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

DocuExchange

SDD System Design Document

DocuExchange Unisa Informatica

Presentato da:

Emanuele Bruno
Giacomo Impronta
Michele de Rosa
Paolo Erra

Revision History

Data	Versione	Cambiamenti	Autori
27/11/19	0.1	Strutturazione documento	Tutti
28/11/19	0.2	Aggiunta introduzione	Paolo Erra, Michele de Rosa, Giacomo Impronta
30/11/19	0.3	Aggiunta capitolo 2 e paragrafi 3.1, 3.2	Paolo Erra, Michele de Rosa
30/11/19	0.4	Stesura "Architettura sistema corrente"	Paolo Erra
1/12/19	0.5	Aggiunta diagramma ER	Emanuele Bruno, Paolo Erra
1/12/19	0.6	Aggiunta "Sistema proposto"	Giacomo Impronta, Michele de Rosa, Paolo Erra
10/01/20	0.7	Correzione Schema-ER e relative tabelle	Giacomo Impronta

Sommario

1. Introduzione

1.1 Obiettivi del sistema

Il sistema che si vuole realizzare ha come scopo lo scambio e la recensione di appunti da parte degli studenti dell'università di Salerno iscritti alla facoltà di Informatica. Obiettivo del sistema è quello di semplificare l'interazione tra gli studenti in modo da poter affrontare al meglio gli esami, grazie ad appunti e schemi che possano aprire la mente a chi ha dubbi o perplessità riguardo uno o più argomenti relativi ad un corso in particolare. Il sistema che si vuole realizzare è una piattaforma a cui avranno accesso sia gli studenti che vogliono condividere appunti, sia amministratori il cui compito è quello di assicurare un corretto utilizzo della piattaforma. Il sistema fornirà all'utente che accede funzionalità in base alla categoria a cui appartiene (studente oppure amministratore). In particolare il sistema deve poter dare la possibilità ad uno studente di:

- Poter inviare una richiesta di caricamento appunti sulla piattaforma, attraverso un'interfaccia semplice ed intuitiva in cui allegare gli appunti con le relative informazioni.
- Poter ricercare un altro studente per visionare i suoi appunti.
- Poter modificare/aggiornare il proprio profilo qualora fosse necessario.
- Poter recensire gli appunti scaricati in modo da fornire un'esperienza diretta per altri utenti, aiutando eventuali indecisi da quali appunti è preferibile studiare ai fini del superamento dell'esame.
- Visualizzare il suo profilo e la cronologia dei suoi appunti condivisi;

Il sistema deve inoltre dare la possibilità ad un amministratore di:

- Visualizzare tutte le richieste di caricamento appunti.
- Visionare una richiesta specifica e decidere se approvarne la pubblicazione o meno, nel momento in cui una richiesta viene verificata, il sistema dovrà generare un'e-mail automatica di avviso e inviarla allo studente che ha inoltrato la richiesta.
- Eliminare eventuali recensioni non consone e/o inappropriate ai fini di una corretta e sana esperienza sulla piattaforma per gli studenti.

Dato che il sistema ha accesso a dati sensibili degli studenti, deve fornire un metodo di autenticazione sicuro in modo che i dati siano protetti da accessi fraudolenti.
Inoltre, per una migliore usabilità, il sistema:
dovrà essere facile da apprendere ed intuitivo da utilizzare;
deve consentire la navigazione agevole per la fruizione delle funzionalità da lui offerte;
ridurre la documentazione utente al minimo;
permettere l'utilizzo del sistema anche senza consultare la documentazione.

1.2 Design Goals

I design goals definiti per il nostro sistema sono:

Criteri di performance

- Tempo di risposta:
  Il tempo per la visualizzazione della lista di appunti deve essere inferiore a 3 secondi;

Il tempo per la visualizzazione di un profilo deve essere inferiore a 2 secondi;

- Memoria:
  La dimensione complessiva del sistema dipende dalla memoria utilizzata per il mantenimento del database.

