

# Progetto "Gestore ToDo"

Università degli studi di Napoli Federico II, anno  
accademico 2025/2026

Corso di Object Orientation, DIETI-L32



Autori: Oliva Sabrina N86004167  
Quaranta Désirée N86004705  
Russo Bartolomeo N86005210

Traccia: 1

# Indice

<b>1</b>	<b>INTRODUZIONE AL PROGETTO</b>	<b>2</b>
1.1	Traccia . . . . .	2
1.2	Funzionalità . . . . .	2
1.3	Analisi dei requisiti . . . . .	2
1.4	Relazioni: . . . . .	3
<b>2</b>	<b>DIAGRAMMA UML DELLE CLASSI DEL DOMINIO DEL PROBLEMA</b>	<b>4</b>
<b>3</b>	<b>DIAGRAMMA UML DI DETTAGLIO DELLE CLASSI DEL DOMINIO DELLA SOLUZIONE</b>	<b>5</b>
<b>4</b>	<b>SEQUENCE DIAGRAM DELLE FUNZIONI AGGIUNTIVE</b>	<b>6</b>
4.1	EliminaTaskSequence . . . . .	6
4.2	CaricaDatiCondivisi . . . . .	7
<b>5</b>	<b>INTERFACCE GUI</b>	<b>8</b>
5.1	Pagina di login e registrazione utente. . . . .	8
5.2	La Dashboard. . . . .	9
5.3	La Checklist. . . . .	10
5.4	Condivisione delle task. . . . .	12

# 1 INTRODUZIONE AL PROGETTO

## 1.1 Traccia

La traccia a noi assegnata prevedeva di sviluppare un sistema informativo composto da un database relazionale e un applicativo Java dotato di GUI, che consentisse all'utente di gestire e tenere traccia delle proprie attività, chiamato "ToDo"; l'applicativo appena citato doveva inoltre essere realizzato ispirandosi al software Trello.

## 1.2 Funzionalità

Il software consente all'utilizzatore di organizzare e gestire le proprie attività, suddividendole in bacheche (che può modificare ed eliminare liberamente) al cui interno sono presenti le varie task (i ToDo), altrettanto modificabili, secondo un ordine dettato dalle preferenze dell'utente.

L'utente, oltre a poter modificare in base alle proprie esigenze l'ordine, il nome, la data e i link collegati alle attività, può anche condividerle con altri utenti; quando si verificherà una tale situazione, l'utente con cui è stato condiviso il ToDo potrà visualizzarne autore e contenuto sulla sua bacheca.

L'utilizzatore del software può inoltre personalizzare l'interfaccia grafica secondo le sue preferenze e "spuntare" le attività completate; nel caso in cui un ToDo sia in scadenza, sarà stesso il sistema a ricordarlo all'utente se questi lo desidera, mostrando l'elenco di tutti i ToDo non ancora spuntati entro un certo giorno prestabilito dall'utilizzatore.

Infine l'utente potrà cercare il titolo del ToDo che desidera e qualora quest'ultimo dovesse apparire in rosso, starà ad indicare che l'attività in questione non è stata eseguita entro la scadenza prevista.

## 1.3 Analisi dei requisiti

Per progettare un applicativo in Java per la realizzazione del software "ToDo" abbiamo come prima cosa, individuato le seguenti classi:

Entità	Attributi	Metodi	Descrizione
Checklistitem	<u>id</u> ; taskId; name; isCompleted.	Non vi sono metodi individuati nella traccia per questa classe.	Costituisce un elemento di una checklist associato ad una task: ciascuna task può avere una lista di elementi di checklist per la suddivisione delle attività complesse in sotto-attività più facilmente gestibili.
Task	<u>id</u> ; <i>userId</i> ; title; description; deadline; category; status.	Non vi sono metodi individuati nella traccia per questa classe.	Descrive una task/attività nel sistema TaskManager.
Utente	<u>id</u> ; username;	aggiungitask(); condividitask(); eliminatask().	Rappresenta l'utilizzatore del Task Manager.

Bacheca	<u>id</u> ; <i>userId</i> ; titolo.	Non vi sono metodi individuati nella traccia per questa classe.	Identifica una bacheca all'interno del Task Manager: la bacheca raggruppa le task di un utente.
---------	---	---	---

## 1.4 Relazioni:

**GESTISCE**(*Utente*, *Task*): un utente può gestire in una bacheca una o più task mentre una task, nonostante possa essere condivisa, può essere gestita solo da un utente [1...N].

