

ARCHITETTURA DEL SISTEMA

Gestione libro

Questo modulo si occupa della gestione della biblioteca per quanto riguarda l'archiviazione dei libri.

In questo modulo troviamo diverse classi: quella Libro che ha lo scopo di identificare una singola entità libro, la classe Autore e la classe per la gestione della biblioteca.

Gestione utente

Anche questo modulo presenta una divisione in classi simile al modulo per la gestione dei libri e dei prestiti quindi anche qui troviamo una classe che identifica la singola entità in questo caso la classe Utente ed una classe che gestisce l'archiviazione di più utenti.

Gestione prestiti

Il modulo ha lo scopo di gestire la parte che riguarda i prestiti effettuati.

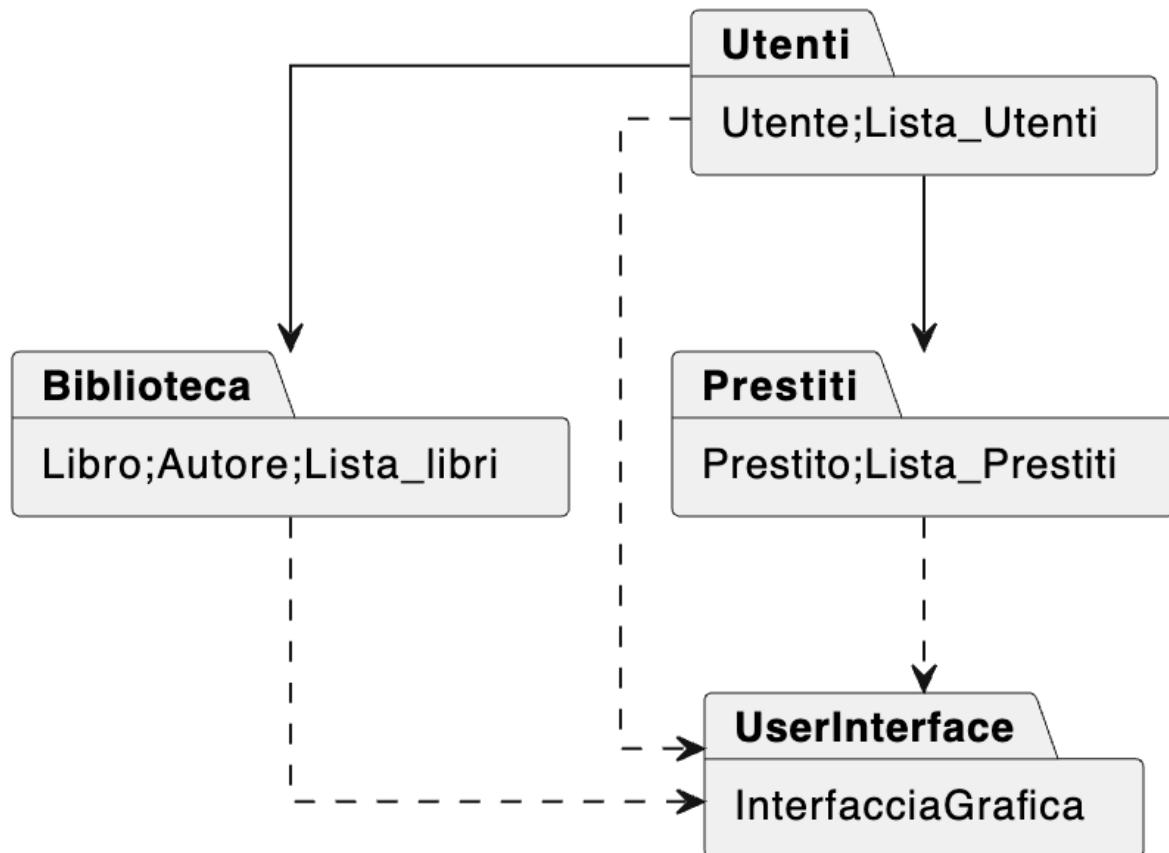
Segue una divisione in 2 classi quella che gestisce l'entità prestito e quella che gestisce la collection di Prestito.

Interfaccia GUI

Il modulo gestisce tutte le interazioni con il Bibliotecario tramite interfaccia grafica.

Il modulo viene diviso in diverse classi per mantenere la divisione in moduli descritta in precedenza.

Dipendenze



Come mostra il grafico tutte le classi comunicano con la User Interface che effettua le chiamate dei metodi e acquisisce i dati inseriti dal Bibliotecario.

Il modulo Utenti e Biblioteca no comunicano direttamente ma passano dati al modulo Prestiti.

