Laurea Triennale in informatica-Università di Salerno
Corso di *Ingegneria del Software*- Prof.ssa F. Ferrucci

System Design Document

US UniSeat

Riferimento	RAD
Versione	1.0
Data	06/12/2019
Destinatario	Prof.ssa Ferrucci Filomena
Presentato da	De Caro Antonio, De Santis Marco, Spinelli Gianluca, Capozzoli Lorenzo, Rocco Simone Pasquale
Approvato da	

Revision History

Data	Versione	Cambiamenti	Autori
26/11/2019	0.1	Stesura scheletro documento	De Caro Antonio
27/11/2019	0.2	Capitolo 1	Rocco Simone Pasquale
28/11/2019	0.3	Capitolo2 e prima versione del Capitolo 3	[tutti]
30/11/2019	0.4	Aggiunta prima versione del capitolo 4.	De Caro Antonio Spinelli Gianluca De Santis Marco
1/12/2019	0.5	Correzione di errori grammaticali	[tutti]
4/12/2019	1.0	Revisione documento	[tutti]

Sommario

1.Introduzione

1.1 Scopo del sistema

Lo scopo del sistema è quello di fornire agli studenti e docenti la possibilità di prenotare posti (per gli studenti) e aule (per i docenti). Ad oggi un sistema similare proposto è il sistema bibliotecario di prenotazione posti offerto dall'Università degli studi di Salerno. L'esigenza di UniSeat nasce dal bisogno di facilitare agli studenti la ricerca di posti a sedere negli edifici F, F2, F3 dell'Università degli Studi di Salerno.

L'obiettivo di UniSeat da un lato è facilitare la ricerca dei posti per gli studenti, al fine di ottimizzare il tempo di studio, dall'altro è permettere ai docenti di avere una piattaforma che gli permette di prenotare intere aule per attività extra-corso e inviare notifiche a tutti gli studenti che hanno effettuato prenotazioni in una determinata aula prenotata.

Il sistema progettato è una web app alla quale avranno accesso l'amministratore di sistema, i docenti e gli studenti. Possiamo dividere il sistema in tre macro-aree:

- Area Amministratore: gestione e vista delle prenotazioni e di tutti gli utenti, aggiunta docenti, aule, eliminazione utenti.
- Area Docente: gestione e prenotazione aule, visualizzazione prenotazioni.
- Area Studente: gestione e prenotazione posto, gestione lista amici, visualizzazione prenotazione

Il sistema deve fornire un metodo di autenticazione sicuro in modo che un utente non possa aver effettuato più di un accesso contemporaneamente e che dopo un tempo limite di sessione, bisogna reinserire la password per evitare accessi fraudolenti. Il sistema inoltre dovrà essere facile da apprendere e intuitivo da utilizzare, consentendo una navigazione fluida e permettendo l'utilizzo del sistema anche senza il consulto della documentazione.

1.2 Design Goals

I Design Goals sono organizzati in cinque categorie: Performance, Dependability, Cost, Maintenance, End User and Criteria. I Design Goals identificati nel nostro sistema sono i seguenti:

Criteri di performance

- Tempo di risposta:

  Il software deve consentire una navigazione rapida a tutti i tipi di utenti, quindi tempi di risposta minimi nello svolgimento delle funzionalità da esso offerte.

- Memoria

  La memoria complessiva del sistema dipenderà dalla memoria utilizzata per il mantenimento del Database.

Criteri di affidabilità

- Robustezza:

  Il sistema informerà l'utente di eventuali input errati attraverso messaggi di errore.

- Affidabilità:

  Il sistema deve garantire l'affidabilità dei servizi proposti. I risultati visualizzati saranno attendibili. Per quanto riguarda gli edifici e le aule, rispecchieranno la reale situazione all'interno dell'Università degli Studi di Salerno. Ad esempio, per ogni edificio ci saranno esattamente quelle aule con i rispettivi posti. Gli orari dei docenti saranno relativi all'anno accademico in corso. Il processo di login da parte di tutti gli utenti sarà gestito in modo affidabile, assicurando il corretto funzionamento del sistema.

- Disponibilità:

  Una volta online, il sistema sarà disponibile per tutti gli studenti e docenti registrati.

- Tolleranza ai guasti:

  Il sistema potrebbe essere soggetto a fallimenti dovuti a varie cause tra cui un sovraccarico di dati nel database. Per risolvere il problema, periodicamente sarà previsto un salvataggio dei dati sotto forma di codice SQL necessario alla rigenerazione del Database

- Sicurezza:

  L'accesso al sistema sarà garantito mediante email e password.

Criteri di costi

- Costo di sviluppo:

  È stimato un costo complessivo di 250 ore per la progettazione e lo sviluppo del sistema (50 per ogni membro del progetto).

