Impact Analysis

Gestione Prenotazioni

|  |  |
| --- | --- |
| Riferimento |  |
| Versione | 1.3 |
| Data | 23/05/2021 |
| Destinatario | Prof.re De Lucia |
| Presentato da | Aldo Claudini e Emilio Schiavo |

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versione | Descrizione | autore |
| 04/05/2021 | 1.0 | Creazione del documento con:  capitolo 1  capitolo 2  capitolo 3 | Aldo Claudini, Emilio Schiavo |
| 05/05/2021 | 1.1 | Studio di fattibilità capitolo 4 | Aldo Claudini,  Emilio Schiavo |
| 22/05/2021 | 1.2 | Definizone del actual impact set | Aldo Claudini,  Emilio Schiavo |
| 23/05/2021 | 1.3 | Report | Aldo Claudini, Emilio Schiavo |

**Indice**

[1. Scopo del documento 4](#_Toc74159309)

[2. Panoramica del sistema attuale 4](#_Toc74159310)

[2.1 Attori 4](#_Toc74159311)

[2.2 Design, implementazione e testing del sistema attuale 5](#_Toc74159312)

[2.2.1 Implementazione 5](#_Toc74159313)

[2.2.2 Testing 5](#_Toc74159314)

[3. Analisi della modifica e impact set 5](#_Toc74159315)

[3.1 Starting Impact Set 6](#_Toc74159316)

[3.2 Candidate Impact Set 6](#_Toc74159317)

[3.2.1 Aggiunta del requisito “Assegna/rimuovi aula per i Tutor” 6](#_Toc74159318)

[3.2.2 Aggiunta del requisito “Assegna/rimuovi Tutor” 7](#_Toc74159319)

[3.2.3 Aggiunta del requisito “Visualizza elenco dei tutor” 7](#_Toc74159320)

[4. Studio di fattibilità 8](#_Toc74159321)

[4.1 Identificazione, descrizione e valutazione dei costi 8](#_Toc74159322)

[4.2 Identificazione, descrizione e valutazione dei benefici 9](#_Toc74159323)

[4.3 Identificazione, descrizione e valutazione dei rischi 9](#_Toc74159324)

[4.5 Risultati previsti 9](#_Toc74159325)

[5. Actual Impact set 9](#_Toc74159326)

[5.1 Client-side 9](#_Toc74159327)

[5.1.1 Aggiunta visualizzazione tipo utente 9](#_Toc74159328)

[5.1.2 Aggiunta del requisito “Assegna/rimuovi aula per i tutor” 10](#_Toc74159329)

[5.1.3 Aggiunta del requisito “Aggiungi/Rimuovi Tutor” 10](#_Toc74159330)

[5.1.4 Aggiunta del requisito “Visualizza lista Tutor” 11](#_Toc74159331)

[5.2 Server-side 12](#_Toc74159332)

[5.2.1 Visualizza lista tutor 12](#_Toc74159333)

[5.2.2 Aggiunta del requisito “Aggiungi /rimuovi tutor ” 12](#_Toc74159334)

[5.2.3 Aggiunta del requisito “Assegna/Rimuovi aula per i tutor” 16](#_Toc74159335)

[5.3 Modifica MySQL 17](#_Toc74159336)

[6. Conclusioni 18](#_Toc74159337)

[6.1 Precision e Recall 18](#_Toc74159338)

# Scopo del documento

Lo scopo del presente documento è la descrizione delle nuove funzionalità introdotte nel documento *“Change Request”* e che si prevede di implementare. Verrà stilata un’analisi d’impatto dei cambiamenti previsti al sistema, evidenziandone i rischi, i costi e i benefici.

# Panoramica del sistema attuale

“Gestione Prenotazioni” è una piattaforma di prenotazione online che, come intuibile dal nome, ha come obiettivo la gestione delle prenotazioni delle aule di Ateneo, sviluppato in particolare per l’Università degli Studi di Salerno.   
Le funzionalità offerte all’utente, oltre alle classiche funzionalità di accesso come login e signup, sono:

* Visualizzazione del calendario;
* Inserimento di una richiesta di prenotazione;
* Handling delle richieste di prenotazione;
* Amministrazione delle infrastrutture e delle aule per gli studenti;
* Gestione degli amministratori di dipartimento.

