Università degli Studi di Salerno

Corso di Ingegneria del Software

CIRECinema
Object Design Document
Versione 1.0

Nome	Matricola
Bufalino Chiara	0512102894
Massaro Edilio	0512103218
Squeglia Raffaele	0512103122
Tiberio Ildo	0512103134

Data	Versione	Descrizione	Autore

Sommario

1. Introduzione

1.1 Compromessi dell'Object Design

Per ridurre i tempi di rilascio è stato deciso di utilizzare alcune componenti software già pronte per gestire alcune delle funzionalità più complesse del sistema. In particolare verrà usata la componente software open-source JDBC MySql per l'interfacciamento col database.

1.1.1 Comprensibilità vs tempo

Il codice verrà arricchito da commenti che aiutino la comprensione del sorgente. Tali commenti però, verranno aggiunti solo a quelle parti del codice ambigue o di difficile comprensione, in modo da non aumentare troppo i tempi di sviluppo.

1.1.2 Sicurezza vs efficienza

Essendo il sistema pensato per funzionare in una rete locale, non è stato dato molto peso al problema della sicurezza: si ritiene sufficiente l'uso di un meccanismo di autenticazione Login/Password. Il mancato uso di meccanismi di crittografia delle comunicazioni migliorerà, anche se minimamente, le prestazioni del sistema.

1.2 Linee guida per la documentazione delle interfacce

Vengono qui elencate le linee guida che gli sviluppatori dovranno seguire durante la

stesura del codice sorgente:

Stile di programmazione

- Lo spazio di indentazione delle classi deve essere di un tab

Naming Convention

- I nomi delle classi devono cominciare con una lettera maiuscola, e che anche le parole seguenti all'interno del nome devono cominciare con una lettera maiuscola. I nomi delle classi inoltre devono fornire informazioni sul loro scopo.

Es. Utente

- I nomi dei metodi devono cominciare con una lettera minuscola, e le parole seguenti con la lettera maiuscola.

Es. aggiungiProiezione()

- I nomi dei metodi per l'accesso e la modifica delle variabili dovranno essere del tipo
  getNomeVariabile(), setNomeVariabile() o nel caso di valori booleani *isNomeVariabileBooleana()*

- I nomi delle variabili devono cominciare con una lettera minuscola, e le parole seguenti con la lettera maiuscola. *Es. annoIscrizione*
- I nomi delle costanti devono essere scritti a lettere maiuscole,m e le parole separate da un underscore. *Es. static final int ESEMPIO_COSTANTE = k*

Documentazione

- Ogni classe deve avere una breve spiegazione del suo scopo, dopo le istruzioni di import. Devono essere inoltre indicati l'autore della classe e altre informazioni utili utilizzando gli appropiati tag JavaDoc

```
/** descrizione della classe
**
@author nome dell'autore
* ...
*/
```

- La descrizione del metodo deve apparire prima di ogni dichiarazione di metodo e deve descriverne lo scopo. Devono essere elencati i parametri del metodo, i valori di ritorno ed eventualmente le eccezioni che possono essere lanciate, utilizzando gli appropiati tag di JavaDoc

```
/**
* Descrizione dello scopo del metodo
* @param Param1 – descrizione parametro 1
* @return descrizione del valore di ritorno
*/
```

- Le variabili devono essere documentate solo se il loro scopo non è immediatamente intuibile. Possono essere usati sia commenti JavaDoc che normali commenti

1.3 Definizioni, acronimi ed abbreviazioni

SDD : System Design Document
JDBC: Java DataBase Connectivity

1.4 Riferimenti

SDD-CeriCinema

PACKAGES

Il package di CeriCinema essendo una Web App è organizzata con la standard gestione delle directory e distinzione fra package funzionali (servlet), package di interfaccia (jsp) e package delle risorse (classi come Proiezione, Film ecc.)

2.1 Package CoreServlets

In questo pacchetto sono contenuti le classi servlet necessarie al funzionamento dell'applicazione web. Il loro scopo sarà quello di essere il ponte di comunicazione fra il database e l'interfaccia grafica dell'utente.

2.2 Package Utils

In questo pacchetto sono contenute le classi Java necessarie al funzionamento della nostra applicazione.

2. CLASS INTERFACE

3.1 Proiezione

CLASS Proiezione		Autore: Edilio Massaro
DESCRIZIONE		La classe che modella l'entità proiezione presente nel DB.
DIPENDENZA		Film
ATTRIBUTI		
Nome		**Accesso**
ID		Private
nomeProiezione		Private
Sala		Private
dataProiezione		Private
film		Private
METODI		
NOME	**ACCESSO**	**DESCRIZIONE**
get/setID	public	Metodi getter e setter per variabile id.
set/setNomeProiezione	Public	Metodi getter e setter per variabile NomeProiezione
Set/getSala	Public	Metodi getter e setter per variabile sala.
Get/SetDataProiezione	Public	Metodi getter e setter per variabile data proiezione.

3.2 Utente

CLASS Utente		Autore: Edilio Massaro
DESCRIZIONE		La classe che modella l'entità utente presente nel DB.
DIPENDENZA		Nessuna
ATTRIBUTI		
Nome		**Accesso**
iDUtente		private
username		private
nomeUtente		private
cognomeUtente		private
residenza		private
dataNascita		private
METODI		
NOME	**ACCESSO**	**DESCRIZIONE**
Metodi getter e setter per ogni variabile		

3.3 Prenotazione

CLASS Prenotazione		Autore: Edilio Massaro
DESCRIZIONE		La classe che modella l'entità prenotazione presente nel DB.
DIPENDENZA		Proiezione, Utente
ATTRIBUTI		
Nome		**Accesso**
iDPrenotazioni		private
proiezione		private
sala		private
posto		private
utente		private
METODI		
NOME	**ACCESSO**	**DESCRIZIONE**
Metodi getter e setter per ogni variabile		

3.4 Film

CLASS Film		Autore: Edilio Massaro
DESCRIZIONE		La classe che modella l'entità film presente nel DB.
DIPENDENZA		Nessuna
ATTRIBUTI		
Nome		**Accesso**
iDFilm		private
nomeFilm		private
dataUscita		private
genere		private
descrizione		private
locandina		private
METODI		
NOME	**ACCESSO**	**DESCRIZIONE**
Metodi getter e setter per ogni variabile		

3.5 DBConnector

CLASS DBConnector	Autore: Edilio Massaro
DESCRIZIONE	La classe che modella l'entità film presente nel DB.
DIPENDENZA	Nessuna

ATTRIBUTI	
Nome	**Accesso**

METODI			
NOME	**ACCESSO**	**DESCRIZIONE**	**ECCEZIONE**
getUtenti	**public**	Metodo che si connette al DB e restituisce i valori presenti nella tabella Utenti.	**SQLException**
getProiezioni	**public**	Metodo che si connette al DB e restituisce i valori presenti nella tabella Proiezioni.	**SQLException**
getFilm	**public**	Metodo che si connette al DB e restituisce i valori presenti nella tabella Film.	**SQLException**
getPrenotazioni	**public**	Metodo che si connette al DB e restituisce i valori presenti nella tabella Prenotazioni.	**SQLException**