Immagine che contiene oggetto

Descrizione generata automaticamente

ODD

Object

Design

Document

Gestione Prenotazione

**Borriello Martina 0512105144**

**Claudini Aldo 0512105486**

**Claro Andrea 0512105252**

**Marcantuono Veronica 051210542**

**Sommario**

[**1. INTRODUZIONE** 3](#_Toc31491637)

[**1.1 Object Design Trade-offs** 3](#_Toc31491638)

[**1.1 Linee Guida per la Documentazione delle Interfacce** 3](#_Toc31491639)

[**1.3 Definizioni, acronimi e abbreviazioni** 5](#_Toc31491640)

[**1.4 Design Pattern** 6](#_Toc31491641)

[**1.4.1 Design Pattern DAO (Data Access Object)** 6](#_Toc31491642)

[**1.4.2 Singleton Pattern** 6](#_Toc31491643)

[**1.5 Riferimenti** 6](#_Toc31491644)

[**2. PACKAGES** 7](#_Toc31491645)

[**2.1 Package Controller** 7](#_Toc31491646)

[**2.2 Package model** 12](#_Toc31491647)

[**2.3 Package Front-end** 13](#_Toc31491648)

[**3. CLASS INTERFACE** 16](#_Toc31491649)

[**3.1 Class Interface Controller** 16](#_Toc31491650)

[**3.2 Class Interface Model** 51](#_Toc31491651)

[**4.GLOSSARIO** 59](#_Toc31491652)

# **1. INTRODUZIONE**

Dopo aver stilato i documenti Requirements Analysis e System Design è necessario porre attenzione sugli aspetti implementativi. Questo documento ha l’obiettivo produrre un modello che integri in modo coerente tutte le informazioni collezionate nelle fasi precedenti. In particolar modo, in tale documento verranno definite le interfacce delle classi, le operazioni supportate, i tipi dei dati, i parametri delle procedure, i signatures dei sottosistemi definiti nel documento di System Design, i trade-offs e le linee guida.

## **1.1 Object Design Trade-offs**

**Comprensibilità vs Tempo:**

Il codice del sistema deve essere comprensibile, in modo da facilitare la fase di testing ed eventuali future modifiche da apportare. Al fine di rispettare queste linee guida il codice sarà integrato da commenti volti a migliorarne la leggibilità; tuttavia questo richiederà una maggiore quantità di tempo necessario per lo sviluppo del nostro progetto.

**Interfaccia vs Usabilità:**

Verrà realizzata un’interfaccia grafica chiara e concisa, usando form e pulsanti predefiniti che hanno lo scopo di rendere semplice l’utilizzo del sistema da parte dell’utente finale.

**Sicurezza vs Efficienza:**

Il sistema mira ad essere chiuso per garantire la privacy, tuttavia ci limiteremo ad implementare sistemi di sicurezza basati su email e password a causa dei tempi limitati in modo da favorire l’efficienza.

## **1.1 Linee Guida per la Documentazione delle Interfacce**

**Naming Convention:**

Alla luce di quanto analizzato, sarà necessario utilizzare nomi:

* Descrittivi;
* Pronunciabili;
* Di uso comune;
* Di lunghezza medio-corta;
* Utilizzando solo caratteri consentiti (a-z, A-Z, 0-9);
* Senza prefissi o suffissi.

**Variabili:**

* I nomi delle variabili dovranno iniziare con la lettera minuscola, e le parole successive con la lettera maiuscola.
* Le variabili locali non saranno dichiarate all’inizio del blocco che le contiene, ma verranno dichiarate in corrispondenza del punto in cui vengono usate per la prima volta: il tutto è volto alla limitazione del loro scope.
* In ogni riga dovrà esserci un’unica variabile dichiarata, eventualmente allineata con quelle del blocco dichiarativo.
* In determinati casi, è possibile utilizzare il carattere underscore “\_”: il caso principale previsto è quello relativo alla dichiarazione di costanti oppure di proprietà statiche.

**Metodi:**

* I nomi dei metodi dovranno iniziare con la lettera minuscola e le parole successive con la lettera maiuscola.
* Il nome del metodo sarà costituito da un verbo che ne identifica l’azione seguito da un sostantivo, eventualmente aggettivato.
* Il nome dei metodi accessori e modificatori seguirà, rispettivamente, i pattern getNomeVariabile e setNomeVariabile.
* Nel caso in cui vengano utilizzati costrutti “if”, “else”, “for”, “do” e “while” nei metodi dovranno essere utilizzate le parentesi graffe, anche se essi constano di una sola istruzione.
* Ai metodi verrà aggiunto un commento JavaDoc, il quale deve essere posizionata prima della dichiarazione del metodo, e deve descriverne lo scopo. Il commento dovrà comprendere anche le informazioni riguardanti i parametri, il valore di ritorno e le eccezioni lanciate.
* I metodi dovranno essere raggruppati in base alla loro funzionalità.

**Classi Java e pagine JSP:**

* I nomi delle classi e delle pagine dovranno iniziare con la lettera maiuscola, così come le parole successive all’interno del nome.
* I nomi delle classi e delle pagine dovranno corrispondere alle informazioni e le funzioni fornite da quest’ultime.
* Ogni file sorgente .class dovrà contenere una singola classe e dev’essere strutturato come segue:
* Breve introduzione alla classe; tale introduzione indica lo scopo della classe, l’autore, la versione e la data.

