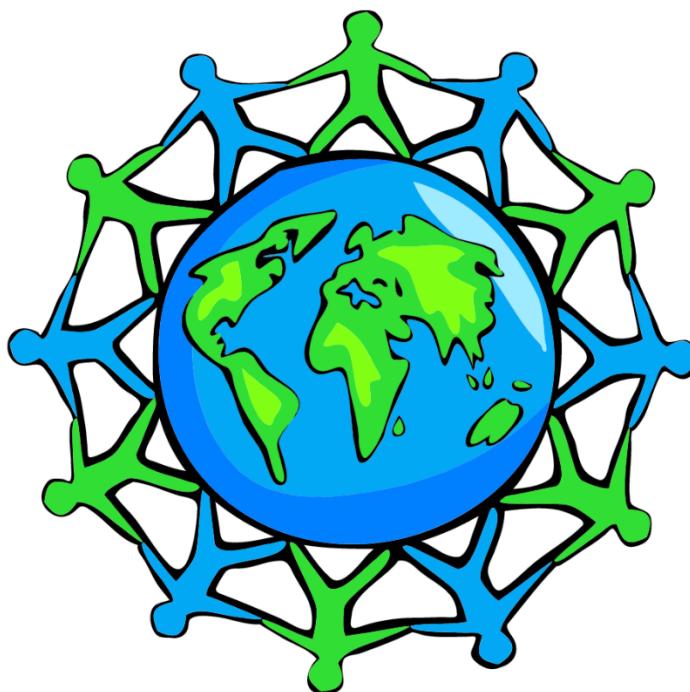




Prof. De Lucia Andrea

IMPACT ANALYSIS



Riferimento	
Versione	1.4
Data	30/05
Destinatario	Prof.re De Lucia
Presentato da	Salvatore Amideo, Alice Vidoni.



Laurea Magistrale in informatica - Università di Salerno

Corso di *Ingegneria, Gestione ed Evoluzione del Software*



Prof. De Lucia Andrea

Data	Versione	Descrizione	Autore
17/04/2021	1.0	Creazione del documento con intro al sistema esistente e riassunto nuove funzionalità	Alice Vidoni
29/04/2021	1.2	-Starting impact set -Candidate impact set	Alice Vidoni
05/05/2021	1.3	<u>-Studio di Fattibilità</u> <u>-Actual impact set</u>	Salvatore Amideo
06/05/2021	1.4	-Conclusioni	Alice Vidoni



Indice

1.	Scopo del documento	5
2.	Panoramica del sistema attuale	5
2.1	Attori.....	6
2.2	Design e implementazione e testing del sistema attuale.....	6
2.2.1	Implementazione.....	6
2.2.2	Testing	6
3.	Analisi della modifica e impact set	6
3.1	Starting Impact Set	7
	CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS	7
	CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE.....	7
3.2	Candidate Impact Set.....	8
	CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS	8
	CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE.....	8
4.	Studio di fattibilità	11
4.1	Identificazione, descrizione e valutazione dei costi.....	11
	CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS	11
	CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE.....	12
4.2	Identificazione, descrizione e valutazione dei benefici	12
	CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS	12
	CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE.....	12
4.3	Identificazione, descrizione e valutazione dei rischi.....	13
	CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS	13
	CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE.....	13
4.5	Risultati previsti	14
5.	Actual Impact set	14
	CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS	14
	Lato client-side	14
	Lato server-side	14
	Modifica MongoDB.....	14
	CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE.....	15
	Lato client-side	15
	Lato server-side	22
	Modifica MongoDB.....	37
6.	Conclusioni	39
	CHANGE REQUEST 01 – VISUALIZZA BANDO ERASMUS	39
	CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE.....	39
6.1	Precisione e Recall	39



Laurea Magistrale in informatica - Università di Salerno
Corso di *Ingegneria, Gestione ed Evoluzione del Software*

Prof. De Lucia Andrea



CHANGE REQUEST 01 – VISUALIZZA BANDO ERASMUS	39
CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE.....	39



1. Scopo del documento

In riferimento al documento “Change Request” verranno descritte tutte le nuove funzionalità implementate e verrà stilata un’analisi d’impatto dei cambiamenti sul sistema, evidenziando i rischi, i costi e i benefici.

2. Panoramica del sistema attuale

“EasyAgreement” è un’applicazione web rivolta agli studenti che partecipano al bando ERASMUS+ per Traineeship, volta a semplificare il processo di comunicazione tra studente, tutor accademico e tutor esterno in merito alla compilazione ed accettazione del modulo di Learning Agreement.

Le funzionalità offerte allo **studente**, oltre alle classiche funzionalità di accesso come login e signup, sono:

- Visualizzare la propria Home page;
- Compilare il Learning Agreement con supporto del tutor accademico;
- Caricare il Learning Agreement sul proprio profilo con la possibilità di visualizzare tutte le precedenti versioni;
- Comunicare con il tutor accademico e con il tutor esterno tramite un sistema di messaggistica istantanea;
- Rendere nota la modifica dello stato del Learning Agreement al tutor accademico ed esterno tramite un sistema di notifica;
- Visualizzare/Aggiornare/Rimuovere i propri documenti dal profilo.

Le funzionalità offerte al **tutor accademico**, oltre alle classiche funzionalità di accesso come login e signup, sono:

- Visualizzare la propria Home page;
- Assistere lo studente durante la compilazione del Learning Agreement;
- Disapprovare il Learning Agreement con conseguente notifica allo studente;
- Ricevere una notifica ad ogni cambio di stato del Learning Agreement e visualizzare tutte le precedenti versioni;
- Comunicare con lo studente tramite un meccanismo di messaggistica istantanea;
- Convalidare e firmare il Learning Agreement proposto dallo studente, direttamente sulla piattaforma.

Le funzionalità offerte al **tutor esterno**, oltre alla funzionalità di login, sono:

- Visualizzare la propria Home page;
- Visualizzare tutte le versioni precedenti del Learning Agreement per ogni studente;
- Disapprovare il Learning Agreement con conseguente notifica allo studente;
- Approvare il Learning Agreement tramite firma, con successiva notifica allo studente;
- Comunicare con lo studente tramite un meccanismo di messaggistica istantanea.



Prof. De Lucia Andrea

Le funzionalità offerte all'**amministratore**, oltre alla funzionalità di login, sono:

- Visualizzare la propria Home page;
- Inserire/Rimuovere dalla piattaforma le istruzioni e i tutor esterni disponibili.

2.1 Attori

Il Sistema attualmente distingue 4 attori: Studente, Tutor Esterno, Tutor Accademico e Amministratore.

2.2 Design e implementazione e testing del sistema attuale

Il sistema attuale, per garantire i servizi, utilizza un'architettura di tipo Client-Server. Il Client è rappresentato dal Browser utilizzato dall'utente. Lato Server è presente un ambiente run-time, open-source e multiplattforma Node.js, grazie al quale viene fornita la possibilità di eseguire il codice JavaScript lato server e produrre pagine web dinamiche prima che le stesse siano inviate al web browser dell'utente. È stato scelto Node.js poiché possiede un'architettura ad eventi capace di realizzare operazioni input/output (I/O) asincrone, questa scelta di design punta a ottimizzare la velocità di esecuzione e la scalabilità nell'applicazione web. Dal punto di vista di memorizzazione dei dati, utilizzando il formato JSON, la soluzione migliore è stata l'utilizzo di MongoDB, un database non relazionale (NoSQL).

