UNIVERSITÀ DEGLI STUDI DI SALERNO

Corso di Laurea in Informatica, prof. Gravino Carmine, a.a. 2022/2023

Progetto di Ingegneria del Software

Object Design Document (ODD)

Versione	0.3
Data	17/02/2023
Presentato da	Guerrera Marco, Lamberti Salvatore, Napolitano Margherita Maria, Zullo Nicola Mario

Revision History

Data	**Versione**	**Descrizione**
02/01/2023	0.1	Prima stesura e package (MMN, NMZ)
15/02/2023	0.2	Aggiunta riferimenti, definizioni
17/02/2023	0.3	Revisione

INDICE

1. Introduzione

"Tommit" vuole incentivare la condivisione di appunti, lo studio mediante gruppi divisi per argomento di interesse. In questa prima sezione del documento, verranno descritti i trade-offs e le linee guida per la fase di implementazione, riguardanti la nomenclatura, la documentazione e le convenzioni sui formati.

1.1 Object Design Goals

Durante la fase di analisi e progettazione del sistema si sono individuati diversi requisiti che il sistema deve soddisfare.

Affidabilità

Il sistema deve risultare robusto, reagendo correttamente a situazioni impreviste attraverso il controllo degli errori e la gestione delle eccezioni. Inoltre il sistema deve garantire un corretto inserimento di dati in input. Deve garantire l'accesso al sistema solo ad utenti autorizzati e le relative funzioni in accordo al proprio ruolo.

Performance

Il Sistema deve essere in grado di fornire servizio contemporaneamente ad almeno 100 utenti. Il sistema deve essere in grado di fornire una navigazione fluida, con tempi di risposta inferiore a 5s.

Modificabilità

Il sistema deve permettere l'aggiunta di nuove funzionalità.

1.2 Linee guida per la documentazione dell'interfaccia

In questo paragrafo verranno definite le linee guida a cui un programmatore deve attenersi nell'implementazione del sistema. In ciascun sotto paragrafo, quando si parla di indentazione ci si riferisce ad una tabulazione.

1.2.1 Classi e Interfacce Java

Le linee guida che abbiamo seguito per le classi ed interfaccia java sono le seguenti:

1. Il nome della classe è con la lettera maiuscola;
2. Il nome dei metodi è con la lettera minuscola.

1.2.2 Java Servlet Pages (JSP)

Le JSP costruiscono pagine HTML in maniera dinamica che devono rispettare il formato definito nel sotto paragrafo sottostante. Devono anche attenersi alle linee guida per le classi Java definito nel sotto paragrafo soprastante. Inoltre, le JSP devono attenersi alle seguenti puntualizzazioni:

1. Il tag di apertura (<%) è seguito immediatamente dalla fine della riga;

2. Il tag di chiusura (%>) si trova all'inizio di una riga che non contiene nient'altro;
3. Istruzioni Java che consistono in una sola riga possono contenere il tag di apertura e chiusura sulla stessa riga.

1.2.3 Pagine HTML

Le pagine HTML devono essere conformi allo standard HTML5. Inoltre, il codice HTML deve utilizzare l'indentazione per facilitare la lettura e di conseguenza la manutenzione. Le regole di indentazione sono le seguenti:
1. Ogni tag deve avere un'indentazione maggiore del tag che lo contiene;
2. Ogni tag di chiusura deve avere lo stesso livello di indentazione di quello di apertura.

1.2.5 Fogli di stile CSS

Ogni regola CSS deve essere formattata nel seguente modo:
1. I selettori della regola si trovano a livello 0 di indentazione, uno per riga;
2. L'ultimo selettore della regola è seguito da parentesi graffa aperta ({);
3. La regola è terminata da una parentesi graffa chiusa (}), collocata da sola su una riga.

1.3 Definizioni, acronimi, e abbreviazioni

Design Pattern: template di soluzioni a problemi ricorrenti impiegati per ottenere riuso e flessibilità;
Package: raggruppamento di classi, interfacce o file correlati

