ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea in Informatica

Progettazione ed analisi dei requisiti di un applicativo mobile per la gestione di messaggi geolocalizzati

Tesi di Laurea in Basi di dati

Relatore:
Chiar.mo Prof.
Danilo Montesi

Presentata da:
Marco Corradini

Sessione I
2011/2012

Indice

1 Introduzione

Proprio come predetto dalla legge di Moore nel lontano 1965, negli ultimi quattro decenni le tecnologie informatiche sono state interessate da un processo di sviluppo esponenziale, così come sono aumentati esponenzialmente gli investimenti per realizzare apparecchiature sempre più complesse e sofisticate.

Dall'uscita dei primi Personal Computer alla diffusione di Notebook, Netbook e Tablet Pc, la tecnologia è diventata il motore informativo ed economico che più ha caratterizzato la società moderna degli ultimi anni.

Parallelamente, la diffusione capillare di Internet ha permesso ed agevolato la comunicazione con milioni di soggetti a distanza, oltre che facilitato l'accesso ad informazioni in precedenza difficilmente consultabili.

Più recentemente l'evoluzione dei telefoni cellulari, diventati veri e propri mini computer che permettono di navigare in rete e rimanere sempre in contatto con amici, colleghi o fonti di informazione, ha radicalmente modificato il rapporto tra uomo e macchina, elevando queste ultime da mere attrezzature lavorative ad elementi indispensabili per la vita sociale ed intellettuale delle persone.

Tutto ciò ha portato a nuovi modelli sociali di utilizzo delle tecnologie medesime, basati sullo scambio di informazioni, sulla comunicazione istantanea, sulla necessità di essere sempre aggiornati e sempre connessi a quel mondo informatico che permette come mai prima l'espressione libera delle proprie idee e la condivisione dei propri pensieri.

Questi nuovi modelli sociali hanno fatto impennare i consumi di applicazioni basate su queste innovazioni, popolando il web di molteplici servizi per lo sharing di dati, piattaforme di pubblicazioni online gratuite, blog e social network come Facebook e Twitter, che ormai contano centinaia di milioni di iscritti in tutto il mondo. Oggi come oggi è possibile trovare sul web applicativi che svolgono praticamente ogni tipo di funzione. La rappresentazione di semplici informazioni statiche non basta più, il trend attuale spinge infatti sempre più verso servizi interattivi per la gestione di dati, lo

scambio di file multimediali come foto, video e musica, l'e-commerce, il gaming online e la creazione di documenti che possono poi essere condivisi al resto del mondo.

Fatte queste considerazioni, questa tesi ha l'obbiettivo di descrivere la progettazione e l'implementazione di un'applicazione che permetta la condivisione di messaggi testuali. Questi messaggi non saranno indirizzati a singoli destinatari o esposti in una "lista" accessibile agli amici, come avviene nei più famosi social network, ma saranno legati ad una posizione geografica scelta dall'utente e visualizzabili tramite una mappa geografica. Una sorta di Post-it virtuali, da cui il nome ***GeoPost-it***

Nel secondo capitolo descriverò le linee guida dell'idea in sé, come è stata concepita e in quale dominio applicativo andrà a collocarsi. Nel terzo e quarto mi riferirò ai concetti generali del ciclo di vita standard del software, parlando prima degli obbiettivi e dell'analisi dei requisiti e poi della progettazione, concludendo infine questo ciclo con la descrizione dell'implementazione nel capitolo cinque. Il sesto capitolo sarà dedicato alla descrizione delle tecnologie utilizzate in questo processo di creazione. Nel capitolo sette presenterò alcuni sviluppi futuri possibili per l'applicazione, arrivando poi a trarre le conclusioni nel capitolo otto.

2 L'idea

2.1 Un approccio diverso

Come discusso nell'introduzione, un mare di nuove applicazioni sociali stanno letteralmente invadendo il panorama tecnologico. A che scopo quindi sviluppare un'idea basata sullo scambio di informazioni se sono già presenti decine e decine di software che fanno più o meno la stessa cosa? Analizzando i più famosi social network presenti su Internet come Facebook, Twitter e Foursquare, notiamo come non sia sempre facile trovare informazioni utili riguardo ad un particolare luogo o un'attività. Inoltre, spesso la quantità di dati presenti confondono l'utente che si limita a leggere quello che gli viene messo davanti, senza andare a ricercare informazioni specifiche che descrivono particolari, magari interessanti, ma non molto popolari.

Il fine di questi social network è infatti la comunicazione tra persone che si conoscono o che hanno interessi in comune. Questo tipo di comunicazione non è pensato per gli appunti personali scritti a se stessi, per le annotazioni temporanee o per le note collegate ad un particolare luogo. Per permettere lo scambio di informazioni tra sconosciuti, gli applicativi che si basano su questo tipo di comunicazione necessitano di una sottoscrizione ad una particolare fonte che deve essere prima ricercata e non viene presentata in modo automatico all'avvicinarsi fisicamente a qualcosa che la riguarda.

Su Facebook è per esempio possibile creare pagine che descrivono luoghi, attività e persone. Attraverso queste pagine gli utenti possono ricevere aggiornamenti e notizie riguardanti i soggetti interessati. Questo tipo di pagine risultano però quasi sempre piene di commenti non inerenti all'attività descritta, contengono informazioni non aggiornate, non sono curate o risultano inefficaci per via della troppa pubblicità che producono. Anche i semplici post lasciati dagli utenti riguardo particolari località ed eventi legati ad esse non sono sempre funzionali, infatti non possono essere letti se non dagli amici e finiscono per non essere presi in considerazione o addirittura dimenticati, sepolti da una valanga di informazioni in continuo divenire.

Foursquare, d'altra parte, consente agli utenti di scrivere opinioni e consigli legati ad attività commerciali, abitazioni, edifici storici e quant'altro. Queste informazioni sono sicuramente utili per chi cerca una particolare attività nei suoi dintorni, ma se l'utente cercasse informazioni generiche sulla zona? O se volesse lasciare informazioni riguardanti una determinata posizione non classificabile come risorsa (come una deviazione, una buca o un'area particolare) oppure informazioni che riguardano anche lui stesso, e non solo la zona geografica in sé? Altro aspetto da considerare è che l'utente, anche solo per condividere notizie generiche su un posto, dovrebbe prima creare tramite l'applicativo di Foursquare il luogo d'interesse e solo successivamente potrebbe annotare le informazioni legate ad esso.
Questo potrebbe essere un procedimento un po' troppo complesso per una semplice nota informativa, magari non legata ad una particolare attività commerciale o che abbia una valenza temporale limitata.

Manca quindi un mezzo che permetta in modo semplice ed immediato di scrivere annotazioni che descrivano una posizione geografica. Per ovviare al problema, ho pensato a quale potesse essere il mezzo più semplice per lasciare messaggi facilmente visibili alle altre persone: il Post-it.
I Post-it sono foglietti di carta colorata semi-adesiva che vengono usati nei luoghi di lavoro per annotazioni e piccoli appunti, facilmente visibili tra i fogli bianchi delle scrivanie.
Così ho cercato di tradurre questo oggetto in forma virtuale, estendendo il campo di utilizzo dall'ufficio/camera all'intera superficie globale sulla quale poter appuntare note di qualsiasi tipo. Per fare questo, è stato necessario trovare un mezzo di utilizzo che fosse portatile, connesso ad Internet e capace di gestire coordinate geografiche da incorporare nelle note.
Ho deciso di riferirmi all'iPhone come dispositivo per lo sviluppo di questa idea. Questo dispositivo infatti presenta tutte le caratteristiche richieste ed è stato già al centro dei miei studi durante il periodo di tirocinio svolto presso l'Apple Development Center dell'Università di Bologna.
Nel paragrafo successivo andrò ad esporre i dettagli dell'idea.

2.2 GeoPost-it

GeoPost-it è un'applicazione ideata per essere sviluppata su iPhone e realizzata a seguito dell'esperienza di stage presso l'Apple Development Center Unibo, laboratorio nato dalla collaborazione fra Apple e Alma Mater Studiorum Bologna. Il progetto prevede anche la possibilità di accedere al servizio tramite un'interfaccia web che presenti le stesse funzionalità dell'applicazione mobile. Il tutto è stato sviluppato sotto la supervisione del docente Danilo Montesi, mentre l'applicativo per iPhone è stato realizzato con la collaborazione del mio collega Paolo Ladisa.

Scopo di questa applicazione è di condividere informazioni legate alla loro posizione geografica tramite la funzione di geolocalizzazione[1]. Questo servizio permette l'identificazione della posizione di un dato oggetto, in questo caso di un telefono cellulare o un computer connesso a internet, secondo coordinate geografiche (latitudine e longitudine).

L'utente potrà scrivere messaggi di lunghezza limitata (fino ad un massimo di 300 caratteri) che, attraverso la geolocalizzazione, saranno vincolati a coordinate geografiche che permetteranno di legare il messaggio ad un luogo preciso.

Le note lasciate dagli utenti saranno visibili a tutti i fruitori del servizio ed avranno una durata temporale limitata e decisa al momento della loro creazione. Al termine di questo periodo saranno eliminate per non creare un eccesso di dati superati e ormai non più utili. L'utente in movimento potrà, tramite una mappa, visualizzare tutti i Post-it lasciati nelle vicinanze dagli altri che utilizzano il servizio.

Inoltre i messaggi saranno anche accessibili attraverso internet tramite una pagina web, dove una comoda interfaccia permetterà, oltre che a leggere i post già inseriti tramite l'applicazione, di scriverne di nuovi e "appuntarli" su una mappa proprio come se fossero vere e proprie note geografiche.

[1] Il funzionamento della geolocalizzazione sarà descritto in dettaglio nel paragrafo 6.1

Questa applicazione, pensata come semplice mezzo per scrivere appunti geolocalizzati, può in realtà espandersi a numerosi altri utilizzi: dal turista che cerca informazioni sui luoghi da visitare, alla madre che lascia un messaggio al figlio per ricordargli di chiudere la porta di casa. Dal commerciante che lascia un messaggio per pubblicizzare la sua attività, al ragazzo che lascia un messaggio davanti alla porta della fidanzata.

3 Analisi dei requisiti

3.1 Obbiettivi

L'applicativo presentato avrà il compito di fornire un servizio per la condivisione di messaggi legati a coordinate geografiche di latitudine e longitudine. Queste note saranno poi visualizzate su una mappa geografica consultabile tramite un applicativo per iPhone ed una pagina web. Il servizio dovrà essere accessibile senza restrizioni a tutti quelli che vorranno utilizzarlo. L'unica prerogativa richiesta all'utente sarà infatti la registrazione al sistema, necessaria per identificare la provenienza dei messaggi e fornire un maggior livello di personalizzazione.
Il sistema dovrà inoltre fornire funzioni per la ricerca di post tramite vocaboli contenuti in essi e la possibilità da parte dell'utente di salvare posizioni geografiche come preferite, per velocizzare il processo di annotazione di nuovi messaggi. L'utente dovrà anche avere la possibilità cancellare i messaggi da lui precedentemente inseriti e di modificarne la posizione.
L'inserimento, la cancellazione e la modifica della posizione delle note dovranno essere possibili attraverso una semplice interazione con la mappa, senza la necessità di utilizzare menu o campi di inserimento testuale. La mappa dovrà quindi essere il centro di quasi tutte le funzionalità e perciò verrà presentata in modo da essere ben visibile e facilmente controllabile con funzioni di zoom e di modifica della posizione. All'avvio, questa mostrerà la posizione attuale dell'utente ed i relativi messaggi presenti nelle sue vicinanze, ma dovrà presentare la possibilità di ritrovare e mostrare nuovamente la sua posizione nel caso in cui l'utente si sia spostato tra un utilizzo e l'altro.
Infine, le note inserite dovranno avere una data di scadenza definita dall'utente, passata la quale il sistema provvederà all'eliminazione automatica del messaggio.

3.2 Requisiti necessari

Il sistema dovrà essere progettato per un utilizzo in movimento, quindi necessiterà di una connessione ad internet per scaricare i dati relativi alle mappe e ad i messaggi degli utenti. Questi ultimi dovranno essere conservati all'interno di un database, che verrà consultato attraverso l'interfaccia dell'applicazione. Questo database sarà memorizzato all'interno di un server web, un computer sempre collegato alla rete e dotato di software specifici per fornire dati e pagine su richiesta dell'utente (client). Una volta trasmesse, queste informazioni potranno essere visualizzate dall'utente attraverso un browser o utilizzate all'interno di altre applicazioni.
Il servizio di hosting[2] sarà l'unica spesa necessaria al completo funzionamento del servizio, oltre all'eventuale acquisto di un dominio[3] per il sito web.
L'efficenza del sistema dovrà essere garantita da uno scambio di informazioni minimo tra client e server. Saranno quindi richiesti al server solo i messaggi più vicini alla posizione dell'utente. Mentre gli altri messaggi saranno scaricati solo su specifica istanza dell'utente.
Il servizio dovrà anche garantire la sicurezza dei dati degli utilizzatori, proteggendo le loro password con un sistema di crittografia prima che queste siano inviate al server.
Il supporto sul quale verrà utilizzato il servizio dovrà inoltre fornire un sistema di geolocalizzazione, per permettere in modo veloce la ricerca e la scrittura di messaggi nell'area in cui si trova l'utente.

