ALMA MATER STUDIORUM

UNIVERSITA' DI BOLOGNA - SEDE DI CESENA

FACOLÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI

CORSO DI LAUREA TRIENNALE IN SCIENZE E TECNOLOGIE INFORMATICHE

thinkmix: dall'idea alla progettazione dello story network

Relazione finale in

Mobile Web Design

Relatore

Dott. Mirko Ravaioli

Presentata da

Luca Comanducci

Sessione Seconda

Anno Accademico 2013 - 2014

Indice

Introduzione

La progettazione e lo sviluppo del network thinkmix, con cui gli utenti possono interagire creando storie di testo e fotografiche saranno gli argomenti di discussione su cui verterà questa tesi.

Per la realizzazione di questo progetto sono stati sviluppati un applicativo per dispositivi mobile, un sito web e il web service, compreso di database, che si interfaccia col client.

Con questa discussione si vogliono illustrare tutti gli step che hanno portato alla realizzazione di thinkmix: la progettazione, lo sviluppo e la pubblicazione.

Il progetto è stato realizzato nel rispetto dell'idea e degli obbiettivi iniziali e il risultato ottenuto, conforme alle aspettative, è un prodotto con buone possibilità di successo sul mercato in virtù delle caratteristiche innovative che lo contraddistinguono.

La diffusione tuttavia si mantiene momentaneamente limitata a causa della mancata attivazione di una vera ed efficace campagna marketing per problemi di budget.

La tesi è strutturata in modo da fornire una panoramica generale del prodotto per focalizzarsi poi su particolari.

Nel capitolo 1 viene presentato il contesto in cui si sviluppa thinkmix offrendo un quadro generale sui dispositivi mobile e altri concetti fondamentali.

Nel capitolo 2 viene mostrato il progetto a livello teorico presentandone la struttura e la fase di progettazione di tutte le sue componenti.

Nel capitolo 3 viene descritta l'implementazione del progetto ricorrendo anche all'utilizzo di schemi e frammenti significativi di codice.

Nel capitolo 4 sono descritte le fasi finali del ciclo di vita del software, vengono esposte le modalità con cui è stato effettuato il testing del prodotto ultimato e infine com'è avvenuto il rilascio.

Il capitolo 5 è dedicato alle conclusioni, viene proposto un ragionamento relativo al budget disponibile in cui si contrappongono aspettative e risultati effettivamente ottenuti. Vengono infine proposte alcune migliorie da apportare al prodotto così da aumentarne la diffusione.

1 - I dispositivi mobile

1.1 - Definizione e introduzione

Il termine *dispositivo mobile* si riferisce in generale a dispositivi elettronici che l'utente può utilizzare in mobilità, caratterizzati per questo da dimensioni e peso ridotti. Ad oggi esistono innumerevoli dispositivi appartenenti a questa categoria, alcuni semplici, studiati per eseguire un solo compito (come i lettori mp3 e i navigatori GPS), altri più complessi dai molteplici scopi e funzionalità.

Negli ultimi anni si è spesso assistito alla nascita di nuovi dispositivi mobile che hanno causato la scomparsa di altri, è il caso degli *smartphone* che hanno ridotto notevolmente la diffusione di lettori mp3, navigatori GPS e console da gioco. Allo stesso modo il mercato dei netbook è stato soppiantato da quello dei *tablet* andando ad intaccare anche quello dei notebook.

In questo settore è in corso un'evoluzione tanto positiva quanto inaspettata dei dispositivi che sono sempre più leggeri, comodi e intuitivi.

Categorizzare i dispositivi mobile è ad oggi un compito complesso sia per la grande varietà disponibile sia perchè alcuni sfuggono ad una netta classificazione per la loro natura ibrida che offre varie possibilità di utilizzo. Di seguito sono illustrate le tipologie di dispositivi mobile:

- Smartphone: telefoni cellulari con sistema operativo evoluto
- Tablet: mini computer che consente l'interazione con lo schermo
- Lettore audio/video: consente di ascoltare canzoni e/o vedere video
- Navigatore GPS: assiste il conducente di un'auto nella navigazione stradale
- Console di gioco: consente di giocare in mobilità
- Foto/Video-camera: consente di scattare e/o catturare video
- Dispositivi indossabili (smartwatch, smartglasses, ...): piccoli dispositivi pensati per essere indossati, rappresentano la più recente forma di dispositivo mobile.

1.2 - Funzionalità e situazione attuale

Lo smartphone è al momento il dispositivo mobile che ha conosciuto la crescita più dirompente finendo per oscurare completamente altri prodotti, ne è un esempio il caso della calcolatrice il cui utilizzo è ormai relegato a concetti specifici soppiantato nel quotidiano dal più pratico e preciso smartphone.

A riprova di tale disparità di sviluppo è sempre più diffusa l'erronea tendenza di utilizzare il termine dispositivo mobile semplicemente come sinonimo di smartphone.

I settori economici più disparati sono stati profondamente condizionati dall'introduzione di tali dispositivi progettati per opsitare un evoluto sistema operativo dotato di uno store che permette in ogni momento di scaricare applicazioni con cui accrescere le funzionalità del dispositivo stesso.

Pertanto è in corso un conseguente adattamento di tale modello di distribuzione ad altri dispositivi anche maggiormente compessi quali PC e console da gioco.

Un'altra caratteristica imprescindibile dei dispositivi mobile è lo schermo touchscreen associato a un'interfaccia software intuitiva al fine di sostituire l'ormai obsoleto pennino.

Ogni dispositivo mobile completo è corredato da componenti come fotocamere, sistema gps, connettività cellulare e wifi, altoparlanti e microfoni, componenti di calcolo(cpu, gpu, ...) molto performanti.

Le componenti di calcolo dei più recenti dispositivi mobile sono talvolta paragonabili a personal computer, infatti è in corso la graduale convergenza di queste due categorie fra le quali sono presenti differenze sempre minori. A riprova di ciò è il tentativo di Microsoft di proporre prodotti a metà tra tablet e personal computer tramite l'utilizzo di Windows 8, seppure con scarsi risultati. Al contrario Apple ha dichiaratamente scelto di adottare per ora una strategia più conservatrice mantenendo le due categorie ben distinte.

Quello dei dispositivi mobile è un settore in continua crescita che sta trainando l'evoluzione di una parte consistente dell'informatica. Il fatto che i dispositivi mobile siano per definizione piccoli e leggeri spesso è un assoluto vantaggio per i produttori agevolati nell'applicazione di nuove tecnologie ai prodotti. Ne sono un esempio i

nuovi schermi dotati di una definizione sempre maggiore e i sensori del touchscreen sempre più precisi e reattivi, queste stesse tecnologie non sono attualmente applicabili a prodotti di più cospique dimensioni. Viceversa il maggiore svantaggio di questi dispositivi è la durata della batteria, quasi mai soddisfacente.

1.3 - I sistemi operativi mobile

I più moderni dispositivi mobile sono gestiti da sistemi operativi altrettanto evoluti, dotati di numerose funzionalità ed elevata efficienza. Nel tempo l'impegno dei produttori dei vari sistemi operativi è stato quello di mantenersi all'avanguardia presentando prodotti completi, innovativi, corredati da sempre maggiori servizi.
Benchè a confronto con altri prodotti informatici il lavoro alla base dei moderni sistemi operativi possa sembrare meno articolato questi sono effettivamente complessi e ottimizzano la gestione delle risorse limitate dei dispositivi garantendo un'interazione con l'utente assolutamente semplice e agevole. Proprio la mancanza di un tempestivo sviluppo volto a perseguire tale obbiettivo ha scalzato dai vertici aziende come Microsoft e Nokia.
Ad ora iOS e Android sono i sistemi operativi mobile maggiormente utilizzati tuttavia non è possibile fare una stima effettiva in modo da poter condurre un confronto equo. Sul web sono disponibili numerosi studi riguardanti il numero di attivazioni, di download di applicazioni e di device venduti, ma l'affidabilità delle fonti è dubbia e le discrepanze nei risultati dei diversi studi sono notevoli.
I due sistemi sono basati su idee fondamentalmente diverde, iOS è chiuso e limitato, non consente modifiche o personalizzazioni, propone però applicazioni e un sistema sempre reattivo e stabile garantendo continui aggiornamenti e sicurezza. Android invece è un sistema aperto, le tipologie di applicazioni disponibili sono molto più ampie e le possibilità di personlaizzazione sono tante, ne consegue però una qualità del software talvolta inferiore e un'estrema frammentazione.
Un ulteriore sistema operativo mobile, seppur non ancora al livello dei precedenti, è Windows Phone. Questo sistema operativo dotato di notevoli potenzialità di sviluppo nasce come successore di Windows Mobile rispetto al quale però offre nu approccio assolutamente nuovo rivolto ad un pubblico più vasto grazie soprattutto ad una veste

grafica moderna e minimale e ad una buona integrazione con gli altri prodotti Microsoft.

I principali sistemi operativi mobile sono elencati in tabella:

	iOS	**Android**	**Windows Mobile**	**Windows Phone**	**BlackBerry OS**	**Symbian**	**Bada**
Compagnia	Apple	Open Handset Alliance (Google Inc.)	Microsoft	Microsoft	RIM	Symbian Foundation	Samsung
Versione corrente	7.0	4.3	6.5.5	8	7.1	9.5	2.0
Famiglia di sistemi operativi	Mac OSX/Unix-like	Linux	Windows CE 5.2	Windows NT	Mobile OS	Mobile OS	Mobile OS
Architettura CPU supportata	ARM	ARM, MIPS, Power Architecture, x86	ARM	ARM	ARM	ARM, x86	ARM
Linguaggio di Programmazione	C, C++, Objective C	C, C++, Java	C++	C++	Java	C++	C++
Application store	App Store	Google Play	Marketplace	Marketplace	BlackBarry World	OVI Store	Samsung App

Tabella 1 - I principali sistemi operativi mobile

[1]

1.3.1 - iOS

Viene introdotto da Apple per la prima volta nel 2007 con il nome di iPhone OS come sistema operativo per il primo iPhone diventato solo in un secondo tempo iOS con la versione 4.0.

iOS oggi è il sistema operativo di iPhone, iPod touch, iPad e Apple TV. Come Mac OSX è una derivazione di UNIX e l'unica architettura supportata è quella ARM. Più precisamente sembra che Apple tenda a personalizzare i chip ARM così da migliorarne l'efficienza e l'integrazione con iOS.

