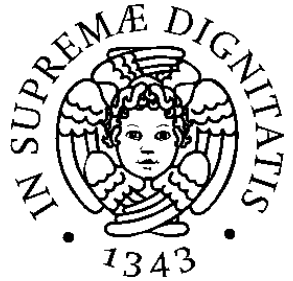

UNIVERSITÀ DI PISA

SCUOLA DI INGEGNERIA
Dipartimento Ingegneria dell'Informazione
Corso LARGE-SCALE AND MULTI-STRUCTURED DATABASES
MSc in ARTIFICIAL INTELLIGENCE AND DATA ENGINEERING

REPORT PROGETTAZIONE E IMPLEMENTAZIONE DI UN SISTEMA PER LA GESTIONE DI EVENTI

PRESENTATO DA:

CARUSO Alberto
TUMMINELLI Gianluca
Di NOIA Antonio
CALABRESE Pietro

Anno Accademico
2019/2020

1 Introduzione

La pratica della vita quotidiana è intrecciata con il digitale, in particolare i media mobili, e questo si estende anche alla ricerca di eventi vicini alla proprie passioni. Siti e app per la ricerca di questo tipo di contenuti, i servizi che supportano la ricerca di eventi sono sempre più numerosi sviluppati per dispositivi mobili, ma in realtà sono supportati da altri social network non sviluppati con questa finalità ma adattati allo scopo. Il boom dei social network negli ultimi anni ha alimentato l'hype nel condividere le proprie esperienze, ma allo stesso tempo ha creato un gap importante nella reale aggregazione al fine di cerare contenuti utili alla condivisione. Attualmente, si può venire a conoscenza di un evento attraverso le funzioni messe a disposizione dai social network, ad esempio, Facebook, ma la conoscenza di questa informazione non implica che si possa partecipare all'evento prenotando il proprio posto o azioni simili, inoltre c'è da considerare i dati generati anche da un punto di vista analitico ed economico, infatti essi ci possono dare una stima molto approssimativa dell'evento e delle persone che ipoteticamente possano prendere parte a questo, in quanto molto spesso il cliccare su "partecipa a evento" risulta essere un comportamento modaiolo e non effettivamente legato ai propri interessi, infine resta da considerare che alcuni social network molto attivi negli anni passati, risultano essere fuori dai trend del momento, a questo dunque, si associa un possibile scarsa visibilità del nostro contenuto. Lo scopo principale di questo lavoro è sviluppare una web app mobile oriented, che possa rendere la gestione di un evento molto più customer oriented, con questo si includono anche tutti i servizi legati alla partecipazione, ad esempio riservare il posto per l'evento o il pagamento del costo del biglietto. A questo si aggiungono tutte le funzionalità riservate all'organizzatore per monitorare l'andamento di un particolare evento, o in generale tutti gli eventi da esso promossi, per successivamente prendere decisioni strategiche sul proprio business e sulle campagne di marketing da attuare. Resta comunque fermo che la pubblicizzazione degli eventi non può prescindere dai canali di diffusione più usati al momento.

2 Analisi del sistema

In questa sezione si analizza in modo approfondito le scelte tecniche, i requisiti, le funzionalità e gli attori del progetto.

2.1 Analisi dei requisiti

Inizialmente ci siamo concentrati sul individuare i principali requisisti che il sistema deve soddisfare. Lo scopo principale del sistema è mostrare e prenotare degli eventi generati dagli organizzatori.
Dunque le principali azioni che il sistema eseguirà sono:

- ricerca di eventi, pubblicati dagli organizzatori, da parte dei partecipanti
- inscrizione ad un evento per un partecipante
- creazione di un evento da parte di un organizzatore
- gestione evento da parte dell'organizzatore.

Oltre a questo, per entrambi gli utenti si definiscono le operazioni basilari quali registrazione al servizio, login e logout. Inoltre si definisce il vincolo che email in atto di registrazione può essere usata per un unico account.

