UNIVERSITÀ DEGLI STUDI DI SALERNO

ANNO ACCADEMICO 2016/2017

System Design Document

Versione 1.0

TOP MANAGER:

Prof. Andrea De Lucia

PROJECT MANAGER:

Antonio Luca D'Avanzo

Fabiano Pecorelli

Top Manager:

Nome
Prof. De Lucia Andrea

Project Manager:

Nome	Matricola
Antonio Luca D'Avanzo	051210 2502
Fabiano Pecorelli	052250 0421

Partecipanti:

Nome	Matricola
Severino Ammirati	051210 2898
Andrea Buonaguro	051210 2490
Angelo Caputo	051210 2204
Ferdinando D'Avino	051210 2360
Paolo Di Filippo	051210 3120
Alfredo Fiorillo	051210 1930
Dario Galiani	051210 2276
Giovanni Leo	051210 3062
Fabricio Nicolas Madaio	051210 2840
Vincenzo Noviello	051210 3198

Andrea Sarto	051210 2912
Lino Sarto	051210 2348
Giorgio Vitiello	051210 2318

Revision History:

Data	Versione	Descrizione	Autore
19/11/2016	1.0	Stesura del System Design Document	Membri del team

Indice

1. Introduzione

1.1 Scopo del sistema

Nel mondo odierno diverse sono le domande e le offerte di lavoro che affollano il mercato lavorativo ogni giorno. Compito di costoro è essere il più visibile possibile sulla piazza. Risulta cruciale quindi, per chi cerca o offre impiego, la capacità di reperire e diffondere velocemente informazioni al fine di essere più accattivante possibile sul mercato. Tale compito per sua natura non risulta semplice ed immediato. Fondamentale nella ricerca di un posto di lavoro è la possibilità di avere tutti i dati organizzati in modo ordinato e sistematico.
Per chi offre occupazione invece, è di vitale importanza raggiungere una platea di persone il più vasta possibile, in modo da agevolare,nella maniera più rapida e proficua possibile, la selezione della figura professionale ricercata. Altro aspetto da tenere in considerazione per chi genera posti di lavoro è l'organizzazione dei colloqui con i candidati non sempre realizzata al meglio. L'obiettivo da raggiungere ha una duplice valenza, rendere più efficiente ed efficace la ricerca di un'opportunità lavorativa e offrire maggiore visibilità e reperibilità per un'offerta di lavoro.

1.2 Obiettivi di design

Il sistema CrowdMine deve essere semplice e intuitivo nell'utilizzo quotidiano da tutti coloro che interagiscono con il sistema. Inoltre, il sistema deve essere efficiente, garantendo tempi di risposta brevi per ogni funzionalità richiesta e tollerante agli errori, grazie a particolari politiche di controllo e prevenzione. In più, il sistema verrà progettato per poter integrare altre funzionalità nel modo più semplice possibile. Tutti questi obiettivi di design, possono essere racchiusi in quattro categorie: Performance, Affidabilità, Manutenzione ed Utente Finale

1.2.1 Criteri di Performance

Tempo di risposta	Uno degli aspetti principali di CrowdMine è il tempo di risposta, che quindi, deve assicurare una risposta rapida alle varie richieste degli utenti. In casi di connessione lenta, si prevede un tempo massimo per gestire l'intera richiesta di 10 secondi, e questo tempo va a diminuire con l'aumento della velocità di connessione
Throughput	Il sistema sarà in grado di gestire più utenti in modo concorrente. Non è prevedibile ora stimare il carico giornaliero del sistema, ma, possiamo prevedere un carico iniziale di 100 utenti. Il sistema verrà progettato per poter resistere a richieste maggiori da parte degli utenti, utilizzando strutture scalabili per i server.
Memoria	Calcolare delle stime per capire le informazioni da salvare all'interno del database del nostro sistema è difficile, ma possiamo prevedere che in media ogni utente del sistema può pubblicare in media 5 annunci. Inoltre è presente l'Admin, con un numero modesto di Moderatori.

1.2.2 Criteri di Affidabilità

Robustezza	CrowdMine deve gestire eventuali input errati da parte dell'utente, attivando politiche di controllo sia lato Client, tramite Javascript, che lato server, utilizzando PHP. Il tutto deve avvenire senza interrompere il funzionamento dell'intero sistema.
Disponibilità	CrowdMine è pensato per essere disponibile ovunque, in qualsiasi momento e accessibile da qualsiasi dispositivo. Quindi, il sistema, deve essere disponibile 24 ore su 24, 7 giorni su 7.
Tolleranza all'errore	CrowdMine deve tollerare gli errori imprevisti nel sistema, continuando ad operare come se non ci fossero problemi, in modo trasparente all'utente. Ciò viene realizzato, grazie ad un basso grado di accoppiamento tra i vari sottosistemi, così da non propagare gli errori in tutto il sistema
Sicurezza	CrowdMine prevede il controllo degli accessi e dell'autenticazione, permettendo l'uso delle varie funzionalità della piattaforma alle categorie di utente specificate nella tabella degli accessi

1.2.3 Criteri di Manutenzione

Estendibilità	CrowdMine deve essere progettato per accogliere al meglio nuove funzionalità, integrandole al meglio con i moduli già presenti. Quindi è necessario che il codice sia ben strutturato in moduli, con il giusto trade-off tra coesione ed accoppiamento. Il tutto aiuta a futuri sviluppatori, ignari dell'architettura del sistema, di apportare le modifiche senza particolari sforzi
Modificabilità	Deve essere possibile modificare il codice in qualsiasi momento, per apportare modifiche o aggiustamenti. Questo comporta che il codice deve essere ben strutturato, con una convenzione sui nomi
Leggibilità	Il codice risulterà agevole da leggere grazie all'indentazione e sarà accompagnato da relativi commenti per comprendere al meglio cosa offre
Tracciabilità dei requisiti	Grazie alla tracciabilità dei requisiti, risulterà facile stimare il costo nel caso in cui venissero modificati alcuni requisiti per il correttamento funzionamento del sistema.

1.2.4 Criteri per l'Utente Finale

Usabilità	CrowdMine offre ogni sua funzionalità in modo semplice ed intuitivo, garantendo una piacevole interazione tra l'utente e il sistema

1.3 Definizioni, acronimi e abbreviazioni

- CrowdMine: Nome del sistema che verrà sviluppato.
- SocialHelp: Nome del sistema da evolvere.
- RAD: Requirement Analysis Document
- DBMS: Database Management System
- UML-WAE: UML-WebApplicationExtensions

1.4 Riferimenti

Documento Requirement Analysis Document - CrowdMine

1.5 Panoramica

Prima di parlare dell'architettura, è importante fare un accenno alle attività di system design che costituiscono le fondamenta per l'architettura software del sistema.

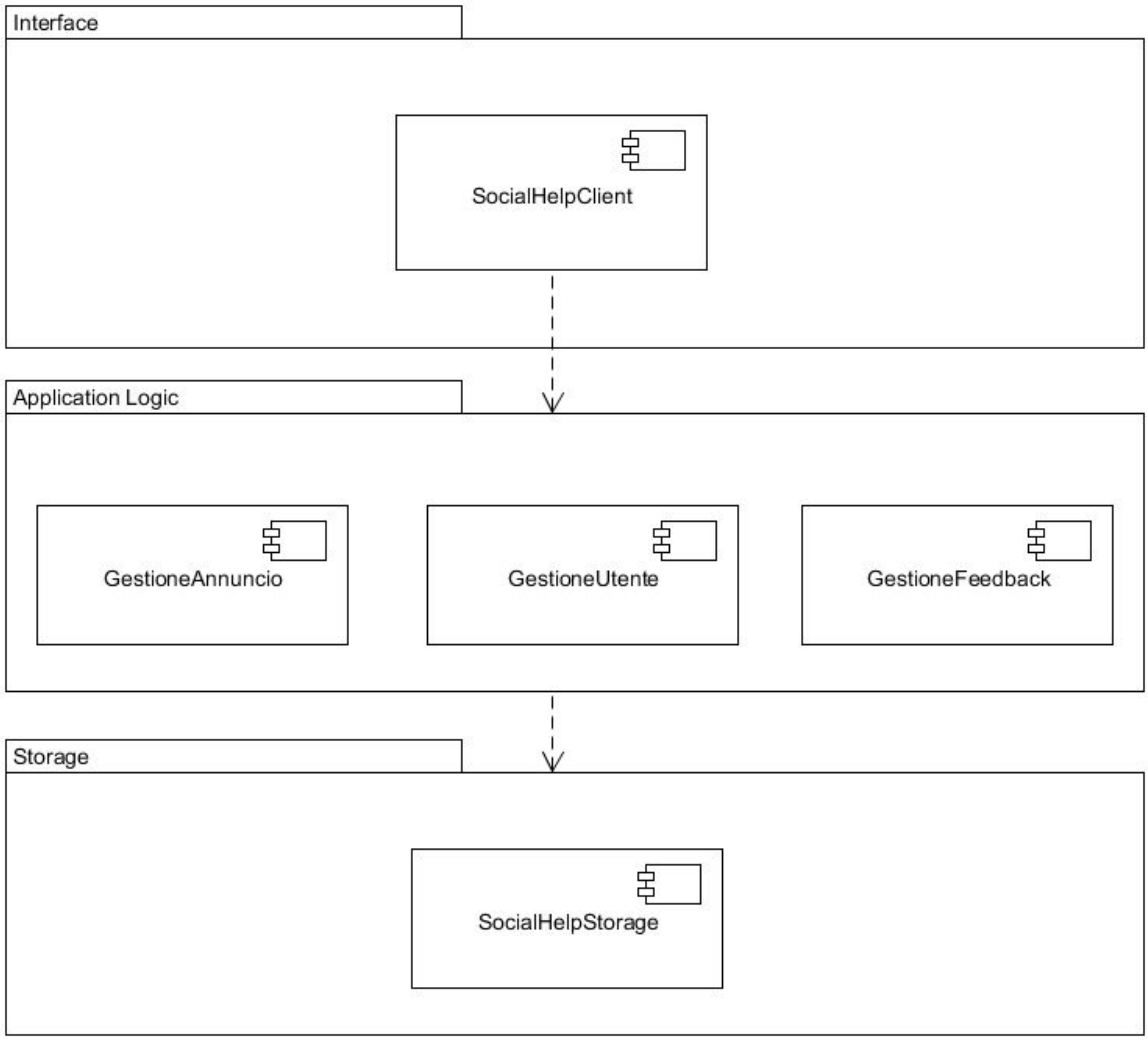
- Decomposizione del sistema, in cui il sistema viene suddiviso in diversi sottosistemi ognuno dei quali, a sua volta, è caratterizzato da servizi che offre ad altri sottosistemi.
  L'insieme dei servizi sarà denominato Interfaccia.
- Mapping Hardware/Software, riguardante la scelta della configurazione hardware del sistema, la comunicazione tra nodi, il come vengano incapsulati i servizi di un sottosistema.
- Gestione dei dati persistenti, nel quale di individuano gli oggetti che devono essere resi persistenti e quale genere di infrastruttura si deve usare per memorizzare tali oggetti.
- Politiche di accesso e Sicurezza, che ci aiuta a rappresentare tramite delle tabelle le operazioni ed informazioni utilizzabili da ogni singolo attore.
- Controllo del software globale, che ci guida su quali operazioni eseguire ed in che ordine, per garantire il corretto flusso di controllo del sistema.
- Condizioni Boundary, che includono oltre l'avvio e lo shutdown anche la gestione dei fallimenti dovuti all'invecchiamento del sistema, interruzione di corrente o anche a errori di progettazione.

