**Università degli Studi di Salerno**

**Corso di Ingegneria del Software**

**BookPoint**

**SDD**

**Versione 1.0**

Partecipanti:

|  |  |
| --- | --- |
| **Nome Cognome** | **Matricola** |
| **Massimo Giordano** | **0512104480** |
| **Giovanni Buonocore** | **0512104612** |
| **Gennaro Teodoro** | **0512104876** |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versione** | **Descrizione** | **Autore** |
|  |  |  |  |

Sommario

# Introduzione

## 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.

**Prestazioni vs Costi:**Dal momento che il budget allocato è spendibile principalmente in risorse umane e non consente l’acquisto di tecnologie proprietarie specifiche verranno utilizzati template open source e componenti hardware di nostra proprietà.

**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:**La sicurezza rappresenta uno degli aspetti principali del sistema. Tuttavia, a causa di tempi di sviluppo molto limitati, ci limiteremo ad implementare sistemi di sicurezza basati su email e password.

## Linee Guida per la documentazione delle interfacce

Gli sviluppatori dovranno seguire precise linee guida per la stesura del codice:

**Naming Convention:**

Per la documentazione delle interfacce bisognerà utilizzare nomi:

* Descrittivi
* Pronunciabili;
* Di lunghezza medio-corta;
* Utilizzando solo caratteri consentiti (a-z, A-Z, 0-9)

**Variabili:**

* I nomi delle variabili dovranno iniziare con la lettera minuscola, e le parole successive con la lettera maiuscola.
* 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, secondo la “Camel Notation”.
* 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.
* Ai metodi verrà aggiunto un commento JavaDoc, il quale deve essere posizionata prima della dichiarazione del metodo, e deve descriverne lo scopo.

**Classi Java e pagine JSP:**

* I nomi delle classi e delle pagine dovrà iniziare con la lettera maiuscola, così come le parole successive all’interno del nome.
* I nomi delle classi e delle pagine dovrà corrispondere alle informazioni e le funzioni fornite da quest’ultime.
* Le classi saranno strutturate prevedendo rispettivamente:

1. Dichiarazione della classe pubblica;
2. Dichiarazione di costanti;
3. Dichiarazioni di variabili di classe;
4. Dichiarazioni di variabili di istanza;
5. Costruttore;
6. Metodi;

**Packages:**

* I nomi dei packages dovranno essere scritti in minuscolo concatenando insieme diversi sostantivi o sigle, separate dal carattere “.” .
* Non saranno ammessi caratteri speciali.

## Definizioni, acronimi e abbreviazioni

**Acronimi:**

* RAD: Requirements Analysis Document
* SDD: System Design Document
* ODD: Object Design Document

## Riferimenti

Documento RAD\_BookPoint.pdf

Documento SDD\_BookPoint.pdf

# Packages

# Class interfaces

# Design Pattern

## Façade Pattern

## Object Pool Pattern