---

[2] Il servizio di hosting consiste nell'allocazione di risorse come pagine HTML, programmi e database su di un server web. Tale servizio è spesso gestito da aziende e difficilmente realizzabile in privato, vista la necessità di strumentazioni costose e di una connessione che garantisca l'accesso alle risorse tramite la rete.

[3] Il dominio è il nome utilizzato per identificare un sito web, quello che si digita all'interno di un browser per accedere ad una pagina e che verrà poi trasformato dalle infrastrutture della rete nell'indirizzo fisico dove si trovano le risorse corrispondenti.

4 Progettazione

4.1 Database

Per la memorizzazione dei dati inseriti dagli utenti è stato utilizzato un database, ossia un archivio per la registrazione delle informazioni che si fonda sul modello relazionale.

Il principio base del modello relazionale è che tutte le informazioni siano rappresentate da valori inseriti in relazioni (tabelle); dunque un database relazionale è un insieme di tabelle contenenti valori, ed il risultato di qualunque interrogazione sui dati può essere rappresentato anch'esso da ulteriori tabelle (relazioni).

La fase di progettazione di un database comprende la stesura di un diagramma denominato E-R, da “entità-relazione”.

Questo modello concettuale fornisce una serie di strutture allo scopo di descrivere la realtà interessata in una maniera facile da comprendere e suddividere il problema per facilitarne la sua risoluzione. I costrutti principali di questo modello sono le entità, le relazioni e gli attributi.

Le entità rappresentano classi di oggetti come cose e persone, che hanno proprietà comuni ed un’esistenza autonoma. Nel mio schema ho definito tre entità: gli utenti (fruitori del servizio), i luoghi preferiti (località salvate per poter essere riutilizzate e che rappresentano i luoghi più frequentati) ed i post (i messaggi). Le entità sono rappresentate nello schema con dei rettangoli.

Le relazioni (o associazioni) rappresentano legami significativi tra due o più entità, come l’azione della scrittura del messaggio da parte dell’utente, o la gestione dei suoi luoghi preferiti. Queste associazioni sono mostrate come dei collegamenti tra entità e sono rappresentate da rombi. Le entità possono avere un diverso numero di occorrenze all’interno di una relazione. Per descrivere questo comportamento si utilizza la cardinalità, riportata come due valori separati da virgola, che rappresentano il minimo ed il massimo numero di partecipazioni di un’entità alla relazione. Per esempio, un

utente può scrivere da zero ad n post, mentre un post può essere scritto da uno ed un solo utente.

Gli attributi descrivono le proprietà elementari di entità o relazioni che interessano l'applicazione, come il nome e la password di un utente o le coordinate geografiche di un messaggio. Questi sono rappresentati come dei pallini legati alle entità. Quelli pieni rappresentano gli identificatori interni (chiavi che permettono di identificare univocamente le tuple delle relazioni).

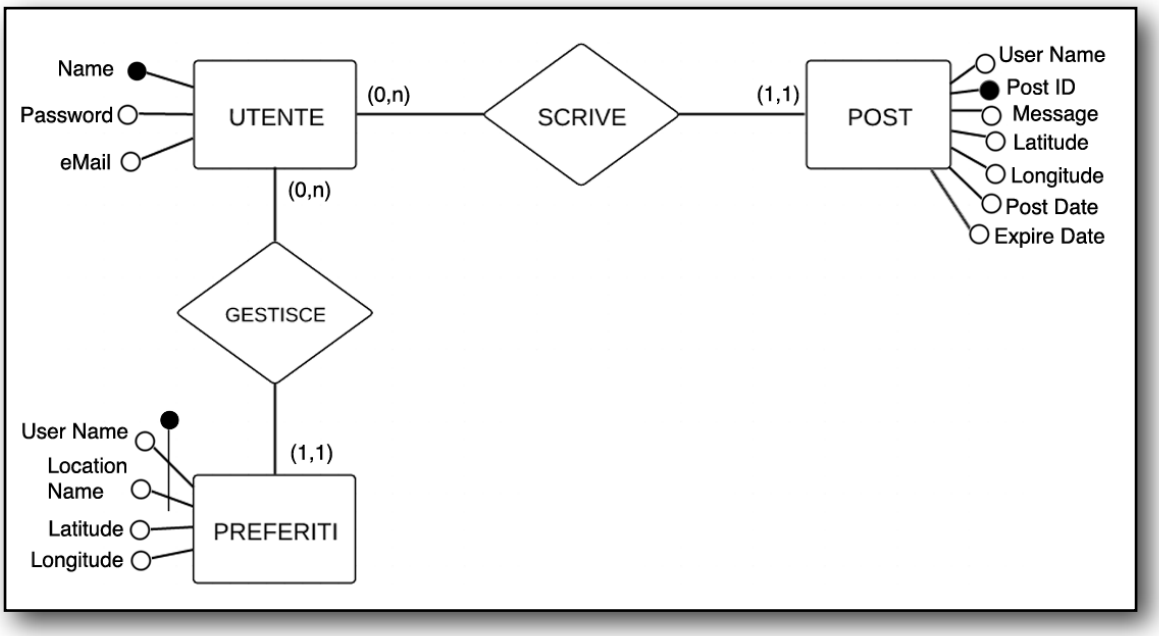

Fig. 1: Schema E-R relativo al database dell'applicazione.

Lo schema ER illustrato si traduce poi in uno schema della base di dati, che descrive le relazioni vere e proprie memorizzate nel database. Troveremo quindi le tre relazioni principali che saranno le tabelle ed i relativi attributi scritti tra parentesi che saranno le colonne.

```
USERS(Name, Password, Mail)
POST(UserName, Message, Latitude, Longitude, Id, PostDate, ExpireDate)
FAVOURITES(Name, LocationName, Latitude, Longitude)
```

Nel database formato avremo quindi come già detto tre tabelle contenenti tutti i dati fondamentali per l'applicazione.

Riassumendo, per gli utenti saranno registrati tre campi: il nome (chiave), la password e l'email. I preferiti invece conterranno, oltre alle coordinate geografiche, il nome dell'utente e della località che andranno a formare la chiave primaria della relazione, identificando univocamente in questo modo ogni riga della tabella. I post infine necessiteranno di una struttura leggermente più complessa, con alcuni campi aggiuntivi. Oltre al nome dell'utente, il messaggio e le coordinate, sarà presente un identificativo numerico univoco (chiave), la data di creazione del post e quella di scadenza. Compito del database sarà infatti anche quello di eliminare tutti i post che hanno superato tale data, diventando quindi obsoleti.

Occorre infine precisare che il database utilizzato sarà lo stesso per ogni tipo di utilizzo del servizio. Infatti sia l'applicativo (o gli applicativi in caso di sviluppi futuri su ulteriori piattaforme) che la pagina web accederanno alla stessa istanza dei dati. Il metodo di interrogazione del database ed il tipo di incapsulamento dei dati richiesti saranno diversi e dovranno essere gestiti a parte per permettere una maggiore flessibilità nella programmazione dei vari applicativi, ma i dati scambiati saranno gli stessi.
Questo permetterà la condivisione delle medesime informazioni attraverso diversi mezzi. Tutti potranno leggere i messaggi scritti da altri utenti e lasciarne a loro volta, permettendone la consultazione a tutti anche attraverso diversi veicoli di comunicazione.

4.2 Interfaccia web ed applicazione

Lo scambio di dati tra il database ed i vari applicativi può essere riassunto brevemente tramite la Fig. 2. Come illustrato, nella prima fase avviene la localizzazione del dispositivo tramite la funzione di geolocalizzazione. Nella figura, questa è stata rappresentata come richiesta alla rete globale di posizionamento (GPS) attraverso il chip contenuto nell'iPhone. Nel caso si utilizzassero altri dispositivi o si accedesse al servizio tramite web, questo metodo potrebbe essere diverso e, in generale, dipendere dal tipo di strumento utilizzato, dal browser e dalla connessione a cui è collegato. Tratterò in dettaglio questi metodi nel paragrafo uno del capitolo sei.

Una volta ottenuta la posizione dell'utente, il software provvederà ad inviare queste informazioni al server attraverso una connessione HTTP[4]. Il server elaborerà la richiesta attraverso alcune pagine PHP, ricercando nel database tutti i post presenti nelle vicinanze dell'utente e rispedendo i dati al mittente, formattati con XML o JSON per permettere la loro elaborazione all'applicazione o al sito[5].

Anche nel caso di inserimento di nuovi messaggi o di ricerca di messaggi già inseriti lo schema rimarrà lo stesso. Il dispositivo invierà le richieste al server che le tradurrà in interrogazioni per il database e le rispedirà all'applicativo che si occuperà di presentarli in maniera adeguata all'utente. L'unica differenza nelle comunicazioni tra server ed applicativo sarà quindi nel diverso metodo utilizzato per la restituzione dei dati. Questo, come accennato nel paragrafo precedente, perché iOs[6] gestisce meglio i dati in formato JSON mentre per le comunicazioni con pagine web si preferisce l'utilizzo dell'XML in virtù della sua semplicità e flessibilità.

---

[4] HTTP, acronimo di HyperText Transfer Protocol, è un protocollo di trasferimento dati usato come principale sistema di trasmissione d'informazione sul web.

[5] PHP, XML e JSON saranno illustrati nel capitolo sei.

[6] iOs è un sistema operativo sviluppato da Apple per iPhone, iPad ed iPod touch.

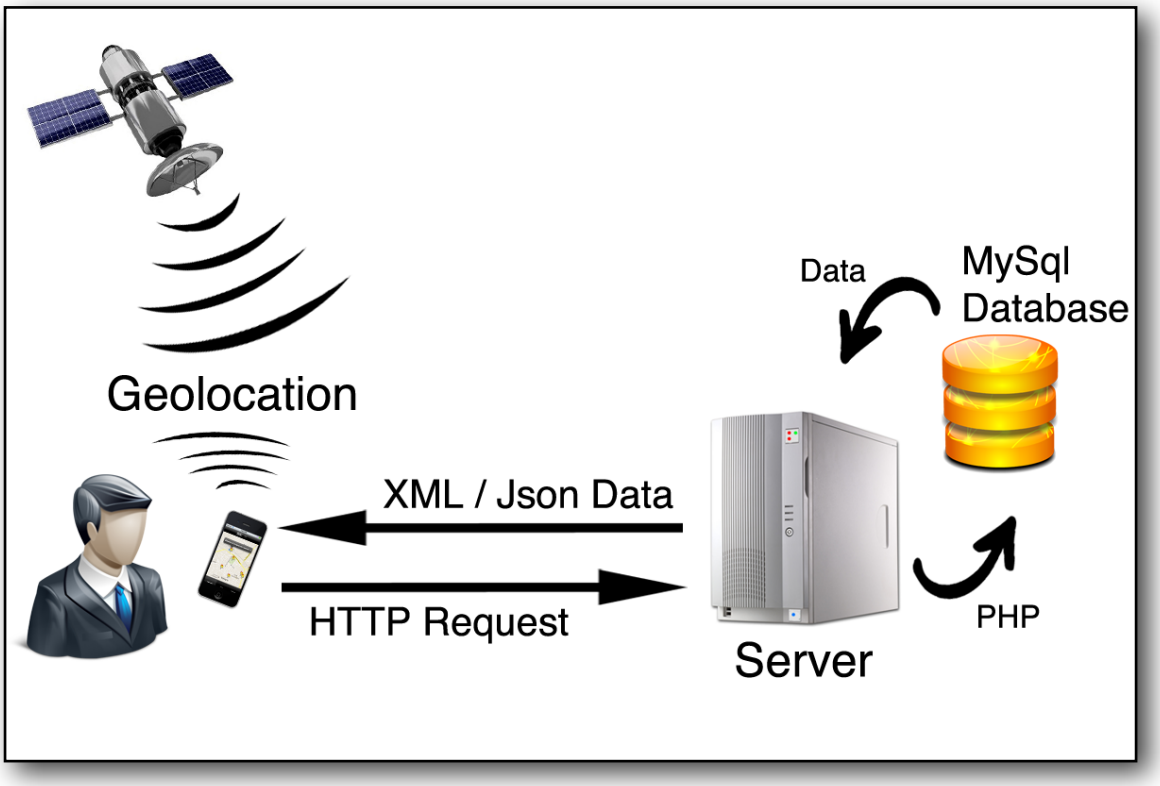

Fig. 2: Schema rappresentativo dello scambio di dati.

Per quanto riguarda l'interfaccia presentata all'utente, attraverso la quale impartire tutti i comandi per la gestione delle varie funzioni, tutto dovrà essere presentato nel modo più semplice e minimale possibile.

Partendo dalla registrazione al servizio, non dovrà essere mostrata all'utente la classica schermata che richiede la compilazione di form per l'adesione al servizio, ma saranno sufficienti l'inserimento di un nome e di una password per essere registrati.

Così anche l'inserimento dei messaggi condivisi dovrà essere pensato in modo da risultare facile come scrivere un Post-it cartaceo. Le uniche cose richieste saranno la posizione del post e la sua data di scadenza, oltre ovviamente al contenuto del messaggio.

Come già descritto dall'analisi dei requisiti, la mappa utilizzata per la visualizzazione delle note si troverà al centro di tutte le funzioni principali. Sarà attraverso quella che gli utenti potranno decidere dove inserire un nuovo post, quali cancellare e dove spostare quelli precedentemente inseriti. Questo renderà la consultazione e la modifica dei messaggi un'operazione realizzabile in modo veloce ed efficace.

