Politecnico di Milano

Progetto Di Ingegneria del Software 2

SWIMv2

Design Document

Autori:
Emanuele Uliana
(799256)
Gabriele Rufolo
(743695)
Walter Rubino
(742519)

Professori:
Raffaela Mirandola

22/12/2012

Indice

1 Generali

1.1 Breve Panoramica del sistema

Il sistema SWIMv2 è pensato per tutte quelle persone che hanno bisogno di un professionista in un determinato ambito e desiderano cercarlo online. Per accedere a tutte le funzionalità e per motivi di sicurezza sono richiesti la registrazione ed il login: entrambe le procedure sono molto semplici, in quanto l'interfaccia della web app è progettata per essere il più user-friendly possibile. Tutte le informazioni che gli utenti forniscono nella fase di registrazione sono memorizzate in un database di tipo relazionale (mysql) e, per ragioni di praticità e sicurezza non sono visibili in chiaro, bensì sotto forma di hash salato (vedi la sezione "Sicurezza"); in particolare la password è crittografata con crittografia !!! TODO : DA DEFINIRSI !!!

1.2 Scopo di questo documento

Lo scopo di questo documento è quello di descrivere nel dettaglio, anche tramite l'ausilio di diagrammi E-R, UML, UX, ecc. , lo scheletro del sistema, i suoi vari componenti (database, interfaccia utente e logica applicativa), le tecnologie utilizzate e la gestione della sicurezza, il tutto opportunamente spiegato anche minuziosamente ove è richiesto per una ottimale comprensione di SWIMv2.

1.3 Assunzioni e rettifiche

Nello scrivere questo documento abbiamo deciso di precisare alcune nostre assunzioni rispetto al sistema e alla progettazione:

- Gli amministratori del sistema sono e saranno sempre i tre autori di questo articolo.
- Il set di abilità iniziali predefinito di sistema è costituito da:
  - Ingegnere del Software
  - Master dei videogiochi "Final Fantasy"
  - Esperto di Computer Security
  - Esperto di Crittografia

  - Esperto di calcio mondiale
  - Esperto di Probabilità e Statistica
  - Esperto della società Mozilla
  - Esperto di automobili
  - Esperto di Apple
  - Esperto di Linux e dell'open source

- Un ospite non può chiedere aiuto da nessuna parte: può solo effettuare ricerche tra gli utenti registrati.
- Conseguenza del punto precedente è che non esiste una bacheca pubblica come inizialmente da noi ipotizzato.

1.4 Glossario

In questo documento vengono usate le seguenti sigle/abbreviazioni:

- HW: Hardware
- SW: Software
- DD: Design Document
- PP: Project Planning
- RASD: Requirements Analysis and Specification Document
- DB: Database
- J2EE: Java Enterprise Edition
- UX: User Experience (diagram)
- DBMS: DataBase Management System
- AS: Application Server

Sono tutte sigle abbastanza note, ma, a scanso di equivoci è sempre meglio precisare (a cominciare dal fatto che questa sezione non è un filler, ma ha una sua utilità).

2 Software impiegato per il documento e per il sistema

Per scrivere questo documento sono stati impiegati i seguenti software:

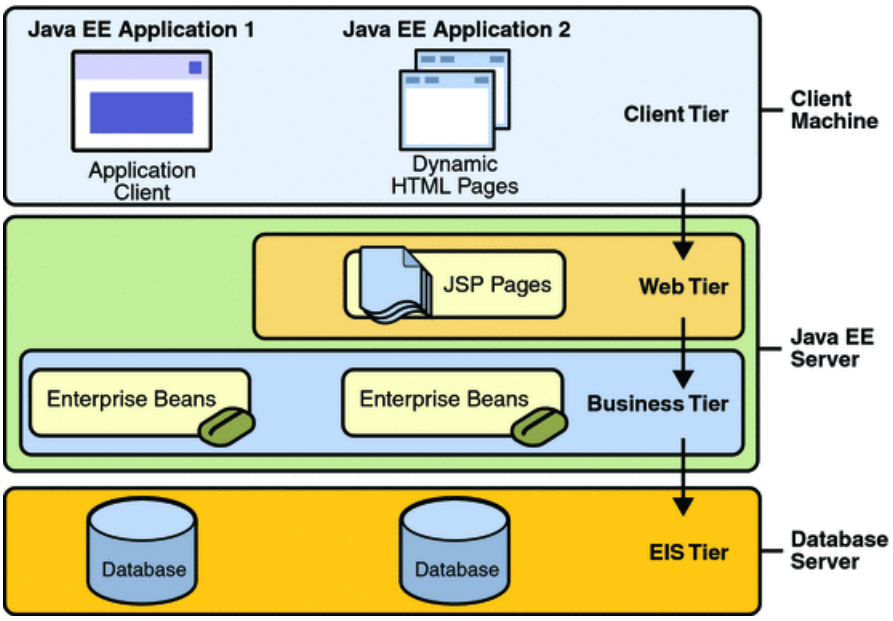
- Kile, un editor per scrivere in LaTeX, e compilare il sorgente in un file PDF
- LiveTex, una distribuzione di TeX, senza cui Kile non può compilare
- JDER, un editor di diagrammi E-R
- Mysql Workbench per il diagramma di vista logica del database
- !!!TODO : ciò che ho usato per lo UX !!!