**INCLUDE** (*Task*, *Checklistitem*): una task può includere zero o più elementi della checklist, mentre un singolo elemento della checklist appartiene esclusivamente a una task specifica. [1...N].

**CONDIVIDE**(*Utente*, *Task*): una task può essere condivisa con zero o più utenti aggiuntivi, mentre un utente può partecipare a zero o più task condivise da altri [N...N].

**Modello tramite tabella di associazione:**

\* *userId* (PK,FK da Utente);

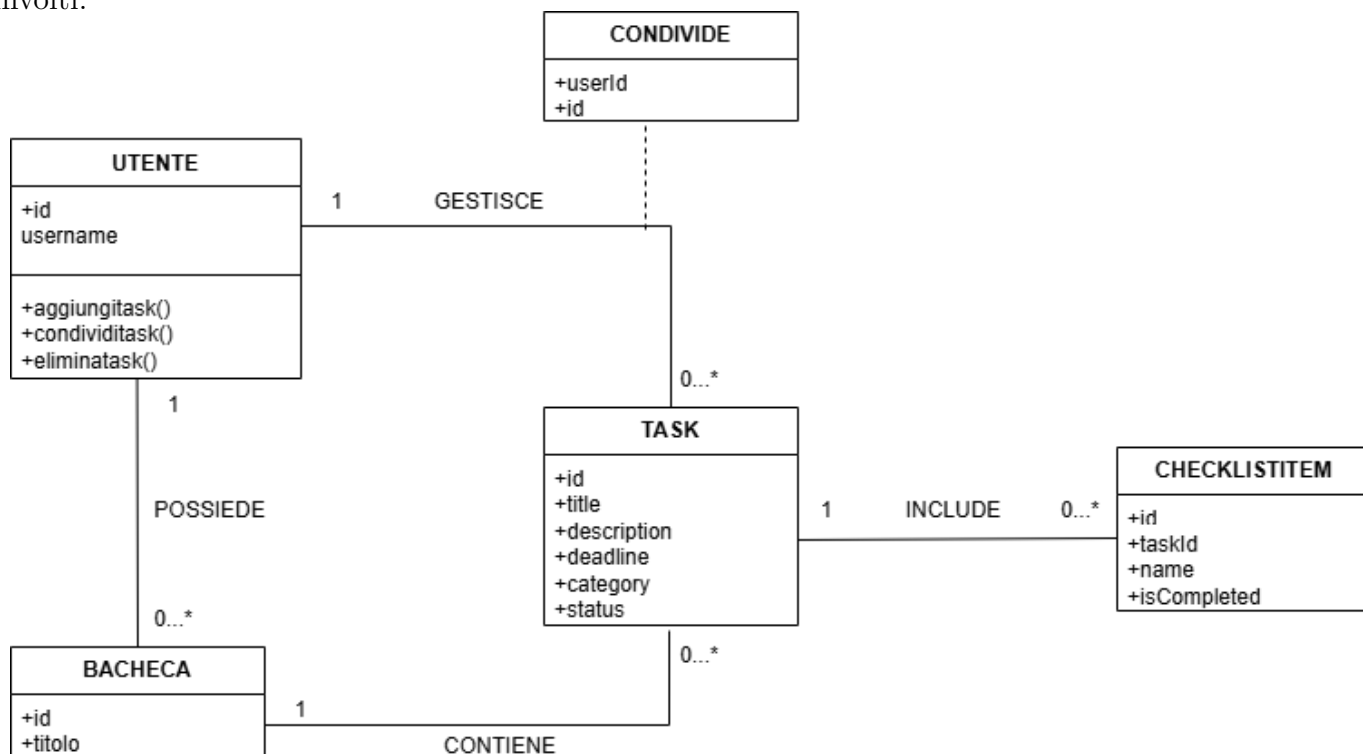
\* id (PK, FK da Task).

**POSSIEDE**(*Utente*, *Bacheca*): un utente possiede e gestisce una o più bacheche, mentre una bacheca appartiene a un singolo utente [1...N].

**CONTIENE**(*Bacheca*, *Task*): una bacheca può contenere al suo interno zero o più task, mentre un singolo task è posizionato in una sola bacheca alla volta [0...N].