La facilità d'uso e l'immediatezza sono infatti, a mio parere, requisiti fondamentali da tenere ben in considerazione nella fase di analisi e progettazione di un applicativo. Requisiti che risultano difficili da mantenere all'aumentare delle funzioni offerte dall'applicazione.

Spesso infatti l'implementazione di una quantità sempre maggiore di servizi comporta un degrado nella presentazione dell'interfaccia, che si appesantisce di innumerevoli pulsanti ed opzioni varie. Queste, seppur indispensabili per il controllo delle funzioni aggiuntive, finiscono per gravare sull'architettura grafica e rendono il prodotto finale non sempre di facile utilizzo, soprattutto per l'utente occasionale.

Fatte queste considerazioni, l'applicazione dovrà quindi essere presentata nel modo più semplice possibile, tralasciando le funzionalità non necessarie per rendere l'utilizzo immediato e rendere la funzionalità principale di scrittura di messaggi intuitiva e veloce.

5 Implementazione

5.1 Server e Database MySQL

Il server impiegato per l'hosting del sito web si basa sulla piattaforma Apache, il servizio più utilizzato al mondo nei server web. La stessa macchina è stata utilizzata per la memorizzazione del database, creato e amministrato attraverso MySQL. MySQL è un sistema per la gestione dei database relazionali, basato sulla sintassi SQL[7], scelto per la sua amplia diffusione, la sua praticità d'uso e la sua gratuità.
Qui sotto sono riportati i comandi SQL impartiti per la creazione delle tre tabelle principali utilizzate nel servizio.

Tabella degli utenti:

```
CREATE TABLE Users (
  Name varchar(20) primary key,
  Password varchar(32),
  Mail varchar(20)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

[7] SQL, acronimo di Structured Query Language, è un linguaggio di interrogazione per database nato nel 1974 nei laboratori IBM ed evolutosi fino ai giorni nostri. Progettato per leggere, modificare e gestire dati memorizzati in un sistema di gestione di basi di dati basato sul modello relazionale (RDBMS), viene utilizzato anche per creare e modificare schemi di database e gestire strumenti di controllo ed accesso ai dati.

Tabella dei Post:

```
CREATE TABLE Post (
  UserName varchar(20) references Users(Name),
  Message varchar(300),
  Lat double,
  Lon double,
  PostId int(11) primary key,
  ExpireDate datetime,
  PostDate datetime
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