2. Architettura software corrente

2.1 Panoramica

L'architettura del sistema SocialHelp è di tipo client/server. Il server riceve le richieste da parte del client, e risponde in tempo utile.

2.2 Decomposizione in sottosistemi



.

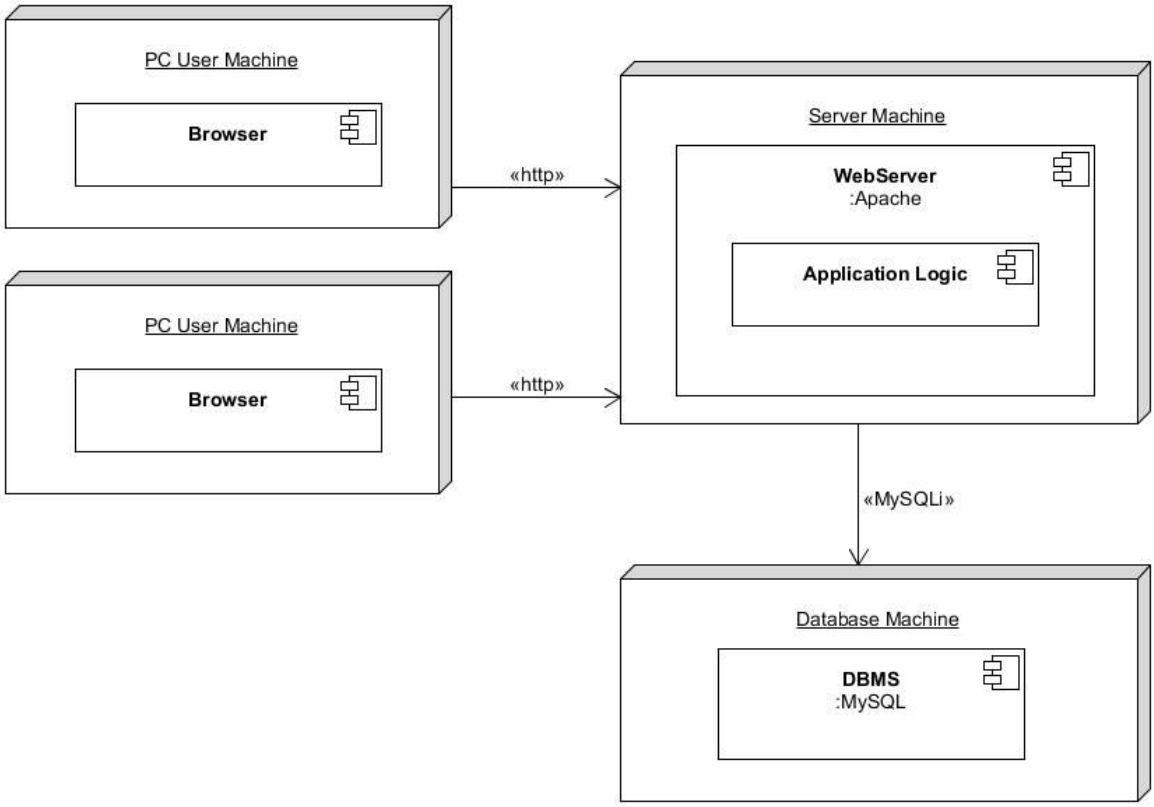
Il sottosistema esistente presenta uno stile architetturale three-tier in cui il sottosistema SocialHelpClient si occupa del front end per gli utenti, con i relativi oggetti utili a soddisfare i casi d'uso. SocialHelpServer è il sottosistema responsabile del controllo degli accessi e delega ai sottostanti sottosistemi la logica applicativa. L'ultimo tier è composto dal sottosistema SocialHelpStorage che è responsabile di memorizzare oggetti persistenti.

Interface	
SocialHelpClient	Questo sottosistema si occupa di gestire le funzionalità del front end del sistema per gli utenti loggati e per l'amministratore

Application Logic	
GestioneAnnuncio	Questo sottosistema si occupa della gestione degli annunci, tra cui la creazione di un annuncio, la cancellazione di un annuncio da parte di un Utente, la ricerca degli annunci e la gestione sulle informazioni del proprietario dell'annuncio e di tutti i dati relativi allo stesso.
GestioneUtente	Questo sottosistema si occupa della gestione degli utenti, tra cui la registrazione di un utente al sistema, il login e il logout al sistema, e la visualizzazione delle informazioni dell'utente
GestioneFeedback	Questo sottosistema si occupa della gestione dei feedback, fra cui la consultazione dei feedback di un utente e la possibilità da parte di un utente di lasciare un feedback ad un altro utente.

Storage	
SocialHelpStorage	Questo sottosistema è responsabile di memorizzare oggetti persistenti.