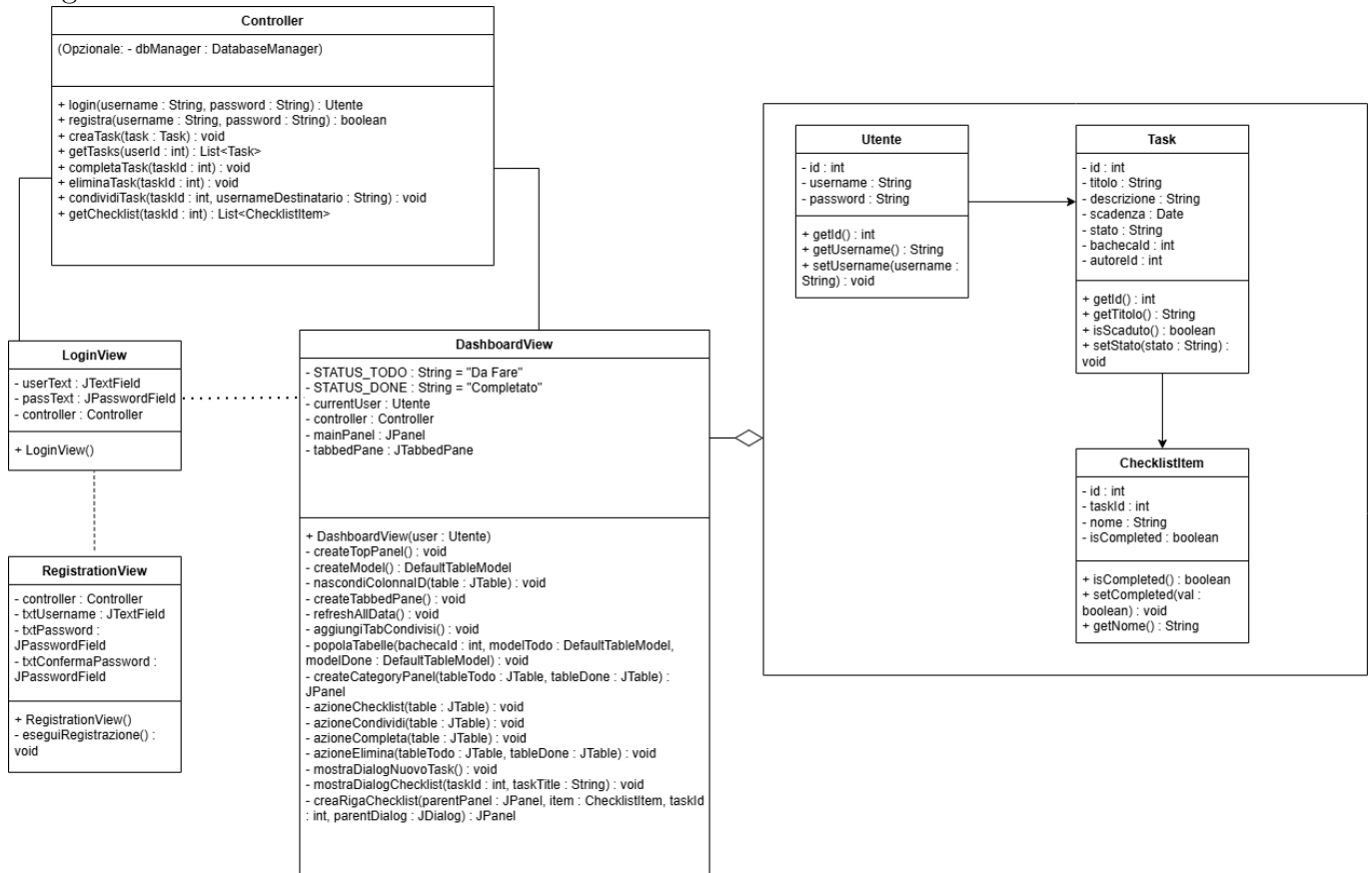
## 2 DIAGRAMMA UML DELLE CLASSI DEL DOMINIO DEL PROBLEMA

In questa sezione è possibile osservare il diagramma ottenuto dalle relazioni viste nel precedente capitolo e realizzato secondo il formalismo UML: attraverso questa schematizzazione è possibile modellare il dominio del problema e quindi analizzare dal punto di vista concettuale il funzionamento dell'applicativo, evidenziandone le entità, gli attributi ad esse correlati e le relazioni tra gli oggetti coinvolti.