Tabella dei preferiti:

```
CREATE TABLE Fav (
  UserName varchar(20) references Users(Name),
  LocationName varchar(30),
  Lat double,
  Lon double,
  primary key(UserName, LocationName)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

In generale, per valori alfanumerici come nomi, password e messaggi sono state usate variabili letterali (*varchar*) che possono quindi contenere stringhe di caratteri numerici ed alfabetici. Per i valori di latitudine e longitudine invece sono state impiegate variabili di tipo *double*, ovvero numeri in virgola mobile a doppia precisione indicati per rappresentare le lunghe cifre con la virgola che caratterizzano le coordinate. Per le date del calendario è stato scelto il tipo di dato *datetime*, formato specifico di MySQL utilizzato per immagazzinare date ed orari (nel formato YYYY-MM-DD HH:MM:SS). Ciò è stato studiato in modo tale che, in futuro, sia possibile aggiungere un orario alla scadenza per garantire una maggiore praticità e precisione.

Infine, come visibile dal codice, la chiave primaria di User è usata come chiave esterna nelle altre due tabelle. Le altre chiavi primarie sono il codice identificativo del messaggio (tabella delle note) e la coppia di valori nome utente e nome località (tabella dei preferiti).

5.2 Sito Web

5.2.1 Presentazione dell'interfaccia

Il sito web è stato realizzato all'interno di una singola pagina web, divisa in due sezioni. La prima riguarda l'area per il login e la registrazione al servizio mentre la seconda contiene la sezione principale con la mappa ed il menu per la gestione dei messaggi.
Se l'utente non è registrato o non ha precedentemente effettuato il login, al momento dell'accesso al sito gli verrà presentata la prima sezione, mostrata nella figura 3.

Fig. 3: Schermata di login dell'interfaccia web.

Nel box presentato, l'utente potrà effettuare il login o registrarsi inserendo nome e password negli appositi campi. In caso di errori nell'inserimento delle credenziali, un messaggio informerà l'utente del tipo di errore in cui è incorso.

Da notare che per la registrazione sono sufficienti il nome e la password, come precedentemente richiesto dal requisito di semplicità ed immediatezza.

Una volta effettuato il login, all'utente verrà mostrata la pagina principale con la mappa ed il menu, come mostrato nella figura 4.

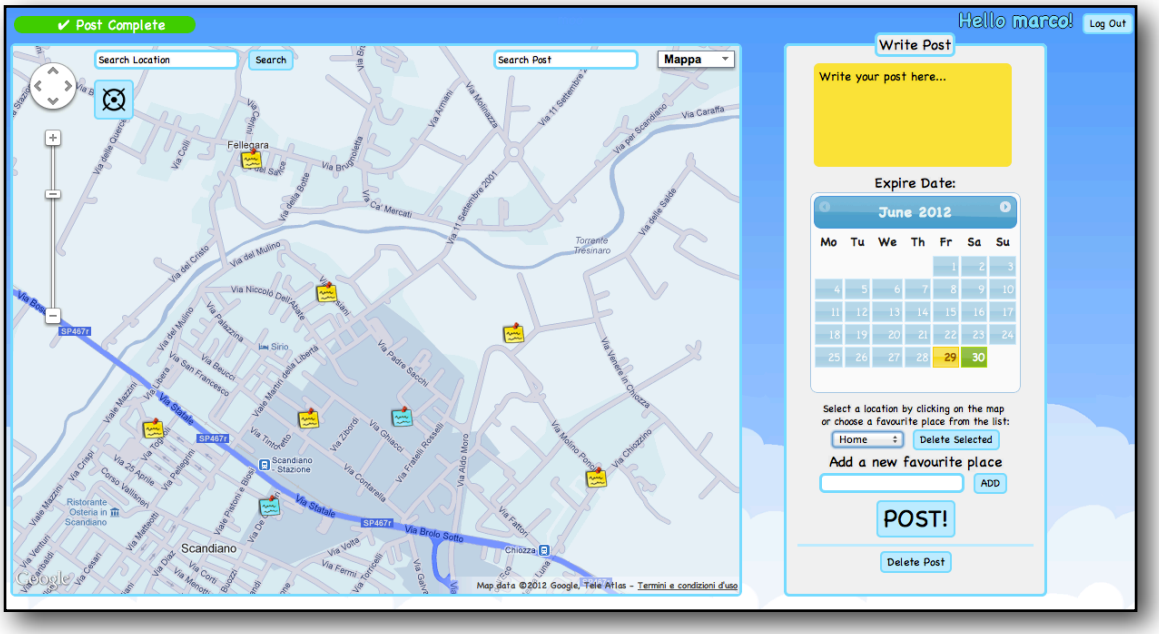

Fig. 4: Schermata principale dell'interfaccia web.

A sinistra della schermata principale troviamo la mappa per la visualizzazione delle note con i relativi comandi (figura 5). Sopra di essa è stato lasciato lo spazio necessario per mostrare il campo dove verranno esposti i messaggi di conferma (in verde) o di errore (in rosso) per l'utente. A destra vi è la sezione per gestire i post (figura 7), mentre nell'angolo in alto troviamo il nome dell'utente ed un pulsante per effettuare il logout e ritornare quindi alla schermata di accesso mostrata precedentemente.

Attraverso il sistema di geolocalizzazione del browser, al momento dell'accesso la mappa sarà automaticamente centrata sulla posizione dell'utente, visualizzando così tutti i post presenti nelle sue vicinanze.

Fig. 5: Mappa dei post-it.

La mappa presenta i classici controlli per lo zoom e lo spostamento. In più, in alto a destra, sono stati aggiunti un campo per la ricerca di luoghi ed un pulsante per la geolocalizzazione. Il primo centra automaticamente la mappa sulla destinazione ricercata, mentre il secondo permette di trovare il luogo in cui si trova attualmente l'utente nel caso abbia cambiato località tra un utilizzo e l'altro del servizio.

Le note sono mostrate attraverso delle icone a forma di piccoli post-it gialli. Cliccando su uno di essi si aprirà un riquadro che ne mostrerà il contenuto. L'autore di un post potrà modificare la sua posizione semplicemente trascinando la relativa icona in una diversa zona della mappa. Selezionando con il mouse un punto sulla mappa, apparirà una puntina per indicare la posizione in cui verrà aggiunto il post che si vuole inserire.

In alto a sinistra troviamo infine un campo per la ricerca di messaggi tramite parole presenti nel loro contenuto (figura 6).

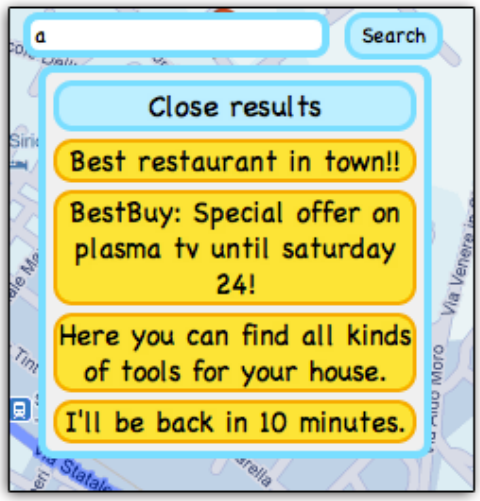

Fig. 6: Campo per la ricerca dei messaggi.

Inserendo lettere e parole nel form per la ricerca, comparirà un menu a cascata con tutti i post che contengono le parole ricercate. Selezionando un messaggio tra quelli presenti, la mappa si centrerà automaticamente su di esso mostrandone la posizione.

Passando al menu principale mostrato nella figura 7, come prima cosa notiamo il campo di testo per la scrittura dei messaggi, appositamente colorato di giallo per ricordare un vero post-it. Sotto al campo di testo per le note trova posto un calendario usato per impostare la data di scadenza, che evidenzia il giorno attuale e permette di scegliere solo date successive a quello.

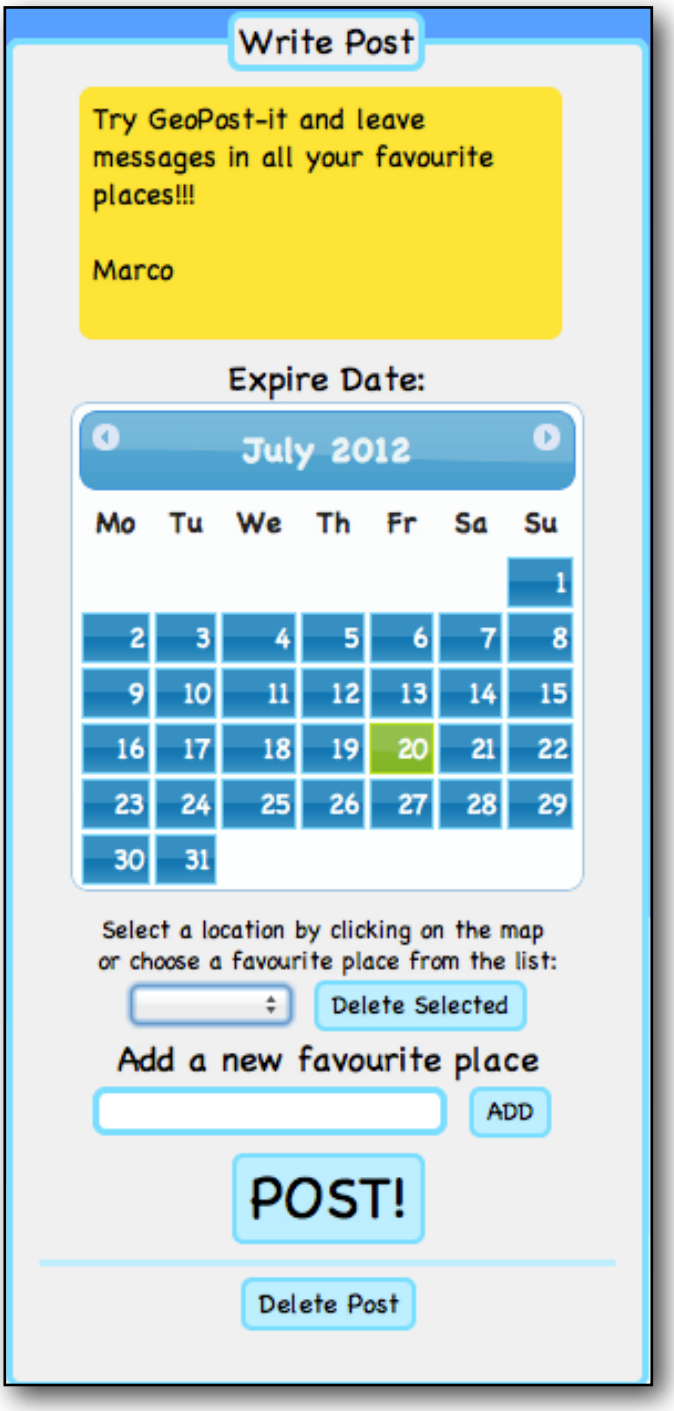

Fig. 7: Menu con i comandi per l'inserimento e la cancellazione dei post-it.

La sezione successiva è dedicata ai luoghi preferiti dell'utente (figura 8). Cliccando sulla mappa comparirà una puntina e inserendo un nome nell'apposito form, sarà possibile salvare quel luogo come preferito. Una volta salvati, questi preferiti potranno essere selezionati tramite una lista a scomparsa ed utilizzati come posizione per scrivere nuovi messaggi. Nel caso di errore nell'inserimento (o per altri motivi) un preferito potrà essere eliminato con l'apposito pulsante.

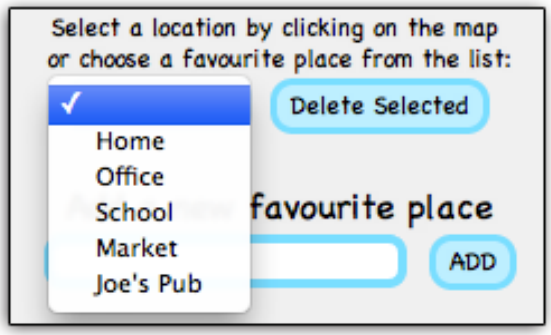

Fig. 8: Sezione del menu dedicata alla gestione dei luoghi preferiti dell'utente.

Una volta scritto il messaggio, scelta la posizione sulla mappa e la data di scadenza, sarà possibile salvare il messaggio cliccando sul pulsante "**POST!**". La pagina, dopo aver effettuato i controlli di correttezza dei dati inseriti, provvederà ad inviare le informazioni al server cosicché vengano aggiunte al database. Fatto questo, il nuovo post comparirà istantaneamente sulla mappa insieme agli altri.
Infine troviamo il comando per cancellare i messaggi posto sul fondo del menù. Una volta premuto, questo pulsante diventerà rosso ed una scritta sopra alla mappa indicherà all'utente di selezionare il messaggio da cancellare. Una volta cliccato, se l'autore del messaggio coincide con l'utente attuale, il messaggio sarà eliminato e tolto dalla mappa.

5.2.2 Realizzazione

La mappa presente nel sito è ricavata da Google Maps e viene gestita attraverso le API[8] per JavaScript[9] fornite da Google, così come in JavaScript sono stati scritti i comandi per l'interazione con il sito. Per facilitare e snellire la programmazione è stato impiegato il framework[10] jQuery.

La mappa è creata come istanza della classe **google.maps.Map**, a cui sono applicati i metodi forniti per le sue funzionalità. I controlli presenti sulla mappa sono quelli standard, mentre lo stile grafico è stato leggermente modificato per differenziarlo dai numerosi siti che offrono servizi legati a questo tipo di mappe. Le icone per i post sono create come istanze della classe **google.maps.Marker** e vengono gestite con l'ausilio di un vettore dove vengono memorizzate. Al caricamento della pagina, attraverso una chiamata AJAX[9], vengono mandate al server le coordinate degli angoli della mappa. Questo risponde a sua volta inviando al sito tutti i post compresi in quelle coordinate. Ogni post viene inserito nella mappa come oggetto della classe sopra indicata, gli vengono inserite le informazioni in esso contenute e gli viene legato un event listener[11] che permetterà alla pagina di mostrare il contenuto del post quando l'utente cliccherà su di esso. Altri listener collegati ai Marker serviranno per la sua eliminazione e per salvare le nuove coordinate in caso che venga spostato (*dragged*).

---

[8] In informatica, con il termine Application Programming Interface (Interfaccia di Programmazione di un'Applicazione) o API si indica un insieme di funzioni e procedure rese disponibili ai programmatori per svolgere determinati compiti all'interno di un programma a scopo di ottenere un'astrazione tra i livelli più bassi dell'architettura dell'elaboratore e quelli più alti. Chi realizza un programma rende spesso disponibili, attraverso questo tipo di interfaccia, le procedure da usare per gestirne il funzionamento attraverso altri software.

[9] Si veda il capitolo 6 per la descrizione delle tecnologie e dei linguaggi utilizzati nella realizzazione del sito.

[10] In ambito di programmazione, un framework è una struttura di supporto su cui un software può essere organizzato e progettato. Alla base di un framework c'è sempre una serie di librerie di codice utilizzabili, spesso corredate da una serie di strumenti di supporto alla creazione del software ideati per aumentare la velocità di sviluppo del prodotto finito.

[11] Il codice JavaScript all'interno del browser è detto *event driven*, cioè risponde alle interazioni che l'utente ha con la pagina (come il click del mouse o la pressione di un tasto) producendo determinati eventi. Questi vengono usati dal programma per definire quali azioni compiere al momento del verificarsi di determinati eventi.

Ulteriori listener sono legati invece alla mappa e permettono l'aggiornamento dei post visualizzati. I post vengono infatti caricati e aggiornati sulla mappa ogni qualvolta la mappa avrà finito di caricare tutte le sue immagini e sarà quindi lanciato il relativo evento (*tilesloaded*).

Come per la richiesta di post da visualizzare, ogni altra comunicazione con il server è stata gestita in modo asincrono attraverso metodi AJAX su una connessione HTTP. Per le operazioni di inserimento, cancellazione e modifica della posizione dei messaggi, di login e registrazione e per l'aggiunta o l'eliminazione dei preferiti, la pagina invia al server valori che vengono utilizzati dagli script PHP presenti su di esso. Questi script fungono da interfaccia per l'accesso al database MySQL e, attraverso i valori ricevuti, inseriscono nuovi elementi nella base di dati, modificano quelli già presenti o effettuano dei controlli sui valori memorizzati, rispondendo alla pagina web con semplici conferme o messaggi di errore.
Invece nelle operazioni di ricerca, di creazione della lista dei favoriti e di inserimento dei post nella mappa la risposta del server è più complessa. Infatti con i dati ricevuti il server interroga il database ed incapsula in formato XML i valori ottenuti per poi rimandarli al sito. Attraverso gli script presenti nella pagina, la struttura dell'XML viene quindi analizzata per estrarne tutti gli elementi, usati poi per creare gli oggetti della pagina o impostarne gli attributi.

Per gestire la geolocalizzazione viene lanciato il metodo **getCurrentPosition** sull'oggetto **navigator.geolocation** non appena la pagina viene caricata. Questo metodo è una specifica API standardizzata dal W3C[12] implementata nel browser, che restituisce i valori presunti di longitudine e latitudine dello strumento su cui viene invocata.
La specifica è implementata in modo diverso nei vari browser ed utilizza diverse tecnologie per identificare la posizione, come vedremo nel capitolo successivo. Se l'utente modifica la sua posizione durante l'utilizzo dell'interfaccia web, può richiamare il metodo della geolocalizzazione attraverso l'apposito pulsante visto nella descrizione dell'interfaccia, che centrerà la mappa sulla sua nuova posizione.

---

[12] Il W3C (World Wide Web Consortium) è un'associazione fondata nel 1994 dal padre del web Tim Berners Lee allo scopo di migliorare e standardizzare i protocolli ed i linguaggi utilizzati nelle infrastrutture che caratterizzano la rete globale di Internet.

A seguito della registrazione o del login, se un utente dovesse lasciare la pagina e farvi ritorno in un secondo momento, non sarebbe costretto a reinserire i dati per la sua identificazione ma verrebbe portato direttamente alla schermata principale con la mappa ed il menù. Per fare questo è stato usato un apposito *cookie*[13], nel quale viene salvato l'username dell'utente e che rimane attivo finché questo non effettua il logout. Su questo cookie si basano anche alcune delle richieste che vengono effettuate al server e che necessitano dell'username. Utilizzando il cookie, gli script possono effettuare queste richieste senza bisogno di dover richiedere ogni volta all'utente l'inserimento dei suoi dati.

Per quanto riguarda la sicurezza dai dati inseriti dall'utente, è importante notare che le password non sono semplicemente inviate al server in chiaro (cioè nella loro forma normale), ma vengono prima criptate attraverso l'algoritmo MD5, rendendo così praticamente impossibile la loro identificazione in caso qualcuno intercetti la trasmissione HTTP o violi la sicurezza del server riuscendo ad entrare nel database. Questo algoritmo prende in input una stringa di lunghezza arbitraria, in questo caso la password, e ne restituisce una di 128 bit che viene salvata nel database al suo posto. Anche conoscendo questa stringa è molto difficile risalire a quella originale, rendendo così sicure le informazioni in essa contenute.

5.3 Applicazione mobile

L'applicazione per iPhone presenta le stesse caratteristiche che troviamo nell'interfaccia web. Attraverso una mappa permette infatti di aggiungere nuovi messaggi, consultare quelli inseriti da altri utenti (tramite la mappa o tramite una ricerca), eliminare quelli vecchi e gestire le località preferite. Anche l'accesso al server rimane pressoché invariato. L'unica differenza è che i dati restituiti saranno incapsulati in formato JSON e

[13] I cookie HTTP (più comunemente denominati web cookies) sono stringhe di testo di piccola dimensione inviate da un server ad un client web e poi rimandati indietro dal client al server ogni volta che il client accede allo stessa porzione dello stesso dominio. Salvando queste informazioni sul client (solitamente un browser) diventa possibile effettuare operazioni di autenticazione automatica o memorizzare informazioni specifiche riguardanti l'utente che accede al server.

non XML. Questo formato, adatto per gli scambi di dati in applicazioni client-server e basato sul linguaggio JavaScript, è stato preferito all'XML perché gestito in modo più efficiente da iPhone.

Attualmente l'applicativo è in fase di sviluppo e si avvale della collaborazione con il mio collega Paolo Ladisa. Una versione funzionante è già stata presentata e crediamo che quella definitiva uscirà sull'Apple App Store a breve.

6 Tecnologie utilizzate

6.1 Geolocalizzazione

Per geolocalizzazione si intende la determinazione della posizione geografica di un oggetto secondo le coordinate geografiche di latitudine e longitudine. Si basa su vari tipi di tecnologie, impiegate a seconda dello strumento utilizzato:

- *GPS*: Global Positioning Sysem (Sistema di Posizionamento Globale), basato sui segnale ottenuto dai satelliti in orbita attorno alla Terra. Per determinare la propria posizione con accuratezza e precisione il ricevitore GPS deve ricevere un segnale radio da almeno tre satelliti. Ogni satellite emette un segnale univoco, il ricevitore GPS può così abbinare il segnale ricevuto al satellite che lo ha emesso e, in base ad un calendario residente nella memoria del ricevitore GPS, ricavare la sua posizione orbitale. La distanza dal ricevitore viene calcolata (per ogni satellite) e, in base ai risultati, il ricevitore GPS ricava con precisione la sua posizione geografica. È sicuramente il metodo più preciso, arrivando ad identificare la posizione di un oggetto sul piano terrestre con uno scarto di soli pochi metri.
- *Reti WiFi* o *WLAN*: tecnica basata sulla triangolazione del segnale delle diverse fonti WiFi presenti nelle vicinanze, le quali vengono a loro volta localizzate attraverso dei LIS (Location Information Server) presenti nella rete e contenenti informazioni sulla posizione fisica delle varie infrastrutture collegate.
- *Indirizzo IP*: attraverso la rete internet, dove la localizzazione dipende dalla registrazione del collegamento ad una database LIS.
- *Localizzazione tramite le celle della rete telefonica cellulare*: si analizza la potenza del segnale di ogni cella telefonica (che hanno determinate coordinate geografiche) collegata con il dispositivo e ne viene determinata la distanza.

Il servizio di geolocalizzazione implementato nel browser web è una specifica API del W3C. Definisce un insieme di oggetti standard ECMAScript (cap. 6.2) che, se eseguiti nell'applicazione del client, restituiscono la posizione del dispositivo utilizzato attraverso una delle tecnologie sopra riportate. Queste tecnologie sono perciò trasparenti all'applicazione e vengono invocate al livello in cui è implementata la specifica, che rimane quindi invisibile all'utente. La posizione è restituita con un'accuratezza che dipende dal tipo di tecnologia utilizzata.

6.2 Linguaggi di scripting web

6.2.1 JavaScript

JavaScript è un linguaggio di scripting orientato agli oggetti comunemente utilizzato nelle pagine web, standardizzato con il nome ECMAScript. L'ultimo standard, di Marzo 2011, è ECMA-262 Edition 5.1. ed è anche uno standard ISO.

Un linguaggio di scripting è un linguaggio di programmazione interpretato, che quindi può essere eseguito direttamente dal codice sorgente senza la necessità di essere prima tradotto in linguaggio macchina da un compilatore. Viene quindi letto ed eseguito direttamente dal browser web e per questo viene anche definito linguaggio lato client(a differenza del PHP che come vedremo viene eseguito lato server).

Per lo sviluppo dell'interfaccia web, oltre alle API standard JavaScript, è stato usato anche il framework(libreria di funzioni fornita da terze parti) jQuery, per semplicizzare e rendere più efficienti alcune operazioni, soprattutto quelle legate ad AJAX ed alla scansione degli elementi del DOM[14].

[14] Il DOM (Documet Object Model) è una forma di rappresentazione dei documenti strutturati come modello orientato agli oggetti. È lo standard ufficiale del W3C per la rappresentazione di documenti, strutturati in maniera da essere neutrali sia per la lingua che per la piattaforma. DOM è inoltre la base per una vasta gamma di interfacce di programmazione delle applicazioni ed è utilizzato per l'identificazione e la ricerca di elementi all'interno della sua struttura.

6.2.2 PHP

PHP è l'acronimo ricorsivo di "PHP: Hypertext Preprocessor", ossia preprocessore di ipertesti. In informatica, un preprocessore (o precompilatore) è un programma che effettua sostituzioni testuali sul codice sorgente.I più comuni tipi di sostituzioni sono l'espansione di macro, l'inclusione di altri file, e la selezione condizionale. Tipicamente il preprocessore viene lanciato nel processo di compilazione di un software e il file risultante verrà preso in input da un compilatore.
PHP è un altro linguaggio di scripting interpretato. A differenza di JavaScript però, non è eseguito dal browser, ma dal server che viene contattato da quest'ultimo. È spesso utilizzato per generare codice HTML che verrà poi inviato al browser, il quale si occuperà della sua visualizzazione.
Nel nostro contesto è stato utilizzato unicamente per la comunicazione tra applicativo - sito web ed il database MySql contenente tutti i dati lasciati dagli utenti .

6.3 AJAX

Asynchronous JavaScript and XML. È è una tecnica di sviluppo per la realizzazione di applicazioni web interattive basata sull'utilizzo di diverse tecnologie. La realizzazione di applicazioni HTML con AJAX si fonda su uno scambio di dati in background fra server e client che consente l'aggiornamento dinamico di una pagina web senza che questa venga esplicitamente ricaricata da parte dell'utente. AJAX è definito asincrono nel senso che i dati sono richiesti al server e caricati in background senza interferire con il comportamento della pagina esistente. Questa tecnologia ruota intorno all'utilizzo dell'oggetto **XMLHttpRequest**, elemento utilizzabile all'interno degli script per creare una connessione asincrona tra la pagina e le risorse necessarie. Tramite questo oggetto è possibile inviare dati ad un server ed attendere una risposta. I dati scambiati durante questo tipo di connessione sono spesso formattati attraverso un linguaggio di markup come XML o altri linguaggi adatti allo scambio di dati come JSON. L'XML (sigla di eXtensible Markup Language) è un linguaggio di markup, ovvero un linguaggio

marcatore (usato per marcare i dati e dividerli in varie sezioni) basato su un meccanismo sintattico che consente di definire e controllare il significato degli elementi contenuti in un documento o in un testo. Viene utilizzato per lo scambio di dati e per la rappresentazione di documenti.

Il JSON (acronimo di JavaScript Object Notation) è un altro tipo di linguaggio usato nello scambio di dati ed è basato sul linguaggio JavaScript. Al contrario di XML però, non è un linguaggio per la marcatura, ma un semplice insieme di regole sintattiche usate per formattare i dati prima che vengano scambiati, rendendo più facile la loro lettura ed il loro utilizzo da parte del ricevente.

7 Sviluppi futuri

Così presentato, il servizio svolge appieno le funzioni per cui era stato progettato e permette la condivisione di messaggi e note legate ad una posizione geografica.
Sono tuttavia in fase di studio alcune idee aggiuntive che andranno a sommarsi a quelle già presentate per portare nuove funzionalità e raffinare quelle già presenti.

- *Incorporamento di link*: sarà possibile aggiungere link ipertestuali ai post-it, che ne mostreranno un'anteprima come avviene su Facebook.

- *Condivisione di file multimediali*: oltre alla condivisione di messaggi, le note potranno essere usate per condividere immagini, video e musica. Il riquadro contenente le informazioni del post dovrà essere adattato per poter ospitare questi contenuti multimediali, visualizzando la loro anteprima nel caso di video ed immagini ed incorporando un player audio nel caso di file musicali.

- *Amici*: sarà possibile aggiungere utenti ad una lista privata di amici, visualizzare solo i post provenienti da un sottoinsieme di questa lista o mostrare quelli inseriti solo a determinate persone.

- *Notifiche*: abilitando le notifiche, gli utenti potranno ricevere avvisi in tempo reale sulle informazioni contenute nei post presenti nelle loro vicinanze. Combinando questo elemento con la possibilità di avere una lista di amici con cui interagire direttamente, le notifiche permetteranno di avvertire l'utente della presenza di un post-it nelle sue vicinanze diretto espressamente a lui. In questo modo le note diventeranno qualcosa di molto più personale, permettendo una maggiore integrazione dell'applicazione con le esigenze della moderna comunicazione informatica.

- *Realtà aumentata*: la Realtà Aumentata consiste nella sovrapposizione di elementi digitali su una visione dal vivo di uno spazio reale. Mentre la videocamera integrata nello smartphone mostrerà in tempo reale l'ambiente circostante, a queste immagini saranno sovrapposte quelle dei post-it, che andranno perciò (grazie all'ausilio del GPS) a trovarsi nella loro posizione reale e saranno visibili come parte della realtà ripresa e consultabili dal display con un semplice click.

- *Sviluppo su altre piattaforme*: le prospettive per il futuro sono di poter espandere questo progetto anche a soluzioni mobili diverse da quella proposta per i sistemi Apple, come Android di Google e Windows Mobile.

8 Conclusioni

Obbiettivo di questo elaborato è stato l'esposizione dell'analisi dei requisiti, della progettazione e dell'implementazione di un sistema che permettesse la condivisione di messaggi legati ad una posizione in base a precise coordinate di latitudine e longitudine.
È stata inizialmente presentata l'idea di base, con un breve accenno alle applicazioni simili già presenti sul mercato e come queste risultino spesso troppo complesse per lo svolgimento di un compito semplice come lasciare un appunto o come non svolgano questo incarico in maniera totalmente soddisfacente.
Successivamente sono stati esposti i requisiti che il sistema finito doveva presentare, con particolare enfasi sulla semplicità d'uso e l'immediatezza, e le caratteristiche necessarie per il suo utilizzo. Nella fase di progettazione sono state descritte le metodologie utilizzate per la gestione delle infrastrutture necessarie al funzionamento del sistema, come il server utilizzato per l'interfaccia web ed il database in esso contenuto. Di quest'ultimo è stato presentato il modello ER che ne rappresenta concettualmente i dati.
Infine in questa sezione è stato mostrato a grandi linee lo schema rappresentativo dello scambio dei dati tra server ed applicativi ed i relativi protocolli di trasmissione.
Questo schema è stato sviluppato a livello pratico nella parte dedicata all'implementazione. In quest'ultima sezione sono stati infatti esposti gli elementi utilizzati nella costruzione vera e propria del servizio, come i costrutti per creare le tabelle del database, gli oggetti impiegati nella programmazione dell'interfaccia web e le tecniche AJAX usate per lo scambio di informazioni.
Le principali tecnologie utilizzate sono state trattate più a fondo nel capitolo sei, dove troviamo tra l'altro una breve esposizione di alcuni dei linguaggi di scripting impiegati per la realizzazione del sito e delle pagine PHP che fungono da interfaccia con il database MySQL.

In conclusione posso affermare di essere riuscito nel mio intento di sviluppare il sistema per lo scambio di informazioni geolocalizzate descritto, ora accessibile da tutti via web all'indirizzo *www.geopostit.com*. L'applicazione mobile che integra e completa il servizio è in fase di creazione e sarà presentata a breve con la collaborazione del mio collega Paolo Ladisa e presenterà ulteriori funzioni e miglioramenti come descritto nel capitolo sugli sviluppi futuri.

Come ulteriore dimostrazione, ho chiesto ad alcuni miei conoscenti di testare l'interfaccia web in modo da capire il grado di soddisfazione dell'utente e come poter migliorare il servizio. Il feedback è stato generalmente molto positivo, il che mi porta alla considerazione che un prodotto simile potrebbe avere utilizzi pratici nel mondo reale. L'esistenza di altre soluzioni più famose, anche se magari meno efficienti per adempiere allo scopo prefisso, mi rende tuttavia un po' pessimista sull'effettiva diffusione dell'utilizzo di questo servizio.

Continuo comunque ad essere convinto che la facilità d'uso e la semplicità siano la chiave per realizzare prodotti efficaci e funzionali.

Infatti auspico nel prossimo futuro, anche grazie all'aiuto di collaborazioni esterne e ad un miglioramento del servizio, di portare il sistema proposto ad un utilizzo più diffuso e conosciuto.

Appendici

A. Codice interfaccia web

index.html