2.2.1 Implementazione

L'implementazione del sistema prevede, lato client-side il pattern MVC ereditato dal framework utilizzato, lato server-side si necessita di un browser capace di interpretare il linguaggio JavaScript (al fine di permettere l'interazione con l'interfaccia del sistema).

2.2.2 Testing

Il sistema è stato sottoposto a:

- Testing di Unità (approccio black-box) in cui è stata testata ogni singola funzione degli oggetti creati.
- Testing di Integrazione (approccio bottom-up) in cui sono state testate le unità combinate come gruppo.
- Testing di Sistema per dimostrare che i requisiti commissionati dal cliente sono soddisfatti. Sono state testate le funzionalità usate più frequentemente dal cliente e quelle che presentavano maggiore possibilità di fallimento.

3. Analisi della modifica e impact set

Dal documento Master è possibile ricavare tutti gli artefatti impattati dalle modifiche proposte, e quindi i nuovi requisiti che dovranno essere implementati o modificati.



3.1 Starting Impact Set

Analizzando le change requests più approfonditamente (in particolare ciò che abbiamo ricavato dal Master) abbiamo individuato il nostro Starting Impact Set, che rappresenta l'insieme delle componenti che verranno direttamente impattate da queste modifiche. Alcune componenti sono state facilmente intuibili anche dal testo in chiaro delle change requests, altre meno poiché avevano collegamenti indiretti con le modifiche proposte. Nelle tabelle seguenti verranno indicate le componenti presenti all'interno del nostro Starting Impact Set, e l'impatto della modifica verrà valutato su quattro livelli.

- FORTE: nel caso in cui sia necessario riscrivere completamente l'artefatto.
- MEDIO: nel caso in cui sia necessario una rivisitazione e modifica di un artefatto.
- DEBOLE: nel caso in cui siano necessarie solo modifiche marginali.
- NULLO: nel caso che non sono presenti modifiche

CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS

Artefatto	Impatto	Descrizione
Requisiti funzionali	FORTE	Aggiunta dei requisiti: <ul style="list-style-type: none">• Visualizzazione bando Erasmus per l'anno corrente.
Casi d'uso	FORTE	Aggiunta casi d'uso: <ul style="list-style-type: none">• Visualizzazione bando Erasmus per l'anno corrente.
DB	NULLO	Non sono state apportate aggiunte/modifiche al database

CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE

Artefatto	Impatto	Descrizione
Requisiti funzionali	FORTE	Aggiunta dei requisiti: <ul style="list-style-type: none">• Visualizzazione Area Personale e Modifica Password per il nuovo attore introdotto.• Assegnamento dei punteggi, creazione e visualizzazione dei colloqui per il nuovo attore introdotto.• Inserimento, Rimozione e Visualizzazione dei Commissari per l'amministratore del sistema.• Visualizzazione graduatoria per lo studente.
Casi d'uso	FORTE	Aggiunta casi d'uso:



		<ul style="list-style-type: none">Assegnamento dei punteggi, creazione e visualizzazione dei colloqui per il nuovo attore introdotto.Inserimento, Rimozione e Visualizzazione dei Commissari per l'amministratore del sistema.Visualizzazione graduatoria per lo studente. <p>Modifica casi d'uso: Login, Logout, Visualizzazione Area Personale e Modifica Password, Ricezione Notifica e Generazione Notifica con l'aggiunta del nuovo attore.</p>
DB	FORTE	<p>Aggiunta del nuovo attributo “punteggio” per l'entità Studente. Aggiunta di due nuove entità a DB: Commissione Internazionale e Appuntamento</p>

3.2 Candidate Impact Set

In seguito a questa prima fase sono stati rivisti i Sequence Diagram, per valutare la dinamicità del sistema relativa ai requisiti coinvolti, e i Class Diagram relativi al nostro sistema. Questo lavoro è stato fatto per individuare le componenti impattate indirettamente da quelle prese in considerazione nel SIS (Starting Impact Set) descritto sopra.

Partendo dal nostro SIS, abbiamo analizzato più approfonditamente il class diagram, evidenziando le varie dipendenze per poi farne un'analisi ed estrapolare le componenti che faranno parte del nostro CIS (Candidate Impact Set).

CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS

Visualizza bando Erasmus dell'anno corrente

Di seguito prenderemo in considerazione l'aggiunta del requisito “Visualizza bando Erasmus dell'anno corrente”.

Classe	Impatto	N°
sidebar.ejs	DIRETTO	1
profile.ejs	INDIRETTO	1.1

Al temine di questa analisi siamo giunti alla conclusione che il Candidate Impact Set conterrà le seguenti componenti: |CIS|=2

CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE

Visualizzazione Area Personale



Prof. De Lucia Andrea

Di seguito prenderemo in considerazione l'aggiunta del requisito per la visualizzazione dell'area personale del nuovo attore introdotto.

Classe	Impatto	N°
profile.ejs	DIRETTO	1
commissioneInter.js	DIRETTO	2
sidebar.ejs	INDIRETTO	1.1
server.js	INDIRETTO	1.1.1
loginControl.js	INDIRETTO	1.1.1.1

Modifica Password

Di seguito prenderemo in considerazione l'aggiunta del requisito per la modifica della password del nuovo attore introdotto.

Classe	Impatto	N°
profile.ejs	DIRETTO	1
commissioneInterControll.js	DIRETTO	2
commissioneInter.js	DIRETTO	3
server.js	INDIRETTO	1.1

Aggiunta requisito “Inserimento Commissario”

Di seguito prenderemo in considerazione l'aggiunta del requisito per l'aggiunta del Commissario Internazionale.

Classe	Impatto	N°
admin/incomi.ejs	DIRETTO	1
commissioneInterControll.js	DIRETTO	2
commissioneInter.js	DIRETTO	3
sidebar.ejs	INDIRETTO	1.1
server.js	INDIRETTO	1.2

Aggiunta requisito “Rimozione Commissario”

Di seguito prenderemo in considerazione l'aggiunta del requisito rimozione Commissario Internazionale.

Classe	Impatto	N°
viewList.ejs	DIRETTO	1
commissioneInterControll.js	DIRETTO	2
commissioneInter.js	DIRETTO	3
server.js	INDIRETTO	1.1



sidebar.ejs	INDIRETTO	1.2
-------------	-----------	-----

Aggiunta requisito “Visualizzazione Commissione”

Di seguito prenderemo in considerazione l'aggiunta del requisito per la visualizzazione del Commissario Internazionale.

Classe	Impatto	N°
viewList.ejs	DIRETTO	1
commissioneInterControll.js	DIRETTO	2
commissioneInter.js	DIRETTO	3
server.js	INDIRETTO	1.1
sidebar.ejs	INDIRETTO	1.2

Aggiunta requisito “Aggiungere Colloquio”

Di seguito prenderemo in considerazione l'aggiunta del requisito per l'aggiunta di un colloquio.

Classe	Impatto	N°
gestioneColloquio.ejs	DIRETTO	1
appuntamentoControl.js	DIRETTO	2
appuntamento.js	DIRETTO	3
server.ejs	INDIRETTO	1.1
student.js	INDIRETTO	1.1.1.1
studentControl.js	INDIRETTO	1.1.1
sidebar.ejs	INDIRETTO	1.2

Aggiunta requisito “Visualizzazione Colloqui”

Di seguito prenderemo in considerazione l'aggiunta del requisito per l'aggiunta di un colloquio.