### 3 DIAGRAMMA UML DI DETTAGLIO DELLE CLASSI DEL DOMINIO DELLA SOLUZIONE

Il diagramma che segue invece rappresenta il dominio della soluzione e descrive l'aspetto logico dell'architettura del sistema: questo schema permette di osservare in maniera approfondita le principali classi software usate nell'applicativo, i loro attributi, i loro metodi e le relazioni tra di esse. Questo diagramma, realizzato anche questo seguendo il formalismo UML, ci permette di avere una corrispondenza tra la parte concettuale e la parte pratica del programma costituita dall'interfaccia utente, dalla logica di funzionamento e dall'accesso ai dati.

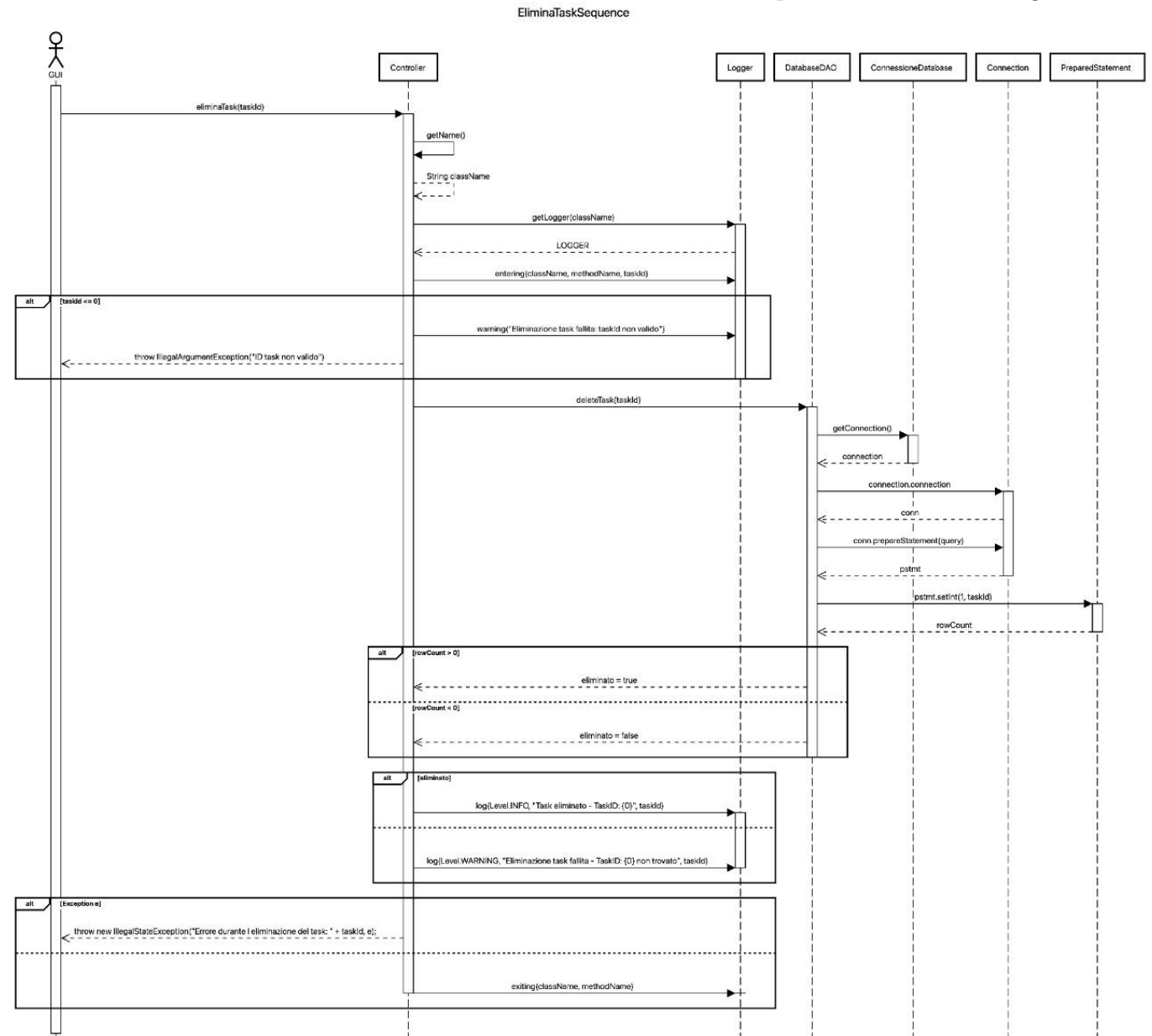


## 4 SEQUENCE DIAGRAM DELLE FUNZIONI AGGIUNTIVE

Di seguito sono riportati i due sequence diagram che hanno lo scopo di rappresentare con maggior dettaglio alcune delle funzionalità dell'applicativo: si è scelto di includere questo tipo di diagramma in quanto è in grado di descrivere l'interazione dinamica tra gli oggetti al manifestarsi di specifici casi d'uso. Ciò ci consente di avere una rappresentazione chiara di come il sistema risponde al verificarsi di determinate situazioni.

