php: script utilizzato per aggiungere una nuova riga alla tabella “online”, richiamando una funzione messa a disposizione da db\_functions.php che necessita in input di id, latitudine, longitudine e orario corrente.
- Login.php: script utilizzato per controllare se i campi di login corrispondono a un utente registrato, ovvero se username e password risultano essere corretti. In caso affermativo, viene restituito l’id dell’utente, ovvero il valore univoco identificativo corrispondente all’interno della tabella “user”.
- LoginFacebook.php: script utilizzato per coloro che eseguono il login collegandosi tramite Facebook. In questo caso non sarà infatti richiesta nessuna password, ma soltanto uno username che viene estrapolato da una combinazione di caratteri uniti a un identificatore univoco fornito dall’utenza Facebook. In questo caso viene restituito un corrispondente id all’interno della tabella “user” qualora l’utente avesse già eseguito l’accesso via Facebook in precedenza, mentre in caso contrario verrà creato un utente sul momento utilizzando i dati presi dal social network.
- Logout.php: script utilizzato per rimuovere una device registrata su server GCM.
- register.php: script utilizzato per la registrazione di una nuova device all’interno della tabella user\_reg\_id. Riceve in input l’id dell’utente e l’id fornito dal server GCM.
- RemoveFriend.php: script utilizzato per rimuovere un amico o rifiutare una richiesta in attesa, dati i due id coinvolti nella relazione.
- SearchName.php: script utilizzato per ricercare un utente conoscendone nome e cognome. Lo script restituisce in output una lista contenente tutti gli utenti che corrispondono a quanto richiesto.
- SearchUser.php: script utilizzato per ricercare un utente conoscendone lo username. Essendo lo username univoco, il risultato saranno sempre i dati di un utente al massimo.
- send\_post\_message.php: script utilizzato per inviare dall’app un messaggio a una seconda device. Utilizza al suo interno il metodo di invio notifiche contenuto in GCM.php.
- SendFriendRequest.php: script utilizzato per inviare una richiesta di amicizia da un utente ad un altro.

- `SendMyLocation.php`: script utilizzato per l'aggiornamento dei dati relativi alla posizione di un certo utente.

### 3.1.2 Pagine web

L'app non è correlata ad alcuna pagina web in quanto la funzione del software è strettamente collegata alla mobilità, e quindi non avrebbe avuto senso creare una versione web del servizio.

È presente una sola pagina “`index.php`” necessaria in fase di test, al fine di visualizzare gli utenti registrati sul server GCM in un dato momento.

### 3.1.3 Eventi

È possibile incorrere in casi in cui un utente online non aggiorni per un certo periodo i dati relativi alla propria posizione. Basti pensare alla perdita di segnale dovuta a una qualsiasi mancanza della rete, o allo spegnimento del cellulare mentre la sessione è ancora attiva.

Qualora avvenisse un mancato aggiornamento della propria posizione, l'utente comparirebbe comunque (erroneamente) nella mappa dei suoi amici. Onde evitare questa situazione, la soluzione è stata quella di creare un'istruzione schedulata il cui compito è quello di rimuovere utenti segnalati come online, inattivi per più di due minuti; ciò permette all'utente di risultare comunque loggato all'interno di GeoFriends, seppur senza godere dei servizi legati agli aggiornamenti di mappa.

Tra i moduli aggiuntivi messi a disposizione da MySQL vi è quello degli Eventi, ovvero oggetti che si attivano in base ad un riferimento temporale ed effettua una determinata azione traducibile in una *statement SQL*. Purtroppo questa funzione non è utilizzabile all'interno del database messo a disposizione da Altvista. In alternativa Altvista fornisce il servizio PHP Cronjob il quale esegue una pagina ogni tot di tempo prestabilito e rende quindi possibile inserire la *query* nella pagina.

## 3.2 Lato Client: l'app

L'ambiente di sviluppo integrato (IDE) utilizzato è il software Eclipse, versione 4.2.0. Questo software open source è compreso all'interno di un package già pronto per l'uso chiamato ADT SDK Bundle (versione 23) che, come si può intuire già dal nome, mette a disposizione le SDK (traducibile come “pacchetto di sviluppo per applicazioni”, ovvero un insieme di strumenti e documentazione forniti dai produttori di software ai programmatori) integrate. L'installazione dell'ambiente

è davvero rapida e intuitiva; c'è da dire però, che le difficoltà possono sorgere in caso di aggiornamenti. Essendo inoltre un software open source e cross platform, non è raro incorrere in bug di rendering dell'interfaccia o di referenziazione e collegamenti a librerie esterne.

Eclipse mette a disposizione anche diverse macchine virtuali che emulano vari modelli di smartphone con Android, utili per il testing. Queste macchine virtuali peccano però in efficienza, e ho pertanto ritenuto opportuno virare su alternative più valide.

Genymotion è un emulatore Android relativamente veloce che integra diverse immagini di macchine Android scaricabili online le quali vengono montate utilizzando VirtualBox in background. Tra le altre funzionalità, è importante ricordare il GPS emulato che permette di impostare manualmente le coordinate, la fotocamera emulata collegabile alla webcam e le installazioni via *Drag&Drop* di APK e Zip. Quest'ultima funzionalità citata è stata fondamentale nel momento dell'inclusione dei Google Play Services, senza i quali la mappa non sarebbe potuta essere utilizzabile.

Verranno ora descritte le principali classi che compongono l'app GeoFriends, mettendo per prime le *activities* di cui il software si serve.