Questo sistema operativo ha il merito di aver dato il via alla rivoluzione del settore mobile. La presentazione del primo iPhone durante la conferenza di apertura del Macworld nel gennaio 2007 tenuta dall' ex CEO Steve Jobs viene già ricordata come un evento storico. iPhone si presentava come un qualcosa di estremamente innovativo discostandosi molto da qualsiasi altro tipo di telefono cellulare o palmare già esistenti. Le numerose novità tecnologiche (quali il grande schermo capacitivo e l'utilizzo dell'accelerometro) e soprattutto l'innovativo sistema operativo furono sufficienti per farlo diventare subito un successo.

Nel corso del tempo sono stati rilasciati numerosi aggiornamenti e 7 major release, dalla 1.x alla 7.x . Nonostante iOS sia installato solo su dispositivi Apple, proprio come OSX solo su Mac, gli *iDevice* (termine utilizzato in gergo per definire i dispositivi Apple) sono tra i dispositivi più diffusi. Le limitazioni di questo sistema operativo sono numerose e le applicazioni pubblicate su App Store vengono sottoposte ad un severo controllo di qualità e dei contenuti per garantire il rispetto di tutte le linee guida fornite da Apple.

Ciò implica un rallentamento nei tempi di pubblicazione nonchè l'esercizio di atti di censura da parte dell'azienda, tuttavia consente a questo sistema di garantire applicazioni sicure e di qualità: la maggior parte delle applicazioni su App Store sono curate in dettaglio sia a livello funzionale sia grafico.

Ulteriori limitazioni sono la mancanza di un file system navigabile e l'utilizzo dell'approccio Sand Boxing per le applicazioni, introdotte al fine di garantire immediatezza e sicurezza.

Per l'attivazione, la gestione e il backup di un dispositivo che utilizzi iOS è sempre stato necessario l'installazione sul computer di iTunes (software Apple per la gestione di musica, video e iDevice). Solo dalla versione 5.x di iOS è possibile prescindere dall'utilizzo di iTunes per tali scopi, questo passo avanti è stato fondamentale per rendere gli iDevice definitivamente indipendenti dai computer. iCloud è un importante servizio inserito gratuitamente in iOS che oltre a garantire l'indipendenza da iTunes, consente anche la sincronizzazione dei propri dispositivi Apple rendendoli maggiormente "uniti" semplificandone l'utilizzo.
La creazione di software per iOS avviene tramite il tool di sviluppo *Xcode*. Questo tool è lo stesso utilizzato per la programmazione per Mac OSX e questo fatto sommato alle similitudini fra i due sistemi operativi e al rilascio di un SDK ben curato ha fatto si che molti sviluppatori attivi sulla piattaforma Mac OSX passassero senza difficoltà a quella iOS. La pubblicazione di applicazioni su App Store è consentita a seguito della sottoscrizione di una licenza da sviluppatore e richiede l'utilizzo di Xcode e dei portali *iTunesConnect* e *iOSDevCenter* per "firmare" le applicazioni. Spesso i numerosi vincoli imposti da Apple risultano problematici, in modo particolare per i piccoli sviluppatori, vincolati all'acquisto di una licenza e di Mac recenti da mantenere sempre aggiornati insieme ai tool di sviluppo. Nonostante ciò gli sviluppatori indipendenti sono molto numerosi e App Store ha fatto la fortuna di molti di essi [2].

1.3.2 - Android

Android Inc. è stata fondata nell'ottobre 2003, il 17 agosto 2005 Google ha acquisito l'azienda al fine di entrare nel mercato della telefonia mobile. Android è caratterizzato da una struttura open source e si basa sul kernel linux, la licenza Apache (presente sui sorgenti) permette a chiunque di modificare e distribuire liberalmente il sistema operativo. A differrenza di iOS, Android viene installato su vari dispositivi, proprio questa caratteristica ne ha consentito una diffusione rapida e cospiqua tale da renderlo il sistema operativo più utilizzato. Tale approccio comporta tuttavia una qualità del software non sempre impeccabile e un'enorme frammentazione. Proprio quest'ultimo è il problema che Google sta assolutamente cercando di risolvere: il tentativo con le ultime versioni di Android è quello di offrire

ai produttori degli standard e delle linee guida che garantiscano la possibilità di reazioni più rapide al rilascio di ogni nuova versione del sistema operativo.
Google si è comportata allo stesso modo per quanto riguarda il Marketplace, l'application store di Android nel quale la pubblicazione delle applicazioni risulta semplice e veloce. Queste non vengono sottoposte a controlli particolarmente rigidi, inoltre sui device Android è possibile installare applicazioni di terze parti ottenendo il file d'installazione con estensione apk. In questo modo viene facilitata la proliferazione di applicazioni a scapito di qualità, sicurezza e tutela degli sviluppatori.
Per la creazione di software per Android, Google mette a disposizione un SDK e un emulatore che devono essere associati a un IDE come ad esempio *Eclipse*, quello supportato ufficialmente. La programmazione per Android può quindi avvenire su qualunque sistema x86: Windows, Linux e OSX [3].

1.3.3 - Windows Phone

Windows Phone è il sistema operativo Microsoft presentato nel febbraio 2010, è il successore di Windows Mobile del quale però non conserva particolari aspetti.
Soprattutto è cambiato il mercato di riferimento col passaggio da quello enterprise a quello consumer, ciò è messo in evidenza dal nuovo stile grafico minimale e semplificato e dalla rimozione di alcune caratteristiche. Windows Phone risulta quindi radicalmente diverso dai predecessori apportando la svolta più radicale con l'annullamento della retro compatibilità. Microsoft ha però dotato il suo sistema operativo mobile delle più importanti funzioni come l'ottima integrazione con i social e la presenza di software come Internet Explorer e Office. La compagnia sceglie la via dell'integrazione tra i propri dispositivi in maniera ancora più aggressiva di Apple tentando persino di dare lo stesso "look" a tutti i suoi software e facendo evolvere di pari passo il settore mobile (con Windows Phone), quello dei personal computer (con Windows 8) e quello delle console (con Xbox).
La distribuzione di Windows Phone è stata seguita da Microsoft con estrema attenzione nel tentativo di acquisire tutti i pregi dai due avversari (Android e iOS).
La compagnia non soltando produce dispositvi ma consente l'installazione del proprio sistema operativo anche sui dispositivi di altri produttori, imponendo però il

rispetto di regole ferree. Gli aggiornamenti inoltre sono molto rapidi così che i produttori riescono ad adeguare i propri prodotti più rapidamente di quanto avvenga con Android. In questo modo è possibile garantire una buona qualità e coerenza dei dispositivi mantenendo comunque varietà. Nonostante ciò Windows Phone fatica ancora ad avvicinarsi ad Android e ad iOS, forse per la natura troppo moderna del sistema oppure per l'application store, il Marketplace, che non è all'altezza.
Il solo produttore che riesce al momento a trarre guadagno dall'utilizzo di Windows Phone sui propri dispositivi è Nokia: grazie alla partnership con Microsoft (rivelatasi produttiva per entrambi) Nokia è in grado di produrre dispositivi di qualità, ben integrati con il sistema. In questo modo sta poco a poco riusciendo a ritagliarsi una fetta di mercato [4].

1.3.4 - BlackBarry OS

BlackBarry OS è un sistema proprietario della RIM (Research In Motion). Questo sistema operativo aveva originariamente un target ben preciso, quello aziendale, essendo l'unico in grado di offrire una serie di servizi e proprietà tali da renderlo la sola soluzione per i dispositivi utilizzati in ambito lavorativo. A tale scopo ha sempre fatto della sicurezza e della posta elettronica il proprio punto di forza. Attualmente però i sistemi operativi mobile più diffusi si stanno adeguando offrendo una qualità paragonabile rendendo così BlackBarry OS un sistema di nicchia. RIM ha più volte tentato invano di risollevare il brand effettuando modifiche, anche radicali, nel sistema operativo e proponendo dispositivi mobile moderni ma senza un riscontro positivo in termine di vendite e diffusione sul mercato [5].

1.3.5 - Symbian

Symbian è il più antico sistema operativo con funzioni moderne quali il multitasking e l'esecuzione di videogiochi con grafica 2D. Nasce nel 1998 ed è prodotto da Symbian Foundation. Diventa molto popolare soprattutto grazie a Nokia che lo utilizza per un lungo periodo sulla maggior parte dei suoi dispositivi di fascia alta garantendogli così un dominio incontrastato, interrotto in seguito all'introduzione dei moderni smartphone. Nel 2008 Nokia ha acquistato il sistema operativo e lo ha modificato così da renderlo competitivo. Tuttavia il tentativo si rivela fallimentare e

dal 2010 Symbian è diventato un sistema operativo libero. Attualmente Symbian sta sparendo dalla scena dal momento che non sono più stati presentati prodotti che lo utilizzano [6].

1.3.6 - Bada

Bada è una piattaforma sviluppata da Samsung con un kernel ibrido che poggia sul kernel Linux, FreeBSD e un kernel real-time OS. In questo modo è applicabile a una vasta gamma di dispositivi. Fa il suo debutto nel 2010 con il Samsung Wave e conosce subito un discreto successo. Questo sistema operativo dimostra come Samsung stia tentando di staccarsi da Android creando un proprio ecosistema, purtroppo però i riscontri tardano ad arrivare e si teme già che il supporto a Bada venga abbandonato [7].

2 - Il progetto "thinkmix"

2.1 - L'idea

L'idea nasce dalla voglia di creare una piattaforma innovativa e basata sulla produttività ma che conservi i più elementari aspetti social. Thinkmix può per alcune sue caratteristiche essere definito un social network, da questo modello tuttavia prende solo alcuni aspetti quali i *follower* e la *bacheca*, si tratta dunque propriamente più di una piattaforma che consente agli utenti di leggere, votare e partecipare alla creazione di storie. Una storia su thinkmix è intesa come un susseguirsi di "frammenti" ciascuno proposto da un utente con la sua personale idea riguardo l'evolversi dei fatti. Attraverso il voto a tutti gli utenti è data la possibilità di scegliere il ramo della storia che preferiscono, in questo modo ogni storia avrà un solo inizio, ma tanti possibili sviluppi. E' possibile creare due diverse tipologie di storie: Testo, con un limite di 300 caratteri per frammento, oppure Foto.

Tale struttura conferisce alle storie un'estrema variabilità perchè chiunque può contribuirvi. La complessità dietro questa idea deve essere necessariamente semplificata e astratta a dovere durante la programmazione in modo da rendere il più possibile intuitive le novità che thinkmix introduce.

