TutoratoSmart

SDD System Design Document

TutoratoSmart

Riferimento	
Versione	0.7
Data	31/01/2020
Destinatario	Prof.ssa F. Ferrucci
Presentato da	Marco Delle Cave, Francesco Pagano, Manuel Pisciotta, Alessia Olivieri
Approvato da	

Revision History

Data	Versione	Cambiamenti	Autori
27/11/2019	0.1	Aggiunta introduzione (Obiettivi del sistema, design goals, definizioni, acronimi, abbreviazioni, riferimenti)	Delle Cave Marco, Pagano Francesco
28/11/2019	0.2	Aggiunta architettura del sistema corrente, sistema proposto, decomposizione in sottosistemi, mapping hardware-software	Delle Cave Marco, Pagano Francesco
3/12/2019	0.3	Aggiunta gestione dati persistenti, controllo degli accessi e sicurezza, controllo globale del software, condizioni limiti, servizi dei sottosistemi	Delle Cave Marco, Pagano Francesco
4/12/2019	0.4	Revisione schema ER, architettura sistema proposto	Delle Cave Marco, Pagano Francesco
6/12/2019	0.4	Revisione SDD	Pisciotta Manuel
16/12/2019	0.5	Revisione SDD	Pagano Francesco, Pisciotta Manuel
18/12/2019	0.6	Rettifica SDD	Pisciotta Manuel
31/01/2020	0.7	Rettifica SDD	Pisciotta Manuel

Sommario

1. Introduzione

1.1 Obiettivi del sistema

Il progetto nasce per fornire uno strumento di supporto agli studenti e ai tutor del Dipartimento di Psicologia dell'Università degli studi della Campania "Luigi Vanvitelli", e alla Commissione Tutorato per consentire un processo rapido, senza perdita di informazioni ed efficiente. Attualmente la prenotazione di un appuntamento allo sportello di tutorato risulta molto lenta e poco pratica in quanto avviene tramite scambio di email tra studente e tutor. Invece, per quanto riguarda l'acquisizione e la consegna finale dei registri dell'attività di tutorato, è necessario rivolgersi fisicamente alla Segreteria di Dipartimento e, per tutta la durata del contratto, ogni tutor deve annotare sul proprio registro cartaceo le attività svolte, attività che successivamente verranno valutate e convalidate dalla Commissione di Tutorato. Questo crea molti disagi e rallentamenti nelle pratiche.

Il sistema progettato sarà un'applicazione Web che dovrà consentire l'accesso a 3 tipologie di utenza (studente del Dipartimento di Psicologia, Tutor e Membro della Commissione di Tutorato), ciascuna con funzionalità a disposizione previste nella fase di analisi e raccolta dei requisiti.
Al fine di facilitare l'iter burocratico sopra descritto, gli obiettivi primari del sistema sono quindi quelli di:

1. Fornire uno strumento per supportare la prenotazione allo sportello di tutorato e la gestione delle attività dei tutor stessi;
2. Migliorare lo scambio di informazioni tra i tutor e gli altri stakeholder coinvolti;
3. Ottimizzare i tempi relativi alla prenotazione allo sportello di tutorato e al riconoscimento delle ore lavorative svolte;
4. Eliminare gli attuali disagi agli stakeholder.

1.2 Design Goals e Trade-off

1.2.1 Design Goals

I design goal identificati per il sistema TutoratoSmart sono i seguenti:

- Criteri di performance
  - Tempo di risposta

    -Per l'accesso all'area utente il tempo di risposta è di 2 secondi (RNF_P1).

    -Per la visualizzazione dei form di richiesta il tempo di risposta è di 1 - secondo (RNF_P1).

    -Per la generazione del file pdf il tempo di risposta è di 3 secondi (RNF_P1).

    -Per la visualizzazione delle liste il tempo di risposta è di 3 secondi (RNF_P1).

    -Per la modifica di appuntamento, il tempo di risposta è di 1 secondo (RNF_P1).
  - Memoria

    -Il sistema dovrà mantenere i dati consistenti (RNF_A2).

    -La dimensione complessiva del sistema dipende dalla memoria utilizzata per il mantenimento del database (Dominio applicativo).

- Criteri di affidabilità
  - Robustezza

    -Eventuali input non validi immessi dall'utente saranno opportunamente segnalati attraverso messaggi di errore (RNF_A1).
  - Affidabilità

    -Il sistema deve garantire l'affidabilità dei servizi proposti. Il prodotto software sarà sviluppato in modo tale da controllarne il corretto funzionamento tramite diverse tipologie di testing, black-box e white-box (Documenti di management).
  - Disponibilità

    -Una volta realizzato il sistema, sarà disponibile ogni qualvolta gli studenti, i tutor e la commissione tutorato ne richiederanno l'utilizzo (RNF_U4).
  - Security

    -L'accesso al sistema è garantito mediante una email e una password (che verrà criptata) (RNF_A4). Inoltre, la sicurezza è garantita in quanto verranno implementati filtri di autenticazione per accedere alle aree riservate (RNF_A3).

- Criteri di costo
  - Costi di sviluppo

    -È stimato un costo complessivo di 200 ore per la progettazione e lo sviluppo del sistema (50 ore per ogni team member) (Documenti di management).