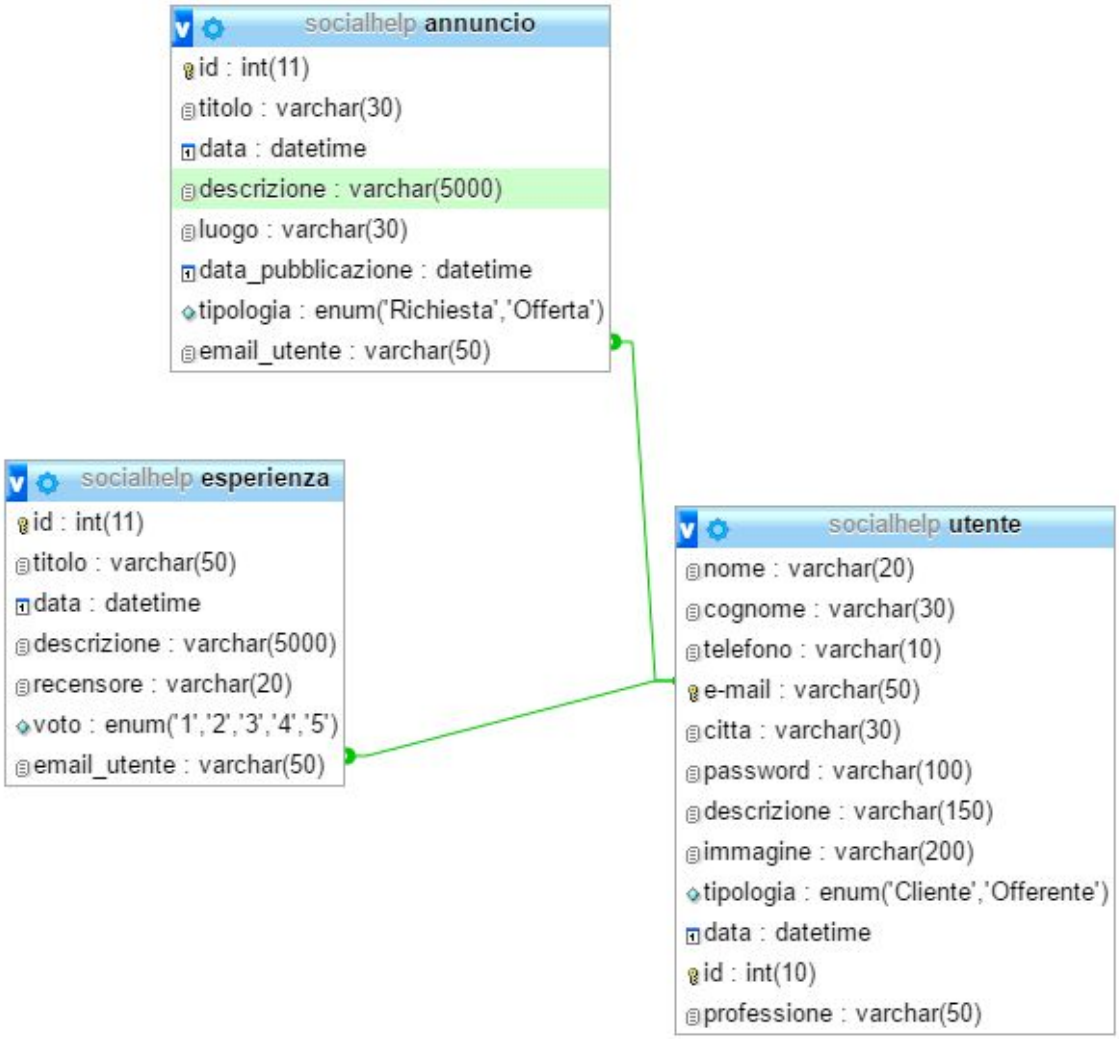
2.3 Mappatura hardware/software



Web Server: Apache versione 2.4.17

DBMS: MySQL versione 6.2

2.4 Gestione dei dati persistenti



- La tabella utente racchiude tutte le informazioni inerenti ad una persona, il particolar modo viene stabilito il ruolo assunto dall'utente all'interno del sistema dalla variabile di tipo enumerativo.
- La tabella annuncio contiene tutte le informazioni inerenti ad un annuncio, come la data di pubblicazione e il titolo, e ogni annuncio viene associato all'utente proprietario.
- La tabella esperienza, rappresenta il giudizio espresso da un utente del sistema per valutare la qualità di un annuncio. Ogni utente può avere diverse opinioni riferite a più annunci.

Utente	
Campo	**Vincoli**
Nome	Lunghezza massima: 20 caratteri
Cognome	Lunghezza massima: 30 caratteri
Telefono	Lunghezza massima: 10 caratteri
E-mail	Lunghezza massima: 50 caratteri
Città	Lunghezza massima: 30 caratteri
Password	Lunghezza massima: 100 caratteri
Descrizione	Lunghezza massima: 150 caratteri
Immagine	Lunghezza massima: 200 caratteri
Tipologia	Enum: Cliente/Offerente
Data	Datetime
Id	Lunghezza massima: 10 caratteri numerici
Professione	Lunghezza massima: 50 caratteri

Annuncio	
Campo	**Vincoli**
Id	Lunghezza massima: 11 caratteri numerici
Titolo	Lunghezza massima: 30 caratteri
Data	Datetime
Descrizione	Lunghezza massima: 5000 caratteri
Luogo	Lunghezza massima: 30 caratteri
Data_pubblicazione	Datetime
Tipologia	Enum: Richiesta/Offerente
Email_utente	Lunghezza massima: 50 caratteri

Esperienza	
Campo	**Vincoli**
Id	Lunghezza massima: 11 caratteri numerici
Titolo	Lunghezza massima: 50 caratteri
Data	Datetime
Descrizione	Lunghezza massima: 5000 caratteri
Recensore	Lunghezza massima: 20 caratteri
Voto	Enum: 1/2/3/4/5
Email_utente	Lunghezza massima: 50 caratteri
Email_utente	Lunghezza massima: 50 caratteri

2.5 Controllo degli accessi e sicurezza

ATTORI **OGGETTI**	**Utente non loggato**	**Utente**	**Amministratore**
Annuncio	visualizza	visualizza inserisci modifica ricerca	visualizza ricerca
Utente	visualizza ricerca	visualizza modifica ricerca	cancella
Feedback	visualizza ricerca	inserisciRecensio ne visualizza ricerca	visualizza ricerca

2.6 Controllo globale del software

Il controllo del flusso software viene gestito da classi php che interagendo con il client, il quale si interfaccia tramite un web browser, svolgono le varie operazioni. Il server smista ogni nuova richiesta alla classe php adeguata, inoltrando poi la risposta al client.

3. Architettura software proposta

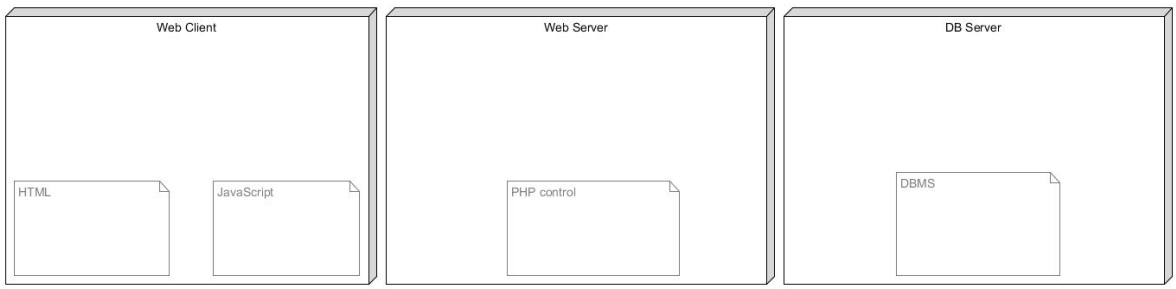
3.1 Panoramica

L'architettura del sistema CrowdMine è di tipo client/server. Il server riceve le richieste da parte del client, e risponde in tempo utile. I motivi di questa scelta sono:

● Portabilità: il sistema potrà essere utilizzato su una varietà di macchine e sistemi operativi, da computer fissi a dispositivi mobili.
● Performance: il client sarà in grado di supportare task interattivi display intensive e il server dovrà fornire operazioni CPU-intensive.
● Scalabilità: il server sarà in grado di gestire un grosso numero di client contemporaneamente, grazie alla funzionalità del cloud, che utilizza una soluzione Paas.
● Flessibilità: per ogni tipologia di utente che effettua l'accesso al sistema, vi sarà un'interfaccia grafica apposita, tramite la quale ogni attore potrà eseguire le operazioni ad esso riservate.
● Affidabilità: entrambi i componenti client e server devono essere affidabili ed essere in grado di mantenere i propri dati anche in seguito a guasti, quindi deve essere possibile effettuare dei backup periodici al database con cadenza trimestrale.

L'applicazione è strutturata su 3 livelli (three-tier):

○ Interface layer
○ Application Logic layer
○ Data layer

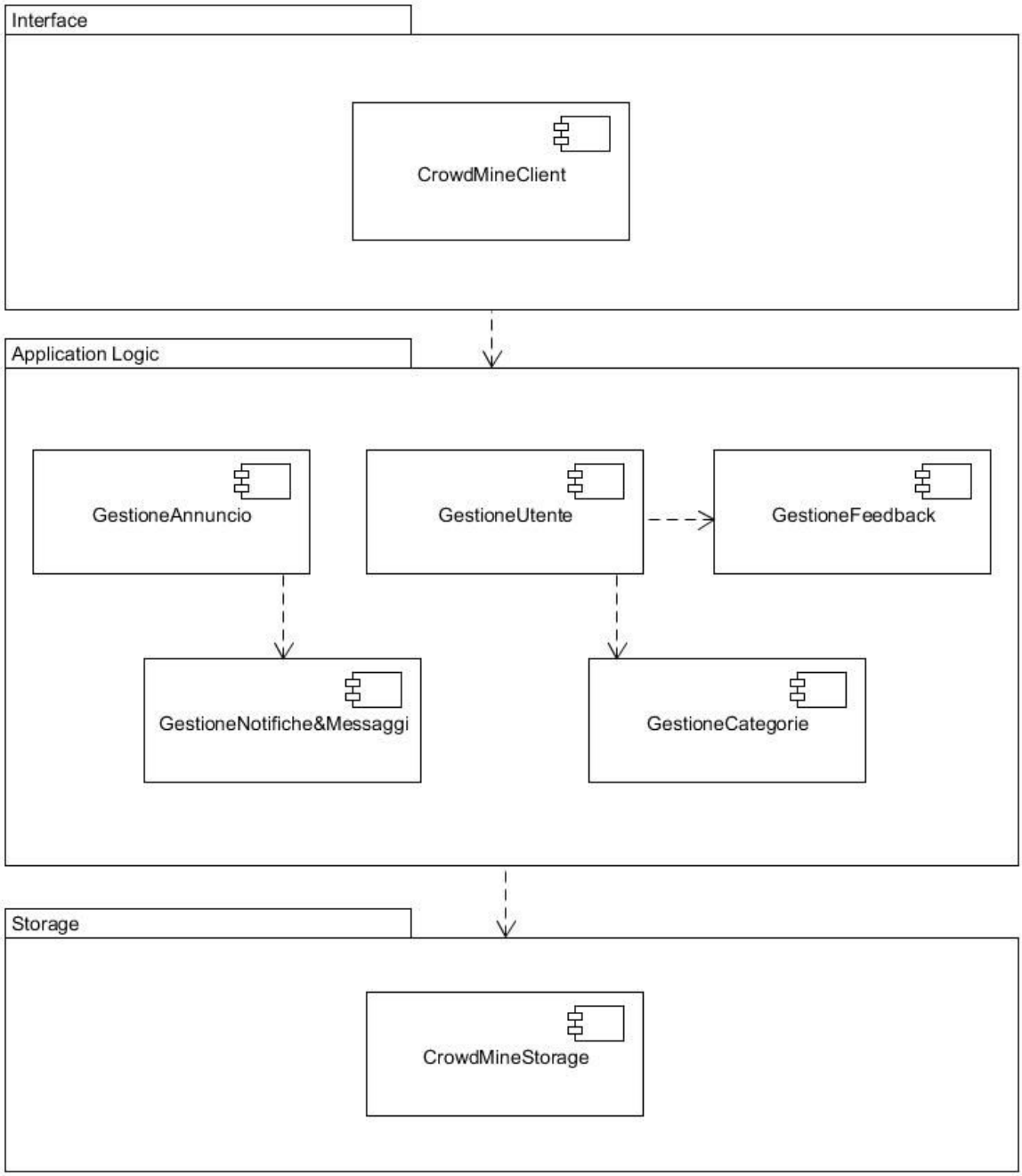


Interface layer rappresenta l'interfaccia che permette all'utente di interagire con il sistema, ricoprendo il ruolo di client in quanto utilizza un browser per richiedere pagine web al server.