MODELLO STATICO

Classe Libro

La classe libro come detto in precedenza gestisce l'entità libro memorizzando il titolo del libro l'anno di pubblicazione, l'ISBN, una lista di oggetti Autore, il numero di copie disponibili e una flag che permette ad un libro di non essere eliminato nonostante tutte le copie siano state prestate.

Inoltre ci sono i tre comparatori per gestire i diversi ordinamenti della lista dei libri

Attributi

```
private String titolo  
private List<Autore> autori  
private String ISBN  
private int anno  
private boolean disponibile  
private int copie  
public static final Comparator<Libro> TITOLO  
public static final Comparator<Libro> ISBN  
public static final Comparator<Libro> PRIMO_AUTORE
```

Metodi

Tralasciando i metodi funzionali la classe mette a disposizione i metodi per la visualizzazione e la modifica di tutti gli attributi e inoltre permette la creazione di una collection di Autore per la gestione di più autori e tutti i metodi che permettono la modifica l'aggiunta e la rimozione di oggetti da suddetta collection.

```
public Libro(String titolo,List<Autore> autori,String ISBN,int anno,int copie)  
public String getTitolo()  
public void setTitolo(String titolo)  
public List<Autore> getAutori()  
public void setAutori(List<Autore> autori)  
public void modificaAutore(Autore autore)  
public String getISBN()  
public void setISBN(String ISBN)  
public int getAnno()  
public void setAnno(int anno)  
public int getCopie()  
public void setCopie(int copie)  
public void aggiungiAutore(Autore a)  
public void rimuovi Autore(Autore a)  
public boolean getDisponibile()  
public void setDisponibile(boolean disponibile)
```

```
public boolean equals(Object o)
public void OrdinaAutori()
```

Classe Autore

La classe Autore è una classe creata come scelta progettuale per semplificare la presenza di diversi autori di un solo libro.

In questa classe sono presenti solo il nome, il cognome e un comparatore per permettere l'ordinamento sul cognome nella classe Libro

Attributi

```
private String nome
private String cognome
public static final Comparator<Autore> COGNOME
```

Metodi

prevede i metodi per la visualizzazione e la modifica di un Autore

```
public Autore (String nome, String cognome)
public String getNome()
public String setNome(String nome)
public String getCognome()
public String setCognome(String cognome)
public static Comparator<Autore> getCOGNOME()
public boolean equals(Object o)
```

Classe Lista_Libri

In questa classe viene gestita la vera e propria struttura della biblioteca dal punto di vista dello storage dei libri.

Per fare questo è sufficiente un solo attributo la collection che gestisce l'aggregazione di libri

Attributi

```
private List<Libro> libri
```

Metodi

In questa classe sono presenti tutti i metodi per offrire i servizi espressi nei casi d'uso

```
public Libro aggiungiLibro(Libro libro)
public Libro rimuoviLibro(Libro libro)
public Libro modificaLibro(Libro libro)
public void modificaCopie(String ISBN, int n)
public List<Libro> cercaPerTitolo(String str)
public List<Libro> cercaPerISBN(String str)
public List<Libro> cercaPerAutore(String str)
```

```
public void ordina(Comparator<Libro> cmp)
public boolean checkISBN(String ISBN)
public void salvataggioLibri(String nomefile)
public Lista_Libri letturaLibri(String nomefile)
```

Classe Utente

Questa classe gestisce la singola entità utente, non comprendendo il Bibliotecario.
Come da requisiti l'utente viene identificato da 4 attributi:il nome,il cognome,la matricola e la mail universitaria

Attributi

```
private String nome
private String cognome
private String matricola
private String mail
```

Metodi

Comprende i metodi funzionali

```
public Utente(String nome, String cognome, String matricola, String mail)
public String getNome()
public void setNome(String nome)
public String getCognome()
public void setCognome(String cognome)
public String getMatricola()
public void setMatricola(String matricola)
public String getMail()
public void setMail(String mail)
public boolean equals(Object o)
public int hashCode()
```

Classe Lista_Utenti

La Lista_Utenti gestisce tutte le operazioni da effettuare sulla collection di Utente

Attributi

```
private List<Utente> utenti
```

Metodi

```
public String aggiungiUtente(Utente utente)
public String rimuoviUtente(Utente utente)
public Utente modificaUtente(Utente utente)
public List<Utente> getUtenti()
public void setUtenti(List<Utente> utenti)
public List<Utente> cercaPerMatricola(String str)
public boolean checkMatricola(String matricola)
```

```
public List<Utente> cercaPerMatricola(String str)
public List<Utente> cercaPerNomeCognome(String str)
public void salvataggioUtenti(String nomefile)
public void letturaUtenti(String nomefile)
```