Classe	Impatto	N°
calendario.ejs	DIRETTO	1
appuntamentoControl.js	DIRETTO	2
appuntamento.js	DIRETTO	3
server.ejs	INDIRETTO	1.1
sidebar.ejs	INDIRETTO	1.2

Aggiunta requisito “Assegnare Punteggio”

Di seguito prenderemo in considerazione l'aggiunta del requisito per l'aggiunta di un colloquio.

Classe	Impatto	N°
--------	---------	----



gestioneGraduatoria.ejs	DIRETTO	1
punteggioControl.js	DIRETTO	2
punteggio.js	DIRETTO	3
server.ejs	INDIRETTO	1.1
sidebar.ejs	INDIRETTO	1.2

Aggiunta requisito “Visualizzazione Graduatoria”

Di seguito prenderemo in considerazione l'aggiunta del per l'aggiunta di un colloquio.

Classe	Impatto	N°
graduatoriaPunteggio.ejs	DIRETTO	1
punteggioControl.js	DIRETTO	2
punteggio.js	DIRETTO	3
server.ejs	INDIRETTO	1.1
sidebar.ejs	INDIRETTO	1.2

Al temine di questa analisi siamo giunti alla conclusione che il Candidate Impact Set conterrà le seguenti componenti: $|CIS|=18$

4. Studio di fattibilità

4.1 Identificazione, descrizione e valutazione dei costi

CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS

Identificatore	Valutazione	Motivazione
Integrazione dei requisiti funzionali	DEBOLE	L'aggiunta del requisito non comporta eccessive modifiche.
Modifica DB	NULLO	Non sono state apportate modifiche o aggiunte al database per l'implementazione di questo requisito.
Implementazione con Node.js e JavaScript	DEBOLE	Lo sviluppatore ha sufficienti conoscenze della tecnologia
Testing funzionale	DEBOLE	Avendo già effettuato il testing di tutte le componenti del



		sistema l'attività non sarà molto dispendiosa.
--	--	--

CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE

Identificatore	Valutazione	Motivazione
Integrazione dei requisiti funzionali	FORTE	L'aggiunta dei requisiti avverrà in modo incrementale.
Modifica DB	MEDIA	Verranno aggiunti 2 nuove entità nel db in questione, per rendere possibile l'aggiunta dei nuovi requisiti.
Implementazione con Node.js e JavaScript	DEBOLE	Lo sviluppatore ha sufficienti conoscenze della tecnologia
Testing funzionale	FORTE	Le componenti del sistema attuale non sono state testate, per cui l'attività di testing sarà molto dispendiosa.

4.2 Identificazione, descrizione e valutazione dei benefici

CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS

Identificatore	Valutazione	Motivazione
Facilità nella visualizzazione del bando	FORTE	Lo studente può finalmente visualizzare il bando dalla stessa piattaforma su cui fare la richiesta di partecipazione.

CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE

Identificatore	Valutazione	Motivazione
Facilità nella gestione dei colloqui	FORTE	La commissione può finalmente gestire i colloqui con maggior semplicità notificando immediatamente allo studente la data e l'ora del colloquio
Facilità nell'assegnazione dei punteggi	FORTE	La commissione può finalmente assegnare il punteggio ottenuto dallo



		studente dalla stessa piattaforma dove gestisce il colloquio
Facilità nella visualizzazione della graduatoria	FORTE	Lo studente può finalmente visualizzare il la graduatoria dalla stessa piattaforma senza aspettare che venga pubblicata sul sito dell'università

4.3 Identificazione, descrizione e valutazione dei rischi

CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS

Identificatore	Valutazione	Motivazione
Introduzione di nuovi fault	DEBOLE	L'attività di testing da effettuare potrà avere una bassa probabilità di riscontrare fault, questo è dovuto al fatto che il sistema è già stato sottoposto ad una fase di testing.
Carico di lavoro eccessivo per il client	DEBOLE	Il client dovrà solo elaborare dei dati provenienti da richieste a servizi forniti dal server di Unisa.
Carico di lavoro eccessivo per il server	NULLO	Non vengono fatte richieste al server per l'implementazione di questo requisito.

CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE

Identificatore	Valutazione	Motivazione
Introduzione di nuovi fault	MEDIO	L'attività di testing da effettuare potrà avere una media probabilità di riscontrare fault, questo è dovuto al fatto che il sistema è già stato sottoposto ad una fase di testing.



Carico di lavoro eccessivo per il client	MEDIO	Il client dovrà elaborare dei dati provenienti dall'inserimento dell'utente tramite la piattaforma
Carico di lavoro eccessivo per il server	MEDIO	Al server saranno attuate lievi modifiche e la maggior parte delle azioni saranno eseguite utilizzando i metodi già esistenti che effettuano operazioni CRUD.

4.5 Risultati previsti

La realizzazione delle modifiche terrà conto di tale studio di fattibilità.

5. Actual Impact set

Di seguito saranno elencate le componenti realmente modificate, tali componenti faranno parte del nostro AIS.

CHANGE REQUEST 01 – VISUALIZZAZIONE BANDO ERASMUS

Lato client-side

Visualizza bando Erasmus per l'anno corrente

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **sidebar.ejs** inserimento del nuovo bottone per visualizzare il bando

```
<li class="nav-item">
  <a href='https://web.unisa.it/uploads/rescue/164/4613/bando-erasmus-2021-2022.pdf'>
    <i class="fas fa-file-alt"></i>
    <p>Visualizza Bando</p>
  </a>
</li>
```

Lato server-side

Visualizza bando Erasmus per l'anno corrente

//nulla

Modifica MongoDB

//nulla



Prof. De Lucia Andrea

mostriamo la tabella delle componenti che fanno parte del nostro Actual impact set, e ne evidenzieremo quelle che fanno parte del DIS

Classe	Impatto	Azione
sidebar.ejs	DEBOLE-DIRETTO	Update

CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE

Lato client-side

Visualizzazione Area Personale

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- profile.ejs

```
<%if(session.utente.type == "commission"){%>
<div class="page-inner py-5" id="CommissionView" >
<h1 class="mt-8">Riepilogo Dati Anagrafici</h1>
<br>
<div class="text-center" id="viewp">
  <ul>
    <li class="list-group-item text-left"><span class="pull-left"><strong>Nome:&nbsp;</strong></span> <%= session.utente.utente.name %></li>
    <li class="list-group-item text-left"><span class="pull-left"><strong>Cognome:&nbsp;</strong></span> <%= session.utente.utente.surname %></li>
    <li class="list-group-item text-left"><span class="pull-left"><strong>Email:&nbsp;</strong></span> <%= session.utente.utente.email %></li>
  </ul>
  <button class = "btn btn-secondary btn-round" onclick = "showEdit()">Modifica</button>
<div class="updatedati commission text-center">
```

Modifica Password

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- profile.ejs

```
<form id="formUpdatePass" method="post" action="/updatePassword">
  <h4 class="mt-4">Modifica password</h4>
  <div class="form-label-groupAT">
    <input type="password" id="inputOldPassword" class="form-control" name="inputOldPassword" placeholder="Vecchia Password" >
    <span id="errOldPassword">Vecchia Password errata</span>
  </div>
  <div class="form-label-groupAT">
    <input type="password" id="inputPassword" class="form-control" name="inputPassword" placeholder="Nuova Password" >
    <span id="errPassword">Nuova Password non valida</span>
  </div>
  <div class="form-label-groupAT">
    <input type="password" id="inputConfirmPassword" class="form-control" name="inputConfirmPassword" placeholder="Ripeti Password" >
    <span id="errConfirmPassword">Password errata </span>
  </div>
  <button class="btn btn-secondary btn-round form-group" type="submit" class="saveButton">Modifica password</button>
</form>
</div>
</div>
<% } %>
```