```
<!DOCTYPE html >
<head>
   <meta name="viewport" content="initial-scale=1.0, user-scalable=no"/>
   <meta name="description" content="Leave notes for everyone all around the world quickly
   and easily with GeoPostIt!"/>
   <meta name="keywords" content="geolocation, notes, post, post-it, share notes,
   location based"/>
   <meta http-equiv="content-type" content="text/html; charset=UTF-8"/>
   <link rel="image_src" href="http://geopostit.com/preview.png" />
   <title>GeoPostIt</title>
   <link href="http://www.geopostit.com/PostItThumb.jpg" rel="image_src"/>
   <link href="css/myCss.css" rel="stylesheet" type="text/css"/>
   <link href="css/jquery-ui-1.8.20.custom.css" rel="stylesheet" type="text/css"/>
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js"></script>
   <script src="jQueryCookie.js"></script>
   <script src="md5.js"></script>
   <script src="http://ajax.googleapis.com/ajax/libs/jqueryui/1.8/jquery-ui.min.js"></script>
   <script type="text/javascript" src="http://maps.googleapis.com/maps/api/js?
   key=AIzaSyAeY0BeBtUl-31Orw8bG-T6YI2zKumYAKM&sensor=false"></script>
   <script type="text/javascript" src="http://google-maps-utility-library-v3.googlecode.com/
   svn/trunk/infobox/src/infobox.js"></script>
   <script type="text/javascript" src="main.js"></script>
</head>
<body>
   <div id="loginDiv">
      <img src="logo.png" alt="geopost logo"/>
      <img src="notty.png" id="notty" alt="notty the cat"/>
      <br />
      <br />
      <form id="loginForm">
         <fieldset id="loginFieldset" class="container">
            <legend class="container">
            Login or Register
            </legend>
            Username:
            <input type="text" id="loginText" maxlength="20" size="25" class="container"/>
            <br />
            <br />
            Password:
            <input type="password" id="passwordText" maxlength="20" size="25"
            class="container" onkeypress="if(event.keyCode==13){login()}"/>
            <br />
            <br />
            <input type="button" class="geoButton" value="Login" id="qwerty"
            onClick="login()" />
            <input type="button" class="geoButton" value="Register" onclick="register()"/>
```

```
        </fieldset>
      </form>
      <div id="bottomDescription">
        Service in beta testing. An application for iPhone will be available soon on the App Store.
      </div>
      <div id="beta">
      ver 0.32
      </div>
      <div id="logResponse"></div>
    </div>
    <div id="mainTable">
      <div id="usrLog">
        <span id="displayUsrName"></span>
        <input type="button" class="geoButton" value="Log Out" onclick="logOut()"/>
      </div>
      <div id="map" class="container"></div>
      <div id="searchDiv">
        <input type="text" class="container" id="searchLocationForm" value="Search Location"
        maxlength="40" size="25" onkeypress="if(event.keyCode==13){searchLocation()}"/>
        <input type="button" class="geoButton" id="searchLocationButton"
        onclick="searchLocation()" value="Search">
        <input type="text" class="container" id="searchPostForm" value="Search Post"
        maxlength="40" size="25" onkeyup="searchPost($('#searchPostForm').val())"
        onclick="$('#searchPostForm').val('')"/>
      </div>
      <form>
        <fieldset class="container"  id="menu">
          <legend class="container">
          Write Post
          </legend>
          <textarea class="container" id = "newPost" rows = "6" cols = "28" maxlength="200"
          style="resize:none">Write your post here...</textarea>
          <br />
          <label>Expire Date:</label>
          <div id="datepicker"></div>
          <div id="inMenuTextBox">
          Select a location by clicking on the map
          <br />
          or choose a favourite place from the list:
          </div>
          <select id="favSelect" onchange="changePositionMarkerToFav()"></select>
          <input type="button" class="geoButton" onclick="delFavourite()" value="Delete Selected"/>
          <div id="addFavText">
          Add a new favourite place
          </div>
          <input type="text" class="container" id="favNameForm" maxlength="20" size="25"/>
          <input type="button" class="geoButton" onclick="addFavourite()" value="ADD"/>
          <br />
          <input type="button" class="geoButton" id="postButton" onclick="postMessage()" value="POST!"/>
          <span class="horizontalLine"></span>
          <input type="button" class="geoButton" id="deleteButton" onclick="deleteAction()" value="Delete Post"/>
        </fieldset>
      </form>
      <span class="geoButton" id="geoLoc" onclick="geoLocate()"><img src="crossair.png" alt="crossair" /></span>
      <span onclick="moo()" style="color:#56a0ff">moo</span>
      <div id="postResponse"></div>
      <div id="deleteInfo"></div>
    </div>
    <div id="searchResultDiv"></div>
  </body>
</html>
```

main.js

```
var map;
var latlng;
var clickedLatlng;
var markersArray = [];
var positionArray = [];
var expireDate;
var divResponse;
var markers;
var pinMarker;
var pin = false;
var postErase = false;

//load map
function showMap(position) {
   latlng = new google.maps.LatLng(position.coords.latitude, position.coords.longitude);
   var myOptions = {
      zoom : 15,
      panControl : true,
      zoomControl : true,
      zoomControlOptions : {
         style : google.maps.ZoomControlStyle.DEFAULT
      },
      mapTypeControl : true,
      mapTypeControlOptions : {
         style : google.maps.MapTypeControlStyle.DROPDOWN_MENU
      },
      scaleControl : false,
      streetViewControl : false,
      overviewMapControl : false,
      center : latlng,
      mapTypeId : google.maps.MapTypeId.ROADMAP
   };
   var myMapStyle = [{
      featureType : "landscape",
      stylers : [{
         hue : "#71dfff"
      }, {
         saturation : 25
      }, {
         lightness : -5
      }, {
         gamma : 1.60
      }]
   }, {
      featureType : "poi",
      stylers : [{
         hue : "#0764b9"
      }, {
         saturation : 30
      }, {
         lightness : -15
      }, {
         gamma : 1.60
      }]
   }, {
      featureType : "road",
      stylers : [{
         hue : "#0029ae"
      }, {
```

```
            saturation : 20
        }, {
            lightness : -20
        }, {
            gamma : 1.60
        }]
    }];
    map = new google.maps.Map(document.getElementById("map"), myOptions);
    map.setOptions({
        styles : myMapStyle
    });
    google.maps.event.addListener(map, 'click', function(event) {
        placePositionMarker(event.latLng);
    });
    google.maps.event.addListener(map, 'tilesloaded', function(event) {
        clearPostMarkers();
        addPostMarkers();
    });
}

//error handling function
function errorHandler(err) {
    if(err.code == 1) {
        showResponse('postResponse', 1, 'Cannot use map without Geolocation');
    } else if(err.code == 2) {
        showResponse('postResponse', 1, 'Position Unknown');
    }
}

//geolocation function
function getLocation() {
    if(navigator.geolocation) {
        // timeout at 20000 milliseconds (20 seconds)
        var options = {
            timeout : 20000
        };
        navigator.geolocation.getCurrentPosition(showMap, errorHandler, options);
    } else {
        showResponse('postResponse', 1, 'Your browser does not support Geolocation');
    }
}

function geoLocate() {
    var options = {
        timeout : 20000
    };
    navigator.geolocation.getCurrentPosition(function(position) {
        latlng = new google.maps.LatLng(position.coords.latitude, position.coords.longitude);
        map.setCenter(latlng);
        $('#searchLocationForm').val("Search Location");
    }, errorHandler, options);

}

//remove all markers from the map
function clearPostMarkers() {
    if(markersArray) {
        for(i in markersArray) {
            markersArray[i].setMap(null);
            markersArray[i].infobox.close();
        }
        markersArray = [];
    }
}
```

```
//place the markers on the map with their messages displayed inside infoboxes
function addPostMarkers() {
   var bounds = new google.maps.LatLngBounds;
   var topLat = map.getBounds().getNorthEast().lat();
   var bottomLat = map.getBounds().getSouthWest().lat();
   var leftLon = map.getBounds().getSouthWest().lng();
   var rightLon = map.getBounds().getNorthEast().lng();
   $.ajax({
      type : "GET",
      url : "genMsgXML.php",
      data : "topLat=" + topLat + "&bottomLat=" + bottomLat + "&leftLon=" + leftLon + "&rightLon=" + rightLon,
      dataType : "xml",
      success : function(xml) {
         $(xml).find('marker');
         var mk = $(xml).find('marker');
         for(var i = 0; i < mk.length; i++) {
            var name = mk[i].getAttribute("user");
            var messagePost = mk[i].getAttribute("message");
            var messageId = mk[i].getAttribute("id");
            var markerPosition = new google.maps.LatLng(parseFloat(mk[i].getAttribute("lat")),
            parseFloat(mk[i].getAttribute("lon")));
            var html = "<b>" + name + "</b> <br/>" + messagePost;
            //special markers icon for admin messages
            if((name == "admin") || (name == "Admin")) {
               var pinIco = new google.maps.MarkerImage('pinB.png', new google.maps.Size(25, 25), new
            google.maps.Point(0, 0), new google.maps.Point(15, 15));
            } else {
               var pinIco = new google.maps.MarkerImage('pin.png', new google.maps.Size(25, 25), new
            google.maps.Point(0, 0), new google.maps.Point(15, 15));
            }
            var marker = new google.maps.Marker({
               map : map,
               position : markerPosition,
               draggable : ($.cookie("GeoPost") == name),
               icon : pinIco
            });

            //define the text and style for all infoboxes
            var boxText = document.createElement("div");
            boxText.style.cssText = "border: 2px solid black; word-wrap:break-word; margin-top: 8px;
            background:#faea7b; color:black; font-family:Arial; font-size:14px; padding: 5px;";
            boxText.innerHTML = html;
            //define the options for all infoboxes
            var myOptions = {
               content : boxText,
               disableAutoPan : false,
               maxWidth : 0,
               pixelOffset : new google.maps.Size(-140, 0),
               zIndex : null,
               boxStyle : {
                  background : "url('http://google-maps-utility-library-v3.googlecode.com/
                  svn/trunk/infobox/examples/tipbox.gif') no-repeat",
                  opacity : 0.85,
                  width : "150px",
               },
               closeBoxMargin : "12px 4px 2px 2px",
               closeBoxURL : "http://www.google.com/intl/en_us/mapfiles/close.gif",
               infoBoxClearance : new google.maps.Size(1, 1),
               isHidden : false,
               pane : "floatPane",
               enableEventPropagation : false
            };
            marker.infobox = new InfoBox(myOptions);
```

```
            markersArray.push(marker);
            google.maps.event.addListener(marker, 'click', (function(marker, i, messageId) {
               return function() {
                  if(postErase) {
                     //delete post
                     marker.setAnimation(google.maps.Animation.BOUNCE);
                     window.setTimeout(function() {
                        marker.setAnimation(null);
                     }, 750);
                     deletePost(messageId);
                  } else {
                     for( h = 0; h < markersArray.length; h++) {
                        markersArray[h].infobox.close();
                     }
                     markersArray[i].infobox.open(map, this);
                  }

               }
            })(marker, i, messageId));

            if($.cookie("GeoPost") == name) {

               google.maps.event.addListener(marker, 'dragstart', function() {
                  for( h = 0; h < markersArray.length; h++) {
                     markersArray[h].infobox.close();
                  }
               });

               google.maps.event.addListener(marker, 'dragend', (function(marker, messageId) {
                  return function() {
                     changePostLocation(marker.getPosition(), messageId)
                  }
               })(marker, messageId));
               ;
            }

         }
      }
   });
}

function moo() {
   for( i = 0; i < markersArray.length; i++) {
      markersArray[i].infobox.close();
      window.setTimeout("animate(" + i + ")", i * 250);
   }
}

function animate(i) {
   markersArray[i].setAnimation(google.maps.Animation.BOUNCE);
}

//change the coordinates of the post when marker is dragged around the map
function changePostLocation(latlng, id) {
   $.ajax({
      type : "POST",
      url : "changeLocation.php",
      data : "lat=" + latlng.lat() + "&lon=" + latlng.lng() + "&id=" + id,
      contentType : "application/x-www-form-urlencoded, text/html, application/xml, application/rdf-xml,
      text/html+xml",
      success : function(r) {
         if(r == 0) {
            //showResponse('postResponse', 0, 'Post location changed');
         } else
```

```
            showResponse('postResponse', 1, 'mySql Error');
        },
        error : function() {
            showResponse('postResponse', 1, 'Server Error');
        }
    });
}

//place on the map the marker used for the new post
function placePositionMarker(location) {
    clickedLatlng = location;
    if(pin) {

        pinMarker.setMap(null);
        pin = false;
    }
    var ico = new google.maps.MarkerImage('markerIco.png',
    // This marker is 20 pixels wide by 28 pixels tall.
    new google.maps.Size(20, 28),
    // The origin for this image is 0,0.
    new google.maps.Point(0, 0),
    // The anchor for this image is the base of the flagpole at 10,28.
    new google.maps.Point(10, 28));
    pinMarker = new google.maps.Marker({
        position : clickedLatlng,
        icon : ico,
        draggable : true,
        map : map
    });

    google.maps.event.addListener(pinMarker, "dragend", function() {
        clickedLatlng = pinMarker.getPosition();
    });
    pin = true;
    $('#favSelect').val("");
}

//display the new mesage marker in the location on the selected favourite
function changePositionMarkerToFav() {
    if($('#favSelect').val() != "") {
        var favLat = $('#favSelect :selected').attr("lat");
        var favLon = $('#favSelect :selected').attr("lon");
        var favMarker = new google.maps.LatLng(favLat, favLon);
        clickedLatlng = favMarker;
        if(pin) {
            pinMarker.setMap(null);
            pin = false;
        }
        var ico = new google.maps.MarkerImage('markerIco.png',
        // This marker is 20 pixels wide by 28 pixels tall.
        new google.maps.Size(20, 28),
        // The origin for this image is 0,0.
        new google.maps.Point(0, 0),
        // The anchor for this image is the base of the flagpole at 10,28.
        new google.maps.Point(10, 28));
        pinMarker = new google.maps.Marker({
            position : clickedLatlng,
            icon : ico,
            draggable : true,
            map : map
        });

        google.maps.event.addListener(pinMarker, "dragend", function() {
            clickedLatlng = pinMarker.getPosition();
```

```
        });

        pin = true;
        map.setCenter(favMarker);
    } else {
        clearPositionMarker();
    }
}

//send the new post to the server
function postMessage() {
    if(document.getElementById("newPost").value == "")
        showResponse('postResponse', 1, 'Post is Empty');
    else if(expireDate == null)
        showResponse('postResponse', 1, 'No Expire date Selected');
    else {
        if((clickedLatlng == null) || (clickedLatlng == undefined))
            showResponse('postResponse', 1, 'Position not Selected');
        else {
            var msgLat = clickedLatlng.lat();
            var msgLon = clickedLatlng.lng();
            var message = document.getElementById("newPost").value;
            $.ajax({
                type : "POST",
                url : "postMsg.php",
                data : "name=" + $.cookie("GeoPost") + "&message=" + message + "&expire=" + expireDate +
                "&lat=" + msgLat + "&lon=" + msgLon,
                contentType : "application/x-www-form-urlencoded, text/html, application/xml, application/rdf-xml,
                text/html+xml",
                success : function(r) {
                    if(r == 0) {
                        showResponse('postResponse', 0, 'Post Complete');
                        clearPostMarkers();
                        addPostMarkers();
                        clearPositionMarker();
                        $('#favSelect').val("");
                    } else
                        showResponse('postResponse', 1, 'mySql Error');
                },
                error : function() {
                    showResponse('postResponse', 1, 'Server Error');
                }
            });
        }
    }
}

//set the global variable postErase to true or false
//when the variable is set to true, clicking on a post
//will cause its deletion
function deleteAction() {
    if(postErase) {
        //end delete action
        postErase = false;
        $("#deleteInfo").fadeOut(200);
        $("#deleteInfo").html("");
        $("#deleteButton").attr("class", "geoButton");
        $("#deleteButton").attr("value", "Delete Post");
        $(this).css('cursor', 'pointer');

    } else {
        //start delete action
        postErase = true;
        $("#deleteInfo").html("Select Post to delete");
```

```
        $("#deleteInfo").fadeIn(200);
        $("#deleteButton").attr("class", "deleteActionButton");
        $("#deleteButton").attr("value", "   Return   ");
        $(this).css('cursor', 'url(rubberCursor.png)');
    }

}

//delete a post from the database and remove it from the map
function deletePost(messageId) {
    $.ajax({
        type : "POST",
        url : "deletePost.php",
        data : "user=" + $.cookie("GeoPost") + "&id=" + messageId,
        contentType : "application/x-www-form-urlencoded, text/html, application/xml, application/rdf-xml,
        text/html+xml",
        success : function(r) {
            if(r == 0) {
                showResponse('postResponse', 0, 'Post erased');
                postErase = false;
                $("#deleteInfo").fadeOut(200);
                $("#deleteInfo").html("");
                $("#deleteButton").attr("class", "geoButton");
                $("#deleteButton").attr("value", "Delete Post");
                $(this).css('cursor', 'pointer');
                clearPostMarkers();
                addPostMarkers();
            } else if(r == 1) {
                showResponse('postResponse', 1, 'Post is not yours!');
            } else
                showResponse('postResponse', 1, 'mySql Error');
        },
        error : function() {
            showResponse('postResponse', 1, 'Server Error');
        }
    });

}

//login procedure
function login() {
    var userName = $("#loginText").val();
    var passWord = $("#passwordText").val();
    if(userName == "" || passWord == "")
        showResponse('logResponse', 1, 'Incomplete data');
    else {
        var md5Pass = calcMD5(passWord);
        $.ajax({
            url : "login.php",
            type : "POST",
            data : "name=" + userName + "&pass=" + md5Pass,
            contentType : "application/x-www-form-urlencoded, text/html, application/xml, application/rdf-xml,
            text/html+xml",
            success : function(data) {
                if(data == 0) {
                    //the user is registered
                    $.cookie("GeoPost", userName);
                    getLocation();
                    createFavSelect();
                    createCalendar();
                    $("#displayUsrName").html("Hello <b>" + userName + "</b>!");
                    $("#logResponse").fadeOut(0);
                    $("#postesponse").fadeOut(0);
                    $("#loginDiv").fadeOut(0);
```