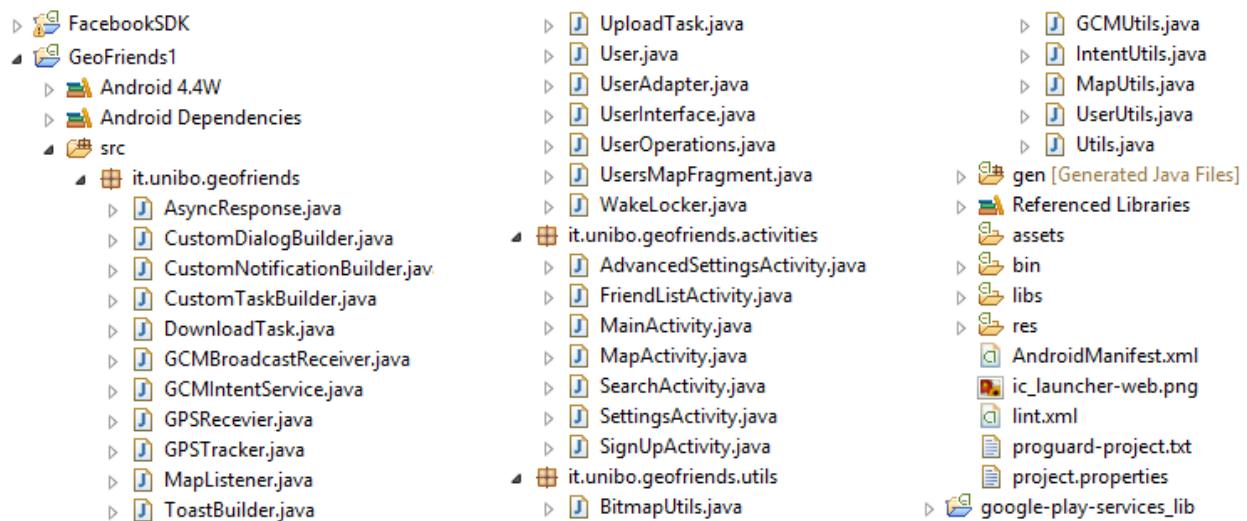


Figura 3.2 - Classi e directories di cui l'app è composta

### 3.2.1 MainActivity

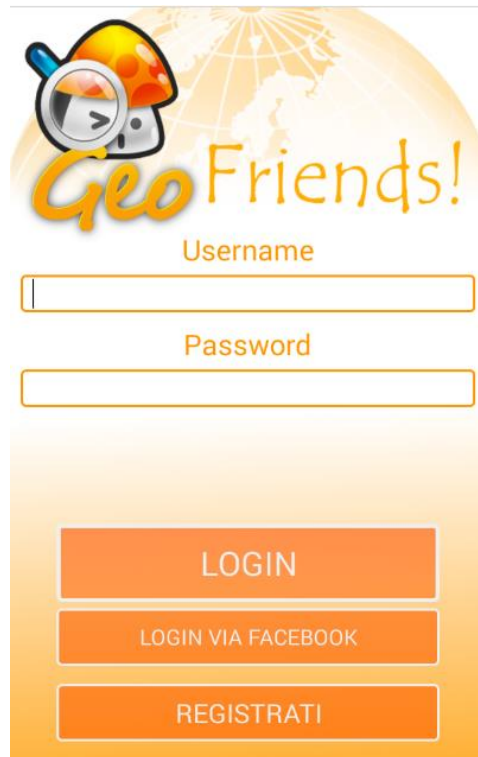


Figura 3.3 - Pagina di login

La prima schermata di fronte alla quale un utente senza ancora alcuna sessione attiva si trova è quella dell'accesso. Qui vengono richiesti all'interno di due *EditText*, i dati utili per l'autenticazione. Questi vengono passati allo script appropriato che restituisce un codice il quale a sua volta, sarà confrontato con quelli già previsti nell'app.

Alla pressione del tasto "Login" viene controllata la posizione corrente per una prima geolocalizzazione; nel caso in cui questa non fosse rintracciabile, è impossibile proseguire. Una volta passato questo primo controllo, gli scenari nei quali è possibile incorrere sono "Errore del server", "Username o password non validi" e "Login riuscito". Nei primi due casi, non viene eseguito alcun comando, mentre nel terzo (in seguito alla ricezione dell'id dell'utente) vengono salvati i dati legati all'utente all'interno delle *shared preferences*. È inoltre importante sottolineare che in questo primo momento avvengono la registrazione all'interno del server GCM e l'avvio del servizio di *tracking* tramite GPS, necessario per l'invio delle proprie coordinate al server.

Vi è la possibilità di eseguire il login pur senza aver eseguito in precedenza alcuna registrazione al servizio, ma semplicemente fornendo i propri dati del profilo Facebook. Alla pressione del tasto "Login via Facebook", viene richiamato un metodo fornito dalle SDK di Facebook il quale estrae

informazioni dalla propria app se installata sul dispositivo, o richiede all'utente di eseguire l'accesso tramite un apposito form. Ciò che viene utilizzato in seguito a tale operazione sono il nome, il cognome e un id univoco, tutti dati contenuti all'interno di una struttura di tipo *GraphUser*, formato fornito da Facebook per la memorizzazione degli utenti.

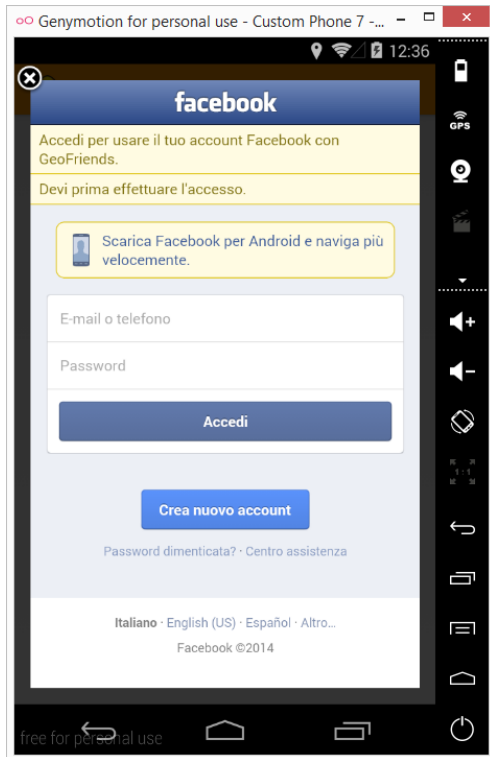


Figura 3.4 - Accesso tramite Facebook

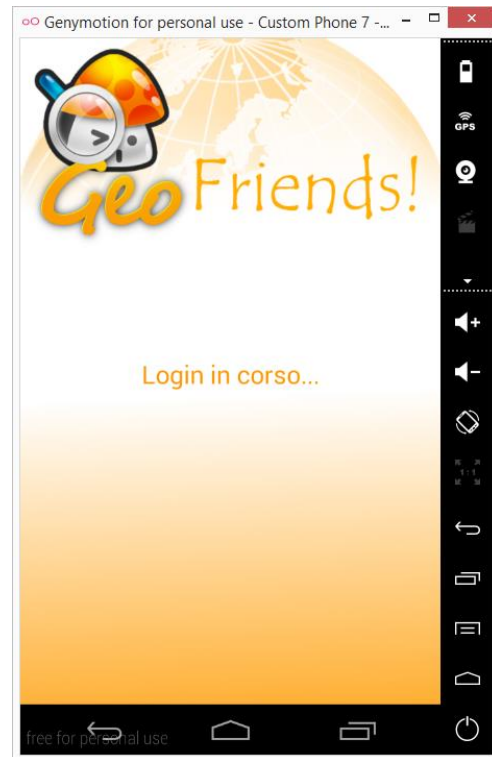


Figura 3.5 - Schermata d'attesa

L'ultimo pulsante è quello della "Registrazione", il quale rimanda all'*activity* descritta di seguito. È importante far notare che qualora l'utente avesse già eseguito l'accesso in precedenza, evitando di fare il logout, questa schermata verrebbe saltata e sarebbe impossibile da raggiungere nuovamente fino ad avvenuto logout.

