SDD: System Design Document

EV English Validation

Riferimento	
Versione	0.8
Data	04/01/2019
Destinatario	Top Management
Presentato da	Alessandro Bacco, Ivan Buccella, Giuseppe Cirino, Alfonso Ingenito, Angelomaria Macellaro, Luigi Melchionno, Vincenzo Passariello
Approvato da	Giammaria Giordano, Valeria Pontillo

Revision History

Data	Versione	Cambiamenti	Autori
27/11/2018	0.1	Inizio stesura SDD	Macellaro Angelomaria
27/11/2018	0.1	Aggiunta Design Goals	Tutti
28/11/2018	0.2	Aggiunta punti 1.3,1.4,1.5	Tutti
30/11/2018	0.3	Aggiunta "Architettura del Sistema corrente"	Macellaro Angelomaria
30/11/2018	0.3	Aggiunta "Architettura del Sistema proposto"	Macellaro Angelomaria
01/12/2018	0.4	Completamento SDD	Macellaro Angelomaria
04/12/2018	0.5	Completamento SDD	Tutti
05/12/2018	0.6	Modifica nomi ER	Macellaro Angelomaria
30/12/2018	0.7	Aggiunta campo state nella tabella ENTE e modifica mapping	Cirino Giuseppe Melichionno Luigi
04/01/2019	0.8	Correzione di errori grammaticali e di formattazione	Cirino Giuseppe Angelomaria Macellaro

Sommario

1. Introduzione

1.1 Obiettivi del sistema

Il sistema che si vuole realizzare ha come scopo il monitoraggio dei dati e dei documenti relativi al conseguimento e alla convalida dei CFU relativi all'esame di lingua inglese del corso di laurea sia triennale che magistrale in informatica dell'Università degli Studi di Salerno.

Il nostro obiettivo è di realizzare un sistema che permetta di facilitare le pratiche burocratiche sia da parte dello studente richiedente che da parte degli organi amministrativi come la segreteria e il/la responsabile di dipartimento, inoltre dovrebbe permettere di gestire meno documentazione cartacea grazie ad un archivio con scopo equivalente ma di gestione digitale.

Per raggiungere questo obiettivo è stato individuato l'attuale processo burocratico che porta alla convalida dei relativi CFU attraverso la validazione dell'attestato posseduto dallo studente.

Il miglioramento delle procedure burocratiche porterebbe eventuali benefici anche nel caso in cui si voglia estendere il sistema ad altri servizi simili a quello proposto.

Il sistema progettato è una vera e propria web app a cui avranno accesso il/la responsabile dell'area didattica, la segreteria e gli studenti, facilitando il processo di convalida sia dei CFU e sia degli attestati in tempo reale in modo da concludere la procedura nel minor tempo possibile.

Possiamo dividere il sistema in tre categorie:

- Gestione dei file: attestati caricati dagli studenti e file Excel prodotti dal sistema con la lista dei relativi.
- Gestione delle email: email generate dal sistema in modo da facilitare il processo di convalida del certificato con l'ente.
- Gestione delle notifiche: semplici notifiche mandate all'utente per informarlo che l'attestato è stato validato/rifiutato o che è stato caricato un altro documento da validare.

Il sistema dovrà consentire il login per tutte e tre le figure individuate (studente, admin e segreteria) a seconda dei dati di login immessi.

Il sistema dovrà consentire allo studente di caricare gli allegati per iniziare la procedura di convalida, dovrà permettere all'admin di effettuare la convalida dello stesso attraverso o una mail auto-generata e mandata all'ente di interesse, oppure attraverso il sito dello stesso.

Il sistema dovrà consentire alla segreteria di validare i CFU nel caso in cui l'attestato venga ritenuto valido.

Dato che il sistema ha accesso a dati sensibili degli studenti il sistema 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 sono organizzati in cinque categorie: Performance, Dependability, Cost, Maintenance, End User Criteria. I Design Goals identificati nel nostro sistema sono i seguenti:

Criteri di performance

- Tempo di risposta:

Il prodotto software deve consentire una navigazione rapida ai vari utenti, quindi, tempi di risposta minimi nello svolgimento delle funzionalità da esso offerte.

