Laurea Magistrale in informatica - Università di Salerno
Corso di *Gestione dei Progetti Software* - Prof.ssa F. Ferrucci

COMUN-ITY

JOIN THE CONNECTED COMMUNITY

System Design Document

Riferimento	C17_SDD
Versione	1.0
Data	28/11/2022
Destinatario	Prof.ssa Filomena Ferrucci,
Presentato da	Michele Iannucci, Elio Testa
Approvato da	

Revision History

Data	Versione	Descrizione	Autori
24/11/2022	0.1	Struttura del documento	Iannucci, Testa
28/11/2022	0.2	Stesura Design Goals	Tutto il Team
29/11/2022	0.3	Architettura Sistema Proposto	Andrea Aceto Biagio Andreucci Alessandro Falcone Michele Rabesco
29/11/2022	0.4	Trade-off	Gabriele Santoro
30/11/2022	0.5	Servizi dei sottosistemi	Biagio Andreucci
30/11/2022	0.6	Mapping Hardware/Software	Andrea Aceto
30/11/2022	0.7	Gestione dei dati persistenti	Alessandro Falcone
1/12/2022	0.8	Identificazione casi limite e vincoli di sicurezza	Michele Rabesco
2/12/2022	0.9	Glossario	Tutto il Team
2/12/2022	1.0	Revisione	Alessandro Falcone

Team

Nominativo	Ruolo
Michele Iannucci	Project Manager
Elio Testa	Project Manager
Alessandro Falcone	Team Member
Andrea Aceto	Team Member
Biagio Andreucci	Team Member
Gabriele Santoro	Team Member
Michele Rabesco	Team Member

Sommario

1. Introduzione

Cavalcando l'onda della transizione digitale, supportata dal progetto "Italia Digitale 2026", **Comun-ity** è l'idea di piattaforma Web comunale per promuovere la creazione delle Smart Communities.

1.1 Obiettivo del sistema

Il sistema che si vuole realizzare ha come obiettivo principale quello di migliorare la vita del singolo cittadino nel quotidiano. Tramite una piattaforma online il cittadino potrà registrarsi e richiedere favori o lavori svolti da professionisti.
Allo stato attuale l'interazione tra i cittadini avviene tramite social network, telefonate e software di messaggistica General Purpose, i quali evidenziano i seguenti problemi:

- Per un cittadino risulta scomodo chiedere aiuto per banali servizi o reperire professionisti contattando ogni conoscente tramite software di messaggistica General Purpose oppure pubblicando annunci sui vari social network, in quanto poi dovrebbe comunque interfacciarsi con ogni utente e verificarne la disponibilità.
- Per un professionista può essere più comodo cercare un lavoro da svolgere avendo una lista di richieste disponibili.

1.2 Design Goals

Rank	ID Design Goal	Descrizione	Categoria	RNF di origine
5	DG_1 Usabilità	L'utente deve essere in grado di accedere dal 90% dei dispositivi con accesso al Web tramite i principali browser.	End User	RNF_US_1
10	DG_2 Usabilità	La grafica dovrebbe essere responsive su tutti i dispositivi sopracitati.	End User	RNF_US_1
6	DG_3 Usabilità	Il sistema deve garantire al 98% degli utenti una fruizione senza errori nel flusso di navigazione.	End User	RNF_US_2
4	DG_4 Disponibilità	Il sistema dovrà poter essere raggiungibile 24/7 a meno di manutenzioni programmate, malfunzionamenti o imprevisti.	Dependability	RNF_AF_1
2	DG_5 Sicurezza	Il sistema deve fornire una connessione sicura, al fine di	Dependability	RNF_AF_2

		salvaguardare i dati degli utenti da accessi non autorizzati.		
3	DG_6 Robustezza	Il sistema deve garantire robustezza, gestendo nel modo previsto il 90% degli input errati da parte dell'utente.	Dependability	RNF_AF_2
11	DG_7 Throughput	Il sistema dovrà supportare 200 task al giorno.	Performance	RNF_PR_1
12	DG_8 Tempo di risposta	Il sistema dovrebbe notificare il richiedente via email in meno di 30 secondi nel 90% dei casi.	Performance	RNF_PR_2
13	DG_9 Tempo di risposta	Il sistema dovrebbe notificare l'admin via email in meno di 30 secondi nel 90% dei casi.	Performance	RNF_PR_3
9	DG_10 Scalabilità	L'architettura del sistema dovrebbe poter essere ampliata per gestire carichi crescenti di utenza.	Maintenance	RNF_SO_1
8	DG_11 Portabilità	Il sistema deve essere sviluppato in maggior parte da codice riusabile.	Maintenance	RNF_SO_2
7	DG_12 Utilizzo di framework	Il sistema deve usare dei framework per aumentare la riusabilità del codice.	Maintenance	RNF_SO_2
1	DG_13 Costi di sviluppo	Il costo previsto per lo sviluppo del sistema ammonta a 50 ore/uomo (5 software engineer + 2 project manager)	Cost	Documenti di management

1.3 Trade-off

Trade-Off	Descrizione
Tempo di risposta vs Sicurezza	Il sistema, per aumentare la sicurezza, potrebbe richiedere un tempo di risposta maggiore.
Tempo di risposta vs Robustezza	Al fine di aumentare la robustezza del codice, verranno effettuati controlli sugli input, a discapito dei tempi di risposta.
Costi di sviluppo vs Usabilità	Al fine di contenere i costi di sviluppo, l'usabilità della piattaforma sarà garantita sul 90% dei dispositivi con accesso al Web tramite i principali browser
Costi di sviluppo vs Portabilità	Al fine di garantire un tempo di sviluppo inferiore, il codice sarà, per la maggior parte, riusabile.
Costi di sviluppo vs Utilizzo di framework	Al fine di ridurre i costi di sviluppo il sistema implementerà framework e componenti già esistenti.