## 2.1 Attori

Il sistema distingue vari tipi di attori principali:

* Utente non iscritto, riesce solo a visualizzare il calendario delle lezioni;
* Studente iscritto, che è in grado di visualizzare gli orari di prenotazione delle aule prenotate e può prenotare le aule che gli sono state assegnate dall’amministratore di dipartimento;
* Docente, che può prenotare tutte le aule a disposizione, può visualizzare le prenotazioni effettuate, accedere al calendario ed eliminare una richiesta di prenotazione;
* Amministratore di dipartimento, che valuta le richieste di prenotazione che afferiscono al proprio dipartimento e gestisce l’assegnazione delle aule studio.
* Amministratore di Ateneo, il quale gestisce gli edifici dell’Università, i dipartimenti e l’assegnazione degli Amministratori di dipartimento, svolge funzione di moderatore ed ha pieno controllo della piattaforma.

## 2.2 Design, implementazione e testing del sistema attuale

Il sistema attuale è stato concepito come un’architettura three-tier.   
Il layer di presentazione, ovvero lo strato software responsabile dell’interazione utente, è stato sviluppato utilizzando la tecnologia di programmazione JSP, basata sul linguaggio Java.  
Lo strato di business logic è stato sviluppato seguendo un approccio object-oriented, utilizzando il linguaggio Java.  
Lo storage, quindi la gestione dei dati persistenti, è basato sul relational database management system MySQL.

### 2.2.1 Implementazione

L’implementazione del sistema prevede lo sfruttamento del pattern architetturale Model-View-Controller. Per la gestione della persistenza ad alto livello è stato utilizzato il pattern strutturale Data Access Object.

### 2.2.2 Testing

La gestione del testing della piattaforma *Gestione Prenotazioni* è stata organizzata secondo un approccio di tipo *black box* ed una strategia top-down, la quale prevede in un primo momento il testing individuale dei sottosistemi nel layer più in alto nella gerarchia, successivamente vengono integrati i layer più in basso, reiterando fino al completamento del test di ogni sottosistema. Tale metodologia è stata applicata sfruttando i *test stub*, delle implementazioni parziali di una componente utilizzata dall’unità testata.

In particolare, le operazioni di testing sono state suddivise in tre macrofasi:

* nella prima fase sono stati eseguiti i test di unità dei singoli componenti, in modo da testare nello specifico la correttezza di ciascuna unità andandone a constatare il corretto funzionamento;
* nella seconda fase è stato effettuato il testing di integrazione, test ad alto livello per verificare la corretta interoperabilità dei vari sottosistemi;
* infine, nella terza fase è stato eseguito il testing di sistema, dove si è esaminato il funzionamento dell’intero sistema assemblato e pronto alla release.

# Analisi della modifica e impact set

La change request presentata prevede l’implementazione di nuovi requisiti funzionali al sistema attuale.

Una prima fase di approccio al progetto prevede l’applicazione di tecniche di Reverse Engineering, in modo da permettere ai Team Members di avere buona familiarità con il contesto applicativo.   
Si prevede quindi una iniziale analisi del codice sorgente, avendo come obiettivo l’analisi delle varie componenti software, così da comprendere l’intera logica di business.

## 3.1 Starting Impact Set

Analizzando la change request più approfonditamente ed esaminando i dati raccolti [1], è possibile individuare uno Starting Impact Set, che rappresenta l’insieme delle componenti che verranno direttamente impattate dalle modifiche proposte.   
Nella tabella seguente sono indicate le componenti dello Starting Impact Set, l’impatto della modifica è stato valutato su quattro livelli:

* FORTE: nel caso in cui sia necessario riscrivere completamente l’artefatto;
* MEDIO: nel caso in cui sia necessaria una rivisitazione e modifica dell’artefatto;
* DEBOLE: nel caso in cui siano necessarie solo modifiche marginali;
* NULLO: nel caso in cui non siano presenti modifiche da apportare.