/\*\*

\* sommario dello scopo della classe.

\*

\* @author [nome dell’autore]

\* @version [numero di versione della classe]

\*/

* Le classi saranno strutturate prevedendo rispettivamente:

1. Dichiarazione della classe pubblica;

2. Dichiarazioni di costanti;

3. Dichiarazioni di variabili di classe;

4. Dichiarazioni di variabili d’istanza;

5. Costruttore;

6. Commento e dichiarazione metodi e variabili.

* Nel caso in cui una classe avesse costruttori o metodi multipli con lo stesso nome, questi saranno posizionati sequenzialmente senza codice posto fra essi.

**Packages:**

* I nomi dei packages dovranno essere scritti in minuscolo concatenando insieme diversi sostantivi o sigle, separate dal carattere “.” .
* Non saranno ammessi caratteri speciali.
* Import statici e non statici in saranno specificati in blocchi singoli e separati;
* Gli import statici non saranno usati per le classi annidate ma verranno importate con import tradizionali

## **1.3 Definizioni, acronimi e abbreviazioni**

**Acronimi:**

RAD: Requirements Analysis Document

SDD: System Design Document

ODD: Object Design Document

**Abbreviazioni:**

NC: Nessuna Condizione

NA: Nessun Attributo

## **1.4 Design Pattern**

### **1.4.1 Design Pattern DAO (Data Access Object)**

Il DAO è un pattern architetturale finalizzato alla gestione dei dati persistenti. L’uso del DAO permette di gestire la connessione con il database per estrarre e/o immagazzinare i dati. Il DAO nasconde i dettagli implementativi dall’applicazione e tutte le complessità coinvolte nell’esecuzione delle operazioni inerenti all’accesso al database. Il suo scopo è:

Il design pattern DAO ha come scopo:

* Separare le operazioni di accesso ai dati dalle operazioni di business;
* Avere un’interfaccia comune di accesso ai dati.

Fornisce diversi vantaggi, tra cui:

* Manutenibilità
* Facilità di utilizzo
* Riuso del codice

### **1.4.2 Singleton Pattern**

Si è scelto di utilizzare il Singleton Pattern che deve assicurarsi che esista una singola istanza della classe Singleton, che nel nostro caso si chiama *DriverManagerConnectionPool*, e fornire un accesso globale a tale istanza. L’istanza viene dichiarata come variabile privata statica e creata quando viene inizializzata.   
Per realizzare il singleton pattern occorre avere:

* una variabile privata statica della classe che rappresenta l'unica istanza creata;
* un metodo pubblico getInstance che torna l’istanza.

Il suo scopo è :

* Avere un accesso controllato all’unica istanza della classe
* Centralizzare informazioni e comportamenti in un'unica entità condivisa dagli utilizzatori

Il principale vantaggio offerto è:

* Mutua esclusione

# **1.5 Riferimenti**

* Object-Oriented Software Engineering Using UML, Patterns, and Java, Bernd Bruegge & Allen H. Dutoit.
* GestionePrenotazione\_RAD
* GestionePrenotazione\_SDD
* Google Java Style: <https://goo.gl/KRYr9s>

# **2. PACKAGES**

Il sistema di Gestione Prenotazioni sarà distribuito in packages:

Il package “controller” conterrà le Java Servlet

Il package “model” conterrà le risorse per l’accesso alle risorse e il relativo formato per la realizzazione dei medesimi.

Il package “front-end” conterrà le pagine html, css, jsp e js che si occuperanno dell’interazione con l’utente.