- Criteri di manutenzione
  - Estendibilità

    -Il sistema sarà facilmente estendibile in quanto sarà possibile aggiungere nuove funzionalità ad esso senza andare a modificare l'intera struttura, rispettando i pattern architetturali adottati (RNF_S3).
  - Leggibilità

    - In concordanza con gli obiettivi di estendibilità e modificabilità il codice del sistema sarà prodotto per avere un alto grado di leggibilità (RNF_S4).
  - Adattabilità

    -Il sistema può funzionare solo in ambito universitario, ma è adattabile a più Università (Dominio applicativo).
  - Tracciabilità dei requisiti

    -La tracciabilità dei requisiti è possibile grazie ad una matrice di tracciabilità, attraverso la quale è possibile retrocedere al requisito associato ad ogni parte del progetto. La tracciabilità è garantita dalla fase di progettazione fino al testing (Documenti di management).
  - Portabilità

    -Il sistema sarà portabile in quanto l'interazione avviene mediante un browser senza interazione con il sistema sottostante, c'è quindi indipendenza dal sistema operativo (RNF_I1).

- Criteri utenti finali

- Usabilità

  -Il sistema sarà di facile comprensione e utilizzo, sarà infatti molto semplice da apprendere senza la consultazione di documentazione associata (RNF_U1, RNF_U3). L'intuitività è garantita in quanto il sistema avrà una buona prevedibilità, cioè la risposta del sistema ad un'azione utente sarà corrispondente alle aspettative (RNF_U2, RNF_IN1, RNF_IN2).

- Utilità

  -Il lavoro dell'utente verrà supportato nel miglior modo possibile dal sistema, infatti l'utente compirà le operazioni richieste senza il carico di lavoro che deriva dal realizzarle diversamente (Dominio applicativo).

1.2.2 Trade-off

1.1.1 Comprensibilità vs costi

Si preferisce aggiungere costi per la documentazione al fine di rendere il codice comprensibile anche alle persone non coinvolte nel progetto o le persone coinvolte che non hanno lavorato a quella sezione di progetto. Commenti diffusi nel codice facilitano la comprensione, di conseguenza migliorare la comprensibilità agevola il mantenimento e anche il processo di modifica.

1.1.2 Tempo di risposta vs Affidabilità

Il sistema sarà implementato in modo tale da preferire l'affidabilità al tempo di risposta, in modo tale da garantire un controllo più accurato dei dati in input a discapito del tempo di risposta del sistema.

1.1.3 Manutenibilità vs efficienza

Si preferisce avere una maggiore manutenibilità del sistema facendo in modo che ogni sottosistema non acceda direttamente al DB, ma che l'accesso ai dati venga gestito da un sottosistema intermedio a discapito dell'efficienza e delle prestazioni generali.

1.3 Definizioni, acronimi e abbreviazioni

Definizioni

- **Deployment Diagram**: Diagramma UML di specifica per le relazioni tra le componenti realizzate (con relative tecnologie implementative) e le risorse Hardware e Software necessarie al corretto funzionamento del sistema;
- **Design Goal**: Obbiettivi di design progettati per il sistema proposto;
- **Design Trade-off**: Scelte e compromessi tra design goals dissonanti;
- **Greenfield Engineering:** Tipologia di sviluppo che comincia da zero, non esiste nessun sistema a priori e i requisiti sono ottenuti dall'utente finale e dal cliente. Nasce, perciò, a partire dai bisogni dell'utente.
- **Pattern MVC**: Modello architetturale del sistema a tre livelli;
- **Model**: Layer del pattern architetturale per la gestione e memorizzazione dei dati persistenti;
- **View**: Layer del pattern architetturale per la gestione e controllo d'interfaccia tra le risorse del sistema e l'utente finale;
- **Controller**: Layer del pattern architetturale per la gestione e il controller della logica di business.

Acronimi

- **DB**: Database.
- **DBMS**: Database Management System
- **MVC**: Model View Controller.
- **RAD**: Requirements Analysis Document.
- **SDD**: System Design Document.
- **TS**: TutoratoSmart.

1.4 Riferimenti

- Ian Sommerville, Software Engineering, Addison Wesely.
- TS_RAD_V_1.1

1.5 Panoramica

- Capitolo 1: contiene l'introduzione con l'obiettivo del sistema, i design goals e un elenco di definizioni, acronimi e abbreviazioni utili alla comprensione dell'intera documentazione.
- Capitolo 2: descrive le funzionalità offerte dal sistema corrente.
- Capitolo 3: presentata l'architettura del sistema proposto, in particolare la decomposizione in sottosistemi, il mapping hardware/software, i dati persistenti, il controllo degli accessi e sicurezza, il controllo del flusso globale del sistema, le condizioni limite.
- Capitolo 4: presenta i servizi dei sottosistemi.

2. Architettura del Sistema corrente

Attualmente non esiste un sistema software che si occupa di gestire questa problematica, ossia la gestione degli appuntamenti di tutorato e la validazione del registro di tutorato. Gli appuntamenti attualmente vengono gestiti tramite scambi di email tra studenti e tutor, ciò risulta essere molto macchinoso e alquanto lento poiché i tempi necessari allo scambio di email e il recupero dei dati da parte dello studente non sono agevoli. Per quanto riguarda il registro di tutorato, esso è cartaceo e viene compilato manualmente dai tutor, per poi essere revisionato e convalidato periodicamente dalla Commissione di Tutorato. Quindi si tratta di un sistema che rientra nel campo della Greenfield Engineering.