Application Logic layer ha il compito di elaborare i dati da inviare al client. Spesso interroga il database, tramite lo Storage Layer, per accedere ai dati persistenti.

Data layer ha il compito di memorizzare i dati sensibili del sistema, utilizzando un DBMS, inoltre riceve le varie richieste dall' Application Logic layer inoltrandole al DBMS e restituendo i dati richiesti.

3.2 Decomposizione in sottosistemi



Lo scopo di questa attività è l'individuazione di sottosistemi a partire dai requisiti funzionali e dal modello di analisi. Per raggiungere questo obiettivo dividiamo il sistema in componenti che possono essere gestite in maniera individuale.
Il sottosistema individuato presenta uno stile architetturale three-tier in cui il sottosistema CrowdMineClient si occupa del front end per gli utenti, con i relativi oggetti utili a soddisfare i casi d'uso. CrowdMineServer è il sottosistema responsabile del controllo degli accessi e delega ai sottostanti sottosistemi la logica applicativa. L'ultimo tier è composto dal sottosistema CrowdMineStorage che è responsabile di memorizzare oggetti persistenti.

Interface	
CrowdMineClient	questo sottosistema si occupa di gestire le funzionalità del front end del sistema per gli utenti non loggati, gli utenti loggati, l'amministratore e i moderatori.

Application Logic	
GestioneAnnuncio	questo sottosistema si occupa della gestione degli annunci, fra cui la creazione di un annuncio, la segnalazione di un annuncio da parte di un utente, la cancellazione di un annuncio da parte di un Utente, la rimozione di un annuncio da parte di un moderatore, la gestione sulle informazioni del proprietario dell'annuncio e di tutti i dati relativi allo stesso.
GestioneUtente	questo sottosistema si occupa della gestione degli utenti, fra cui la registrazione di un utente al sistema, la gestione delle informazioni relative al profilo pubblico e al profilo privato di un utente, la gestione delle informazioni anagrafiche con la possibilità di eventuali modifiche, il riferimento alle macrocategorie ed alle microcategorie inserite nel profilo di un utente, la cancellazione del profilo di un utente con i relativi "dati sensibili", il ban di un utente da parte di un moderatore, la riattivazione di un utente bannato da parte di un moderatore, la segnalazione di un utente da parte di un utente, il blocco di un utente da parte di un altro utente.
GestioneFeedback	questo sottosistema si occupa della gestione dei feedback, fra cui la consultazione dei feedback di un utente, la possibilità di utilizzare un feedback come parametro di ricerca, l'ordinamento dei feedback in base ad un parametro, la possibilità da parte di un utente di lasciare un feedback ad un altro utente fra i quali è stata confermata la collaborazione, la segnalazione di un feedback, la valutazione di un feedback da parte di un moderatore.
GestioneNotifiche&Messaggi	questo sottosistema si occupa della gestione delle notifiche e dei messaggi fra cui la notifica di invio di un messaggio e risposta ad un messaggio da parte di un utente, la notifica di una candidatura di un utente ad un annuncio, la gestione dei messaggi di una conversazione privata fra due utenti, la notifica di invio di una richiesta di collaborazione e la notifica di accettazione o di rifiuto della collaborazione.

GestioneCategorie	questo sottosistema si occupa della gestione delle macro categorie e micro categorie del sistema fra cui l'aggiunta e la rimozione di una macro categoria o micro categoria al profilo di un utente, l'aggiunta di una macro categoria o micro categoria ad un annuncio, l'ordinamento delle macro categorie o micro categorie in base ad un parametro, la possibilità di avere una lista di tutte le micro categorie associate ad una macro categoria.

Storage	
CrowdMineStorage	questo sottosistema è responsabile di memorizzare oggetti persistenti.

3.3 Mappatura hardware/software

La nostra applicazione rispecchia perfettamente il **pattern MVC** (Model, View, Controler) dove il Modello sono i componenti che rappresentano i dati e la logica del problema, la Vista è l'interfaccia per l'interazione con l'utente; infine il Controllo è la logica per la gestione del flusso di controllo tra gli schemi e le operazioni sul modello.

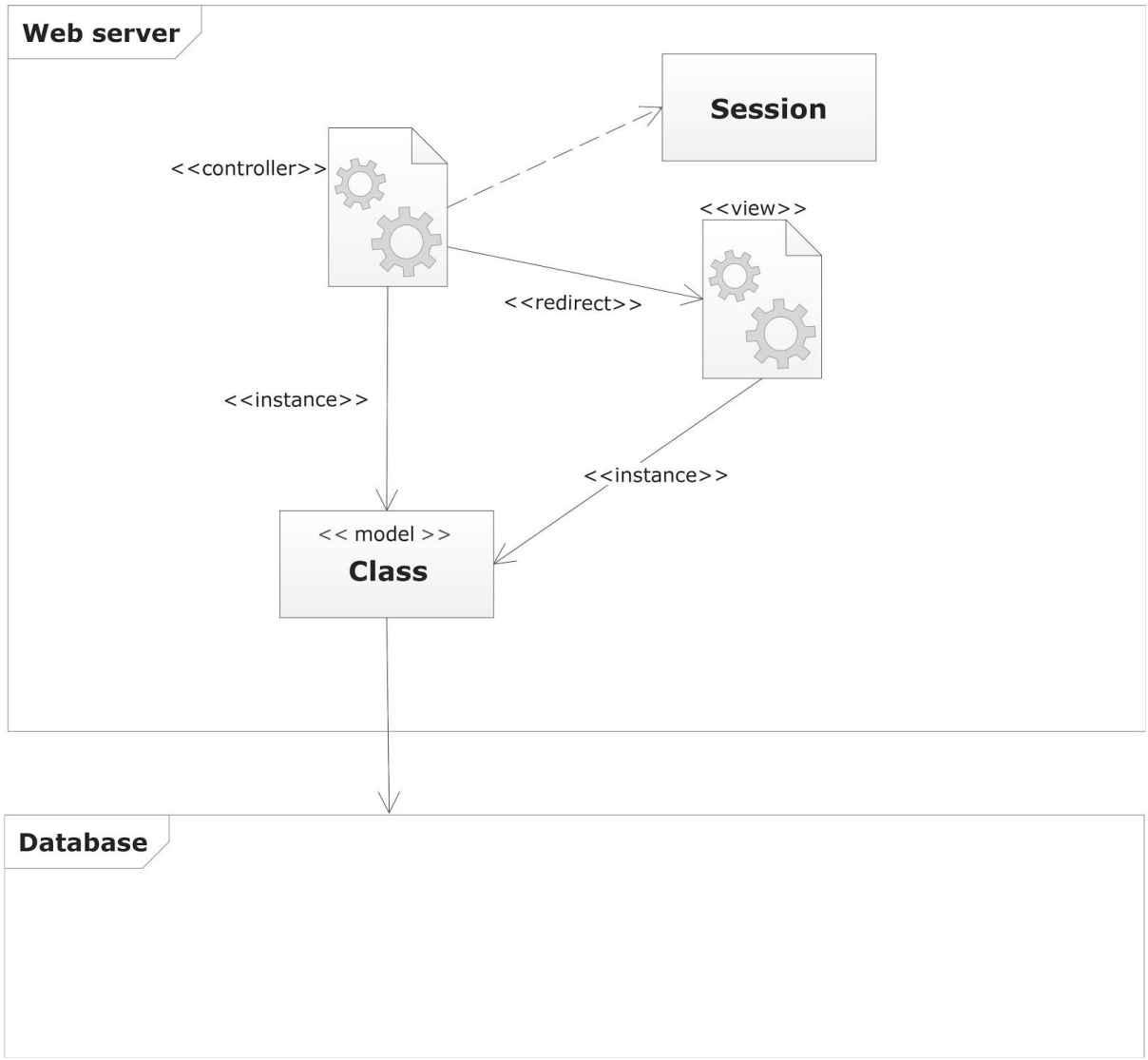
Una buona progettazione deve interessare ciascuno dei tre livelli. Tuttavia il semplice UML non ci permette di rappresentare un'applicazione web in un modello astratto, per questo utilizzeremo le **Web Application Extensions (WAE)**, definite per la prima volta da Conallen, che ci permettono di modellare:

- Pagine client
- Pagine server
- Form
- Le associazioni tra le pagine (link,submit…)

Di seguito vi è una rappresentazione dell'interazione fra le componenti a design time attraverso un **Component Diagram**. Successivamente vi è la rappresentazione della creazione dinamica delle pagine HTML e della gestione del flusso di controllo per i casi d'uso presenti nel Sistema, rispettando il pattern del MVC, attraverso **Deployment Diagram**. Per ogni diagramma UML-WAE saranno associati i sequence diagram (descritti nel Requirements Analysis Document) rappresentati da esso. Per farlo sarà indicato l'ID dei sequence diagram corrispondenti.

