UNIVERSITÀ DEGLI STUDI DI SALERNO

FACOLTÀ' DI SCIENZE MM. FF. NN.

CORSO DI LAUREA TRIENNALE IN INFORMATICA

Corso di Ingegneria del Software

LinkedPeople

LinkedPeople
System Design Document

Versione 3.0

ANNO ACCADEMICO 2018/2019

Coordinatore del progetto:

Nome	Matricola
Andrea De Lucia	
Rita Francese	

Partecipanti:

Nome	Matricola
Bruno D'Agostino	0512103598

Revision History

Data	Versione	Descrizione	Autore
08/10/2018	1.0	Prima stesura del documento	Membri del team
19/10/2018	2.0	Seconda revisione	Membri del team
22/10/2018	3.0	Terza revisione	Membri del team

Indice

Introduzione

Obiettivi del Sistema

La missione di LinkedPeople è dare alle persone il potere di costruire comunità e avvicinare il mondo. Le persone usano LinkedPeople per rimanere in contatto con amici e familiari, per scoprire cosa sta succedendo nel mondo e per condividere ed esprimere ciò che conta per loro. Il nostro obiettivo è quello di realizzare un sistema che permetta alle persone di comunicare tra di loro e condividere proprie esperienze con amici senza limiti di barriere e senza limiti di distanza, ovvero ogni persona può in qualsiasi momento sentirsi vicina al proprio amico anche se l'amico si trova a migliaia di kilometri di distanza.
Il sistema deve mettere a disposizione un sito web per gli utenti che per essere attivi sul sito basta avere una connessione ad Internet e un dispositivo dotato di un qualsiasi web browser.
Il sito web per l'utente deve essere strutturato in modo tale da poter offrire all'utente servizi quali la condivisione di posts, la possibilità di inviare messaggi, di lasciare commenti o likes ai posts condivisi dagli amici, ecc.
Data l'importanza e la sensibilità dei dati trattati (ad esempio i messaggi scambiati tra gli utenti), il software deve poter garantire una certa affidabilità per quanto riguarda la trasmissione e ricezione di informazioni e la loro correttezza; dunque, oltre ad un backup periodico e automatico, prevede: controlli sugli input sia su lato client che su lato server; connessione tramite canali sicuri e certificati; salvataggio dei dati in locale prima di essere trasmessi nel caso dovesse presentarsi un errore e mancare la connessione, così da poterli instradare appena possibile.
In fine, per una migliore usabilità, il sistema dovrà essere facile da apprendere e intuitivo da utilizzare così da permettere all'utente la navigazione senza obbligatoriamente dover visualizzare la documentazione associata.

Obiettivi di Progettazione

Gli obiettivi di progettazione identificati per il sistema sono i seguenti:

Criteri di Performance

Tempo di Risposta	LinkedPeople deve assicurare una risposta rapida alle richieste dell'utente. Una semplice richiesta dell'utente deve essere gestita ed elaborata entro 10 secondi, nel caso di una connessione molto lenta (20-25KBps). I tempi di risposta saranno più rapidi quanto più è veloce la tipologia di connessione utilizzata.
Throughput	Il sistema sarà in grado di gestire più utenti contemporaneamente. Essendo difficile effettuare una stima precisa del numero di utenti da gestire, è comunque possibile prevedere la gestione di circa 1 milione di utenti annuali.

Memoria	Anche qui non è facile preventivare una stima precisa della memoria necessaria. Ad ogni modo il sistema dovrà essere munito delle risorse necessarie per memorizzare almeno: 1 milione di utenti.

Criteri di Affidabilità

Robustezza	LinkedPeople deve gestire eventuali errori di input senza interrompere il funzionamento dell'intero sistema assicurandosi che l'errore venga correttamente gestito tramite eccezioni e messaggi di avvertimento.
Disponibilità	Gli utenti possono accedere al sistema in qualunque momento, 24 ore su 24, grazie ad un server sempre attivo.
Tolleranza all'errore	LinkedPeople sarà progettato per operare anche in presenza di errori. L'accurata suddivisione in sotto parti in grado di gestire l'insorgere di eccezioni sul momento, così da evitarne la propagazione a sotto parti cooperanti, innalza il livello di tolleranza dell'errore del sistema.
Sicurezza	La sicurezza del sistema è garantita da apposita interfaccia di autenticazione che funge da "setaccio", impedendo a utenti senza permessi di accedere ad aree sensibili e riservate del sistema.

Criteri di Manutenzione

Estensibilità	Il sistema sarà progettato in modo da facilitare la futura implementazione di nuove funzionalità. Il codice sarà dunque ben strutturato e di facile comprensione per facilitare future estensioni.
Modificabilità	Anche in questo caso il codice sarà ben strutturato e accurato così da permettere future e rapide modifiche anche in presenza di bug.
Leggibilità	Il codice sarà correttamente indentato e commentato in ogni sua parte così da essere comprensibile per qualunque lettore competente.

Tracciabilità dei Requisiti	La tracciabilità dei requisiti è possibile grazie alla matrice di tracciabilità, con la quale si può retrocedere ai requisiti associate a ogni parte del progetto

Criteri per l'Utente Finale

Usabilità	L'interfaccia del sistema deve essere di facile e diretta comprensione, offrendo feedback su ogni azione eseguita dall'utente. Gli oggetti inoltre di interfaccia dovranno essere inoltre facili da raggiungere in qualsiasi posizione ci si possa trovare.
Utilità	Il sistema dovrà ridurre in maniera considerevole i tempi di comunicazione tra gli utenti.

Definizioni, Acronomi e Abbreviazioni

Definizione	**Descrizione**
Account	Attore già registrato nel sistema con nome utente univoco e password.
Login	Operazione che permette ad ogni attore di poter accedere alla piattaforma previa immissione di email e della password associata ad esso.
Logout	Operazione che permette ad ogni attore di poter uscire dalla piattaforma.
Username	Sinonimo di nome utente.
Nome utente	Nome univoco che identifica l'utente all'interno della piattaforma; viene definito all'atto della registrazione.
Password	Parola d'ordine associata ad un'email; viene scelta all'atto della registrazione.
Registrazione	Procedura che assegna all'utente un account, permettendogli di accedere al sistema selezionando un'opportuna email e password.
Utente	Qualsiasi attore che ha l'autorizzazione di interagire con il sistema.
LinkedPeople	Nome assegnato al sistema.

Sistema	Sinonimo di piattaforma.

Acronimo / Abbreviazione	**Descrizione**
Admin	Amministratore
User	Utente
GUI	Graphical User Interface
DBMS	DataBase Management System
RAD	Requirements Analysis Document
SDD	System Design Document

Riferimenti

- B. Bruegge, A. H. Dutoit, Object Oriented Software Engineering-Using UML, Pattern and Java, Prentice Hall, 3rd edition, 2009.
- Documento RAD del progetto LinkedPeople.

Panoramica

Nel SDD dobbiamo affrontare i problemi a livello di sistema durante la decomposizione del sistema stesso. In particolare dobbiamo affrontare i seguenti problemi:

- **Mapping hardware/software**, riguardante la scelta della configurazione hardware del sistema, di quali nodi forniscono determinate funzionalità, della comunicazione tra i nodi, come i componenti software del sistema sono incapsulati e quali sottosistemi forniscono quel determinato servizio;
- **Gestione dei dati persistenti**, riguardante la scelta dei dati da rendere persistenti, di dove andare a memorizzare e di come possiamo accedere a tali dati;
- **Controllo degli accessi**, che ci aiuta, a seconda dell'attore, a limitare e controllare l'accesso ai dati e definire come queste politiche vengono ideate e implementate;
- **Controllo del flusso delle operazioni**, che ci aiuta a capire e controllare quali operazioni devono essere eseguite dal sistema e il loro ordine nel flusso delle operazioni;
- **Condizioni dei boundary**, riguardante come avviene l'avvio/arresto del sistema, la gestione dei casi eccezionali come sbalzi/interruzione di corrente, guasti hardware o errori di progettazione.

Sistema Proposto

Panoramica

Il sistema LinkedPeople è un sistema basato sull'architettura Three Tier. Grazie alle interfacce proposte dal livello "Presentation" l'utente potrà facilmente interagire con il sistema. I dati raccolti in questo livello vengono passati e processati dal livello "Business" il quale elabora i dati in questioni sulla base delle strutture contenute all'interno del livello "Data Resource", agendo come intermediario tra i due livelli precedentemente discussi.
In fine il livello "Data Resource" si occupa della comunicazione tra il sistema e il database al fine di mantenere lo stato del sistema costantemente aggiornato.
Tale struttura garantisce la corretta ed efficiente gestione della concorrenza in lettura e scrittura dei dati da parte degli utenti del sistema.

Decomposizione in Sottosistemi

Come si evince dalla disposizione dei sottosistemi, si è deciso di utilizzare una architettura Three Tier, soprattutto per evitare il noto effetto "collo di bottiglia" dovuto agli accessi concorrenti al sistema da parte di più utenti. I tre livelli adottati per l'architettura sono come da standard:

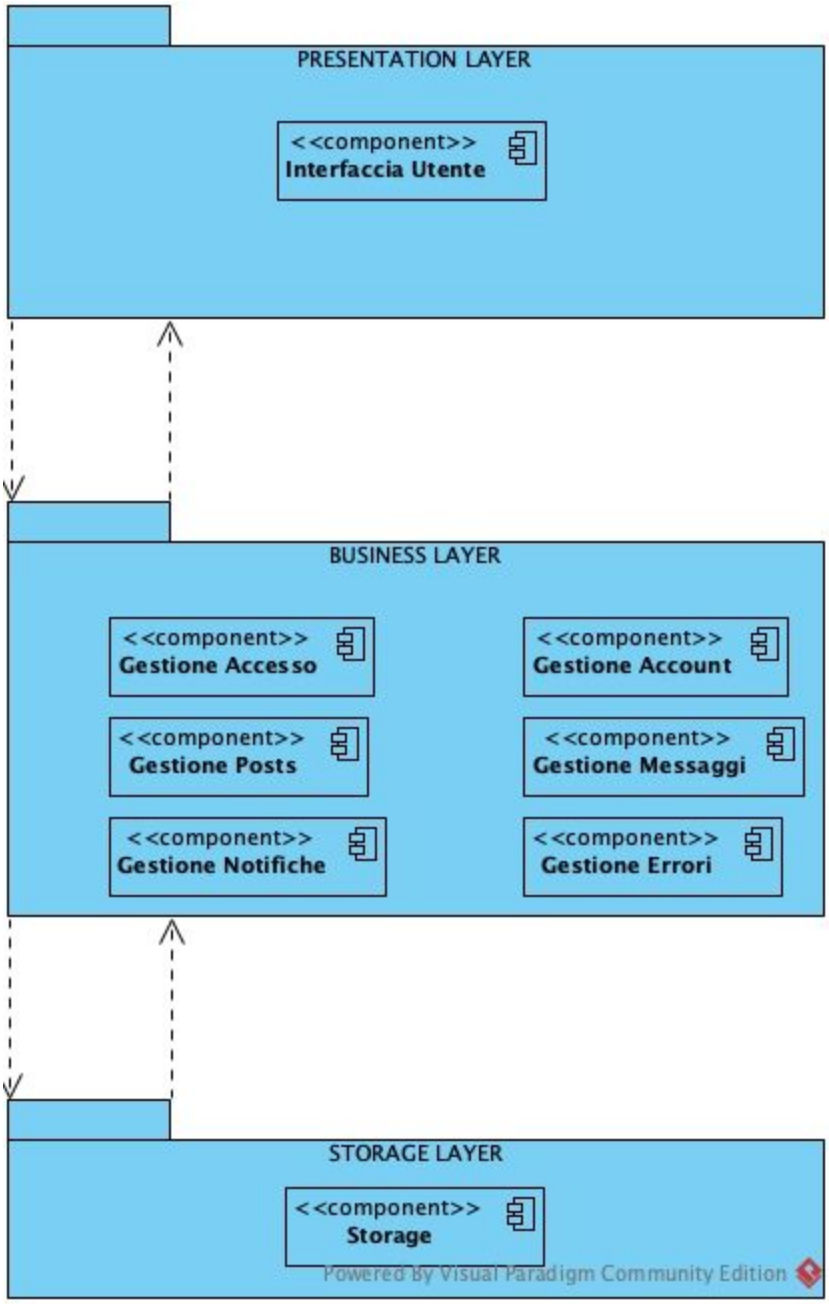
- *Business Logic*: si occupa di tutte le componenti logiche e in particolare della business logic. Permette di effettuare operazioni sullo strato *Data Resource* e sullo strato *view*. Possiamo identificare gli oggetti al suo interno come gli oggetti "control" individuati nel RAD. I sottosistemi individuati al suo interno sono i seguenti:
    - Gestione Accesso
    - Gestione Account
    - Gestione Posts
    - Gestione Messaggi
    - Gestione Notifiche
    - Gestione Errori

- *Data Resource*: si occupa di comunicare con il database per effettuare operazioni relative alla persistenza dei dati. Comunica con il *Business Logic*. Possiamo identificare i suoi oggetti come gli oggetti "entity" individuati nel RAD. I sottosistemi individuati al suo interno sono i seguenti:
    - Storage Utenti
    - Storage Posts
    - Storage Messaggi
    - Storage Notifiche

- *Presentation*: si occupa di comunicare tutti gli stati del sistema all'utente. E' aiutato ad adempiere questo compito grazie alle GUI che il sistema mette a disposizione. Spesso è richiesto che l'utente interagisca con il sistema, quindi lo strato *Presentation* può comunicare con lo strato *Business Logic*. Possiamo identificare i suoi oggetti come oggetti "boundary" individuati nel RAD. I sottosistemi individuati al suo interno sono i seguenti:

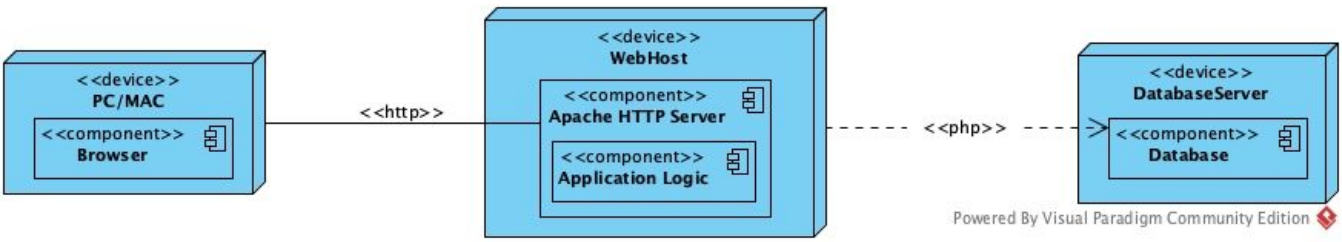
- GUI Accesso
- GUI Account
- GUI Post
- GUI Messaggio
- GUI Amico
- GUI Errori

E’ giusto porre l’attenzione su due sottosistemi, ovvero: “Gestione Errori” e “GUI Errori”; il cui compito è riservato alla gestione degli errori, siccome ogni sottosistema è sottoposto a possibilità di errore, si è deciso di inserire il sottosistema “Gestione Errori”, che appunto permetta o di risolverli oppure di indirizzarli verso una “GUI Errori”, in modo da rendere l’utente informato di tale errore. Non c’è motivo di aggiungere la gestione degli errori anche nel model, poiché si collega al database, e come ben sappiamo possiede dei meccanismi di gestione errori già consolidati in sé, e si dovrebbe sempre evitare di mostrare all’utente errori relativi a tale ambito di esecuzione (soprattutto per motivi di sicurezza), quindi il *Business Layer* si occuperà di mostrare all’utente un errore generico tramite la “GUI Errori” in caso di eccezioni relative a tale ambiente. Di seguito il sistema decomposto in vari sottosistemi, sulla base dell’architettura utilizzata:



Mapping Hardware/Software

In LinkedPeople distinguiamo tre nodi, il primo: PC/MAC, il device che l'utente usa per interfacciarsi con il sistema; SERVER, che fornisce la logica e i servizi dell'applicazione; DATABASE SYSTEM, che fornisce e gestisce il database.



Per la realizzazione di LinkedPeople abbiamo scelto di utilizzare il linguaggio di programmazione **PHP** e del DBMS MySQL.
PHP è un linguaggio di scripting lato server progettato per lo sviluppo Web, ma anche usato come linguaggio di programmazione generico. È stato originariamente creato da Rasmus Lerdorf nel 1994, l'implementazione di riferimento PHP ora è prodotta da The PHP Group. PHP originariamente era la Home page personale, ma ora è sinonimo dell'inizializzazione ricorsiva PHP: Hypertext Preprocessor.
MySQL è un software detto **R**elational **D**ata**B**ase **M**anagement **S**ystem (**RDBMS**), open source e sviluppato per essere il più possibile conforme agli standard SQL.

Il grande vantaggio offerto dall'uso di questi standard è che sono implementati da diversi prodotti open-source e commerciali, consentendo così a LinkedPeople di avere scalabilità (ad es. numero di utenti) e costi relativamente bassi.
Inoltre, nella loro forma più semplice, i componenti open-source di questi framework sono facili da usare.

L'utente può richiedere le funzionalità di LinkedPeople attraverso un **Web Browser**.
I sottosistemi di LinkedPeople sono accessibili attraverso **Apache HTTP Server**.
Apache HTTP Server (o semplicemente **Apache**) è un server web open source sviluppato dalla Apache Software Foundation. Implementa le specifiche PHP, fornendo quindi una piattaforma software per l'esecuzione di applicazioni Web sviluppate in linguaggio PHP.

Per realizzare il sistema e i relativi sottosistemi utilizziamo PHP e HTML come principale tecnologia per l'implementazione degli oggetti boundary.

Gestione dei Dati Persistenti

Si rimanda al documento "LinkedPeople_SystemDesignDocument - PersistentDataManagement" allegato al presente.

Controlli degli Accessi e Sicurezza

Poiché HelpPharma è un sistema multiutente, a diversi attori è consentito visualizzare diversi insiemi di oggetti e invocare diversi tipi di operazioni su di loro.
Per documentare in modo sintetico i diritti di accesso, abbiamo costruito una matrice di controllo degli accessi che descrive le operazioni consentite sugli oggetti entità per ciascun attore del sistema.

Utente

OGGETTO	**OPERAZIONI CONSENTITE**
Utente	«create» aggiungi, aggiorna, visualizza, rimuovi
Post	«create» aggiungi, visualizza, rimuovi
Messaggio	«create» aggiungi,

	visualizza, rimuovi

Controllo Globale del Software

Il Web Server attende le richieste dal Web Browser. Ricevuta una richiesta, il Web Server la elabora e produrrà una risposta che verrà inoltrata al client.
Il controllo del flusso software è quindi basato sugli eventi.
Il Web Server assegna un nuovo thread per ogni richiesta, consentendo la gestione parallela delle richieste. Ciò si traduce in un sistema più reattivo a discapito di un throughput più elevato.

Servizi del Sottosistema

Gestione Account

Sottosistema che gestisce le operazioni relative all'account dell'utente e le informazioni che lo caratterizzano.

Servizio	**Descrizione**
getUsername()	Permette di visualizzare l'username associato all'utente.
visualizzaDettagliAccount()	Permette di visualizzare i dettagli associati all'account dell'utente.
getNumberOfFriendRequests()	Permette di visualizzare il numero di richieste di amicizie ricevute.
getNumPosts()	Permette di visualizzare il numero di posts pubblicati dall'utente.
getFirstAndLastName()	Permette di visualizzare il nome e cognome dell'utente.
getProfilePic()	Permette di visualizzare la foto del profilo dell'utente.
getFriendArray()	Permette di visualizzare la lista degli amici dell'utente.
isClosed()	Permette di verificare se l'account dell'utente è stato chiuso.
isFriend()	Permette di verificare se l'utente è amico con un altro utente.
didReceiveRequest()	Permette di verificare se l'utente ha ricevuto

	richieste di amicizie.
didSendRequest()	Permette di verificare se l'utente ha inviato richieste di amicizie.
removeFriend()	Permette di rimuovere un utente dalla lista degli amici.
sendRequest()	Permette di inviare richieste di amicizie.
getMutualFriends()	Permette di visualizzare quanti amici in comune ha l'utente con un altro amico sulla piattaforma.

Gestione Posts

Sottosistema che gestisce le operazioni relative ai posts dell'utente e le informazioni che lo caratterizzano.

Servizio	**Descrizione**
submitPost()	Permette di pubblicare un post sulla piattaforma.
calculateTrend()	Permette di visualizzare i trends associati ai posts degli utenti.
loadPostsFriends()	Permette di visualizzare i posts pubblicati dagli amici.
loadProfilePosts()	Permette di visualizzare i posts pubblicati dall'utente sul profilo personale.
getSinglePost()	Permette di visualizzare il singolo post a seguito dell'apertura di una notifica.

Gestione Messaggi

Sottosistema che gestisce le operazioni relative ai messaggi dell'utente e le informazioni che lo caratterizzano.

Servizio	**Descrizione**
getMostRecentUser()	Permette di visualizzare l'utente più recente nelle conversazioni.
sendMessage()	Permette di inviare un messaggio.
getMessages()	Permette di visualizzare i messaggi ricevuti.
getLatestMessage()	Permette di visualizzare l'ultimo messaggio ricevuto.

getConvos()	Permette di visualizzare le conversazioni iniziate.
getConvosDropdown()	Permette di visualizzare il menu dropdown delle conversazioni iniziate.
getUnreadNumber()	Permette di visualizzare il numero di messaggi non letti.

Gestione Notifiche

Sottosistema che gestisce le operazioni relative alle notifiche dell'utente e le informazioni che lo caratterizzano.

Servizio	**Descrizione**
getUnreadNumber()	Permette di visualizzare il numero di notifiche non lette.
getNotifications()	Permette di visualizzare tutte le notifiche ricevute.
insertNotification()	Permette di inserire nuove notifiche per gli utenti.

Glossario

DBMS	Sistema di gestione di basi di dati
PHP	Hypertext Preprocessor
HTML	HyperText Markup Language