1.4 Riferimenti

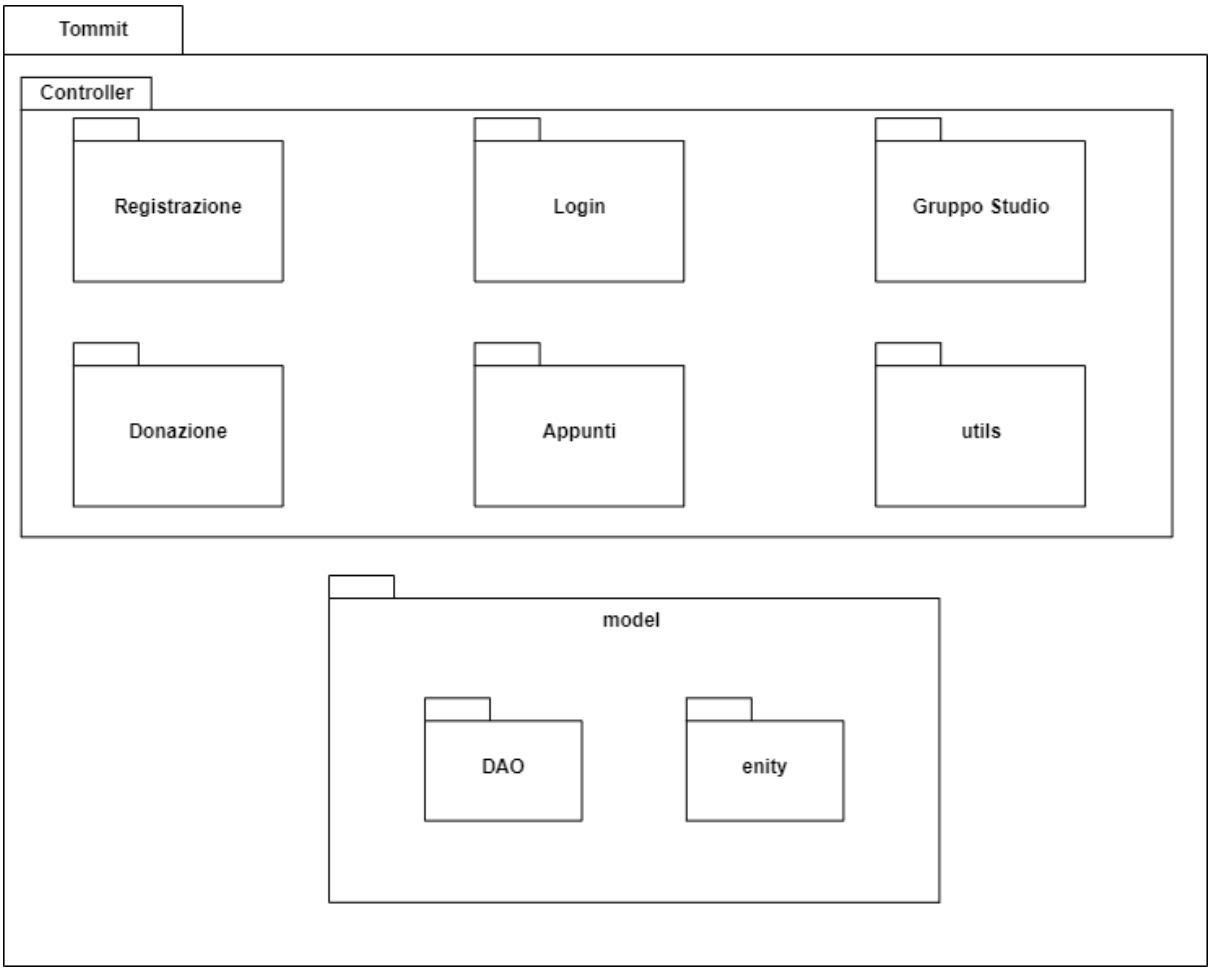
- RAD (suddivisione in sottosistemi)
- System Design Document
- Object Design Document

2. Packages

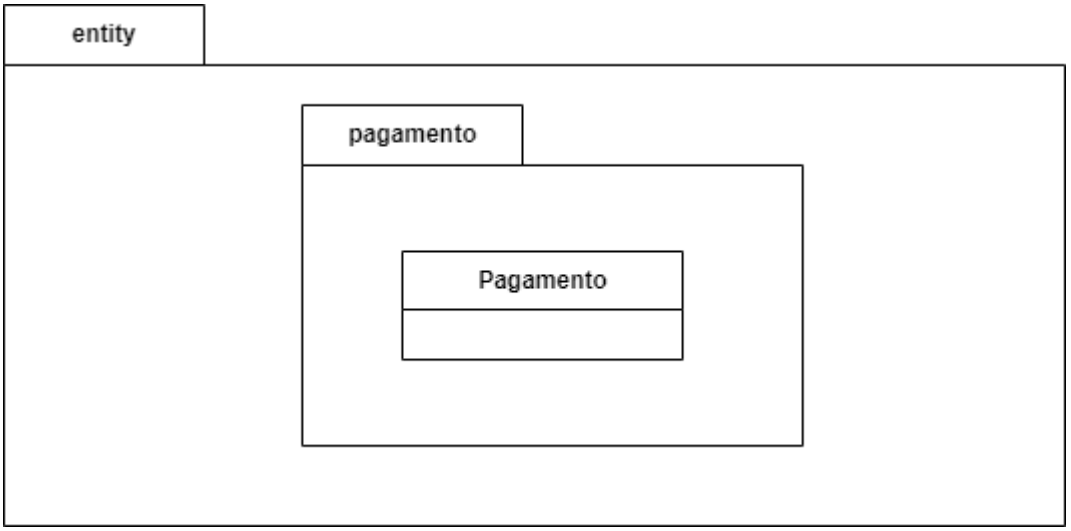
In questa sezione è mostrata la suddivisione in pacchetti del sistema a partire dall'architettura e dai sottosistemi descritti nel System Design Document. Si è voluto creare per ogni sottosistema un package ed un pacchetto "model" per gestire l'accesso al Database e i dati presenti in esso.