|  |  |  |
| --- | --- | --- |
| Artefatto | Impatto | Descrizione |
| Requisiti funzionali | MEDIO | Aggiunta dei requisiti:   * Assegna/rimuovi aula per i tutor * Assegna/rimuovi tutor * Visualizza elenco dei tutor |
| Casi d’uso | MEDIO | Aggiunta casi d’uso:   * Assegna/rimuovi aula per i tutor; * Assegna/rimuovi tutor; * Visualizza elenco tutor. |
| DB | MEDIO | Aggiunta tabella corso |

## 3.2 Candidate Impact Set

In questa successiva fase viene effettuato un ulteriore lavoro di reverse engineering [1] dove sono esaminati i Sequence Diagram per valutare la dinamicità del sistema software in relazione ai requisiti prefissati e i Class Diagram relativi al sistema.

Questo lavoro è necessario per individuare le componenti impattate indirettamente da quelle prese in considerazione in quanto definito nello Starting Impact Set descritto in quanto precede.

Partendo dal SIS, è stato analizzato più approfonditamente il Class Diagram, evidenziando le varie dipendenze per poi farne un’analisi ed estrapolare le componenti che faranno parte del Candidate Impact Set.

### 3.2.1 Aggiunta del requisito “Assegna/rimuovi aula per i Tutor”

È stato preso inizialmente in considerazione il requisito “Assegna/Rimuovi aula per i Tutor”, il quale è implementato tramite la JSP **CalendarioAula.jsp,** dove sarà inserito un nuovo pulsante per la rimozione o l’assegnazione dei permessi dei tutor, l’analisi che segue intende esplicare i componenti che dipendono da essa e che quindi potrebbero essere realmente impattati, direttamente o indirettamente, dai cambiamenti previsti.

|  |  |  |
| --- | --- | --- |
| Classe | Impatto | N° |
| AulaStudenti.java | DIRETTO | 1 |
| CalendarioAula.jsp | DIRETTO | 2 |
| Corso.java | DIRETTO | 3 |
| CorsoDAO.java | DIRETTO | 4 |
| ServletBasic.java | INDIRETTO | 4.1 |

### 3.2.2 Aggiunta del requisito “Assegna/rimuovi Tutor”

Di seguito viene preso in considerazione l’aggiunta del nuovo requisito “Assegna/Rimuovi Tutor”, che impatta direttamente la JSP **Header.jsp,** nella quale sarà inserito un nuovo pulsante che, grazie a NavGestisciTutor, reindirizza alla servlet EditTutor, la quale avrà come compito le funzionalità di assegnazione e rimozione dei tutor in base al corso. Da qui bisogna creare un ulteriore tabella nel DB per i corsi, poichè ogni tutor deve essere associato ad un corso.

Inoltre dato che queste componenti non sono state ancora implementate, non è possibile sapere l’impatto effettivo che potrebbe avere questa modifica, cominciamo a vedere però l’impatto che ha sui componenti già esistenti.

Nella seguente tabella vengono mostrate le componenti che potrebbero essere impattate direttamente o indirettamente da tale modifica.

|  |  |  |
| --- | --- | --- |
| Classe | Impatto | N° |
| Header.jsp | DIRETTO | 1 |
| Areapersonale.jsp | DIRETTO | 2 |
| NavGestisciTutor.jsp | DIRETTO | 3 |
| GestisciTutor.jsp | DIRETTO | 4 |
| EditTutor.java | DIRETTO | 5 |
| NavEditTutor.java | DIRETTO | 6 |
| AulaStudenti.java | INDIRETTO | 2.1 |
| ServletBasic.java | INDIRETTO | 5.1 |
| DeleteTutor | DIRETTO | 7 |

### 3.2.3 Aggiunta del requisito “Visualizza elenco dei tutor”

In questo paragrafo si prende in esame l’aggiunta del requisito “Visualizza elenco dei tutor”.   
Sarà la nuova funzionalità implementata grazie alla componente **GestisciTutor,** che permetterà di visualizzare l’elenco di tutti gli studenti che possono essere nominati tutor, tramite la classe già esistente **Utente**.  
Dato che **GestisciTutor** non è ancora implementata non è possibile vedere le probabili componenti impattate da essa, ma sappiamo che sicuramente verrà utilizzata la componente **Header.jsp,** di conseguenza andiamo a considerare nella seguente tabella le componenti che possono essere impattate da essa.

|  |  |  |
| --- | --- | --- |
| Classe | Impatto | N° |
| Header.jsp | DIRETTO | 1 |
| ElencoUtenti.jsp | INDIRETTO | 1.1 |
| UtenteDAO.java | INDIRETTO | 1.1.1 |

Al temine di questa analisi si giunge alla conclusione che il Candidate Impact Set conterrà le seguenti componenti:

|CIS|=14

|  |  |
| --- | --- |
| Classe | Impatto |
| AulaStudenti.java | DIRETTO |
| CalendarioAula.jsp | DIRETTO |
| Corso.java | DIRETTO |
| CorsoDAO.java | DIRETTO |
| ServletBasic.java | INDIRETTO |
| Areapersonale.jsp | DIRETTO |
| NavGestisciTutor.jsp | DIRETTO |
| GestisciTutor.jsp | DIRETTO |
| EditTutor.java | DIRETTO |
| NavEditTutor.java | DIRETTO |
| DeleteTutor | DIRETTO |
| ElencoUtenti.jsp | INDIRETTO |
| Header.jsp | DIRETTO |
| UtenteDAO | INDIRETTO |

# Studio di fattibilità

## Identificazione, descrizione e valutazione dei costi

|  |  |  |
| --- | --- | --- |
| Identificatore | Valutazione | Motivazione |
| Integrazione dei requisiti funzionali | FORTE | L’aggiunta dei requisiti avverrà in modo incrementale. |
| Modifica DB | MEDIA | Verrà aggiunto un nuovo attributo all’entità utente del db in questione e verrà creata una nuova tabella “corsi” in relazione agli utenti che diventeranno tutor. |
| Implementazione con Java e jsp | DEBOLE | Lo sviluppatore ha sufficienti conoscenze della tecnologia. |
| Testing funzionale | MEDIA | Le componenti del sistema sono già state testate, bisogna testare le nuove funzionalità implementate e la compatibilità con quelle presenti nel sistema corrente. |

## Identificazione, descrizione e valutazione dei benefici

|  |  |  |
| --- | --- | --- |
| Identificatore | Valutazione | Motivazione |
| Facilità nella gestione delle aule per i tutor | FORTE | L’amministratore di dipartimento può gestire e riservare con facilità le aule per i tutor |
| Facilità nella gestione dei Tutor per i corsi | FORTE | L’ amministratore di dipartimento può gestire i tutor in maniera estremamente comoda e veloce tramite un’interfaccia apposita. |

## Identificazione, descrizione e valutazione dei rischi

|  |  |  |
| --- | --- | --- |
| Identificatore | Valutazione | Motivazione |
| Introduzione di nuovi fault | FORTE | L’attività di testing da effettuare potrà avere un’alta probabilità di riscontrare fault, questo è dovuto alle nuove modifiche. |
| Carico di lavoro eccessivo per il client | DEBOLE | Il client dovrà solo elaborare dei dati provenienti da richieste a servizi forniti dal server. |
| Carico di lavoro eccessivo per il server | DEBOLE | Al server saranno attuate lievi modifiche e la maggior parte delle azioni saranno eseguite utilizzando i metodi già esistenti che effettuano operazioni CRUD. |

## 

## Risultati previsti

La realizzazione della modifica terrà conto di tale studio di fattibilità.

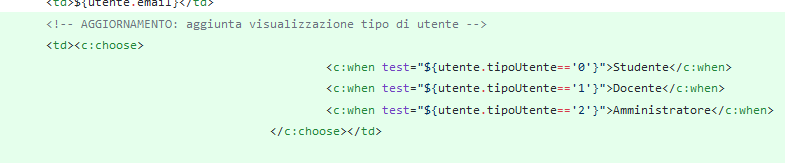
# Actual Impact set

Di seguito saranno elencante le componenti realmente modificate, tali componenti faranno parte dell’**AIS**.