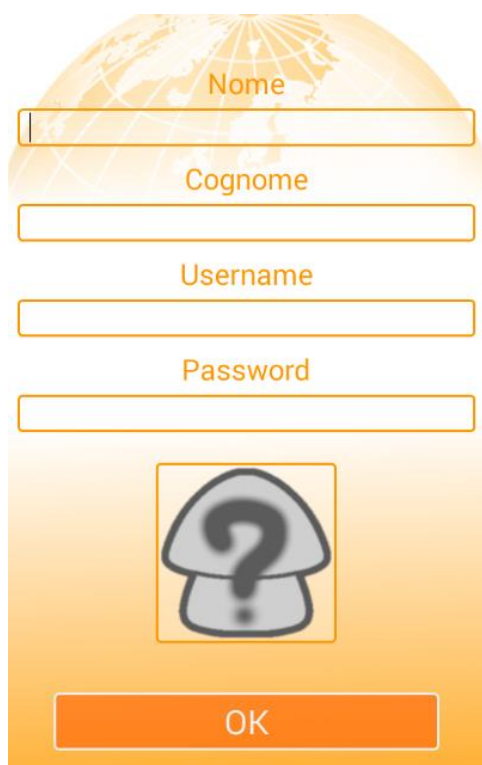
### 3.2.2 *SignUpActivity*

Questa pagina serve per registrarsi al servizio, inserendo i propri dati personali e un'eventuale immagine profilo. Durante la compilazione dei campi, avvengono diversi controlli:

- *Name* e *Surname*: il nome e il cognome non possono essere vuoti e la loro lunghezza deve essere compresa tra i 3 e i 31 caratteri. Al momento del salvataggio avviene inoltre una

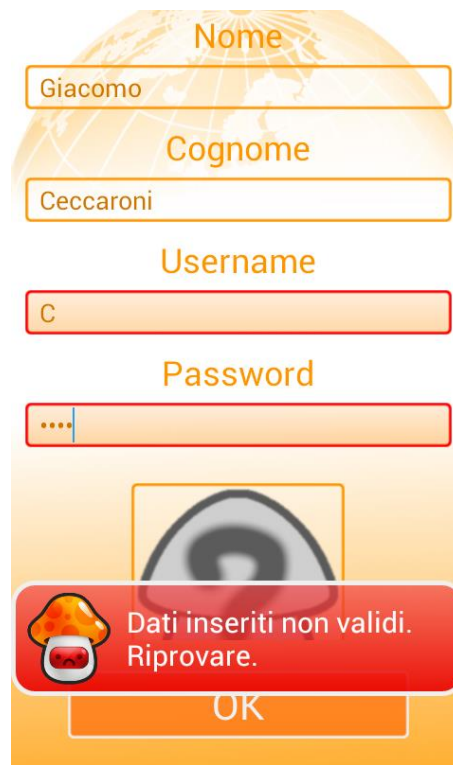
formattazione automatica della stringa: la prima lettera viene convertita in maiuscola mentre tutte le altre vengono forzate a minuscole.

- *Username*: lo username non può essere vuoto e la sua lunghezza deve essere compresa tra 5 e 21. Sono necessari inoltre un altro paio di controlli: dovendo essere questa una stringa univoca nel database, è obbligatorio che al momento del salvataggio avvenga una ricerca su tutti gli utenti del database e che in caso di uguaglianza con uno di essi, la registrazione non possa proseguire.
- *Password*: la password non può essere vuota e deve essere di lunghezza compresa tra i 6 e i 21 caratteri.



The screenshot shows a registration form with five input fields: 'Nome', 'Cognome', 'Username', and 'Password'. Below the fields is a placeholder for a profile picture, represented by a question mark icon. At the bottom is an 'OK' button.

Figura 3.6 – Schermata di registrazione



The screenshot shows the same registration form, but with incorrect data entered: 'Nome' is 'Giacomo', 'Cognome' is 'Ceccaroni', 'Username' is 'c', and 'Password' is '....'. A red error message box at the bottom reads 'Dati inseriti non validi. Riprovare.' with a small icon of a mushroom. The 'OK' button is still visible below the error message.

Figura 3.7 – Tentativo di registrazione inserendo dati non corretti

È inoltre possibile associare al proprio utente un'immagine profilo utilizzando un oggetto di tipo *ImageButton*. Il caricamento può avvenire tramite galleria o fotocamera e non è obbligatorio; in caso di mancato upload dell'immagine profilo, ne verrà assegnata all'utente una di default. L'immagine viene quindi scalata, codificata e inviata al server sotto forma di stringa.



### 3.2.3 MapActivity

Questa *activity* è centrale all'interno dell'app in quanto risulta essere generalmente quella più utilizzata nonché la più importante. Come già dal titolo si può intuire, la pagina contiene la mappa su cui vengono visualizzati gli amici in caso di utente online.

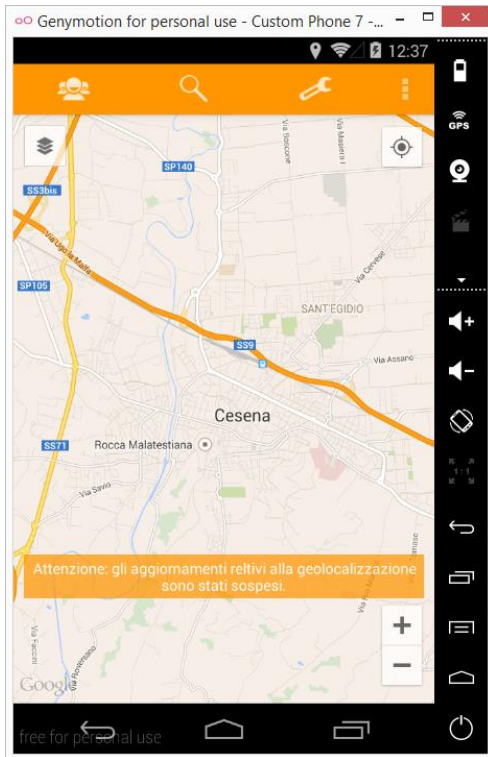


Figura 3.8 – Schermata della mappa offline

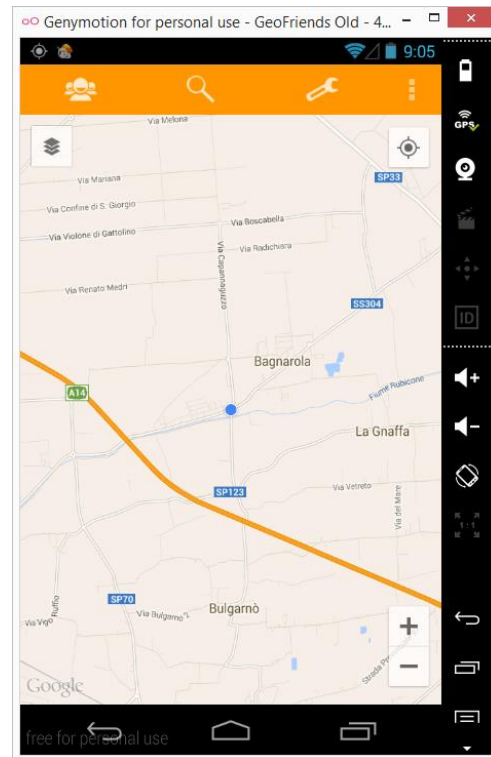


Figura 3.9 – Schermata della mappa online

L'accesso a questa schermata può avvenire in diverse maniere:

- Via login, ovvero subito dopo aver premuto il pulsante “Login” dall’omonima pagina. All’avvenire dell’evento, verrà mostrato all’utente un messaggio di benvenuto contenuto in un *Toast* personalizzato che verrà descritto in seguito.
- Direttamente al tap sull’icona dell’app, nel caso in cui il login fosse già avvenuto in precedenza e la sessione fosse ancora aperta (caso più frequente)
- Al tap sull’icona nella barra nelle notifiche di Android, ovvero il logo di GeoFriends visualizzabile nell’area superiore del dispositivo in caso di sessione online.
- Al tap su un’icona secondaria di GeoFriends nella barra nelle notifiche di Android; incorreremo quest’evenienza qualora ricevessimo una notifica push da parte di un secondo utente,

come ad esempio la richiesta di mostrare la posizione corrente o la segnalazione di un amico passato online.

- Dalla pagina della lista dei contatti: è infatti possibile aprire la mappa centrata sulla posizione di un amico, selezionando quest'ultimo dalla lista tramite il bottone "Visualizza sulla mappa".

All'apertura dell'*activity* La creazione della mappa avviene da zero se la pagina deve essere ricreata da zero, ovvero se non ho alcuna istanza dell'*activity* salvata nello stack.

Una volta viene iniziato un *loop* eseguito ogni tot secondi, all'interno del quale avviene un continuo *refresh* della mappa a condizione che l'utente risulti essere collegato. Quest'ultima circostanza può non verificarsi in caso di logout forzato, il quale avviene se un utente con le medesime credenziali dell'attuale effettua l'accesso da una seconda periferica. Al riconoscimento di questa particolare situazione, l'id salvato all'interno della *device* viene impostato a 0 e all'utilizzatore viene mostrato un messaggio d'allerta che lo costringe a tornare alla schermata di login. Il controllo che l'identificatore dell'utente sia diverso da 0 è presente in numerosi frangenti di codice.

Il metodo che esegue il *refresh* della mappa è richiamato in diverse parti del codice. La prima è, come già detto, all'interno di un *loop* temporizzato che si occupa di far sì che non passi troppo tempo tra una chiamata al server e un'altra. Un altro caso nel quale è necessario eseguire un aggiornamento, è al momento del tap sul messaggio che avverte l'utente di essere offline. Nello specifico, il messaggio può comparire per quattro cause:

- Assenza di connessione a internet.
- GPS non attivato.
- Scelta da parte dell'utente di non ricevere aggiornamenti relativi alla posizione.
- Mancanza di copertura GPS.

Qualora nessuna di queste eventualità si verificasse, il messaggio d'errore verrebbe settato a invisibile e l'utente entrerebbe in modalità online.

All'entrata nella modalità online, viene inviato un messaggio apposito a tutti gli amici collegati, utilizzando le Google Cloud Notification. Allo stesso tempo, l'utente viene aggiunto alla tabella "online" inviando la posizione corrente e il momento del passaggio in linea. Ha così inizio il ciclo di aggiornamento continuo che permette il tracking dell'utente nonché la visualizzazione degli amici in mappa.

L'aggiornamento è suddiviso in due parti: l'invio della propria posizione attuale e la ricezione delle coordinate dei contatti associati.

L'invio della propria posizione è possibile grazie ad una classe di appoggio che fornisce le coordinate dell'utente in un dato momento, in formato numerico. Latitudine e longitudine vengono dunque sfruttate per l'invio al server, ma anche per centrare la mappa sulla posizione corrente all'apertura della stessa dei contatti

Per l'aggiornamento dei dati relativi agli amici, è necessario prima di tutto un *task* atto al download della lista contatti corrente. Una volta ottenuta si passa alla fase di decodifica. Come descritto nel capitolo relativo agli *script* in uso, il server restituisce tre vettori distinti (amici online, amici offline e richieste in attesa) i quali sono a loro volta scomposti al fine di estrarre i singoli campi (tra cui l'immagine profilo, se presente). I dati sono salvati in una lista apposita e inoltre l'array degli amici online è salvato in una lista a parte che chiamo *onlineUsers*. Quest'ultima viene utilizzata per costruire, grazie a latitudine e longitudine, i marker da inserire nella mappa.

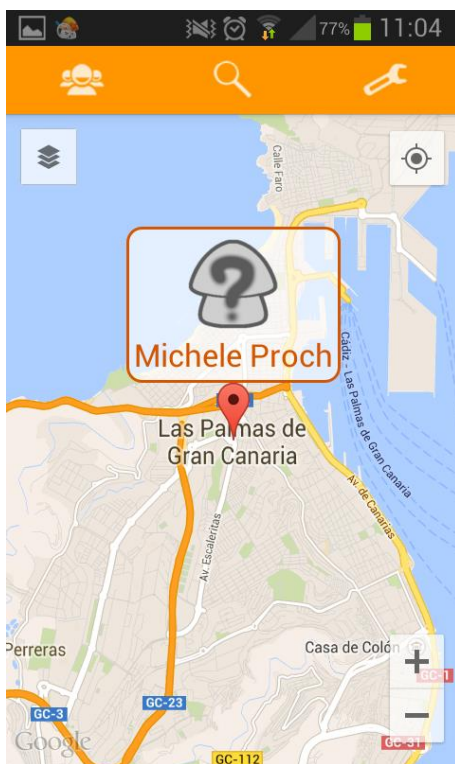


Figura 3.10 – Mappa con marker aperto

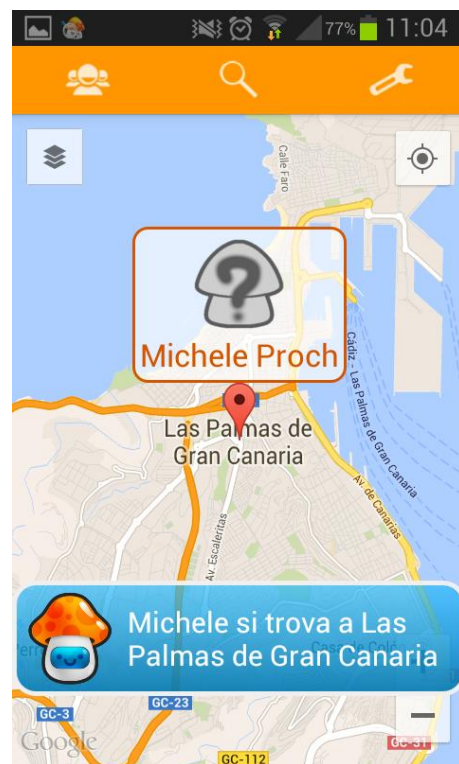


Figura 3.11 – Mappa con marker cliccato

Oltre al *MapFragment*, all'interno della pagina è possibile notare un menu di navigazione posizionato sulla parte alta dello schermo. Esso contiene i collegamenti che permettono di raggiungere la quasi totalità delle altre *activity* che verranno descritte di seguito.

L'ultimo elemento da cui la pagina è composta è il menu a tendina, azionabile grazie alla pressione sul tasto fisico "menu" presente in tutti i dispositivi Android; questo contiene tre voci, ovvero:

- "Aggiorna", che permette appunto di aggiornare manualmente la mappa nonché la propria posizione corrente.
- "Condividi", che apre una *Dialog* la quale permette di scegliere se condividere la propria posizione su Facebook o Twitter. In entrambi i casi, ciò che viene pubblicato è un semplice messaggio inclusivo di nome e cognome dell'utente, città da cui il messaggio viene inviato, link all'app e alla pagina di Google Maps, con in evidenza un marker relativo alla locazione dell'utente.
- "Avanzate", che porta all'*activity* descritta in seguito delle opzioni avanzate.

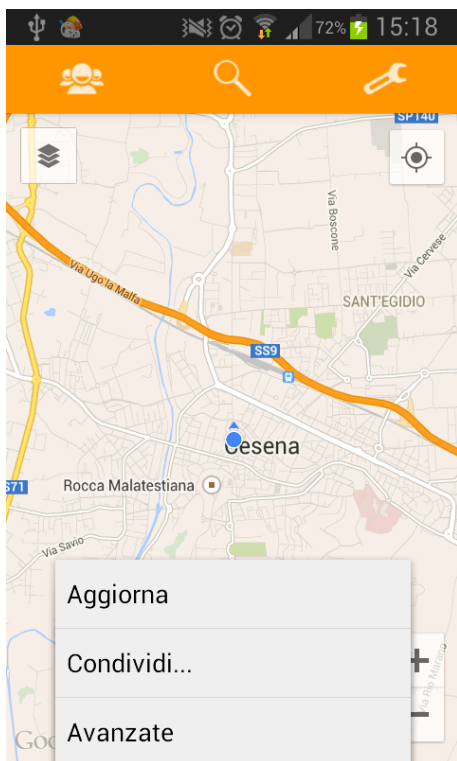


Figura 3.12 – Mappa con menu aperto

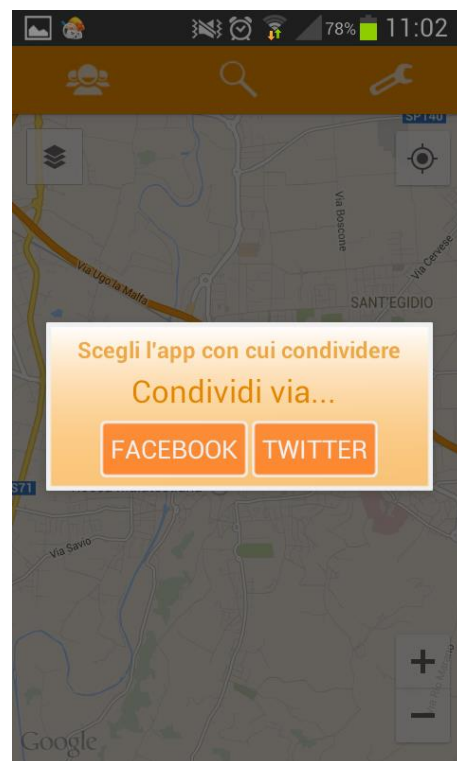


Figura 3.13 – Dialog per permettere la condivisione sui social

All'uscita della pagina è fondamentale eseguire il *detach* (ovvero lo “scollamento”) del frammento che contiene la mappa, al fine di evitare errori. Verrà inoltre stoppato l'aggiornamento della posizione degli amici, ma non l'invio continuo della posizione dell'utente.

### 3.2.4 FriendListActivity

All'interno di questa pagina è possibile visualizzare l'elenco dei propri contatti, in maniera ordinata per tipo (prima quelli online, poi quelli offline e infine le richieste in attesa).

La prima operazione che viene eseguita è dunque quella di lettura della lista. Questa è già presente all'interno dell'app in quanto la sua compilazione era avvenuta al momento dell'apertura della mappa.

I casi possibili di fronte a cui è possibile trovarsi sono due:

- Lista vuota, ovvero non ho ancora aggiunto alcun amico. Per comodità dell'utente viene visualizzata una *Dialog* che con un tap porta alla pagina di ricerca dei contatti in rete.
- Lista con almeno un contatto; questa è l'evenienza più frequente. Tramite un *adapter*, gli utenti facenti parte dell'elenco vengono convertiti in modo da poter essere “attaccati” alla lista ed avere un aspetto più *user friendly*.

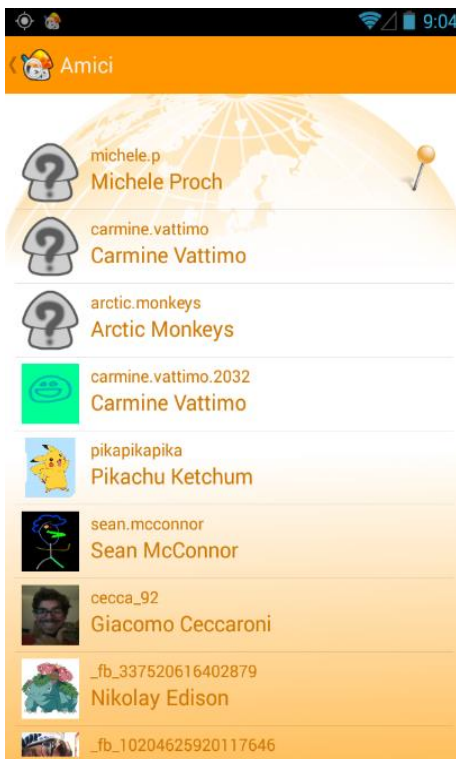


Figura 3.14 – Esempio di lista contatti

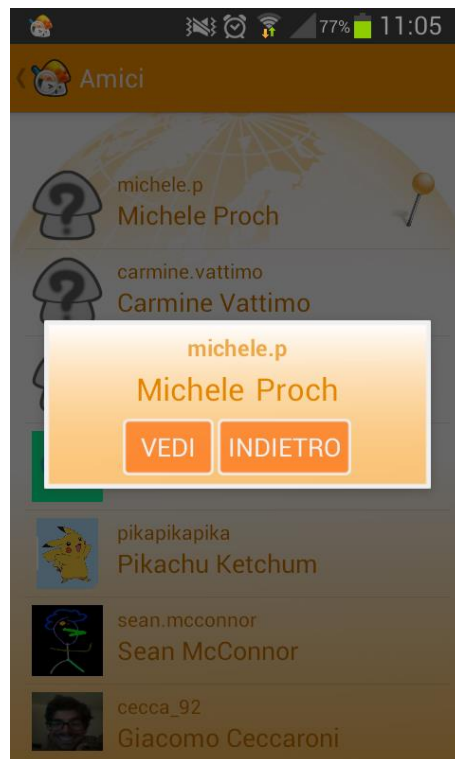


Figura 3.15 – Dialog al click su un contatto

I singoli oggetti contenuti nell'elenco si differenziano nei tre tipi elencati a inizio paragrafo, e ognuno di questi differisce per aspetto e per le azioni effettuabili su di esso.

- Utente online: si riconosce per l'icona sulla destra che indica che un utente è attualmente visualizzabile sulla mappa. Al tap su di esso è dunque possibile visualizzarne la posizione geografica.
- Utente offline: simile al precedente ma senza l'icona sulla destra. Al tap su di esso è possibile inviare un *poke*, ovvero una richiesta di passaggio online che verrà inviata al dispositivo del destinatario utilizzando il servizio di notifica messo a disposizione da Google Cloud Notification.
- Utente in attesa di amicizia: ha un layout grafico differente in fatto di colore e stile delle scritte (grigie e corsive). Al tap su di esso è possibile decidere di accettare o rifiutare la richiesta di amicizia. Nel primo caso, l'utente entrerà a far parte di uno dei due tipi sopra citati mentre nel secondo caso questo potrà in futuro inviare nuovamente una richiesta d'amicizia.

### 3.2.5 SearchActivity

All'interno di questa pagina è possibile cercare un utente tra i contatti esistenti e inviare richieste d'amicizia. Vale la pena notare subito che tramite la barra superiore sono selezionabili due tipi di ricerca: per username o per nome e cognome. Nel primo caso viene visualizzato soltanto un campo di testo mentre nel secondo i campi sono due e devono essere entrambi compilati onde far sì che la ricerca vada a buon fine.

Ciò che viene restituito è in entrambi i casi una lista di utenti corrispondenti ai dati inseriti. La decodifica e il posizionamento dei dati sullo schermo, avviene tramite la stessa tecnica usata nel caso della *FriendListActivity*. Al tap su un'oggetto dell'elenco, viene visualizzata una *Dialog* che permette di inviare una richiesta d'amicizia all'utente selezionato qualora questo non fosse già presente nella lista dei contatti.



Figura 3.16 – Schermata di ricerca per username



Figura 3.17 – Schermata di ricerca per nome e cognome

Nel momento in cui viene inviata una richiesta d’amicizia a un utente, questo riceve una notifica push che, se premuta, porta direttamente alla pagina *FriendListActivity*.

### 3.2.6 *SettingsActivity*

Questa pagina torna utile all’utente nel momento in cui questo desidera cambiare i propri dati personali. Il layout proposto è molto simile a quello visto per la registrazione in quanto anche le funzionalità sono simili a quelle della registrazione. La differenza sostanziale sta nell’obbligo di inserire la vecchia password per poterne poi settare una nuova. I controlli sui campi e la modalità di scelta dell’immagine profilo, sono i medesimi di quelli descritti per la *SignUpActivity*. Per far sì che le modifiche vengano salvate, è necessario semplicemente premere il pulsante fisico “indietro” presente in tutti i dispositivi Android. Alla pressione di tale pulsante vengono eseguiti i controlli già visti durante la descrizione della registrazione e, nel caso di modifica della password, avviene un confronto tra la vecchia password salvata su server e quella immessa nel campo di testo.

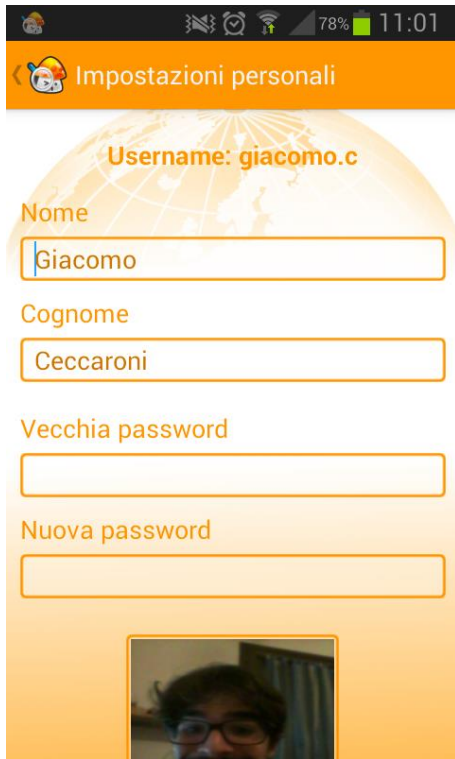


Figura 3.18 – Schermata impostazioni personali



Figura 3.19 – Vecchia password non corretta

### 3.2.7 *AdvancedSettingsActivity*

Questa pagina, aggiunta soltanto in un secondo momento, permette all'utente dell'app un ulteriore livello di personalizzazione. In questo caso le funzionalità su cui è permesso mettere mano sono legate all'app e non all'utente, ovvero è possibile:

- Scegliere se ricevere o meno le notifiche push legate all'app.
- Scegliere se attivare o meno gli aggiornamenti relativi alla geolocalizzazione.
- Modificare l'intervallo di tempo che trascorre tra un aggiornamento e un altro.

Il valore che definisce se ricevere o meno le notifiche push legate all'app, è salvato all'interno delle *shared preferences* in maniera da poterlo così riutilizzare ad una successiva apertura dell'applicazione.



Tramite il pulsante fisico “menù” è inoltre possibile eseguire il logout dall’app, tornando così alla pagina principale di login.

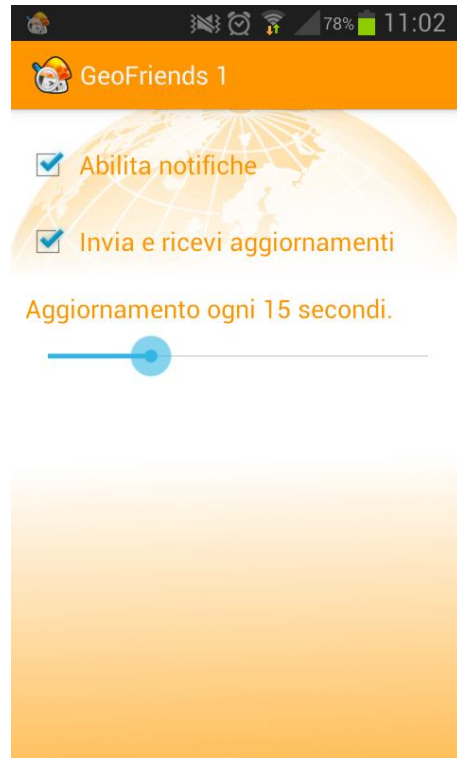


Figura 3.20 – Pagina delle impostazioni avanzate

### 3.2.8 CustomDialogBuilder, CustomNotificationBuilder e CustomToastBuilder

Queste tre classi sono state create al fine di incrementare il livello di personalizzazione dell’interfaccia grafica.

“CustomDialogBuilder” consente di costruire *Dialog* (ovvero finestre di dialogo) con un layout appositamente definito. Ad essa sono collegabili fino a due bottoni con i relativi metodi associati in caso di attivazione.

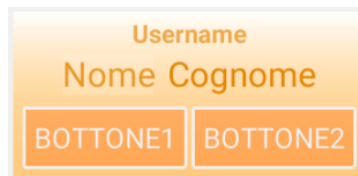


Figura 3.21 – Oggetto di tipo CustomDialogBuilder

“CustomNotificationBuilder” viene utilizzata nel momento in cui è richiesta l’aggiunta di una notifica nella barra delle notifiche di Android. Nella maggior parte dei casi, l’uso di questa classe avviene a partire da una notifica push in arrivo dal server GCM, la quale contiene anche informazioni sul tipo di icona da associare ad essa.

Infine “CustomToastBuilder” consente di costruire *Toast* personalizzati nell’interfaccia, mettendo a disposizione due layout di default: uno per gli eventi di errore e uno per quelli di notifica.

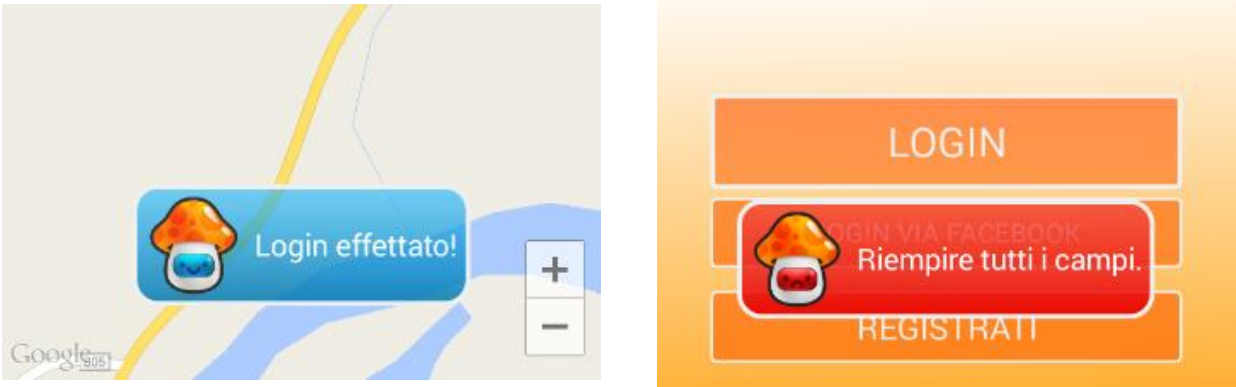


Figura 3.22 – Oggetti di tipo CustomToastBuilder

### 3.2.9 CustomTaskBuilder

Questa classe risulta fondamentale all’interno dell’app, e lo si capisce dalle numerose volte in cui viene richiamata all’interno della stessa. Il ruolo che svolge è quello di far comunicare l’app con il server. In particolare, alla classe vengono forniti in input il nome dello script da richiamare e i parametri da inviare a quest’ultimo. Durante l’esecuzione del comando avviene una chiamata al server in background e l’app si mette in attesa di una stringa di risposta; questa viene poi restituita attraverso un’interfaccia appositamente creata.

“CustomTaskBuilder” copre quasi tutti i casi di comunicazione con il server. L’unica eccezione si ha quando si parla delle immagini di profilo; vista la notevole lunghezza di questi elementi è stato infatti necessario creare due classi a parte che eseguissero appositamente il compito di caricamento e scaricamento dei .PNG codificati in stringhe base 64. La parte di codifica e decodifica avviene attraverso un’altra classe di appoggio.

### 3.2.10 *GCMIntentService*

Questa classe esegue principalmente la “decodifica” dei messaggi in arrivo dal servizio Google Cloud Notification, impostando la corretta visualizzazione di questi ultimi (icona e messaggio) e l’azione idonea da eseguire nel momento della pressione da parte dell’utente sulle relative notifiche push che vengono generate.

### 3.2.11 *GPSTracker*

Come si può intuire facilmente dal nome, questa classe svolge le funzione di localizzazione dell’utente. Nello specifico, all’attivazione di un oggetto “GPSTracker” viene fatto partire un servizio che in background esegue l’invio delle coordinate dell’utente. Queste sono determinate grazie a servizi di localizzazione messi a disposizione da Android, i quali agiscono sfruttando il GPS o la rete internet.

Quando avviene lo stop del servizio “GPSTracker”, gli aggiornamenti della posizione vengono fermati e l’icona di notifica relativa all’attività dell’app viene rimossa.

### 3.2.12 *User e UserAdapter*

Questa classe è stata creata allo scopo di contenere i dati degli utenti. I campi che contiene sono molto simili a quelli della tabella “user” contenuta nel database. La maggior parte dei metodi presenti sono *getter* o *setter*, i quali permettono di visualizzare o valorizzare i parametri dell’utente. È importante sottolineare che questa classe non è utilizzata per salvare i dati dell’utente correntemente loggato, ma quelli dei contatti esterni (come, ad esempio, gli amici).

A dare vita a “User”, vi è una classe cosiddetta *adapter*, la quale trasforma i campi dell’utente in un oggetto grafico utilizzabile per essere associato, ad esempio, a una lista.

### 3.2.13 *UsersMapFragment*

Questa classe costruisce il *fragment* atto al contenimento della mappa fornita da Google. All’interno sono presenti tutte le funzioni di cui la mappa necessita per essere visualizzata correttamente nonché modificata.

È ad esempio implementato il metodo che prende in input le coordinate e le utilizza per centrare la mappa sui tali punti x e y, attraverso un movimento lineare della “telecamera”.

Un altro metodo di non poca rilevanza, è quello che disegna i marker degli amici sulla mappa nelle coordinate apposite, per poi gestire il tap su di essi definendo l'azione da svolgere e il corretto popup da visualizzare.

Un'oggetto di tipo "UsersMapFragment", proprio come i normali *fragment* di Android, ha bisogno di essere aggiunto e rimosso dinamicamente ad ogni apertura o chiusura della pagina che lo contiene e ad ogni aggiornamento dello stesso.

### 3.2.14 Classi di appoggio

All'interno dell'applicativo sono presenti diverse classe con il suffisso "Utils". Queste sono le cosiddette classi di appoggio, ovvero atte a fornire quasi esclusivamente metodi e campi statici.

- "BitmapUtils": contiene le funzioni di codifica e decodifica in stringa delle immagini profilo, nonché le chiamate ai metodi di download e upload di quest'ultima.
- "GCMUtils": contiene le funzioni di registrazione e di invio messaggi al server.
- "IntentUtils": contiene i numeri interi identificativi dei diversi *intent* interni all'app.
- "MapUtils": contiene informazioni relative alla mappa come, ad esempio, l'intervallo di tempo tra un aggiornamento e l'altro. È inoltre implementata una funzione di creazione di una stringa con l'indirizzo corrente.
- "UserUtils": contiene le informazioni relative all'utente corrente. La quasi totalità dei metodi sono *getter* o *setter* e la maggior parte di questi, utilizzano anche le *shared preferences* al fine di mantenere i dati relativi all'utente anche una volta usciti dall'app.
- "Utils": contiene informazioni generiche, come può ad esempio essere l'indirizzo dell'*host* in utilizzo.

## 3.3 Grafica

Per modificare l'aspetto di un elemento in Android è sufficiente modificare le relative proprietà all'interno del file .xml contenente la descrizione della struttura dell'elemento. Ciò può essere però poco conveniente nel caso di proprietà ripetute come, ad esempio per le caselle di testo in una pagina di registrazione.

Per questo all'interno dell'app è stato fatto utilizzo di stili e temi, ovvero delle risorse atte a contenere proprietà facilmente applicabili agli elementi o all'intera applicazione scrivendo semplicemente il riferimento al nome dello stile.



Figura 3.23 - Esempio di tema applicato su ActionBar

Il *drawable* (letteralmente: “qualcosa che possa essere disegnato”) è stato un altro elemento fortemente utilizzato durante lo sviluppo dell'app, sia perché consente di utilizzare immagini in maniera semplice e intuitiva, sia perché permette di creare forme di diversa fattura, con bordi e colori di sfondo personalizzabili.

L'utilizzo dei *drawable* risulta particolarmente evidente durante l'utilizzo dell'app, in particolare all'apertura di una *Dialog* o di un *Toast* personalizzato.



## 4 CONCLUSIONE

---

Il settore delle app che sfruttano i servizi di localizzazione offerti dai dispositivi mobili è stracolmo; ogni giorno vengono pubblicate centinaia di app ed è pertanto certo che esistessero già software per dispositivi mobili in grado di svolgere le stesse funzioni presenti in GeoFriends.

In questa tesi si è cercato di sviluppare un applicativo che fornisse le conoscenze per poter essere in grado di costruire da zero un'app che utilizzi servizi web e che comunichi con un server via internet.

Questa esperienza mi ha permesso di crescere dal punto di vista professionale in quanto scrivere così tanto codice ha incrementato le mie conoscenze in ambito di programmazione. Inoltre nella prima parte della tesi è stato fondamentale il lavoro di gruppo, fattore che si rivela estremamente utile durante lo sviluppo di software complesso.

Lavorare con Android è consigliabile; grazie anche alle numerose pagine web di documentazione e forum riguardanti le problematiche affrontate, è stato infatti facile imbattersi in soluzioni che, se riadattate, sarebbero potute tornare utili. L'ambiente offerto dal pacchetto ADK Bundle è stato facile da installare e in generale non ha causato un gran numero di problemi, salvo qualcuno in fase di aggiornamento da una versione all'altra.

Purtroppo al momento l'app risulta essere non molto performante in fatto di tempistiche; le richieste al server rallentano molto l'esecuzione e su questo aspetto c'è ancora molto da lavorare. La problematica è emersa nel momento del passaggio da web server emulato in locale a web server online.

Anche la scalabilità, può essere decisiva per un buon prodotto di questo tipo. In fase d'implementazione e progettazione si è deciso di sviluppare un codice flessibile e aperto a nuovi sviluppi. Il database contiene tabelle e campi poco utilizzati che possono essere valorizzati diversamente in maniera da garantire una maggiore utilità. Inoltre la pagina delle impostazioni è attualmente piuttosto scarna ed è quindi possibile aggiungere nuove funzionalità al suo interno nel corso del tempo. I test effettuati su diversi dispositivi con diverse risoluzioni hanno mostrato come lo stesso codice sia eseguito perfettamente da piattaforme diverse e, con una buona programmazione, i contenuti mantengano lo stesso aspetto garantendo un'esperienza utente solida.

## 4.1 Sviluppi futuri

L'applicazione permette numerosi update e modifiche sia per quanto riguarda le prestazioni e la stabilità, sia per quanto riguarda l'integrabilità con i nuovi dispositivi. Durante una prima fase di testing nel "mondo reale", sono infatti apparsi evidenti due punti cruciali che deteriorano la qualità dell'app: le latenze in caso di connessione internet deficitaria e l'elevato consumo di batteria dovuto alla necessità di aggiornare perennemente la posizione (e mantenere quindi un contatto periodico col server).

Tra i punti da sviluppare vi è sicuramente l'utilizzo delle informazioni relative all'orario di aggiornamento della posizione. In particolare, questo dato può essere utile al fine di mostrare quello che ormai tutti i software legati al social network chiamano "ultimo accesso", mostrando a quale orario risale l'ultimo invio delle coordinate al server da parte del dispositivo dell'utente. La visualizzazione dovrebbe avvenire nel momento in cui l'utente preme su di un marker, all'interno del popup che compare.

Un altro punto su cui si può intervenire in futuro, è il numero al momento ristretto di notifiche push che un utente può ricevere. Potrebbe tornare utile per un utilizzatore dell'app, ad esempio, sapere quando un amico è vicino a lui entro un certo raggio.

Un'altra funzionalità implementabile potrebbe essere la possibilità per gli utenti di impostare marker sulla mappa in maniera da definire punti di ritrovo visualizzabili dagli amici. Sarebbe quindi necessario definire un ulteriore layout al fine di garantire l'inserimento della descrizione dell'evento e, magari, una foto descrittiva di tale.

Un'ultima osservazione fatta durante la fase di testing dell'applicativo; l'immagine profilo presenta diverse limitazioni. A fotocamera ruotata di 90 gradi infatti, la foto che viene scattata viene impostata senza alcun accorgimento e sarebbe quindi opportuno lavorare su questo piccolo bug per riportarla nella posizione originaria. L'immagine profilo durante il salvataggio viene inoltre scalata automaticamente in modo da renderla quadrata; sarebbe più corretto se in fase di selezione, venisse data all'utente la possibilità di ritagliarla in modo da evitare deformazioni in fase poi di salvataggio sul database.

Al momento non sono proprietario di uno spazio sul Play Store, ma qualora lo diventassi senz'altro i requisiti minimi da correggere prima di una pubblicazione sarebbero quelli elencati all'interno di questo paragrafo.



## 5 SITOGRAFIA

---

- <http://stackoverflow.com>
- <http://www.focus.it/>
- <http://www.lastampa.it/>
- <http://therealtime.com/>
- <http://mediacenter.runtastic.com/>
- <http://it.wikipedia.org/>
- <http://www.androidhive.info>
- <http://www.tuxjournal.net/>
- <http://developer.android.com/>
- <http://www.fastweb.it/>
- <http://www.nielsen.com/>