L'idea si adatta quindi facilmente a un utilizzo mobile poichè i frammenti sono brevi. Si è scelto così di puntare sulla piattaforma iOS tralasciando in un primo momento le altre per creare poi in una fase successiva delle semplici pagine web che permettessero la sola lettura e condivisione delle storie sui social in modo da avvicinare al progetto un pubblico più vasto possibile.

2.2 - Idee simili già esistenti e differenze

Prima e durante lo sviluppo di thinkmix si è cercato di tener d'occhio la "concorrenza" per valutare se ci fossero prodotti simili che potessero interferire. In fase iniziale è stata rilevata la presenza di solo alcuni progetti con caratteristiche simili, ma si trattava per lo più di forum o siti web sconosciuti. Durante la fase finale dello sviluppo invece ha iniziato ad emergere *20lin.es*, un progetto molto simile a thinkmix dal quale lo differenziano solo alcuni particolari tra cui l'impossibilità di

creare storie fotografiche, un limite di caratteri molto maggiore per ciascun frammento e una lunghezza prefissata per le storie nel loro insieme. 20lin.es inoltre è attualmente sviluppato sotto forma di sito web mentre l'applicazione è utilizzabile per la sola lettura delle storie, in maniera esattamente opposta a thinkmix [8].

2.3 - Struttura del progetto

Il progetto è suddivisibile in tre parti:

- Server: il cervello del servizio che elabora i dati, li memorizza e fornisce le informazioni necessarie ai client.
- Applicazione: il client iOS con cui si interfacciano gli utenti, invia e riceve dati dal Server fornendo un'astrazione del sistema tale da rendere facilmente utilizzabili tutte le funzioni.
- Sito Web: il client web che consente la lettura e la condivisione delle storie create dagli utenti tramite l'applicazione. Sono presenti inoltre altre pagine di supporto agli utenti: privacy e condizioni d'uso, domande frequenti, descrizione e link all'app.
- Portale amministratori: poche pagine web che consentono agli amministratori di gestire i contenuti di thinkmix.

L'attenzione nei prossimi paragrafi verrà focalizzata principalmente sui primi due punti avendo il sito web e il portale amministratori importanza decisamente minore.

2.4 - Progettazione dell'applicazione

Nella fase iniziale della progettazione dell'applicazione sono state definite le caratteristiche principali del progetto ed è stata scelta l'interfaccia utente. Si è cioè proceduto a stilare un elenco delle principali funzionalità che il servizio doveva offrire:

1. Registrarsi
2. Effettuare il login
3. Cambiare password
4. Modificare dati personali
5. Cercare storie per titolo
6. Cercare utenti per nome

7. Consultare la bacheca
8. Consultare la sezione "post consigliati"
9. Creare nuove storie (testo o foto)
10. Rispondere a una storia
11. Votare frammento di storia
12. Scegliere proseguimento preferito
13. Iniziare a seguire un altro utente
14. Controllare le notifiche
15. Cancellare l'account

In corso d'opera a questo primo modello sono state apportate notevoli modifiche essendosi evoluto il progetto iniziale specialmente nella logica di "costruzione" delle storie.

E' seguita la realizzazione in cartaceo dell'interfaccia iOS sulla base dell'elenco qui riportato; infine si è scelto di costruire la parte server progettando il database.

La progettazione dell'applicazione non ha seguito un programma rigoroso e completo e si è posta più volte la necessità di ricominciare il lavoro da zero o di effettuare modifiche radicali. Ciò è da imputarsi al fatto che inizialmente l'idea non era abbastanza matura e perchè crescesse con uno sviluppo adeguato è stato necessario procedere per tentativi seguiti da una valutazione all'utilizzo pratico.

2.4.1 - Meccaniche di creazione delle storie, primo tentativo

Come si può vedere dalla prima versione dell'elenco, inizialmente la meccanica di costruzione delle storie si basava sulla scelta dell'utente ideatore della storia che poteva scegliere i proseguimenti che riteneva migliori: se un utente A scriveva un frammento di storia e un utente B rispondeva, allora solo l'utente A aveva la possibilità di scegliere come prosecuzione la risposta data da B. In questo modo c'era il rischio che le storie rimanessero bloccate, inoltre le risposte di altri utenti, se non scelte, avrebbero perso di significato.

Questi problemi sono emersi chiaramente alla prima prova di questa versione dell'applicazione, durante la progettazione invece sarebe stato difficile capirli e correggerli con precisione.

2.4.2 - Meccaniche di creazione delle storie, secondo tentativo

Nella seconda versione dell'applicazione sono state riesaminate e corrette le meccaniche che nel test della prima erano state giudicate insoddisfacenti. Si è scelto così di eliminare la possibilità di scegliere il proseguimento preferito (punto 12) lasciando tutte le possibili prosecuzioni sempre visibili, senza che nessuno potesse scegliere i frammenti migliori. Semplicemente veniva mostrato il ramo di storia più lungo lasciando comunque la possibilità di navigare fra gli altri rami.
Questa soluzione sembrava decisamente migliore della precedente, ma la lunghezza delle storie non rappresentava un indice di qualità, col rischio di avere storie lunghe ma raramente sensate e coerenti.

2.4.3 - Meccaniche di creazione delle storie, terzo ed ultimo tentativo

L'obbitettivo perseguito con la terza versione è stato quello di mantenere l'elasticità conferita dalla seconda mostrando però agli utenti il "ramo migliore" delle storie senza inibire la navigazione tra gli altri.
A tale scopo si è deciso di introdurre un tipo di votazione semplice ma efficace che potesse garantire la qualità delle storie. Votare un frammento di storia (punto 11) significava inizialmente dare la possibilità agli utenti di esprimere un semplice voto sui singoli frammenti, con il solo scopo di mostrare loro quali fossero quelli più apprezzati. Questa funzionalità è stata poi così modificata: gli utenti non dovevano più eseguire una singola votazione ma votare interi rami della storia. Si mantiene così l'opportunità di navigare tramite l'interfaccia tra i diversi rami, raggiungere l'ultimo frammento di un ramo e votarlo positivamente.
Il "peso" di un ramo è dato dal numero di voti registrati nell'ultimo frammento, quindi per ogni frammento viene memorizzato un valore che rappresenta il peso del ramo dalla radice al frammento stesso. Il ramo con maggior peso diventa il ramo migliore, quello mostrato quando l'utente accede alla storia. Quando "nasce" un nuovo frammento questo eredita il voto di quello a cui risponde. In questo modo non viene data importanza alla lunghezza della storia, ma solo al voto degli utenti.
Quest'ultima soluzione si è rilevata la migliore durante le prove, riuscendo a garantire massima elasticità e qualità, il ramo migliore infatti può variare

continuamente, seppure con la tendenza a stabilizzarsi per la difficoltà di creare un ramo parallelo altrettanto apprezzato.

2.4.4 - Evoluzione dell'interfaccia

Anche la progettazione dell'interfaccia, è stata più volte rimaneggiata. Molte delle modifiche grafiche apportate sono legate semplicemente ad un fattore estetico, ma alcune modifiche all'interfaccia sono state necessarie per adeguare il progetto alle meccaniche delle varie versioni.

Figura 1 - Prima versione, scheda di ricerca

Figura 2 - Prima versione, scheda di scelta proseguimento

Con una serie di screenshot si vuole qui mostrare come l'app si sia evoluta nel tempo.

In Figura 1 e 2 si nota l'interfaccia elementare della prima versione dell'app che dava la possibilità all'utente di scegliere il proseguimento per la propria storia. Sono visibili inoltre, per ciascun frammento due pulsanti, uno per visualizzare le risposte e uno per votare il frammento. In Figura 1 si nota come le azioni principali fossero rese disponibili: in alto i pulsanti per il profilo e per le notifiche, in basso le schede Amici per visualizzare la bacheca, Scopri per vedere i post consigliati e Cerca per trovare storie o altri utenti.

Figura 3 - Seconda versione dell'interfaccia

Figura 4 -Terza versione dell'interfaccia

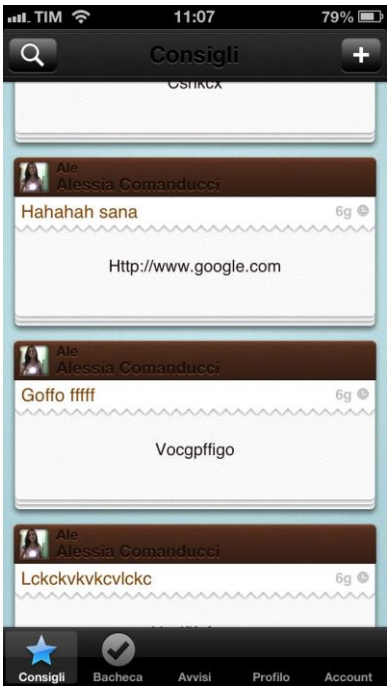


Figura 5 - Quarta versione dell'interfaccia

Figura 6 - Quinta versione dell'interfaccia

Le figure riportate qui rappresentano versioni intermedie della grafica dell'applicazione attraverso cui si è passati per raggiungere la forma ottimale.

2.4.5 - Le schermate dell'applicazione finale

La versione definitiva della grafica di thinkmix, come si può vedere, è caratterizzata da uno stile minimal unito ad elementi di decororo (fogli, puntine, tessuto ecc) che rendano l'applicazione più famigliare all'utilizzatore.

Si è scelto di utilizzare una struttura "classica" tipica delle applicazioni iOS con una barra nella parte inferiore contenete tutti i collegamenti alle varie schede dell'applicazione e una in alto per navigare tra le schermate.

Figura 7 - Schermata di Login

Figura 8 - Schermata di Registrazione nuovo account

L'applicazione si apre con una breve introduzione che spiega il servizio, a seguito una schermata introduttiva contenente alcune storie consigliate così da consentire anche agli utenti non registrati di visualizzare le storie e comprendere del tutto il funzionamento senza offrire però alcuna possibilità di interazione. L'unico pulsante messo a loro disposizione è quello per raggiungere la schermata "Login" (Figura 7) dalla quale è possibile effettuare l'accesso oppure creare un accont. Se si sceglie di creare un accont si accede alla schermata di Registrazione (Figura 8) nella quale è richiesta la compilazione di alcuni campi obbligatori: email, nome, congnome, nickname, password e conferma, anno di nascita (è necessario essere maggiorenni), sesso, paese, viene inoltre richiesto di leggere e accettare i documenti di Privacy e Condizioni d'uso. Confermata la registrazione l'utente dovrà verificare l'indirizzo email cliccando su un link inviatogli per mail.

