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

ERASMUSSMART

System Design Document

Riferimento	
Versione	1.1
Data	16/12/2018
Destinatario	Prof.ssa F. Ferrucci
Presentato da	Andrea Valenti Tammaro Ruggero
Approvato da	

Revision History

Data	Versione	Descrizione	Autori
28/11/2018	0.1	Agginta introduzione, architettura del sistema corrente	Martin cioffi Cristiano Binetti
28/11/2018	0.2	Prima stesura	Giovanni Buonincontri Andrea Massaro Marianna Corinaldesi Cristiano Binetti
03/12/2018	1.0	Revisione stesura	Andrea Valenti Ruggero Tammaro
16/12/2018	1.1	Modifiche e revisione	Andrea Valenti Ruggero Tammaro

Project Team

Nome	Cognome
Giosuè	Sulipano
Andrea	Massaro
Giovanni	Buonincontri
Marianna	Corinaldesi
Vincenzo	Veniero
Cristiano	Binetti
Emilio	Clemente
Martin	Cioffi

Sommario

1. Introduzione

1.1 Obiettivi del sistema

Il dipartimento di Informatica dell'Università degli Studi di Salerno intende migliorare la propria posizione nei ranking nazionali e internazionali. Poiché tra gli elementi presi in considerazione nelle valutazioni l'internazionalizzazione assume un ruolo centrale, il dipartimento intende incrementare il numero di studenti in mobilità internazionale, sviluppando un sistema che fornisca uno strumento in grado di supportare lo studente nelle operazioni necessarie al progetto di mobilità Erasmus, automatizzando il processo e riducendo al minimo l'incremento di risorse e di personale da dedicare a questa attività. Per raggiungere tale obiettivo, il sistema offre le seguenti funzionalità:

- Una guida completa su tutte le procedure da seguire, composta da pagine Web statiche dove sono spiegate le operazioni da effettuare esternamente alla Web Application proposta (compilazione della candidatura, svolgimento del colloquio, compilazione del LA) e pagine Web dinamiche che hanno lo scopo di interagire con l'utente per permettergli di utilizzare le funzionalità offerte dal sistema (lista delle sedi aderenti e recensite tramite ES, inserimento di una recensione a seguito dell'esperienza di mobilità Erasmus).
- Un servizio di recensioni scritte da studenti che hanno concluso l'esperienza di mobilità, consultabile agevolmente dagli studenti che hanno intenzione di intraprendere il programma Erasmus, tramite una ricerca per nome delle sedi Università ed un filtro applicabile in base alla votazione (da 1 a 5) associata alla recensione.
- Un servizio di messaggistica online, in modo da agevolare l'interazione docente-studente. Gli studenti infatti possono comunicare tramite una chat istantanea direttamente con il proprio docente tutor assegnato ed i docenti tutor possono a loro volta comunicare con gli studenti seguiti, selezionando la chat con uno specifico studente tramite una lista contatti.
- Un servizio di news, in modo da offrire all'utente un aggiornamento constante relativo alle novità del programma di mobilità Erasmus.
- Un servizio di autenticazione, in modo da garantire all'utente l'accesso sicuro ad ES e la protezione dei dati sensibili. Inoltre, ad ogni profilo utente è associato un servizio di notificazione, il quale avvisa quando sono presenti nuovi messaggi non letti o nuove news non ancora visualizzate.

- Un sistema di FAQ dove ogni studente può rivolgere alla community di ES una domanda e rispondere a domande poste da altri utenti.

Infine, il sistema mira ad essere intuitivo da utilizzare, consentendo la navigazione agevole e permettendo l'utilizzo delle funzionalità offerte anche senza consultare la documentazione.
Il sistema ha quindi lo scopo di migliorare l'esperienza e l'attrattività della mobilità Erasmus.

1.2 Design Goals

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

Criteri di performance

- Tempo di risposta:

  Mediamente una richiesta da parte del client è soddisfatta in un tempo di risposta limite di 3 secondi. Queste prestazioni sono garantite grazie alla scelta di un server capace di sopportare il carico applicativo massimo.

- Throughput:

  Il sistema porta a compimento 1 task in un tempo limite di 3 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 garantisce la correttezza delle informazioni fornite, salvo eventuali cambiamenti nel regolamento per la partecipazione al bando di mobilità Erasmus. Il sistema verrà aggiornato con le nuove direttive il prima possibile.

- Disponibilità:

  Una volta realizzato il sistema sarà sempre disponibile, in quanto è totalmente appoggiato sulla rete, salvo problemi relativi alla caduta del server.

- Tolleranza ai guasti:

Il sistema può subire guasti dovuti al sovraccarico del database con successivo fallimento. Per ovviare al problema, periodicamente è previsto un salvataggio dei dati necessario per la rigenerazione del database.

- Security:

  Il sistema deve essere sicuro, deve essere protetto da accessi fraudolenti. Ogni utente deve poter accedere alla propria area, e le operazioni devono avvenire in modo sicuro per preservare la sicurezza dei dati personali e sensibili. Ciò avviene tramite un metodo di autenticazione fornendo una E-mail e una password. La sicurezza del database è garantita dal fatto che è accessibile solo all'amministratore.

- Safety:

  Il sistema in presenza di errori e fallimenti non comporta il danneggiamento fisico di chi lo utilizza.

Criteri di costo

- Costi di sviluppo:

  È stimato un costo complessivo di 500 ore per la progettazione e lo sviluppo del sistema (50 ore per ogni team member).

- Costi di distribuzione:

  Il costo necessario per la registrazione di un dominio sulla rete internet.

- Costi di aggiornamento:

  I costi di aggiornamento verranno stabiliti di volta in volta quando necessari.

- Costi di manutenzione:

  I costi di aggiornamento verranno stabiliti di volta in volta quando necessari.

- Costi di amministrazione:

  I costi di amministrazione verranno stabiliti in base al personale richiesto.

Criteri di manutenzione

- Estendibilità:

  Il sistema è estendibile in quanto esso stesso può essere esteso ad altri dipartimenti o ad altre università cambiando i dati su cui lavora.
  Anche perché sarà possibile aggiungere in futuro, nuove funzionalità al sistema, oppure creare nuove classi, con l'estensione di quelle già esistenti.

- Modificabilità:

Il sistema ha un'ottima modificabilità in quanto sarà progettato con un'ottima scalabilità, ossia avrà un basso accoppiamento ed un'alta coesione.

- Adattabilità:

  Il sistema può funzionare solo in ambito universitario, ma è adattabile a più dipartimenti modificando i dati su cui lavora.

- 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. Anche perché il sito sarà responsive e si adatterà ad ogni tipo di dispositivo (mobile, tablet, ecc.…).

- Leggibilità:

  La comprensione del sistema dalla lettura del codice sarà abbastanza facile; grazie all'utilizzo del modello MVC (model view controller), che renderà la struttura del codice più semplice e flessibile.

- Tracciabilità dei requisiti:

  La tracciabilità è garantita tramite opportuna documentazione dalla fase di progettazione fino al testing del sistema.

Criteri utenti finali

- Utilità:

  Il lavoro dell'utente verrà supportato nel miglior modo possibile dal sistema, infatti l'utente sarà guidato e agevolato nelle operazioni principali.

- Usabilità:

  L'usabilità di un sistema può essere analizzata considerando diverse caratteristiche. Questo sistema sarà molto semplice da apprendere anche senza la consultazione della documentazione; questo avviene soprattutto grazie all'ausilio di un'interfaccia estremamente intuitiva progettata sulla base della stereotipazione dell'utente "tipo" che lo utilizzerà (ovvero studenti giovani o comunque persone smart-friendly) e quindi delle sue capacità di comprensione.

1.3 Priorità dei design goal

I criteri di affidabilità sono prioritari per il sistema sviluppato, a seguire sono ritenuti importati i criteri di manutenzione e i criteri di utenti finali.

1.4 Trade-offs

- Prestazioni vs. Costi: Il sistema utilizza principalmente materiale open source, in modo da minimizzare i costi ed essere comunque adeguatamente performante.
- Funzionalità vs. Usabilità: Il sistema mira ad essere intuitivo da utilizzare, consentendo la navigazione agevole e permettendo l'utilizzo delle funzionalità offerte anche senza consultare la documentazione.
- Costi vs. Robustezza: Il sistema mira ad essere robusto e tollerante ai guasti, in modo da garantire la correttezza delle informazioni e delle funzionalità fornite.
- Efficienza vs. Portabilità: Il sistema mira ad essere portabile, infatti l'utilizzo di ES avviene mediante un web browser senza interazione con il sistema sottostante, c'è quindi indipendenza dal sistema operativo.
- Sviluppo rapido vs. Funzionalità: Il sistema mira ad offrire all'utente un'esperienza ottimale, consentendo l'utilizzo di funzionalità complete e sviluppate adeguatamente.
- Costi vs. Riutilizzabilità: Le funzionalità critiche del sistema verranno ideato e sviluppato esclusivamente dal team di sviluppo, non è comunque esclusa a priori la possibilità di utilizzare componenti precedentemente sviluppare dal team stesso o da esterni.

1.5 Definizioni, acronimi e abbreviazioni

- ES: Erasmus Smart
- LA: Learning Agreement
- SDD: System Design Document.
- ODD: Object Design Document.
- DB: Database.
- 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.

1.6 Riferimenti

- Bernd Bruegge e Allen H. Dutoit - Object-Oriented Software Engineering (using UML, Patterns and JavaTM) – Prentice Hall.
- Documentazione generata dalle interviste con i futuri stakeholder del sistema software.
- Documentazione reperita tramite ricerche su Internet.
- Documentazione relativa al corso di Ingegneria del Software: RAD_DDI_2.0

1.7 Panoramica

Al secondo punto del documento verrà presentato il sistema corrente.
Al terzo punto verrà presentata l'architettura del sistema proposto in cui gestiremo la decomposizione in sottosistemi, il mapping hardware/software, i dati persistenti, il controllo degli accessi e sicurezza, il controllo del flusso globale del sistema e le condizioni limite.
Al quarto punto verranno presentati i servizi dei sottosistemi.

2. Architettura del Sistema corrente

Il sistema che andremo a realizzare è ideato per ovviare alle criticità dell'architettura del Sistema corrente. Attualmente in modo del tutto autonomo, lo studente interessato è tenuto a compilare tutta la modulistica necessaria per la richiesta Erasmus, relativamente alle informazioni riguardanti le strutture ospitanti, passando poi dalla candidatura alla selezione ed infine all'eventuale accettazione. In seguito a questa prima fase, il candidato sarà tenuto a compilare il modulo "Learning Agreement" e provvedere al confronto tra gli esami dell'università ospitante e quelli della propria università, per poi inserire all'interno del documento quelli a lui idonei. In quest'ultima fase, lo scambio e-mail tra studente e docente non è di tipo lineare. La ricerca delle soluzioni per risolvere le criticità presenti nell'architettura del Sistema corrente è stata effettuata principalmente confrontando sistemi molto simili a quello da realizzare. Analizzando tali sistemi e più nello specifico la struttura che concerne l'organizzazione del "Learning Agreement":
La fase di "presentazione del Learning Agreement"…

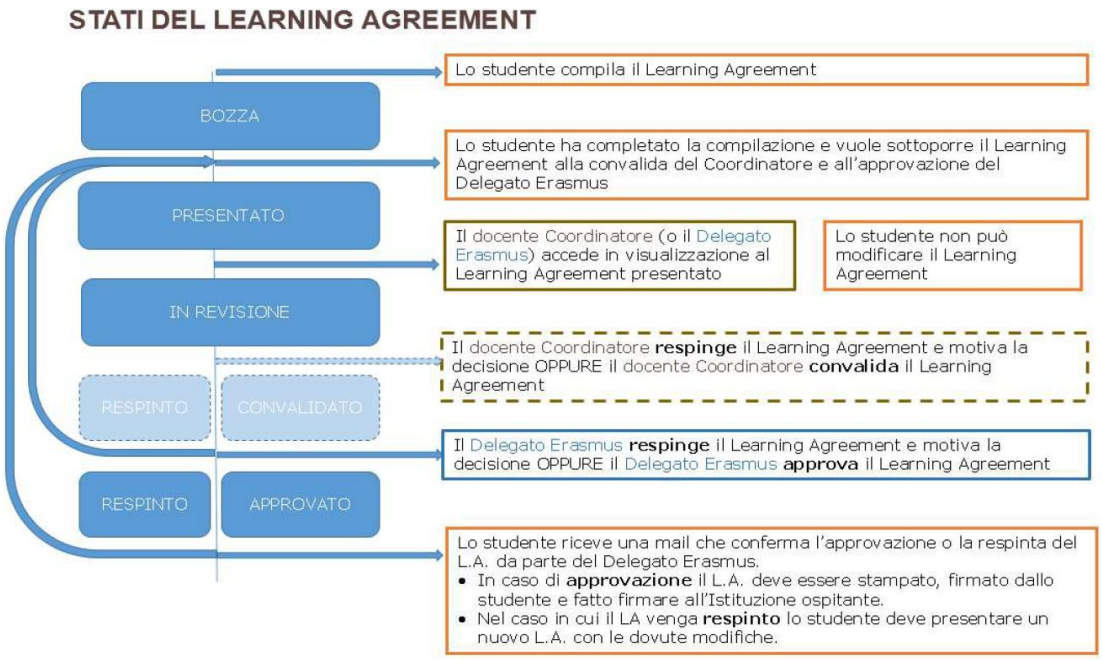
Prima di presentare il LA lo studente dovrà accertarsi di aver inserito i seguenti dati nella sezione NOTE

1. *se nel LA è presente l'ATTIVITA' DI RICERCA TESI, indicare il nome e cognome del DOCENTE RELATORE*
2. *se nel LA sono presenti attività di CREDITI LIBERI non presenti nel proprio libretto ed elencarle*
3. *se nel LA sono presenti attività SOVRANNUMERARIE elencarle in modo da garantire al docente che dovrà convalidare/approvare il LA che le attività non associate sono volutamente da considerarsi come "fuori piano" (queste attività sono ammesse marginalmente)*
4. *se nel LA sono presenti MODULI di un corso integrato che dovrà poi essere completato nell'università di partenza (queste attività sono ammesse marginalmente)*
5. *indicare il numero di mesi di mobilità*

Per presentare il LA si dovrà cliccare sul pulsante "Presenta il LA"

Lo stato del LA passerà da "BOZZA" a "PRESENTATO"

…e la fase di comprensione dei vari stati dello stesso,



-LA ACCETTATO

Lo studente riceverà una mail e dovrà provvedere a stampare il LA accettato, firmarlo e farlo firmare al responsabile della struttura ospitante.

-LA RIFIUTATO

Lo studente riceverà una mail e dovrà provvedere a modificare il LA creandone uno nuovo che andrà automaticamente a sostituire quello esistente, ma prima di questo è buona cosa leggere nella sezione "NOTE DOCENTE" il perché il LA è stato rifiutato.

-LA ACCETTATO E SI HA LA VOLONTA' DI MODIFICARLO DURANTE IL CORSO DELL'ERASMUS

Lo studente dovrà selezionare "Crea nuovo LA", a questo punto verrà mostrato il LA accettato precedentemente e si avrà la possibilità di creare nuove associazioni, eliminarle motivando la scelta e di presentare il LA modificato. Lo stato del LA passerà da "ACCETTATO" a "PRESENTATO" per poi seguire l'iter per la conferma.

Ci ha permesso di approfondire al meglio gli aspetti del dominio in questione, in modo da poterlo affrontare il più efficacemente possibile.

3. Architettura del Sistema proposto

3.1 Panoramica

Il sistema da noi proposto è un'applicazione web.

L'obiettivo è fornire uno strumento che permetta di agevolare lo studente e il docente durante tutto il percorso Erasmus e che permetta:

1. La sottomissione di domande in una sezione apposita
2. Una guida passo-passo che aiuti lo studente durante il suo percorso Erasmus
3. La comunicazione tra lo studente e il tutor con il quale poter scambiare messaggi e documenti.
4. La recensione delle esperienze Erasmus presso le università convenzionate
5. La possibilità di essere informato riguardo le ultime novità grazie ad una sezione apposita

Lo studente potrà inserire recensioni, domande e risposte, mentre l'inserimento delle news è compito dell'amministratore. Il sistema sarà di tipo 2-tier: il client quindi gestirà la parte di presentazione, e la parte di logica. La parte relativa ai dati sarà salvata in un database presente in cloud, evitando di utilizzare server dedicati.

Le funzionalità saranno divise in layer logici in base alle differenti funzionalità: View per la parte dell'interfaccia grafica, Model per la parte di gestione dei dati e Controller per la gestione del flusso del sistema.

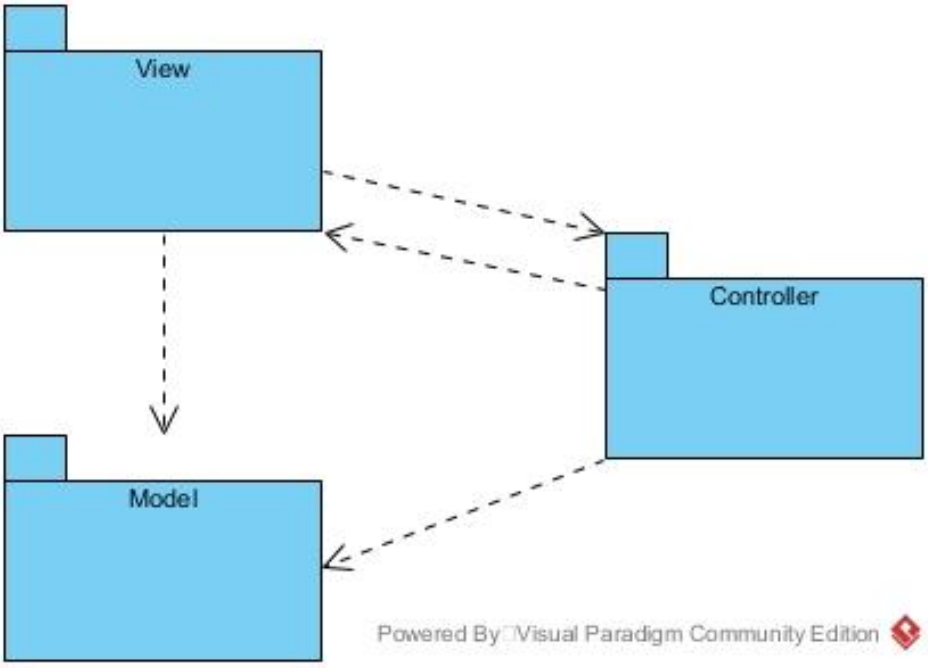
3.2 Decomposizione in sottosistemi

3.2.1 Decomposizione in layer

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

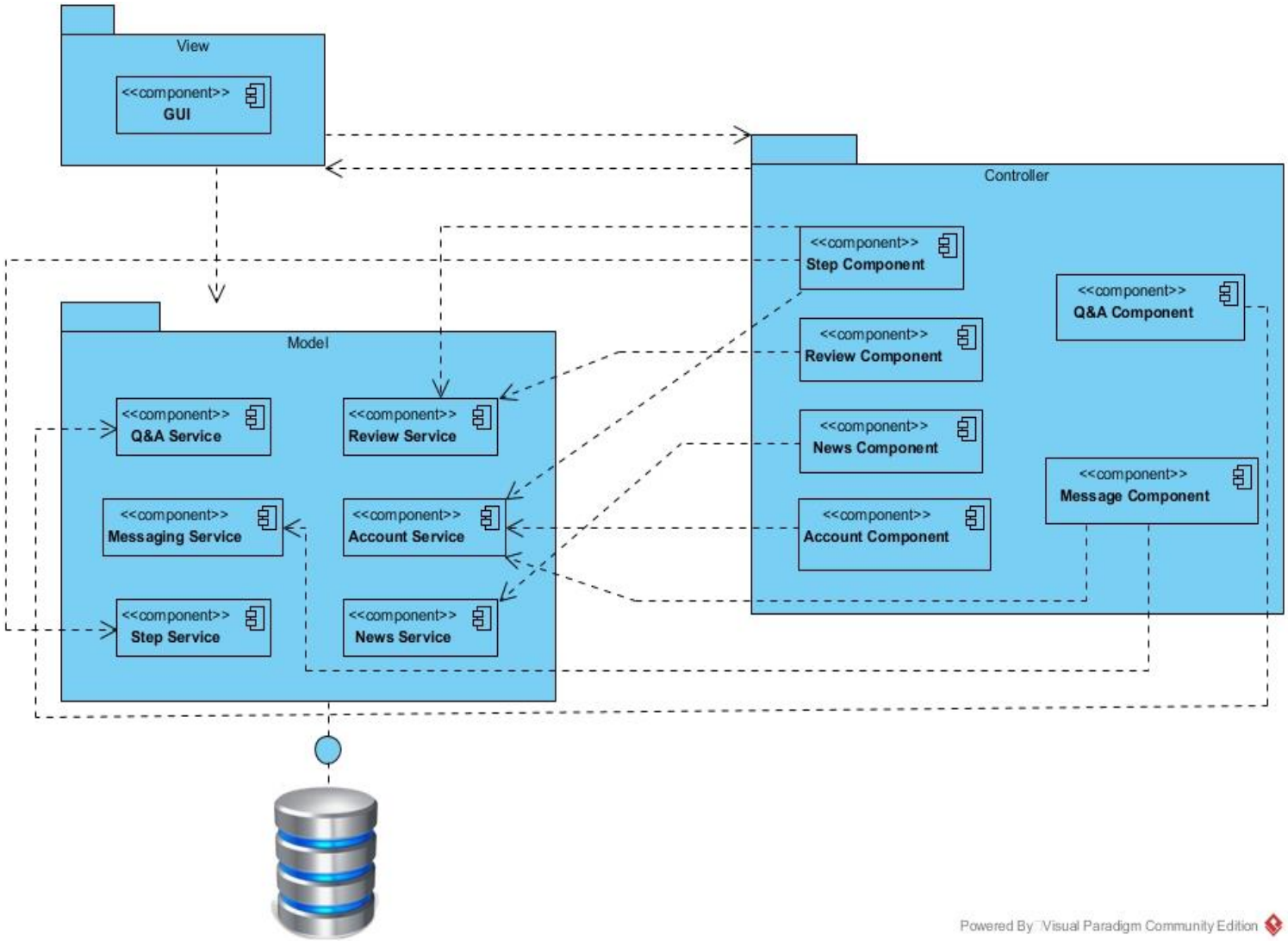
- **View**: raccoglie e gestisce l'interfaccia grafica e gli eventi generati dall'utente;
- **Model**: gestisce i dati e la logica che interagisce con essi.

- **Controller**: si occupa della gestione della logica del sistema;



Il modello MVC è stato scelto per non stravolgere l'architettura del framework utilizzato, e per essere coerenti con i Design Goal definiti. Questa scelta permette di aumentare la manutenibilità del codice, separando in maniera netta e definita i compiti che ogni parte del sistema deve svolgere.

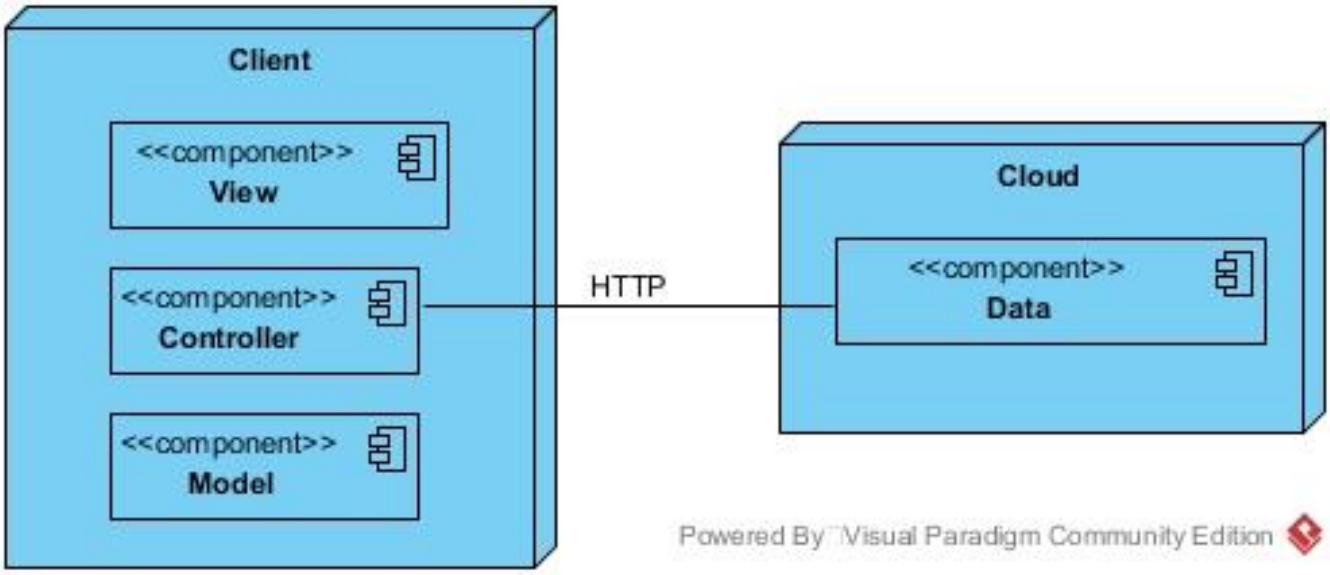
3.2.2 Decomposizione in Sottosistemi



3.3 Mapping hardware/software

Il sistema che verrà realizzato si basa su un'architettura Web-based.

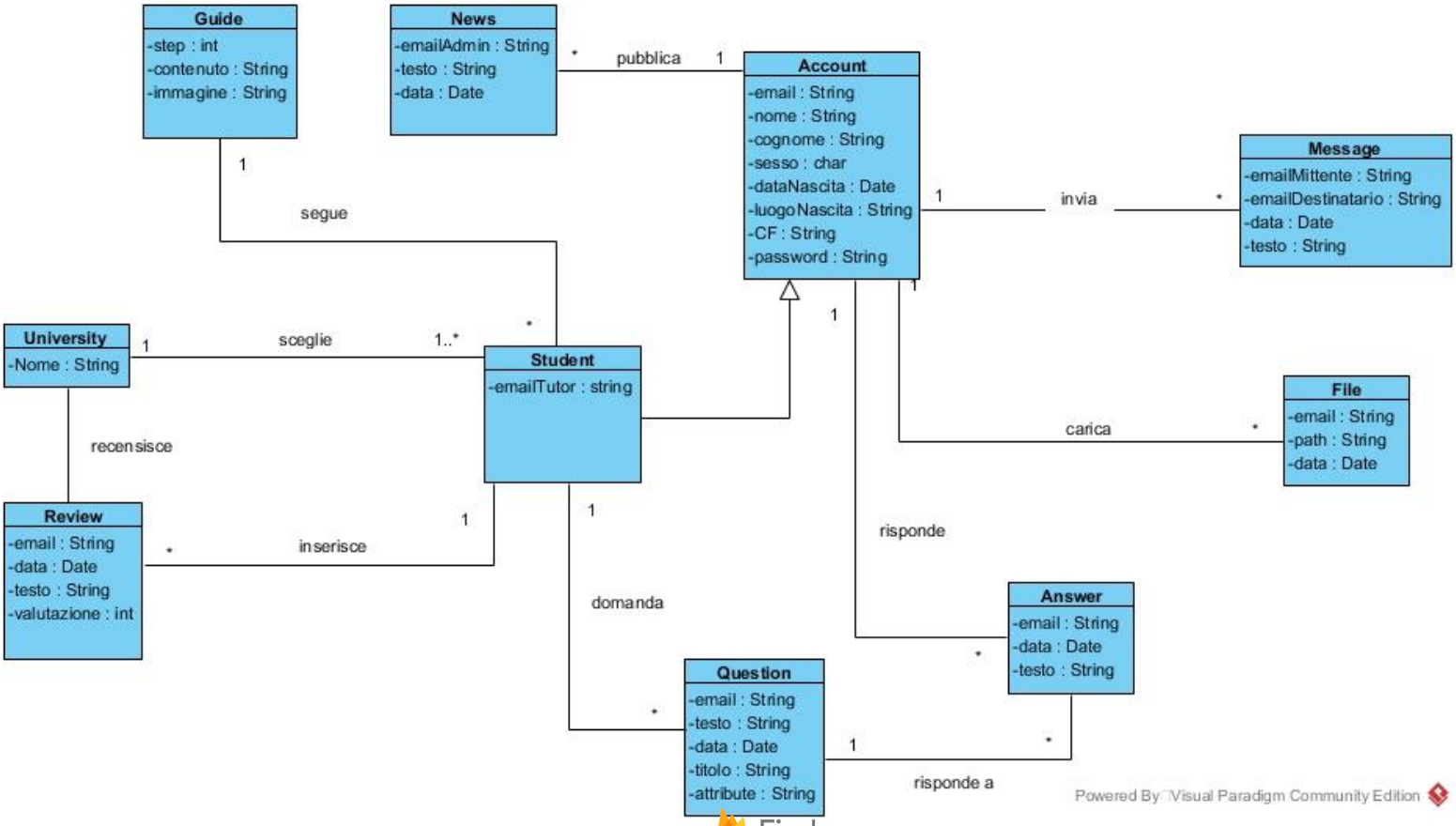
- **Protocollo richiesto**: HTTP
- **Memorizzazione dei dati**: Firebase
- **WebServer**: Firebase Hosting
- **Tecnologie utilizzate**: Typescript, CSS3, HTML5
- **Framework necessari:** Angular 5, Ionic 3



I dati e alcuni dei servizi utilizzati vengono gestiti da Firebase, rappresentante il Cloud.

La scelta fatta è la naturale conseguenza dell'architettura software scelta in precedenza, nel client ci sarà tutta la parte MVC del sistema, e tramite protocollo HTTP il client comunicherà con il cloud (Firebase) che fornirà molti dei servizi necessari, come l'hosting, il Database e lo storage.
Ovviamente questa scelta presenta il vantaggio principale di basare il sistema sui servizi offerti da Firebase, consentendo quindi di diminuire il lavoro necessario per la realizzazione di alcune funzionalità del sistema, ed evitando di allocare risorse e denaro per l'utilizzo di un server dedicato per questi servizi.

3.4 Gestione dati persistenti



Per la gestione dei dati persistenti utilizzeremo Firebase.

Firebase è un database non relazionale orientato ai documenti, scritti in JSON. I documenti sono divisi in collezioni e ogni documento è costituito da un insieme di coppie chiave-valore. Per accedere al database e operare su di esso utilizzeremo le API Google ufficiali.

3.5 Controllo degli accessi e sicurezza

Il controllo degli accessi è garantito tramite l'utilizzo di username (in questo caso l'username è l'email istituzionale dello studente, tutor o amministratore) e password, che verranno richieste per ogni singolo accesso.

Ogni tipo di utente potrà accedere al sito.

Il sistema prevede 4 figure di utenti: admin, tutor, student e guest (ospiti).

- L'admin può inserire una news.
- Il tutor può accettare gli studenti nelle fasi iniziali dello step-by-step, rispondere alle domande, visualizzare i file degli studenti, partecipare alla messaggistica con gli studenti, caricare i file.
- Gli studenti possono creare una domanda, rispondere ad una domanda, creare una recensione, valutare una recensione, utilizzare la messaggistica, consultare la guida, caricare i file.
- Gli utenti "guest" (non loggati o non registrati) potranno visualizzare le news, le recensioni, le valutazioni, creare un account ed eseguire l'accesso.

Ogni attore può accedere alle diverse funzionalità del sistema ma con diritti di accesso regolamentati sulla base delle differenti tipologie di utenza. Per documentare i diritti di accesso e per tenere traccia all'interno del sistema usufruiamo di una tabella di controllo di accesso che descrive le varie operazioni permesse agli attori sui diversi oggetti. Le colonne della matrice rappresentano gli attori del sistema mentre le righe rappresentano gli oggetti su cui sono regolamentati gli accessi.

Attori Oggetti	Admin	Tutor	Studenti	Guest
Inserire news	X			
Accettare gli studenti		X		
Rispondere a domande		X	X	
Visualizzare i file degli studenti affidati		X		
Visualizzare file personali		X	X	
Partecipare alla messaggistica		X	X	
Caricare i file		X	X	
Creare un account				X
Creare una domanda			X	
Valutare una recensione			X	
Creare una recensione			X	
Consultare la guida			X	
Visualizzare le news		X	X	X
Visualizzare le recensioni		X	X	X
Visualizzare le valutazioni		X	X	X

3.6 Controllo flusso globale del sistema

Il sistema ES 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 anche il meccanismo di controllo del flusso delle tecnologie da noi utilizzate (Angular).

Gli eventi sono quindi principalmente innescati dagli utenti mediante interfaccia grafica Web (CSS3, HTML5, TypeScript). I gestori degli eventi (event handlers, Angular) si attivano in risposta agli eventi esterni.

3.7 Condizione limite

3.7.1 Start-up

Per il primo start-up del sistema Erasmus Smart è necessario che il server si connetta tramite le Google's API al database di Firebase per la gestione dei dati persistenti. In seguito, tramite l'interfaccia di Login, sarà possibile autenticarsi tramite opportune credenziali (username e password) e accedere alle funzionalità permesse.
Una volta effettuato l'accesso, ES presenterà all'utente la home, dal quale si possono effettuare tutte le operazioni che il sistema fornisce.

3.7.2 Terminazione

Al momento della chiusura dell'applicativo si ha la terminazione del sistema con un regolare Logout dal sistema. Viene assicurata la consistenza dei dati, annullando eventuali operazioni che erano in esecuzione.

4. Servizi dei Sottosistemi

4.1 Model

Sottosistema	Q&A Service
Descrizione Sottosistema	Sottosistema che gestisce le operazioni relative ai servizi di Q&A.

Sottosistema	Review Service
Descrizione Sottosistema	Sottosistema che gestisce le operazioni relative ai servizi di recensione.

Sottosistema	Messaging Service
Descrizione Sottosistema	Sottosistema che gestisce le operazioni relative ai servizi di messaggistica.

Sottosistema	Account Service
Descrizione Sottosistema	Sottosistema che gestisce le operazioni relative al servizio di autenticazione e gestione del profilo personale.

Sottosistema	Step Service
Descrizione Sottosistema	Sottosistema che gestisce le operazioni relative alla guida Step by Step.

Sottosistema	News Service
Descrizione Sottosistema	Sottosistema che gestisce le operazioni relativi ai servizi di news.

4.2 View

Sottosistema	GUI
Descrizione Sottosistema	Sottosistema che gestisce l'interfaccia grafica di tutti i servizi.

4.3 Controller

Sottosistema	Step Component
Descrizione Sottosistema	Sottosistema che gestisce la logica delle operazioni relative alla guida Step by Step

Sottosistema	Review Component
Descrizione Sottosistema	Sottosistema che gestisce la logica delle operazioni relative ai servizi di recensione.

Sottosistema	News Component
Descrizione Sottosistema	Sottosistema che gestisce la logica delle operazioni relative ai servizi di news.

Sottosistema	Account Component
Descrizione Sottosistema	Sottosistema che gestisce la logica delle operazioni relative al servizio di autenticazione e gestione del profilo personale.

Sottosistema	Q&A Component
Descrizione Sottosistema	Sottosistema che gestisce la logica delle operazioni relative ai servizi di Q&A.

Sottosistema	Message Component
Descrizione Sottosistema	Sottosistema che gestisce la logica delle operazioni relative ai servizi di messagistica.