Classe Prestito

Questa classe identifica un singolo prestito associando un oggetto di tipo Utente ad un ISBN di un libro inoltre aggiunge la data di restituzione del prestito e altri 2 attributi inseriti come scelta progettuale:L'ID univoco che identifica un prestito e il comparator usato da Lista_Prestiti per ordinare in base alla scadenza

Attributi

```
private String ISBN
private Utente utente
private LocalDate dataScadenza
private final int ID
private static int counter=0
private final static Comparator<Prestito> SCADENZA=Comparator.comparing(Prestito::getDataScadenza)
```

Metodi

I metodi implementati in questa classe prevedono la visione e la modifica degli attributi e i metodi funzionali

```
public Prestito(String ISBN,int Matricola,LocalDate dataScadenza)
public Comparator<Prestito> getComp()
public Libro getISBN()
public void setISBN(String ISBN)
public int getMatricola()
public void setMatricola(int Matricola)
public int getID()
public LocalDate getDataScadenza()
public void setDataScadenza(LocalDate dataScadenza)
public boolean equals(Object o)
```

Classe Lista_Prestiti

Questa classe gestisce la collection di Prestito e tutte le funzioni ad esso collegato

Attributi

```
private List<Prestito> prestiti
```

Metodi

Tutti i metodi presenti in questa classe si occupano di gestire l'aggiunta la rimozione e l'ordinamento dei prestiti e dei ritardi

```
public List<Prestito> getPrestiti()
```

```
public void setPrestiti(List<Prestito> prestiti)
public String aggiungiPrestito(Prestito p)
public String rimuoviPrestito(Prestito p)
public List<Prestito> aggiornaRitardi()
public List<Prestito> cercaPerISBN(String ISBN)
public List<Prestito> cercaPerID(int ID)
public List<Prestito> cercaPerMatricola(int Matricola)
public List<Prestito> cercaPerData(LocalDate dataScadenza)
public void OrdinaLibriPerScadenza(Comparator<Prestito> cmp)
public String ModificaPrestito(Prestito p, Prestito l)
public void salvataggioPrestiti(String nomefile)
public void letturaPrestiti(String nomefile)
public boolean checkUtente(Prestito p)
```

ControllerPortale

Attributi

```
private Button btnEntra
```

Metodi

```
public void initialize()
public void handleEntra(ActionEvent event)
```

ControllerHome

Attributi

```
private Button btnGestioneLibri
private Button btnGestionePrestiti
private Button btnGestioneUtenti
private Button btnPortale
private AnchorPane contentPane
private final Lista_Utenti listaUtenti = new Lista_Utenti();
private final Lista.Libri listaLibri = new Lista.Libri();
private final Lista.Prestiti listaPrestiti = new Lista.Prestiti();
private static final String FILE_UTENTI = "FileEsternoUtenti.bin";
private static final String FILE_LIBRI = "FileEsternoLibri.bin";
private static final String FILE_PRESTITI = "FileEsternoPrestiti.bin";
```

Metodi

```
private void initialize()
private void leggiTutto()
private void salvaTutto()
private void onBackToPortale()
private void onGestioneLibri()
private void onGestionePrestiti()
private void onGestioneUtenti()
```

ControllerDashboard

Attributi

```
private Button btnPortale
```

Metodi

```
private void onBackToPortale()
```

ControllerGestioneLibri

Attributi

```
private Lista.Libri listalibri;
final String FILE_LIBRI = "FileEsternoLibri.bin";
@FXML
private Button btnModificaLibro;
private Button btnRimuoviLibro;
private Button btnVisualizzaLibri;
private Button btnAggiungiLibro;
private TextField txtRicercaLibro;
private TableView<Libro> tabellaLibri;
private TableColumn<Libro, String> colTitolo;
private TableColumn<Libro, String> colAutore;
private TableColumn<Libro, String> colISBN;
private TableColumn<Libro, Integer> colAnno;
private TableColumn<Libro, Integer> colCopie;
private final ObservableList<Libro> data = FXCollections.observableArrayList();
private Button btnCercaLibro;
private Button btnResetFiltroLibri;
private Label lblConteggioLibri;
```