- .idea
- src
    - main
        - webapp
            - static (css…)
        - java, contiene le classi Java relative alle componenti Control e Model
    - test
        - java, contiene le classi Java per l'implementazione el testing
- target

Package Tommit



Package entity



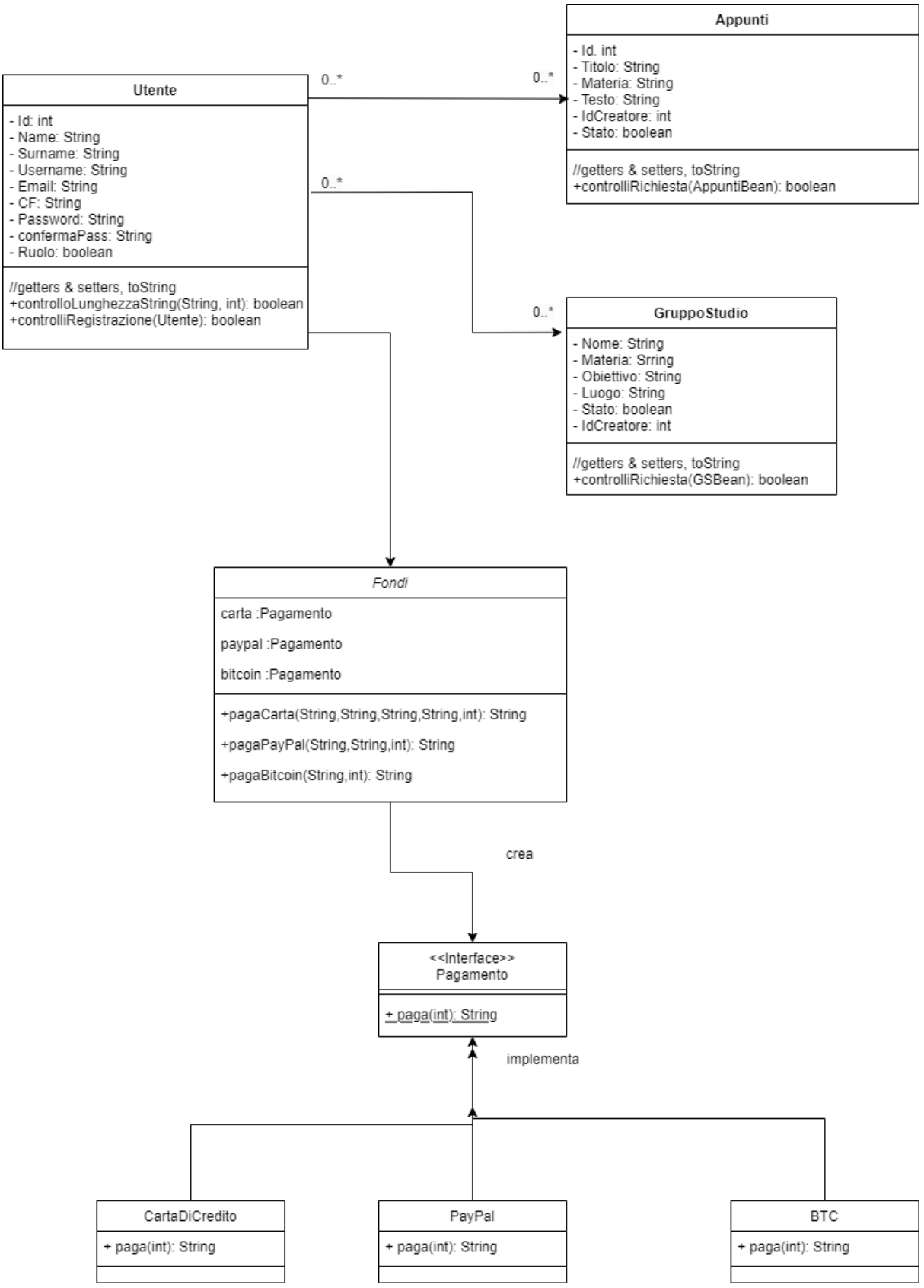
4. Class Interfaces

Di seguito sarà presentata l'interfaccia utilizzata per implementare le donazioni.

Nome classe	Pagamento
Descrizione	Gestisce l'operazione relativa al pagamento.
Metodo	+Paga(int importo): String

Nome Metodo	+Paga(int importo): String
Descrizione	Questo metodo consente il pagamento.
Pre-condizione	Utente loggato
Post-condizione	/

3. Class Diagram Ristrutturato



5. *Design Pattern*

Per rappresentare al meglio la funzione di “donazione” è stato deciso di utilizzare il pattern Facade.Il Facade nasconde le complessità del sistema e fornisce un’interfaccia al client mediante la quale può accedere al sistema.
Questo tipo di design pattern rientra nella tipologia di pattern **Strutturali** poiché aggiunge un’interfaccia al sistema esistente per nascondere la sua complessità.
Le classi parte
cipanti al Facade sono:
Pagamento: interfaccia che rappresenta il servizio di pagamento.
CartaDiCredito: tipologia di pagamento.
Paypal: tipologia di pagamento.
Bitcoin: tipologia di pagamento.
Fondi: delega la creazione degli oggetti alle relative classi e gestisce il lavoro.