- Memoria:
  La memoria necessaria al funzionamento del sistema dipende dalla memoria utilizzata per il mantenimento del Database.

Criteri di affidabilità

- Robustezza:
  Il sistema informerà l'utente di eventuali errori nel caso di immissione di input non validi attraverso degli appositi messaggi.

- Affidabilità:
  Il sistema deve garantire l'affidabilità dei servizi proposti. Il prodotto software sarà sviluppato in modo tale da controllare accuratamente le informazioni inserite in input dagli utenti.

  Il processo di login da parte di tutti gli utenti sarà gestito in modo affidabile, assicurando il corretto funzionamento del sistema.

- Disponibilità:
  Una volta caricato il sistema sarà disponibile a tutti gli utenti ogni qualvolta gli utenti ne richiederanno l'utilizzo.

- 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, è stato previsto un salvataggio automatici dei dati.
- Sicurezza:
  L'accesso al sistema avviene mediante username e password. Inoltre, la sicurezza è garantita in quanto ogni utente può svolgere solo le operazioni a lui consentite.

Criteri di costi

- Costo di sviluppo:
  È stimato un costo complessivo di 350 ore per la progettazione e lo sviluppo del sistema (50 per ogni project member).

Criteri di manutenzione

- Estensibilità:
  È possibile aggiungere nuove funzionalità al sistema, dettate dalle esigenze del cliente o dall'avvento di nuove tecnologie.

- 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 modificando i relativi controlli.

- 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 il browser Mozilla Firefox installato. Questa caratteristica garantisce la portabilità dello stesso.

- Tracciabilità dei requisiti:

Attraverso una matrice di tracciabilità, sarà possibile tracciare i requisiti.

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 user-friendly.
- Utilità:
  Il sistema si rende utile in quanto velocizza il processo burocratico gestendo i dati in formato digitale e non in maniera cartacea.

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 sempre essere 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 gli sviluppatori nel processo di aggiornamento del software a discapito delle performance del sistema.

1.3 Definizioni, acronimi e abbreviazioni

EV: English Validation;

RAD: Requirements Analysis Document;

SDD: System Design Document;

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

EV_RAD_v.1.8

Slide del corso, presenti sulla piattaforma e-learning,

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

-- Bernd Bruegge & Allen H. Dutoit

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

Capitolo 3: Viene presentata 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 presentati i servizi dei sottosistemi.

2. Architettura del Sistema corrente

Non esiste, attualmente, un prodotto software finalizzato a gestire le richieste di convalida dei CFU di lingua inglese da parte degli studenti del dipartimento di Informatica dell'Università degli Studi di Salerno; queste, infatti, vengono fatte manualmente collegandosi al sito dell'università e scaricando l'apposito modulo da compilare. Una volta compilato il documento, precedentemente scaricato, lo studente dovrà scannerizzarlo insieme al certificato e successivamente inviarli all'ufficio carriere del dipartimento di informatica tramite mail, aspettare che venga approvato dopo essere stato visionato dalla segreteria, dal presidente del dipartimento e infine dal consiglio didattico. Tutto ciò risulta molto sconveniente per uno studente, il cui tempo a disposizione è spesso limitato. Per agevolare le operazioni sopra citate, abbiamo deciso di sviluppare una piattaforma online che permette di svolgere le varie procedure da casa, ottimizzando i tempi, semplicemente registrandosi al sito. Questo, infatti, concederà allo studente la possibilità di scaricare il documento, compilarlo e ricaricarlo sulla piattaforma. Tramite un sistema di notifiche, inoltre, lo studente verrà informato sullo stato della propria richiesta. La procedura risulta dunque più rapida e ordinata, evitando inoltre eventuali casi di errore dovuti alle numerose richieste.