## **2.1 Package Controller**

-controller

AccettaRichiesta.java

AssegnaAulaDipartimento.java

AulaGetter.java

AulaStudenti.java

CalGetter.java

CreaAula.java

CreaDipartimento.java

CreaEdificio.java

DipGetter.java

DomandaPrenotazione.java

EditAmministratore.java

EliminaPrenotazione.java

EliminaRichiesta.java

Login.java

Logout.java

ModificaPassword.java

NavAreaPersonale.java

NavAula.java

NavCreaAula.java

NavCreaDipartimento.java

NavCreaEdificio.java

NavDipartimento.java

NavEdifici.java

NavEditAmm.java

NavElencoUtenti.java

NavEliminaAula.java

NavEliminaDipartimento.java

NavLogin.java

NavModificaPassword.java

NavRegistrazione.java

NavRichiestePrenotazioni.java

NavRimuoviEdificio.java

NavScegliAmministratore.java

NavScegliDipartimento.java

NavStoricoPrenotazioni.java

Registrazione.java

RimAulaGetter.java

RimuoviAula.java

RimuoviDipartimento.java

RimuoviEdificio.java

ScegliAmministratore.java

ServletBasic.java

ServletHome.java

|  |  |
| --- | --- |
| Classe | Descrizione |
| AccettaRichiesta.java | Controller che permette l’accettazione di una richiesta di prenotazione da parte di un amministratore. |
| AssegnaDipartimentoAula.java | Controller che gestisce l’assegnazione di un dipartimento ad un’aula da parte dell’amministratore dell’Ateneo. |
| AulaGetter.java | Controller che permette la gestione delle aule all’interno dell’edificio di cui ne fanno parte. |
| AulaStudenti.java | Controller che permette l’assegnazione dell’aula prenotata da uno studente. |
| CalGetter.java | Controller che gestisce le prenotazioni riguardanti una determinata aula. |
| CreaAula.java | Controller che gestisce la creazione di un’aula da parte dell’amministratore dell’Ateneo. |
| CreaDipartimento.java | Controller che gestisce la creazione di un dipartimento da parte dell’amministratore dell’Ateneo. |
| CreaEdifico.java | Controller che gestisce la creazione di un edificio da parte dell’amministratore dell’Ateneo. |
| DipGetter.java | andrea |
| DomandaPrenotazione.java | Controller che gestisce la creazione di una domanda di prenotazione di uno studente. |
| EditAmministratore.java | Gestisce la modifica del dipartimento ad un amministratore. |
| EliminaPrenotazione.java | Controller che permette l’eliminazione di una prenotazione dallo storico prenotazioni che sia stata accettata o meno. |
| EliminaRichiesta.java | Controller che permette ad un amministratore l’eliminazione effettiva di una prenotazione ricevuta da un utente. |
| Login.java | Controller che permette l’accesso da parte di un utente registrato nel sistema Gestione Prenotazioni. |
| Logout.java | Controller che conferma l’uscita di un utente dal sistema Gestione Prenotazioni. |
| ModificaPassword.java | Controller che permette la modifica della password di un utente registrato al sistema Gestione Prenotazioni. |
| NavAreaPersonale.java | Controller che gestisce il reindirizzamento all’area personale. |
| NavAula.java | Controller che gestisce il reindirizzamento alla pagina del calendario relativo all’aula. |
| NavCreaAula.java | Controller che gestisce il reindirizzamento alla pagina di creazione dell’aula. |
| NavCreaDipartimento.java | Controller che gestisce il reindirizzamento alla pagina per la creazione del dipartimento. |
| NavCreaEdificio.java | Controller che gestisce il reindirizzamento alla pagina per la creazione dell’edificio. |
| NavDipartimento.java | Controller che gestisce il reindirizzamento alla pagina dei dipartimenti per la loro visualizzazione. |
| NavEdifici.java | Controller che gestisce il reindirizzamento alla pagina degli edifici per la loro visualizzazione. |
| NavEditAmm.java | Controller che gestisce il reindirizzamento alle modifiche delle funzionalità dell’amministratore |
| NavElencoUtenti.java | Controller che gestisce il reindirizzamento alla pagina elenco tenti per la loro visualizzazione. |
| NavEliminaAula.java | Controller che gestisce il reindirizzamento alla pagina per la rimozione dell’aula. |
| NavEliminaDipartimento.java | Controller che gestisce il reindirizzamento alla pagina per la rimozione del dipartimento. |
| NavLogin.java | Controller che gestisce il reindirizzamento alla pagina di Login. |
| NavModificaPassword.java | Controller che gestisce il reindirizzamento alla sezione modifica password. |
| NavRegistrazione.java | Controller che gestisce il reindirizzamento alla pagina di Registrazione. |
| NavRichiestePrenotazioni.java | Controller che gestisce il reindirizzamento alla pagina per la visualizzazione delle richieste di prenotazione. |
| NavRimuoviEdificio.java | Controller che gestisce il reindirizzamento alla pagina per la rimozione di un edificio. |
| NavScegliAmministratore.java | Controller che gestisce il reindirizzamento alla sezione per l’assegnazione di un amministratore. |
| NavScegliDipartimento.java | Controller che gestisce il reindirizzamento alla pagina per la scelta del dipartimento una volta selezionati aula ed edificio. |
| NavStoricoPrenotazioni.java | Controller che gestisce il reindirizzamento alla pagina dello storico prenotazioni. |
| Registrazione.java | Controller che gestisce la registrazione di un nuovo utente. |
| RimAulaGetter.java |  |
| RimuoviAula.java | Controller che permette l’eliminazione di un’aula ed una volta eliminata gestisce il reindirizzamento alla pagina degli edifici. |
| RimuoviDipartimento.java | Controller che permette la rimozione di un dipartimentoed una volta eliminato gestisce il reindirizzamento alla pagina dei dipartimenti. |
| RimuoviEdificio.java | Controller che permette la rimozione di un edificio ed una volta eliminato gestisce il reindirizzamento alla pagina degli edifici. |
| ScegliAmministratore.java | Controller che permette l’assegnazione del titolo di un amministratore ad un utente ed una volta assegnato gestisce il reindirizzamento alla pagina dell’elenco degli utenti. |
| ServletBasic.java | Controller che gestisce la data. |
| ServletHome.java | Controller che permette lo start del sistema e gestisce il reindirizzamento alla homepage. |

## **2.2 Package model**

-model

Dipartimento.java

DipartimentoDAO.java

DriverManagerConnectionPool.java

MyCalendar.java

Prenotazione.java

PrenotazioneDAO.java

Struttura.java

StrutturaDAO.java

Utente.java

UtenteDAO.java

|  |  |
| --- | --- |
| Classe | Descrizione |
| Dipartimento.java | Model che rappresenta le informazioni riguardanti il dipartimento. |
| DipartimentoDAO.java | Model che permette di eseguire le operazioni sul DB per l’entità Dipartimento. |
| DrivenManagerConnectionPool.java | Model la cui istanza rappresenta la connessione con il database. |
| MyCalendar.java | Model che rappresenta le informazioni riguardanti il calendario. |
| Prenotazione.java | Model che rappresenta le informazioni riguardanti la prenotazione di un’aula. |
| PrenotazioneDAO.java | Model che permette di eseguire le operazioni sul DB per l’entità Prenotazione. |
| Struttura.java | Model che rappresenta le informazioni riguardanti la struttura. |
| StrutturaDao.java | Model che permette di eseguire le operazioni sul DB per l’entità Struttura. |
| Utente.java | Model che rappresenta tutti gli utenti che possono utilizzare i servizi inerenti alla piattaforma Gestione Prenotazioni. |
| UtenteDao.java | Model che permette di eseguire le operazioni sul DB per l’entità Utente. |

## **2.3 Package Front-end**

-front-end

AreaPersonale.jsp

CalendarioAula.jsp

CreaAula. jsp

CreaDipartimento. jsp

CreaEdificio. jsp

Dipartimenti. jsp

Edifici. jsp

EditAmministratore.jsp

ElencoUtenti. jsp

Header.jsp

Footer.jsp

HomePage.jsp

Login.jsp

ModificaPassword.jsp

Registrazione.jsp

RichiestePrenotazioni.jsp

RimuoviAula.jsp

RimuoviDipartimento.jsp

RimuoviEdificio.jsp

ScegliAmministratore. jsp

ScegliDipartimento.jsp

StroricoPrenotazioni. jsp

|  |  |
| --- | --- |
| Classe | Descrizione |
| AreaPersonale.jsp | Pagina che permette ad un utente loggato di visualizzare la propria area personale. |
| CalendarioAula.jsp | Pagina che permette la visualizzazione del calendario relativo all’aula. |
| CreaAula.jsp | Pagina che permette la creazione di un’aula da parte dell’amministratore dell’Ateneo. |
| CreaDipartimento.jsp | Pagina che permette la creazione di un dipartimento da parte dell’amministratore dell’Ateneo. |
| CreaEdificio.jsp | Pagina che permette la creazione di un edifico da parte dell’amministratore dell’Ateneo. |
| Dipartimenti.jsp | Pagina che permette la visualizzazione dell’elenco relativo ai dipartimenti esistenti. |
| Edifici.jsp | Pagina che permette la visualizzazione dell’elenco relativo agli edifici esistenti. |
| EditAmministratore.jsp | Pagina che permette la visualizzazione dell’elenco degli utenti e la conseguente scelta o modifica di un amministratore. |
| ElencoUtenti.jsp | Pagina che permette la visualizzazione dell’elenco degli utenti iscritti al sistema. |
| Header.jsp | Header del sistema. |
| Footer.jsp | Foote del sistema. |
| HomePage.jsp | Pagina principale del sistema da cui si può effettuare Login, Registrazione, visualizzazione degli edifici e dei dipartimenti presenti. |
| Login.jsp | Pagina che permette l’accesso da parte di un utente registrato nel sistema Gestione Prenotazioni. |
| ModificaPassword.jsp | Pagina che permette di modificare la password del proprio profilo. |
| Registrazione.jsp | Pagina in cui lo studente o il docente può effettuare la registrazione al sistema Gestione Prenotazioni compilando i campi richiesti. |
| RichiestaPrenotazione.jsp | Pagina che permette la visualizzazione delle richieste di prenotazione ad un amministratore con la conseguente possibilità di accettare o rifiutare. |
| RimuoviAula.jsp | Pagina visualizzabile solo dall’amministratore dell’ateneo e permette la rimozione di un’aula. |
| RimuoviDipartimento.jsp | Pagina visualizzabile solo dall’amministratore dell’ateneo e permette la rimozione di un dipartimento. |
| RimuoviEdificio.jsp | Pagina visualizzabile solo dall’amministratore dell’ateneo e permette la rimozione di un edificio. |
| ScegliAmministratore.jsp | Pagina che permette all’amministratore dell’ateneo la visualizzazione e la conseguente assegnazione per il titolo di amministratore ad un docente. |
| ScegliDipartimento.jsp | Pagina che permette all’amministratore dell’ateneo la visualizzazione e la conseguente assegnazione del dipartimento ad un docente. |
| StroricoPrenotazioni.jsp | Pagina che permette ad un utente di visualizzare il proprio storico delle prenotazioni. |

# **3. CLASS INTERFACE**

## **3.1 Class Interface Controller**

|  |
| --- |
| **Nome Classe** |
| AccettaRichiesta |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** AccettaRichiesta:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“id”) !=null && request.getParameter(“oraInizio”) !=null  && request.getParameter(“oraFine”) !=null && request.getParameter(“data”) !=null && request.getParameter(“aula”) !=null && request.getParameter(“edificio”) !=null  **context** AccettaRichiesta:: do Post(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** AccettaRichiesta:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“lista prenotazioni”) && request.setAttribute(“mesaggio”, “prenotazione effettutata”)  **context** AccettaRichiesta:: do Post(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| AssegnaDipartimentoAula |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** AssegnaDipartimentoAula:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“aula”) !=null && request.getParameter(“edificio”) !=null && request.getParameter(“dipartimento”) !=null  **context** AssegnaDipartimentoAula:: do Post(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** AssegnaDipartimentoAula:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”, “Aula Assegnata”)  **context** AssegnaDipartimentoAula:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| AulaGetter |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** AulaGetter:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“struttura”) !=null  **context** AulaGetter:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** AulaGetter:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.getParameter(“struttura”) !=null  **context** AulaGetter:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| AulaStudenti |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** AulaStudenti:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“aula”) !=null && request.getParameter(“edificio”) !=null && request.getParameter(“tipoaula”) !=null  **context** AulaStudenti:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** AulaStudenti:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.getParameter(“aula”) !=null && request.getParameter(“edificio”) !=null && request.getParameter(“tipoaula”) !=null  **context** AulaStudenti:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| CalGetter |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** CalGetter:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“data”) !=null && request.getParameter(“aula”) !=null  **context** CalGetter:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** CalGetter:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post : :** request.getParameter(“data”) !=null && request.getParameter(“aula”) !=null  **context** CalGetter:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| CreaAula |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** CreaAula:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“nome”) !=null && request.getParameter(“edificio”) !=null && request.getParameter(“descrizione”) !=null  **context** CreaAula ::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** CreaAula:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”, “Aula creata”)  **context** CreaAula:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| CreaDipartimento |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** CreaDipartimento:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“dipartimento”) !=null  **context** CreaDipartimento::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** CreaDipartimento:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”,”Dipartimento creato”)  **context** CreaDipartimento:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| CreaEdificio |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** CreaEdificio:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“aula”) !=null && request.getParameter(“edificio”) !=null && request.getParameter(“descrizione”) !=null  **context** CreaEdificio::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** CreaEdificio:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”,”edificio creato”)  **context** CreaEdificio:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| DipGetter |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** DipGetter:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“struttura”) !=null  **context** DipGetter::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** DipGetter:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.getParameter(“struttura”) != null  **context** DipGetter:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| DomandaPrenotazione |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** DomandaPrenotazione:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“titolo”) !=null && request.getParameter(“oraInizio”) !=null && request.getParameter(“data”) !=null && request.getParameter(“oraFine”) !=null && request.getParameter(“descrizione”) !=null && request.getParameter(“aula”) !=null && request.getParameter(“edificio”) !=null && request.getParameter(“utente”) !=null  **context** DomandaPrenotazione::do Post(request:HttpServletRequest,response:HttpServletResponse) |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** DomandaPrenotazione:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”,”hai prenotato l’aula”) || **:** request.setAttribute(“messaggio”,”hai richiesto la prenotazione”) **:** request.setAttribute(“messaggio”,”Richiesta non avvenuta orario già prenotato”)  **context** DomandaPrenotazione:: do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| EditAmministratore |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** EditAmministratore:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“email”) !=null && request.getParameter(“dip”) !=null  **context** EditAmministratore::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** EditAmministratore:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute (“risultato a seconda se è rimossa o meno”, result)  **context** EditAmministratore::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| EliminaPrenotazione |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** EliminaPrenotazione:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“id”) != null  **context** EliminaPrenotazione::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** EliminaPrenotazione:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.getParameter(“messaggio:”,”eliminazione effettuata”)  **context** EliminaPrenotazione::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| EliminaRichiesta |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** EliminaRichiesta:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“id”) != null  **context** EliminaRichiesta::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** EliminaRichiesta:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“id”) = null  **context** EliminaRichiesta::do Post(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| Login |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** Login::doGet(request:HttpServletRequest,response:HttpServletResponse) **pre:** request!=null && response!=null && request.getParameter(“email”)!=null  && request.getParameter(“password”)!=null  **context** Login:: doPost(request:HttpServletRequest,response:HttpServletResponse)  - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** Login:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.getParameter(“email”) =p && request.getParameter(“password”) =c  **context** Login:: do Post(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| Logout |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** Logout:: do Get(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.gestSession().removeAttribute (“utente”)!= null && request.gestSession().removeAttribute (“storico prenotazioni”) !=null  **context** Logout:: do Post(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** Logout::doGet(request:HttpServletRequest,response:HttpServletResponse) **post:** request.gestSession().removeAttribute (“utente”)= null && request.gestSession().removeAttribute (“storico prenotazioni”)= null  **context** Logout:: doPost(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| ModificaPassword |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** ModificaPassword::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“email”) !=null && request.getParameter(“password”) !=null  **context** ModificaPassword::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** ModificaPassword::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post:** request.setAttribute(“messaggio”,”Modifica Avvenuta  **context** ModificaPassword:: doPost(request:HttpServletRequest,response:HttpServletResponse)  - |

|  |
| --- |
| **Nome Classe** |
| NavAreaPersonale |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  - |

|  |
| --- |
| **Nome Classe** |
| NavAula |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavAula::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“aula”) !=null && request.getParameter(“edificio”) !=null  **context** NavAula::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavAula::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“struttura”,struttura) && request.setAttribute(“dip”,dip)  **context** NavAula::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavCreaAula |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavCreaAula::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“struttura”,struttura)  **context** NavCreaAula::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavCreaDipartimento |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response)  - |

|  |
| --- |
| **Nome Classe** |
| NavCreaEdificio |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response)  - |

|  |
| --- |
| **Nome Classe** |
| NavDipartimento |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavDipartimento::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaDipartimenti”,listaDipartimenti) && request.setAttribute(“listaAmministratori”,listaAmministratori)  **context** NavDipartimento::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavEdifici |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| **context** NavEdifici::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaEdifici”,listaEdifici)  **context** NavEdifici::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavEditAmm |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavEditAmm::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“email”) !=null  **context** NavEditAmm::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavEditAmm::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaDipartimenti”,listaDipartimenti) && request.setAttribute(“email”,email)  **context** NavEditAmm::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavElencoUtenti |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavElencoUtenti::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaUtente”,listaUtente)  **context** NavElencoUtenti::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavEliminaAula |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavEliminaAula::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaEdifici”,listaEdifici)  **context** NavEliminaAula::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavEliminaDiparimento |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavEliminaDipartimento::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaDip”,listaDip )  **context** NavEliminaDipartimento::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavLogin |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response)  - |

|  |
| --- |
| **Nome Classe** |
| NavModificaPassword |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response)  - |

|  |
| --- |
| **Nome Classe** |
| NavRegistrazione |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response)  - |

|  |
| --- |
| **Nome Classe** |
| NavRichiestaPrenotazione |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavRichiestaPrenotazione::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaPrenotazioni”,listaPrenotazioni )  **context** NavRichiestaPrenotazione::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavRimuoviEdificio |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  - |

|  |
| --- |
| **Nome Classe** |
| NavScegliAmministratore |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavScegliAmministratore::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaUtenti”,listaUtenti )  **context** NavScegliAmministratore::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavScegliDipartimento |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavScegliDipartimento::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“dipartimento”) !=null && request.getParameter(“aula”) != null && request.getParameter(“edificio”) != null  **context** NavScegliDipartimento::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavScegliDipartimento::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaDip”,listaDip ) && request.setAttribute(“aula”,aula)&& request.setAttribute(“edificio”,edificio)  **context** NavScegliDipartimento::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| NavStoricoPrenotazioni |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavStoricoPrenotazioni::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“utente”) !=null  **context** NavStoricoPrenotazioni::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** NavStoricoPrenotazioni::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“listaPrenotazioni”,listaPrenotazioni )  **context** NavStoricoPrenotazioni::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| Registrazione |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** Registrazione::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“nome”) !=null && request.getParameter(“email”) !=null && request.getParameter(“password”) !=null && request.getParameter(“cognome”) !=null  **context** Registrazione::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** Registrazione::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”,”Registrazione avvenuta con successo“)  **context** Registrazione::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| RimAulaGetter |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** RimAulaGetter::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“str”) !=null  **context** RimAulaGetter::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** RimAulaGetter::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.getParameter(“str”) !=null  **context** RimAulaGetter::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| RimuoviAula |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** RimuoviAula::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“edificio”) !=null && request.getParameter(“aula”) !=null  **context** RimuoviAula::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** RimuoviAula::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”,”Aula eliminata”)  **context** RimuoviAula::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| RimuoviDipartimento |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** RimuoviDipartimento::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“dipartimento”) !=null  **context** RimuoviDipartimento::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** RimuoviDipartimento::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”,”Dipartimento eliminato”)  **context** RimuoviDipartimento::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| RimuoviEdificio |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** RimuoviEdificio::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“edificio”) !=null  **context** RimuoviEdificio::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** RimuoviEdificio::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”,”Edificio eliminato”)  **context** RimuoviEdificio::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| ScegliAmministratore |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** ScegliAmministratore::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“email”) !=null && request.getParameter(“flag”) !=null  **context** ScegliAmministartore::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** ScegliAmministratore::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“messaggio”,”Modifica avvenuta  **context** ScegliAmministratore::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

|  |
| --- |
| **Nome Classe** |
| ServletBasic |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| - |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  gestione calendario. |

|  |
| --- |
| **Nome Classe** |
| ServletHome |
| **Attributi** |
| - |
| **Metodi** |
| +doGet(HttpServletRequest request, HttpServletResponse response) : void  +doPost(HttpServletRequest request, HttpServletResponse response) :void |
| **Pre-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** ServletHome::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **pre :** request.getParameter(“utente”)==null  **context** ScegliAmministartore::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |
| **Post-Condizione** |
| doGet(HttpServletRequest request, HttpServletResponse response)/doPost(HttpServletRequest request, HttpServletResponse response) :  **context** ServletHome::  doGet(request:HttpServletRequest,response:HttpServletResponse)  **post :** request.setAttribute(“utente”,null )  **context** ServletHome::  doPost(request:HttpServletRequest,response:HttpServletResponse)  **-** |

## **3.2 Class Interface Model**

|  |
| --- |
| **Nome Classe** |
| Dipartimento |
| **Attributi** |
| -dip : String  -ammDip : String |
| **Metodi** |
| +getters() & setters()  +toString() |
| **Pre-Condizione** |
| NC |
| **Post-Condizione** |
| NC |

|  |
| --- |
| **Nome Classe** |
| DipartimentoDAO |
| **Attributi** |
| NA |
| **Metodi** |
| +doSave(String dip) : int  +doDelete(String dip) : int  +doDeleteAmm(String email) : int  +doUpdate(String dip, String email) : int  +doRetriveAll() : ArrayList<Dipartimento>  +doRetriveAllDip() : ArrayList<Dipartimento>  +doRetriveAllDipartimenti() : ArrayList<String>  +doRetriveByKey(String email) : String  +doDipartimentoByStruttura(String aula,String edificio) : Dipartimento |
| **Pre-Condizione** |
| **Context** DipartimentoDAO::doSave(String dip) **pre:** dip != null  **Context** DipartimentoDAO::doDelete(String dip) **pre:** dip != null  **Context** DipartimentoDAO::doDelete(String email) **pre:** email != null  **Context** DipartimentoDAO::doUpdate(String dip, String email) **pre:** dip != null && email != null  **Context** DipartimentoDAO::doRetriveByKey(String email) **pre:** email != null  **Context** DipartimentoDAO::doRetriveByStruttura(String aula,String edificio) **pre:** aula != null && edificio != null |
| **Post-Condizione** |
| **Context** DipartimentoDAO:: doRetriveAll() **post:** listaDip != null  **Context** DipartimentoDAO:: doRetriveAllDip() **post:** listaDip !=null  **Context** DipartimentoDAO:: doRetriveAllDipartimenti() **post:** listaNomiDipartimenti != null |

|  |
| --- |
| **Nome Classe** |
| DriverManagerConnectionPool |
| **Attributi** |
| -datasource : DataSource |
| **Metodi** |
| +getConnection : Connection |
| **Pre-Condizione** |
| NC |
| **Post-Condizione** |
| NC |

|  |
| --- |
| **Nome Classe** |
| MyCalendar |
| **Attributi** |
| -date : Calendar |
| **Metodi** |
| +getters() & setters()  +getToday() String  +splitData(String tmp) void  +setDayofWeek() int  +getDayofWeek(String data) int |
| **Pre-Condizione** |
| NC |
| **Post-Condizione** |
| NC |

|  |
| --- |
| **Nome Classe** |
| Prenotazione |
| **Attributi** |
| -Idprenotazione : int  -titolo : String  -data : String  -oraInizio : int  -oraFine : int  -descrizione : String  -aulaPrenotata : String  -edificio : utente  -utente : String  -aula : String |
| **Metodi** |
| +getters() & setters()  +toString() |
| **Pre-Condizione** |
| NC |
| **Post-Condizione** |
| NC |

|  |
| --- |
| **Nome Classe** |
| PrenotazioneDAO |
| **Attributi** |
| NA |
| **Metodi** |
| +doSave(String titolo, String data, int oraInizio, int oraFine, String descrizione) : int  +doDelete(int id) : int  +doDeleteByStruttura(String edificio, String aula) : int  +doDeleteByEdificio(String edificio) : int  +doUpdate(int id) : void  +doRetriveAll() : ArrayList<Prenotazione>  +doRetriveByUtente(String nomeUtente) : ArrayList<Prenotazione>  +doRetriveByDip(String dipartimento, String data) : ArrayList<Prenotazione>  +doRetriveByDate(String email,String data) : ArrayList<Prenotazione>  +doRetriveByCalendario(String aula,String edificio, String data) :  ArrayList<Prenotazione>  +splitData(String temp) : String  +doAulabyDate(String data, String aula) : ArrayList<Prenotazione>  +controlloOra(String data, int oraInizio, int oraFine, String aula, String edificio) : int |
| **Pre-Condizione** |
| **Context** PrenotazioneDAO::doSave(String titolo, String data, int oraInizio, int oraFine, String descrizione) **pre:** titolo != null && data != null && oraInizio != null oraFine != null && descrizione != null  **Context** PrenotazioneDAO::doDelete(int id) **pre:** id != null  **Context** PrenotazioneDAO::doDeleteByStruttura(String edificio, String aula) **pre:** edificio != null && aula != null  **Context** PrenotazioneDAO::doDeleteByEdificio(String edificio) **pre:** edificio != null  **Context** PrenotazioneDAO::doUpdate(int id) **pre:** id != null  **Context** PrenotazioneDAO::doRetriveBuUtente(String nomeUtENTE) **pre:** nomeUtente != null  **Context** PrenotazioneDAO:: doRetriveByDip(String dipartimento, String data ) **pre:** dipartimento != null && data != null  **Context** PrenotazioneDAO:: doRetriveByDate(String email, String data ) **pre:** email != null && data != null  **Context** PrenotazioneDAO:: doRetriveByCalendario(String aula, String edificio, String data ) **pre:** aula != null && edificio != null && data != null  **Context** PrenotazioneDAO:: doAulabyDate(String data, String aula ) **pre:** data != null && aula != null  **Context** PrenotazioneDAO:: controlloOra(String data, int oraInizio, int oraFine, String aula, String edificio) **pre:** data != null && oraInizio != null && oraFine != null && aula != null && edificio != null |
| **Post-Condizione** |
| **Context** PrenotazioneDAO:: doRetriveAll() **post:** listaPrenotazioni != null |

|  |
| --- |
| **Nome Classe** |
| Struttura |
| **Attributi** |
| -aula : String  -edificio : String  -tipoAula : String  -dipartimento : String  -descrizione : String |
| **Metodi** |
| +getters() & setters() |
| **Pre-Condizione** |
| NC |
| **Post-Condizione** |
| NC |

|  |
| --- |
| **Nome Classe** |
| StrutturaDao |
| **Attributi** |
| NA |
| **Metodi** |
| +doSave(String aula, String edificio, String descrizione) : int  +doUpdate(String aula, String edificio, String tipoAula) : int  +doSaveEdificio(String aula, String edificio, String descrizione) : int  +doDelete(String aula, String edificio) : int  +doDeleteEdificio(String edificio) : int  +doUpdateDipartimento(String aula, String edificio, String dip) : int  +doRetriveAll() : ArrayList<Struttura>  +doRetriveByKey(String nome) : Dipartimento  +doRetriveAllEdifici() : ArrayList<String>  +doAulabyEdificio(String edificio) : ArrayList<Struttura>  +doStrutturabyName (String aula,String edificio) : Struttura  +doStrutturabyDip (String dip) : ArrayList<Struttura>  +doStrutturaByAula(String aula) : Struttura  +doStrutturabyDipartimenti(String dip) : ArrayList<Struttura>  +doRetrieveAllAule() : ArrayList<Struttura> |
| **Pre-Condizione** |
| **Context** StrutturaDAO::doSave(String aula, String edificio, String descrizione) **pre:** aula != null && edificio != null && descrizione != null  **Context** StrutturaDAO::doUpdate(String aula, String edificio, String tipoAula) **pre:** aula != null && edificio != null && tipoAula != null  **Context** StrutturaDAO::doSaveEdificio(String aula, String edificio, String descrizione) **pre:** aula != null && edificio != null && descrizione != null  **Context** StrutturaDAO::doDelete(String aula, String edificio) **pre:** aula != null && edificio != null  **Context** StrutturaDAO::doDeleteEdificio(String edificio) **pre:** edificio != null  **Context** StrutturaDAO::doUpdateDipartimento(String aula, String edificio, String dip) **pre:** aula != null && edificio != null && dip != null  **Context** StrutturaDAO::doRetriveByKey(String nome) **pre:** nome != null  **Context** StrutturaDAO::doAulaByEdificio(String edificio) **pre:** edificio != null  **Context** StrutturaDAO::doAulabyName(String aula, String edificio) **pre:** aula != null && edificio != null  **Context** StrutturaDAO::doStrutturabyDip(String dip,) **pre:** dip != null  **Context** StrutturaDAO::doStrutturaByAula(String aula) **pre:** aula != null  **Context** StrutturaDAO::doStrutturaByDipartimento(String dip) **pre:** dip != null |
| **Post-Condizione** |
| **Context** StrutturaDAO:: doRetriveAll() **post:** listaStrutture != null  **Context** StrutturaDAO:: doRetriveAllEdifici() **post:** listaNomiEdifici != null  **Context** StrutturaDAO:: doRetriveAllAule() **post:** listaAule != null |

|  |
| --- |
| **Nome Classe** |
| Utente |
| **Attributi** |
| -email : String  -password : String  -nome : String  -cognome : String  -tipoUtente : int |
| **Metodi** |
| +getters() & setters()  +toString() |
| **Pre-Condizione** |
| NC |
| **Post-Condizione** |
| NC |

|  |
| --- |
| **Nome Classe** |
| UtenteDAO |
| **Attributi** |
| NA |
| **Metodi** |
| +doSave(String email, String password, String nome, String cognome, int tipoUtente) : int  +doUpdate(String email, int flag) : int  +doChangeDip(String email, String dip) : int  +doRetriveByKey(String email, String password) : int  +doRetriveAll() : ArrayList<Utente>  +doUpdate(String email, String password) : int |
| **Pre-Condizione** |
| **Context** UtenteDAO::doSave(String email, String password, String nome, String cognome, int tipoUtente) **pre:** email != null && password != null && nome != null && cognome != null && tipoUtente != null  **Context** UtenteDAO::doUpdate(String email, int flag,) **pre:** email != null && flag != null  **Context** UtenteDAO::doChageDip(String email, String dip) **pre:** email != null && dip != null  **Context** UtenteDAO::doRetriveByKey(String email, String password) **pre:** email != null && password != null  **Context** UtenteDAO::doUpdate(String email, String password) **pre:** email != null && password != null |
| **Post-Condizione** |
| **Context** UtenteDAO:: doRetriveAll() **post:** listaUtente != null |

# **4.GLOSSARIO**

**CSS:** Cascading Style Sheet è un linguaggio di stile per pagine html.

**DBMS:** sistema software per la creazione, manipolazione e interrogazione efficiente di database.

**Design Pattern:** si tratta di una descrizione o modello logico da applicare per la risoluzione di un problema che può presentarsi in diverse situazioni durante le fasi di progettazione e sviluppo software.

**HTML:** Html è un linguaggio di mark-up per pagine web.

**JSON:** Json è un formato adatto per lo scambio dei dati in applicazioni Client-Server

**Jsp:** Una pagina jsp è un documento di testo, scritto con una sintassi specifica, che rappresenta una pagina Web di contenuto parzialmente o totalmente dinamico.

**Package:** Un package è un meccanismo per organizzare classi all’interno di sottogruppi ordinati.

**Piattaforma:** Definisce l’insieme delle funzionalità fornite dal sistema attraverso l’applicazione web.

**RAD (Requirement Analysis Document):** documento contenente informazioni inerenti al sistema da realizzare raccolte durante la fase di Requirement Analysis e Requirement Elicitation.

**SDD (System Design Document):** Documento formalizzato alla definizione di obiettivi di progettazione del sistema, decomposizione del sistema in sottosistemi più piccoli e scelta di architettura software più adatta al sistema.

**Servlet:** oggetti Java all'interno del server web che permettono di creare web applications in combinazione con JSP.

**Sistema:** Definisce la totalità delle componenti dal punto di vista strutturale/implementativo.

**Web Browser:** applicazione software installata sul client che permette di visualizzare e navigare le risorse del web.