## 5.1 Client-side

### 5.1.1 Aggiunta visualizzazione tipo utente

1. In AreaPersonale.jsp aggiunto campo tipoutente



### 5.1.2 Aggiunta del requisito “Assegna/rimuovi aula per i tutor”

Per l’inserimento di questo nuovo requisito sono state modificate le seguenti componenti:

1. CalendarioAula.jsp, inserimento del nuovo bottone aggiungi o rimuovi aula tutor.



### 5.1.3 Aggiunta del requisito “Aggiungi/Rimuovi Tutor”

1. In Header.jsp Aggiunto pulsante “Gestisci Tutor”

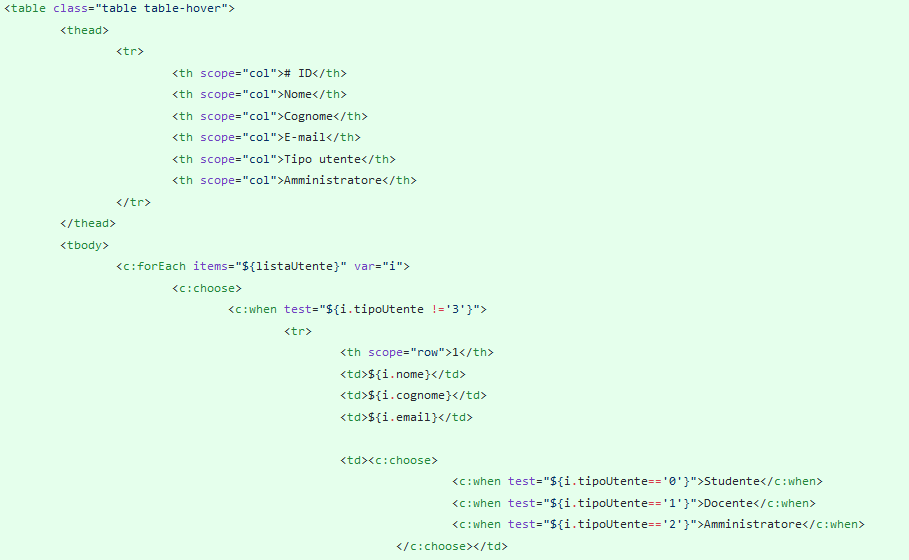


1. Creazione dell’elenco di studenti che sono tutor e non in GestisciTutor.jsp



### 5.1.4 Aggiunta del requisito “Visualizza lista Tutor”

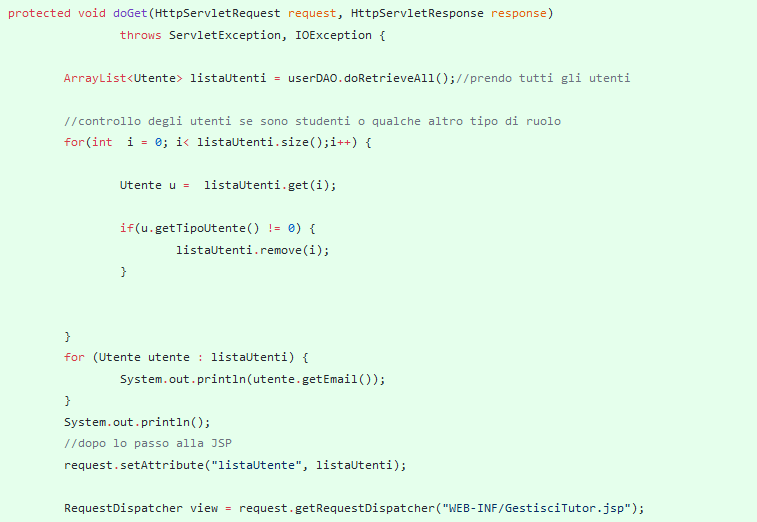
1. In GestisciTutor.jsp, implementata la lista per la visualizzazione dei tutor



## 5.2 Server-side

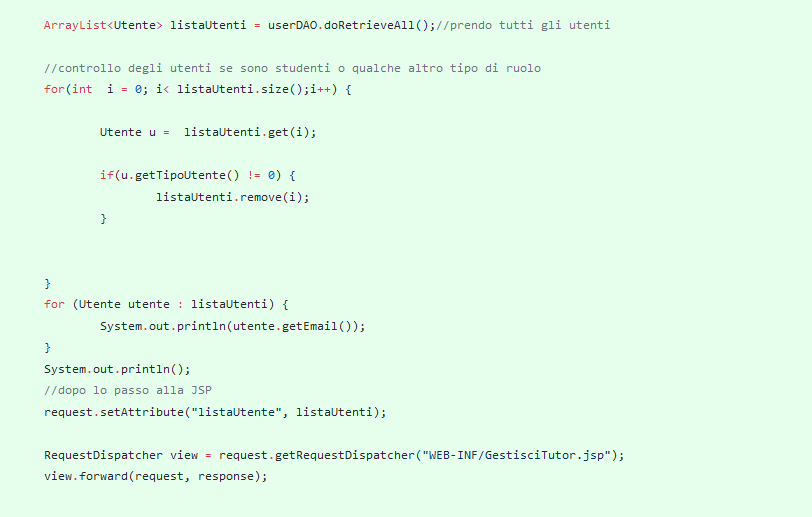
### 5.2.1 Visualizza lista tutor

1. In NavGestisciTutor.java, creata lista degli utenti da passare al front-end

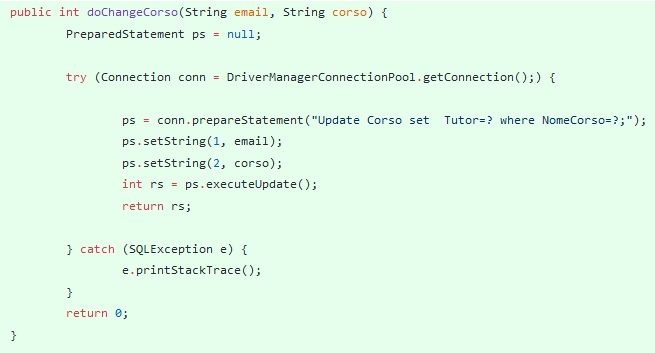


### 5.2.2 Aggiunta del requisito “Aggiungi /rimuovi tutor ”

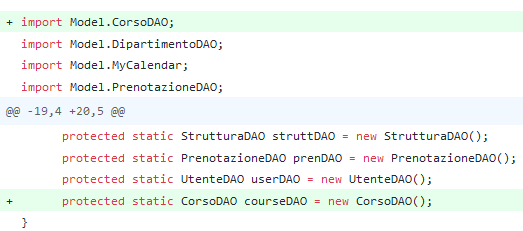
1. Creazione della servlet NavGestisciTutor.java per il reindirizzamento alla pagina

