Università degli Studi di Salerno
Corso di Ingegneria del Software

Dream&Fly
Object Design Document
Versione 0.2

Data: 02/02/2024

Torna all'indice

Torna all’indice

Coordinatore del progetto:

Nome	Matricola
Singh Amandeep	0512113476
Paolillo Valentina	0512114820

Partecipanti:

Nome	Matricola
Paolillo Valentina	0512114820
Singh Amandeep	0512113476

Scritto da:	Singh Amandeep, Paolillo Valentina

Revision History

Data	Versione	Descrizione	Autore
01/02/2024	0.1	Prima stesura documento	Singh Amandeep, Paolillo Valentina
02/02/2024	0.2	Stesura sezione 2. Packages	Singh Amandeep, Paolillo Valentina

Indice

1. INTRODUCTION

1.1. Object design trade-offs

1.1.1. Tempo di rilascio VS Funzionalità

Per garantire un'esperienza dotata di tutte le funzionalità essenziali per suscitare un impatto positivo sui primi clienti, abbiamo scelto di mettere in secondo piano il tempo di rilascio per prioritizzare la qualità del sistema.

1.1.2. Velocità VS Memoria

Utilizzare più memoria per ottimizzare l'accesso veloce ai dati può aumentare le prestazioni, anche se potrebbe richiedere risorse aggiuntive.

1.1.3. Costruire VS Comprare

Sebbene utilizzare software già realizzato da altri permetta, ad esempio, l'utilizzo di funzionalità già complete oppure una minore quantità di lavoro per gli sviluppatori, è stato deciso di realizzare la maggior parte del Sistema partendo da zero, utilizzando componenti esterne soltanto in alcuni casi.

Trade-Off	
Tempo di rilascio	**Funzionalità**
Velocità	Memoria
Costruire	Comprare

1.2. Off-the-shelf components

Il Sistema utilizzerà i seguenti componenti off-the-shelf:

1. **JQuery:** framework ampiamente utilizzato per semplificare la scrittura di codice JavaScript, facilitando l'interazione con il DOM (Document Object Model) e semplificando le operazioni comuni.
2. **MySQL (v.8.0):** sistema open source di gestione di database relazionali SQL sviluppato e supportato da Oracle.
3. **MySQL Connector/J (JDBC driver):** driver JDBC ufficiale che consente e semplifica la comunicazione tra applicazioni Java e il database MySQL.
4. **Tomcat (v.9.X):** Web Server con annesso application container per applicazioni scritte in Java; utilizzato per eseguire il back-end dell'applicazione.

1.3. Interface documentation guidelines

Tali linee guida includono una lista di regole che gli sviluppatori devono seguire durante la progettazione del sistema. Il progetto Dream&Fly è realizzato con l'IDE di sviluppo EclipseIDE ed è strutturato nel seguente modo:

- **Package:** il nome del pacchetto deve essere sempre in minuscolo
- **Classi**: i nomi delle classi devono essere descrittivi e scritti in UpperCamelCase
- **Interfacce:** i nomi delle interfacce devono essere scritte in UpperCamelCase

- **Metodi**: i metodi devono essere scritti in forma camelCase
- **Variabili:** il nome delle variabili deve essere descrittivo, si evita quindi di utilizzare variabili con nomi composti da una sola lettera se non per variabili temporanee. Il formato del nome delle variabili è il lowerCamelCase.
- **Indentazione**:
  - Le istruzioni racchiuse all'interno di un blocco (esempio: for), devono essere indentate di un'unità all'interno dell'istruzione composta.
  - La parentesi di apertura del blocco deve trovarsi alla fine della riga dell'istruzione composta.
  - La parentesi di chiusura del blocco deve trovarsi allo stesso livello di indentazione dell'istruzione composta.
- **HTML:** convenzione usata per definire le linee guida su html: HTML Style Guide and Coding Conventions - W3School
- **CSS:** tutti gli stili non in-line devono essere collocati in fogli di stile separati.
- **JavaScript**:
  - Gli script devono essere collocati in file appositi.
  - Il codice JavaScript deve seguire le stesse convenzioni per il layout e i nomi del codice Java.
- **DataBase SQL:** i nomi delle tabelle e dei campi devono seguire le seguenti regole:
  - Devono essere costituiti da sole lettere minuscole.
  - Se il nome è costituito da più parole, è previsto l'uso di underscore (_).