1.4 Definizioni, acronimi e abbreviazioni

Definizioni: vedi Glossario.

Acronimi:

- **RAD**: Requirement Analysis Document;
- **SDD**: System Design Document;
- **DG**: Design Goal;
- **RNF**: Requisito Non Funzionale;
- **US**: Usabilità;
- **AF**: Affidabilità;
- **PR**: Prestazioni;
- **SO**: Sostenibilità;
- **CT**: Cittadino;
- **PRO**: Professionista;
- **AD**: Admin;
- **MVC**: Model View Controller;
- **HTTP**: HyperText Transfer Protocol;
- **DBMS**: DataBase Management System;

- **JSP**: Java Server Page;
- **SC**: Scenario;
- **SU**: Startup;
- **SD**: Shutdown;
- **FA**: Failures;
- **UC**: Use Case;
- **UPS**: Uninterruptible Power Supply.

1.5 Riferimenti

- B. Bruegge, A.H. Dutoit, Object Oriented Software Engineering – Using UML, Patterns and Java, Prentice Hall.
- Documento di Statement of Work relativo a questo progetto.
- Documento di Requirement Analysis Document relativo a questo progetto.

1.6 Panoramica

Il presente documento è organizzato in cinque sezioni:

- **Introduzione**: Viene descritto in generale lo scopo del sistema, gli obiettivi di design che il sistema propone di raggiungere;
- **Architettura Sistema Corrente**: Descrive, nel caso esista, lo stato attuale e le funzionalità offerte dal sistema corrente;
- **Architettura Sistema Proposto**: Viene presentata l'architettura del sistema proposto, in cui sarà gestita la decomposizione in sottosistemi, il mapping hardware/software, la gestione dei dati persistenti, il controllo degli accessi e sicurezza, il controllo del flusso globale del sistema, le condizioni limite;
- **Servizi dei Sottosistemi**: Vengono presentati i servizi dei sottosistemi;
- **Glossario**: Raccolta di vocaboli meno comuni utilizzati nella stesura del documento.

2. Architettura del Sistema corrente

Al momento, non esiste alcun software che racchiude tutte le funzionalità di Comun-ity in un unico servizio. Le possibili alternative a questo software sono dispersive e frammentate, pertanto non esiste una reale architettura su cui operare un confronto adeguato con il sistema.

3. Architettura del Sistema proposto

3.1 Panoramica

La piattaforma "Comun-ity" è un'applicazione software che interagisce con gli utenti mediante un'interfaccia web e gestisce la persistenza dei dati mediante un database non relazionale orientato ai documenti.

Il sistema proposto è basato sullo stile architetturale Three-Tier combinato con un'implementazione lineare del design pattern MVC. Il sistema si presenta quindi diviso in tre componenti:

- **Model**: contiene i metodi di accesso ai dati.
- **View**: si occupa di visualizzare i dati all'Utente e gestisce l'interazione fra quest'ultimo e l'infrastruttura sottostante.
- **Controller**: riceve i comandi dell'Utente attraverso il View e reagisce eseguendo delle operazioni che possono interessare il Model e che portano generalmente ad un cambiamento di stato del View.

Questa scelta ci permette una migliore separazione dei concetti e organizzazione del codice, che comporta la semplificazione dei processi di modifica e manutenzione del codice. Comporta anche importanti vantaggi in termini di: scalabilità, riusabilità, flessibilità e testabilità.