### 4.1 EliminaTaskSequence

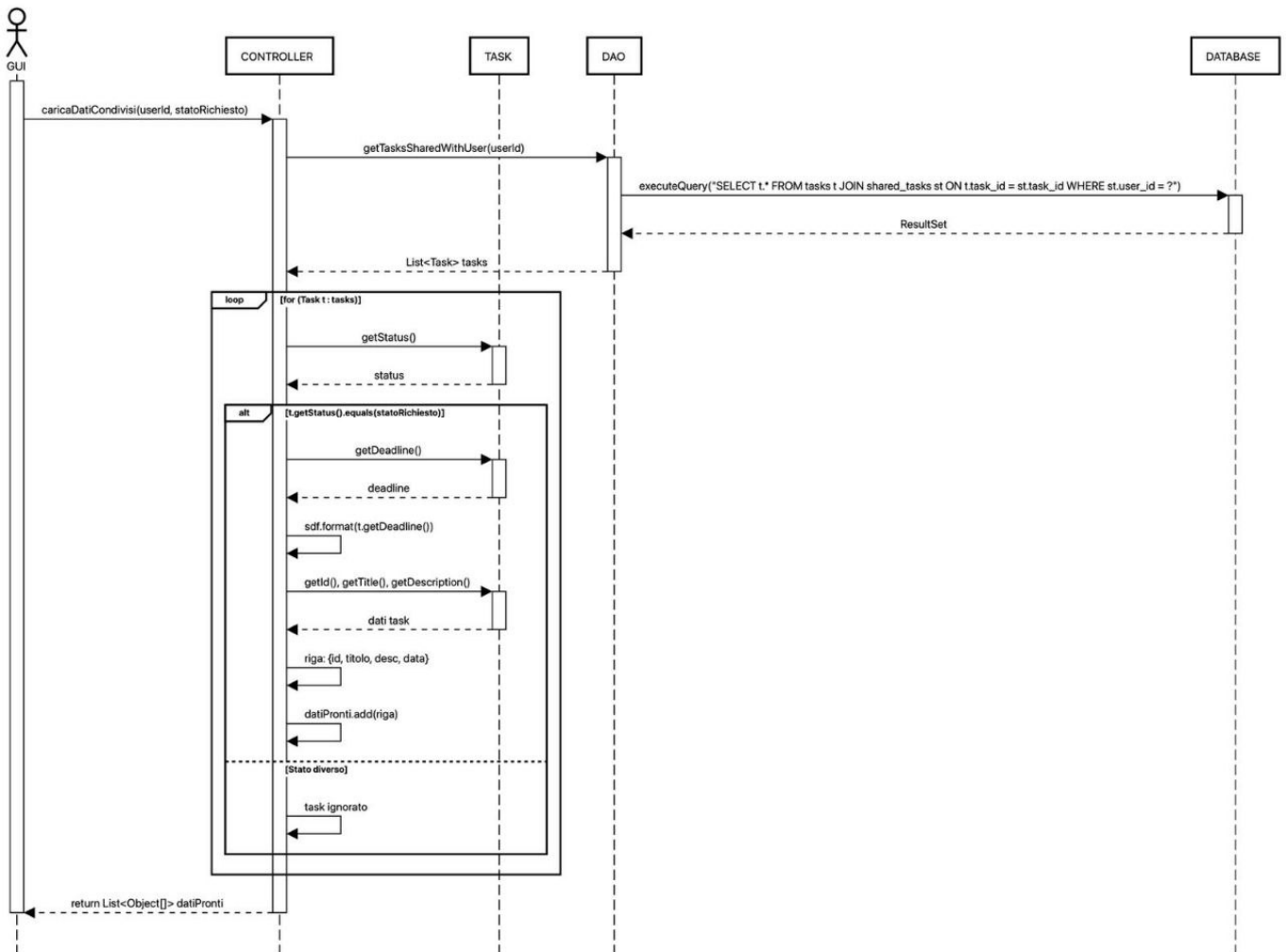
EliminaTask è un metodo CRUD del Controller per l'eliminazione delle task: esso consente all'utente di eliminare una task "da fare", "completata" o condivisa da un altro utente e parte quando viene cliccato il tasto "Elimina" dopo aver selezionato la task nella Dashboard. Il diagramma sottostante descrive le interazioni tra le diverse componenti del progetto: Controller, Logger, che acquisisce i record di registro per eventuali warning, ad esempio un caricamento fallito della task o un caricamento avvenuto con successo specificando l'userID, il DAO per il retrieval dei dati salvati nel database il quale a sua volta connette servendosi dell' URL, dello username e della password a task\_manager\_db.