Invece per lo sviluppo di SWIMv2 sono stati impiegati i seguenti software e le seguenti tecnologie:

- Eclipse Juno per J2EE per la scrittura del codice java/jsp
- Openjdk 1.7 come versione di java, senza cui Eclipse non può nemmeno partire
- Mysql Server come server per il DB
- JBoss AS 5 come server locale su cui far girare il sistema
- WaveMaker !!!O chi per lui !!! per disegnare l'interfaccia grafica

3 Architettura del sistema

SWIMv2 è sviluppato secondo l'architettura client-server, e quest'ultimo secondo il pattern MVC, in modo da tenere nettamente separati lo stato del sistema, la logica di business e l'interfaccia grafica. In particolare, per quanto riguarda il client, possiamo immaginare che esso sia un comunissimo browser che manda delle richieste ad un server ed elabora le sue risposte, mostrando all'utente le informazioni desiderate tramite la view che gli viene passata. Il server invece è locale (fondamentalmente http://127.0.0.1:8080), ospita il database e la logica di business e fornisce un'interfaccia grafica personalizzata a seconda dello stato del sistema; tale stato varia a seconda dei comandi ricevuti dal client, ma non in modo arbitrario: il controller si occupa di mantenere il DB in uno stato consistente e di comporre volta per volta l'HTML da passare al browser; a questo proposito, particolare importanza hanno le servlet, componenti che appunto generano le pagine web dinamiche che l'utente vede. Il server, come già specificato è JBoss AS 5, un'implementazione open source della piattaforma J2EE.
La seguente figura rappresenta graficamente l'architettura del sistema:

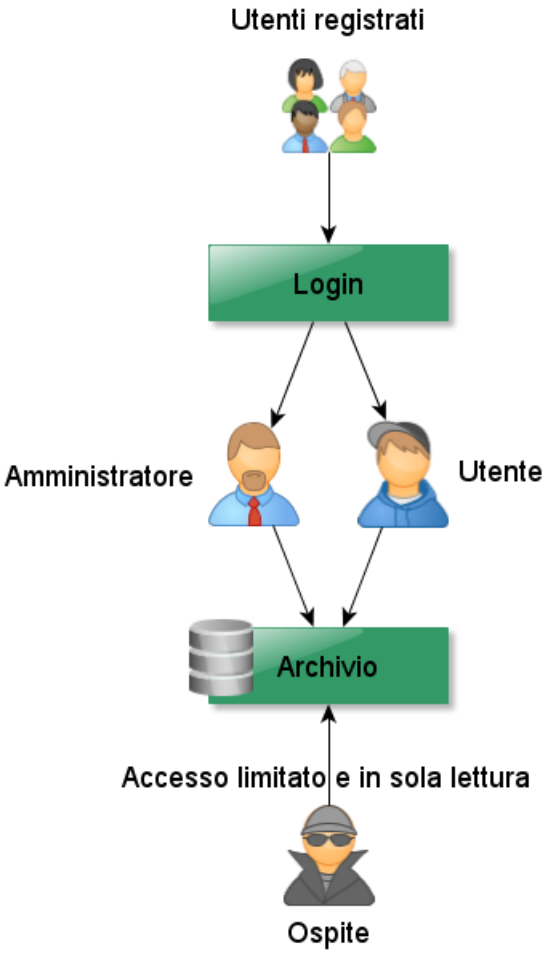


4 Scomposizione in sottosistemi

A livello di funzionalità è possibile individuare tre grandi categorie, che le raggruppano in base ai tipi di utenti; è necessario precisare che queste categorie non sono mutuamente esclusive (ad esempio la ricerca degli utenti può essere effettuata da chiunque):