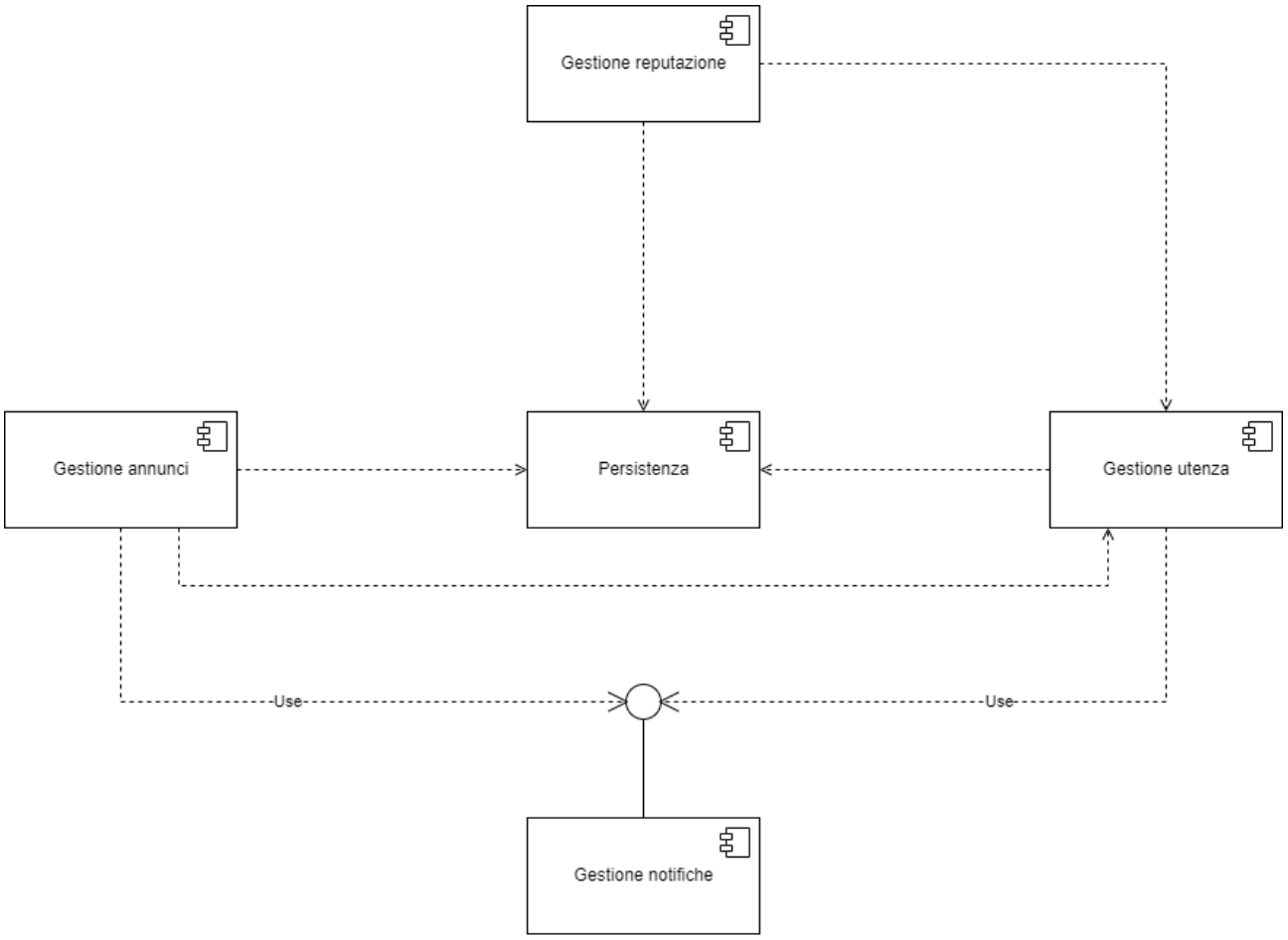
Ovviamente questo pattern architetturale può presentare anche degli svantaggi se non implementato correttamente, come una verbosità maggiore del codice e problemi di performance. Per questo la sua implementazione, che richiede una profonda conoscenza del pattern stesso, può essere più dispendiosa in termini di tempo rispetto ad altri pattern.

3.2 Decomposizione in sottosistemi

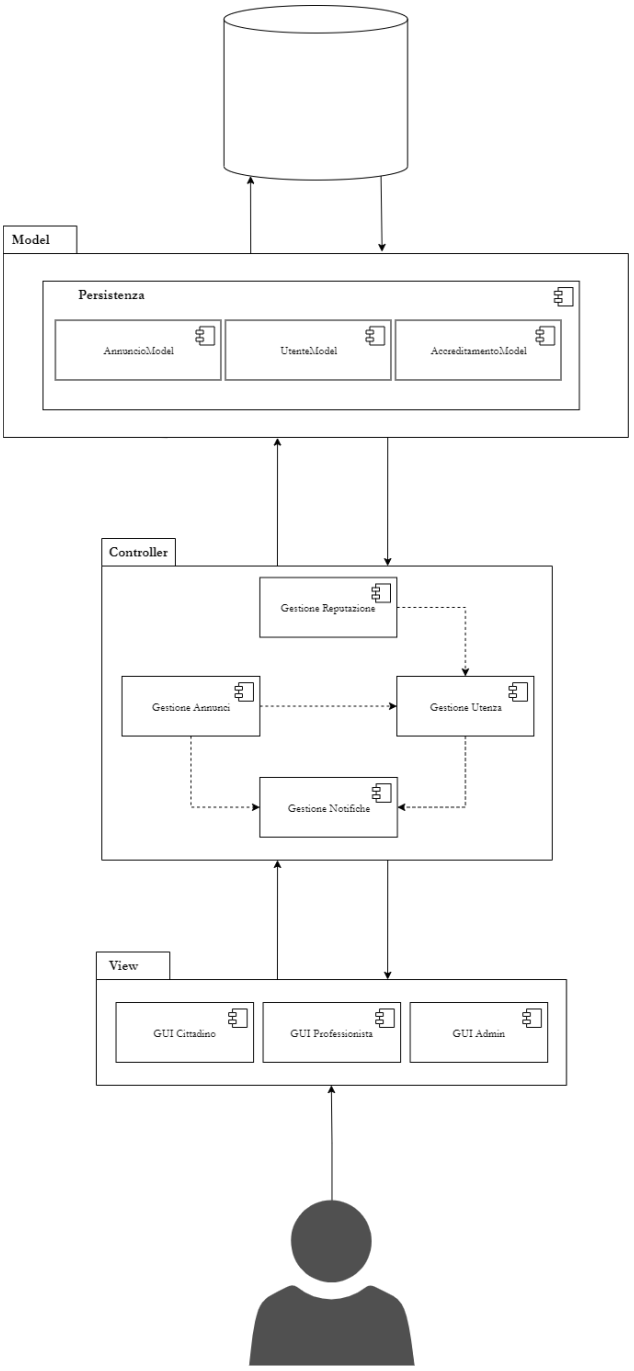
Il sistema è suddiviso in cinque sottosistemi principali:

- **Gestione utenza:** è responsabile della gestione delle funzioni di registrazione, login/logout, inserimento e modifica dei dati personali, delle richieste di accreditamento allo status di professionista e dei privilegi admin, come l'accreditamento di un professionista o le operazioni di moderazione dell'utenza;
- **Gestione reputazione:** si occupa dell'assegnazione di una valutazione agli utenti;
- **Gestione annunci:** è responsabile dell'inserimento, cancellazione, presa in carico ed elaborazione degli annunci;
- **Gestione notifiche:** si occupa del canale di comunicazione via email;
- **Persistenza:** si occupa della gestione della persistenza tramite l'ausilio di un database.

3.2.1 Diagramma di decomposizione



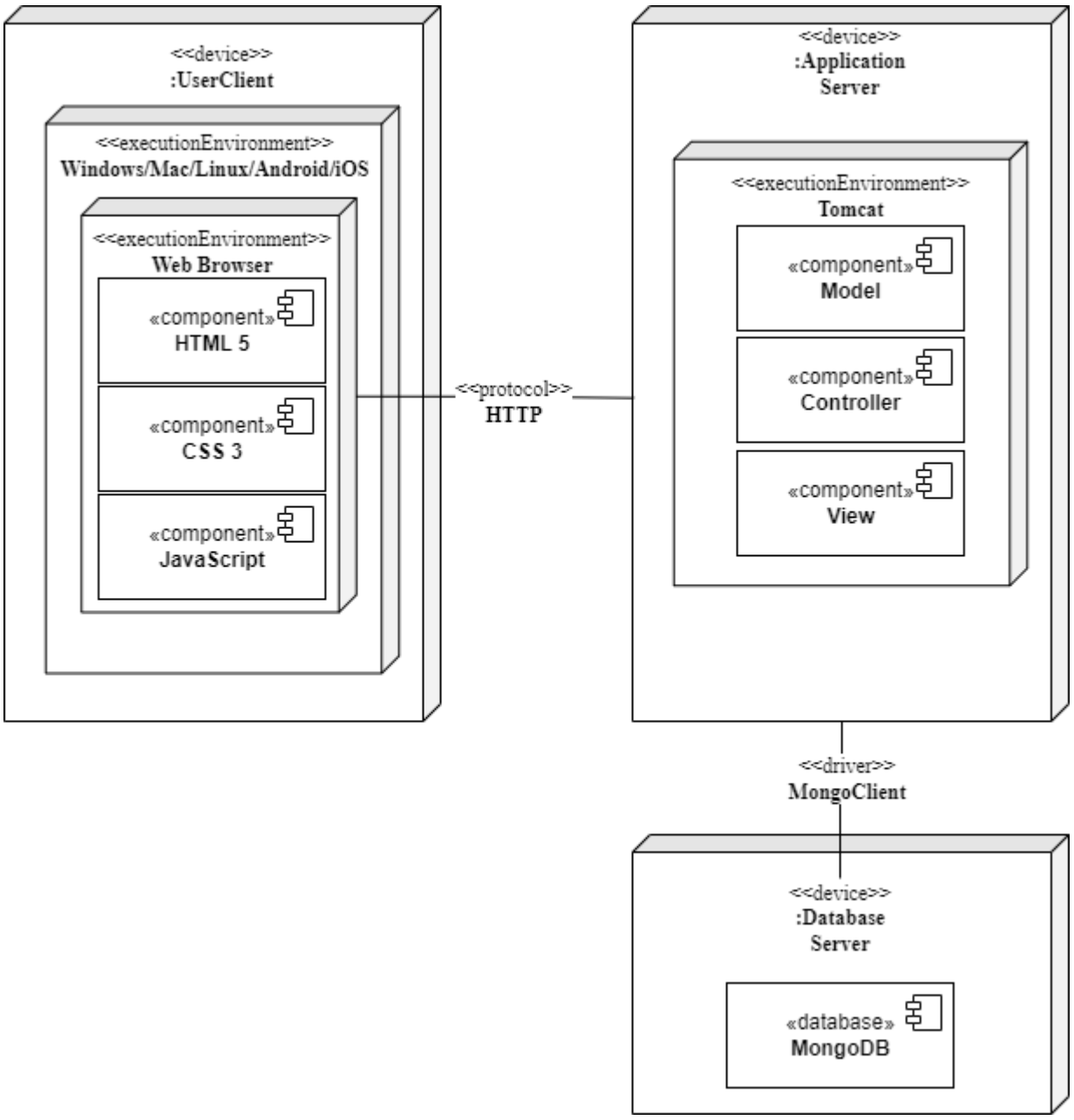
Di seguito una vista più dettagliata del sistema, dove i sottosistemi appena descritti vengono inquadrati all'interno del pattern architetturale scelto:



3.3 Mapping Hardware/Software

Il Sistema che si desidera sviluppare, utilizza un'architettura Client/Server, in cui un Server fornisce servizi a più Client. Su una macchina Client è eseguito un web browser che consente all'Utente di interagire con l'Application Server per inoltrare richieste e visualizzare le risposte ricevute.
L'Application Server, inoltre, gestisce la logica applicativa, mentre il Database Server gestisce i dati persistenti. La comunicazione tra Client e Server avviene tramite protocollo HTTP. Il Client inoltra una richiesta al Server, che verrà soddisfatta con la risposta di quest'ultimo.
Le specifiche hardware e software necessarie per il Client sono rispettivamente una macchina dotata di connessione a Internet e un sistema operativo con un web browser installato.
Le specifiche hardware necessarie per il Server consistono in una macchina connessa a Internet, la quale sia capace di immagazzinare grandi quantità di dati.
Per quanto riguarda le specifiche software necessarie, esse comprendono un Database Management System (MongoDB) per la gestione dei dati persistenti, un System per la gestione della logica applicativa e della comunicazione con più Client.

3.3.1 Deployment Diagram



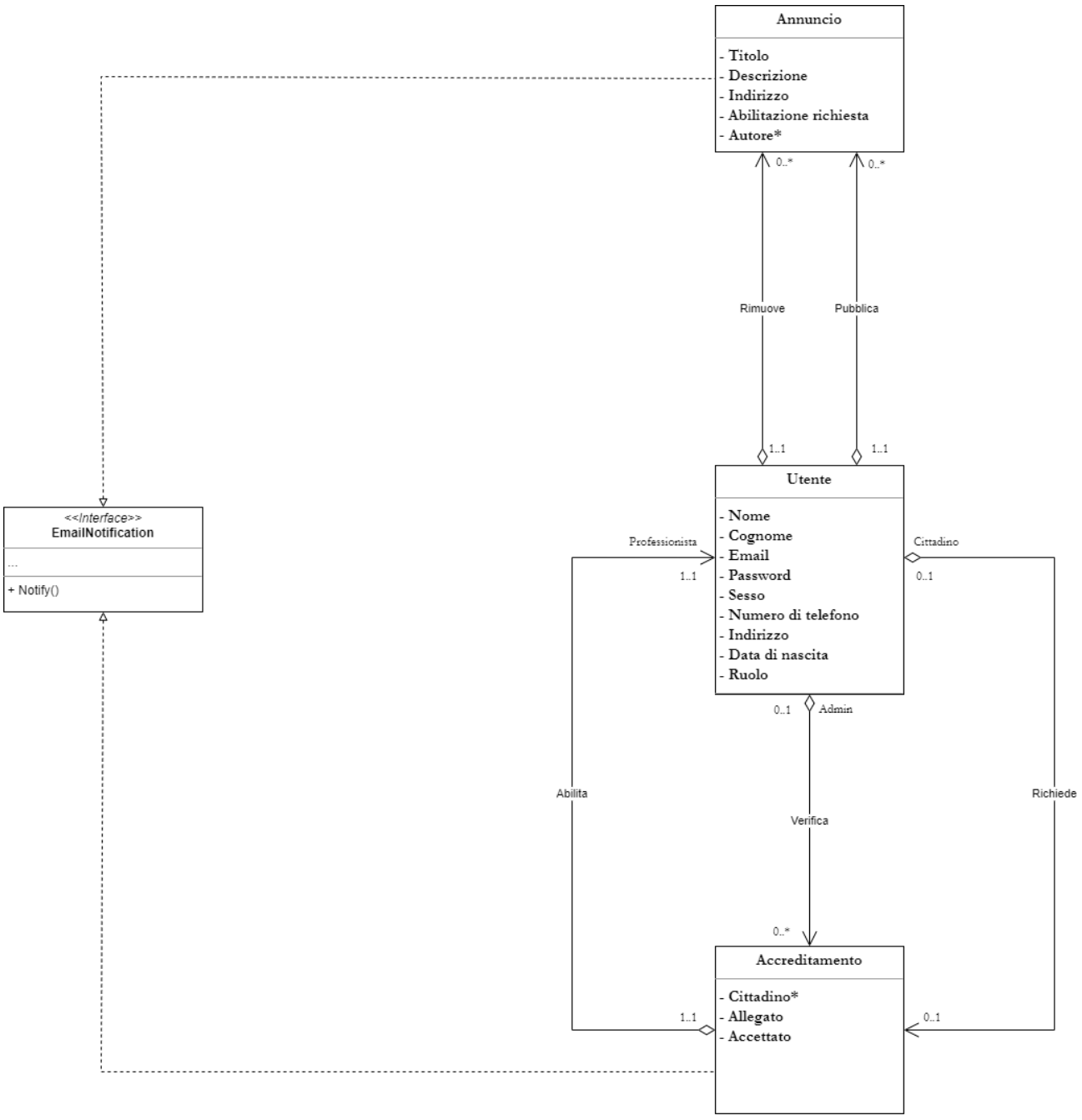
3.4 Gestione dei dati persistenti

Per la gestione dei dati persistenti la scelta è ricaduta su un database noSQL, ovvero un database non relazionale. Questa scelta deriva dalla flessibilità e dalla capacità di conservare e gestire più tipi di dati rispetto a un database relazionale tradizionale.