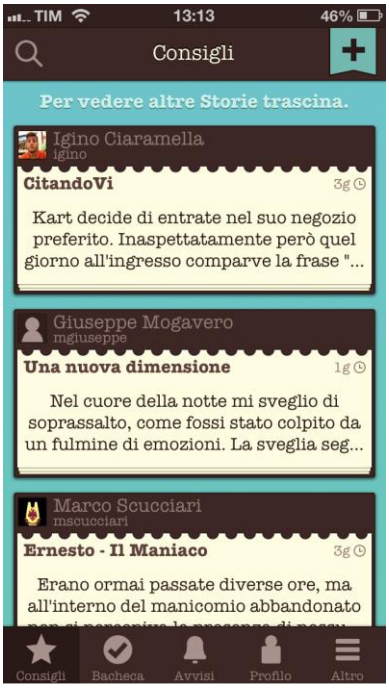


Figura 9 - Schermata Consigli

Effettuato il login compare la schermata principale dell'applicazione e viene mostrata la scheda "Consigli" (Figura 9).

Fra le storie consigliate si trovano sia le storie più votate sia quelle più recenti, precisamente vengono mostrate 30 storie di cui:

1. 5 scelte randomicamente tra le 100 storie con maggior peso di sempre.
2. 20 scelte randomicamente tra le 500 storie con maggior peso dell'ultimo mese.
3. 5 scelte randomicamente tra le storie pubblicate negli ultimi tre giorni.

Lo scopo è quello di dare la giusta importanza alle nuove storie, le quali verranno mantenute visibili se riusciranno ad avere abbastanza successo. Tutto ciò avviene facendo attenzione che le storie scelte nei tre punti non si ripetano.

Questi criteri di scelta sono adatti per una prima fase in cui si suppone che ci siano pochi utenti e poche storie.

Per ogni storia sono mostrati avatar, nome, cognome e nickname dell'utente che ha scritto il primo frammento, poi la data e un'anteprima. Selezionandone una si aprirà la schermata successiva che mostra la storia.

Nella schermata di dettaglio, in Figura 10, è possibile leggere e proseguire la storia creata da altri utenti, scorrendo la schermata fino in fondo (Figura 11) vengono offerte diverse possibilità: votare il ramo, proseguire la storia aggiungendo un frammento e riordinarla, accedere cioè a una schermata che permette di scegliere passo passo, sin dall'inizio, ogni frammento.

Figura 10 - Ramo migliore di una storia

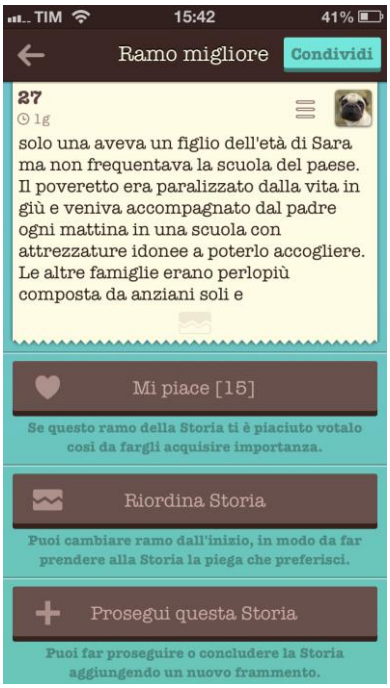


Figura 11 - Azioni del ramo

Figura 12 - Cambio di ramo

Tra un frammento e l'altro un pulsante in trasparenza dà la possibilità di "spezzare" in quel punto la storia. Se è presente più di un proseguimento è possibile cambiare ramo, in questo caso il pulsante diventa marrone e ben visibile così che l'utente possa utilizzarlo per raggiungere la schermata in Figura 12. Qui tramite uno slide orizzontale sarà possibile visualizzare tutti i possibili proseguimenti e scegliere il preferito fino ad arrivare alla fine del ramo.

Le Figure 13 e 14 mostrano le schermate visualizzate quando si comincia una nuova storia o se ne prosegue una già cominciata. Sono possibili l'inserimento di un contenuto testuale o grafico e di un titolo. Per le storie di testo viene mostrato il contatore di caratteri.

Figura 13 - Schermata nuova storia di Testo

Figura 14 - Schermata nuova storia di Foto

Figura 15 - Schermata Cerca

Figura 16 - Schermata Avvisi

In Figura 9 in alto a sinistra è visibile un pulsante a forma di lente d'ingrandimento per l'accesso alla schermata "Cerca" (Figura 15) dalla quale è possibile cercare storie oppure altri utenti.

Dalla schermata principale è anche possibile accedere alla scheda "Avvisi" (Figura 16) in cui è possibile vedere tutte le notifiche relative al proprio account:

- Un altro utente ha iniziato a seguirti
- La storia che hai creato ha ricevuto il primo proseguimento
- Il ramo che hai creato ha ricevuto il primo proseguimento

Figura 17 - Schermata Profilo

La Figura 17 mostra la schermata "Profilo" da cui l'utente può controllare le proprie storie iniziate e quelle a cui partecipa. Può inoltre modificare l'avatar scattando una foto o scegliendone una dalla propria libreria.

Sono inoltre presenti due pulsanti, uno per mostrare gli utenti che seguono e uno per mostrare gli utenti seguiti. Questo aspetto è comune a molti social network: se l'utente A inizia a seguire l'utente B allora A vedrà nella sua bacheca le storie create da B (e da tutti gli altri utenti che segue) in ordine cronologico.

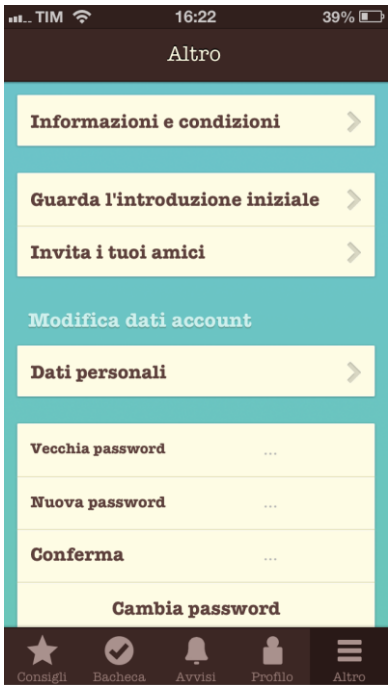


Figura 18 - Schermata Altro

Nella scheda Altro (Figura 18) si trovano i collegamenti e le impostazioni dell'applicazione e dell'account:

- Informazioni e condizioni: è data qui la possibilità di rileggere i documenti accettati in fase di registrazione, viene inoltre mostrato un terzo documento riguardante i ringraziamenti e l'elenco delle librerie usate.
- Guarda l'introduzione iniziale: permette di rivedere l'introduzione esplicativa del servizio.
- Invita i tuoi amici: apre una schermata di invio mail precompilata che aiuta l'utente a coinvolgere altre persone.
- Dati personali: dà accesso ad una schermata in cui si possono modificare i dati personali.
- Cambia password: dà la possibilità di modificare la password senza effettuare il logout immettendo quella vecchia poi per due volte quella nuova.
- Logout: l'applicazione dimentica i dati personali e le credenziali dell'utente e ripresenta la schermata iniziale dei consigli con la sola possibilità di visualizzare storie ed effettuare il login.
- Cancella account: rende possibile la cancellazione dell'account. L'utente viene in tal caso avvisato più volte che l'operazione è definitiva, dopo la conferma il server ne cancellerà tutti i dati personali compreso l'avatar, ma i contenuti creati dallo stesso rimarranno su thinkmix così da non invalidare le storie.

2.5 - Progettazione server

La parte server di thinkmix comprende l'insieme delle pagine web che permettono la comunicazione con l'applicazione e il database, queste sono collocate su un server dedicato acquistato su aruba.it con la formula Basic 2.2 [9].
La comunicazione client/server è costruita attorno a una libreria php già pronta quindi non è stato necessario effettuare una progettazione vera e propria. Per il database invece è stata condotta una progettazione approfondita.

2.5.1 - Comunicazione Client/Server

La comunicazione tra il client e il server avviene tramite continue richieste da parte dell'applicazione. Non vi è quindi una connessione continuamente attiva, ma l'applicazione contatta il server periodicamente e quando ha necessità di ricevere informazioni. Il server in questo modo riesce a servire un numero maggiore di utenti, stabilire connessioni con molti dispositivi e tenerle attive potrebbe infatti significare un carico troppo elevato, inoltre i dispositivi mobile sono soggetti a continue disconnessioni a causa dell'instabilità del segnale cellulare.
Per piccoli dati, come nel caso di thinkmix, il tempo necessario per stabilire la connessione equivale circa al tempo totale di risposta, perchè il tempo di trasmissione dei dati è quasi impercettibile. Per minimizzare il numero di connessioni si è scelto di raggruppare, quando possibile, le varie "richieste" dell'applicazioni spedendole al server come una. In questo modo i dati che devono essere trasmessi (o richiesti) senza urgenza vengono messi in attesa e raggruppati con altri per essere spediti insieme successivamente.
In particolare una richiesta è una POST http al server con la quale si passa un file JSON strutturato in modo da contenere le informazioni suddivise per tipologia di azione.

2.5.2 - Struttura JSON

Il formato JSON (JavaScript Object Notation) è basato sul linguaggio JavaScript ed è utilizzato per memorizzare i vari tipi di dati utilizzati nei linguaggi di programmazione, supporta fra gli altri booleani, numeri, stringhe, array, array associativi.

Si è scelto l'utilizzo del formato JSON perchè adatto all'ambito mobile grazie alla leggerezza che ne velocizza la trasmissione e alla semplicità con cui è possibile leggere e modificare un file che utilizzi questo formato.
Segue un esempio di file JSON di richiesta:

```
{
	"actions":[
		{
			"action" : "nuovo_post_di_testo",
			"data" : {
				"title" : "Titolo storia",
				"text_content" : "C'era una volta..."
			}
		}
	],
	"username" : "pippo@gmail.com",
	"password" : "lamiapassword"
}
```

In questa richiesta esemplificativa si vuole inviare un pacchetto per creare una nuova storia di testo. L'array associativo principale contiene gli oggetti:

- "actions" : rappresenta un array che può contenere più "action", ognuna delle quali rappresenta un'azione identificata con una parola chiave, qui "nuovo_post_di_testo", che permetta al server di capire quale azione si vuole eseguire, in "data" invece sono contenuti i dati utilizzati dal server per eseguire l'azione in questione.
- "username" e "password" : possono essere omessi, è necessario che siano specificati solo in caso le azioni soprastanti necessitino il login dell'utente.