Criteri di affidabilità

- Robustezza:
  Eventuali input non validi immessi dall'utente saranno opportunamente segnalati attraverso messaggi di errore.

- Affidabilità:
  Il sistema deve garantire l'affidabilità dei servizi proposti, gli input degli utenti saranno accuratamente controllati per prevenire guasti.
  Gli appunti visualizzati, saranno affidabili e inerenti in quanto saranno controllati affinchè rispettano le caratteristiche date dai campi immessi per l'invio della richiesta di condivisione.
  Il login sarà gestito in modo affidabile garantendo il corretto funzionamento del sistema.

- Disponibilità:
  Una volta realizzato il sistema, sarà disponibile da tutti gli studenti del dipartimento di informatica ogni qualvolta ce ne sia la necessità.

- Tolleranza ai guasti:
  Il sistema può subire guasti dovuti al sovraccarico del database. Per ovviare al problema, periodicamente è previsto un salvataggio dei dati sotto forma di codice SQL necessario per la rigenerazione del database.

- Security:
  L'accesso al sistema è garantito mediante una email istituzionale e una password.
  La sicurezza è garantita in quanto ogni utente può svolgere solo le funzionalità da lui consentite.

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).

Criteri di manutenzione

- Estendibilità:
  Il sistema è estendibile in quanto potrà essere esteso per altri dipartimenti.

- Modificabilità:
  Il sistema è flessibile e modificabile, in caso di implementazione di nuove funzionalità o di modifiche a funzioni preesistenti.

- Adattabilità:
  Il sistema funziona solo per il dipartimento di informatica e per l'università degli Studi di Salerno, in ambito universitario, ma è adattabile a più dipartimenti.

- Tracciabilità dei requisiti:
  La tracciabilità dei requisiti è possibile grazie ad una matrice di tracciabilità, attraverso la quale è possibile riconoscere il requisito associato ad ogni parte del progetto.
  La tracciabilità è garantita dalla fase di progettazione fino al testing.

- 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.

Criteri utenti finali

- Usabilità:
  Il sistema sarà di facile comprensione e utilizzo, permettendo di effettuare in modo semplice e immediato le varie operazioni grazie a un'interfaccia user-friendly.
  L'intuitività è garantita in quanto il sistema avrà una buona prevedibilità, cioè la risposta del sistema ad unazione utente sarà corrispondente alle aspettative.

- Utilità:
  Il sistema si rende utile in quanto velocizza l'acquisizione di appunti, condivisi in modo immediato e semplice, gestendoli in formato digitale e non in maniera cartacea.

1.3 Definizioni, acronimi e abbreviazioni

DOCU: DocuExchange;
RAD: Requirements Analysis Document;
SDD: System Design Document;
HW:hardware ;
SW:software;
USER-FRIENDLY: Letteralmente "amichevole per l'utente", di facile utilizzo anche per chi non è esperto.
DB: DataBase;
Mysql: È un database Open Source basato sul linguaggio SQL, composto da un client a riga di comando e un server.
DBMS: Database Management System.
SQL: Structured Query Language; linguaggio standardizzato per database basati sul modello relazionale (RDBMS) progettato per: creare e modificare schemi di database.

1.4 Riferimenti

- NC_7_RAD_v0.12
- Slide del corso, presenti sulla piattaforma e-learning;
- Object-Oriented Software Engineering (Using UML, Patterns, and Java) Third Edition Autori: - Bernd Bruegge & Allen H. Dutoit;

1.5 Panoramica

Inseguito verranno elencati e descritti i capitoli presenti nel documento:

- **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, nel caso esista, le funzionalità offerte dal sistema corrente.

- **Capitolo 3:** Viene descritta l'architettura del sistema proposto, in cui sarà gestita 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:** Vengono descritti servizi dei sottosistemi.

2. Architettura del Sistema corrente

Attualmente non esiste un sistema software che si occupa di gestire tale problematica, ossia la condivisione di appunti tra studenti della facoltà di Informatica. Tale sistema nasce a partire dai bisogni dello studente.