3. Architettura del Sistema proposto

3.1 Panoramica

Il sistema da noi proposto è un'applicazione web con lo scopo di offrire un supporto alle attività di tutorato. L'obiettivo che si pone è quello di fornire uno strumento di gestione sia per quanto riguarda le richieste di tutorato, sia per la compilazione e convalida dei registri di tutorato. Metterà a disposizione dello studente le interfacce per registrarsi alla piattaforma, per prenotare un nuovo appuntamento e per modificarlo; mentre invece il tutor potrà accettare gli appuntamenti, confermare questi ultimi inserendo ulteriori dettagli e compilare il registro del tutorato inserendo le attività lavorative svolte, sempre attraverso delle interfacce grafiche. Verrà utilizzata un'architettura di tipo MVC, dove il model fornirà le operazioni per accedere ai dati utili all'applicazione, ed implementerà quindi la struttura dati centrale; il controller gestirà il control flow, ovvero ottiene gli input dall'utente tramite la view e manda messaggi al model; la view visualizzerà il model e verrà notificato ogni volta che il model sarà modificato. I gestori saranno individuati in

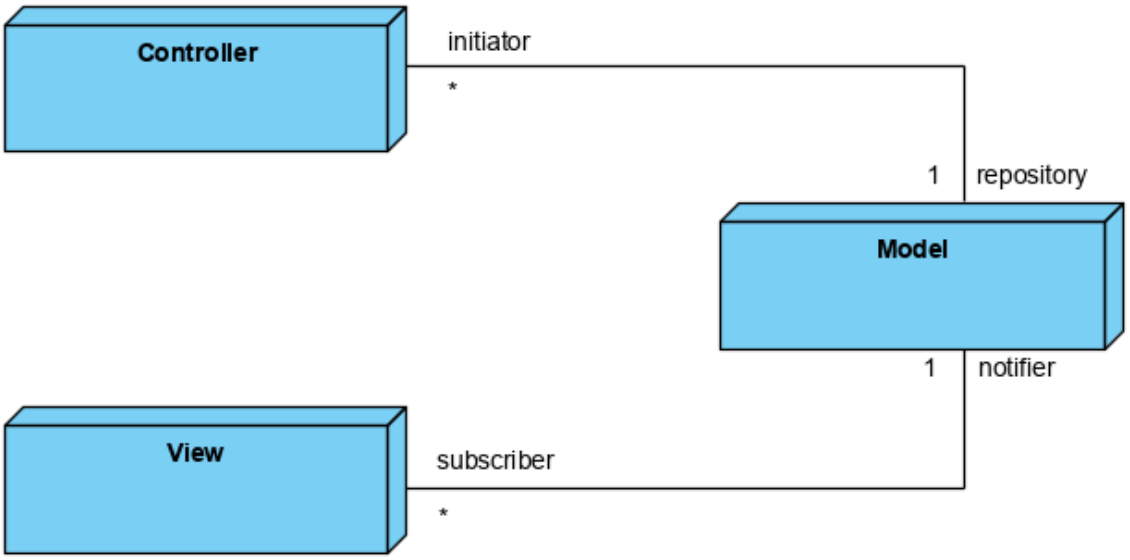
base alle funzionalità per poter rendere massima la coesione e minimo l'accoppiamento tra i sottosistemi in modo che i cambiamenti in un sottosistema non influiscano sugli altri.

3.2 Decomposizione in sottosistemi

3.2.1 Decomposizione in Layer

La decomposizione prevista per il sistema è composta da tre layer che si occupano di gestirne aspetti e funzionalità differenti:

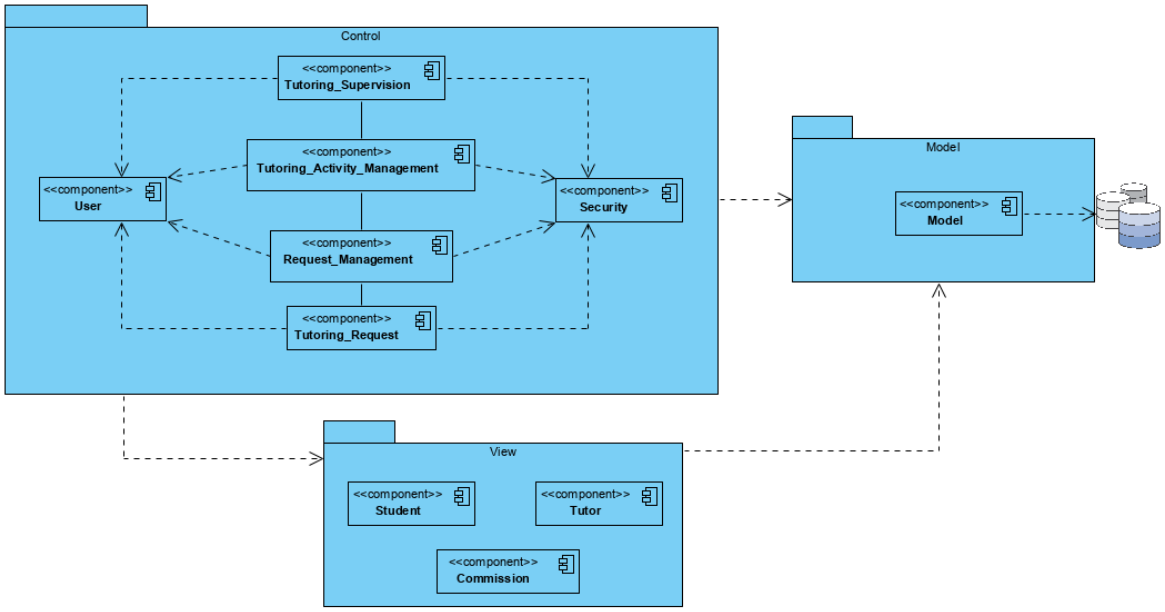
- **View**: raccoglie e gestisce elementi di interfaccia grafica e gli eventi generati su di essi;
- **Controller**: si occupa della gestione della logica del sistema;
- **Model**: si occupa della gestione e dello scambio dei dati tra i sottosistemi;