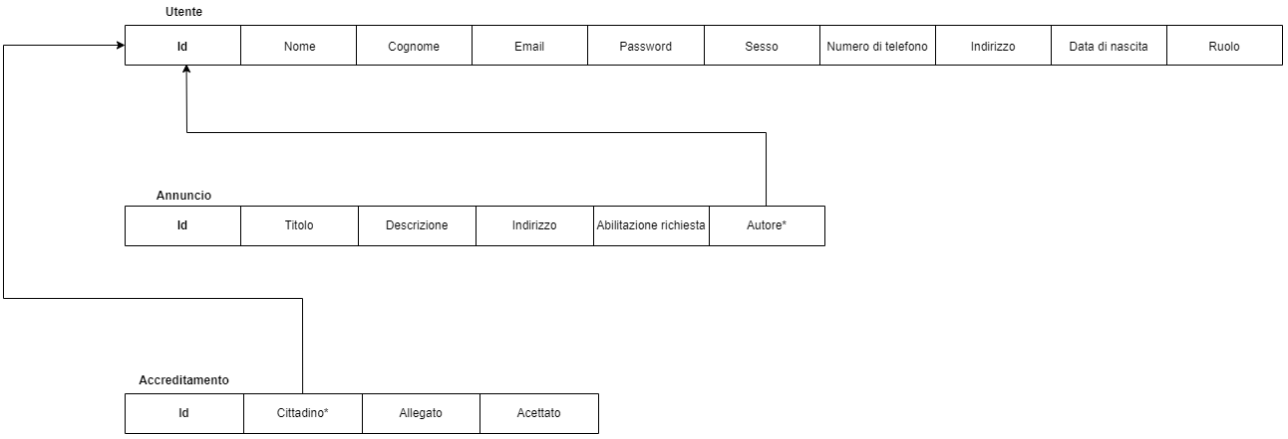
La ristrutturazione dell'entity class diagram è avvenuta secondo i seguenti criteri:

- Le entità "Admin", "Professionista" e "Cittadino" sono state accorpate all'entità "Utente" aggiungendo a quest'ultima l'attributo "Ruolo".
- Le entità "Commissione" e "Lavoro" sono state accorpate all'entità "Annuncio" aggiungendo a quest'ultima l'attributo "Abilitazione richiesta".

3.4.1 Entity Class Diagram ristrutturato



3.4.2 Schema logico



3.5 Controllo accessi e sicurezza

ATTORI / OGGETTI	Gestione Utenza	Gestione Reputazione	Gestione Annunci
Utente	• Registrazione • Login • Logout • Visualizzazione area personale • Inserimento dati personali • Modifica dati personali	• Assegnazione valutazione	• Inserimento • Cancellazione • Presa in carico
Admin	• Login • Logout • Visualizzazione area personale • Visualizzazione lista utenti • Accreditamento professionista • Moderazione utenza	NA	• Elaborazione

Professionista	• Registrazione • Login • Logout • Visualizzazione area personale • Inserimento dati personali • Modifica dati personali • Richiesta Accreditamento	• Assegnazione valutazione	• Inserimento • Rimozione • Presa in carico

3.6 Controllo flusso globale del software

Il controllo del flusso viene effettuato mediante delle richieste, le quali vengono generate da un client, e dalle classi preposte a gestire quel determinato evento associato alla richiesta.
Una volta ottenuto il risultato dell'operazione, la classe gestore si preoccupa di inoltrarlo al client che aveva generato la richiesta.
Il sistema software è gestito con l'uso di Servlet e JSP. Il Server centrale attende le richieste di un client (web browser) e una volta ricevuta una richiesta, la processa e la smista alla Servlet incaricata.

3.7 Condizioni limite

Nella seguente sezione vengono riportate le condizioni limite del sistema, come l'avvio, la terminazione e i fallimenti del sistema. In caso di malfunzionamento, il Sistema mostrerà un avviso di manutenzione, precludendo all'utente il normale utilizzo delle funzionalità del Sistema. Nel caso di un crash del sistema dovuto ad un errore, si effettuerà un ripristino al backup più recente.

3.7.1 Start Up

NOME SCENARIO	**SC_SU StartUpServer**
ATTORI	**FRANCESCO: ADMIN**

FLUSSO DI EVENTI	ATTORE	SISTEMA
	Francesco decide di avviare il Server usando l'apposito comando.	
		Il Server diventa operativo e attiva i servizi disponibili per gli utenti.

IDENTIFICATIVO	UC_SU	**DATA**	1/12/22
NOME	Avvio del Server	**VERSIONE**	0.1
		AUTORE	Michele Rabesco
DESCRIZIONE	Lo UC fornisce la funzionalità di avvio del Server.		
ATTORE PRINCIPALE	ADMIN: Vuole avviare il Server per rendere disponibile il Sistema.		
ATTORI SECONDARI	NA		
ENTRY CONDITION	Il Server è pronto all'avvio.		
EXIT CONDITION ON SUCCESS	Il Server è avviato e il Sistema rende disponibili le funzionalità agli utenti.		
EXIT CONDITION ON FAILURE	Il Server non è avviato.		
RILEVANZA/ USER PRIORITY	Elevata		
FREQUENZA STIMATA	1/mese		
EXTENSION POINT	NA		
GENERALIZATION OF	NA		
FLUSSO DI EVENTI PRINCIPALE/MAIN SCENARIO			
1	Admin:	Utilizza l'apposito comando per avviare il Server.	
2	Sistema:	Inizializza una connessione con il database.	

3	Sistema:	Verifica l'integrità dei dati persistenti salvati nel database.
4	Sistema:	Avvia il server e mostra all'amministratore la sua area riservata.
SCENARIO / FLUSSO DI EVENTI DI ERRORE: Connessione assente		
2.a1	Sistema:	Il sistema mostra all'amministratore un messaggio di errore: non è stato possibile avviare il Server per via della mancanza di connessione.
SCENARIO / FLUSSO DI EVENTI ALTERNATIVI: Dati non validi		
3.b1	Sistema:	Notifica l'amministratore di errori sull'integrità dei dati persistenti nel database e non procede all'avvio del server.
3.b2	Admin:	Accede al database e corregge gli errori sui dati persistenti.
3.b3	Sistema:	Riesegue il primo step.
NOTE		NA
SPECIAL REQUIREMENTS		NA

3.7.2 Shut down

NOME SCENARIO	**SC_SD ShutDownServer**	
ATTORI	**FRANCESCO: ADMIN**	
FLUSSO DI EVENTI	ATTORE	SISTEMA
	Francesco decide di arrestare il Server usando l'apposito comando.	
		Il Server avvisa gli utenti dell'imminente arresto, dunque procede allo spegnimento..

IDENTIFICATIVO	UC_SD	**DATA**	1/12/22

NOME	Arresto del Server	**VERSIONE**	0.1
		AUTORE	Michele Rabesco
DESCRIZIONE	Lo UC fornisce la funzionalità di arresto del Server.		
ATTORE PRINCIPALE	ADMIN: Vuole arrestare il Server per spegnere il Sistema.		
ATTORI SECONDARI	NA		
ENTRY CONDITION	Il Server è avviato.		
EXIT CONDITION ON SUCCESS	Il Server viene arrestato.		
EXIT CONDITION ON FAILURE	Il Server non viene arrestato.		
RILEVANZA/ USER PRIORITY	Elevata		
FREQUENZA STIMATA	1/mese		
EXTENSION POINT	NA		
GENERALIZATION OF	NA		

FLUSSO DI EVENTI PRINCIPALE/MAIN SCENARIO		
1	Admin:	Utilizza l'apposito comando per arrestare il Server.
2	Sistema:	Effettua il salvataggio dei dati persistenti.
3	Sistema:	Chiude le connessioni attive ed arresta il Server.

SCENARIO / FLUSSO DI EVENTI DI ERRORE: Dati non salvati		
2.a1	Sistema:	Il sistema mostra all'amministratore un messaggio di errore: il salvataggio dei dati persistenti non è andato a buon fine, quindi non procede all'arresto.
NOTE		NA

SPECIAL REQUIREMENTS	NA

3.7.3 Failures

IDENTIFICATIVO	UC_FA	**DATA**	1/12/22
NOME	Errore di accesso ai dati persistenti.	**VERSIONE**	0.1
		AUTORE	Michele Rabesco
DESCRIZIONE	Lo UC descrive il comportamento del sistema al verificarsi di un errore di accesso ai dati persistenti.		
ATTORE PRINCIPALE	ADMIN: Vuole risolvere la condizione di fallimento.		
ATTORI SECONDARI	NA		
ENTRY CONDITION	Il Server riscontra un errore nell'accesso ai dati persistenti e notifica l'admin.		
EXIT CONDITION ON SUCCESS	Il Server riprende il corretto funzionamento.		
EXIT CONDITION ON FAILURE	Il Server non riprende il corretto funzionamento.		
RILEVANZA/ USER PRIORITY	Elevata		
FREQUENZA STIMATA	1/mese		
EXTENSION POINT	NA		
GENERALIZATION OF	NA		

FLUSSO DI EVENTI PRINCIPALE/MAIN SCENARIO		
1	Admin:	Riceve la notifica di errore dal sistema.
2	Admin:	Arresta forzatamente il sistema tramite l'apposito comando.
3	Admin:	Risolve l'errore sui dati persistenti.

4	Admin:	cfr. (UC_SU): Il sistema viene riavviato.
NOTE		NA
SPECIAL REQUIREMENTS		NA

3.7.4 Gestione dei fallimenti

- Nel caso in cui il Server riscontri un errore sui dati persistenti, è previsto un ripristino dell'ultimo backup automatico dei dati persistenti. Grazie al backup automatico il ripristino dei dati è garantito.
- Nel caso in cui il Server riscontri un errore a livello hardware, il sistema deve essere per forza arrestato e l'amministratore deve provvedere manualmente a risolvere il problema.
- Nel caso in cui il Server riscontri un'improvvisa interruzione dell'alimentazione, viene attivato un "UPS - Uninterruptible Power Supply", il quale permette solo di effettuare un backup di emergenza, procedendo poi all'arresto.