Criteri di manutenzione

- Estensibilità:

  È possibile aggiungere nuove funzionalità al sistema, dettate dalle esigenze del cliente o dall'avvento di nuove tecnologie. Inoltre, se modificato adeguatamente, il sistema potrà essere esteso ad altri edifici dell'Università degli Studi di Salerno.

- Adattabilità:

  Il sistema funziona solo per il dipartimento di informatica e per l'Università degli studi di Salerno, ma è adattabile a più dipartimenti o a più università.

- Portabilità:

  L'interazione con il sistema avviene tramite browser, quindi possiamo definirlo portabile. Poiché il sistema viene sviluppato come una web application, esso è accessibile da qualunque dispositivo, che sia esso mobile o meno, purché abbia un browser installato. Questa caratteristica garantisce la portabilità dello stesso.

- Tracciabilità dei requisiti:

  La tracciabilità dei requisiti sarà possibile grazie ad una matrice di tracciabilità, attraverso la quale sarà possibile retrocedere al requisito associato ad ogni parte del progetto. La tracciabilità sarà garantita dalla fase di progettazione fino al testing.

Criteri di usabilità

- Usabilità:

  Il sistema sarà di facile comprensione e utilizzo, permettendo di effettuare in modo semplice e immediato le varie operazioni grazie a un'interfaccia intuitiva, di facile comprensione e utilizzo. 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.

- Utilità:

  Il lavoro dell'utente verrà supportato nel miglior modo possibile dal sistema, infatti l'utente compirà le operazioni consentite al fine di ottimizzare il tempo per la ricerca di un posto in cui studiare.

1.2.1 Design Trade-off

Memoria vs Estensibilità:

Il sistema deve permettere l'estensibilità a discapito della memoria utilizzata. Tale preferenza permette al cliente di richiedere agli sviluppatori nuove funzionalità, dando meno importanza alla memoria utilizzata.

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.

Disponibilità vs Tolleranza ai guasti

Il sistema deve essere sempre disponibile all'utente in caso di errore in una funzionalità, anche al costo di rendere non disponibile quest'ultima per un lasso di tempo.

Criteri di manutenzione vs Criteri di performance

Il sistema sarà implementato preferendo la manutenibilità alla performance in modo da facilitare agli sviluppatori nel processo di aggiornamento del software a discapito delle performance del sistema.

1.3 Definizioni, acronimi e abbreviazioni

Definizioni

Posto: rappresenta una singola postazione all'interno di un'aula, dotata di seduta e piano d'appoggio, con eventuali prese elettriche o computer.

Aula o Laboratorio: luogo fisico dove sono collocati i *Posti*.

Studente: rappresenta uno studente universitario, ognuno avente un proprio account in cui saranno specificati:

1. Nome
2. Cognome
3. E-Mail (@studenti.unisa.it)
4. Password

Docente: rappresenta un docente universitario, ognuno avente un proprio account in cui saranno specificate le seguenti informazioni:

1. Nome
2. Cognome
3. E-Mail (@unisa.it)
4. Password

Admin: rappresenta la figura amministrativa del sistema, in grado di aggiungere, rimuovere o modificare aule.

Prenotazione: rappresenta l'azione che effettua lo studente per occupare un *Posto*.

Servizi extra: indica la presenza di prese sui banchi, computer ed altre attrezzature messe a disposizione degli *Studenti*.

Amico: studente presente all'interno di una lista amici di un altro studente.

Posto = **Postazione**

All'interno del documento, la parola **"Aula"** fa riferimento anche al **"Laboratorio"**.

Draft RAD: Per Draft RAD si intende la versione del documento RAD prodotta dalla fase di Raccolta dei Requisiti.

Quando un utente è **"loggato"** vuol dire che ha effettuato l'autenticazione.

Acronimi

SDD = System Design Document (Documento di Progettazione del Sistema)

HW = Hardware

SW = Sowftware

DBMS = DataBase Management System

DI = Dipartimento di Informatica

AUT = Autenticazione

ACC = Account

STUD = Studente

DOC = Docente

NA = Nessuna

CD = Class Diagram

GUI = Graphical User Interface (Interfaccia utente)

DBA = Data Base Administrator (Amministratore del DataBase)

1.4 Riferimenti

Libro:

-- Object-Oriented Software Engineering (Using UML, Patterns, and Java) Third Edition

Autori:

-- Bernd Bruegge

-- Allen H. Dutoit

Documenti:

-- US_RAD.docx – Requirements Analysis Document

1.5 Panoramica del documento

Capitolo 1

Contiene l'introduzione gli obiettivi del sistema, i design goals, i trade-off e un elenco di definizioni, acronimi e abbreviazioni utili alla comprensione dell'intera documentazione.

Capitolo 2

Contiene la descrizione del sistema Corrente, o un sistema similare di riferimento.