3.2.2 Decomposizione in Sottosistemi

Dopo un'analisi funzionale dettagliata, si è deciso di gestire i singoli componenti con basso accoppiamento ed elevata coesione in modo tale da garantire, in caso di successive modifiche il minor numero di aggiornamenti da apportare tra tutti i sottosistemi. Abbiamo inoltre deciso di suddividere le funzionalità per area di gestione (studente, tutor e commissione tutorato) e creato un'interfaccia intermedia storage tra i sistemi della logica di business e il database; l'assunzione dietro questa scelta di design è che lo storage ha una interfaccia più stabile rispetto al database e quindi nel caso in cui cambi l'interfaccia del sottosistema database, solo il sottosistema storage dovrà essere modificato.

Il sistema si compone di dieci componenti che si occupano di gestirne aspetti e funzionalità differenti:



Il livello View prevede la gestione di 3 sottosistemi:

- Student: gestisce l'interfaccia grafica e gli eventi generati dall'interazione con il sistema da parte degli studenti.
- Tutor: gestisce l'interfaccia grafica e gli eventi generati dall'interazione con il sistema da parte dei tutor.
- Commission: gestisce l'interfaccia grafica e gli eventi generati dall'interazione con il sistema da parte dei membri della Commissione di Tutorato.

Il livello Control prevede la gestione di 6 sottosistemi:

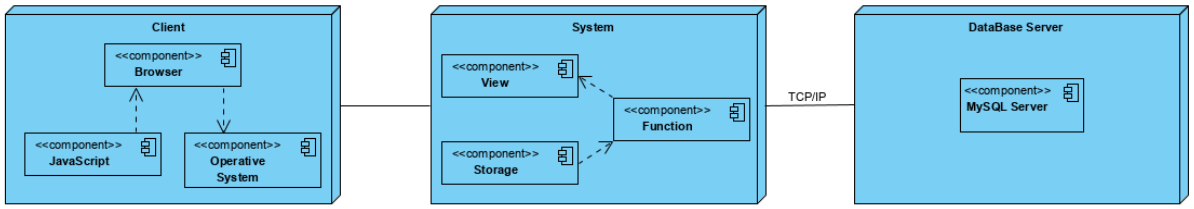
- Request_Management: si occupa di visualizzare le richieste di appuntamento con i relativi dettagli, di confermare l'avvenuto appuntamento con uno studente, di modificare i dati relativi agli appuntamenti registrati e visualizzare il calendario indicante gli appuntamenti in agenda;
- Security: è un sottosistema che contiene i filtri di autenticazione e di accesso alle aree riservate;
- Tutoring_Activity_Management: consente di visualizzare il registro personale di tutorato, inserire un'attività svolta, di modificare i dati relativi ad un'attività non ancora convalidata e di generare un documento contenente il registro personale;
- Tutoring_Request: si occupa di gestire le funzionalità relative allo studente, in particolar modo la compilazione, modifica e cancellazione delle richieste di appuntamento, la visualizzazione dello storico richieste di appuntamento, i dettagli di una richiesta, il calendario indicante gli orari e impegni dello sportello;
- Tutoring_Supervision: è un sottosistema che permette di valutare e convalidare le attività svolte dai tutor e di registrare un nuovo tutor sulla piattaforma;
- User: permette agli utenti di registrarsi alla piattaforma ed effettuare il login e logout da essa.

Il livello Model prevede la gestione di un unico sottosistema:

- Model: sistema che gestisce ed immagazzina i dati persistenti.

3.2.3 Diagramma di Deployment

L'utente (Client) richiede le funzionalità tramite l'interfaccia che il sistema mette a disposizione a patto che si possieda un browser capace di interpretare javascript, in modo che le funzioni definite dal sistema possano eseguire in maniera corretta. Il tier del Client connette lo strato di view del System sul quale vengono eseguite le funzioni apposite al completamento degli obiettivi del Client. La parte Server racchiude e gestisce la persistenza dei dati. L'intera architettura non richiede ausilio di componenti hardware/software esterni.