3. Architettura del Sistema proposto

3.1 Panoramica

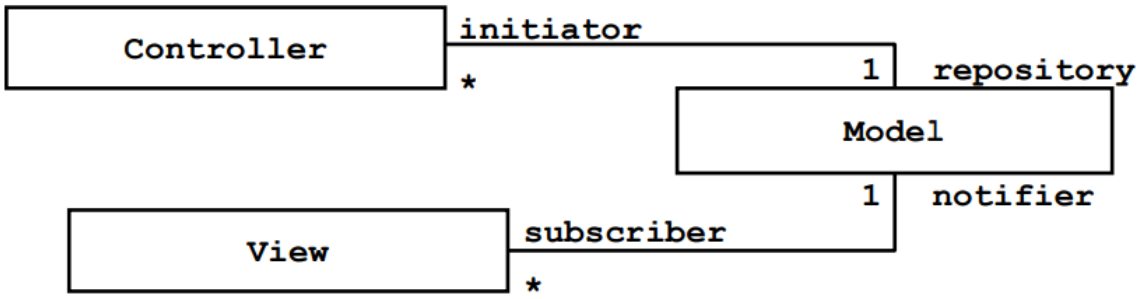
Il sistema da noi proposto è un'applicazione web, in locale per motivi di sicurezza, che verrà sottoposto a reengineering allo scopo di aggiungere nuove funzionalità e di migliorare quelle già esistenti. Gli utenti saranno di vario tipo: studente, segreteria e admin. Tutti gli utenti potranno effettuare login e log-out; gli studenti avranno la possibilità di registrarsi al sito mentre tutti gli altri utenti verranno registrati grazie al gestore. Ovviamente la piattaforma metterà a disposizione diverse tipologie di funzionalità, a seconda del tipo di utente che effettua l'accesso. In particolare, lo studente potrà compilare un form, per poter richiedere la convalida dei CFU di lingua inglese. Tale richiesta verrà poi inoltrata ai restanti utenti (Admin, Segreteria) che informati tramite il nostro sistema, provvederanno a convalidare i certificati ricevuti contattando gli enti preposti. Lo stile architetturale scelto è di tipo repository, perché sono adatti per applicazioni con task di elaborazione dati che cambiano di frequente, nello specifico è un sistema MVC. Quest' ultimo (Model View Controller) è un pattern architetturale molto diffuso nello sviluppo di interfacce grafiche di sistemi software object-oriented, in grado di separare logica di presentazione dei dati, dalla logica di business. È un'architettura multi-tier ovvero 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

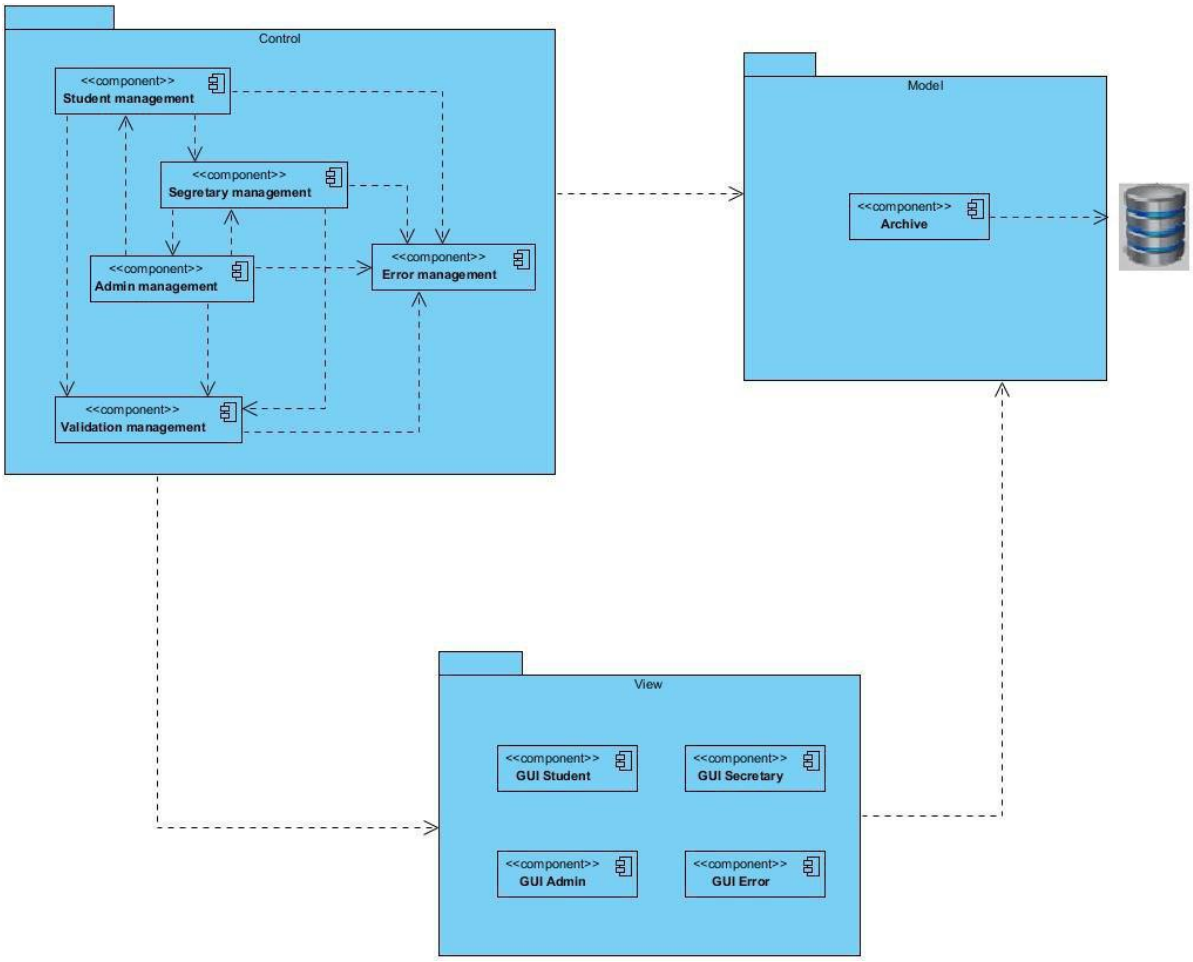
La decomposizione prevista per il sistema è composta da tre layer che si occupano di gestirne aspetti e funzionalità differenti:

- **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 effettuata sul sistema, abbiamo deciso di suddividerlo nei seguenti sottosistemi in modo tale da utilizzare un'architettura aperta per motivi di efficienza. 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. È stato deciso, inoltre, di aggiungere un ulteriore componente nel gestore dei file Excel che consente la rapida conversione dei dati processati in un file Excel formattato appositamente.
Il sistema si compone di otto sottosistemi:



Il livello View prevede la gestione di quattro sottosistemi e possiamo identificarli come oggetti "boundary" individuati nel rad:

- GUI Student
- GUI Secretary
- GUI Admin
- GUI Error

Il livello Control prevede la gestione di quattro sottosistemi:

- Admin Management
- Student Management
- Validation Management
- Error Management

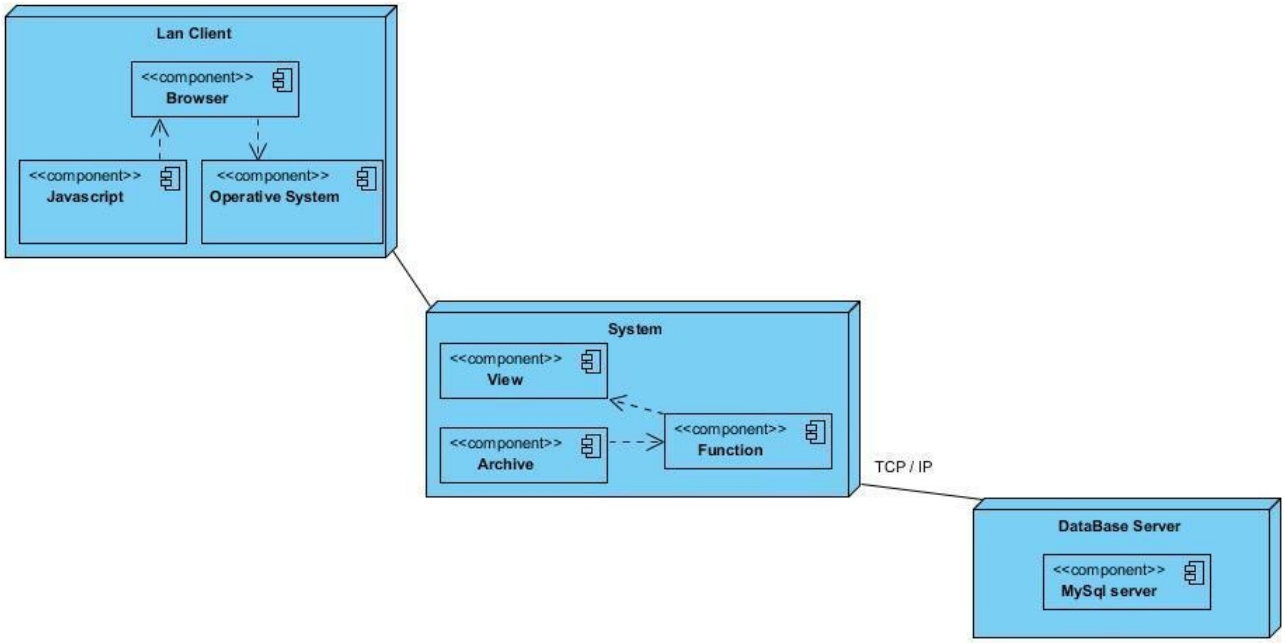
Il livello Model prevede la gestione di un sottosistema:

- Archive

Deployment Diagram

L'utente (Lan 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 Lan Client connette lo strato di view del System

sul quale vengono eseguite le funzioni apposite al completamento degli obiettivi del Lan Client. La parte Server racchiude e gestisce la persistenza dei dati. L'intera architettura non richiede ausilio di componenti hardware/software esterni.

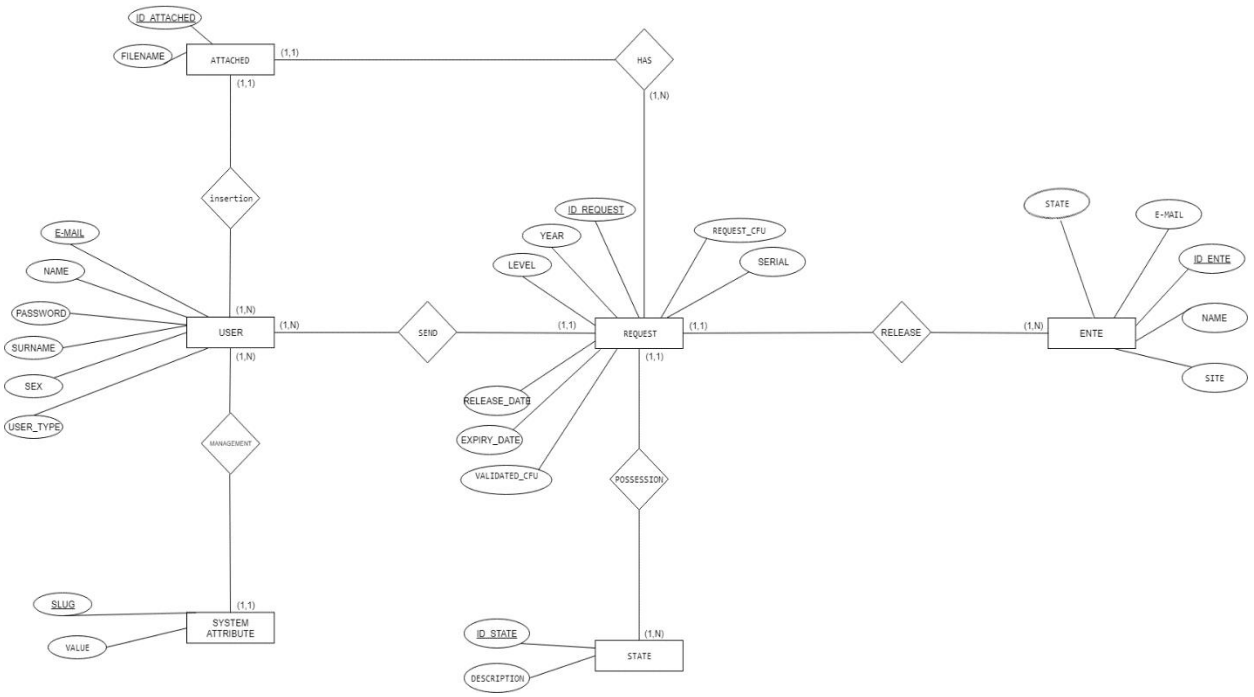


3.3 Mapping hardware/software

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.