1.4. Definitions, acronyms and abbreviations

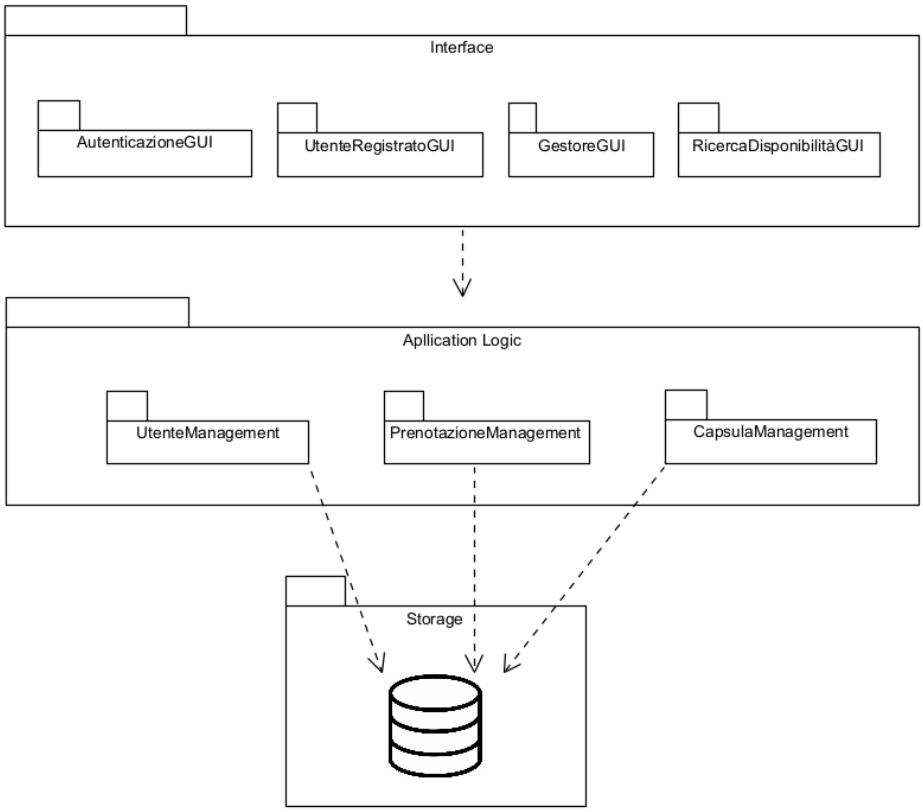
1.5. References

Commentato [AS1]: Da fare

2. PACKAGES

In questa sezione viene mostrata la suddivisione del Sistema in package, in base a quanto definito nel documento di System Design. Tale suddivisione è motivata dalle scelte architetturali prese e sottolinea la struttura di directory standard definita da Maven.

2.1. Subsystem decomposition



2.1.1. Interface

Questo package contiene i seguenti sub-package e le seguenti classi:

- Package AutenticazioneGUI
- Package UtenteRegistratoGUI
- Package GestoreGUI
- Package RicercaDisponibilitàGUI

Commentato [VP2]: Scrivere classi all interno del package autenticazione.

2.1.2. Application logic

Questo package contiene i seguenti sub-package e le seguenti classi:

- Package UtenteManagement
- Package PrenotazioneManagement
- Package CapsulaManagement

2.1.3. Storage

Questo package contiene le seguenti classi:

- AccountUser
- AccountUserDao
- Prenotazione
- PrenotazioneDao
- Capsula
- CapsulaDao
- Prenotabile
- PrenotabileDao
- FasciaOraria
- FasciaOrariaDao

3. CLASS INTERFACES