3.3 Mapping hardware/software

Il sistema che si desidera sviluppare utilizzerà una struttura hardware costituita da un Server che risponderà ai servizi richiesti dai client. Il Client è una qualsiasi macchina attraverso il quale un utente può collegarsi, utilizzando una connessione internet, per accedere al sistema mentre la macchina server gestisce la logica e i dati persistenti contenuti nel database. Il client e il server saranno connessi tramite il protocollo HTTP, con il quale il client inoltra delle richieste al server e quest'ultimo provvederà a fornire i servizi richiesti.

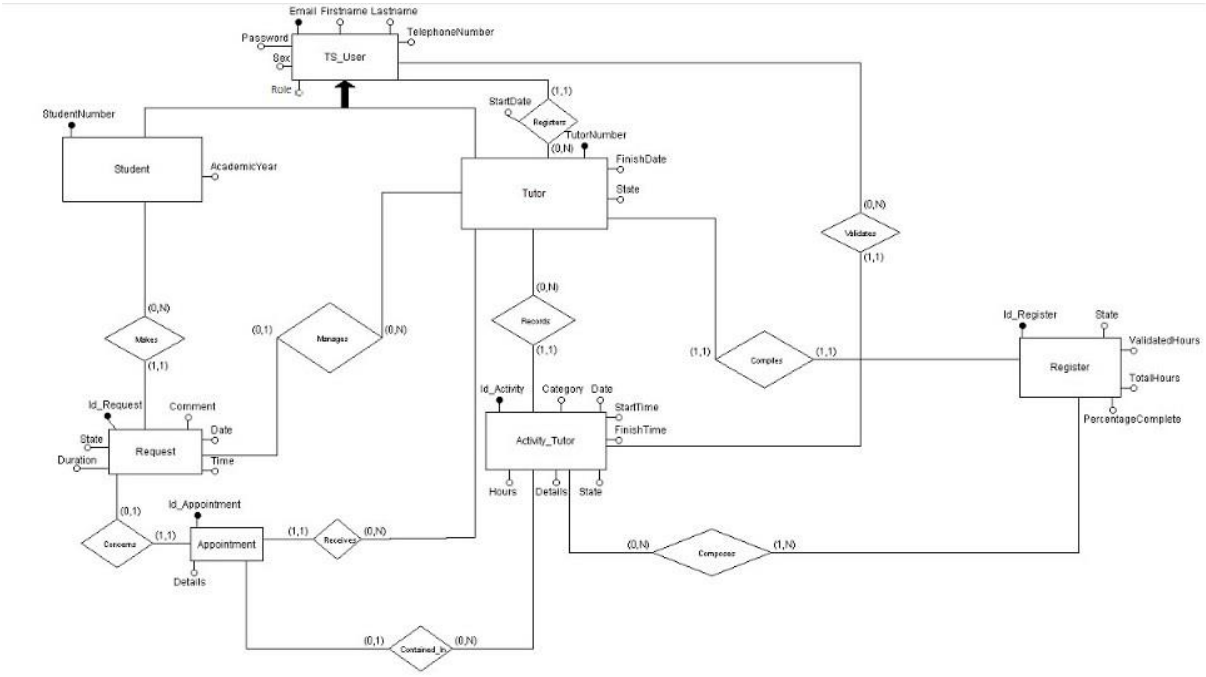
Le componenti hardware e software necessarie per il client sono un computer dotato di connessione internet e di un web browser installato su di esso.

Per il server, invece, c'è necessità di una macchina con connessione ad Internet e con la capacità di immagazzinare una grande quantità di dati. La componente software necessaria è dunque un DBMS, per consentire la comunicazione con più client.