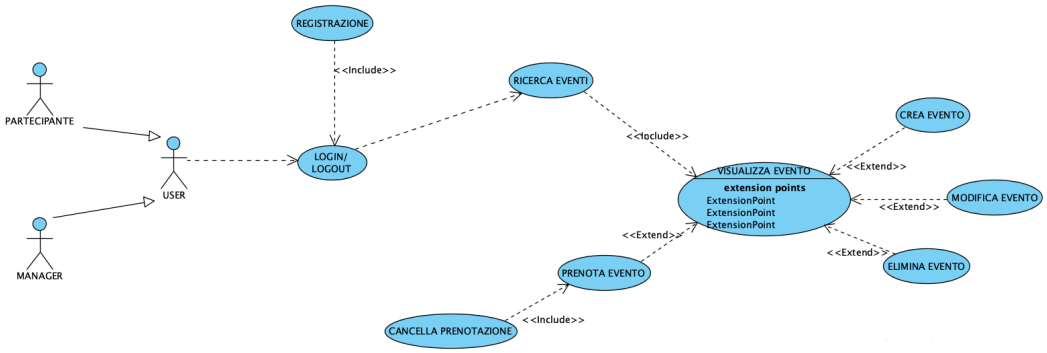

Figura 1: Use Cases Diagram

2.2 Identificazione degli attori

In questa sezione, ci soffermiamo nel definire gli attori che sono stati identificati, e che possono interagire con il sistema. Questi due attori sono: organizzatore e partecipanti, ognuno di questi ha ruoli e operazioni differenti performabili sugli eventi, come descritto nella Figura 1.

Organizzatore – gestore di un'attività o un suo delegato – che al fine di promuovere la propria attività crea eventi. Le principali funzioni associate ad *Organizzatore* sono: visualizzare la lista degli eventi da esso creati, aggiungere o rimuovere un evento, modificare alcuni parametri legati ad un evento, visualizzare la lista dei partecipanti ad un evento. I privilegi legati a questo tipo di utente non possono essere in alcun modo inclusivi dei privilegi dell'utente *Partecipante*.

Partecipante, utente interessato a visionare gli eventi disponibili e nel caso a parteciparvi. Le principali funzioni associate a *Partecipante* sono: visualizzare la lista degli eventi disponibili, raffinare la ricerca attraverso dei filtri, inscriversi ad un evento, visualizzare la lista degli eventi a cui è inscritto, cancellare la partecipazione ad un evento. I privilegi legati a questo tipo di utente non possono essere in alcun modo inclusivi dei privilegi dell'utente *Organizzatore*.

2.3 Diagramma Classi

Nella figura 2 viene mostrato il diagramma delle classi ottenuto dopo l'analisi dei requisiti e la definizione degli attori. Per favorire la leggibilità, i dettagli e le procedure di ogni singola classe sono state omesse. Infatti, l'obbiettivo principale è descrivere, attraverso l'uso delle associazioni, le relazioni che intercorrono tra le classi. Per fare questo è stato adottato il formalismo definito dal UML 2.