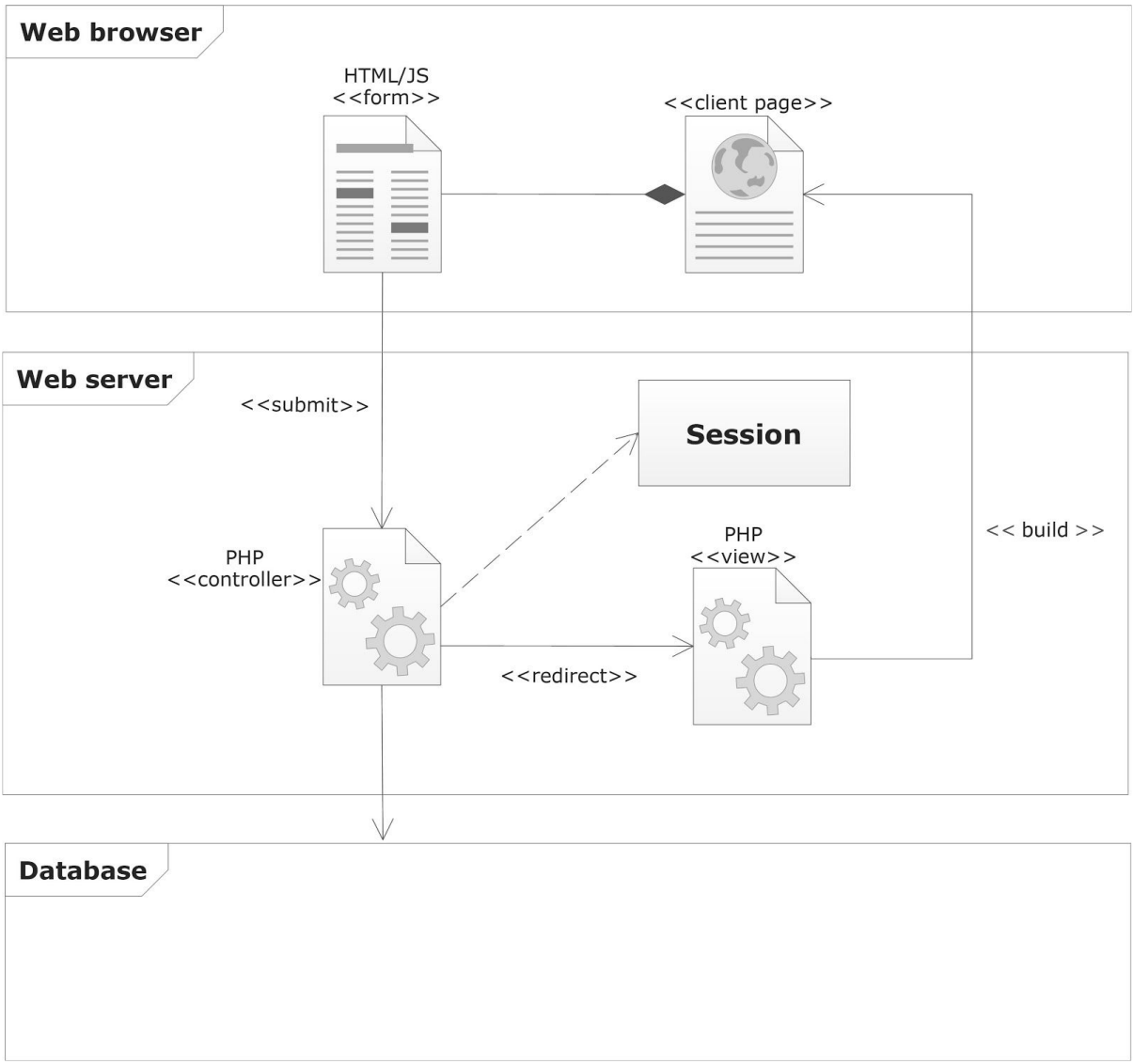
UML Component Diagram - Design Time

UML-WAE_0



UML Deployment Diagram - Run Time

UML-WAE_1

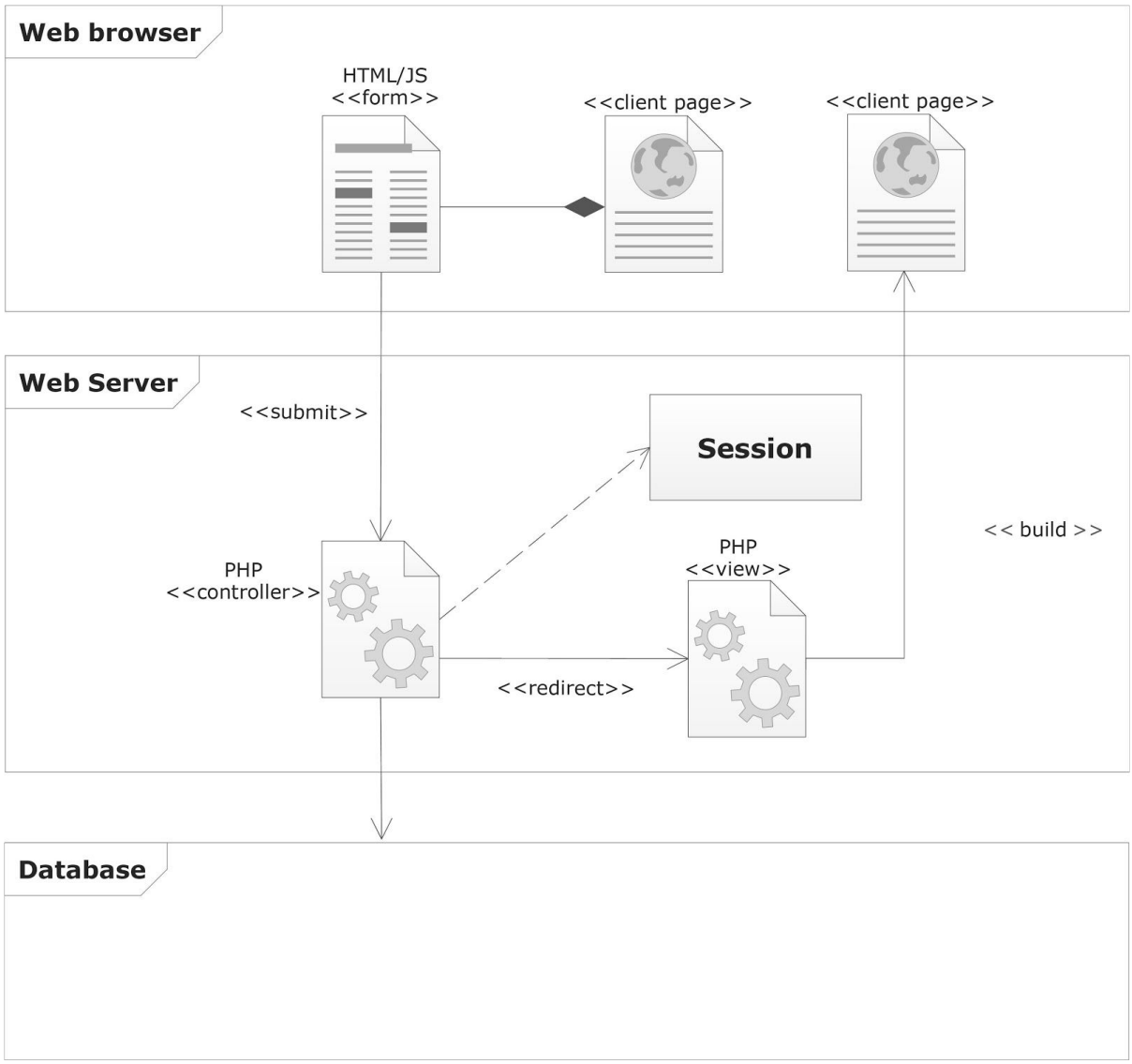


I Sequence Diagram (presenti nel Requirements Analysis Document) che seguono **UML-WAE_1** sono:

- **SD_0.2 Chiave di registrazione duplicata**
- **SD_0.3 Registrazione errore nei dati**
- **SD_2.2 Accesso alla sezione Privacy e Sicurezza**
- **SD_2.3 Visita Profilo Personale**
- **SD_2.4 Ban Utente, SD_2.8 Modifica Password**
- **SD_2.6 Accesso alla pagina di Gestione di un Moderatore**
- **SD_2.16 Destituzione Moderatore**
- **SD_2.17 Elimina Macrocategoria**
- **SD_2.18 Elimina Micro Categoria**
- **SD_2.19 Lista Ricorso Ban**
- **SD_2.20 Modifica Dati**
- **SD_2.21 Modifica MicroCategoria**
- **SD_2.22 Nessuna Macrocategoria**