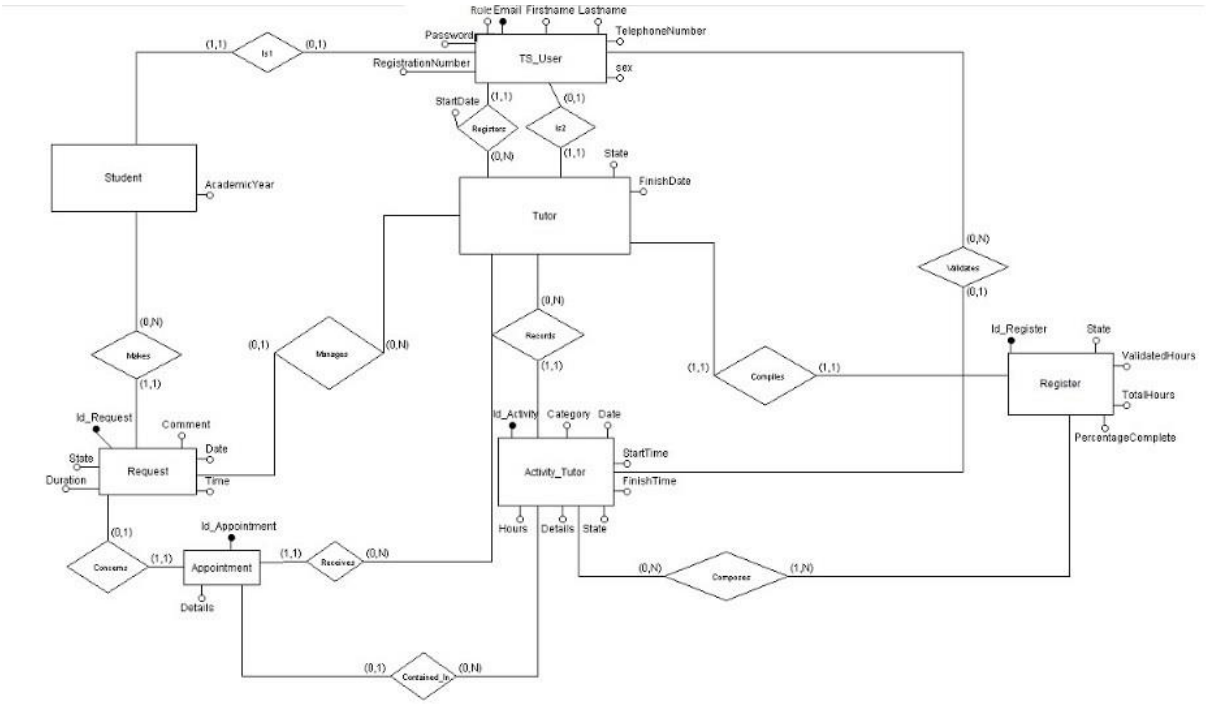
3.4 Gestione dati persistenti

Per la memorizzazione dei dati si è scelto un Database relazionale che consente un accesso efficiente ai dati, brevi tempi di risposta e un ampio spazio di archiviazione. Inoltre, è garantito l'accesso concorrente ai dati affidabili, ovvero ne viene salvata una copia ed è possibile ripristinare lo stato del database in caso di danni software o hardware. Infine, i dati sono privatizzati, cioè il DBMS ne consente un accesso protetto, quindi utenti diversi con operazioni diverse possono accedere a diverse sezioni del database.

Schema ER



Schema ER ristrutturato



Mapping ER-relazionale

User (**Email**, Role, Password, FirstName, LastName, TelephoneNumber, Sex, RegistrationNumber).
Student (**User.Email** ↑, AcademicYear).
Tutor (**User.Email** ↑, State, StartDate,**User.Email** ↑, **Register.IdRegister** ↑).
Request (**IdRequest**, Comment, Date, Duration, State, Time, **Student.Email** ↑).
Appointment (**IdAppointment**, Details, **Request.IdRequest** ↑, **Tutor.Email** ↑).
Activity_Tutor (**IdActivity**, Category, Date, StartTime, FinishTime, State, Hours, Details, **Tutor.Email** ↑, **Register.IdRegister** ↑).
Register (**IdRegister**, State, ValidatedHours, TotalHours, PercentageComplete).
Manages (**Tutor.Email** ↑, **Request.IdRequest** ↑).
Contained_In (**Appointment.IdAppointment** ↑, **Activity_Tutor.IdActivity** ↑).
Validates (**User.email**↑, **Activity_Tutor.IdActivity** ↑).

User

Nome	Tipo	Null	Key
Email	Varchar(45)	Not null	Primary key
Role	TinyInt(1)	Not null	
Password	Varchar(8)	Not null	
FirstName	Varchar(20)	Not null	
LastName	Varchar(20)	Not null	
TelephoneNumber	Varchar(10)	Not null	
Sex	Char(1)	Not null	
RegistrationNumber	Int(10)	Nullable	

Student

Nome	Tipo	Null	Key
Email	Varchar(45)	Not null	Primary key/Foreign key
AcademicYear	Int	Not null	

Tutor

Nome	Tipo	Null	Key
Email	Varchar(45)	Not null	Primary key/Foreign key
State	Enum	Not null	
StartDate	Date	Not null	

FinishDate	Date	Nullable	
CommissionMember	Varchar(45)	Not null	Foreign key
RegisterId	Int	Not null	Foreign key

Request

Nome	Tipo	Null	Key
IdRequest	Int	Not null	Primary key
State	Enum	Not null	
Comment	Varchar(240)	Not null	
Date	Date	Not null	
Time	Int	Not null	
Duration	Int	Not null	
Student	Varchar(45)	Not null	Foreign key

Appointment

Nome	Tipo	Null	Key
IdAppointment	Int	Not null	Primary key
Details	Varchar(240)	Not null	
RequestId	Int	Not null	Foreign key
Tutor	Varchar(45)	Not null	Foreign key

Activity_Tutor

Nome	Tipo	Null	Key
IdActivity	Int	Not null	Primary key
Category	Enum	Not null	
Date	Date	Not null	
StartTime	Int	Not null	
FinishTime	Int	Not null	
Hours	Float	Not null	
State	Enum	Not null	
Details	Varchar(320)	Not null	
Tutor	Varchar(45)	Not null	Foreign key
RegisterId	Int	Not null	Foreign key

Register

Nome	Tipo	Null	Key
Id_Register	Int	Not null	Primary key

State	Enum	Not null	
ValidatedHours	Float	Not null	
TotalHours	Int	Not null	
PercentageComplete	Float	Not null	

Manages

Nome	Tipo	Null	Key
Tutor	Varchar(45)	Not null	Primary key/Foreign key
Request_Id	Int	Not null	Primary key/Foreign key

Contained_In

Nome	Tipo	Null	Key
AppointmentId	Int	Not null	Primary key/Foreign key
ActivityId	Ind	Not null	Primary key/Foreign key

Validates

Nome	Tipo	Null	Key
Commission_Member	Varchar(45)	Not null	Primary key/Foreign key
Activity_Id	Int	Not null	Primary key/Foreign key

3.5 Controllo degli accessi e sicurezza

Il controllo degli accessi è garantito tramite l'utilizzo di email e password per lo studente, il tutor e la commissione tutorato, che verranno richieste per ogni singolo accesso.

La sicurezza sui dati sensibili degli studenti è garantita dall'accesso controllato in quanto solo la commissione tutorato può avere accesso ai dati relativi agli studenti.

In TutoratoSmart ci sono diversi attori che hanno il permesso di eseguire diverse operazioni. Per schematizzare al meglio il controllo degli accessi si è utilizzata una matrice degli accessi, dove le righe rappresentano gli attori e le colonne le classi. Ogni entry (attore, classe) contiene le operazioni consentite da quell'attore sulle istanze di quella classe.

Le operazioni che gli utenti possono fare sono:

Sottosistema \ Attori	Gestione				
	Tutoring Request	Request Management	Tutoring Activity Management	Tutoring Supervision	User
Studente	• Compilazione richiesta • Visualizzazione stato richiesta • Modifica prenotazione	/	/	/	• Registrazione • Logout • Login
Tutor	/	• Visualizzazione dettagli richiesta • Gestione richiesta • Conferma appuntamento	• Visualizzazione calendario appuntamenti • Generazione registro • Visualizzazione registro • Visualizzazione attività lavorativa • Aggiunta attività lavorativa • Modifica attività lavorativa • Visualizzazione dettagli appuntamento • Modifica appuntamento	/	• Logout • Login
Commissione Tutorato	/	/	/	• Visualizzazione studenti • Convalida attività tutor • Registrazione tutor • Visualizzazione tutor	• Login • Logout

				• Visualizzazione dettagli attività • Visualizzazione dettagli registro	

Inoltre tutti gli attori utilizzeranno indirettamente il sottosistema Security che controlla gli accessi alle aree riservate e verifica l'avvenuta autenticazione degli utenti connessi prima di accedere alle pagine private.

3.6 Controllo flusso globale del sistema

Il sistema Tutorato Smart fornisce funzionalità che richiedono una continua interazione da parte dell'utente, per tal ragione abbiamo adottato un controllo del flusso globale del sistema di tipo event-driven, che è un tipo di controllo flessibile e guidato dagli eventi.

3.7 Condizione limite

3.7.1 Start-up

Per il primo start-up del sistema "Tutorato Smart" è necessario l'avvio di un web server che fornisca il servizio di un Database MySQL per la gestione dei dati persistenti e l'interpretazione ed esecuzione del codice lato server. In seguito, tramite l'interfaccia di Login, sarà possibile autenticarsi tramite opportune credenziali (e-mail e password). Una volta effettuato l'accesso, "Tutorato Smart" presenterà all'utente la propria HomePage, dalla quale sarà possibile usufruire di tutte le funzionalità che la piattaforma offre.