## 4.2 CaricaDatiCondivisi

Questo metodo ha la responsabilità di recuperare, filtrare e formattare le task che sono state condivise con un determinato utente, preparandole per la visualizzazione direttamente nella tabella grafica, la Jtable, dell'interfaccia utente. Una volta condivisa una task con un altro utente presente nel sistema, quest'ultima dovrà trovarsi nella sezione "condivisi con me". A questo punto il Controller riceve l'INPUT, valida l'Id utente, chiama il DAO, che esegue la query specifica convertendo il ResultSet SQL in oggetti Task, e processa i dati ricevuti. Il Database esegue quindi la query inviata dal DAO e restituisce le righe trovate. Il Task invece rappresenta il Model e viene interrogato dal Controller tramite i metodi getter, getStatus, getDeadline, getId, getTitle, getDescription, per fornire le informazioni necessarie al filtraggio e alla visualizzazione.

caricaDatiCondivisi Sequence

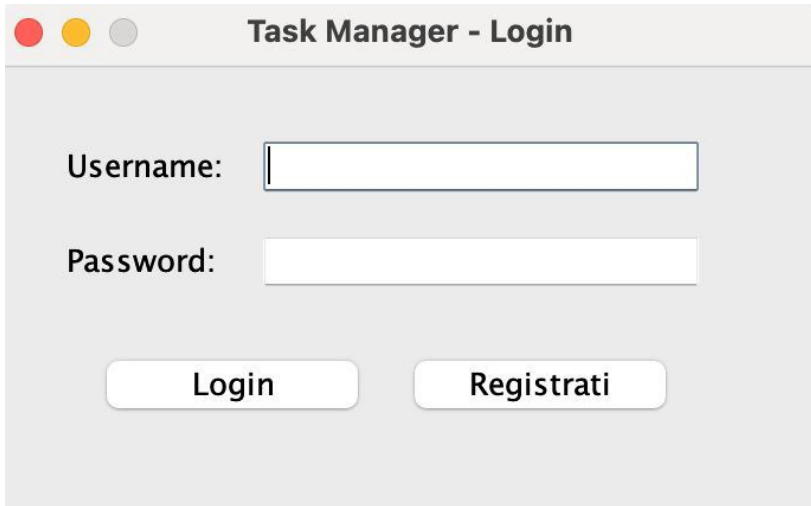




## 5 INTERFACCE GUI

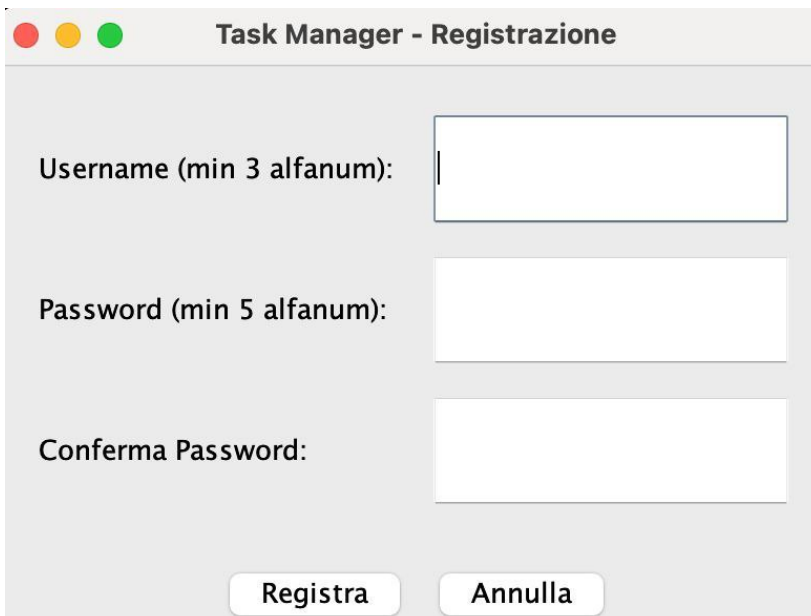
### 5.1 Pagina di login e registrazione utente.