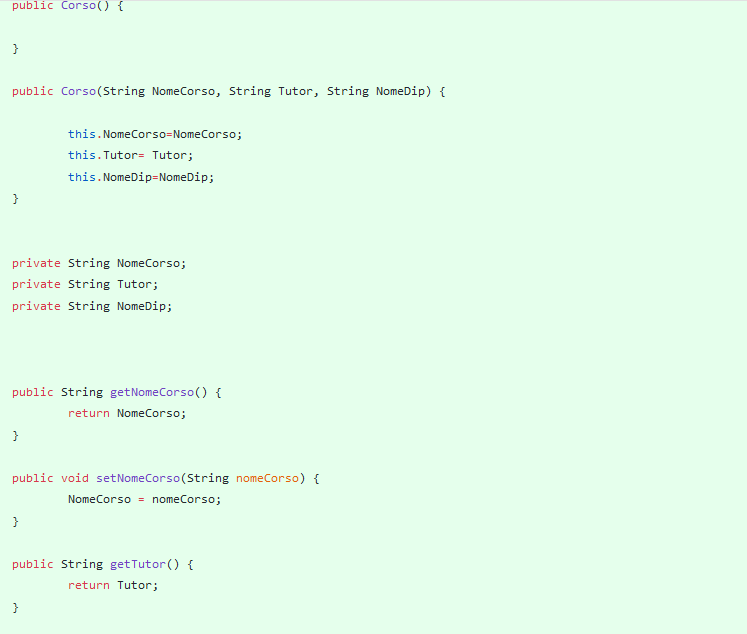
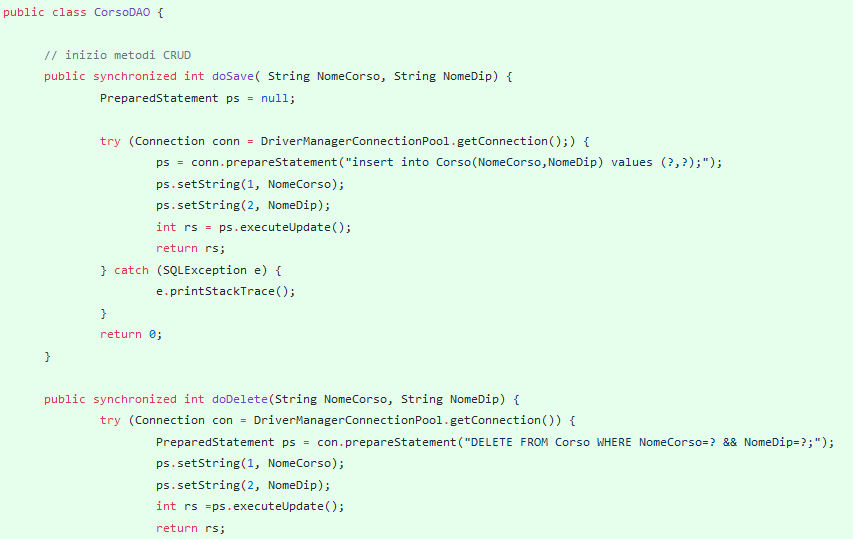


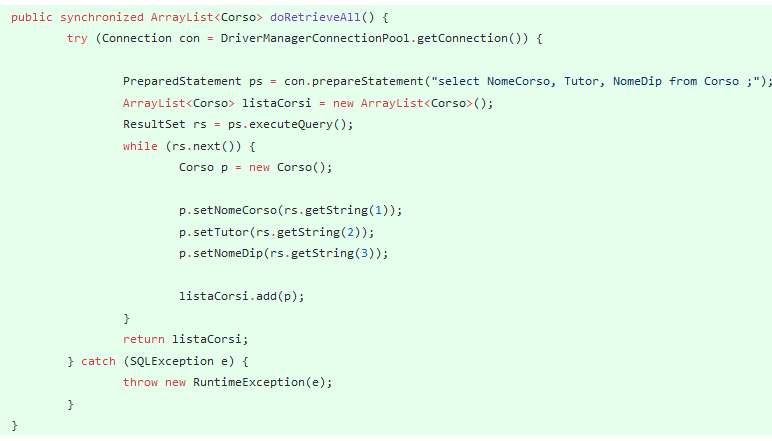
1. In UtenteDAO.java aggiunta funzione *doChangeCorso*



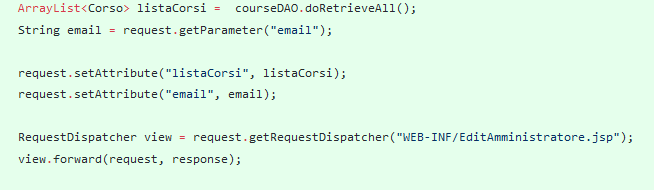
1. In ServletBasic.java, aggiunta di una nuova variabile d’ambiente courseDAO



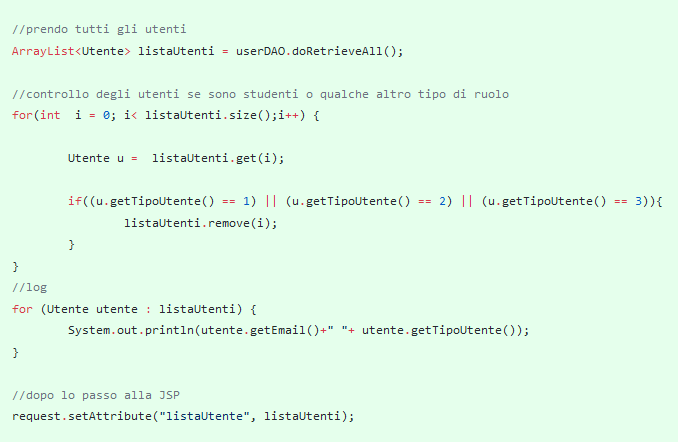
1. Creazione della classe Corso 
2. Creazione della classe CorsoDAO.java e aggiunta dei metodi CRUD 
3. In CorsoDAO. Aggiunta metodo doRetrieveAll()