3.7.2 Terminazione

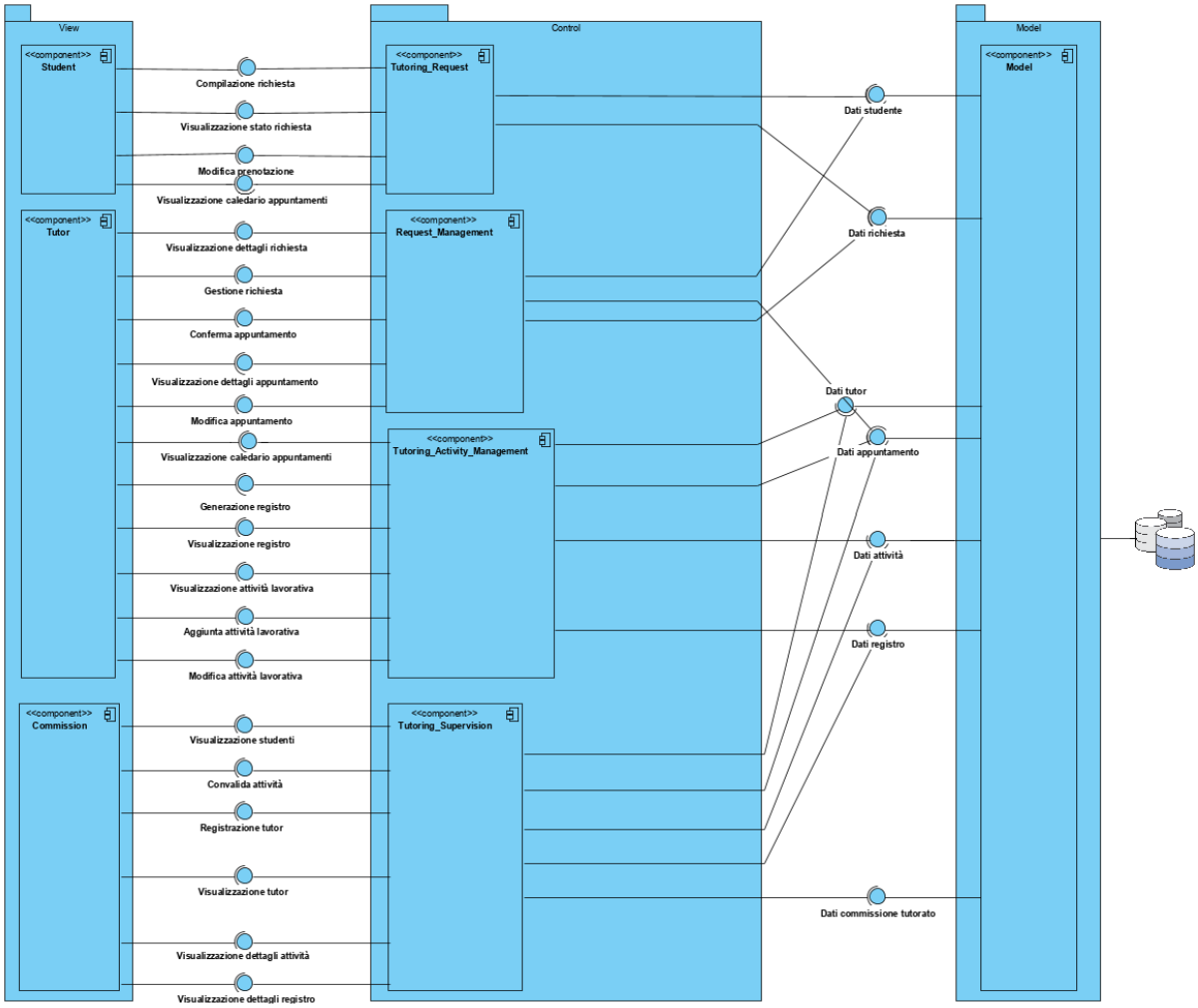
Al momento della chiusura dell'applicativo si ha la terminazione del sistema con un regolare Log-out dal sistema. Per consentire la corretta terminazione del server, l'amministratore del sistema dovrà effettuare la procedura di terminazione, dopo la quale nessun client potrà connettersi al sistema.

3.7.3 Fallimento

Possono verificarsi diversi casi di fallimento del sistema:

1. Nel caso di guasti dovuti al sovraccarico del database con successivo fallimento dello stesso, è prevista come procedura preventiva il salvataggio periodico dei dati sotto forma di codice SQL per la successiva rigenerazione del DB.
2. Nel caso in cui si verifichi un'interruzione inaspettata dell'alimentazione, non sono previsti metodi che ripristinino lo stato del sistema a prima dello spegnimento inaspettato.
3. Un altro caso di fallimento potrebbe derivare dal software stesso che causa una chiusura inaspettata dovuta ad errori commessi durante la fase di implementazione, non sono previste politiche correttive, l'unico processo che potrà essere eseguito è la chiusura del sistema e il suo successivo riavvio.
4. Un altro caso di fallimento potrebbe essere dovuto ad un errore critico nell'hardware, non è prevista alcuna misura correttiva.
5. Un altro caso di fallimento potrebbe essere dovuto ad un mancato salvataggio di dati causato da un malfunzionamento del sistema.

4. Servizi dei Sottosistemi



View: Interfacce che gestiscono l'interfaccia grafica e gli eventi generati dall'interazione dell'utente con il sistema. Suddivisa in:

- **Student**.
- **Tutor**.
- **Commission**.

Tutoring_Request: Offre 4 servizi all'interfaccia Student:

- Compilazione richiesta.
- Visualizzazione stato richiesta.
- Modifica prenotazione.
- Visulizzazione dettagli appuntamento.

Request_Management: Offre 5 servizi all'interfaccia Tutor:

- Visualizzazione dettagli richiesta.
- Gestione richiesta.
- Conferma appuntamento.
- Visualizzazione dettagli appuntamento.

- Modifica appuntamento.

Tutoring_Activity_Management: Offre 6 servizi all'interfaccia Tutor:

- Visualizzazione calendario appuntamenti.
- Generazione registro.
- Visualizzazione registro.
- Aggiunta attività lavorativa.
- Visualizzazione attività lavorativa.
- Modifica attività lavorativa.

Tutoring_Supervision: Offre 6 servizi all'interfaccia Commission:

- Visualizzazione studenti.
- Convalida attività tutor.
- Registrazione tutor.
- Visualizzazione tutor.
- Visualizzazione dettagli attività.
- Visualizzazione dettagli registro.

Model offre 2 servizi a Tutoring_Request:

- Dati studente.
- Dati richiesta.

Model offre 3 serivizi a Request_Management:

- Dati studente.
- Dati richiesta.
- Dati appuntamento.

Model offre 4 servizi a Tutoring_Activity_Management:

- Dati tutor.
- Dati appuntamento.
- Dati attività.
- Dati registro.

Model offre 5 servizi a Tutoring_Supervision:

- Dati tutor.
- Dati appuntamento.
- Dati attività.
- Dati registro.
- Dati commissione tutorato.