**Trade-Off:**

1. **Comprensibilità vs Tempo:** Il codice del sistema deve essere comprensibile il più possibile, in modo da facilitare la fase di testing ed eventuali future modifiche da apportare. Per rispettare queste linee guida il codice sarà accompagnato da commenti che serviranno a semplificarne la comprensione.
2. **Prestazioni vs Costi:** Dato che il nostro progetto è sprovvisto di budget, per poter mantenere prestazioni elevate, in determinate funzionalità verranno utilizzati dei template open source esterni.
3. **Interfaccia vs Usabilità:** L’interfaccia grafica è stata realizzata in maniera molto semplice, chiara e concisa, vengono utilizzati i form e pulsanti con lo scopo di rendere semplice l’utilizzo del sistema da parte dell’utente finale.
4. **Sicurezza vs Efficienza:** La sicurezza, rappresenta uno degli aspetti importanti del sistema. Implementeremo sistemi di sicurezza basati su username e password degli utenti.

**Linee Guida per la documentazione delle interfacce:**

Bisogna seguire delle linee guida per la stesura del codice:

* **Variabili:** i nomi delle variabili devono iniziare con la lettera maiuscola, e le parole successive con la lettera minuscola. La dichiarazione delle variabili deve essere effettuata ad inizio blocco; in ogni riga vi deve essere una sola dichiarazione di variabile e va effettuato l’allineamento per migliorare la leggibilità.
* **Metodi:** I nomi dei metodi devono iniziare con la lettera maiuscola, e le parole successive con la lettera minuscola. Di solito il nome del metodo è costituito da un verbo che identifica un’azione, seguito dal nome di un oggetto. Ai metodi va aggiunta una descrizione, la quale deve essere posizionata prima della dichiarazione del metodo, e deve descriverne lo scopo. La descrizione del metodo deve includere anche informazioni riguardanti gli argomenti, il valore di ritorno, le eccezioni. I metodi devono essere raggruppati in base alla loro funzionalità.
* **Classi e pagine:** I nomi delle classi e delle pagine devono iniziare con la lettera maiuscola, e anche le parole successive all’interno del nome devono iniziare con la lettera maiuscola.
* **Indentazione:** L’indentazione deve essere effettuata con un TAB e qualunque sia il linguaggio usato per la produzione di codice, ogni istruzione deve essere opportunamente indentata.
* **Inizializzazione:** Inizializzare le variabili locali nel punto in cui sono state dichiarate a meno che il suo valore iniziale non dipenda da un calcolo che occorre eseguire prima.
* **Posizione:** Mettere le dichiarazioni all’inizio dei blocchi. Non aspettare di dichiarare le variabile al loro primo uso: può confondere il programmatore inesperto e impedire la portabilità del codice dentro lo scope.

**Definizioni,Acronimi ed Abbreviazioni:**

RAD: Requirement Analysis Document  
SDD: System Design Document  
ODD: Object Design Document  
DI: Dipartimento di Informatica  
UNISA: Università degli Studi di Salerno

**Riferimenti**

*Bernd Bruegge& Allen H. Dutoit, Object-Oriented Software Engineering: Using UML,Patterns and Java, (2nd edition),Prentice-Hall, 2003.   
Ian Sommerville, Software Engineering, Addison Wesely.*[*http://corsi.unisa.it/informatica/attivita-e-servizi/tirocini*](http://corsi.unisa.it/informatica/attivita-e-servizi/tirocini)