Aggiunta requisito “Inserimento Commissario”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:



- admin/incomi.ejs

```
<% include ('../header') %>
<% include ('../sidebar') %>

<link href="/css/simple-sidebar.css" rel="stylesheet">
<script type="text/javascript" src="../js/validation.js"></script>
<script src="https://unpkg.com/sweetalert/dist/sweetalert.min.js"></script>
<script type="text/javascript" src="/js/cookie.js"></script>



<div class="content">
        <div class="panel-header bg-primary-gradient">
            <div class="page-inner py-5">
                <div class="d-flex align-items-left align-items-md-center flex-column flex-md-row">
                    <div>
                        <h2 class="text-white pb-2 fw-bold">Inserimento</h2>
                        <h5 class="text-white op-7 mb-2">Commissione Internazionale</h5>
                    </div>
                </div>
            </div>
        <div class="page-inner py-5 text-center" >
            
            <h4 style="text-align: justify; font-style: italic; color: darkblue;">Erasmus+</h4><br>
            <hr>
            <form style="width: 50%; margin: auto;" id="formCompile" method="post" action="/addCommiInter"><br>
                <div class="form-label-group">
                    <input type="text" id="inputNameCI" class="form-control" name="inputNameCI" placeholder="Nome" required >
                    <label for="inputNameCI">Nome<label>
                </div>
                <div class="form-label-group">
                    <input type="text" id="inputSurnameCI" class="form-control" name="inputSurnameCI" placeholder="Cognome" required >
                    <label for="inputSurnameCI">Cognome<label>
                </div>
                <div class="form-label-group">
                    <input type="text" id="inputEmailCI" class="form-control" name="inputEmailCI" placeholder="E-Mail" required >
                    <label for="inputEmailCI">E-Mail<label>
                </div>
            </form>
        </div>
    </div>


```

```
</div>
<div class="form-label-group">
    <input type="password" id="inputPassword" class="form-control" name="inputPassword" placeholder="Password" required >
    <label for="inputPassword">Password</label>
    <span id="errPassword">Password non valida</span>
</div>
<div class="form-label-group">
    <input type="password" id="inputRePassword" class="form-control" name="inputRePassword" placeholder="Ripeti Password" required >
    <label for="inputRePassword"> Ripeti Password</label>
    <span id="errConfirmPassword">La password non corrisponde</span>
</div>
<hr>
<button class="btn btn-secondary btn-round form-group" type="submit" class="saveButton" id="send">Aggiungi</button>
</form>
<!-- /#page-content-wrapper -->
</div>

<!-- Menu Toggle Script -->
<script>
    $("#menu-toggle").click(function(e) {
        e.preventDefault();
        $("#wrapper").toggleClass("toggled");
    });
</script>

<% include ('../footer') %>
```

Aggiunta requisito “Rimozione Commissario”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **viewList.ejs**

```
<%-- include ('header') %>
<%-- include ('sidebar') %>
<link href="/css/simple-sidebar.css" rel="stylesheet">
<script type="text/javascript" src="/js/cookie.js"></script>
<link rel="stylesheet" href="/css/atlantis.min.css">
<link rel="stylesheet" href="/css/list.css">



<div class="content">
        <div class="panel-header bg-primary-gradient">
            <div class="page-inner py-5">
                <div class="d-flex align-items-left align-items-md-center flex-column flex-md-row">
                    <div>
                        <h2 class="text-white pb-2 fw-bold">Visualizzazione Liste</h2>
                        <h5 class="text-white op-7 mb-2">Seleziona il tipo di utente da visualizzare</h5>
                    </div>
                </div>
            </div>
        <div class="page-inner py-5">
            
            <h4 style="text-align: justify; font-style: italic; color: #darkblue;">Erasmus+</h4><br>
            <hr>
            <br>
            <div class="form-group">
                <div class="list_version">
                    <label>Scegli la lista di utenti da visualizzare: </label>
                </div>
            </div>
        </div>
    </div>


```

```
else if(session.utente.type=="admin"){%
    <select class="custom-select mr-sm-2" name="inputType" id="tipo">
        <option selected id="selected" value="">--Seleziona--</option>
        <option id="externalTutor" value="externalTutor">Tutor Esterni</option>
        <option id="hostOrganization" value="host">Organizzazioni Ospitanti</option>
        <option id="academicTutor" value="academicTutor">Tutor Accademici</option>
        <option id="commissioneInter" value="commissioneInter">Commissione Internazionale</option>
    </select>
%}>

function appendCIList(commissione){
    $('.elements').append([
        '<div class="card">',
        '  <div class="card__image"></div>',
        '  <div class="card__copy">',
        '    <h3>' + commissione.name + ' ' + commissione.surname + '</h3>',
        '    <h4 class="toPass">' + commissione.email + '</h4>',
    ]);
}
```

Aggiunta requisito “Visualizzazione Commissione”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **viewList.ejs**

```
'<button class="btn btn-secondary btn-round form-group" class="deleteUserButton" id="delUser" onclick="clickDelete(this)">Elimina utente</button>',
  '<%}>',
  '/div>',
  '>'
```

Aggiunta requisito “Aggiungere Colloquio”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **gestioneColloquio.ejs**

```
<!-- include ('header') -->
<%-- include ('sidebar') %>
<link href="/css/simple-sidebar.css" rel="stylesheet">
<script type="text/javascript" src="/js/cookie.js"></script>
<link rel="stylesheet" href="/css/atlantis.min.css">
<link rel="stylesheet" href="/css/list.css">

<div class="main-panel">
  <div class="content">
    <div class="panel-header bg-primary-gradient">
      <div class="page-inner py-5">
        <div class="d-flex align-items-left align-items-md-center flex-column flex-md-row">
          <div>
            <h2 class="text-white pb-2 fw-bold">Gestione Colloqui</h2>
            <h5 class="text-white op-7 mb-2">Seleziona lo studente per organizzare il colloquio</h5>
          </div>
        </div>
      </div>
    </div>
    <div class="page-inner py-5">
      
      <h4 style="text-align: justify; font-style: italic; color: #darkblue;">Erasmus+</h4><br>
      <hr>
      <br>
      <div class="form-group">
        <div class="list_version">
        </div>
      </div>
      <div class="elements"></div>
    </div>
  </div>
<script>
```

```
function appendStudentList(student){
  $('.elements').append([
    '<div class="card">',
    '  <div class="card__image"></div>',
    '  <div class="card__copy">',
    '    <h3>' + student.Name + ' ' + student.Surname + '</h3>',
    '    <h4 class="toPass">' + student.StudentID + '</h4>',
    '    <h4 class="toPass">' + student.Email + '</h4>',
    '    <form method="POST" action="/insertAppointment?studentEmail=' + student.Email + '&studentID=' + student.StudentID + '">',
    '      <input type="text" id="inputDate" class="form-control" name="inputDate" placeholder="Inserisci la data del colloquio (YYYY-MM-DD)" required>',
    '      <input type="text" id="inputSchedule" class="form-control" name="inputSchedule" placeholder="Inserisci l\'orario del colloquio (HH:MM)" required>',
    '      <button class="btn btn-secondary btn-round form-group" class="infoAppuntamentoButton" type="submit">Organizza Colloquio</button>',
    '    </form>',
    '  </div>',
    '</div>'].join('\n'))
}
```