Di seguito invece l'esempio di JSON di risposta:

```
{
	"response" : {
		"nuovo_post_di_testo" : true
	},
	"errors" : [],
	"success" : true,
	"message" : ""
}
```

Il server risponde alla richiesta inoltratagli in cui l'array associativo principale contiene gli oggetti:

- "response" : un array associativo che per ogni azione restituiscie dei dati, qui un booleano per confermare che l'azione è stata eseguita correttamente.
- "errors" : in caso si riscontrino problemi durante l'esecuzione delle azioni, vengono qui riportati i messaggi di errore con un codice identificativo.
- "success" : identifica la riuscita generale della richiesta indipendentemente dalle action.
- "message" : se success è false allora questo oggetto contiene il messaggio di errore che spiega il problema verificatosi (esempio: "sintassi JSON non riconosciuta").

[10]

2.5.3 - Progettazione database

Una buona progettazione del database è stata fondamentale date le limitate risorse e l'importanza di garantire risposte rapide da parte del server nel caso di numerosi utenti.

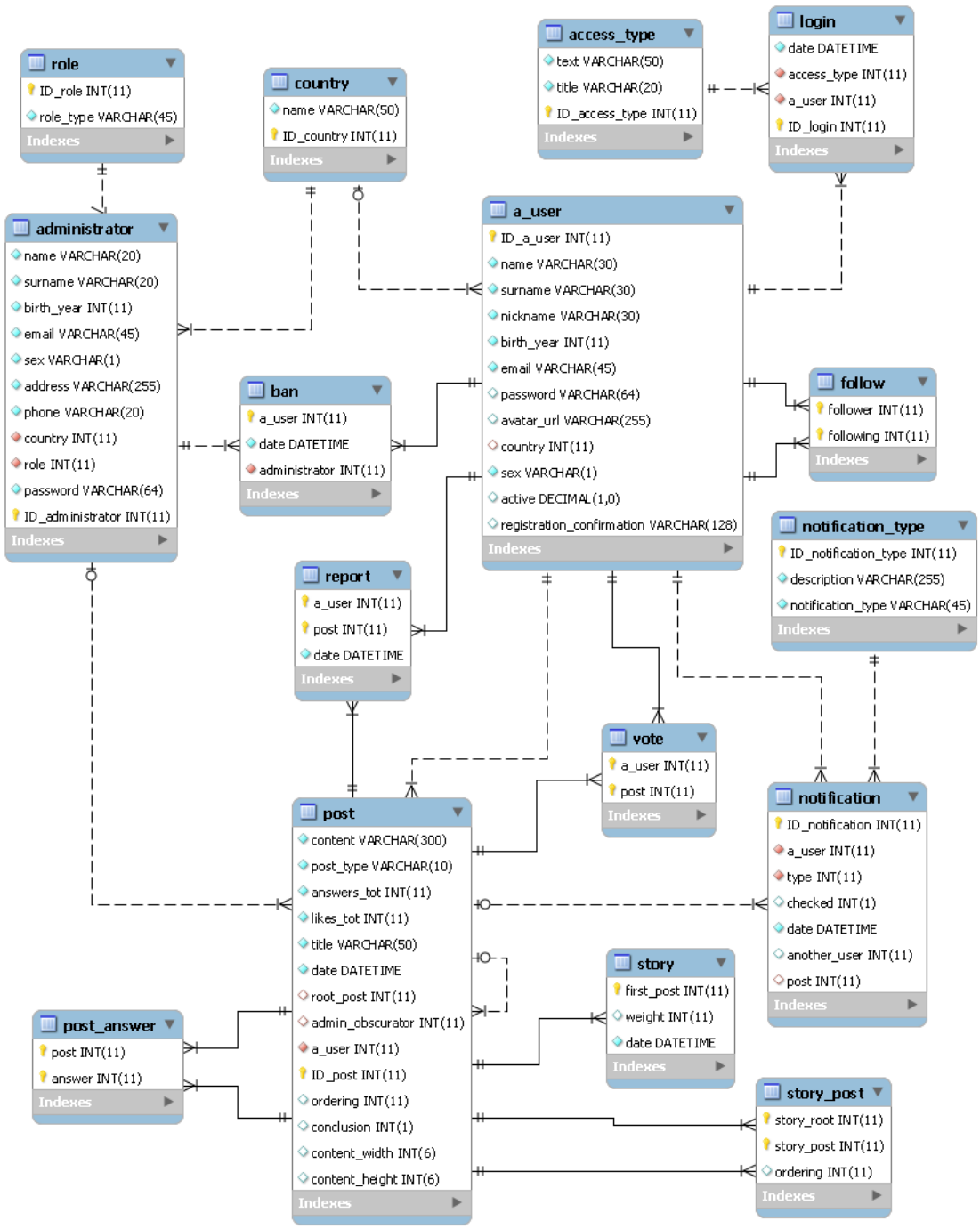


Figura 19 - database

Lo schema in Figura 19 rappresenta il database attualmente utilizzato da thinkmix.
Segue un'illustrazione delle componenti:

- administrator: rappresenta persone che hanno particolari privilegi su thinkmix possono bannare un utente oppure oscurare i frammententi di storie che siano stati segnalati.
- role: ruoli in cui sono suddivisi gli amministratori.
- ban: rappresenta il ban di un utente con memorizzazione di data e amministratore.
- report: ogni utente può segnalare i frammenti, così che gli amministratori decidano se questi debbano essere oscurati.
- a_user: rappresenta l'utente utilizzatore di thinkmix.
- country: contiene la lista di paesi di residenza che utenti e amministratori possono scegliere durante la registrazione.
- login: ogni volta che un utente effettua un login questo viene memorizzato con le relative data e ora.
- access_type: ogni login può essere di più tipi, iOS, Web, Android. Momentaneamente è disponibile soltanto la prima tipologia.
- follow: memorizza quando un utente ne segue un altro.
- post: rappresenta un frammento di una storia, l'attributo "root_post" vale NULL quando il frammento è l'incipit della storia, se invece è valorizzato assume l'ID del primo Post.
- vote: permette di memorizzare i voti degli utenti sui singoli frammenti.
- post_answer: permette di capire la gerarchia dei frammenti ovvero di concatenare una risposta con il relativo frammento "padre".
- story_post: memorizza interamente i rami migliori delle storie, in particolare ogni tupla è composta dall'ID del primo frammento, dall'ID di un altro frammento facente parte del ramo e dall'attributo "ordering" che ne memorizza la posizione rispetto agli altri frammenti del ramo.
- story: memorizza gli ID di tutti i primi frammenti, affiancando a questi la data dell'ultima modifica alla storia e il peso del ramo migliore.

- notification: ogni utente può ricevere notifiche riguardanti storie a cui partecipa oppure altri utenti.
- notification_type: a seconda del tipo di notifica l'avviso riguarderà un post o un utente e conterrà la relativa descrizione.

Le varie ridondanze, riscontrabili dallo schema, sono state introdotte al fine di semplificare il lavoro e migliorare le prestazioni complessive. Ad esempio in "post" si ha l'attributo "likes_tot", che potrebbe essere ottenuto tramite l'entità "vote", ma si è comunque ritenuta preferibile la sua introduzione. Stesso discorso vale per l'attributo "answers_tot".

L'entità "story_post" è stata introdotta per memorizzare il ramo migliore di una storia, evitando in questo modo di effettuare una query ricorsiva utilizzando "post_answer", procedura questa che avrebbe reso inaccettabili le prestazioni in presenza di un numero di frammenti elevetao.

L'entità "story" è stata realizzata al fine di bypassare la memorizzazione degli attributi "date" e "weight" altrove con l'introduzione di ulteriori ridondanze, inoltre in questo modo l'esecuzione di azioni quali la generazione della lista di storie consigliate risulta estremamente performante.

2.6 - Sito Web

Il sito web si compone di poche semplici pagine, tutte con la stessa struttura.

Figura 20 - Sito web, home

Il alto è presente una barra marrone con il logo e cliccando su questo si torna alla pagina principale (Figura 20). A sinistra una colonna con la spiegazione del servizio e il collegamento all'iTunes Store per il dowload dell'app, a destra i pulsanti per la condivisione della pagina sui principali social e una sezione per i banner pubblicitari. Al centro vi è il contenuto. In Figura 20, ad esempio, si può vedere la pagina principale il cui contenuto è la lista delle storie più votate.

Figura 21 - Sito web, storia

Cliccando su una storia fra quelle dell'elenco la si può visualizzare ma senza navigare fra i rami, infatti viene visualizzato solamente il migliore, come mostra la Figura 21.

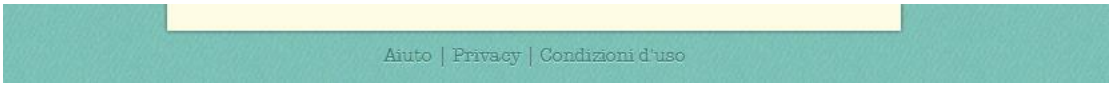


Figura 22 - Sito web, footer

A fondo pagina sono sempre presenti i collegamenti ai documenti di privacy e condizioni d'uso e alla pagina "Aiuto" dove si trovano le istruzioni per contattare il supporto e le domande frequenti.

2.7 - Portale amministratori

Il portale amministratori, è basilare, composto da alcune pagine web, ed è necessario per semplificare il compito di moderatore su thinkmix.