- Le funzionalità accessibili dagli ospiti: ricerca degli utenti, registrazione, login
- Le funzionalità accessibili dagli utenti registrati: ricerca, gestione del profilo, richiesta/accettazione di amicizie, invio di messaggi, invio di feedback, richiesta di nuove abilità
- Le funzionalità accessibili dagli amministratori: ricerca, gestione del profilo, richiesta/accetazione di amicizie, invio di messaggi, invio di feedback, gestione del set di abilità di sistema

Tutte queste categorie si appoggiano ad un altro sottosistema, che è il database, il quale funge da archivio dei dati persistenti (vedi la sezione sul DB); infine come ultimo gruppo si può identificare l'insieme di funzionalità volte a garantire e a disciplinare l'accesso al sistema a seconda del proprio status: la registrazione ed il login. Appare evidente una parziale intersezione tra quest'ultimo sottosistema e quello delle funzionalità accessibili dagli ospiti: per essere più chiari possiamo considerare registrazione e login lato client come parte del sottosistema dell'ospite, mentre gli stessi lato server come parte del sistema controllo.
Il grafico nella prossima pagina mostra quello che ho appena scritto:



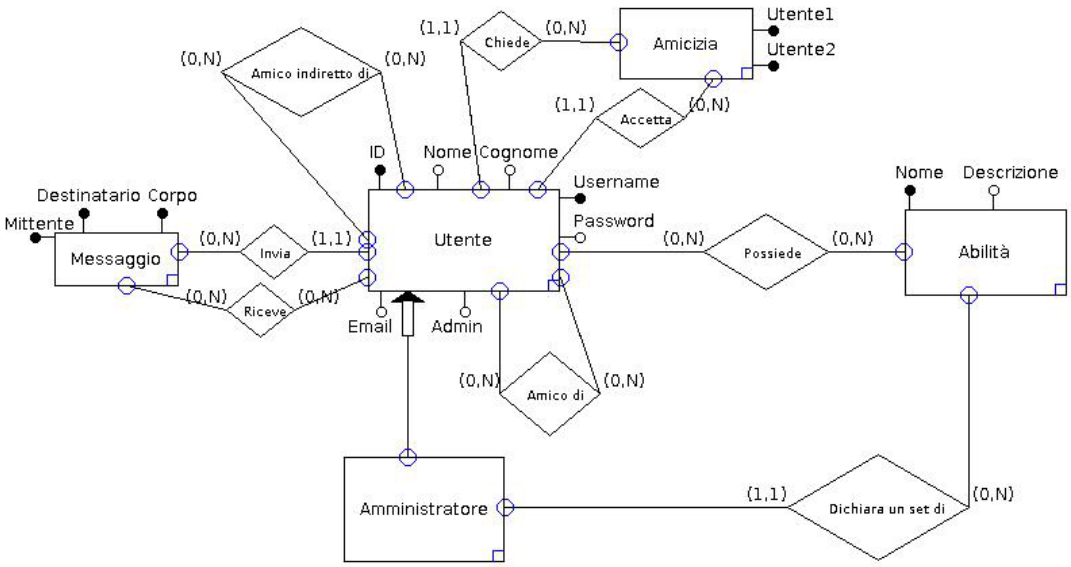
Il fatto che il grafo non sia orientato indica che ci sono interazioni in entrambi i versi degli archi.

5 Gestione del Database

5.1 Diagramma E-R

Il diagramma entità-relazione (progetto concettuale) è formato dalle entità che entrano in gioco in SWIMv2 e dalle relazioni che intercorrono tra di esse; è un passo intermedio verso la definizione delle tabelle del database del sistema e mette in evidenza in che modo le entità interagiscono tra di loro.