3. Architettura del Sistema proposto

3.1 Panoramica

Il sistema da noi proposto è una piattaforma utilizzabile dal browser.
Gli utenti saranno di due tipi: studente e admin. Entrambi gli utenti avranno la possibilità di effettuare il login e logout, lo studente potrà registrarsi all'interno del sistema mentre i nuovi admin saranno registrati dagli stessi.
La piattaforma offrirà diverse funzionalità a seconda della tipologia dell'utente.
La funzionalità principale del sistema è l'invio della richiesta di condivisione dell'appunto da parte dello studente. In particolare, lo studente potrà compilare un form inserendo i dati relativi alla richiesta e caricare il suo appunto in formato pdf. Tale richiesta verrà poi inoltrata all'admin che si preoccuperà per la verifica della qualità dell'appunto, se la richiesta verrà accettata l'appunto sarà automaticamente pubblicato sulla piattaforma, se la richiesta verrà rifiutata essa verrà cancellata.
Lo studente potrà scaricare gli appunti condivisi sulla piattaforma da altri studenti a patto che si rispetti il seguente vincolo:
Dopo il download di tre appunti lo studente dovrà condividere un suo appunto per poter scaricare altri tre appunti dalla piattaforma. Senza questo vincolo lo studente non sarebbe spronato a condividere i suoi appunti e quindi la piattaforma rischierebbe di essere poco utilizzata.

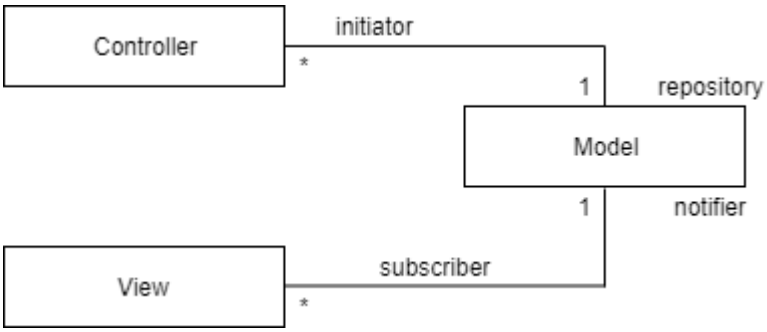
Lo stile architetturale adottato per il sistema è di tipo repository, si tratta di un sistema MVC (Model View Controller) un pattern architetturale diffuso nello sviluppo di sistemi software nell'ambito della programmazione orientata agli oggetti, in grado di separare la logica di presentazione dei dati dalla logica di business.
È un'architettura multi-tier in cui le varie funzionalità del sito sono logicamente separate e suddivise su più strati o livelli software differenti in comunicazione tra loro.

3.2 Decomposizione in sottosistemi

3.2.1 Decomposizione in Layer

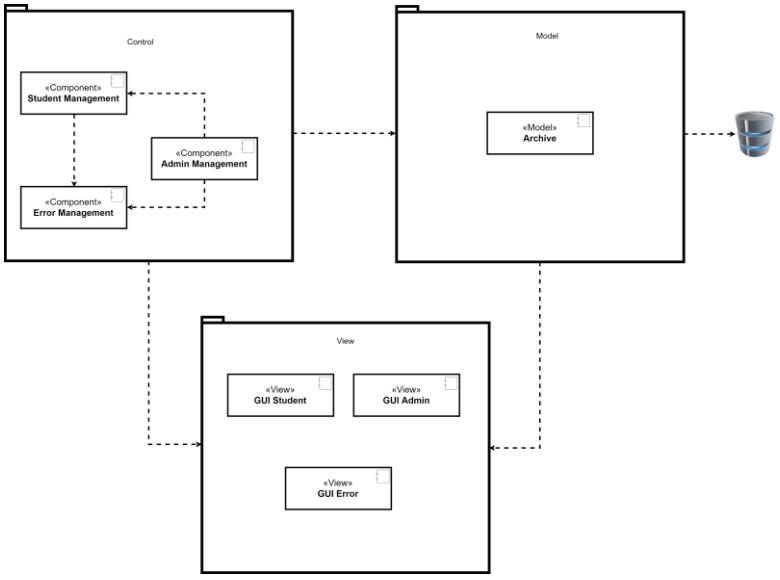
La decomposizione per il sistema è formata da tre layer responsabili per la gestione di funzionalità differenti:

- **View:** responsabile per la gestione dell'interfaccia grafica per l'utente e per le interazioni tra utente e sistema;
- **Controller:** responsabile della gestione logica del sistema;
- **Model:** responsabile della gestione dei dati persistenti del sistema e dello scambio dei dati tra i sottosistemi;



3.2.2 Decomposizione in sottosistemi

A seguito di una attenta analisi, abbiamo deciso di dividere il sistema nei seguenti sottosistemi in modo da avere un'architettura efficiente. Si è deciso di gestire i componenti con basso accoppiamento e con un elevata coesione in quanto si garantirà, in caso di future modifiche il minor numero di aggiornamenti da effettuare in tutti i sottosistemi.



Il sottosistema View sarà responsabile della gestione di due sottosistemi identificati come oggetti di tipo boundary:

- GUI Student
- GUI Admin

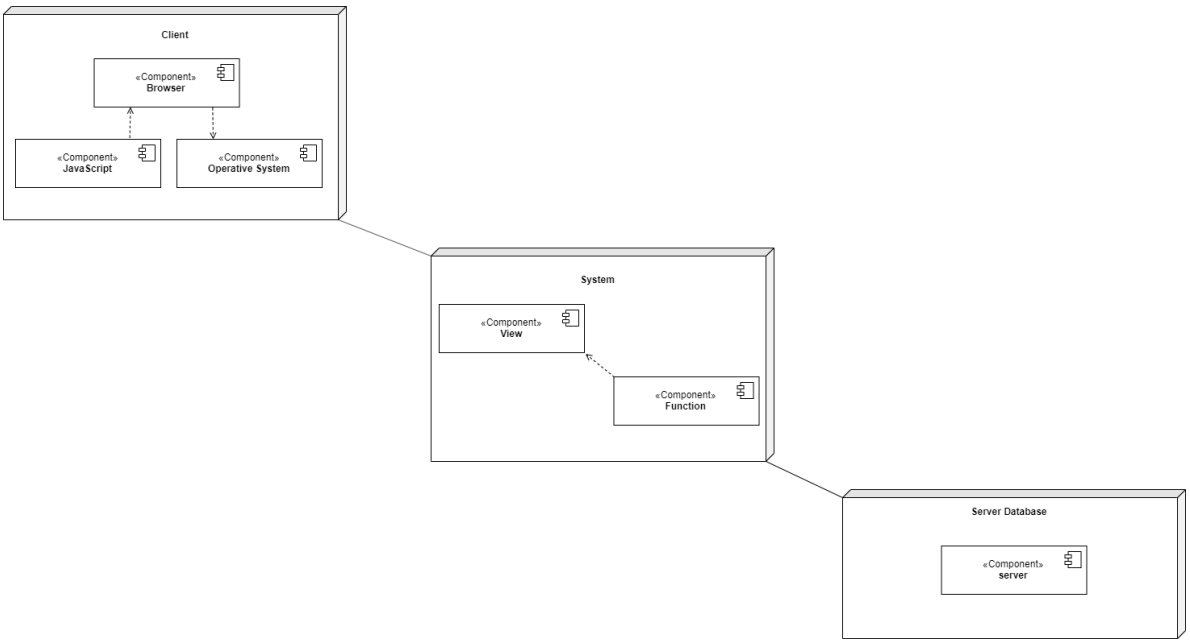
Il sottosistema Control è responsabile della gestione di tre sottosistemi:

- Admin Management
- Student Management

Il livello Model sarà responsabile per la gestione del database:

3.2.3 Deployment Diagram

L'utente potrà richiedere le funzionalità utilizzando l'interfaccia che il sistema mette a disposizione. Il sistema sarà un sito web quindi sarà utilizzabile attraverso un browser capace di interpretare JavaScript in modo tale da eseguire in maniera corretta tutte le funzionalità definite dal sistema. Il client connette lo strato di View del System nel quale vengono eseguite le funzionalità richieste, mentre il Server si occupa della gestione dei dati persistenti del sistema. L'architettura del sistema non ha la necessità dell'utilizzo di componenti hardware e software esterni.



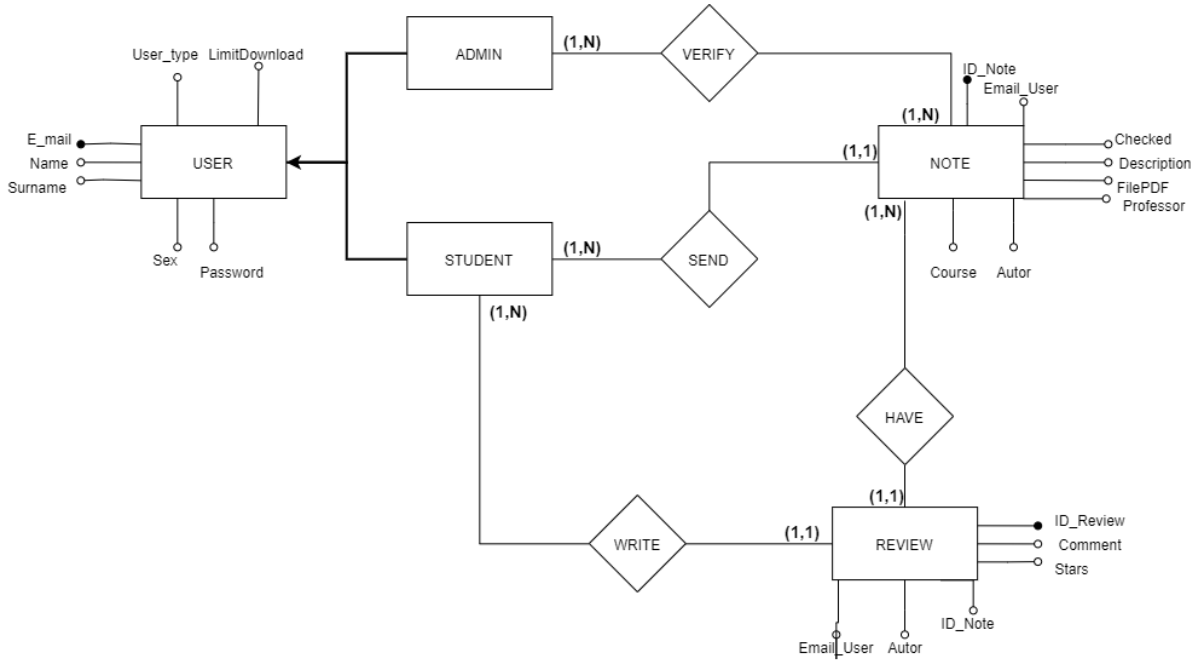
3.3 Mapping hardware/software

La struttura Hardware che si vuole andare a realizzare consta di un server che risponderà alle richieste inoltrate di un client. Il Server avrà il compito di gestire la logica e i dati persistenti contenuti nel database. Il client può essere invece una qualsiasi macchina avente connessione ad Internet ed un browser per interagire con la piattaforma. Client e server comunicheranno tra di loro attraverso il protocollo HTTP in modo da dare la possibilità al client di inoltrare richieste al server che provvederà a soddisfarle. Il server necessita di una macchina con connessione ad Internet ed in grado di immagazzinare alte quantità di dati. Per quanto riguarda il lato software è necessario un DBMS in modo da consentire la gestione del database.

3.4 Gestione dati persistenti

I dati all'interno del Sistema vengono gestiti tramite un database relazionale, così da avere un accesso efficiente ai dati con conseguenti vantaggi in brevi tempi di risposta e un ampio spazio di archiviazione.
Inoltre, viene garantito l'affidabilità dei dati salvando una copia, cosicché in caso di danni HW/SW si possa ripristinare lo stato del database.
Viene garantita la sicurezza poiché i dati sono privatizzati, ovvero il DBMS consente un accesso protetto. Ciò permette a utenti diversi di accedere a diverse sezioni del database ed eseguire diverse operazioni.

SCHEMA ER



User

E_mail	Name	Surname	Sex	Password	User_type

Note

ID_Note	Checked	Description	FilePDF	Professor

Review

ID_Review	Comment	Stars

User

Nome	Tipo	Null	Key
EMail_User	Varchar(35)	Not null	Primary key
Name	Varchar(20)	Not null	
Surname	Varchar(20)	Not null	
Sex	Varchar(1)	Not null	
Password	Varchar(20)	Not null	
Type	Integer(1)	Not null	
LimitDownload	Integer(1)	nullable	

Note

Nome	Tipo	Null	Key
ID_Note	Integer(10)	Not null	Primary key
Course	Varchar(40)	Not null	
Professor	Varchar(20)	Not null	
Description	Varchar(255)	Not null	
FilePDF	LongBlob	Not null	
Email_User	Varchar(35)	Not null	Foreign key
Checked	Integer(1)	Not null	
Autor	Varchar(45)	Not null	

Review

Nome	Tipo	Null	Key
ID_Review	Integer(11)	Not null	Primary key
Comment	Varchar(255)	Not null	
Stars	Int(10)	Not null	
Email_User	Varchar(35)	Not null	Foreign key
Autor	Varchar(35)	Not null	
ID_Note	Int(10)	Not null	Foreign key

3.5 Controllo degli accessi e sicurezza

Nel sistema saranno presenti due diversi tipi di attori: studente e admin. Ogni tipo di attore avrà il permesso di richiedere diverse funzionalità. Per schematizzare al meglio il controllo degli accessi si vuole utilizzare una matrice degli accessi, le righe rappresentano gli attori mentre le colonne rappresentano le classi.
Ogni entry (attore, classe) contiene le operazioni consentite da quell'attore sulle istanze di quella classe.

Sottosistema	**Gestione**		
Attore	**Appunti**	**Richiesta**	**Gestione utente**
Studente	• Visualizza appunti; • Recensione appunti; • Download appunti; • Ricerca appunti:	• Invio richiesta di condivisione di un appunto;	• Login utente; • Gestione profilo utente; • Visualizza profilo; • Visualizza profilo studenti;
Admin	• Gestione appunti;	• Verifica richieste di condivisione;	• Login

3.6 Controllo flusso globale del sistema

Il flusso del sistema DocuExchange fornisce delle funzionalità che richiedono continue interazioni da parte dell'utente, quindi sarà necessario un controllo del flusso globale di tipo event-driver.

3.7 Condizione limite

Start-Up
Per far partire il sistema si deve avviare un web server che fornisca il servizio di un Database MySQL per la gestione dei dati persistenti. Dopo la partenza del sistema un utente può accedere al sistema tramite l'identificazione (Login) con le proprie credenziali. Dopo l'accesso l'utente sarà indirizzato alla propria home page, che sarà diversa a seconda dell'utente (Home page per studenti oppure Home page per admin).

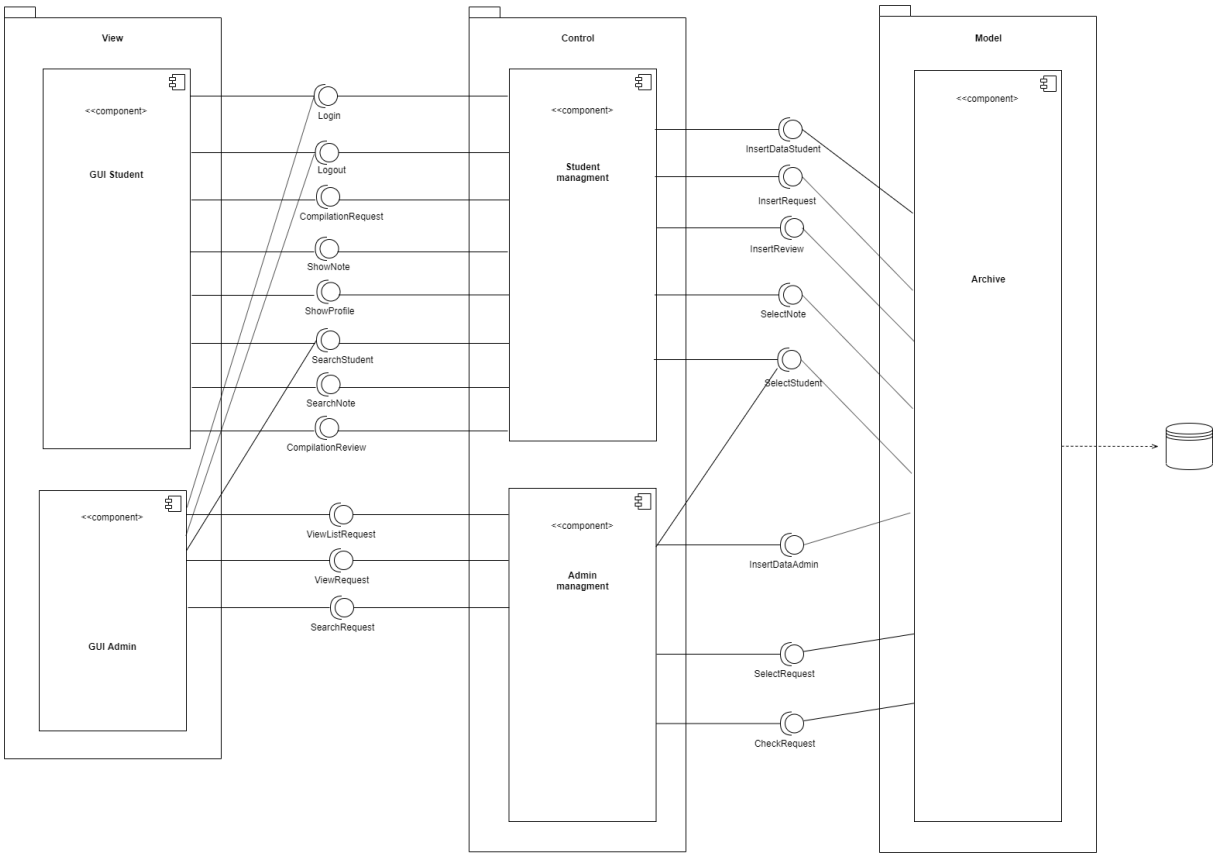
Terminazione
La propria sessione termina con un logout, mentre nel caso in cui si volesse far terminare correttamente il server sarà l'amministratore di sistema a provvedere alla corretta chiusura, dopo la quale nessun utente potrà connettersi fino al prossimo avvio.

Fallimento
Possono esserci vari fallimenti del sistema:

1. Interruzione alimentazione imprevista;
2. Chiusura inaspettata del sistema a causa del sistema stesso;
3. Problema dovuto all'hardware;
4. Sovraccarico di richieste per il database;

Per ovviare, in parte, a queste problematiche abbiamo periodici salvataggi delle informazioni da parte del database. È previsto inoltre un riavvio manuale qualora il sistema non dovesse più rispondere correttamente.

4. Servizi dei Sottosistemi



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

GUI Student offre 8 servizi all'interfaccia Control:

- Login
- Logout
- CompilationRequest
- ShowNote
- ShowProfile
- SearchStudent
- SearchNote
- CompilationReview

GUI Admin offre 5 servizi all'interfaccia Control:

- Login
- Logout
- ViewListRequest
- ViewRequest
- SearchRequest

Student Management offre 5 servizi all'interfaccia Model:

- InsertDataStudent
- InsertRequest
- InsertReview
- SelectNote
- SelectStudent

Admin Management offre 4 servizi all'interfaccia Model:

- InsertDataAdmin
- SelectRequest
- CheckRequest