4. Servizi dei sottosistemi

4.1 Gestione Utenza

Servizio	Descrizione	Interfaccia
Registrazione	Questa funzionalità permette ad un cittadino/professionista di registrarsi al sistema.	GestioneUtenza
Login	Questa funzionalità permette di effettuare l'accesso al sistema tramite le proprie credenziali per sfruttare tutte le funzionalità offerte.	GestioneUtenza
Logout	Questa funzionalità permette di effettuare l'uscita dal sistema.	GestioneUtenza
Inserimento dati personali	Questa funzionalità permette di far inserire ulteriori dati personali da utenti che usano il sistema.	GestioneUtenza
Modifica dati personali	Questa funzionalità permette di cambiare le impostazioni di un cittadino/professionista all'interno del sistema.	GestioneUtenza
Richiesta accreditamento	Questa funzionalità permette a un cittadino di richiedere che la sua professionalità venga riconosciuta, previa sottomissione della documentazione necessaria.	GestioneUtenza
Accreditamento professionista	Questa funzionalità permette ad un admin di accreditare come "professionista" un cittadino che lo richiede..	GestioneUtenza
Moderazione utenza	Questa funzionalità permette ad un admin di eliminare definitivamente o temporaneamente l'account di un utente dal sistema.	GestioneUtenza

4.2 Gestione Reputazione

Servizio	Descrizione	Interfaccia
Assegnazione	Questa funzionalità permette ad un cittadino di assegnare una valutazione a chi ha effettuato la commissione.	GestioneReputazione

4.3 Gestione Annunci

Servizio	Descrizione	Interfaccia

Inserimento	Questa funzionalità permette ad un utente di inserire un nuovo annuncio.	GestioneAnnunci
Cancellazione	Questa funzionalità permette ad un utente di cancellare un annuncio se inserito precedentemente nel sistema da lui stesso e non ancora accettato da terzi.	GestioneAnnunci
Presa in Carico	Questa funzionalità permette ad un utente di prendere in carico una commissione/lavoro.	GestioneAnnunci
Elaborazione	Questa funzionalità permette ad un admin di operare azioni su un annuncio.	GestioneAnnunci

4.4 Gestione Notifiche

Servizio	Descrizione	Interfaccia
Invio	Questa funzionalità permette al sistema di notificare gli utenti via email.	GestioneNotifiche

4.5 Gestione Persistenza

Servizio	Descrizione	Interfaccia
Salvataggio	Questa funzionalità permette al sistema di salvare i dati necessari per il funzionamento, all'interno del database.	GestionePersistenza
Eliminazione	Questa funzionalità permette al sistema di eliminare i dati all'interno del database.	GestionePersistenza
Modifica	Questa funzionalità permette al sistema di modificare i dati all'interno del database.	GestionePersistenza
Ricerca	Questa funzionalità permette al sistema di ricercare dati all'interno del database.	GestionePersistenza

5. Glossario

TERMINE	SIGNIFICATO
System Design Document	Documento che definisce gli obiettivi di progettazione e di decomposizione del sistema.
Design Goal	Obiettivi di design progettati per il sistema proposto.
Trade-off	Scelte e compromessi tra design goals dissonanti.
Sottosistema	Un sottoinsieme dei servizi del dominio applicativo, formato da servizi legati da una relazione funzionale.
Front-end	La parte visibile all'utente di un programma e con cui egli può interagire.
Back-end	La parte che si occupa di gestire il funzionamento del sistema a seguito delle interazioni da parte del cliente nel Front-end, include anche la gestione dei dati persistenti nel Database e del sistema.
Database	Architettura esterna che si occupa della gestione e della memorizzazione dei dati persistenti.
Persistenza	Caratteristica dei dati di un programma di sopravvivere all'esecuzione del programma stesso che li ha creati, salvando i dati in uno storage non volatile
Framework	Un'architettura logica di supporto che fornisce strumenti per facilitare e velocizzare il lavoro di programmazione.
Web Browser	Applicazione software installata sul client che permette di visualizzare e navigare. le risorse del web.
Client	Componente che accede a servizi e risorse di un altro componente detto Server.
Server	Componente che gestisce traffico di informazioni e fornisce servizi e risorse attraverso la rete.
Model	Contiene i metodi di accesso ai dati.
View	Si occupa di visualizzare i dati all'Utente e gestisce l'interazione fra quest'ultimo e l'infrastruttura sottostante.
Controller	Riceve i comandi dell'Utente attraverso il View e reagisce eseguendo delle operazioni che possono interessare il Model e che portano generalmente ad un cambiamento di stato del View.
Application Server	Una tipologia di server che fornisce l'infrastruttura e le funzionalità logiche di supporto, sviluppo ed esecuzione di applicazioni.

DataBase Management System	Sistema software per creazione, manipolazione e interrogazione efficiente di database.
HyperText Transfer Protocol	Un protocollo di trasferimento di ipertesti che consente a due macchine, Client e Server, di interagire attraverso un meccanismo di richiesta/risposta.
Servlet	Oggetti Java all'interno del server web che permettono di creare web applications in combinazione con JSP.
Java Server Page	Tecnologia di programmazione web utilizzata per fornire contenuti dinamici.
Startup	Avvio di sistema.
Shutdown	Spegnimento di sistema.
Backup	La messa in sicurezza dei dati del sistema attraverso la creazione di ridondanza dei dati stessi.