Il diagramma è questo (segue spiegazione delle entità e delle relazioni):



5.1.1 Entitá

- **Utente**: è l'entità principale e contiente i dati della persona che si è registrata su SWIMv2: nome, cognome, indirizzo email, username e password, oltre ad un campo univoco ID e ad un campo Admin (che

nella tabella sarà di tipo tinyint(1)) che indica se l'utente è o non è un amministratore.

- **Amministratore**: è un sottotipo di utente che ha delle funzionalità in più: può infatti gestire il set di abilità di sistema.
- **Messaggio**: è un entità che modella i messaggi tra gli utenti: infatti ha come attributi il mittente, il destinatario e il corpo.
- **Abilità**: è un entità che indica il set di abilità di sistema da cui gli utenti possono dichiarare le proprie abilità.
- **Amicizia**: indice il fatto che due utenti siano amici tra di loro, e infatti i suoi attributi sono proprio tali utenti

5.1.2 Relazioni

- **Chiede - Accetta**: sono riferite al fatto che un utente possa chiedere o accettare una richiesta di amicizia a/da un altro utente.
- **Invia - Riceve**: sono riferite al fatto che un utente possa inviare o ricevere un messaggio
- **Amico diretto - Amico indiretto**: sono riferite al fatto che un utente possa essere amico diretto o indiretto di un altro amico
- **Possiede**: è riferita al fatto che ogni utente possiede un set di abilità
- **Utente**: è riferita al fatto che un amministratore può modificare il set di abilità di sistema

5.2 Progetto Logico

Il progetto logico consiste nell'insieme di tabelle che compongono il database, ottenute a partire dall'ER.

5.2.1 Tabelle risultanti

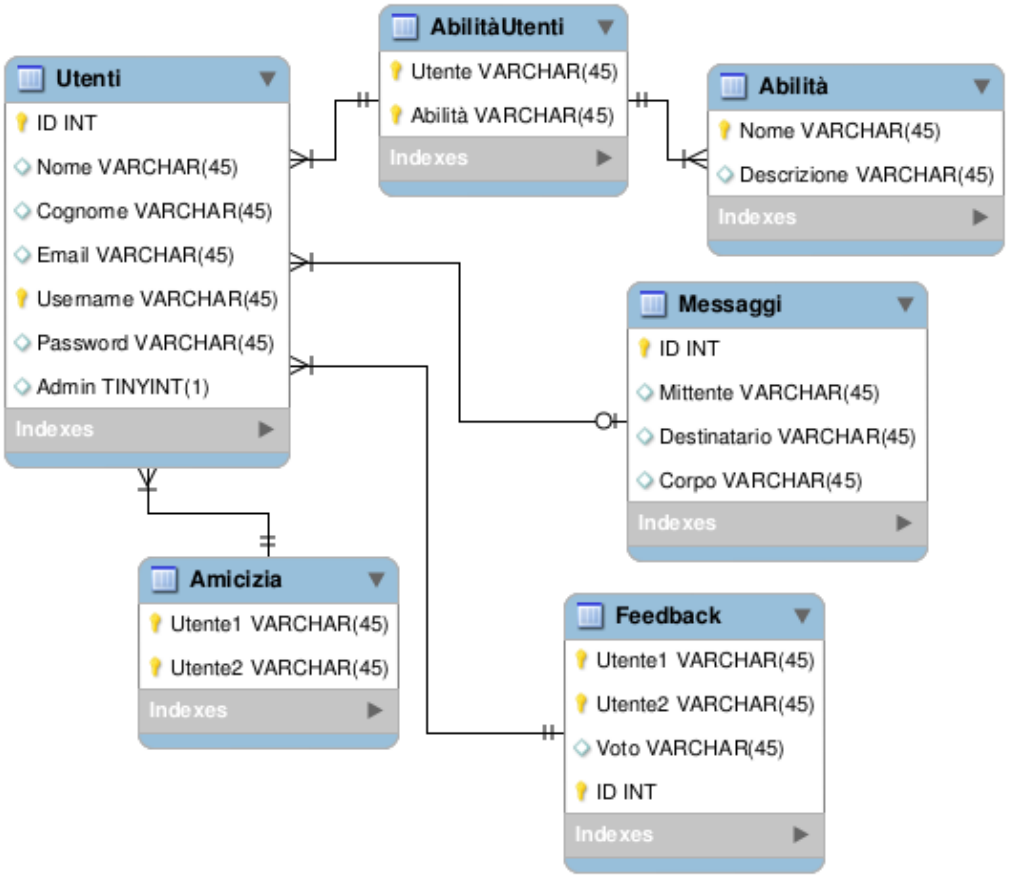
UTENTE(**ID**,NOME,COGNOME,EMAIL,**USERNAME**,PASSWORD,ADMIN)
ABILITA'(**NOME**,DESCRIZIONE)
ABILITA'UTENTI(**UTENTE,ABILITA'**)