USER

EMAIL	NAME	SURNAME	USER_TYPE	SEX	PASSWORD

SYSTEM_ATTRIBUTE

SLUG	VALUE

ATTACHED

ID_ATTACHED	FILENAME

REQUEST

ID_REQUEST	REQUESTED_CFU	LEVEL	RELEASE_DATE	EXPIRY_DATE
VALIDATED_CFU	SERIAL	YEAR		

ENTE

ID_ENTE	NAME	EMAIL	SITE	STATE

STATE

ID_STATE	DESCRIPTION

USER

NAME	TYPE	NULL	KEY

EMAIL	VARCHAR (50)	NOT NULL	PRIMARY KEY
NAME	VARCHAR (50)	NOT NULL	
SURNAME	VARCHAR (50)	NOT NULL	
SEX	CHAR	NOT NULL	
PASSWORD	VARCHAR (50)	NOT NULL	
USER_TYPE	TINYINT (1)	NOT NULL	

SYSTEM_ATTRIBUTE

NAME	TYPE	NULL	KEY
SLUG	VARCHAR (20)	NOT NULL	PRIMARY KEY
VALUE	VARCHAR (50)	NOT NULL	
FK_USER	VARCHAR (50)	NOT NULL	FOREIGN KEY

ATTACHED

NAME	TYPE	NULL	KEY
ID_ATTACHED	INT (20)	NOT NULL	PRIMARY KEY
FILENAME	VARCHAR (50)	NOT NULL	
FK_REQUEST	INT (20)	NOT NULL	FOREIGN KEY
FK_USER	VARCHAR (50)	NOT NULL	FOREIGN KEY

REQUEST

NAME	TYPE	NULL	KEY
ID_REQUEST	INT (20)	NOT NULL	PRIMARY KEY
LEVEL	VARCHAR (7)	NOT NULL	
RELEASE_DATE	DATE	NOT NULL	
EXPIRY_DATE	DATE	NOT NULL	
YEAR	YEAR	NOT NULL	
REQUESTED_CFU	TINYINT (2)	NOT NULL	
SERIAL	INT (10)	NOT NULL	
VALIDATED_CFU	TINYINT (2)	NOT NULL	
FK_USER	VARCHAR (50)	NOT NULL	FOREIGN KEY
FK_CERTIFIER	INT (20)	NOT NULL	FOREIGN KEY
FK_STATE	INT (20)	NOT NULL	FOREIGN KEY

ENTE

NAME	TYPE	NULL	KEY
ID_ENTE	INT (20)	NOT NULL	PRIMARY KEY
EMAIL	VARCHAR (50)	NOT NULL	
NAME	VARCHAR (50)	NOT NULL	
SITE	VARCHAR (50)	NOT NULL	
STATE	VARCHAR (50)	NOT NULL	

STATE

NAME	TYPE	NULL	KEY
ID_STATE	INT (20)	NOT NULL	PRIMARY KEY
DESCRIPTION	VARCHAR (50)	NOT NULL	

3.5 Controllo degli accessi e sicurezza

In EV English Validation 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.

Sottosistema	Gestione		
Attore	Excel	Certificate Table	PDF
Studente			• Download • Upload
Admin	• Genera Excel • Compila Excel • Aggiorna File	• Visualizza • Determina Esito Richiesta • Correzione errori di battitura	
Segreteria		• Visualizza • Aggiungi CFU • Correzione errori di battiture	• Visualizza PDF

3.6 Controllo flusso globale del sistema

Il flusso del sistema EV English Validation fornisce una funzionalità che richiede una continua interazione da parte dell'utente, ragione per cui, il controllo del flusso globale del sistema è di tipo event-driven, ovvero guidato dagli eventi.

3.7 Condizione limite

Start-Up

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

Terminazione

Al momento della corretta chiusura dell'applicazione, si ha la terminazione del sistema con un regolare Log-out. 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.