Capitolo 3

Contiene la descrizione del sistema che verrà realizzato, degli obiettivi che andrà a realizzare, 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.

Capitolo 4

Contiene la rappresentazione dei servizi dei sottosistemi.

2. Architettura del Sistema Corrente

Attualmente non esiste un sistema per la gestione e prenotazione dei posti all'interno delle aule dell'Università degli Studi di Salerno. Un sistema simile proposto è quello relativo alla prenotazione dei posti nelle Biblioteche universitarie dell'Università degli Studi di Salerno.

3. Architettura del Sistema Proposto

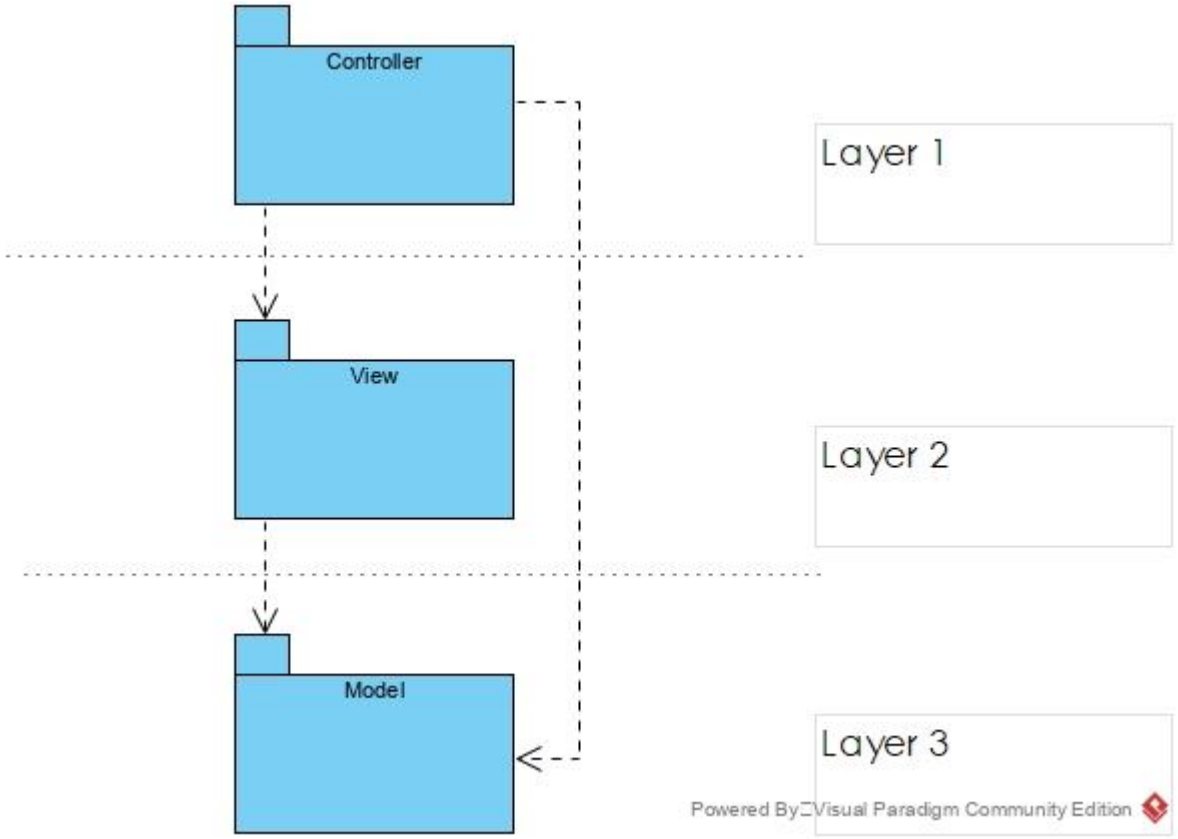
3.1 Panoramica

Il sistema che si vuole realizzare rientra nel campo della Green-field Engineering. Il nuovo sistema UniSeat è un'applicazione web, in locale per motivi di sicurezza, che verrà sottoposto a reengineering con il fine di aggiungere nuove funzionalità e migliorare quelle già presenti. Il sistema è rivolto ai docenti e agli studenti della facoltà di Informatica. Tutti gli utenti potranno effettuare login e log-out; gli studenti avranno la possibilità di registrarsi al sito mentre i docenti verranno registrati dall'admin. Il software da noi proposto metterà a disposizione varie funzionalità, a seconda dell'utente che effettuerà l'accesso. Lo studente potrà visualizzare l'elenco degli edifici e delle aule libere del dipartimento di Informatica, e per ogni aula potrà visualizzare i posti disponibili, occupati ed eventuali servizi offerti. Una volta scelta una determinata aula, lo studente potrà prenotare un posto indicando la durata dell'occupazione (non più di quattro ore). Il docente potrà anch'esso visualizzare l'elenco degli edifici e delle aule del dipartimento di Informatica, ed eventualmente prenotare un'aula al di fuori dell'orario di lezione indicando la durata dell'occupazione. Lo studente potrà visualizzare ed eventualmente modificare i dati personali. Inoltre, sia studente che docente potranno visualizzare le prenotazioni e nel caso disdirle. Un'altra figura che compare in UniSeat è quella dell'admin che si occuperà di: registrare un docente, gestire l'orario di disponibilità dell'aula sempre aggiornato per permettere la corretta visualizzazione delle aule libere, gestire le aule e gli edifici e gli utenti registrati alla piattaforma e può rendere non disponibile un'aula in caso di manutenzione. Lo stile architetturale usato è di tipo repository in quanto i sottosistemi che compongono il software accedono e modificano una singola struttura dati, nel nostro caso un database. L'architettura ci permette una gestione centralizzata di backup, sicurezza, controllo di accesso e recupero da errori, inoltre, risulta facile aggiungere nuovi sottosistemi. Nello specifico il sistema implementa un pattern di tipo MVC, diffuso nello sviluppo di interfacce grafiche di sistemi software object-oriented in grado di separare la logica di presentazione dei dati dalla logica di business. Si tratta di un'architettura multi-tier ovvero le funzionalità del sito sono separate e suddivise in più sottosistemi su più livelli in comunicazione tra loro.

3.2 Decomposizione in sottosistemi

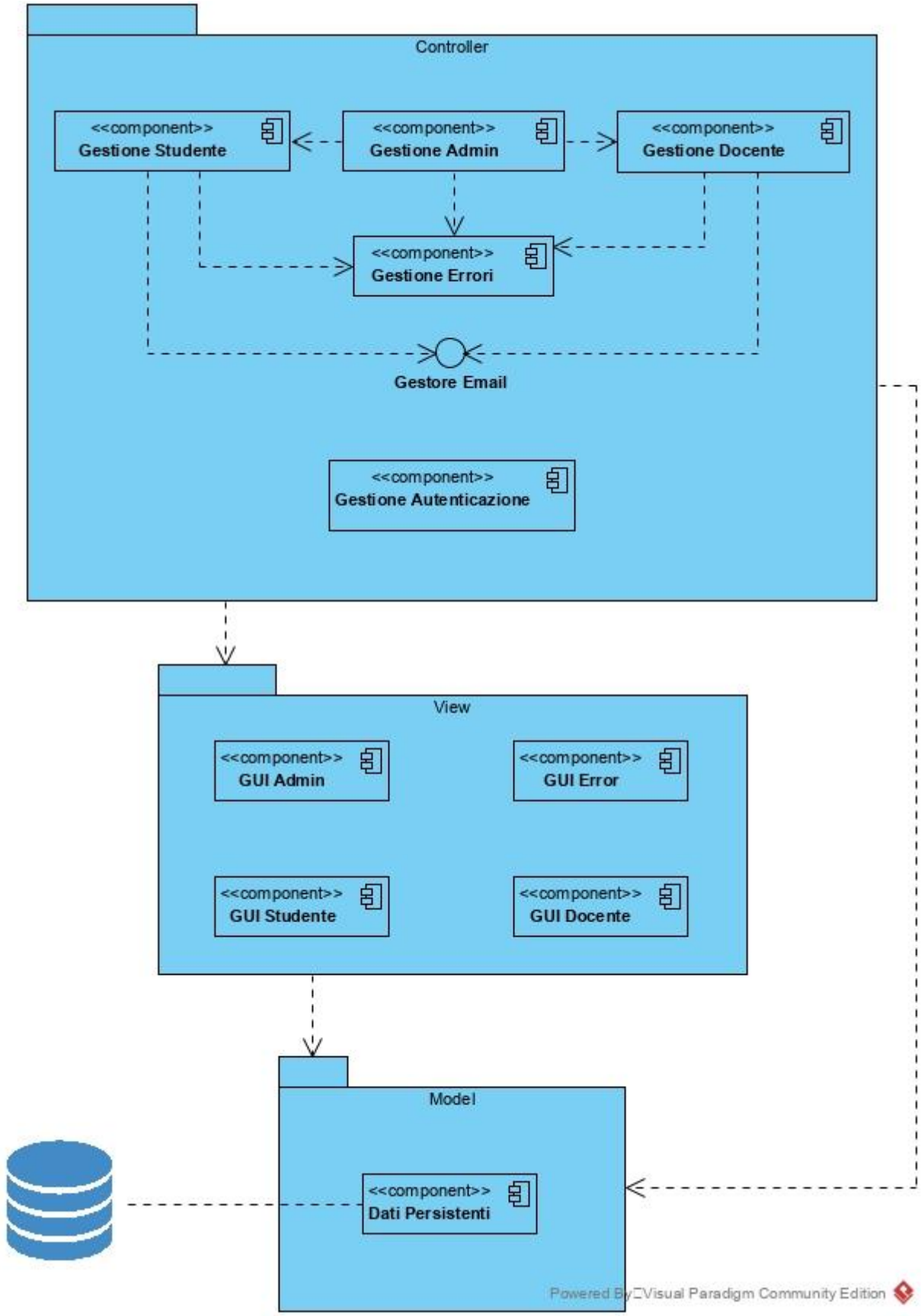
La decomposizione prevista per il sistema è composta da tre layer che si occupano di gestirne aspetti e funzionalità differenti:

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



Seguendo le esigenze del nostro sistema, abbiamo notato che un'architettura aperta migliorerebbe l'efficienza del nostro sistema, mantenendo comunque un basso accoppiamento tra le componenti. Ciascuna componente inoltre è stata idealizzata per avere un'alta coesione, che permetterà una facile manutenzione quando ce ne sarà il bisogno.

Component Diagram



Il layer **Control** gestisce 5 sottosistemi:

- Gestione Studente
- Gestione Docente
- Gestione Admin
- Gestione Prenotazioni
- Gestione Errori
- Gestione Autenticazione

Le componenti Gestione Studente, Gestione Docente, Gestione Admin, si interfacciano inoltre con una componente già esistente, ovvero Gestione Email, che si occupa di gestire le email che il nostro sistema deve inviare/ricevere.

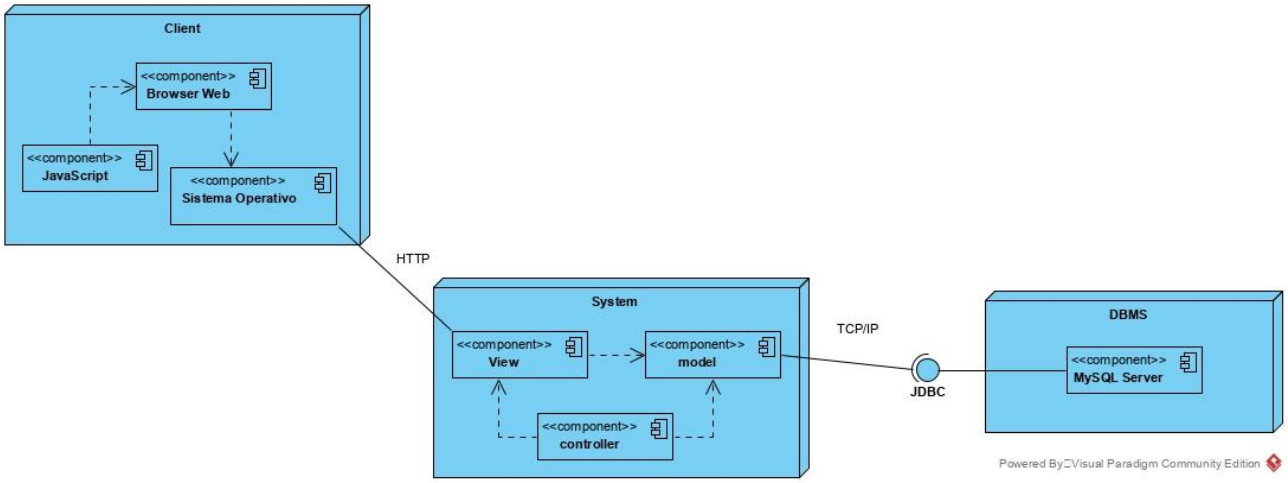
Il layer **View** gestisce 4 sottosistemi:

- GUI Studente
- GUI Docente
- GUI Admin
- GUI Errore

Il layer **Model** gestisce un solo sottosistema: Dati Persistenti, che si occupa della gestione dei dati persistenti del nostro sistema, e si interfaccia con il DBMS.

Deployment Diagram

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 Sistema sul quale vengono eseguite le funzioni apposite al completamento degli obiettivi del Client. La parte DBMS racchiude e gestisce la persistenza dei dati. Il Sistema viene eseguito sul web server Tomcat.



3.3 Mapping Hardware/Software

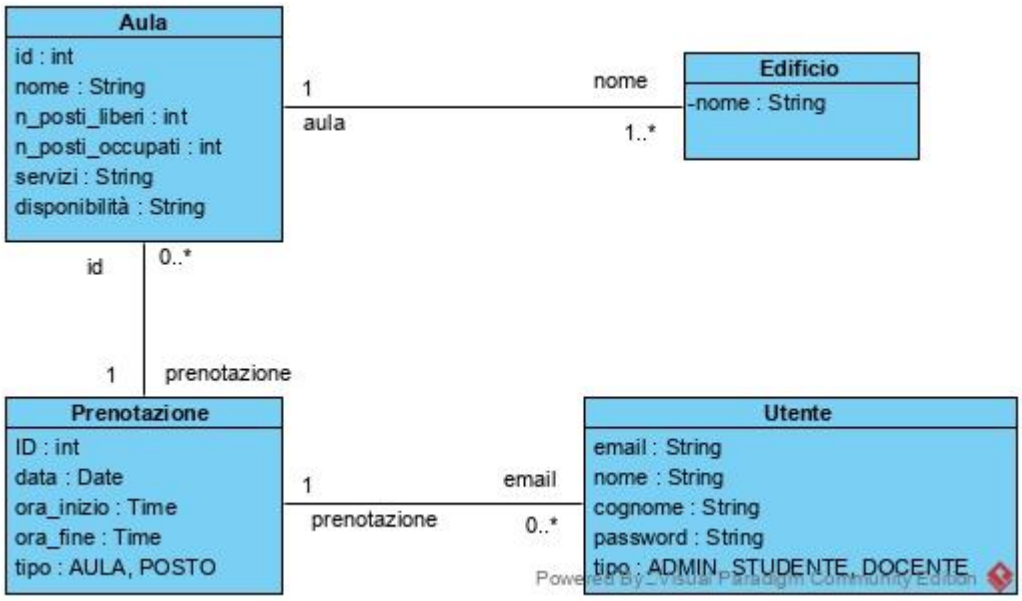
Il sistema che si desidera sviluppare utilizzerà una struttura hardware composta da un Server che risponderà ai servizi richiesti dal client. Il client è una macchina attraverso la quale un utente può collegarsi, usando una connessione a internet, per accedere al sistema mentre la macchina server gestisce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

Le componenti hardware e software di cui ha bisogno il client sono un computer, un tablet oppure un qualsiasi mobile dotato di connessione internet.

Per quanto riguarda il server, vi è la necessità di avere a disposizione una macchina avente connessione a internet e con capacità di immagazzinare grandi quantità di dati. La componente di cui ci si ha bisogno è un DBMS, nel nostro caso MySQL, per consentire il salvataggio dei dati in maniera persistente. Inoltre, è necessario usufruire dei servizi offerti da un web server, nello specifico Tomcat, per consentire la comunicazione con più client.

3.4 Gestione dati persistenti

Per la memorizzazione dei dati è stato scelto di utilizzare un Database Relazionale per consentire brevi tempi di risposta e ridurre i limiti di spazio di archiviazione. È possibile ripristinare lo stato del database in caso di danni software o hardware attraverso una copia dei dati fatta periodicamente. I dati presenti nel database sono privatizzati, vale a dire che il DBMS consente un accesso protetto, di conseguenza con operazioni diverse, l'utente, può accedere a diverse sezioni del database.



EDIFICIO

NOME

AULA

ID	NOME	EDIFICIO*	N_POSTI_LIBERI	N_POSTI_OCCUPATI	SERVIZI	DISPONIBILITÀ

UTENTE

EMAIL	NOME	COGNOME	PASSWORD	TIPO

PRENOTAZIONE

ID	UTENTE*	AULA*	DATA	ORA_INIZIO	ORA_FINE	TIPO

EDIFICIO

NOME	TIPO	CONSTRAINTS	KEY
NOME	VARCHAR(16)	NOT NULL	PRIMARY KEY

AULA

NOME	TIPO	CONSTRAINTS	KEY
ID	INT	NOT NULL, AUTO INCREMENT	PRIMARY KEY
NOME	VARCHAR(16)	NOT NULL	
EDFICIO	VARCHAR(16)	NOT NULL	FOREIGN KEY
N_POSTI_LIBERI	INT	NOT NULL	
N_POSTI_OCCUPATI	INT	NOT NULL	
SERVIZI	VARCHAR(1024)		
DISPONIBILITÀ	VARCHAR(1024)	NOT NULL	

UTENTE

NOME	TIPO	CONSTRAINTS	KEY
EMAIL	VARCHAR(48)	NOT NULL	PRIMARY KEY
NOME	VARCHAR(20)	NOT NULL	
COGNOME	VARCHAR(20)	NOT NULL	
PASSWORD	VARCHAR(256)	NOT NULL	
TIPO	ENUM('STUDENTE', 'DOCENTE')	NOT NULL	

PRENOTAZIONE

NOME	TIPO	CONSTRAINTS	KEY
ID	INT	AUTO_INCREMENT	PRIMARY KEY
UTENTE	VARCHAR(48)	NOT NULL	FOREIGN KEY
AULA	INT	NOT NULL	FOREIGN KEY
DATA	DATE	NOT NULL	
ORA_INIZIO	TIME	NOT NULL	
ORA_FINE	TIME	NOT NULL	

Per quanto riguarda l'inserimento degli edifici, essendo un'operazione che avviene molto raramente, viene effettuata direttamente dal DBA.

3.5 Controllo degli accessi e sicurezza

Con il sistema "Uni Seat" possono interagire diversi attori che possono eseguire diverse funzioni. Utilizziamo quindi una matrice degli accessi per capire quali attori possono eseguire quali funzioni. Quando un attore prova ad effettuare un'operazione da lui non permessa, viene rimandato ad una pagina di errore.

Nella seguente tabella verranno riportate:

- in alto, un'astrazione delle istanze delle classi del nostro sistema;
- sul lato sinistro, gli attori;
- la cella che incrocia attore e istanza rappresenta il permesso che quell'attore ha su quella istanza.

	Studente	Docente	Aula	Edificio	Prenotazione
Utente non registrato	1) Registra		1) Visualizza	1) Visualizza	
Studente	1) Modifica dati personali;		1) Visualizza 2) Prenota Posto	1) Visualizza	1) Visualizza 2) Elimina
Docente			1) Visualizza 2) Prenota Aula	1) Visualizza	1) Visualizza 2) Elimina
Admin	1) Visualizza 2) Elimina	1) Visualizza 2) Elimina 3) Registra	1) Visualizza 2) Aggiungi 3) Modifica	1) Visualizza 2) Aggiungi	1) Visualizza

3.6 Controllo flusso globale del sistema

Il flusso del sistema di "UniSeat" richiede una continua interazione dell'utente, per cui il controllo del flusso globale del sistema è di tipo event-driven, vale a dire che le azioni del sistema sono guidate dall'input dell'utente.

Per quanto riguarda la concorrenza, le funzionalità offerte dal Web Server, garantiscono un'interazione concorrente con tutti gli utenti connessi al sistema. Infatti, ogni utente connesso al sistema, tramite il suo browser web, avrà un thread dedicato tramite il quale il server interagirà con lui.

3.7 Condizione limite

Start-up

Lo start-up del sistema prevede l'avvio del web server nel quale il sistema è installato e l'avvio del DBMS per accedere ai dati persistenti memorizzati nel database. Quando sia Web Server che DBMS sono in esecuzione, il sistema carica in memoria centrale le servlet principali attraverso le quali gli utenti possono effettuare le operazioni. Dopo l'avvio del sistema l'utente può interagire con esso.

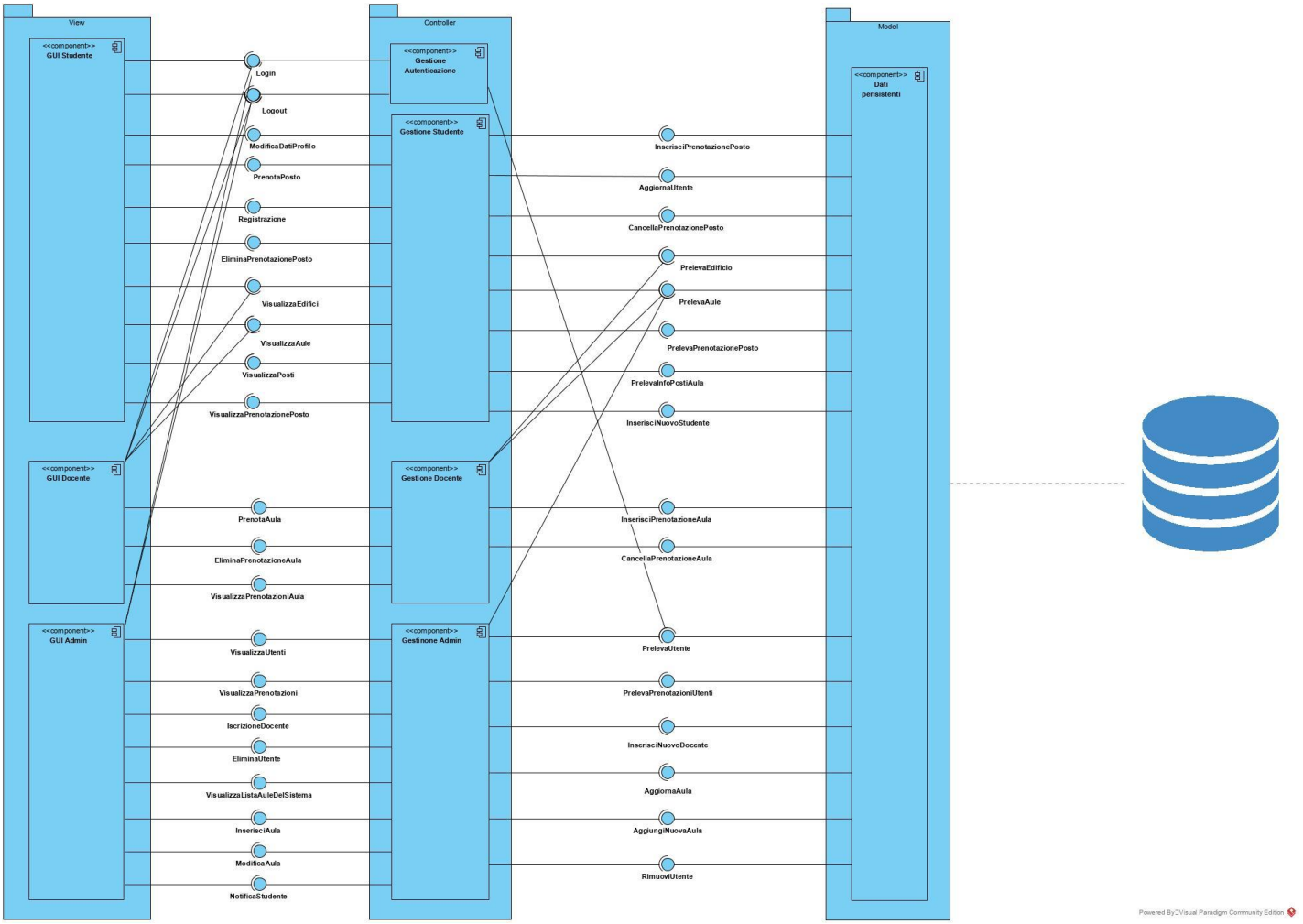
Shut-down

Quando il sistema deve essere arrestato, il gestore del sistema termina l'esecutivo del web server. Quando ciò avviene tutte le risorse che il sistema utilizza (connessione al database e connessione alla rete) vengono rilasciate, e nessun utente potrà più connettersi al sistema.

Fallimento

1. Nel caso in cui si presentasse un'interruzione inaspettata dell'alimentazione, non vi sono metodi che possano ripristinare lo stato del sistema precedente allo spegnimento non voluto. Qualsiasi transazione con il database viene annullata e viene ripristinato lo stato consistente più recente delle informazioni persistenti.
2. In caso di guasti dovuti al sovraccarico di informazioni al database, la rete viene congestionata. Il Web Server percepirà questo stato e inviterà tutti i client connessi di riprovare ad effettuare le operazioni in un secondo momento.
3. Nel caso in cui si presentasse una chiusura inaspettata del software, dovuta a errori avvenuti durante la fase di implementazione, non vi sono metodi per gestire queste condizioni. L'unica opzione consentita è riavviare il sistema.
4. Nel caso in cui si verificasse un sovraccarico di richieste al server, questo rimarrà congestionato.
5. Se un utente invia al server informazioni errate (volutamente o meno), o che non permettono la corretta esecuzione di un'operazione, il server risponderà con una pagina di errore.
6. Nel caso di errore critico dell'hardware, non è prevista una soluzione.

4. Servizi dei Sottosistemi



Servizi offerti dal controller per il sottosistema view <u>GUI Admin</u>:

- visualizza prenotazioni;
- login;
- logout;
- visualizza utenti;
- iscrizione docente;
- visualizza lista aule;
- inserisci aula;
- modifica aula;
- notifica studente;
- elimina utente;

Servizi offerti dal controller per il sottosistema view GUI Studente:

- login;
- logout;
- visualizza edifici;
- visualizza aule;
- visualizza posti disponibili;
- modifica dati profilo;
- visualizza prenotazione posto;
- registrazione;
- prenota posto;
- elimina prenotazione posto;

Servizi offerti dal controller per il sottosistema view GUI Docente:

- login;
- logout;
- visualizza edifici;
- visualizza aule;
- prenota aula;
- visualizza prenotazione aula;
- elimina prenotazione aula;

Servizi offerti dal model per il sottosistema controller Gestione Autenticazione:

- preleva utente;

Servizi offerti dal model per il sottosistema controller Gestione Studente:

- inserisci prenotazione posto;
- aggiorna utente;
- cancella prenotazione posto;
- preleva edificio;
- preleva aule;
- preleva prenotazione posto;
- preleva info posti aula;
- inserisci nuovo studente;

Servizi offerti dal model per il sottosistema controller Gestione Docente:

- preleva edificio;
- preleva aule;
- inserisci prenotazione aula;
- cancella prenotazione aula;

Servizi offerti dal model per il sottosistema controller <u>Gestione Admin</u>:

- preleva aule;
- preleva utente;
- preleva prenotazioni utenti;
- inserisci nuovo utente;
- aggiorna aula;
- aggiungi nuova aula;
- rimuovi utente;