MESSAGGIO(**ID,MITTENTE,DESTINATARIO**,CORPO)
AMICIZIA(**UTENTE1,UTENTE2**)
FEEDBACK(**ID, UTENTE1, UTENTE2**, VOTO)

in cui in ABILITA'UTENTI.NOME si riferisce a UTENTE.USERNAME, ABILITA'UTENTI.ABILITA' si riferisce a ABILITA'.NOME, MESSAGGIO.MITTENTE, MESSAGGIO.DESTINATARIO, AMICIZIA.UTENTE1, AMICIZIA.UTENTE2, FEEDBACK.UTENTE1 e FEEDBACK.UTENTE2 si riferiscono tutti a UTENTE.USERNAME. Le chiavi primarie sono in grassetto.

5.3 Vista Logica

L'ultimo diagramma relativo al database è la vista logica delle tabelle del database:



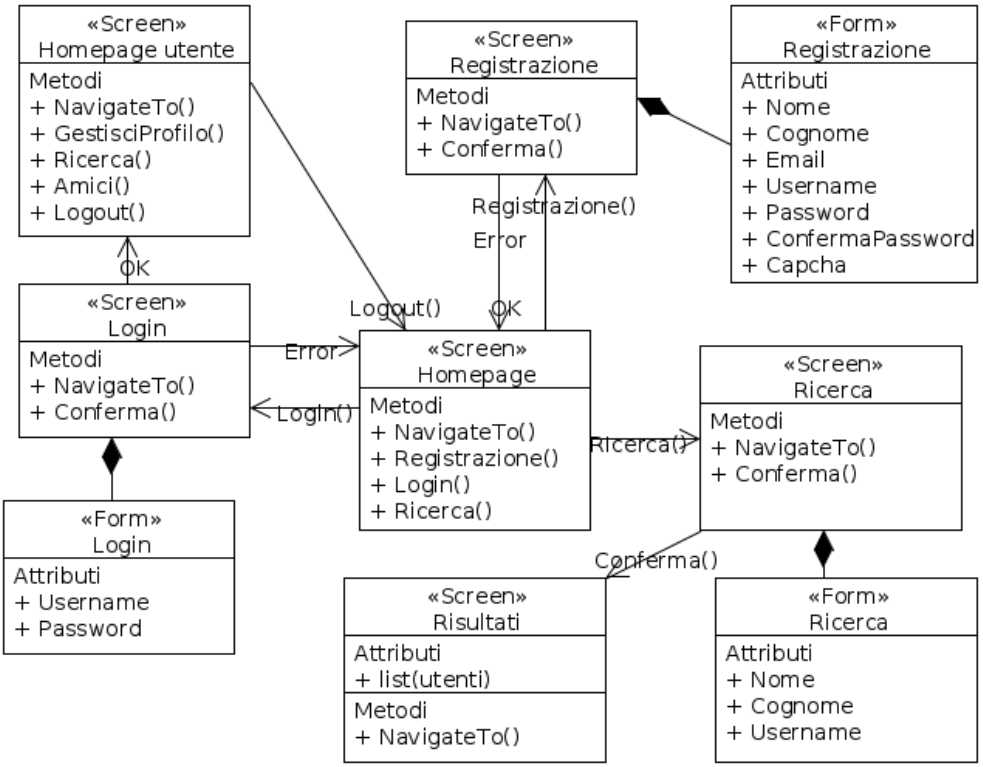
6 Design

In questa sezione infine si trovano gli UXDiagram, ovvero i diagrammi che mostrano le schermate visibili agli utenti del sistema.