1. Creazione della classe EditTutor.java che assegna o rimuove il grado di tutor di dipartimento ad uno studente 
2. Creazione della classe NavEditTutor per la reindirizzazione alla lista di tutti i corsi di ateneo

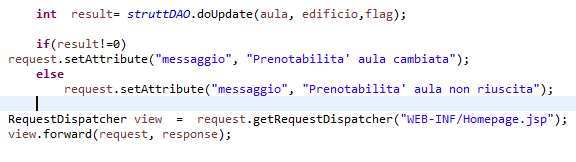


9)In DeleteTutor.java, eliminazione del tutor



### 5.2.3 Aggiunta del requisito “Assegna/Rimuovi aula per i tutor”

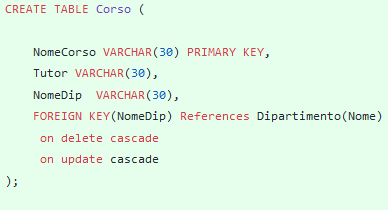
1. In AulaStudenti.java è stato usato il metodo doUpdate per salvare la modifica dell’ aula nel database



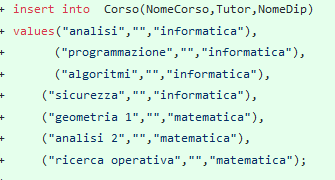
## 5.3 Modifica MySQL

## 

1. Aggiunta tabella corso al database



1. Popolamento tabella corso



Viene mostrata la tabella delle componenti che fanno parte dell’Actual Impact Set, evidenziando quelle presenti anche nel DIS

|  |  |  |
| --- | --- | --- |
| Classe | Impatto | Azione |
| NavGestisciTutor.java | MEDIO-DIRETTO | Create |
| GestisciTutor.jas | DEBOLE-DIRETTO | Create |
| AreaPersonale.jsp | DEBOLE-DIRETTO | Update |
| CalendarioAula.jsp | DEBOLE-DIRETTO | Update |
| Header.jsp | FORTE-DIRETTO | Update |
| UtenteDAO.java | DEBOLE-INDIRETTO | Update |
| Utente | DEBOLE-INDIRETTO | Update |
| Corso.java | MEDIO-DIRETTO | Create |
| CorsoDAO.java | MEDIO-DIRETTO | Create |
| ServletBasic.java | DEBOLE-INDIRETTO | Update |
| EditTutor.java | MEDIO-DIRETTO | Create |
| NavEditTutor.java | MEDIO-DIRETTO | Create |
| EditTutor.jsp | FORTE-DIRETTO | Create |
| DeleteTutor.java | MEDIO-DIRETTO | Create |

# Conclusioni

In questo paragrafo vengono presentate le metriche relative ai dati del lavoro svolto.

Di seguito le cardinalità dei nostri insiemi:

* Starting Impact Set |SIS| = 3
* Candidate Impact Set |CIS| = 14
* Actual Impact Set |AIS| = 14
* Discovered Impact Set |DIS| = 2 (Sottostima dell’impatto)
* False Positive Impact Set |FPIS| = 2 (Sovrastima dell’impatto)

## 6.1 Precision e Recall

Esistono diverse metriche per valutare l’accuratezza del processo di Impact Analisys, le più diffuse sono:

* Recall: percentuale degli impatti reali inclusi nel CIS;
* Precision: percentuale di impatti candidati che sono impatti reali.

Affinché risulti efficiente tale processo, recall e precision dovrebbero avere valore maggiore del 50%.

Di seguito verranno mostrati gli esiti relativi all’analisi svolta.

( CIS ∩ AIS) = 12

Recall= (CIS ∩ AIS)/AIS = 12/14=85%

Precision= (CIS ∩ AIS)/CIS = 12/14=85%