La prima pagina che compare all'avvio del task manager è la schermata di login, dove l'utente tramite l'uso delle sue credenziali, username e password, può accedere all'applicativo.



The image shows a window titled "Task Manager - Login". It has a standard macOS-style title bar with red, yellow, and grey buttons. The window contains two text input fields: "Username:" and "Password:". Below these fields are two buttons: "Login" and "Registrati".

Nel caso in cui l'utente non abbia mai effettuato accesso all'applicativo potrà registrarsi scegliendo un nome utente ed una password che dovrà essere di minimo cinque caratteri alfanumerici e dovrà essere reinserita al fine di evitare errori di battitura.



The image shows a window titled "Task Manager - Registrazione". It has a standard macOS-style title bar with red, yellow, and green buttons. The window contains three text input fields: "Username (min 3 alfanum):", "Password (min 5 alfanum):", and "Conferma Password:". Below these fields are two buttons: "Registra" and "Annulla".

# 5.2 La Dashboard.

Nell'immagine sottostante possiamo vedere due esempi dei due riquadri dove vanno inserite le task da fare, con relativa descrizione e scadenza e quelle completate.

Task completata:

Task Manager - teoTeopoli

Nuovo Task

Aggiorna

Esci

Università

Lavoro

Tempo Libero

Condivisi con me

DA FARE

Titolo	Descrizione	Scadenza
--------	-------------	----------

COMPLETATO

Titolo	Descrizione	Scadenza
esame OO	scritto	21-01-2026

Vedi Checklist

Condividi

Sposta a Completato

Elimina

Task da fare:

Task Manager - teoTeopoli

Nuovo Task

Aggiorna

Esci

Università

Lavoro

Tempo Libero

Condivisi con me

DA FARE

Titolo	Descrizione	Scadenza
call informativa	capire da farsi	22-01-2026

COMPLETATO

Titolo	Descrizione	Scadenza
--------	-------------	----------

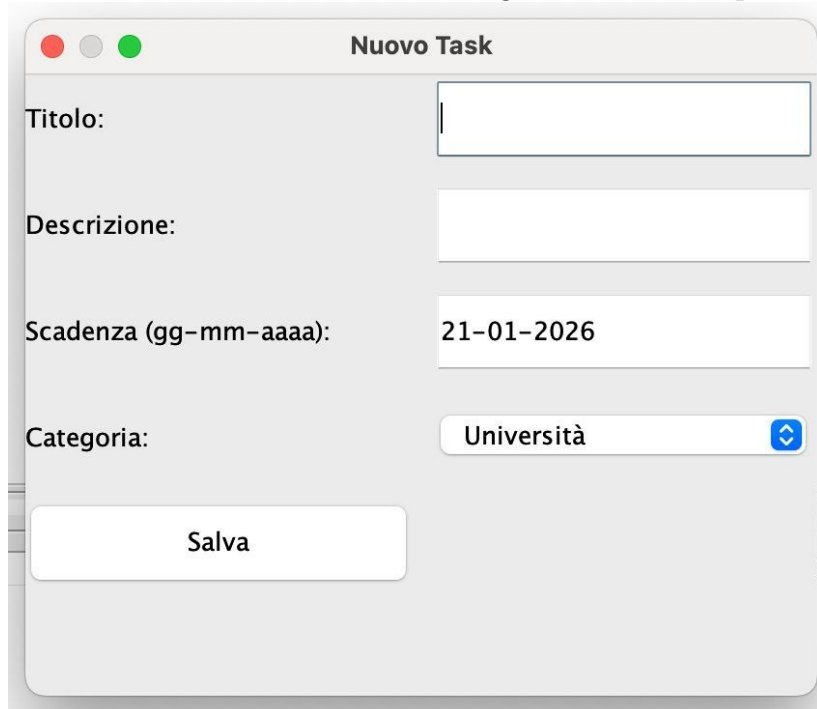
Vedi Checklist

Condividi

Sposta a Completato

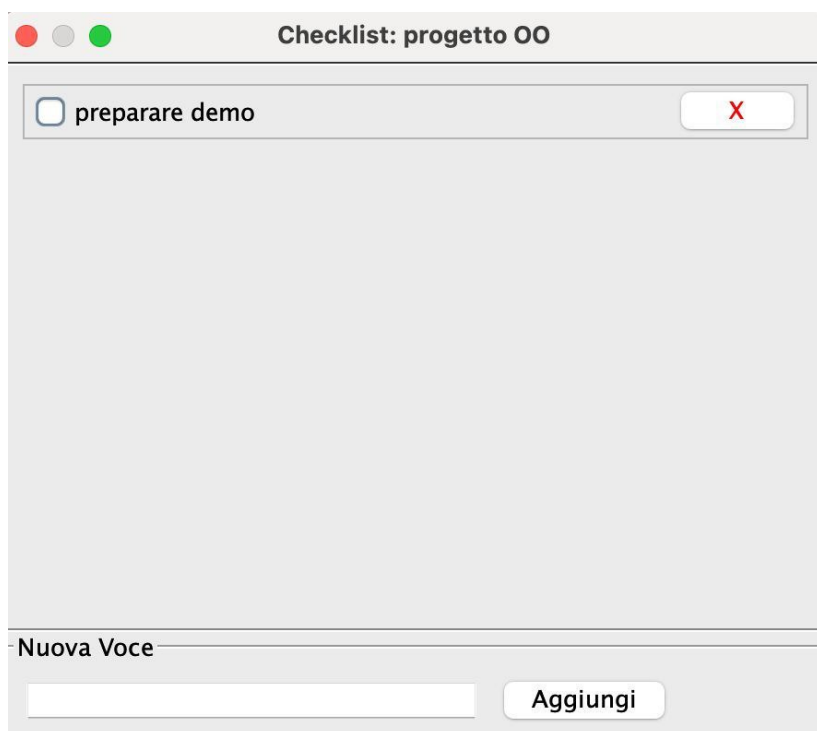
Elimina

Per creare una nuova task invece si può sfruttare la finestra sottostante che consente anche di andare ad inserire l'attività in una categoria secondo la quale verrà raggruppata:



### 5.3 La Checklist.

La Checklist invece è rappresentata da una finestra che consente di spuntare le attività e di aggiungere nuove voci alla raccolta.



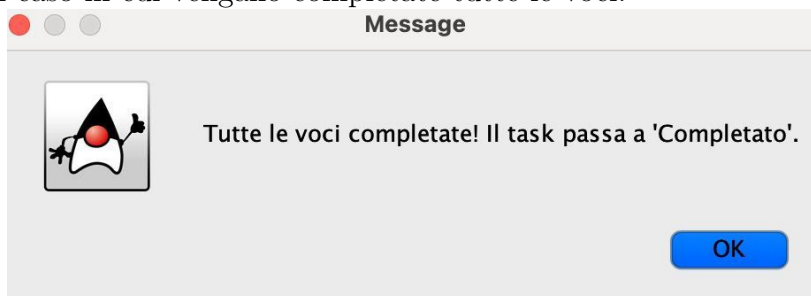
Checklist: progetto OO

<input checked="" type="checkbox"/> preparare demo	X
<input type="checkbox"/> review documentazione	X

Nuova Voce

Aggiungi

Nel caso in cui vengano completate tutte le voci:

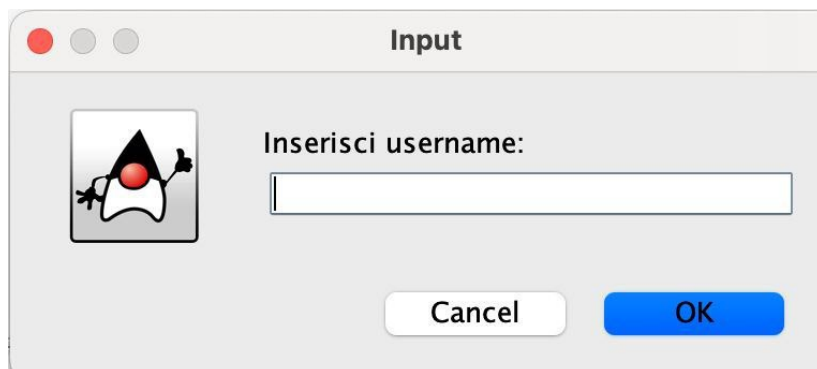


Nel caso in cui si voglia eliminare una delle voci:



## 5.4 Condivisione delle task.

Per condividere una task con un altro utente basterà inserire l'username del destinatario: a condivisione avvenuta, apparirà una schermata di conferma.



Nel caso in cui si vogliano vedere le task condivise:

