Bertoni Tommaso
Cadorin Andrea
Sist Matteo
Zeni Marco

Progetto GeoAd

Documento di progetto – Client

github repository

Tecnologie

Il Client è sviluppato nativamente per la piattaforma Android ed è compatibile da Android 4.1 (API 16) in su, che attualmente corrisponde a circa il 90% dei devices in circolazione.

Activities

Le activity del progetto sono:

- MainActivity: é l'activity principale dell'applicazione.
- AugmentedRealityActivity: dedita alla gestione della realtà aumentata.
- NewOfferActivity: viene visualizzata quando si clicca su una push notification ricevuta, e ne visualizza il dettaglio.
- RegisterActivity: viene visualizzata quando un gestore di una attività vuole registrarsi in modo da poter gestire le proprie attività commerciali

Database locale

- SQLite attraverso l'uso di Content Provider, le tabelle sono le seguenti:
- Tabella offerte: mantiene memorizzate nel device tutte le offerte che vengono ricevute dall'applicazione. Vengono rimosse automaticamente al momento della scadenza. La tabella viene mantenuta aggiornata con delle push di sincronizzazione per modifica/cancellazione delle offerte da parte del proprietario della relativa attività commerciale.
- Tabella mie attività commerciali: contiene tutte le attività di cui l'utente loggato è il proprietario. Al momento del login richiedo al server le attività presenti sul server di cui sono proprietario.
- Tabella luoghi ignorati: memorizza il nome e l'id dei luoghi ignorati per visualizzarli nella lista “ignorati”.
- Tabella luoghi preferiti: memorizza tutti i dati dei luoghi preferiti (ad eccezione di offerte e immagini) per consentirne la visualizzazione anche in stato di offline.

Servizio principale GeoAd

Il servizio principale della nostra applicazione ha le seguenti caratteristiche:

- Viene avviato automaticamente all'avvio del device
- Si occupa di mantenere vivo il servizio di geolocalizzazione, registrandosi come suo ascoltatore e ponendosi nello stato di foreground per non essere terminato dal sistema operativo
- Contiene in memoria i luoghi “vicini” all'utente. Sono i luoghi che verranno visualizzati nel modulo di realtà aumentata e i luoghi del quale l'utente può ricevere le notifiche delle offerte. Il location manager è impostato per inviare aggiornamenti della posizione solo se l'utente si sta effettivamente muovendo. Il servizio, ogni volta che riceve un aggiornamento, verifica che la distanza da una precedente posizione sia oltre una certa soglia. Solo in quel caso richiede l'aggiornamento al server dei luoghi vicini. Ciò consente una precisione sufficientemente buona per l'utilizzo della realtà aumentata, ma tenta di contenere le chiamate tra client e server per risparmio di batteria e traffico dati.
- Invia l'intent per aggiornare il widget delle offerte.
- Gestisce i “comandi” ricevuti tramite push dal server per tenere aggiornate la tabella delle offerte e la lista dei luoghi vicini.
- Gestisce la comparsa delle notifiche per le nuove offerte dei luoghi vicini o preferiti . Se l'applicazione non ha ancora rilevato la posizione dell'utente l'offerta viene messa da parte fino alla localizzazione dell'utente (per poter verificare sia effettivamente relativa ad un luogo vicino o preferito). Cliccando sulla notifica apro una schermata di visualizzazione della stessa.
- Si occupa di recuperare in modo asincrono i dati di un luogo se vengono richiesti (se non sono in RAM vengono effettuate la query al Content Provider ed eventualmente la richiesta HTTP)

Engine (application)

- Si occupa della gestione delle operazioni sulle tabelle del content provider relative ai luoghi e delle relative notifiche al server. Notifica lo stato di “ignorato” al server per impedire al server di inviare notifiche push relative ad un luogo e viceversa notifica lo stato di “preferito” per forzare l'invio delle push delle offerte anche per luoghi “lontani” dall'utente.
- Avvia il servizio GeoAd
- Si occupa di controllare che l'applicazione sia registrata per la ricezione delle notifiche push e mantiene memorizzata la chiave (la chiave delle notifiche server per identificare un device lato server, anche se questo non è un utente loggato)
- In caso di reset dei dati dell'applicazione segnala al server di cancellare tutte le preferenze del device relative ai luoghi.
- Contiene l'oggetto che fa da cache per le immagini. Vengono memorizzate tutte le miniature e le immagini delle location per migliorare le performance e ridurre il traffico HTTP. Ad ogni riavvio dell'application la cache risulta inevitabilmente azzerata.
- Legge da file l'indirizzo del server web: quest'implementazione è stata utilizzata per non rendere pubblico su Github l'indirizzo del server.

ConnectionManager

- Esegue tutte le chiamate HTTP dell'applicazione nei vari metodi http GET, POST, PUT e DELETE
- Mantiene monitorato lo stato della connessione evitando che vengano inviate richieste se connessione assente (risparmio batteria)
- Inserisce automaticamente come header della chiamata il token di autorizzazione di OAuth se l'utente è loggato
- Inserisce automaticamente la chiave delle push come parametro della chiamata per rendere identificabile il device

Widget

- Mostra tutte le offerte memorizzate nella tabella del content provider. Cliccando sull'offerta viene aperta una schermata di riepilogo della stessa.
- Visualizza un tasto per accedere alla schermata di gestione delle attività dell'utente (solo se l'utente è loggato)
- Dimensione minima 4x2, ridimensionabile da parte dell'utente.
- La frequenza di aggiornamento del widget è impostata a 30 minuti, ma essendo effettuati gli aggiornamenti forzati quando necessario potrebbe essere innalzata per risparmio di batteria.

Realtà aumentata

- Mostra in realtà aumentata i punti di interesse e le attività nelle vicinanze.
- Viene usata la libreria BeyondAR.
- Viene utilizzata una Camera view sulla quale vengono disegnati, in accordo alla posizione dei punti relativa al telefono, i punti di interesse e le location.
- location e punti di interesse hanno icone diverse e nel caso vi siano offerte in una location viene visualizzata una stellina di fianco.
- Cliccando su un punto si apre un pannello con il riepilogo del punto di interesse, con nome, distanza e descrizione. Nel pannello è presente anche un pulsante che carica il dettaglio della location ed uno per chiudere il pannello.

Fragments:

SearchListFragment:

- Questo fragment viene caricato all'avvio dell'applicazione e mostra, di default, la lista di location in un raggio di 8 km dalla posizione dell'utente.

- Nella action bar sono disponibili i pulsanti per switchare tra fragment mappa (SearchMapFragment) e fragment lista e per inserire filtri di ricerca.
- Premendo su filtro si aprirà un dialog che permette all'utente di filtrare I risultati secondi diversi campi quali categoria, distanza, nome e tipo (attività commerciale o punto di interesse), premendo infine su filtra la lista di location o la mappa verranno aggiornati adeguatamente ai filtri.
- Se si sta visualizzando il fragment con la mappa, un cerchio intorno alla posizione attuale indicherà il raggio di ricerca delle location, un marker per ogni location verrà aggiunto alla mappa (colorato diversamente in base al tipo di location), al click sul marker si aprirà un pannello di riepilogo e, cliccando sul pannello si procede al dettaglio della location.

DetailFragment:

- Fragment che contiene tutte le informazioni su una specifica location: nome luogo, categoria principale e secondaria (se c'è), tipo (punto di interesse o attività commerciale) una serie di immagini in miniatura con eventuale scroll orizzontale, descrizione e lista di offerte attuali.
- Nella action bar sono presenti le opzioni per visualizzare la location su google maps e per inserire la location tra i preferiti o gli ignorati
- Le foto vengono presentate in una lista orizzontale che permette lo scrolling. premendo sulle miniature si aprirà un dialog nel quale verrà scaricata dal server la foto a grandezza reale. Le miniature vengono scaricate dal server all'avvio del fragment e sia le miniature che le foto intere vengano cachate in memoria.
- La lista di offerte presenta di default solo il nome dell'offerta e al click la view si espande e mostra la descrizione, la data di scadenza e i pulsanti per la condivisione su Facebook e la condivisione su altri social.
- La condivisione attraverso i social (escluso facebook al quale abbiamo dato maggior importanza) permette di spedire un messaggio contenente tutti i dati dell'offerta attraverso vari servizi (Whatsapp, Gmail, Hangout etc)
- Condivisione su Facebook: Abbiamo utilizzato la libreria di Facebook in modo da dare all'utente un modo semplice per condividere un'offerta. Utilizzando la libreria abbiamo la possibilità di creare un messaggio personalizzato collegato alla nostra applicazione GeoAd su Facebook, l'utente dovrà solamente premere condividi. Il messaggio in questione è composto dal nome e descrizione dell'offerta e cliccando sul post si seguirà un link al nostro portale dove si possono vedere ulteriori dettagli.

EditLocationFragment:

- Questo fragment viene caricato quando un utente loggato va al dettaglio di una location che gli appartiene.
- Contiene le stesse informazioni del dettaglio (usa lo stesso layout).
- Le differenze riguardano le opzioni nella actionbar e la possibilità di rimuovere foto e offerte, in questo caso le opzioni sono aggiungi foto, inserisci nuova offerta, modifica nome, modifica descrizione.
- Su aggiungi foto verrà aperto un dialog che permette di scegliere tra scattare una foto attraverso la fotocamera oppure selezionare un'immagine dalla galleria. Una volta effettuata l'operazione

desiderata la foto verrà ridimensionata (in modo da non mandare foto eccessivamente pesanti) e poi mandata al server che ritornerà come risposta la thumbnail corrispondente che verrà inserita dinamicamente nella lista di miniature.

- Su inserisci nuova offerta si aprirà un dialog con i campi di testo "nome" e "descrizione" e un picker per la data di scadenza. una volta confermato, l'offerta verrà mandata al server che si occuperà di mandare la push ai device.
- Su modifica nome e modifica descrizione si aprirà un dialog con un campo di testo dove è possibile modificare i valori correnti.
- Tenendo premuto a lungo su una miniatura si aprirà un dialog che chiederà conferma per l'eliminazione della foto, alla conferma verrà inviata al server la richiesta di cancellazione della foto corrispondente e verra tolta la miniatura dalla pagina di dettaglio.
- Stessa situazione prememendo a lungo su un'offerta

MarkedLocationFragment:

- Fragment che contiene un ExpandableListView contenente i luoghi preferiti e i luoghi ignorati impostati dall'utente
- La prima volta che viene aperto il fragment compare un dialog di aiuto che spiega all'utente cosa significa impostare i luoghi in una modalità piuttosto che un altra. Questo dialog comparirà all'avvio del fragment fino a che l'utente non metterà la spunta sulla checkbox per impedirne la comparsa
- La lista è popolata leggendo il contenuto dal DB locale
- Un click lungo su un elemento fa comparire un dialog che chiede se rimuovere il luogo dalla lista in cui si trova. In caso di risposta affermativa il luogo scompare dalla lista, viene rimosso dal DB e viene notificata l'azione al server. Piccolo bug: la cancellazione dal DB avviene solo in caso di successo di notifica al server. Quella dalla lista avviene immediatamente e ciò potrebbe ovviamente portare a situazioni non coerenti tra quanto visualizzato e quanto memorizzato lato server.
- Solo per quanto riguarda i preferiti un click sull'elemento fa spostare la navigazione sul fragment di dettaglio, dopo aver raccolto i dati necessari da mostrare dal DB locale

MyLocationFragment

- Fragment che visualizza tutte le attività commerciali di cui l'utente loggato è proprietario.
- Durante l'onAttach viene chiamato il metodo checkAuth della MainActivity che verifica che il token dell'utente sia ancora valido, visto che sono necessari i permessi per cancellare o modificare le attività.
- La lista è popolata tramite un cursorAdapter
- In maniera molto simile al MarkedLocationFragment sono gestiti il click e il long click per accedere al fragment di modifica e al dialog di cancellazione.
- E' compito della MainActivity decidere se lanciare il fragment di edit o di modifica a seconda che l'utente stia aprendo un suo luogo o quello di qualcun altro. Per evitare di appesantire le operazioni dovendo effettuare ogni volta query per scoprire se sono il proprietario di un'attività commerciale, quest'operazione avviene alla creazione dell'Engine e viene salvata una lista degli ID delle mie attività commerciali.

Bug noti

- L'eventuale assenza di connessione non è notificata correttamente all'utente tramite GUI.
- Non è segnalato e nemmeno gestito il caso di geolocalizzazione spenta o non avvenuta. Si dovrebbe far comparire all'utente un dialog che lo invita ad attivarla oppure notificar un messaggio di errore se non è possibile localizzare l'utente entro un ipotetico timeout da definire.