Metodi

```
public void setListaLibri(Lista.Libri listalibri)
@FXML private void initialize()
private void inizializzaColonne()
private void aggiornaTabella()
@FXML private void onBack(ActionEvent event)
@FXML private void onModificaLibro(ActionEvent event)
@FXML private void onRimuoviLibro(ActionEvent event)
@FXML private void handleVisualizzaLista(ActionEvent event)
@FXML private void handleAggiungiLibro(ActionEvent event)
@FXML private void handleCercaLibro(ActionEvent event)
@FXML private void handleResetFiltroLibri(ActionEvent event)
```

ControllerListaLibri

Attributi

```
private Lista.Libri listalibri;
final String FILE_LIBRI = "FileEsternoLibri.bin";
@FXML
private TableView<Libro> tabellaLibri
private TableColumn<Libro, String> colTitolo
private TableColumn<Libro, String> colAutore
private TableColumn<Libro, Integer> colAnno
private TableColumn<Libro, String> colISBN
private TableColumn<Libro, Integer> colCopie
private Button btnModificaLibro;
private Button btnRimuoviLibro;
private Button btnChiudi;
```

Metodi

```
@FXML
public void initialize()
private void onModificaLibro(ActionEvent event)
private void onRimuoviLibro(ActionEvent event)
private void handleChiudi()
```

ControllerModificaLibro

Attributi

```
private Lista.Libri listalibri;
private Libro libro
private final ObservableList<Autore> autoriTemp = FXCollections.observableArrayList();
final String FILE_LIBRI = "FileEsternoLibri.bin";
@FXML
private Button btnModificaLibro;
private TextField txtTitolo;
private TextField txtNomeAutore;
private TextField txtCognomeAutore;
private TextField txtAnno;
private TextField txtISBN;
private ListView<String> listAutori;
```

Metodi

```
@FXML public void setLibro(Libro libro)
@FXML public void setListaLibri(Lista.Libri listalibri)
private void refreshListaAutori()
@FXML public void initialize()
@FXML private void onAggiungiAutore()
@FXML private void onRimuoviAutore()
@FXML private void onModificaLibro(ActionEvent event)
@FXML private void onCancella()
@FXML private void chiudiFinestra()
```

ControllerPopup

Attributi

@FXML

```
private Label lblIcona;  
private Label lblTitolo;  
private Label lblMessaggio;  
private Button btnOk;
```

Metodi

```
public void setSuccesso(String messaggio)
```

```
public void setErrore(String messaggio)
```

@FXML private void onOk()

```
public static void showSuccess(Window owner, String messaggio);
```

```
public static void showError(Window owner, String messaggio);
```

```
private static void show(Window owner, boolean success, String messaggio);
```

ControllerAggiungiLibro

Attributi

private Lista.Libri listalibri;
final String FILE_LIBRI = "FileEsternoLibri.bin";

@FXML

```
private TextField txtTitolo;  
private TextField txtNomeAutore;  
private TextField txtCognomeAutore;  
private TextField txtAnno;  
private TextField txtISBN;  
private Button btnAggiungiLibro;  
private Button btnAggiungiAutore;  
private Button btnRimuoviAutore;  
private ListView<String> listAutori;  
private final ObservableList<Autore> autoriTemp = FXCollections.observableArrayList();
```

Metodi

@FXML

```
public void initialize()  
private void onAggiungiAutore()  
private void onRimuoviAutore()  
private void onAggiungiLibro(ActionEvent event)  
private void pulisciCampiLibro()  
private void onCancella()  
private void chiudiFinestra()
```

ControllerGestioneUtenti

Attributi

```
private Lista_Utenti listaUtenti;
private ObservableList<Utente> masterData;
private FilteredList<Utente> filteredData;
@FXML
private TextField txtRicercaUtente;
private Button btnCercaUtente;
private Button btnResetFiltroUtenti;
private Button btnVisualizzaUtenti;
private Button btnAggiungiUtente;
private TableView<Utente> tabellaUtenti;
private TableColumn<Utente, String> colMatricola;
private TableColumn<Utente, String> colNome;
private TableColumn<Utente, String> colCognome;
private TableColumn<Utente, String> colEmail;
```

Metodi

```
@FXML
private void initialize()
private void configuraColonneSePresenti()
private void onBack()
private void handleAggiungiUtente(ActionEvent event)
private void handleVisualizzaUtenti(ActionEvent event)
private void handleCercaUtente(ActionEvent event)
private void handleResetFiltroUtenti(ActionEvent event)
```

ControllerListaUtenti

Attributi

```
final String FILE_UTENTI = "FileEsternoUtenti.bin";
private Lista_Utenti listaUtenti;
private ObservableList<Utente> data;
private TableView<Utente> tabellaUtenti
private TableColumn<Utente, String> colNome
private TableColumn<Utente, String> colCognome
private TableColumn<Utente, Integer> colMatricola
private TableColumn<Utente, String> colEmail
private Button btnRimuoviUtente;
private Button btnModificaUtente;
private Button btnChiudi;
```