6.1 UXDiagram

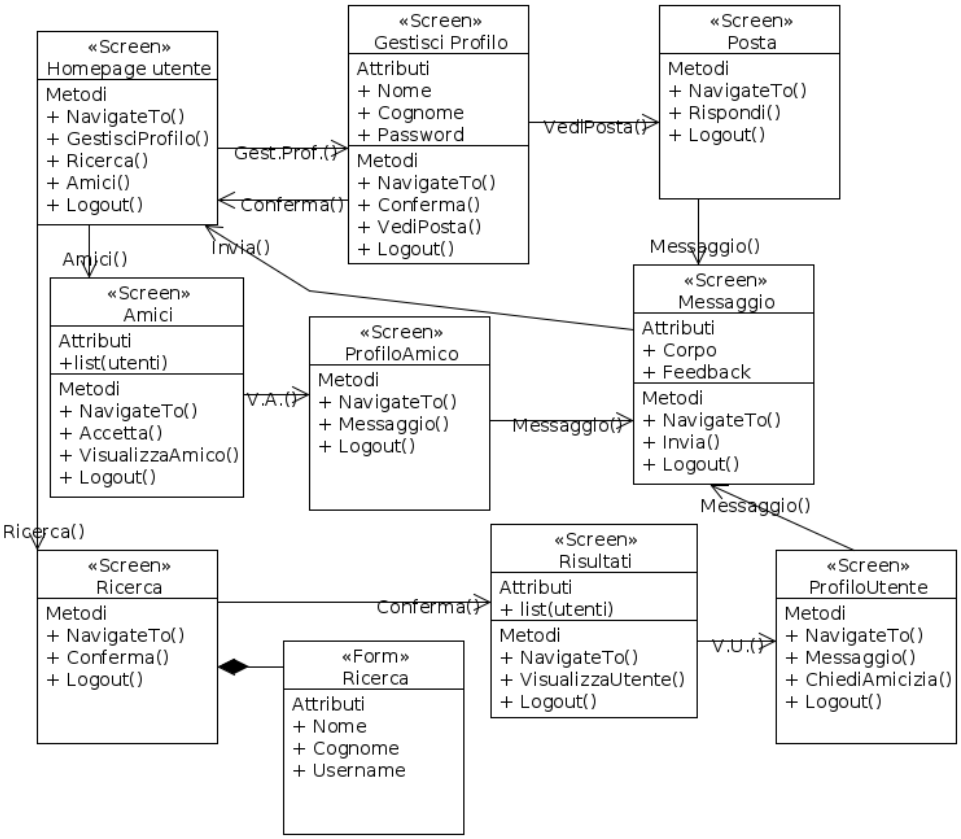
Si suppone che la Homepage sia un landmark.

6.1.1 UX per un ospite



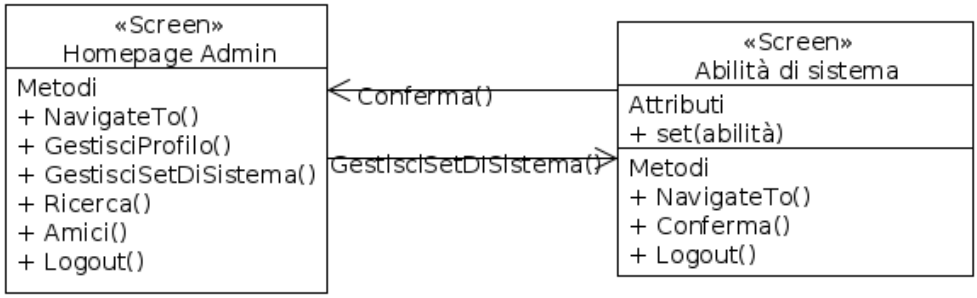
6.1.2 UX per un utente registrato

Si suppone che la homepage dell'utente sia un landmark limitatamente agli utenti registrati e che si possa effettuare il logout da qualunque pagina



6.1.3 UX per un amministratore

Questo UX Diagram contiene solo le funionalità esclusive dell'amministratore per motivi di leggibilità



7 Sicurezza

L'ultima sezione del Design Document riguarda le metodologie impiegate per garantire la sicurezza del sistema: innanzittutto, come già detto, vengono impiegati hash salato e crittografia per evitare che nel DB siano presenti in chiaro dati sensibili; poi, l'altro grande punto è il filtraggio delle stringhe in input a qualunque form per evitare attacchi del tipo SQL Injection e XSS.