Aggiunta requisito “Visualizzazione Colloqui”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:



- **calendario.ejs**

```
<%- include ('header') %>
<%- include ('sidebar') %>
<link href='https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/4.2.0/core/main.min.css' rel='stylesheet' />
<link href='https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/4.2.0/daygrid/main.min.css' rel='stylesheet' />
<link href='https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/4.2.0/timegrid/main.min.css' rel='stylesheet' />
<link href='https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/4.2.0/list/main.min.css' rel='stylesheet' />
<link href='fullcalendar/main.css' rel='stylesheet' />

<style>
    html, body {
        font-size: 14px;
        background: #e2e2e2;
    }
    #calendar{
        margin-top: -44px;
        margin-bottom: -42px;
        width: 104%;
        margin-left: -28px;
        box-shadow: 0px 0px 10px #000;
        padding: 15px;
        background: #fff;
    }
</style>


<div class="content">
        <div class="panel-header bg-primary-gradient">
            <div class="page-inner py-5">
                <div class="d-flex align-items-left align-items-md-center flex-column flex-md-row">
                    <div>
                        <h2 class="text-white pb-2 fw-bold">Calendario Colloqui</h2>
                    </div>
                </div>
            </div>
            <div class="page-inner py-5">
                <div id='calendar'></div>
            </div>
        </div>
    </div>


```

```
<script src='https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/4.2.0/core/main.min.js'></script>
<script src='https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/4.2.0/interaction/main.min.js'></script>
<script src='https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/4.2.0/daygrid/main.min.js'></script>
<script src='https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/4.2.0/timegrid/main.min.js'></script>
<script src='https://cdnjs.cloudflare.com/ajax/libs/fullcalendar/4.2.0/list/main.min.js'></script>
<script>

    document.addEventListener('DOMContentLoaded', function() {
        var calendarEl = document.getElementById('calendar');

        var calendar = new FullCalendar.Calendar(calendarEl, {
            plugins: [ 'interaction', 'dayGrid', 'timeGrid', 'list' ],
            events: {
                url: '/getallAppuntamenti',
                cache: true
            }
        });

        calendar.render();
    });
</script>
<%- include ('footer') %>
```



Aggiunta requisito “Assegnare Punteggio”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **gestioneGraduatoria.ejs**

```
<%-- include ('header') %>
<%-- include ('sidebar') %>
<link href="/css/simple-sidebar.css" rel="stylesheet">
<script type="text/javascript" src="/js/cookie.js"></script>
<link rel="stylesheet" href="/css/atlantis.min.css">
<link rel="stylesheet" href="/css/list.css">

<div class="main-panel">
    <div class="content">
        <div class="panel-header bg-primary-gradient">
            <div class="page-inner py-5">
                <div class="d-flex align-items-left align-items-md-center flex-column flex-md-row">
                    <div>
                        <h2 class="text-white pb-2 fw-bold">Gestione Graduatoria</h2>
                        <h5 class="text-white op-7 mb-2">Seleziona lo studente per assegnare un punteggio</h5>
                    </div>
                </div>
            </div>
        <div class="page-inner py-5">
            
            <h4 style="text-align: justify; font-style: italic; color: #darkblue;">Erasmus+</h4><br>
            <hr>
            <br>
            <div class="form-group">
                <div class="list_version">
                </div>
            </div>
            <div class="elements"></div>
        </div>
    </div>
<script>
```



Prof. De Lucia Andrea

```
function appendStudentList(student){
    $('.elements').append([
        '<div class="card">',
        '  <div class="card__image"></div>',
        '  <div class="card__copy">',
        '    <h3>' + student.Name + ' ' + student.Surname + '</h3>',
        '    <h4>' + toPass + '<h4>' + student.StudentID + '</h4>',
        '    <form method="POST" action="/setPunteggioGrad?email=' + student.Email + '">',
        '      <input type="text" id="inputPunteggio" class="form-control" name="inputPunteggio" placeholder="Inserisci il punteggio ottenuto" required>',
        '      <button class="btn btn-secondary btn-round form-group" class="setPunteggio" type="submit">Assegna Punteggio</button>',
        '    </form>',
        '</div>',
    ]).join('\n')
}

function appendNothing(){
    $('.elements').append(['Nessun utente trovato'].join('\n'))
}

$(document).ready(function() {
    $.post( "/getUserList", {type: "student"}).done(function(result){
        var i;
        $('.elements').append('<section class="cards clearfix">');
        for(i=0; result[i] != null; i++){
            if(i&1){
                if((i%3) == 0){
                    $('.elements').append('</section>');
                }
            }
            if(result[i].Punteggio == null){
                appendStudentList(result[i]);
            }
            else {
                $('.elements').append('</section>');
            }
        }
    })
});
```

Aggiunta requisito “Visualizzazione Graduatoria”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **graduatoriaPunteggio.ejs**



```
<%- include ('header') %>
<%- include ('sidebar') %>
<link href="/css/simple-sidebar.css" rel="stylesheet">
<script type="text/javascript" src="/js/cookie.js"></script>
<link rel="stylesheet" href="/css/atlantis.min.css">
<link rel="stylesheet" href="/css/list.css">

<div class="main-panel">
  <div class="content">
    <div class="panel-header bg-primary-gradient">
      <div class="page-inner py-5">
        <div class="d-flex align-items-left align-items-md-center flex-column flex-md-row">
          <div>
            <h2 class="text-white pb-2 fw-bold">Graduatoria</h2>
          </div>
        </div>
      </div>
    </div>
    <table id="my_table" class="table table-striped">
      <tbody>
        <tr>
          <th scope="col">#</th>
          <th scope="col">Nome</th>
          <th scope="col">Cognome</th>
          <th scope="col">Punteggio</th>
        </tr>
      </tbody>
    </table>
  </div>
  <script>
```

```
$(document).ready(function() {
  $.get( "/getGraduatoria", function(data) {
    var i=1;
    $.each(data, function(key, val) {
      var pos = '<td>' + i++ + '</td>';
      var tdStudenteNome = '<td>' + val.Name+ '</td>';
      var tdStudenteCognome = '<td>' + val.Surname+ '</td>';
      if(val.Punteggio == null){
        var tdPunteggio = '<td>' + "In Attesa"+ '</td>';
      }else {
        var tdPunteggio = '<td>' + val.Punteggio+ '</td>';
      }
      $('#my_table').append('<tr>' +pos+ tdStudenteNome+ tdStudenteCognome + tdPunteggio + '</tr>')
    })
  })
}

</script>
- include ('footer') %>
```

Lato server-side Login



Prof. De Lucia Andrea

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **loginControl.js**

```
if (resultAd == null) {
    var checkCommission = commissionModel.findByEmail(username)
    checkCommission.then(function (resultComm) {
        if (resultComm == null) {
            res.cookie('errLogin', '1')
            resolve(false)
        } else {
            if (hash.checkPassword(resultComm.getPassword().hash, resultComm.getPassword().salt, password)) {
                var commissionSession = {
                    utente: resultComm,
                    type: 'commission'
                }
                res.cookie('logEff', '1')
                resolve(commissionSession)
            } else {
                res.cookie('errLogin', '1')
                resolve(false)
            }
        }
    })
}
```