Metodi

```
public void initialize()
private void configuraColonneSePresenti()
private void onModificaUtente(ActionEvent event)
private void onRimuoviUtente(ActionEvent event)
private void handleChiudi()
```

ControllerModificaUtente

Attributi

```
final String FILE_UTENTI = "FileEsternoUtenti.bin";
private Lista_Utenti listaUtenti;
private Utente utente;
private TextField txtNome;
private TextField txtCognome;
private TextField txtMatricola;
private TextField txtEmail;
private Button btnModificaUtente;
```

Metodi

```
public void setListaUtenti(Lista_Utenti listaUtenti)
public void setUtente(Utente utente)
private void onModificaUtente(ActionEvent event)
private void onCancella()
private void chiudiFinestra()
```

ControllerAggiungiUtente

Attributi

```
private static final String DOMINIO_EMAIL = "@studenti.unisa.it";
private static final String PREFISSO_MATRICOLA = "06127";
final String FILE_UTENTI = "FileEsternoUtenti.bin";
private Lista_Utenti listaUtenti;
private TextField txtNome;
private TextField txtCognome;
private TextField txtMatricola;
private TextField txtEmail;
private Button btnAggiungi;
private Button btnCancella;
```

Metodi

```
public void initialize()
private void onAggiungi()
private void onCancella()
private void chiudiFinestra()
```

ControllerGestionePrestiti

Attributi

```
private Button btnVisualizzaPrestiti;
private Button btnAggiungiPrestito;
private TextField txtRicercaPrestito;
private Button btnVisualizzaRitardi;
private Button btnCercaPrestito;
private Button btnResetFiltroPrestiti;
private TableView<Prestito> tabellaPrestiti;
private TableColumn<Prestito, String> colMatricola;
```

```
private TableColumn<Prestito, String> colID;
private TableColumn<Prestito, String> colISBN;
private TableColumn<Prestito, LocalDate> colDataScadenza;
private Lista_Prestiti listaPrestiti;
private Lista_Utenti listaUtenti;
private Lista.Libri listaLibri;
```

Metodi

```
public void setListaPrestiti(Lista_Prestiti listaPrestiti)
public void setListe(Lista_Prestiti lp, Lista_Utenti lu, Lista.Libri ll)
private void initialize()
    private void inizializzaColonne()
    private void aggiornaTabella()
private void onBack()
private void handleVisualizzaListaPrestiti(ActionEvent event)
private void handleAggiungiPrestito(ActionEvent event)
private void handleVisualizzaRitardi(ActionEvent event)
private void handleCercaPrestito(ActionEvent event)
private void handleResetFiltroPrestiti(ActionEvent event)
```

ControllerListaPrestiti

Attributi

```
final String FILE_PRESTITI = "FileEsternoPrestiti.bin";
final String FILE_LIBRI = "FileEsternoLibri.bin";
private Lista_Prestiti listaPrestiti;
private Lista.Libri listaLibri;
private Lista_Utenti listaUtenti;
private TableView<Prestito> tabellaPrestiti;
private Button btnModificaPrestito;
private Button btnRimuoviPrestito;
private Button btnChiudi;
private TableColumn<Prestito, Integer> colID;
private TableColumn<Prestito, String> colISBN;
private TableColumn<Prestito, String> colMatricola;
private TableColumn<Prestito, LocalDate> colDataScadenza;
```

Metodi

```
public void setListe(Lista_Prestiti lp, Lista_Utenti lu, Lista.Libri ll)
private void initialize()
    private void inizializzaColonne()
    private void aggiornaTabella()
private void onModificaPrestito(ActionEvent event)
private boolean cambiaCopieLibro(String isbn, int delta)
private void onRimuoviPrestito(ActionEvent event)
private void handleChiudi()
```

ControllerModificaPrestito

Attributi

```
final String FILE_PRESTITI = "FileEsternoPrestiti.bin";
final String FILE_LIBRI = "FileEsternoLibri.bin";
    private Lista_Prestiti listaPrestiti;
    private Lista_Utenti listaUtenti;
    private Lista.Libri listaLibri;
    private Prestito prestitoOriginale;
private Button btnModificaPrestito;
private TextField txtISBN;
private TextField txtMatricola;
private TextField txtEmail;
private DatePicker dateScadenza;
```

Metodi

```
public void setListe(Lista_Prestiti lp, Lista_Utenti lu, Lista.Libri ll)
public void setPrestitoDaModificare(Prestito p)
private void onModificaPrestito()
private void onCancella()
private void chiudi()
```

ControllerAggiungiPrestito

Attributi

```
final String FILE_PRESTITI = "FileEsternoPrestiti.bin";
    final String FILE_LIBRI = "FileEsternoLibri.bin";
    private boolean prestitoCreato = false;
private Lista_Prestiti listaPrestiti;
private Lista_Utenti listaUtenti;
    private Lista.Libri listaLibri;
private TextField txtISBN;
private TextField txtMatricola;
private TextField txtEmail;
private DatePicker dateScadenza;
private Button btnAggiungi;
private Label lblPrefisso;
```

Metodi

```
public boolean isPrestitoCreato()
public void setListe(Lista_Prestiti p, Lista_Utenti u, Lista.Libri l)
public void initialize()
private void onAggiungi()
private void onCancella()
private void chiudiFinestra()
```

ControllerListaRitardi

Attributi

```
private Lista_Prestiti listaPrestiti;
private Lista_Utenti listaUtenti;
    private Lista.Libri listaLibri;
private TableView<Prestito> tabellaRitardi;
```

```

private TableColumn<Prestito, String> colISBN
private TableColumn<Prestito, Utente> colUtente
private TableColumn<Prestito, Integer> colID
private TableColumn<Prestito, LocalDate> colDataScadenza
private TableColumn<Prestito, String> colMatricola;
private Button btnChiudi;
private TextField txtRicercaRitardo;
private Button btnCercaRitardo;
private Button btnResetRitardi;
private Button btnModificaRitardo;
private Button btnRimuoviRitardo;

```

Metodi

```

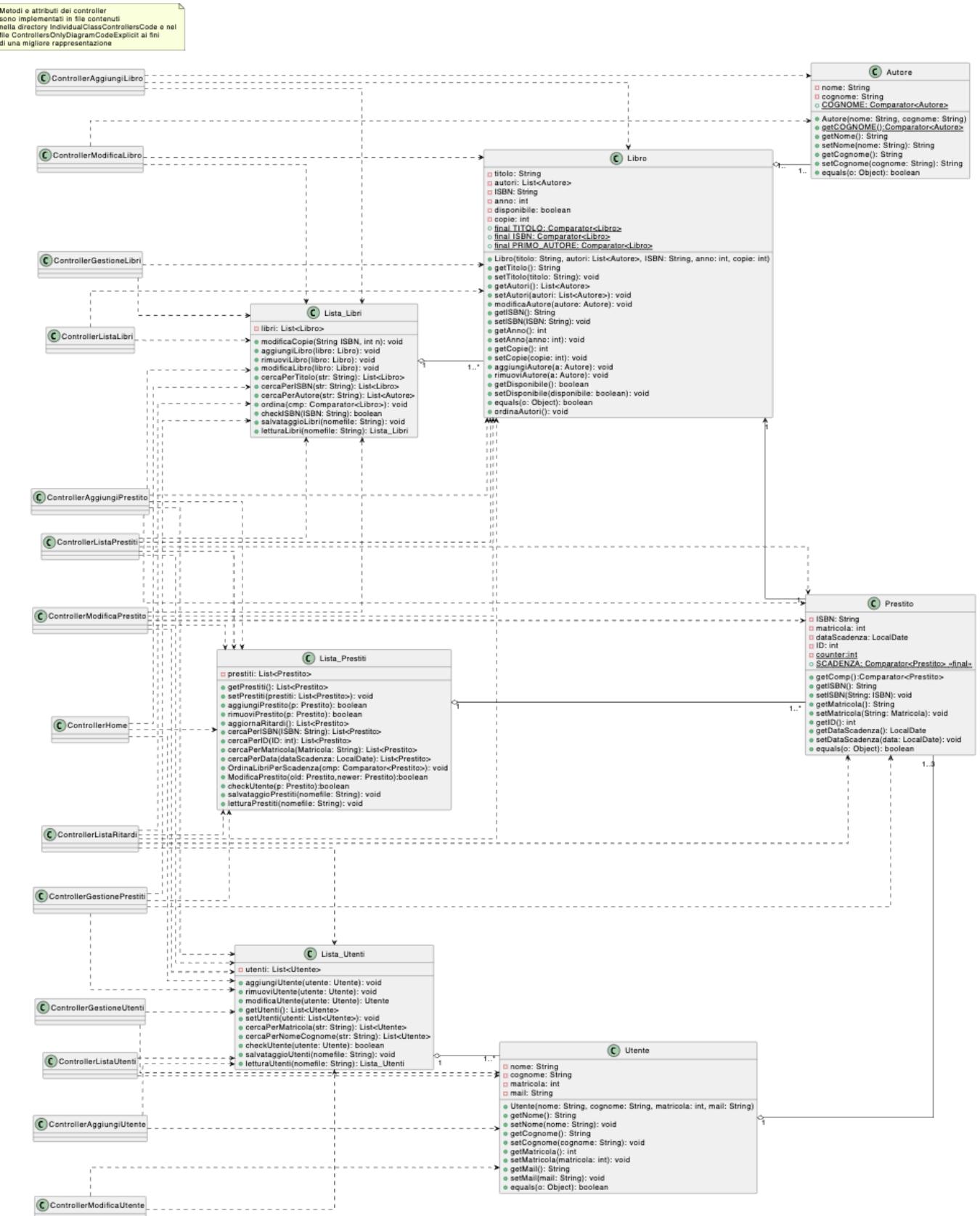
public void setListe(Lista_Prestiti lp, Lista_Utenti lu, Lista_Libri ll)
private void initialize()
private void aggiornaTabella()
private void handleChiudi()
private void handleCercaRitardo(ActionEvent event)
private void handleResetFiltroRitardi(ActionEvent event)
private void onModificaRitardo(ActionEvent event)
private boolean cambiaCopieLibro(String isbn, int delta)
private void onRimuoviRitardo(ActionEvent event)