- **SD_2.23 Password Errata Cancellazione Utente**
- **SD_2.24 Riabilita Utente**
- **SD_2.25 Segnala Utente**
- **SD_2.28 L'Amministratore visualizza la lista di tutti gli Utenti bannati dove due Moderatori sono in disaccordo**
- **SD_2.29 Segnalazione di un Moderatore da parte di un altro Moderatore**
- **SD_2.30 L'Amministratore visualizza la lista dei Moderatori segnalati**
- **SD_2.32 Tempo ricorso scaduto**
- **SD_2.34 Elimina Blocco Utente**
- **SD_2.35 Eliminazione Microcategoria**
- **SD_2.36 Visualizzazione Macrocategoria**
- **SD_2.37 Creazione Macrocategoria**
- **SD_2.38.Eliminazione Macrocategoria**
- **SD_2.42 Nessun Risultato Trovato Ricerca Utente**
- **SD_3.1 Inserisci annuncio**
- **SD_3.2 Modifica annuncio**
- **SD_3.3 Cancella annuncio**
- **SD_3.5 Rispondi annuncio**
- **SD_3.6 Visualizza i miei annunci**
- **SD_3.7 Aggiungi annuncio ai preferiti**
- **SD_3.8 Rimuovi annuncio dai preferiti**
- **SD_3.9 Verifica annuncio**
- **SD_3.10 Analizza e lascia attivo annuncio segnalato**
- **SD_3.11 Visualizza annunci preferiti**
- **SD_3.12 Visualizza annunci homepage**
- **SD_3.13 Segnala annuncio**
- **SD_3.14 Visualizza annunci utente ricercato**
- **SD_3.16 Errore nei dati (Annuncio)**
- **SD_3.17 Visualizza lista candidature**
- **SD_3.25 Visualizza annunci inseriti**
- **SD_3.26 Visualizza annunci reclamati**
- **SD_4.1 Utente visualizza notifiche**
- **SD_4.2 L'Utente visualizza la notifica relativa ad un annuncio di suo interesse**
- **SD_4.3 Moderatore visualizza notifiche**
- **SD_4.4 L'utente si candida per un annuncio**
- **SD_4.7 Utente accede ai Messaggi Ricevuti**
- **SD_5.1 Consultare Feedback**
- **SD_5.2 Utilizzare Feedback come parametro di ricerca**
- **SD_5.3 Ordinare i Feedback**
- **SD_5.4 Lasciare un Feedback - Collaborazione confermata**
- **SD_5.5 Segnalare un Feedback**
- **SD_5.6 Visualizzazione Feedback Segnalati**
- **SD_5.7 Valutazione ed back Accettazione Feedback**
- **SD_5.8 Valutazione Feedback Eliminazione Feedback**
- **SD_6.1 Visualizzazione statistiche MacroCategorie**
- **SD_6.2 Visualizzazione statistiche feedback profilo**
- **SD_6.3 Visualizzazione classifica dei migliori utenti in base ai feedback per MacroCategoria**
- **SD_6.4 Visualizzazione classifica dei migliori utenti in base ai feedback per MicroCategoria**

UML-WAE_2

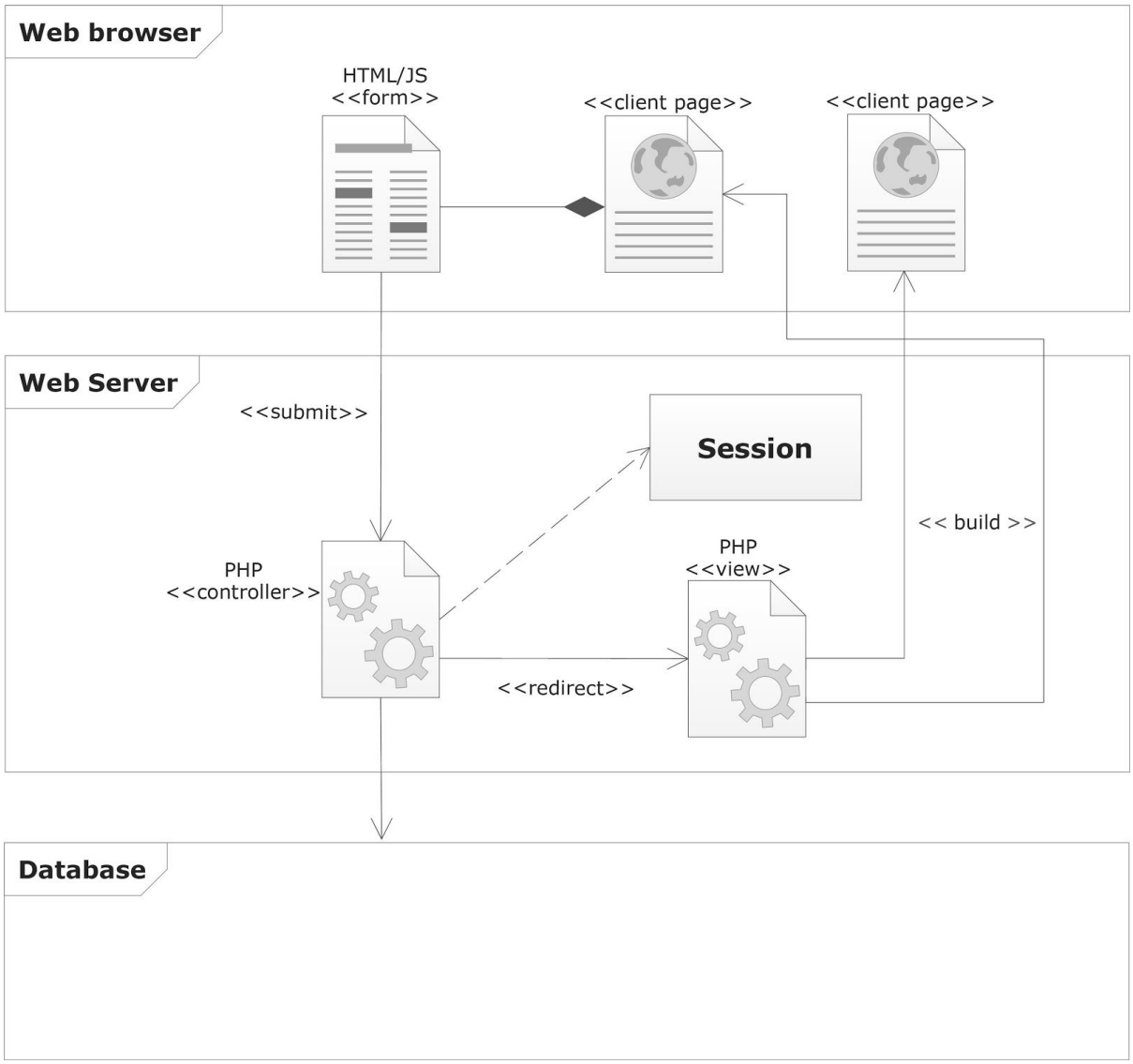


I Sequence Diagram (presenti nel Requirements Analysis Document) che seguono **UML-WAE_2** sono:

- **SD_0.1 Registrazione,**
- **SD_1.2 Logout,**
- **SD_2.5 Visualizzazione Annunci segnalati,**
- **SD_2.7 Visualizzazione Lista Utenti Segnalati,**
- **SD_2.9 Ricerca Utente,**
- **SD_2.13 Visualizzazione Utenti bannati,**
- **SD_2.14 Visualizzazione Microcategorie,**
- **SD_2.26 Visualizza Profilo Utente,**
- **SD_2.31 Ricorso al Ban,**
- **SD_3.4 Ricerca annuncio,**
- **SD_3.15 Visualizza annunci segnalati,**
- **SD_3.18 Proposta sospensione annuncio segnalato,**
- **SD_3.20 Reclamo di un annuncio,**
- **SD_3.21 Discordanza tra moderatori,**
- **SD_3.22 Concordanza tra moderatori,**
- **SD_3.23 Conferma amministratore,**
- **SD_3.24 Sospensione amministratore,**

- **SD_4.8 L’utente accede ad una conversazione privata,**
- **SD_4.9 L’utente invia richiesta di collaborazione,**
- **SD_4.10 L’utente accetta la richiesta di collaborazione,**
- **SD_4.11 L’utente rifiuta il candidato,**
- **SD_6.7 Generazione numero di annunci pubblicati sulla piattaforma,**
- **SD_6.8 Generazione numero annunci pubblicati in un intervallo di tempo,**
- **SD_6.9 Generazione numero annunci in un intervallo di tempo in base alla MacroCategoria,**
- **SD_6.10 Generazione numero annunci in un intervallo di tempo in base alla MicroCategoria,**
- **SD_6.11 Generazione Statistiche sulle MacroCategorie Scelte dall’Utente per l’Offerta,**
- **SD_6.12 Generazione Statistiche sulle MicroCategorie Scelte dall’Utente per l’Offerta,**
- **SD_6.13 Generazione Statistiche sulle MacroCategorie Scelte dall’Utente,**
- **SD_6.14 Generazione Statistiche sulle MicroCategorie Scelte dall’Utente**