Fallimento

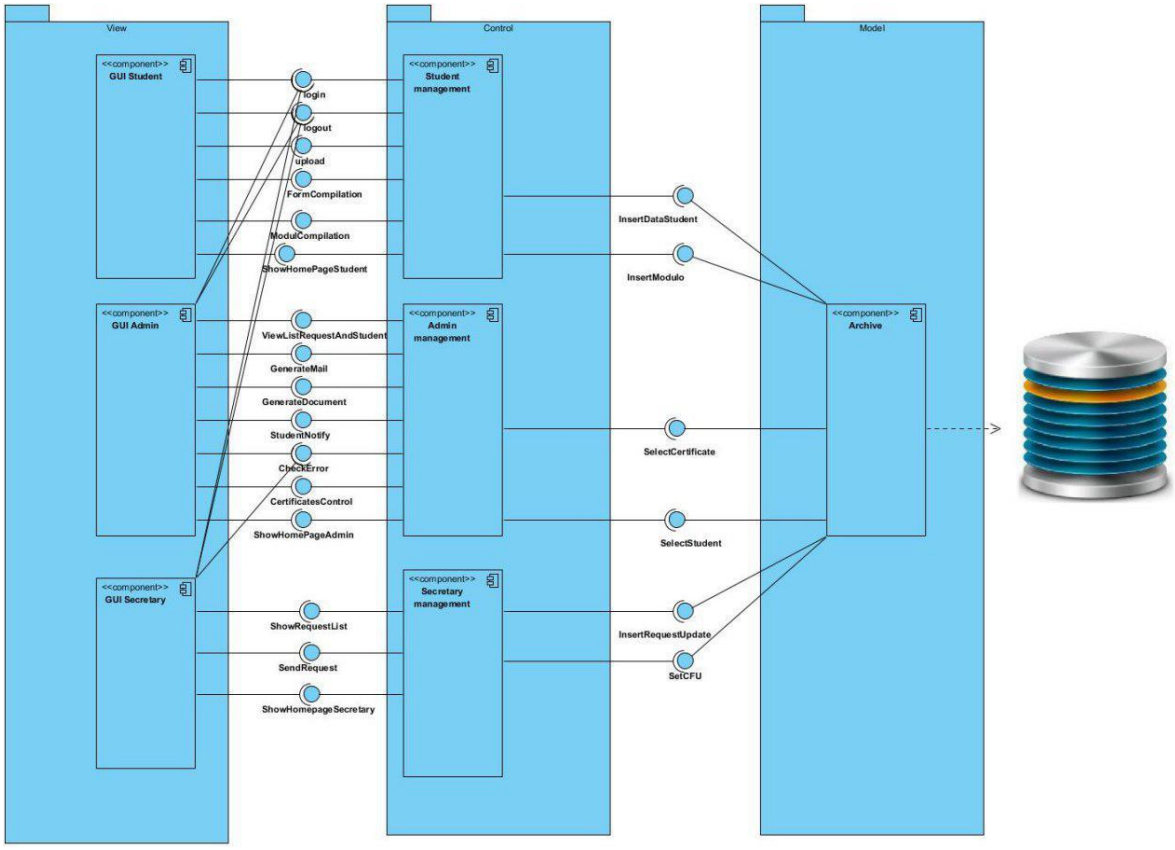
1. Nel caso di guasti dovuti al sovraccarico del database con successivo fallimento dello stesso è prevista come procedura preventiva il salvataggio periodico dei dati sotto forma di codice SQL per la successiva rigenerazione del DB.

2. Nel caso in cui si verifichi un'interruzione inaspettata dell'alimentazione non sono previsti metodi che ripristino lo stato del Sistema precedente allo spegnimento non voluto.

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 essendo previste politiche correttive, l'unica operazione consentita in questa particolare situazione 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, contro il quale non è prevista alcuna contromisura.

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 6 servizi all'interfaccia Control:

- Login
- Logout

- Upload
- FormCompilation
- ModuleCompilation
- ShowHomePageStudent

GUI Admin offre 9 servizi all'interfacciaControl:

- Login
- Logout
- ViewListRequestAndStudent
- GenerateMail
- GenerateDocument
- StudentNotify
- CheckError
- CertificatesControl
- ShowHomePageAdmin

GUI Secretary offre 2 servizi all'interfaccia di Visualizzazione:

- Login
- Logout
- ShowRequestList
- SendRequest
- ShowHomePageSecretary

Student Management offre 2 servizi all'interfaccia Model:

- InserDataStudent
- InsertModule

Admin Management offre 2 servizi all'interfaccia Model

- SelectStudent
- SelectCertificate

Secretary Management offre 2 funzionalità:

- InsertRequestUpdate
- SetCFU