

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

Dipartimento di Informatica

PROGETTO OBJECTED ORIENTATION:

GESTORE ToDo

Anno Accademico 2024/2025

Studenti :

* PETRUCELLI ALDO N86005288,
* STARACE ANDREA N86005271
* RIGIONE ANTONIO N86005292

ANALISI DEI REQUISITI:

Questa sezione serve a stabilire quali requisiti deve avere la base di dati a partire dalla richiesta della traccia. É importante trovare le entità e le relazioni che definiranno in modo esaustivo il dominio del problema.

L'obiettivo è quello di modellare un sistema informativo in Java per la gestione delle attività personali (ToDo), strutturato attorno al concetto di bacheche, attività condivise e checklist.

Di seguito sono elencate le entità principali definite con i relativi attributi e metodi:

-**UTENTE:**

La classe UTENTE rappresenta l'accesso al sistema. L'utente è identificato da uno username e da una password (entrambe chiavi primarie, in quanto univoche).

**Attributi:**

•username: String → identificativo univoco dell'utente (PK).

•password: String → password univoca per l'autenticazione (PK).

**Metodi:**

• registrazione(): registra un nuovo utente nel sistema.

• login(): verifica username e password.

• crea\_bacheca(): consente di creare una nuova bacheca associata all'utente.

• cerca\_toDo(): ricerca un ToDo specifico per titolo.

• check\_scadenze(): restituisce un elenco di ToDo in scadenza relative alla data inserita dall’utente.

• sposta\_toDo(): sposta un toDo da una bacheca all’altra.

**-BACHECA:**

La BACHECA è una raccolta di ToDo creati da un utente, e può essere classificata tramite l'enumerazione TIPO\_BACHECA.

**Attributi**:

•titolo: enumerazione grazie alla classe enumerativa TIPO\_BACHECA → stringa che classifica Ie tre possibili tipologie di bacheca (università, lavoro, tempo libero) (PK).

•descrizione: String → spiegazione opzionale della bacheca.

**Metodi**:

•modifica(): modifica titolo o descrizione.

•delete(): elimina una bacheca.

•crea\_toDo(): crea un ToDo all’interno della bacheca.

•elimina\_toDo(): elimina un ToDo.

•modifica\_posizione(): cambia l'ordine di visualizzazione dei ToDo.

**-TIPO\_BACHECA:**

Enumerazione utilizzata per indicare il tipo di bacheca. I valori sono fissi e riflettono le principali tipologie di bacheca.

• università

• lavoro

• tempo\_libero

-**TO\_DO:**

La classe TO\_DO rappresenta un’attività personale da svolgere. Include informazioni di tipo testuale, visivo e di scadenza. Contiene inoltre meccanismi di condivisione e gestione dello stato.

**Attributi**:

• titolo\_toDo: String → titolo dell'attività.

• URL: String → link correlato all’attività.

• data\_scad: String → data di scadenza.

• descrizione: String → descrizione estesa.

• immagine: String → percorso immagine.

• id\_toDo: int → identificativo univoco (PK).

• colore\_sfondo: String → colore dello sfondo dell'interfaccia grafica.

• stato\_toDo: boolean → stato di completamento del toDo (true se completato).

• utenti\_condivisi: Stringlist → elenco di utenti con cui è condiviso il toDo.

• ordine: int → posizione nella bacheca.

• username: String → autore del ToDo (FK, collegato all’utente).

**Metodi**:

• modifica\_titolo(): modificha il titolo del toDo.

• modifica\_colore(): cambia il colore dello sfondo dell’interfaccia grafica.

• modifica\_ordine(): cambia la posizione nell’elenco dei toDo.

• change\_stato(): cambia lo stato del toDo tra completato e non completato.

• modifica\_condivisi(): funzione di modifica e cancellazione della condivisione, applicabile solo se l’autore del toDo è uguale all’username.

• leggi\_lista(): funzione che permette di leggere la lista degli utenti con cui è stato condiviso il toDo (concessa a tutti).

• modifica\_descrizione(): modifica della descrizione del toDo.

• modifica\_URL(): modifica dell’URL .

• modifica\_data(): modifica la data di scadenza del toDo.

• modifica\_image(): modifica l’immagine legata al toDo.

• set\_colore(): modifica il colore dello sfondo del toDo.

• check\_titolo(): cambia il colore del titolo del toDo in rosso se il toDo è scaduto.

-**CHECKLIST**

Rappresenta una sotto-attività associata a un ToDo.

**Attributi**:

• nome: String → nome dell’attività secondaria.

• stato: boolean → stato di completamento dell’attività true se completato.

**Metodi**:

• check\_stato(): aggiorna lo stato del ToDo padre se sono stati completati tutti I toDo figli.

• change\_state(): cambia lo stato del toDo nella check list.

ANALISI DELLE RELAZIONI:

Una volta identificate le classi primarie sono state definite le relazioni tra le classi.

- Utente → Bacheca: relazione 1:3 (un utente ha un massimo di 3 bacheche).

- Bacheca → ToDo: relazione 1:N (una bacheca ha più ToDo).

- ToDo → CheckList: relazione 1:N (un ToDo ha più elementi di checklist).

- ToDo ↔ Utente (condivisione): relazione N:N simulata tramite lista di nomi utenti con cui è condiviso.

CONCLUSIONE:

Il class diagram ottenuto mediante l’analisi della traccia fornisce una struttura coerente. Le classi rappresentano in modo chiaro le entità coinvolte e le relazioni tra di esse.