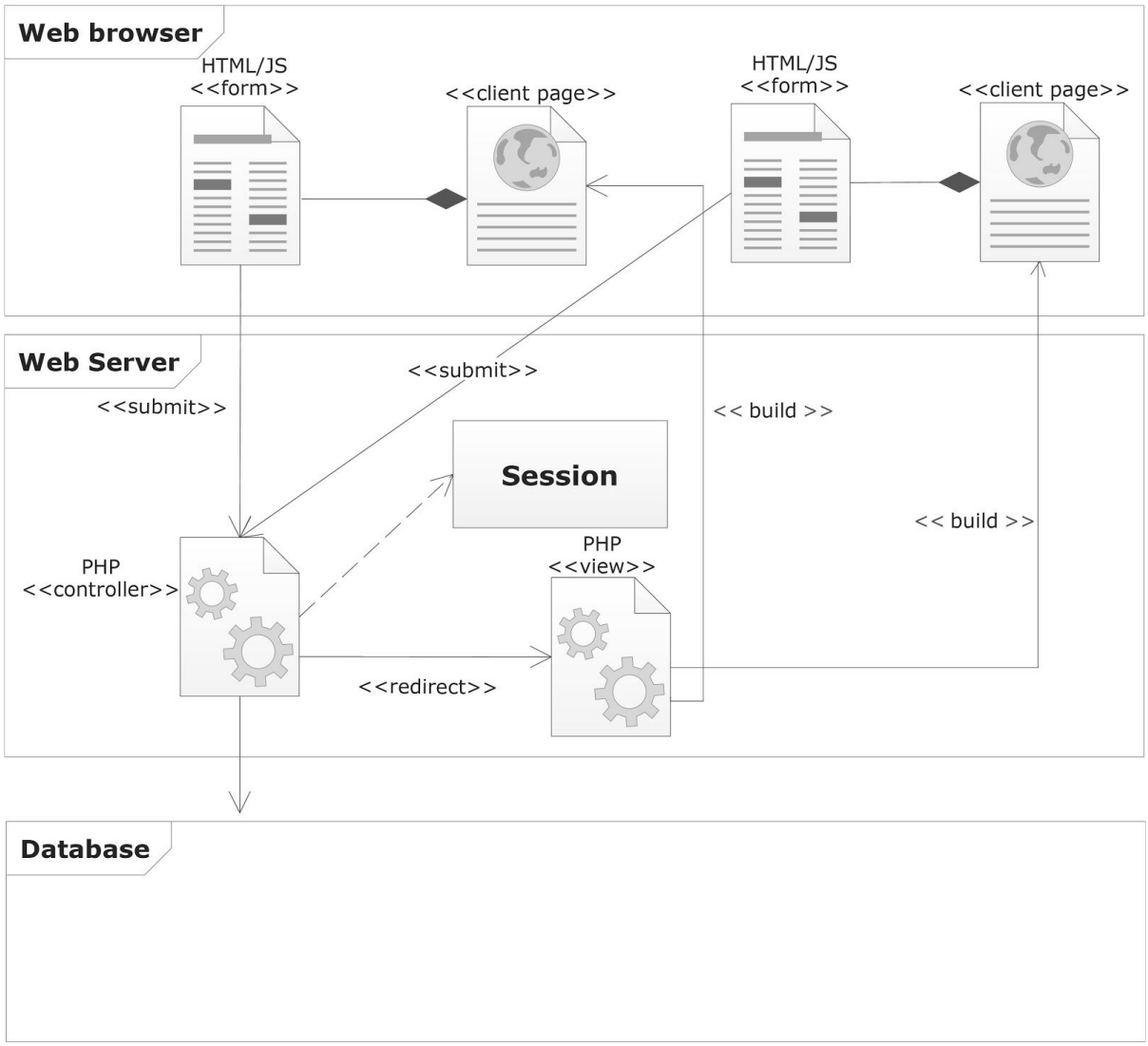
UML-WAE_3



I Sequence Diagram (presenti nel Requirements Analysis Document) che seguono **UML-WAE_3** sono:

- **SD_4.5 Utente invia messaggio**
- **SD_4.6 L'Utente risponde ad un messaggio privato**
- **SD_3.19 Commento ad un annuncio**

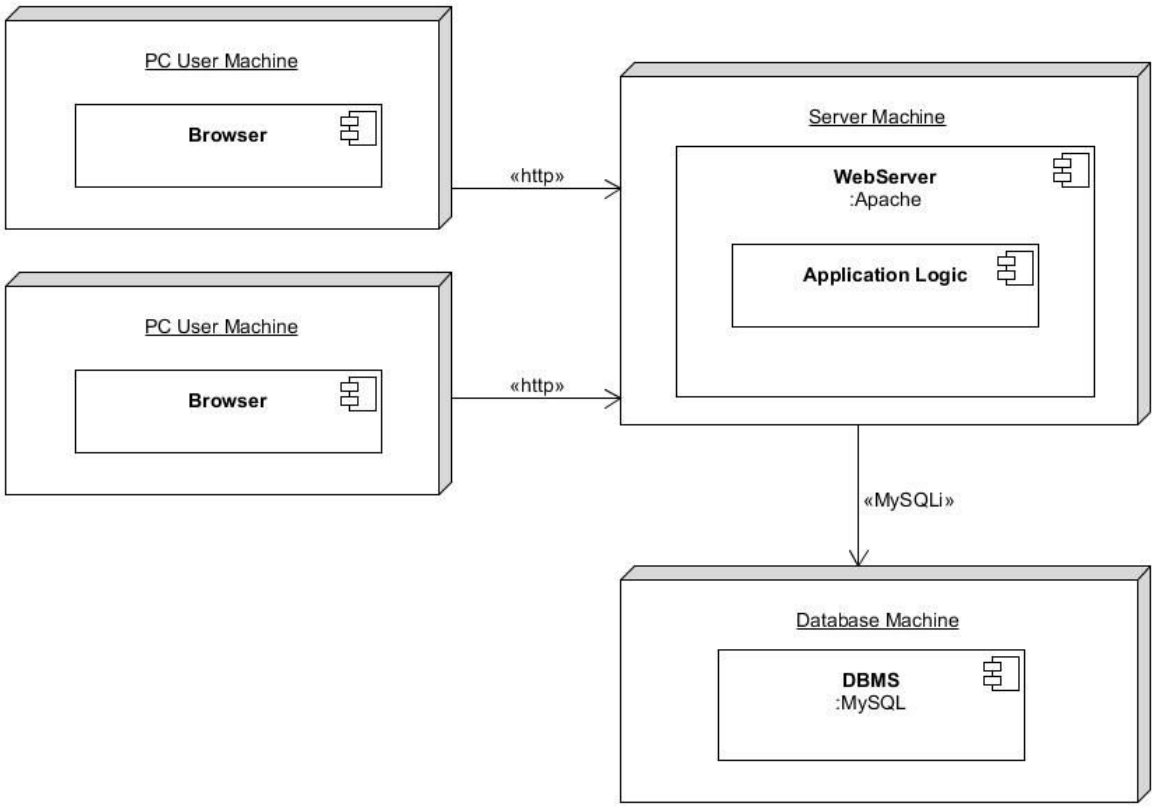
UML-WAE_4



I Sequence Diagram (presenti nel Requirements Analysis Document) che seguono **UML-WAE_4** sono:

- **SD_2.1 Cancellazione Account**
- **SD_2.4 Ban Utente**
- **SD_2.8 Modifica Password**
- **SD_2.10 Aggiunta Macrocategoria nella propria area di competenza**
- **SD_2.11 Aggiunta Micro Categoria nella propria area di competenza**
- **SD_2.12 Riattivazione di un Utente Bannato**
- **SD_2.15 Elezione Moderatore**
- **SD_2.39 Password Errata Ban Utente**
- **SD_2.40 Password Errata Modifica Password**
- **SD_2.41 Password Errata Elezione Moderatore**

Nell'applicazione proposta i Client sono Web browser, mentre il Web Server contiene le componenti che si occupano della logica applicativa, dell'interfaccia e dei dati. Per realizzare le componenti del Client vengono utilizzate pagine php che permettono di creare gli oggetti Boundary. Quando il Web server riceve una richiesta per una pagina php essa viene analizzata dall'interprete del linguaggio, il quale restituisce un file contenente solo il codice HTML e Java Script che deve essere inviato al Web browser. Il Web server invia query al Database MySQL utilizzando il protocollo MySQLi.



Web Server: Apache versione 2.4.17

DBMS: MySQL versione 6.2

3.4 Gestione dei dati persistenti

Si rimanda al file System Design Document - Gestione dei dati persistenti - CrowdMine

3.5 Controllo degli accessi e sicurezza

CrowdMine è un sistema multiutente, quindi attori differenti hanno il permesso di eseguire diverse operazioni su vari insiemi di oggetti. Per schematizzare al meglio il controllo degli accessi abbiamo suddiviso per tipologia di utente le azioni consentite, al fine di ottenere una visione più compatta e dettagliata grazie ad una matrice degli accessi riportata di seguito:

ATTORI **OGGETTI**	**Utente non loggato**	**Utente**	**Moderatore**	**Amministratore**
Annuncio	visualizza	visualizza, crea, cancella annuncio personale, modifica, segnala	visualizza, crea, cancella annuncio personale, modifica, segnala, rimuovi con ricorso, vota al ricorso	visualizza, crea, cancella annuncio personale, modifica, segnala, rimuovi
Utente	visualizza	crea Account, cancella Account, modifica Dati, segnala, blocca	crea Account, cancella Account, modifica Dati, segnala, blocca, banna con ricorso, vota al ban con ricorso	crea Account, cancella Account, modifica Dati, segnala, blocca, banna con ricorso, rimuovi
Macrocategoria		aggiungi al profilo, rimuovi dal profilo, aggiungi ad un annuncio	aggiungi al profilo, rimuovi dal profilo, aggiungi ad un annuncio	aggiungi al profilo, rimuovi dal profilo, aggiungi ad un annuncio, aggiungi al sistema, rimuovi dal sistema
Microcategoria		crea, aggiungi al profilo, rimuovi dal profilo	crea, aggiungi al profilo, rimuovi dal profilo, rimuovi dal sistema	crea, aggiungi al profilo, rimuovi dal profilo, rimuovi dal sistema
Feedback	visualizza	aggiungi, rimuovi feedback	aggiungi, segnala,	aggiungi, rimuovi

		personale, segnala	rimuovi	
Notifica		visualizza	visualizza	visualizza
Commento		aggiungi, segnala, rimuovi c. personale	aggiungi, segnala, rimuovi	aggiungi, segnala, rimuovi
Messaggio		invia, visualizza	invia, visualizza	invia, visualizza
Candidatura		effettua	effettua	effettua

3.6 Controllo globale del software

Il controllo del flusso software viene gestito da classi php che interagendo con il client, il quale si interfaccia tramite un web browser, svolgono le varie operazioni. Il server smista ogni nuova richiesta alla classe php adeguata, inoltrando poi la risposta al client.

3.7 Boundary conditions

Le condizioni limite riguardano l'accensione e lo spegnimento del sistema per quanto riguarda il lato Server. Dal lato Client si riferiscono agli errori di connessione al server.