```

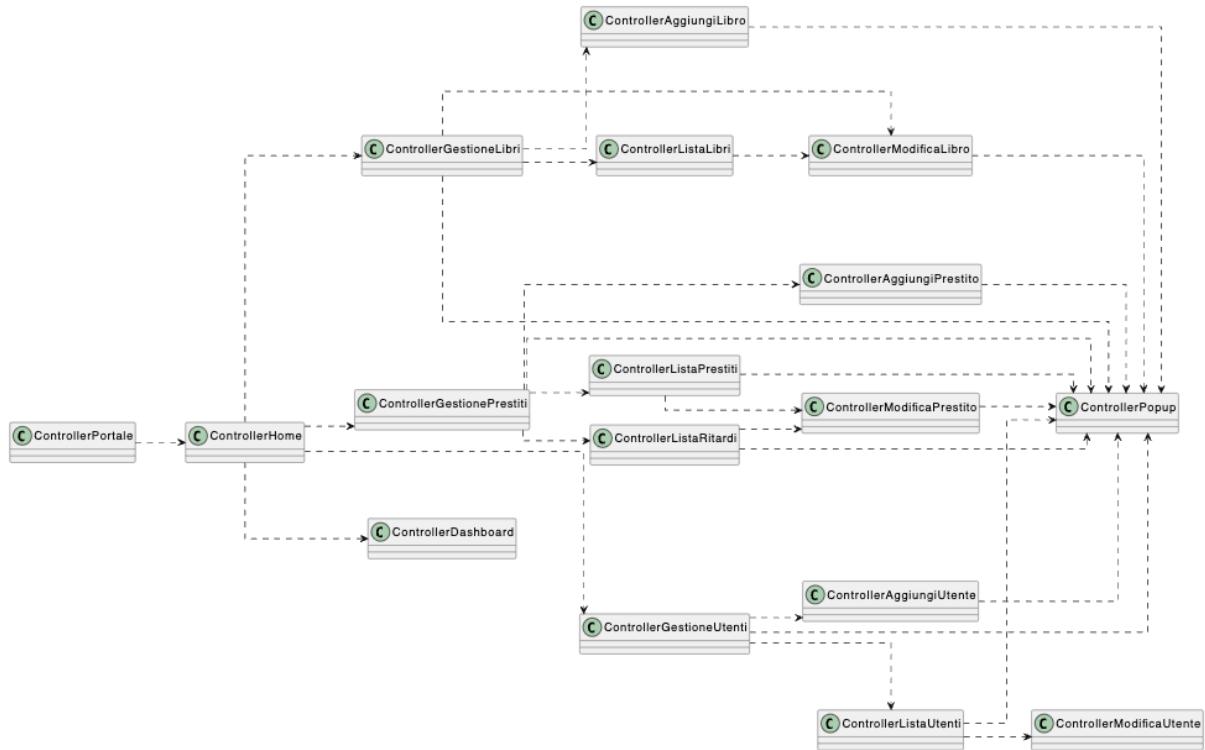
COESIONE E ACCOPPIAMENTO

- I moduli **Biblioteca**, **Utenti** e **Prestiti** presentano una coesione in linea di massima **Comunicazionale** dato che ogni modulo lavora sulla sua struttura anche se alcune funzionalità interne presentano una coesione sequenziale fra di loro.
- La User Interface ha una coesione **Sequenziale** per la maggior parte delle funzionalità mentre altre funzionalità prevedono una coesione temporale.
- I moduli Biblioteca Utenti non presentano nessun accoppiamento diretto ma entrambi hanno una accoppiamento **per dati** con Prestiti.
- La User Interface sfruttando le funzionalità intrinseche di java prevedono solo un passaggio di dati quindi presentano un **accoppiamento per dati** per l'appunto.