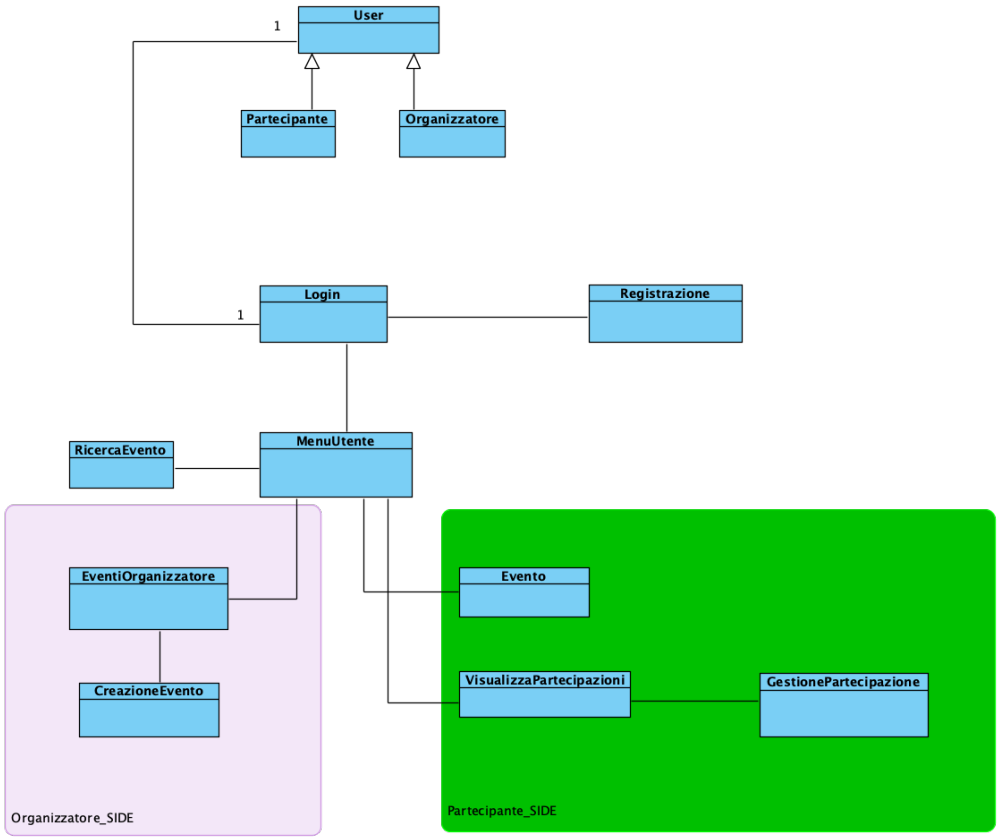

Figura 2: Class Diagram

Come si può notare nel diagramma è presente una classe astratta **User**, che può essere specializzata nei due attori definiti in precedenza, **Partecipante** e **Organizzatore**. La classe **User** è associata alla classe **Login** che gestisce il login e l'individuazione del ruolo all'interno del sistema, a sua volta associata alla classe **Registrazione** e alla classe **MenuUtente**, quest'ultima si occupa della visualizzazione dell'interfaccia grafica e con le associazioni successive delle funzioni disponibili ai vari utenti, come mostrato dai rettangoli di diverso colore.
Le classi appena descritte sono state le basi per definire le strutture dati principali utilizzate nell'implementazione del sistema, in particolare per disegnare le entità che rappresentano i dati all'interno del database.

3 Implementazione

in questa sezione, si descrive, in modo più dettagliato, le scelte implementative adottata per lo sviluppo del sistema. Attualmente si tratta di una java app, basata su architettura Client-Server, il prototipo del sistema è stato sviluppato attenendosi al architettura multi-tier, questo per tenere separati il più possibile il lato Client da quello Server.

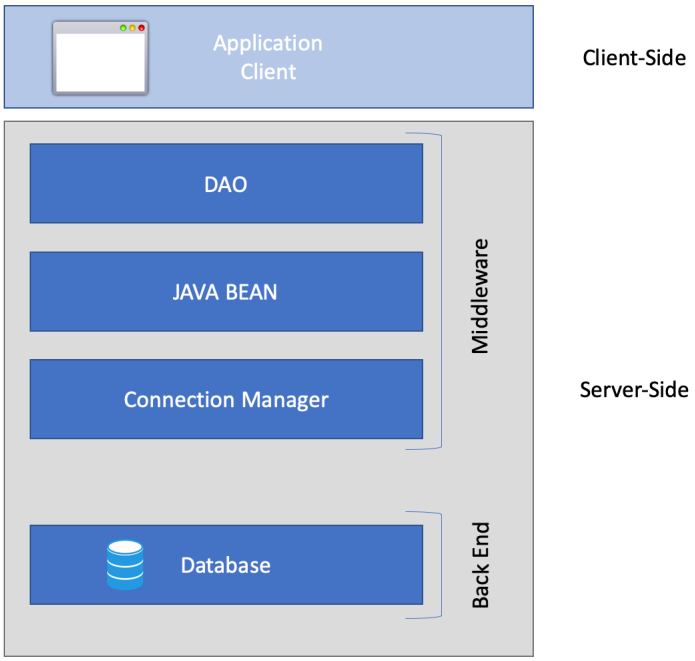

Figura 3: Multi-Tier Architecture Design

Come mostrato nella Figura 3, il lato Client è gestito con java application che si interfaccia con il lato server attraverso l'uso API mysql-connector. Per lo sviluppo della parte di Back End si è scelto come DBMS MySQL, vista la buona affidabilità e la quasi piena implementazione dello Standard SQL. Ciò non toglie che per sviluppi futuri, si possa passare a POSTgre SQL, in quanto, come spiegato successivamente, sono previsti, per future estensioni del sistema, attributi che si riferiscono allo Standard Extended-SQL, è che POSTgre implementa e gestisce al meglio, nell'ambito dei DBMS open-source. Per assicurare la consistenza dei dati all'interno del DB, garantendo allo stesso tempo la concorrenza, il sistema è stato realizzato adottando il pattern architetturale MVC. Questo tipo di approccio risulta essere il metodo

di controllo della concorrenza comunemente utilizzato dai sistemi di gestione per garantire accesso simultaneo al database e nei linguaggi di programmazione per implementare la memoria transazionale, senza questo ci potremo trovare di fronte a casi di inconsistenza dei dati all'interno del database.

3.1 Database

Nel definire le entità usate per descrivere i requisiti definiti dal sistema, si è partito dall'analisi dei requisisti e dalle considerazioni fatte sugli attori e sul diagramma delle classi, come linee guida principali. In particolare come evidenziato dalla figura 4, che descrive lo schema E-R, è presente l'entità Utente che è una generalizzazione delle due entità che descrivono gli utenti che sono stati previsti come utilizzatori del software. Poi è presente un'altra entità che descrive l'oggetto alla base di questo sistema, Evento. Infine vi sono due relazioni che servono per definire la principali operazioni all'interno del sistema.

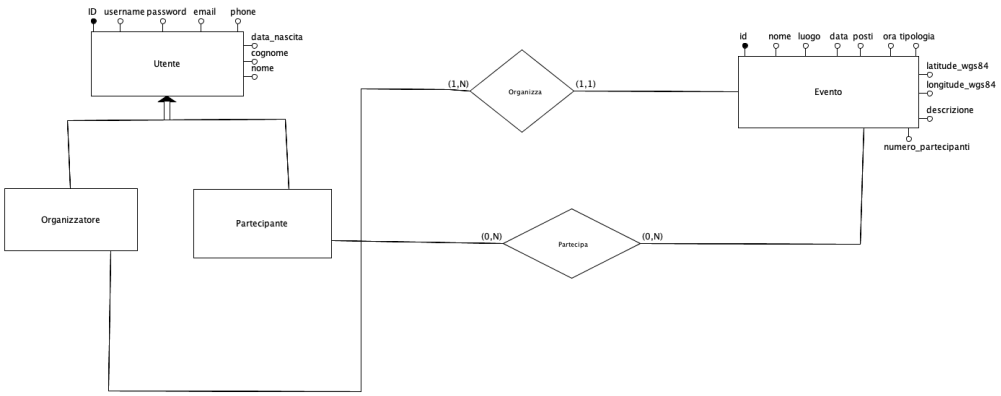

Figura 4: E-R Diagram

In fase di ristrutturazione, si è deciso per garantire maggiore velocità nell'esecuzione e per una migliore gestione dei dati di non aggregare le due entità figlie all'entità padre. Le tabelle modellate dopo la ristrutturazione risultano essere in Terza Forma Normale.

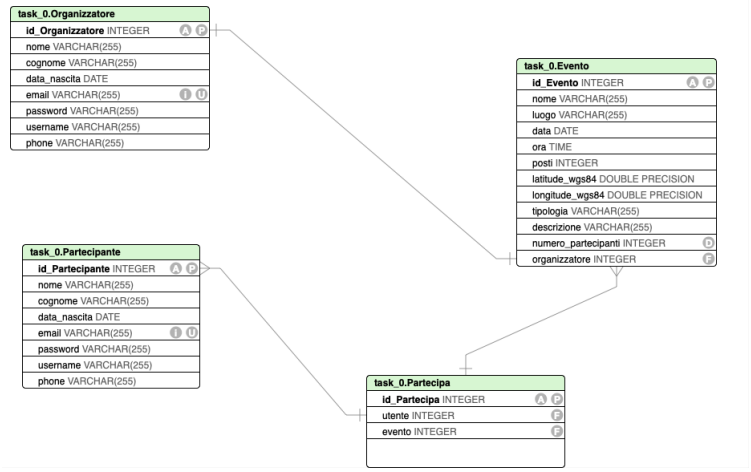

Figura 5: Logic Model

4 Conclusioni

In questo progetto è stato sviluppato un software per la gestione di eventi, in particolare si è voluto l'attenzione sulla progettazione e l'implementazione del lato Back End, quindi della parte legata al database, visto anche le specifiche richiesta dal corso. Ciò non toglie che in caso di una reale implementazione del sistema di dovrebbe procedere ad alcune migliorie in modo da garantire la fruibilità del servizio a un largo numero di utenti, su un più vasto numero di dispositivi, come ad esempio quelli mobili, quindi potrebbe essere sviluppata una web app basata su Java EE o un app specifica per i vari Mobile OS.