Scenari

Nome Scenario	Startup Server
Istanze di Attori **Partecipanti**	Angelo: Admin
Flusso di Eventi	1. Giorgio decide di voler avviare il sistema e quindi clicca sul pulsante "Avvia". 2. Il sistema, con le opportune procedure di avvio, attiva i server e i relativi servizi in remoto rendendosi disponibile ad eventuali richieste. 3. Il sistema notifica il successo della procedura.

Nome Scenario	Shutdown Server

Istanze di Attori **Partecipanti**	Angelo: Admin
Flusso di Eventi	1. Lino decide di voler arrestare il sistema e quindi accede alla pagina dedicata e clicca sul pulsante “Arresta”. 2. Il sistema effettua una scansione per verificare se ci sono ancora richieste in sospeso. 3. Il sistema porta a termine le eventuali richieste in sospeso 4. Tramite le opportune procedure di arresto il sistema disattiva i servizi in remoto e il server. 5. Il sistema notifica il successo della procedura.

Casi d'uso

ID	UC_Startup
Nome Caso Uso	Startup Server
Istanze di Attori **Partecipanti**	Angelo: Admin
Condizione di Entrata	L'amministratore accede al sistema.
Flusso di Eventi	Utente 1. Giorgio accede al sistema e clicca sul pulsante “Avvia”. Sistema 2. CrowdMine accende il server e attiva i servizi in remoto rendendosi disponibile per le richieste notificando il successo dell’operazione all’utente.
Condizione di Uscita	Il server è attivo e i relativi servizi sono disponibili.

ID	UC_Shutdown

Nome Caso Uso	Shutdown Server
Istanze di Attori **Partecipanti**	Angelo: Admin
Condizione di Entrata	L'amministratore accede al sistema.
Flusso di Eventi	Utente 1. Lino accede al sistema e clicca sul pulsante “Spegni”. Sistema 2. CrowdMine effettua una scansione per verificare se ci sono eventuali richieste in sospeso, porta a termine le richieste e avvia le procedure di arresto. Il sistema notifica il successo dell’operazione all’utente.
Condizione di Uscita	Il server si è spento correttamente.

4. Servizi dei sottosistemi

4.1 Gestione Utente

Sottosistema	Descrizione
Gestione Utente	Sottosistema che gestisce la registrazione di un utente, la sua autenticazione e le operazioni necessarie alla sua gestione.
Servizio	**Descrizione**
creaUtente()	Permette di inserire un nuovo utente all'interno del database.
inoltraLogin()	Permette ad un utente di poter effettuare l'accesso al sistema.
inoltraLogout()	Permette ad un utente di uscire dal sistema.
cancellaDatiUtente()	Permette ad un utente di cancellare il proprio account.
bannaUtente()	Permette di bannare un utente.
getUtentiSegnalati()	Permette di visualizzare la lista di tutti gli utenti segnalati.
cambiaPassword()	Permette ad un utente di cambiare la propria password.
getUtenti()	Permette di visualizzare una determinata lista di utenti.
aggiungiMacroCategoria()	Permette ad un utente di aggiungere una macro categoria alla propria area di competenza.
aggiungiMicroCategoria()	Permette ad un utente di aggiungere una micro categoria alla propria area di competenza.
getUtentiBannati()	Permette di visualizzare la lista di tutti gli utenti bannati.
eleggiMod()	Permette di eleggere un utente a moderatore.
destituisci()	Permette di destituire un moderatore a utente.
getListaRicorsoBan()	Permette di visualizzare la lista di tutti gli utenti che hanno fatto ricorso al ban.
modificaDati()	Permette ad un utente di modificare i propri dati.
riabilitaUtente()	Permette di riattivare un utente.
segnalaUtente()	Permette ad un utente di segnalare un altro utente.
cercaUtente()	Permette di cercare un utente all'interno del database.

bloccaUtente()	Permette ad un utente di bloccare i messaggi provenienti da un determinato utente.
sbloccaUtetne()	Permette ad un utente di sbloccare i messaggi provenienti da un determinato utente.
getListaPreferiti()	Permette di visualizzare la lista degli annunci preferiti
getListaUtenti()	Permette di visualizzare la lista di tutti gli utenti.

4.2 Gestione Annunci

Sottosistema	**Descrizione**
Gestione Annunci	Sottosistema che gestisce le operazioni riguardanti gli annunci.
Servizio	**Descrizione**
getAnnunciSegnalati()	Permette di visualizzare la lista di tutti gli annunci segnalati.
creaAnnuncio()	Permette di creare un annuncio.
aggiornaAnnuncio()	Permette ad un utente di modificare uno dei suoi annunci.
cancellaAnnuncio()	Permette di eliminare un annuncio dal sistema.
ricercaAnnuncio()	Permette di cercare un determinato annuncio.
getAnnuncio()	Permette di visualizzare un annuncio.
ricercaAnnunciUtente()	Permette di visualizzare la lista di annunci in base ad un utente.
inserisciCandidatura()	Permette ad un utente di candidarsi ad un annuncio.
inserisciPreferito()	Permette ad un utente di inserire un annuncio nella propria lista di annunci preferiti.
rimuoviPreferito()	Permette ad un utente di rimuovere un annuncio dalla propria lista di annunci preferiti.
convalidaAnnuncio()	Permette ad un moderatore di convalidare un annuncio.
confermaValidita()	Permette ad un moderatore di convalidare un annuncio segnalato.

getAnnunciHomePage()	Permette di visualizzare la lista di annunci che andranno nella home page.
segnalaAnnuncio()	Permette ad un utente di segnalare un annuncio.
getListaAnnunciSegnalati()	Permette di visualizzare la lista di tutti gli annunci segnalati.
getAnnuncioConCandidati()	Permette di visualizzare la lista di tutti i candidati ad un annuncio.
commentaAnnuncio()	Permette di aggiungere un commento ad un annuncio.
inoltraReclamo()	Permette ad un utente di effettuare un reclamo ad un annuncio sospeso.
sendSospensione()	Permette di sospendere un annuncio.
sendConferma()	Permette all'amministratore di confermare un annuncio segnalato.
getListaAnnunciInseriti()	Permette di visualizzare la lista di tutti gli annunci inseriti.
getAnnunciListaReclamati()	Permette di visualizzare la lista di tutti gli annunci eliminati a cui l'utente ha fatto il reclamo.
getNumeroAnnunci()	Permette di visualizzare il numero di annunci attivi nel sistema.
getClassificaAnnunci()	Permette di visualizzare la classifica degli annunci attivi nel sistema

4.3 Gestione Notifiche e Messaggi

Sottosistema	Descrizione
Gestione Notifiche e Messaggi	Sottosistema che gestisce le operazioni riguardanti i messaggi e le notifiche.
Servizio	**Descrizione**
inoltraNotifica()	Permette ad un utente di inviare una notifica ad un altro utente.
visualizzaListaNotifiche()	Permette di visualizzare la lista di tutte le notifiche.
inviaMessaggio()	Permette ad un utente di inviare un messaggio ad un altro utente.
caricaMessaggi()	Permette ad un utente di visualizzare i messaggi scambiati in precedenza con un altro utente.
inviaRichiestaCollaborazione()	Permette ad un utente di richiedere una collaborazione ad un altro utente tramite un annuncio.
accettaCollaborazione()	Permette ad un utente di accettare una collaborazione con un altro utente.

4.4 Gestione Feedback

Sottosistema	Descrizione
Gestione Feedback	Sottosistema che gestisce le operazioni riguardanti ai feedback.
Servizio	**Descrizione**
getListaFeedback()	Permette di visualizzare la lista di tutti i feedback inerenti ad un annuncio.
findUserBy()	Permette di cercare un utente in base
sortListaFeedback()	Permette di ordinare una lista di feedback in base ad un parametro passato alla funzione.
insertFeedback()	Permette di inserire un feedback ad un utente.
setStatus()	Permette cambiare lo status di un determinato feedback.
findFeedbackSegnalati()	Permette di visualizzare la lista di tutti i feedback segnalati.
removeFeedback()	Permette di togliere un feedback ad un utente.

4.5 Gestione Categorie

Sottosistema	Descrizione
Gestione Categorie	Sottosistema che gestisce le operazioni riguardanti sia le macrocategorie che le microcategorie.
Servizio	**Descrizione**
inserisciMicroCategoria()	Permette di inserire una nuova microcategoria.
getMicroCategorie()	Premette di visualizzare microcategorie.
eliminaMicroCategoria()	Permette di eliminare una microcategoria.
modificaMicroCategoria()	Permette di modificare una microcategoria.
getListaMicroCategoria()	Premette di visualizzare la lista di tutte le microcategorie.
addMacroCategoria()	Permette di aggiungere una Macrocategoria.
verificaMacroCategoria()	Permette di verificare se una macrocategoria esiste nel sistema.

5. Glossario

- **CrowdMine:** nome del sistema che si sta sviluppando
- **Amministratore**: il termine identifica la persona che gestisce la piattaforma CrowdMine
- **Moderatore:** il termine identifica le persone che hanno dei privilegi in più rispetto all'utente loggato
- **Utente loggato:** il termine identifica le persone che sono correttamente loggate al sistema
- **Utente non loggato:** il termine identifica le persone che non sono loggate al sistema, che quindi, avranno un insieme di funzionalità disponibili più ristretto rispetto agli utenti loggati
- **Feedback:** valutazione recensita da un utente per valutare un annuncio
- **RAD:** documento di Analisi dei Requisiti
- **DBMS:** sistema di gestione di basi di dati
- **Database:** insieme organizzato di dati persistenti