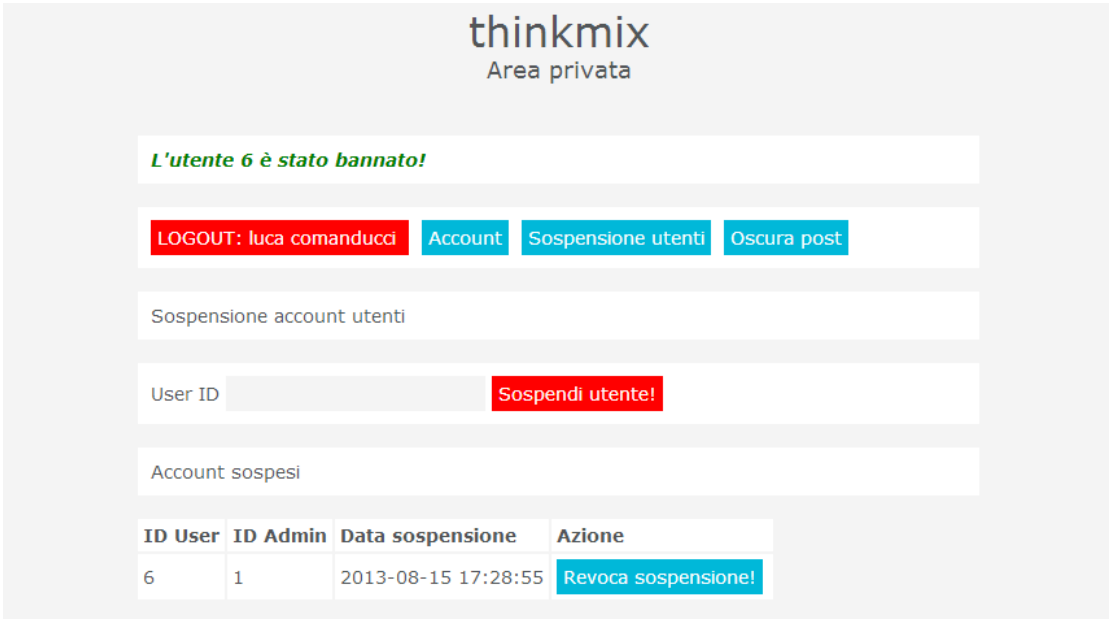


Figura 23 - Portale amministratori, Sospensione utenti

In Figura 23 si vede la pagina che consente di sospendere gli utenti inserendo il relativo ID. Nell'area sottostante la lista degli utenti sospesi con la possibilità di riabilitarli istantaneamente.

Figura 24 - Portale amministratori, Oscura post

In Figura 24 è mostrata la pagina da cui è possibile oscurare i frammenti segnalati dagli utenti. L'amministratore ha la possibilità di vedere il contenuto del frammento segnalato e decidere con i relativi pulsanti se oscurarlo o meno.

2.8 - Revenue model

Ad ora l'applicazione e tutti i servizi anessi vengono fortini gratuitamente, la sola fonte di guadagno è rappresentata da banner pubblicitari presenti nell'applicazione e sul sito web posizionati in modo da non interferire con l'interfaccia [11].

Questo revenue model scarsamente efficace rappresenta solo un'opzione momentanea in attesa che thinkmix cresca tanto da consentire l'introduzione di soluzioni alternative che richiedono però un bacino di utenti più ampio.

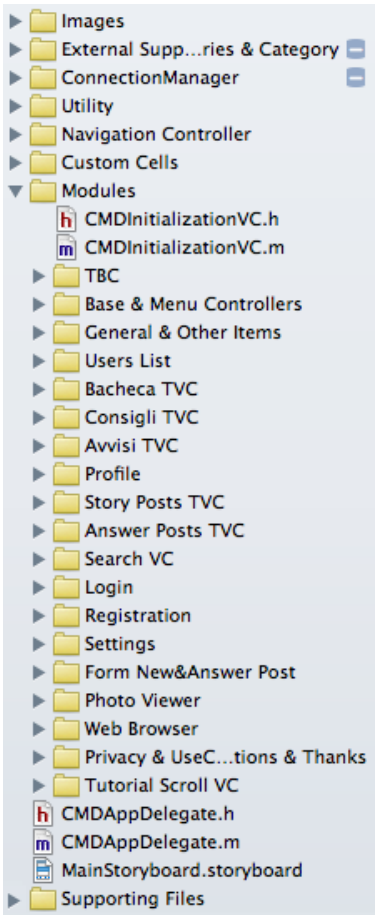
3 - Implementazione

3.1 - Implementazione client iOS

Nella fase di implementazione del client iOS sono state necessarie abilità nella programmazione e buone capacità grafiche.

Per la realizzazione dell'applicazione si è utilizzato il tool di sviluppo Xcode reso disponibile da Apple con il linguaggio di programmazione Objective-C. thinkmix utilizza la versione 5.0 dell'SDK nonostante sia disponibile la più recente 7.0, per essere compatibile con la maggior parte dei dispositivi iOS [12]. Al momento l'applicazione non è ottimizzata per iPad, dunque su tale dispositivo potrà essere avviata ma non avrà un aspetto grafico studiato per l'ampio schermo di cui è dotato.

3.1.1 - Organizzazione del progetto



In Figura 25 è mostata l'organizzazione dei file e delle classi nel progetto, di seguito sono illustrate le componenti:

- La cartella "Images" contiene le immagini utili per creare la grafica dell'applicazione. Spesso è preferibile utilizzare immagini PNG in sostituzione di librerie grafiche che potrebbero compromettere la reattività delle schermate.
- La cartella "External Sup...ries & Category" contiene le librerie e le classi esterne al progetto sulla maggior parte delle quali si basano le classi di connessione e di caching.
- La cartella "ConnectionManager" contiene le classi di connessione create appositamente per gestire le richieste asincrone al server di thinkmix.
- "Utility" contiene piccole classi interne al progetto utili per lo sviluppo di altre.

- "Navigation Controller" contiene le classi che ereditano da UINavigationController e altre classi accessorie.
- "Custom Cells" contiene le classi che ereditano da UITableViewCells. La maggior parte delle schermate dell'applicazione offre uno scorrimento verticale grazie alla presenza di una tabella (classe UITableView) suddivisa in righe ciascuna delle quali è occupata da un oggetto cella. Queste celle sono il cuore dell'interfaccia dell'applicazione, ad esempio, in Figura 8, i campi sono suddivisi in celle di tipo CMDMenuCell. In Figura 9, invece, le storie sono tre celle di tipo CMDStoryCell.
- "Modules" contiene tutti i controller dell'applicazione.
- "CMDAppDelegate" è la classe che monitora le transizioni di stato dell'applicazione mentre è in esecuzione; ad esempio ha il compito di rispondere agli eventi che rappresentano l'apertura, la chiusura e l'inattività dell'applicazione stessa.
- "MainStoryboard" è il solo *Storyboard* utilizzato nell'applicazione.
- "Supporting Files" raggruppa file d'impostazione, file conteneti le macro e i testi localizzati.

3.1.2 - Storyboard e principali controller

Lo Storyboard è uno strumento di Xcode introdotto da Apple con la versione 4.2 come evoluzione del vecchio *Interface Builder*, offre la possibilità di creare le principali schermate dell'applicazione e metterle in relazione tra loro attraverso un'elementare rappresentazione grafica.

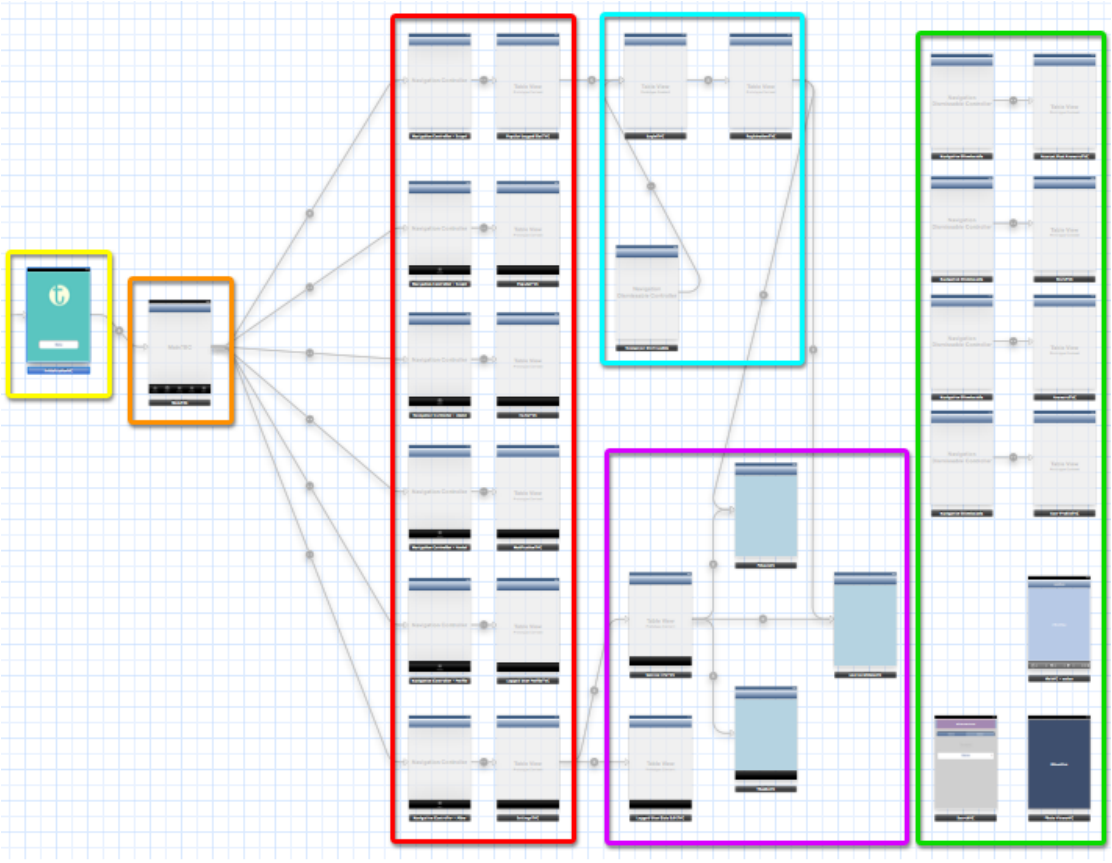


Figura 26 - Storyboard

In Figura 26 è mostrato lo Storyboard completo dell'applicazione, segue la legenda per esplicare l'associazione dei colori dei rettangoli alle schermate in essi contenute:

- Giallo: è la schermata che compare in fase di inizializzazione dell'applicazione per cui viene eseguita una prima richiesta al server, si tratta della classe CMDInitializationVC che eredita da UIViewController.
- Arancione: rappresenta il controller "contenitore" che consente l'accesso alle principali schermate (controller) dell'applicazione tramite una tab bar, si tratta della classe CMDMainTBC che eredita da UITabBarController.

- Rosso: contiene i principali controller visualizzati nell'applicazione e i relaviti navigation controller. Le classi di questi controller ereditano tutte da UITableViewController e sono CMDHomeTVC, CMDPopularTVC, CMDNotificationTVC, CMDLoggedUserTVC, CMDSettingsTVC.
- Azzurro: contiene i controller utili per la registrazione e il login, le classi sono CMDRegistrationTVC e CMDLoginTVC ed ereditano da UITableViewController.
- Viola: contiene i controller che mostrano alcune informazioni e documenti utili come il documento di privacy e quello condizioni d'uso. Le principali classi utilizzate sono CMDPrivacyVC, CMDUseConditionsVC, CMDThanksVC ed ereditano da UIViewController.
- Verde: contiene altri controller (e navigation controller) non direttamente collegati alla schermata principale. Si tratta di schermate dedicate alla visualizzazione e alla modifica delle storie e alla visualizzazione del profilo degli utenti. Le classi principali sono CMDStoryTVC, CMDAnswersTVC, CMDUserProfileTVC ed ereditano da UITableViewController.

I navigation controller presenti nello schema sono controller che ne "inglobano" altri, gestendo il passaggio tra diverse schermate. Nell'applicazione sono rappresentati tramite la classe CMDNavigationController che eredita da UINavigationController.

L'utilizzo dello Storyboard è stato ridotto al minimo ed è stato sfruttato solamente per dare chiarezza alla struttura del progetto e per marcare le relazioni tra i principali controller. L'interfaccia delle schermate è stata realizzata invece tramite codice, questo sistema, infatti, garantisce un'interazione più profonda.

3.1.3 - Gestione della connessione

Nell'applicazione thinkmix la gestione della connessione assume un ruolo fondamentale poichè i dati visualizzati nelle varie schermate devono essere sempre aggiornati. Per garantire con il server uno scambio di dati funzionale, asincrono e continuo è stato creato un pacchetto di classi che astraessero la complessità, lasciando ai controller il solo compito di visualizzare le informazioni.

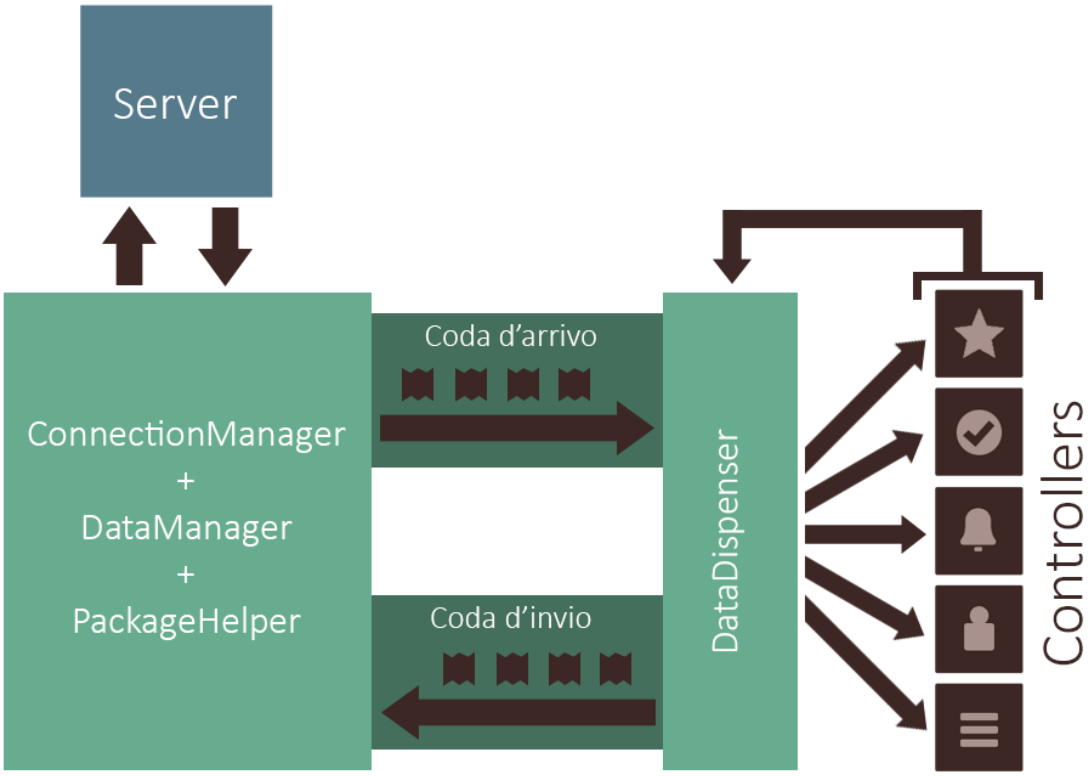


Figura 27 - Gestione della connessione nel clinet iOS

In Figura 27 è mostrato come opera il pacchetto sopracitato. Con il colore verde sono evidenziate le principali classi utili per la connessione e la distribuzione delle informazioni, mentre con il colore marrone sono rappresentati i controller e il flusso dei dati.

La classe MLConnectionManager è la componente principale di questo pacchetto, si tratta di un singleton che gestisce e coordina le altre classi, questa contiene i metodi per eseguire le richieste al server, il login e il logout.

Tramite l'utilizzo di un timer il connection manager avvia periodicamente la procedura di richiesta al server, a questo punto l'istanza della classe MLPackageHelper si occupa di creare il file JSON raggruppando le richieste effettuate dai vari controller e quelle di default. Queste ultime sono riferite a semplici azioni, quali l'aggiornamento della bacheca o del profilo personale, si tratta dunque di richieste che devono venir eseguite ad ogni scatto del timer.
La classe MLDataManager si occupa di trasformare le informazioni così da renderle comprensibili all'applicazione, se provengono dal server, e viceversa al server, se provengono dall'applicazione. Il data manager entra in gioco quindi sia durante l'invio, sia durante la ricezione dei dati, inoltre è contattato anche dal package helper durante la costruzione del JSON.
Un'altra classe fondamentale è MLDataDispenser, questa si occupa di dialogare con i controller distribuendo loro le informazioni e ricevendo le richieste.
A fronte di una risposta da parte del server, i dati ricevuti verranno tradotti in oggetti dal data manager, il quale li inserirà, suddivisi per tipologia, nella coda d'arrivo (oggetto di tipo MLTSQueue), sarà poi il dispenser che si occuperà di prendere questi pacchetti e distribuirli ai controller pulendo eventualmente la coda.
Ogni controller e ogni pacchetto sono affiancati da una stringa identificativa che rappresenta l'azione da eseguire. In questo modo il dispenser conosce esattamente la destinazione dei pacchetti.
I controller possono creare pacchetti di richiesta consegnandoli al dispenser, questo poi li inserisce nella coda d'invio in attesa che il data manager li traduca in testo JSON comprensibile al server e il package helper li raggruppi, costruendo il file completo. Anche in questo passaggio la stringa identificativa è fondamentale perchè nel JSON diventa la chiave "action" già vista nel capitolo riguardante la progettazione.
Ovunque nell'app è possibile eseguire una "forceRequest" cioè avviare una richiesta senza attendere il timer, inoltre i controller, creando i pacchetti di invio, possono scegliere se inviarli con o senza le richieste di default, forzando o meno la richiesta. Spesso si rende utile eseguire una richiesta immediata senza includere i pacchetti di default che rallenterebbero la risposta del server.

Per effettuare le connessioni con il server viene utilizzata la libreria "AFNetworking" che supporta richieste con contenuto JSON e, tramite l'utilizzo dei blocchi di codice, fornisce risposte asincrone.

Segue un frammento di codice appartenente alla classe MLConnectionManager:

```
- (void)sendRequest:(NSDictionary *)data
{
    NSURL *url = [NSURL URLWithString:BASE_SERVER_URL];
    AFHTTPClient *httpClient = [[AFHTTPClient alloc] initWithBaseURL:url];
    httpClient.parameterEncoding = AFJSONParameterEncoding;
    NSMutableURLRequest *request = [httpClient requestWithMethod:@"POST"
path:@"" parameters:data];
    AFJSONRequestOperation* operation;

    operation = [AFJSONRequestOperation
                 JSONRequestOperationWithRequest:request
                 success:^(NSURLRequest *request, NSURLResponse *response,
id JSON)
                 {
                     // sblocca il data manager che libera la coda d'invio
                     [dataManager dataUseSuccess];
                     [MLPackageHelper responseChecker:JSON];

                     // eseguo il blocco
                     successBlock();
                     // dopo di che lo svuoto
                     successBlock = ^{};

                     [self startSendTimer];
                 }
                 failure:^(NSURLRequest *request, NSURLResponse *response,
NSError *error, id JSON)
                 {
                     // sblocca il datamanager
                     [dataManager dataUseFail];

                     [self startSendTimer];
                 }];
    [operation start];
}
```

Si tratta del metodo che esegue la richiesta al server: nelle prime righe di codice viene costruita la richiesta utilizzando le classi della libreria sopracitata.

I blocchi "success" e "fail" contengono il codice che dovrà essere eseguito in caso si abbia un successo o un fallimento della richiesta.

In caso di successo viene liberata la coda d'invio e richiamato il package helper per preparare i dati per il data manager.

3.1.4 - Funzionamento dei controller

I controller sono gli oggetti che gestiscono le schermate dell'applicazione, ricevono quindi i dati e dopo averli elaborati li visualizzano.

Nell'applicazione tutti i controller che abbiano necessità di ricevere o inviare dati tramite il pacchetto di connessione adottano il protocollo UpdatableControllerProtocol:

```
- (NSString*)getIdentifier;
- (void)clearContent;
- (BOOL)blockForUpdateWithData:(NSMutableArray*)newData;
```

Ogni controller dovrà quindi fornire tramite il metodo "getIdentifier" una stringa identificativa, rappresentata dall'azione cui è collegato. Il metodo "clearContent" ha invece il compito di riportare il controller allo stato iniziale e viene principalmente utilizzato dal connection manager che, prima di eseguire il logout, effettua la pulizia di tutti i controller così da prepararli alla sessione successiva.

Il metodo utilizzato dal dispenser per contattare i controller e passare loro il pacchetto di dati tramite il parametro newData è "blockForUpdateWithData:". Questo metodo ritorna un valore booleano: il controller può rifiutarsi di ottenere i dati restituendo "NO", in questo caso il dispenser non eliminerà il pacchetto dalla coda d'arrivo e ritenterà in un secondo momento.

Segue un'implementazione esemplificativa all'interno del controller CMDPostPopularTVC:

```
- (NSString*)getIdentifier {
    // ritorno la define dei post popolari
    return ACTION_POST_POPULAR;
}

- (BOOL)blockForUpdateWithData:(id)newData {

    if (newData) {      // se i dati sono presenti
        // li tipizzo ed eseguo il filtro
        receivedItem = (CMDPostPopularItemReceived *)newData;
        receivedItem.posts = [[self class] filterPosts:receivedItem.posts];
    }
    // ricarico la tabella con i dati aggiornati e blocco l'animazione di
loading
    [self.tableView reloadData];
    [self stopLoadIndicator];

    // dati accettati
    return YES;
}
```

```
- (void)clearContent {

    // basta resettare i dati e ricaricare la tabella
    receivedItem = [[CMDPostPopularItemReceived alloc] init];
    [self.tableView reloadData];

    return;
}
```

In questo caso "blockForUpdateWithData:", richiamato, controlla se i dati esistono, in caso affermativo questi sono "salvati" ed è eseguito un filtro che elimina eventuali storie di tipo non riconosciuto. Viene poi ricaricata la tabella che si aggiornerà con i dati contenuti in "receivedItem" e viene fatto ritornare "YES" per comunicare al dispenser che i dati sono stati accettati e che può procedere all'eliminazione dalla coda d'arrivo.

La maggior parte dei controller dell'applicazione ereditano da UITableViewController, sono dunque basati su una struttura tabellare dove ogni elemento è rappresentato per riga. Di seguito è mostrato il codice necessario alla visualizzazione dei dati ricevuti:

```
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView {
    return 1;
}

- (NSInteger)tableView:(UITableView *)tableView
numberOfRowsInSection:(NSInteger)section {
    return receivedItem.posts.count;
}

- (float)tableView:(UITableView *)tableView
heightForRowAtIndexPath:(NSIndexPath *)indexPath {
    return [CMDStoryCell cellHeight]; // ritorna l'altezza che è sempre
uguale
}

- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath {
    CMDAPostItemReceived *item = [receivedItem.posts
objectAtIndex:indexPath.row];

    static NSString *idCell = @"StoryCell";

    CMDStoryCell *cell = [tableView
dequeueReusableCellWithIdentifier:idCell];
    if (cell == nil)
        cell = [[CMDStoryCell alloc]
initWithStyle:UITableViewCellStyleDefault reuseIdentifier:idCell];
    // passo i dati alla cella e questa li visualizza
    [cell setCellWithPost:item];
```

```
    return cell;
}
- (void)tableView:(UITableView *)tableView
didSelectRowAtIndexPath:(NSIndexPath *)indexPath {

    CMDAPostItemReceived *item = [receivedItem.posts
objectAtIndex:indexPath.row];

    UIStoryboard *storyboard = [UIStoryboard
storyboardWithName:@"MainStoryboard" bundle:[NSBundle mainBundle]];

    id vc;

    if (item.IDRootPost == 0)
        vc = [storyboard instantiateViewControllerWithIdentifier:@"STORY"];
    else
        vc = [storyboard instantiateViewControllerWithIdentifier:@"ANSWER"];

    [vc setFirstPost:item];

    [vc setHidesBottomBarWhenPushed:YES];
    [self.navigationController pushViewController:vc animated:YES];
}
```

I primi tre metodi sono necessari per generare la tabella con le dimensioni corrette, questa, infatti, è suddivisa in sezioni (ovvero gruppi di righe) e righe. Si parla, qui, di una sola sezione e tante celle quanti sono gli elementi (le storie) contenuti in receivedItem.

Con il metodo "tableView: heightForRowAtIndexPath:" vengono costruite e visualizzate le celle, vengono cioè riempite le righe della tabella. In questo metodo (richiamato ciclicamente per ogni riga) si istanzia una cella di tipo CMDStoryCell alla quale viene passato l'oggetto "item", contenete i dati relativi alla storia rappresenta in modo che possa visualizzare dati come il titolo, l'anteprima e la data.

Quando l'utente esegue un tap su una cella viene richiamato "tableView: didSelectRowAtIndex:", come per gli altri metodi il parametro "indexPath" contiene gli indici di sezione e riga ai quali si riferisce l'evento. Una cella, quando viene selezionata procede mostrando il controller di dettaglio. Come si nota dal codice, tramite lo Storyboard si ottiene il controller necessario cui vengono passati i dati relativi alla storia, infine viene mostrato tramite il metodo "pushViewController: animated:" del navigation controller.

3.2 - Implementazione Server

La realizzazione della parte server è avvenuta utilizzando l'editor notepad++ e il linguaggio di programmazione php [13].

L'implementazione del server è stata portata avanti parallelamente all'implementazione dell'applicazione. La struttura delle pagine adibite alla comunicazione con il client iOS è costruita attorno a una libreria già pronta, la quale rappresenta la controparte del pacchetto di connessione presente nell'applicazione. Questa libreria è scritta in php e non solo gestisce la connessione con un client mobile, ma offre anche una visione più ampia dell'intero progetto ponendosi come struttura di base dello stesso, essendo organizzata per ospitare pagine php sia per la comunicazione con un client esterno e per pagine web. Questa mette a disposizione numerosi servizi, tra i quali una comunicazione con il database semplificata e l'upload di immagini, che automatizza il salvataggio di copie di diverse dimensioni.

Per quanto riguarda la comunicazione mobile questa struttura richiede che venga creata una classe per ogni azione a cui il server deve rispondere. Si tratta di classi che estendono la classe "Action", la quale contiene i metodi e gli oggetti comuni ad ogni azione.

Segue il codice della classe "PostBachecaAction.class.php":

```
<?php
class PostBachecaAction extends Action {

    function __construct() {
        parent::__construct(true);
    }

    public function doExecute(&$response) {

        $query = ''.
        getPostElemSelectQuery().
        'FROM POST p, a_user u '.
        'WHERE u.ID_a_user IN ( '.
        '   SELECT following '.
        '   FROM follow '.
        '   WHERE follower = ? '.
        ')  '.
        'AND p.root_post IS NULL '.
        'AND u.ID_a_user = p.a_user '.
        'ORDER BY p.date DESC '.
        'LIMIT 50';

        $ps = Slash::database()->prepare($query);
```

```
            $ps->setInt(1, $this->utente->getId());
            $rs = $ps->execute();

            $list = array();

            while ($row = $rs->each()) {

                $elem = getPostElem($row, $this->utente->getId());
                array_push($list, $elem);
            }
            $response['post_bacheca'] = $list;
        }
}
$_ACTION_MAP['post_bacheca'] = new PostBachecaAction();
?>
```

Questa classe segue la struttura tipica di ogni action, nella prima funzione viene richiamato il metodo "__construct()" della classe che è stata estesa, il booleano che viene passato indica se l'azione necessita che l'utente sia loggato, se il valore è "false" verrà controllata la presenza delle credenziali e in caso queste non fossero presenti l'azione non verrà eseguita e la risposta conterrà l'errore relativo.

La funzione "doExecute" è invece riempita con il codice necessario all'esecuzione dell'azione. Qui è possibile elaborare i dati ricevuti, se presenti, ed effettuare le operazioni sul database. In caso si incontri un errore è possibile interrompere l'azione e aggiungere un messaggio alla risposta per avvisare l'utente. In caso contrario, questo metodo ritornerà i dati richiesti, sarà poi la libreria ad occuparsi di inviarli al client, sotto forma di risposta JSON.

3.3 - Implementazione Sito Web

L'implementazione del sito web si basa su poche e semplici pagine web. E' presente il file "_base.php" contenente la struttura comune alle pagine de sito, "_content-util.php" con funzioni utili e tutte le pagine accessibili dal broswer come "index.php" e "storyGenerator.php". Queste pagine si limitano a generare il contenuto, ad esempio accedendo al database e utilizzando le funzioni in "_content-util.php", richiamano poi la funzione "printFinalPage()" appartenente a "_base.php" passando il contenuto sotto forma di stringa. Questa genera semplicemente il codice html inserendo il contenuto nella posizione corretta.

4 - Testing e rilascio

La fase di testing si è sviluppata in diversi momenti. Il collaudo delle varie componenti e funzionalità è stato effettuato durante lo sviluppo in modo da evitare una concatenazione di errori semplificando il lavoro finale; a progetto ultimato è stato possibile effettuare un testing più approfondito che ne valutasse il funzionamento globale. In particolare è stato necessario coinvolgere persone esterne simulando un utilizzo del network con più dispositivi contemporaneamente.

La pubblicazione dell'applicazione thinkmix su App Store è avvenuta nel giugno 2013 [14].

Il sito web è stato reso disponibile in seguito all'acquisto del dominio "thinkmix.net" [15].

5 - Conclusioni e Sviluppi futuri

Il risultato del progetto è nel complesso soddisfacente. L'idea iniziale è stata realizzata in modo conforme rispetto il budget disponibile. Tuttavia l'impossibilità di avviare una campagna marketing adeguata è andata a compromettere le possibilità di guadagno. In fase di progettazione, infatti, è stato curato in modo preponderante lo sviluppo del software, a discapito di una corretta focalizzazione sul revenue model.

Ad ora si contano un centinaio di utenti, molti dei quali hanno espresso pareri positivi e soddisfazione. Ciò incoraggia un eventuale futuro rilancio del prodotto.

I primi aggiornamenti di thinkmix, rilasciati dopo la pubblicazione, si sono concentrati sull'introduzione di servizi che facilitino la diffusione del prodotto fidelizzando gli utenti che già lo utilizzano. Anche gli aggiornamenti futuri saranno pensati per perseguire tale fine, sarà inoltre necessario introdurre un nuovo revenue model accompagnato da una robusta campagna marketing.

Bibliografia

[1] Wikipedia, "Sistema operativo per dispositivi mobili", http://it.wikipedia.org/wiki/Sistema_operativo_per_dispositivi_mobili, 2013

[2] Wikipedia, "iOS (Apple)", http://it.wikipedia.org/wiki/IOS_(Apple), 2013

[3] Wikipedia, "Android", http://it.wikipedia.org/wiki/Android, 2013

[4] Wikipedia, "Windows Phone", http://it.wikipedia.org/wiki/Windows_Phone, 2013

[5] Wikipedia, "BlackBarry OS", http://it.wikipedia.org/wiki/BlackBerry_OS, 2013

[6] Wikipedia, "Symbian OS", http://it.wikipedia.org/wiki/Symbian, 2013

[7] Wikipedia, "Bada", http://it.wikipedia.org/wiki/Bada, 2013

[8] 20lines, http://it.20lines.com, 2013

[9] Aruba, "Server dedicati & Housing", http://serverdedicati.aruba.it/server-dedicati/basic, 2013

[10] "Introducing JSON", http://www.json.org/, 2013

[11] Google, "Google AdSense", https://www.google.com/adsense, 2013

[12] Apple, "iOS Developer Library", https://developer.apple.com/library/ios, 2013

[13] The PHP Group, "PHP Manual", http://www.php.net/manual, 2013

[14] Apple, "iOS Dev Center", https://developer.apple.com/devcenter/ios, 2013

[15] Aruba, "Registra un dominio", http://hosting.aruba.it, 2013