- **commissioneInter.js**

```
/**
 * Find commission by email
 * @param {String} email- email of commision
 * @returns {boolean} - return true if the object does not exist in database, else false
 */
static findByEmail (email) {
    return new Promise(function (resolve, reject) {
        MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, function (err, db) {
            if (err) reject(err)
            var dbo = db.db(dbName)
            dbo.collection('Commission').findOne({ email: email }, function (err, result) {
                if (err) reject(err)
                if (result != null) {
                    var comm = new Commission()
                    comm.setEmail(result.email)
                    comm.setName(result.name)
                    comm.setSurname(result.surname)
                    comm.setPassword(result.Password)
                    resolve(comm)
                } else {
                    resolve(null)
                }
                db.close()
            })
        })
    })
}
```

Generazione notifica

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **appuntamentoControl.js**

```
inserted.then(function (result) {
    var d = new Date()
    var date = { hour: d.getHours().toString().padStart(2, 0), minutes: d.getMinutes().toString().padStart(2, 0), seconds: d.getSeconds().toString().padStart(2, 0), day: d.getDate() }
    socket.emit('send-notification', { associatedID: email, text: { title: 'Nuovo Colloquio', text: 'Il Commissario Internazionale ha fissato il colloquio per il giorno ' + date.day } })
})
```

Modifica Password

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **server.js**

```
app.post('/updatePassword', function (req, res) {
    if (req.session.utente != null) {
        if (req.session.utente.type == 'commission') {
            var updateC = commissionControl.update(req, res)
            updateC.then(function (result) {
                if (result == true) { res.render('profile') } else { res.render('profile') }
            })
        }
    }
})
```

- **commissioneInterControl.js**

```
exports.update = function (req, res) {
    return new Promise(function (resolve, reject) {
        var oldPassword = req.body.inputOldPassword
        var password = req.body.inputPassword
        var passwordConfirm = req.body.inputConfirmPassword
        var isRight = true

        if ((oldPassword == null) || (oldPassword.length <= 7) || (!/^[A-Za-z0-9]+$/ .test(oldPassword))) {
            res.cookie('errOldPassword', '1')
            isRight = false
        }

        if ((password == null) || (password.length <= 7) || (!/^[A-Za-z0-9]+$/ .test(password))) {
            res.cookie('errPassword', '1')
            isRight = false
        }

        if (passwordConfirm != password) {
            res.cookie('errPasswordConfirm', '1')
            isRight = false
        }

        if (!isRight) {
            resolve(false)
            return
        }

        if (hash.checkPassword(req.session.utente.utente.password.hash, req.session.utente.utente.password.salt, oldPassword)) {
            var passwordHashed = hash.hashPassword(password)
            var checkPass = commissioneModel.updatePassword(passwordHashed, req.session.utente.utente.email)
            /**
             * It checks the result of updatePassword function and updates the Commission session
             * @param {Object} result - The result of updatePassword function (about Commission)
             * @returns {Boolean} - It returns true and generates an "edit password complete" cookie if result != null, else it returns a reject
             */
        }
    })
}
```



```
        ,
        checkPass.then(function (result) {
          if (result != null) {
            req.session.utente.utente = { name: result.getName(), email: result.getEmail(), surname: result.getSurname(), Password: result.getPassword() }
            res.cookie('updatePassEff', '1')
            resolve(true)
          } else { resolve() }
        })
      } else {
        res.cookie('errOldPassword', '1')
        resolve(false)
      }
    })
  }
}
```

- **commissioneInter.js**

```
/** 
 * This method update the commission password
 * @param {String} password - the new commission password
 * @param {String} emailv - commission email
 * @returns {Object} - It returns the updated commission if result != null, else it returns null
 */
static updatePassword (pass, emailv) {
  return new Promise(function (resolve, reject) {
    MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, function (err, db) {
      if (err) reject(err)
      var dbo = db.db(dbName)
      var myquery = { email: emailv }
      var newvalues = { $set: { Password: pass } }
      dbo.collection('Commission').updateOne(myquery, newvalues, function (err, res) {
        if (err) reject(err)
      })
      dbo.collection('Commission').findOne({ email: emailv }, function (err, result) {
        if (err) reject(err)
        if (result != null) {
          var comm = new Commission()
          comm.setEmail(result.email)
          comm.setName(result.name)
          comm.setSurname(result.surname)
          comm.setPassword(result.Password)
          resolve(comm)
        } else {
          resolve(null)
        }
        db.close()
      })
    })
  })
}
```

Aggiunta requisito “Inserimento Commissario”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **server.js**



```
app.post('/addCommiInter', function (req, res) {
  if (req.session.utente != null) {
    if (req.session.utente.type == 'admin') {
      var administratorAddCommiss = commissionControl.addCommisInter(req, res)
      administratorAddCommiss.then(function (result) {
        if (result) {
          res.cookie('insertCommiIntern', '1')
          res.redirect('/addCommissioneInternazionale')
        } else {
          res.cookie('alreadyCommiInter', '1')
          res.redirect('/addCommissioneInternazionale')
        }
      })
    } else {
      res.cookie('onlyForAdmin', '1')
      res.render('index')
    }
  } else {
    res.cookie('cannotAccess', '1')
    res.redirect('/')
  }
})
```

- **commissioneInterControl.js**

```
/**
 * This method inserts an commissione internazionale
 * @param {Object} req - The HTTP req
 * @param {Object} res - The HTTP res
 * @returns {Boolean} - This method returns true if the insert of commissione internazionale was successfull, else false
 */
exports.addCommisInter = function (req, res) {
  return new Promise(function (resolve, reject) {
    var name = req.body.inputNameCI
    var surname = req.body.inputSurnameCI
    var email = req.body.inputEmailCI
    var password = req.body.inputPassword
    var repassword = req.body.inputRePassword

    var isRight = true
    if (password != repassword) {
      isRight = false
    }

    isRight = true
    if ((name == null) || (name.length <= 1) || (!/^([A-Za-z]+$/).test(name))) {
      res.cookie('errComInterName', '1')
      isRight = false
    }

    if ((surname == null) || (surname.length <= 1) || (!/^([A-Za-z]+$/).test(surname))) {
      res.cookie('errComInterSurname', '1')
      isRight = false
    }

    if ((email == null) || (email.length <= 6) || (!/[a-zA-Z0-9!#$%&*+=?^_`{|}~-]+(%[a-zA-Z0-9!#$%&*+=?^_`{|}~-]{0,2})+@[a-zA-Z0-9](%[a-zA-Z0-9-]{0,2})+.[a-zA-Z]{2,4}/.test(email))) {
      res.cookie('errComInterEmail', '1')
      isRight = false
    }

    if ((password == null) || (password.length <= 7) || (!/^([A-Za-z0-9]+$/).test(password))) {
      res.cookie('errPassword', '1')
      isRight = false
    }
  })
}
```



```
if (repassword != password) {
    res.cookie('errPasswordConfirm', '1')
    isRight = false
}

if (!isRight) {
    resolve(false)
    return
}

// hashing e salt of password
var passwordHashed = hash.hashPassword(password)

// Create external tutor object
var commission = new commissioneModel()
commission.setSurname(surname)
commission.setName(name)
commission.setEmail(email)
commission.setPassword(passwordHashed)

commissioneModel.addCommiInter(commission)
resolve(true)
})
}
}
```

- **commissioneIter.js**

```
static addCommiInter (CommiInter) {
    return new Promise(function (resolve, reject) {
        MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, function (err, db) {
            if (err) reject(err)
            var dbo = db.db(dbName)
            dbo.collection('Commission').insertOne(CommiInter, function (err) {
                if (err) throw err
                resolve()
                db.close()
            })
        })
    })
}
```



Aggiunta requisito “Rimozione Commissario”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **server.js**

```
app.post('/deleteCommissioneInternazionale', function (req, res) {
  if (req.session.utente != null && req.session.utente.type == 'admin') {
    var deleteCommis = commissionControl.deleteCommissionInternazionale(req.body.email, res)
    deleteCommis.then(function (result) {
      if (result) {
        res.json(true)
      } else {
        res.json(false)
      }
    })
  } else {
    res.json('no')
  }
})
```

- **commissioneInterControll.ejs.**

```
/**
 * This method deletes an commissario
 * @param {String} email - The commissario email
 * @param {Object} res - The HTTP res
 * @returns {Boolean} - This method returns true if the delete of external tutor was successfull, else false
 */
exports.deleteCommissionInternazionale = function (email, res) {
  return new Promise(function (resolve, reject) {
    var delcommInter = commissioneModel.deleteCommInter(email)
    delcommInter.then(function (result) {
      if (!result) {
        resolve(false)
      } else {
        resolve(true)
      }
    })
  })
}
```

- **commissioneInter.js**



```
/**  
 * This method deletes an commis by email  
 * @param {email} - commissario_email  
 * @returns {Boolean} - It returns true if the delete was sucessfull, else false  
 */  
static deleteCommInter (email) {  
    return new Promise(function (resolve, reject) {  
        MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, function (err, db) {  
            if (err) reject(err)  
            var dbo = db.db(dbName)  
            dbo.collection('Commission').findOneAndDelete({ email: email }, function (err, result) {  
                if (err) throw err  
                if (result.value != null) {  
                    resolve(true)  
                } else {  
                    resolve(false)  
                }  
                db.close()  
            })  
        })  
    })  
}
```

Aggiunta requisito “Visualizzazione Commissione”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **viewListControll.js**

```
exports.retrieveAll = function (type) {  
    return new Promise(function (resolve, reject) {  
        if (type == 'commissioneInter') {  
            var getCI = commissioneModel.RetrieveAll()  
            getCI.then(function (result) {  
                resolve(result)  
            })  
        }  
    })  
}
```

- **commissioneInter.js**



```
/**  
 * Retrieve all commission  
 *  
 * @returns {promise} – return promise  
 */  
static RetrieveAll () {  
    return new Promise(function (resolve, reject) {  
        MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, function (err, db) {  
            if (err) reject(err)  
            var dbo = db.db(dbName)  
  
            dbo.collection('Commission').find().toArray(function (err, result) {  
                if (err) throw err  
                resolve(result)  
                db.close()  
            })  
        })  
    })  
}
```

Aggiunta requisito “Aggiungere Colloquio”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **server.js**

```
app.post('/insertAppointment', function (req, res) {  
    var appuntamento = appuntamentoControl.insertAppuntamento(req, res)  
    appuntamento.then(function (result) {  
        if (result) {  
            res.cookie('insertAppunta', '1')  
            res.redirect('/index.html')  
        } else {  
            res.redirect('/index.html')  
        }  
    })  
})
```

- **appuntamentoControl.js**



Prof. De Lucia Andrea

```
/*
 * This method inserts a specific appointment
 * @param {Object} req - The HTTP req
 * @param {Object} res - The HTTP res
 * @returns {Boolean} - This method returns true if the insert of appointment was successfull, else false
 */
exports.insertAppuntamento = function (req, res) {
  return new Promise(function (resolve, reject) {
    var date = req.query.date
    var dataNotifica = req.body.inputDate
    var studentID = req.query.studentID
    var email = req.query.studentEmail
    var RegExpData = /\d{4}-\d{2}-\d{2}/
    var RegExpOra = /\d{1,2}[:]\d{2}\d{2}/

    var schedule = req.body.inputschedule
    var isRight = true;

    if ((date == null) || (!date.match(RegExpData))) {
      isRight = false
    }

    if ((schedule == null) || (!schedule.match(RegExpOra))) {
      isRight = false
    }

    if(isRight == true){
      var appuntamento = new AppuntamentoModel()
      appuntamento.setTitle(studentID)
      appuntamento.setStart(date, schedule)
      var inserted = AppuntamentoModel.insertAppuntamento(appuntamento)
      inserted.then(function (result) {
        var d = new Date();
        var date = {hour: d.getHours().toString().padStart(2, 0), minutes: d.getMinutes().toString().padStart(2, 0), seconds: d.getSeconds().toString().padStart(2, 0), day: d.getDate().toString().padStart(2, 0)}
        socket.emit('send-notification', {associatedID: email, text: {title: 'Nuovo Colloquio', text: 'Il Commissario Internazionale ha fissato il colloquio per il giorno ' + dataNotifica + ", ore " + schedule}}, resolve(result))
      })
    }
  })
}
```

- **appuntamento.js**

```
/**
 * This method inserts the appointment
 * @param {Object} appuntamento - the appointment to insert
 * @returns {Promise} - return a promise
 */
static insertAppuntamento (appuntamento) {
  return new Promise(function (resolve, reject) {
    MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, function (err, db) {
      if (err) reject(err)
      var dbo = db.db(dbName)
      dbo.collection('Appuntamento').insertOne(appuntamento, function (err, result) {
        if (err) reject(err)
        resolve(result.insertedId)
        db.close()
      })
    })
  })
}
```

Aggiunta requisito “Visualizzazione Colloqui”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **server.js**



```
app.get('/getallAppuntamenti', function (req, res) {
  if (req.session.utente != null) {
    if (req.session.utente.type == 'commission') {
      var getAllAppuntamenti = appuntamentoControl.getAllAppointments()
      getAllAppuntamenti.then(function (data) {
        if (data) {
          res.json(data)
        }
      })
    } else {
      res.cookie('onlyForCommission', '1')
      res.redirect('/index.html')
    }
  } else {
    res.cookie('cannotAccess', '1')
    res.redirect('/')
  }
})
```

- **appuntamentoControl.js**

```
/**
 * This method retrieves all appointment
 * @param {Object} req - The HTTP req
 * @param {Object} res - The HTTP response
 * @returns {Int} -
 */
exports.getAllAppointments = function () {
  return new Promise(function (resolve, reject) {
    var appointments = AppuntamentoModel.getAllAppuntamenti()
    appointments.then(function (result) {
      if (result.length > 0) {
        resolve(result)
      } else {
        resolve(null)
      }
    })
  })
}
```



- **appuntamento.js**

```
/**  
 * This method gets all appointment  
 * @returns {Promise} – return a promise  
 */  
  
static getAllAppuntamenti () {  
    return new Promise(function (resolve, reject) {  
        MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, function (err, db) {  
            if (err) reject(err)  
            var dbo = db.db(dbName)  
            dbo.collection('Appuntamento').find().toArray(function (err, result) {  
                if (err) reject(err)  
                resolve(result)  
                db.close()  
            })  
        })  
    })  
}
```

Aggiunta requisito “Assegnare Punteggio”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **server.js**

```
app.post('/setPunteggioGrad', function (req, res) {  
    var punteggio = studentControl.insertPunteggio(req, res)  
    punteggio.then(function (result) {  
        if (result) {  
            res.cookie('setPunteggio', '1')  
            res.redirect('/gestioneGraduatoria.html')  
        } else {  
            res.redirect('/index.html')  
        }  
    })  
})
```

- **studentControll.js**



```
/**  
 * This method inserts a specific punteggio  
 * @param {Object} req - The HTTP req  
 * @param {Object} res - The HTTP res  
 * @returns {Boolean} - This method returns true if the insert of punteggio was successfull, else false  
 */  
exports.insertPunteggio = function (req, res) {  
    return new Promise(function (resolve, reject) {  
        var email = req.query.email  
        var RegExpPunteggio = /^d+$/  
        var points = req.body.inputPunteggio  
        var isRight = true;  
  
        if(!points.match(RegExpPunteggio)){  
            isRight = false;  
        }  
  
        if(isRight == true){  
            var punteggioConvertito = parseInt(points);  
            var inserted = StudentModel.insertPunteggio(email,punteggioConvertito)  
            inserted.then(function (result) {  
                resolve(result)  
            })  
        }  
    })  
}
```

- **student.js**

```
/**  
 * This method inserts the punteggio  
 * @param {String} punteggio - the punteggio to insert  
 * @param {String} email - student's email  
 * @returns {Promise} - return a promise  
 */  
static insertPunteggio (email,punteggio) {  
    return new Promise(function (resolve, reject) {  
        MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, function (err, db) {  
            if (err) reject(err)  
            var dbo = db.db(dbName)  
            dbo.collection('Student').updateOne({ Email: email }, { $set: { Punteggio: punteggio } }, function (err, result) {  
                if (err) throw err  
                resolve(result)  
                db.close()  
            })  
        })  
    })
}
```

Aggiunta requisito “Visualizzazione Graduatoria”

Per l'inserimento di questo nuovo requisito è stata modificata la seguente componente:

- **server.js**



```
app.get('/getGraduatoria', function (req, res) {
  if (req.session.utente != null) {
    if (req.session.utente.type == 'student') {
      var getAllGraduatoria = studentControl.getAllPunteggi()
      getAllGraduatoria.then(function (data) {
        if (data) {
          res.send(data)
        }
      })
    } else {
      res.cookie('onlyForStudent', '1')
      res.redirect('/index.html')
    }
  } else {
    res.cookie('cannotAccess', '1')
    res.redirect('/')
  }
})
```

- **studentControl.js**



```
/**  
 * This method retrieves all appointment  
 * @param {Object} req - The HTTP req  
 * @param {Object} res - The HTTP response  
 * @returns {Int} -  
 */  
exports.getAllPunteggi = function () {  
    return new Promise(function (resolve, reject) {  
        var graduatoria = StudentModel.getAllPunteggi()  
        graduatoria.then(function (result) {  
            if (result.length > 0) {  
                resolve(result)  
            } else {  
                resolve(null)  
            }  
        })  
    })  
}
```

- **student.js**

```
/**  
 * This method gets all punteggi  
 * @returns {Promise} - return a promise  
 */  
static getAllPunteggi () {  
    return new Promise(function (resolve, reject) {  
        MongoClient.connect(url, { useNewUrlParser: true, useUnifiedTopology: true }, function (err, db) {  
            if (err) reject(err)  
            var dbo = db.db(dbName)  
            dbo.collection('Student').find({}).sort({ Punteggio: -1 }).toArray(function (err, result) {  
                if (err) throw err  
                resolve(result)  
                db.close()  
            })  
        })  
    })  
}
```



Prof. De Lucia Andrea

Modifica MongoDB

È stata modificata la collection “**Student**” poiché è stato inserito l’attributo “Punteggio”:

```
_id: ObjectId("60bb864bc160d71e2d888517")
StudentID: "0512103456"
DegreeCourse: "Computer Science"
Address: "Via delle Vie 121"
City: "Porto Sant'Elpidio"
Email: "d.devito@studenti.unisa.it"
Surname: "De Vito"
Name: "Danny"
CV: null
IDCard: null
Punteggio: "195"
> Password: Object
```

È stata aggiunta la collection “**Commission**”:

```
_id: ObjectId("60bb864cc160d71e2d888532")
name: "Iole"
surname: "Laudato"
email: "i.laudato@commis.internazione.com"
> Password: Object
```

È stata aggiunta la collection “**Appointment**”:

```
_id: ObjectId("60cf69d7ebcd5e474298b84e")
title: "Studente: 0512103456"
start: "2021-06-20T10:30:00"
```

Mostriamo la tabella delle componenti che fanno parte del nostro Actual impact set, e ne evidenzieremo quelle che fanno parte del DIS:

Classe	Impatto	Azione
sidebar.ejs	DEBOLE-INDIRETTO	Update
profile.ejs	MEDIO-DIRETTO	Update
incomi.ejs	FORTE-DIRETTO	Create
viewList.ejs	DEBOLE-DIRETTO	Update



Laurea Magistrale in informatica - Università di Salerno

Corso di *Ingegneria, Gestione ed Evoluzione del Software*



Prof. De Lucia Andrea

gestioneColloquio.ejs	FORTE-DIRETTO	Create
calendario.ejs	FORTE-DIRETTO	Create
gestioneGraduatoria.ejs	FORTE-DIRETTO	Create
graduatoriaPunteggio.ejs	FORTE-DIRETTO	Create
loginControl.js	DEBOLE-INDIRETTO	Update
commisioneInter.js	FORTE-DIRETTO	Create
appuntamentoControl.js	FORTE-DIRETTO	Create
commissioneInterControl.js	FORTE-DIRETTO	Create
viewListControll.js	DEBOLE-DIRETTO	Update
appuntamento.js	FORTE-DIRETTO	Create
studentControll.js	DEBOLE-DIRETTO	Update
student.js	DEBOLE-DIRETTO	Update
server.js	MEDIO-INDIRETTO	Update



6. Conclusioni

In questo paragrafo presenteremo le motivazioni per il quale il lavoro svolto risulta corretto.
Di seguito le cardinalità dei nostri insiemi:

CHANGE REQUEST 01 – VISUALIZZA BANDO ERASMUS

- Starting impact set $|SIS|= 1$
- Candidate impact set $|CIS|=2$
- Actual impact set $|AIS|=1$
- Discovered impact set $|DIS|=0$
- False positive impact set $|FPIS|=1$

CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE

- Starting impact set $|SIS|= 13$
- Candidate impact set $|CIS|=19$
- Actual impact set $|AIS|=17$
- Discovered impact set $|DIS|=0$
- False positive impact set $|FPIS|=2$

6.1 Precion e Recall

Esistono diverse metriche per valutare l'accuratezza del processo di impact analysis, le più diffuse sono:

- Recall: percentuale degli impatti reali inclusi nel CIS
- Precision: percentuale di impatti candidati che sono impatti reali

Affinchè risultino efficiente tale processo, recall e precision dovrebbero avere valore maggiore del 50%.

Di seguito verranno mostrati gli esiti relativi alla nostra analisi.

CHANGE REQUEST 01 – VISUALIZZA BANDO ERASMUS

Recall= $(CIS \cap AIS) / AIS = 2 \cap 1 / 1 = 1 / 1 = 100\%$

Precision= $(CIS \cap AIS) / CIS = 2 \cap 1 / 2 = 1 / 2 = 50\%$

CHANGE REQUEST 02 – GESTIONE COLLOQUI E GRADUATORIE

Recall= $(CIS \cap AIS) / AIS = 19 \cap 17 / 17 = 17 / 17 = 100\%$

Precision= $(CIS \cap AIS) / CIS = 19 \cap 17 / 19 = 17 / 19 = 89\%$