```
                $("#mainTable").fadeIn(1500);
            } else if(data == 1)
                showResponse('logResponse', 1, 'mySql Error');
            else if(data == 2)
                showResponse('logResponse', 1, 'Wrong Username/Password');
        },
        error : function() {
            showResponse('logResponse', 1, 'Server Error');
        }
    });
  }
}

//register procedure
function register() {
  var userName = $("#loginText").val();
  var passWord = $("#passwordText").val();
  if(userName == "" || passWord == "")
    showResponse('logResponse', 1, 'Incomplete data');
  else {
    var md5Pass = calcMD5(passWord);
    $.ajax({
      url : "register.php",
      type : "POST",
      data : "name=" + userName + "&pass=" + md5Pass,
      contentType : "application/x-www-form-urlencoded, text/html, application/xml, application/rdf-xml,
      text/html+xml",
      success : function(data) {
        if(data == 0) {
          $.cookie("GeoPost", userName);
          getLocation();
          createFavSelect();
          createCalendar();
          $("#displayUsrName").html("Hello <b>" + userName + "</b>!");
          $("#logResponse").fadeOut(0);
          $("#postesponse").fadeOut(0);
          $("#loginDiv").fadeOut(0);
          showResponse('postResponse', 0, 'User Registered');
          $("#mainTable").fadeIn(2000);
        } else if(data == 1)
          showResponse('logResponse', 1, 'mySql Error');
        else if(data == 2)
          showResponse('logResponse', 1, 'Username alredy used');
      },
      error : function() {
        showResponse('logResponse', 1, 'Server Error');
      }
    });
  }
}

//logout procedure
function logOut() {
  $.cookie("GeoPost", null);
  $("#displayUsrName").html("");
  $("#logResponse").fadeOut(0);
  $("#postesponse").fadeOut(0);
  if(postErase) {
    //end delete action
    postErase = false;
    $("#deleteInfo").fadeOut(200);
    $("#deleteInfo").html("");
    $("#deleteButton").attr("class", "geoButton");
    $("#deleteButton").attr("value", "Delete Post");
```

```
        $(this).css('cursor', 'pointer');
    }
    $("#searchResultDiv").fadeOut(400);
    $("#searchResultDiv").empty();
    $("#searchLocationForm").val("Search Location");
    $("#searchPostForm").val("Search Post");
    $("#passwordText").val("");
    $("#mainTable").fadeOut(600);
    $("#loginDiv").fadeIn(600);
}

//function used to display the calendar needed to set the
//expire date of the post
function createCalendar() {
    $('#datepicker').datepicker({
        dateFormat : 'yyyy-mm-dd',
        inline : true,
        firstDay : 1,
        minDate : "+0",
        maxDate : "+4y",
        onSelect : function() {
            var expDay = ("0" + $("#datepicker").datepicker('getDate').getDate()).slice(-2);
            var expMonth = ("0" + ($("#datepicker").datepicker('getDate').getMonth() + 1)).slice(-2);
            var expYear = $("#datepicker").datepicker('getDate').getFullYear();
            expireDate = expYear + "-" + expMonth + "-" + expDay;
        }
    });
}

//take the saved favourites of the current user from the server
//and display them as options in an HTML select menu
function createFavSelect() {
    $('#favSelect').html("<option></option>");
    $.ajax({
        type : "GET",
        url : "genFavXML.php",
        data : "name=" + $.cookie("GeoPost"),
        dataType : "xml",
        success : function(xml) {
            $(xml).find('fav');
            var favourites = $(xml).find('fav');
            for(var i = 0; i < favourites.length; i++) {
                var location = favourites[i].getAttribute("location");
                var favLat = parseFloat(favourites[i].getAttribute("lat"));
                var favLon = parseFloat(favourites[i].getAttribute("lon"));
                $('#favSelect').append("<option lat=" + favLat + " lon=" + favLon + ">" + location + "</option>");
            }
        }
    });
}

//add a favourite to the database by taking the coordinates of the
//pin placed on the map and a name from the form
function addFavourite() {
    if($('#favNameForm').val() == "")
        showResponse('postResponse', 1, 'Set Favourite Name');
    else if(clickedLatlng == null)
        showResponse('postResponse', 1, 'Select Location');
    else {
        $.ajax({
            type : "POST",
            url : "addFavourite.php",
            data : "name=" + $.cookie("GeoPost") + "&locationName=" + $('#favNameForm').val() + "&lat=" +
            clickedLatlng.lat() + "&lon=" + clickedLatlng.lng(),
```

```
            contentType : "application/x-www-form-urlencoded, text/html, application/xml, application/rdf-xml,
            text/html+xml",
            success : function(r) {
                if(r == 0) {
                    showResponse('postResponse', 0, 'Favourite Added!');
                    $('#favSelect').append("<option lat=" + clickedLatlng.lat() + " lon=" + clickedLatlng.lng() + ">" +
                    $('#favNameForm').val() + "</option>");
                    clearPositionMarker();
                    $('#favNameForm').val("");
                } else if(r == 1)
                    showResponse('postResponse', 1, 'Favourite name alredy used');
                else {
                    showResponse('postResponse', 1, 'mySql Error');
                }
            },
            error : function() {
                showResponse('postResponse', 1, 'Server Error');
            }
        });
    }
}

//delete a favourite place from the server and removes it from the
//select list
function delFavourite() {
    if($('#favSelect').val() == "")
        showResponse('postResponse', 1, 'No Favourite Selected');
    else {
        $.ajax({
            type : "POST",
            url : "delFavourite.php",
            data : "name=" + $.cookie("GeoPost") + "&locationName=" + $('#favSelect').val(),
            contentType : "application/x-www-form-urlencoded, text/html, application/xml, application/rdf-xml,
            text/html+xml",
            success : function(r) {
                if(r == 0) {
                    showResponse('postResponse', 0, 'Favourite Removed');
                    $('#favSelect :selected').remove();
                    clearPositionMarker()
                } else
                    showResponse('postResponse', 1, 'mySql Error');
            },
            error : function() {
                showResponse('postResponse', 1, 'Server Error');
            }
        });
    }
}

//remove from the map the pin placed by the user
function clearPositionMarker() {
    if(pin) {
        pinMarker.setMap(null);
        pin = false;
    }
    clickedLatlng = null;

}

//function used to display messages at the user
//type can be set to 0 for success messages
//or 1 for error messages
//the messages will be displayed in the given DIV
function showResponse(divID, type, message) {
```

```
    var html;
    divResponse = '#' + divID;
    if(type == 1) {
        $(divResponse).css({
            "background-color" : "#f02b2b",
        });
        html = " <b>✘</b> " + message;
        $(divResponse).html(html);
        $(divResponse).fadeIn(100);
        window.setTimeout("$(divResponse).fadeOut(800)", 3500);
    } else {
        $(divResponse).css({
            "background-color" : "#1ad027",
        });
        html = "  <b>✔</b> " + message;
        $(divResponse).html(html);
        $(divResponse).fadeIn(100);
        window.setTimeout("$(divResponse).fadeOut(800)", 3500);
    }
}

//search post in the database and display them in a list
function searchPost(element) {
    if(element == "") {
        $("#searchResultDiv").fadeOut(200);
    } else {
        $("#searchResultDiv").empty();
        var center = map.getCenter();
        $.ajax({
            type : "GET",
            url : "search.php",
            data : "search=" + element + "&lat=" + center.lat() + "&lon=" + center.lng(),
            dataType : "xml",
            success : function(xml) {
                $(xml).find('post');
                var searchedPosts = $(xml).find('post');
                if(searchedPosts.length == 0) {
                } else {
                    $('#searchResultDiv').append("<p class=geoButton onClick=closeSearchResult()>Close results</p>");
                    for(var i = 0; i < searchedPosts.length; i++) {
                        var userName = searchedPosts[i].getAttribute("user");
                        var message = searchedPosts[i].getAttribute("message");
                        var lat = parseFloat(searchedPosts[i].getAttribute("lat"));
                        var lon = parseFloat(searchedPosts[i].getAttribute("lon"));
                        $('#searchResultDiv').append("<p class=searchListElement onClick=goToPost(" + lat + "," + lon + ")>"
                        + message + "</p>");
                    }
                    $("#searchResultDiv").fadeIn(400);
                }
            }
        });

    }
}

//set the center of the map to a gived location
function searchLocation() {
    if($('#searchLocationForm').val() == "") {
        showResponse('postResponse', 1, 'Select a location');
    } else {
        var location = $('#searchLocationForm').val()
        var geocoder = new google.maps.Geocoder();
        geocoder.geocode({
            'address' : location
```

```
        }, function(results, status) {
            if(status == google.maps.GeocoderStatus.OK) {
                var loc = results[0].geometry.location;
                var searchedLatLng = new google.maps.LatLng(loc.lat(), loc.lng());
                map.setCenter(searchedLatLng);
            } else {
                showResponse('postResponse', 1, status);
            }
        });
    }
}

//go to the location of a written post when it is selected
//from the results of a search
function goToPost(searchLat, searchLon) {
    var searchMarker = new google.maps.LatLng(searchLat, searchLon);
    map.setCenter(searchMarker);
    map.setZoom(16);
    closeSearchResult();
    $('#searchPostForm').val("");
}

function closeSearchResult() {
    $("#searchResultDiv").fadeOut(400);
    $("#searchResultDiv").empty();
    $('#searchPostForm').val("");
}

//the code below will load as soon as the DOM is loaded
//and before the page contents are loaded
$(document).ready(function() {
    $("#mainTable").fadeOut(0);
    $("#loginDiv").fadeOut(0);
    $("#logResponse").fadeOut(0);
    $("#postesponse").fadeOut(0);
    $("#searchResultDiv").fadeOut(0);

    //check if user alredy logged
    var usernameCookie = $.cookie("GeoPost");
    if(usernameCookie != null) {
        //the user is alredy logged in
        getLocation();
        createFavSelect();
        createCalendar();
        $("#displayUsrName").html("Hello <b>" + usernameCookie + "</b>!");
        $("#mainTable").fadeIn(600);
    } else
        $("#loginDiv").fadeIn(600);
});
```

myCss.css

```
* {
    font-family: "Chalkboard";
}
.horizontalLine {
    display: block;
    height: 3px;
    background-color: #baefff;
    margin-top: 5px;
}
body {
    background-image: url('../bg.png');
    background-repeat: repeat-x;
    text-align: center;
}
#backGroundImage {
    position: absolute;
    top: 0;
    left: 0;
    z-index: -1;
}
#notty {
    position: absolute;
    top: 200px;
    left: 800px;
}
#loginDiv {
    width: 300;
    margin: auto;
    text-align: center;
}
#loginForm {
    width: 300px;
    margin: 0 auto;
}
#loginFieldset {
    width: 280px;
    text-align: center;
    margin: 0 auto;
}
#mainTable {
    margin: auto;
    text-align: center;
}
p {
    margin: 4px;
}
.container {
    border-radius: 6px;
    -webkit-border-radius: 6px;
    -moz-border-radius: 6px;
    border-color: #71dfff;
    border-width: 3px;
    border-style: solid;
    background-color: #f0f0f0;
}
.container #newPost {
    border-color: #fde35a;
    border-width: 4px;
    background-color: #fde35a
```

```
}
.container[type=text] {
    background-color: white;
}
.container[type=password] {
    background-color: white;
}
.geoButton {
    background-image: none;
    border-style: solid;
    border-radius: 5px;
    -webkit-border-radius: 5px;
    -moz-border-radius: 5px;
    border-color: #71dfff;
    border-width: 2px;
    background-color: #baefff
}
.geoButton:hover {
    border-color: #84ff59;
    border-width: 2px;
    background-color: #ceffbd
}
#geoLoc {
    width: 40px;
    height: 40px;
    position: absolute;
    top: 85px;
    left: 102px;
}
.deleteActionButton {
    background-image: none;
    border-style: solid;
    border-radius: 2px;
    -webkit-border-radius: 5px;
    -moz-border-radius: 5px;
    border-color: #fd2121;
    border-width: 2px;
    background-color: #fca789
}
#postResponse {
    border-radius: 25px;
    border: 3;
    width: 200px;
    color: white;
    font-size: 15px;
    position: absolute;
    left: 14px;
    top: 15px
}
#logResponse {
    text-align: center;
    border-radius: 25px;
    border: 3;
    width: 200px;
    color: white;
    font-size: 4;
    margin-top: 60px;
    margin-left: auto;
    margin-right: auto;
}
#deleteInfo {
    width: 200px;
    color: orange;
    font-size: 18px;
```

```
    position: absolute;
    left: 625px;
    top: 15px
}
#usrLog {
    position: absolute;
    top: 5px;
    right: 5px;
    font-size: 22px;
    text-shadow: -1px 0 black, 0 1px black, 1px 0 black, 0 -1px black;
}
#map {
    width: 800px;
    height: 603px;
    position: absolute;
    top: 45px;
    left: 10px;
}
#menu {
    text-align: center;
    width: 260px;
    height: 600px;
    position: absolute;
    top: 33px;
    left: 860px;
}
#displayUsrName {
    color: #71dfff;
}
#newPost {
    font-size: 0.8em;
}
#inMenuTextBox {
    margin-top: 10px;
    font-size: 0.7em;
}
#addFavText {
    margin-top: 0px;
    font-size: 1em;
}
#postButton {
    margin-top: 5px;
    font-size: 1.6em;
}
#deleteButton {
    margin-top: 5px;
    font-size: 12px;
}
#searchDiv {
    width: 600px;
    height: 20px;
    position: absolute;
    top: 50px;
    left: 100px;
}
#searchLocationForm {
    position: absolute;
    left: 0px;
}
#searchLocationButton {
    position: absolute;
    left: 170px;
}
#searchPostForm {
```

```
    position: absolute;
    left: 440px;
}
#searchResultDiv {
    text-align: center;
    width: 200px;
    height: 200px;
    position: absolute;
    top: 80px;
    left: 520px;
    overflow: scroll;
    border-radius: 6px;
    -webkit-border-radius: 6px;
    -moz-border-radius: 6px;
    border-color: #71dfff;
    border-width: 3px;
    border-style: solid;
    background-color: #f0f0f0;
}
.searchListElement {
    word-wrap: break-word;
    background-image: none;
    border-style: solid;
    border-radius: 10px;
    -webkit-border-radius: 10px;
    -moz-border-radius: 10px;
    border-color: #faab21;
    border-width: 2px;
    background-color: #fde35a;
    font-size: 15px;
}
.searchListElement:hover {
    border-color: #faab21;
    border-width: 3px;
    background-color: #faab21
}
#bottomDescription {
    position: absolute;
    left: 0px;
    bottom: 0px;
}
#beta {
    position: absolute;
    bottom: 0px;
    right: 0px;
}
```

B. Codice dei file PHP del server

login.php

```
<?php
require("php_dbInfo.php");
$usr =$_POST["name"];
$pass = $_POST["pass"];

$con = mysql_connect(localhost, $username, $password);
      if (!$con){
            exit("1");
      }else{
            mysql_select_db($database, $con);

            $result = mysql_query("select Name, Password from Users");
            while($row = mysql_fetch_array($result)){

                  if (($usr==$row['Name'])&&($pass==$row['Password'])) exit("0");
        }
echo "2";
mysql_close($con);
}
?>
```

register.php

```
<?php
require ("php_dbInfo.php");
$usr = $_POST["name"];
$pass = $_POST["pass"];

$connection = mysql_connect(localhost, $username, $password);
if (!$connection) {
   exit("1");
} else {
   mysql_select_db($database, $connection);
   $result = mysql_query("select Name from Users");
   if ($result) {
      while ($row = mysql_fetch_array($result)) {

         if ($usr == $row['Name'])
            exit("2");
      }
      $result = mysql_query('INSERT INTO Users (Name, Password) VALUES ("' . $usr . '","' . $pass . '")');
      echo 0;
   }
```

```
    mysql_close($connection);
}
?>
```

search.php

```
<?php
require ("php_dbInfo.php");
$message = $_GET["search"];
$lat = $_GET["lat"];
$lon = $_GET["lon"];

$connection = mysql_connect(localhost, $username, $password);
if (!$connection) {
    exit("1");
} else {
    mysql_select_db($database, $connection);
    $result = mysql_query("select UserName,Message,Lat,Lon from Post where Message LIKE '%" . $message .
    "%' ORDER BY Message");
    if (!$result) {
        die('Invalid query: ' . mysql_error());
    }
    $dom = new DOMDocument("1.0");
    $node = $dom -> createElement("messages");
    $parnode = $dom -> appendChild($node);
    header("Content-type: text/xml");
    while ($row = @mysql_fetch_assoc($result)) {
        // ADD TO XML DOCUMENT NODE
        $node = $dom -> createElement("post");
        $newnode = $parnode -> appendChild($node);
        $newnode -> setAttribute("user", $row['UserName']);
        $newnode -> setAttribute("message", $row['Message']);
        $newnode -> setAttribute("lat", $row['Lat']);
        $newnode -> setAttribute("lon", $row['Lon']);
    }
    echo $dom -> saveXML();
    mysql_close($connection);
}
?>
```

postMsg.php

```
<?php
require ("php_dbInfo.php");
$usr = $_POST["name"];
$message = $_POST["message"];
$expire = $_POST["expire"];
$lat = $_POST["lat"];
$lon = $_POST["lon"];

$connection = mysql_connect(localhost, $username, $password);
if (!$connection) {  exit('1');
}

// Set MySQL database
$db_selected = mysql_select_db($database, $connection);
if (!$db_selected) {
```

```
   exit('1');
}
$r = mysql_query("select max(Id) from Post");
$row = mysql_fetch_array($r);
$id = $row['max(Id)'] + 1;
$postDateArray = mysql_fetch_array(mysql_query("SELECT curdate() as CurrentDate"));
$postDate = $postDateArray['CurrentDate'];
//remove post if there is alredy one in the location
$result = mysql_query('DELETE FROM Post WHERE Lat="' . $lat . '" AND Lon="' . $lon . '"');
//insert into db
$result = mysql_query('INSERT INTO Post (UserName, Message,Lat,Lon,Id,ExpireDate,PostDate)
VALUES ("' . $usr . '","' . $message . '",' . $lat . ',' . $lon . ',' . $id . ',"' . $expire . '","' . $postDate . '")');
if (!$result) {
   die('Invalid query: ' . mysql_error());
}
mysql_close($connection);
echo "0";
?>
```

genMsgXML.php

```
<?php
require ("php_dbInfo.php");
$topLat = $_GET['topLat'];
$bottomLat = $_GET['bottomLat'];
$leftLon = $_GET['leftLon'];
$rightLon = $_GET['rightLon'];

// Start XML file
$dom = new DOMDocument("1.0");
$node = $dom -> createElement("markers");
$parnode = $dom -> appendChild($node);
// Opens connection to MySQL server
$connection = mysql_connect(localhost, $username, $password);
if (!$connection) {  die('Not connected : ' . mysql_error());
}
// Set MySQL database
$db_selected = mysql_select_db($database, $connection);
if (!$db_selected) {
   die('Can\'t use db : ' . mysql_error());
}
$query = "SELECT UserName,Message,Id,Lat,Lon FROM Post WHERE Lat BETWEEN " . $bottomLat . " AND " . $topLat . "
AND Lon BETWEEN " . $leftLon . " AND " . $rightLon;
$result = mysql_query($query);
if (!$result) {
   die('Invalid query: ' . mysql_error());
}

header("Content-type: text/xml");

while ($row = @mysql_fetch_assoc($result)) {
   // ADD TO XML DOCUMENT NODE
   $node = $dom -> createElement("marker");
   $newnode = $parnode -> appendChild($node);
   $newnode -> setAttribute("user", $row['UserName']);
   $newnode -> setAttribute("message", $row['Message']);
   $newnode -> setAttribute("id", $row['Id']);
   $newnode -> setAttribute("lat", $row['Lat']);
   $newnode -> setAttribute("lon", $row['Lon']);
}
```

```
echo $dom -> saveXML();
mysql_close($connection);
?>
```

genFavXML.pho

```
<?php
require ("php_dbInfo.php");
$usr = $_GET["name"];

// Start XML file
$dom = new DOMDocument("1.0");
$node = $dom -> createElement("favourites");
$parnode = $dom -> appendChild($node);
// Opens connection to MySQL server
$connection = mysql_connect(localhost, $username, $password);
if (!$connection) {  die('Not connected : ' . mysql_error());
}
// Set MySQL database
$db_selected = mysql_select_db($database, $connection);
if (!$db_selected) {
   die('Can\'t use db : ' . mysql_error());
}
$result = mysql_query("select * from Fav where Name='" . $usr . "'");
if (!$result) {
   die('Invalid query: ' . mysql_error());
}
header("Content-type: text/xml");
while ($row = @mysql_fetch_assoc($result)) {
   // ADD TO XML DOCUMENT NODE
   $node = $dom -> createElement("fav");
   $newnode = $parnode -> appendChild($node);
   $newnode -> setAttribute("location", $row['LocationName']);
   $newnode -> setAttribute("lat", $row['Lat']);
   $newnode -> setAttribute("lon", $row['Lon']);
}
echo $dom -> saveXML();
mysql_close($connection);
?>
```

addFavourite.php

```
<?php
require ("php_dbInfo.php");
$usr = $_POST["name"];
$locationName = $_POST["locationName"];
$lat = $_POST["lat"];
$lon = $_POST["lon"];

$connection = mysql_connect(localhost, $username, $password);
if (!$connection) {
   die("2");
} else {
   mysql_select_db($database, $connection);
   $result = mysql_query("SELECT LocationName FROM Fav WHERE Name = '" . $usr . "'");
   if ($result) {
      while ($row = mysql_fetch_array($result)) {
```

```
        if ($locationName == $row['LocationName'])
            die("1");
      }
      $result = mysql_query("INSERT INTO Fav (Name, LocationName, Lat,Lon)
VALUES ('" . $usr . "','" . $locationName . "'," . $lat . "," . $lon . ")");
      echo "0";
   }
   if (!$result) {
   die('Invalid query: ' . mysql_error());
}
   mysql_close($connection);
}
?>
```

delFavourite.php

```
<?php
require ("php_dbInfo.php");
$usr = $_POST["name"];
$locationName = $_POST["locationName"];

$connection = mysql_connect(localhost, $username, $password);
if (!$connection) {
   die("1");
} else {
   mysql_select_db($database, $connection);
   $result = mysql_query("DELETE FROM Fav WHERE Name='" . $usr . "' AND LocationName='" . $locationName . "'");
      if (!$result) {
   die('Invalid query: ' . mysql_error());
}
   echo "0";
   mysql_close($connection);
}
?>
```

changeLocation.php

```
<?php
require ("php_dbInfo.php");
$lat = $_POST["lat"];
$lon = $_POST["lon"];
$id = $_POST["id"];

$connection = mysql_connect(localhost, $username, $password);
$db_selected = mysql_select_db($database, $connection);
$query="UPDATE Post SET Lat=".$lat.", Lon=".$lon." WHERE Id=".$id;
$result = mysql_query($query);
mysql_close($connection);
echo "0";
?>
```

php_dbInfo.php

```
<?
$username = "*****";
$password = "*****";
$database = "*****";
?>
```

deletePost.php

```
<?php
require ("php_dbInfo.php");
$user = $_POST["user"];
$id = $_POST["id"];

$connection = mysql_connect(localhost, $username, $password);
$db_selected = mysql_select_db($database, $connection);
$query="SELECT UserName FROM Post WHERE Id=".$id;
$result = mysql_query($query);
$r = mysql_fetch_array($result);
if ($user == $r['UserName'])
{
   //delete post
$query="DELETE FROM Post WHERE UserName='".$user."' AND Id=".$id;
$result = mysql_query($query);
mysql_close($connection);
echo "0";
}
else
{
mysql_close($connection);
echo "1";
}
?>
```

Bibliografia

- World Wide Web Consortium. Geolocation API Specification, http://dev.w3.org/geo/api/spec-source.html, 2012.
- Wikipedia. W3C Geolocation API. http://en.wikipedia.org/wiki/W3C_Geolocation_API.
- Neil Vidyarthi. The Expansion of the Social Media Bubble. http://socialtimes.com/the-expansion-of-the-social-media-bubble_b75432, 2011.
- Wikipedia. Moore's Law. http://en.wikipedia.org/wiki/Moore's_law.
- PHP Online Manual. http://www.php.net/manual/en/, 2012.
- MySQL Reference Manual. http://dev.mysql.com/doc/refman/5.0/en/, 2012.
- Nicholas C. Zakas (Wrox Professional Guides). Professional JavaScript for Web Developers, 2012.
- Sadun Erica (Addison-Wesley). The iPhone developer's cookbook, 2010.
- JavaScript and HTML DOM Reference. http://www.w3schools.com/jsref/default.asp, 2012.
- Paolo Atzeni, Stefano Ceri, Stefano Paraboschi, Riccardo Torlone (McGraw-Hill). Basi di dati - Modelli e linguaggi di interrogazione. Seconda edizione, 2006.
- Larry Ullman (Peachpit press). PHP 6 AND MYSQL 5, 2012.
- jQuery Documentation. http://docs.jquery.com/Main_Page.
- GPS: Accuratezza e precisione. http://www.gpsinfo.it/drupal/node/63, 2011.

Ringraziamenti

Vorrei ringraziare il professor Danilo Montesi per la sua disponibilità e per avermi dato la possibilità di approfondire con questa tesi il mio interesse per lo sviluppo di applicazioni in campo web e mobile, in particolare per la piattaforma iOs.
Ringrazio Ambra per avermi sempre aiutato e sopportato, soprattutto durante la stesura di questa tesi. Ti voglio tanto bene mia musa:3
Ringrazio i miei genitori che mi hanno permesso di coltivare la mia passione per l'informatica con questa laurea universitaria.
Ringrazio anche tutti gli amici e colleghi conosciuti durante questi anni all'università.
Fabio Quinzi, Davide Leonardi e Michele Caldaretti, gruppo ltw0903 sempre carichi per finire dei progetti.
Martino Bonfiglioli e Paolo Ladisa, il Tomorrowland ci aspetta anche quest'anno.
Francesca Guadagnini, grazie per avermi dato ospitalità a Bologna evitandomi tristissime trasferte in treno.
Giacomo Gilli e Matteo Fagiolino, colleghi nel tirocinio e compagni nelle dure sessioni di esami.
E poi Michele F., Lorenzo, Gianluca, Lambo, Nikolas e Marco M., per tutte le esperienze condivise insieme in questi anni. È stato un bel viaggio.