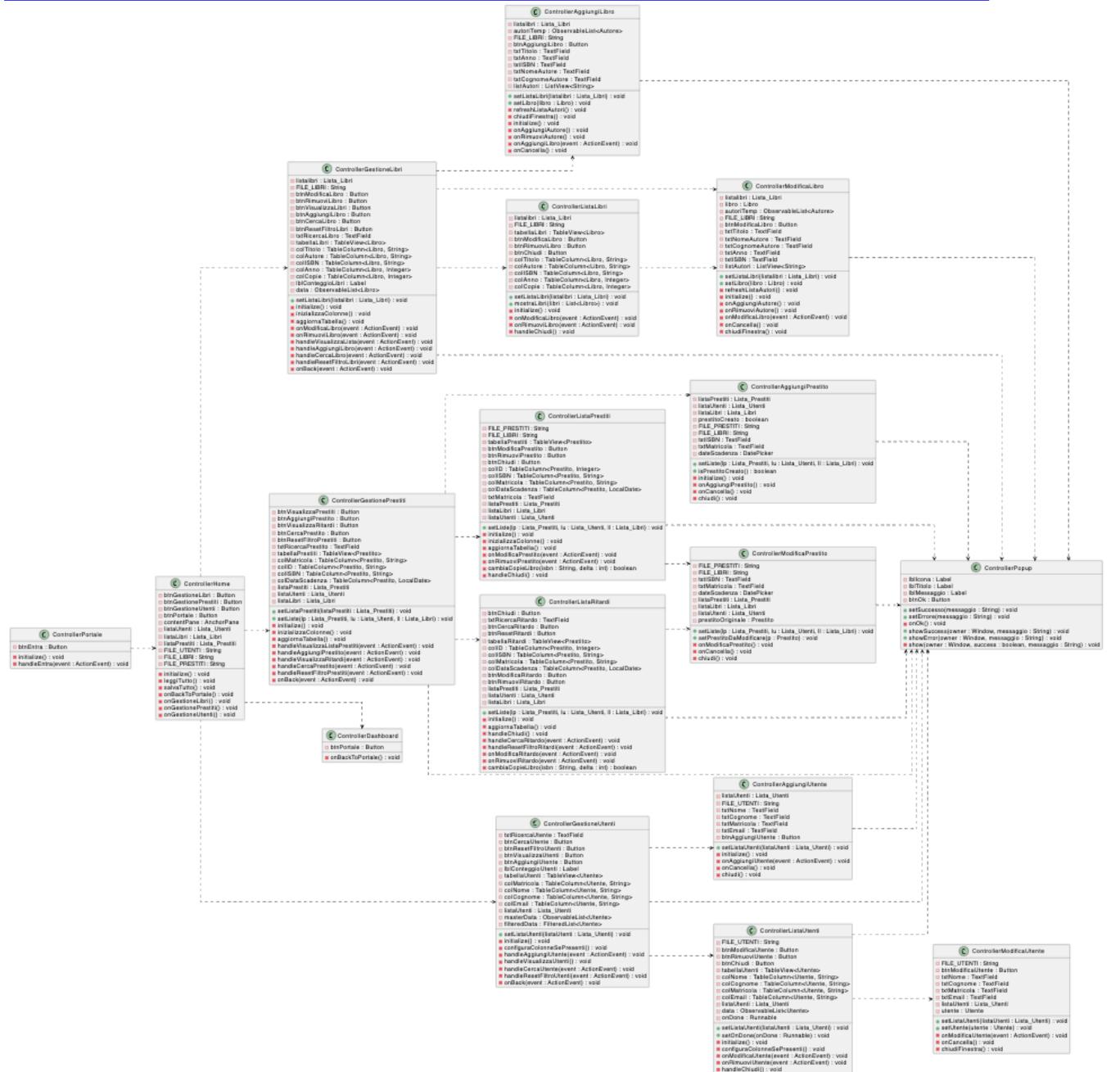
CLASS DIAGRAM



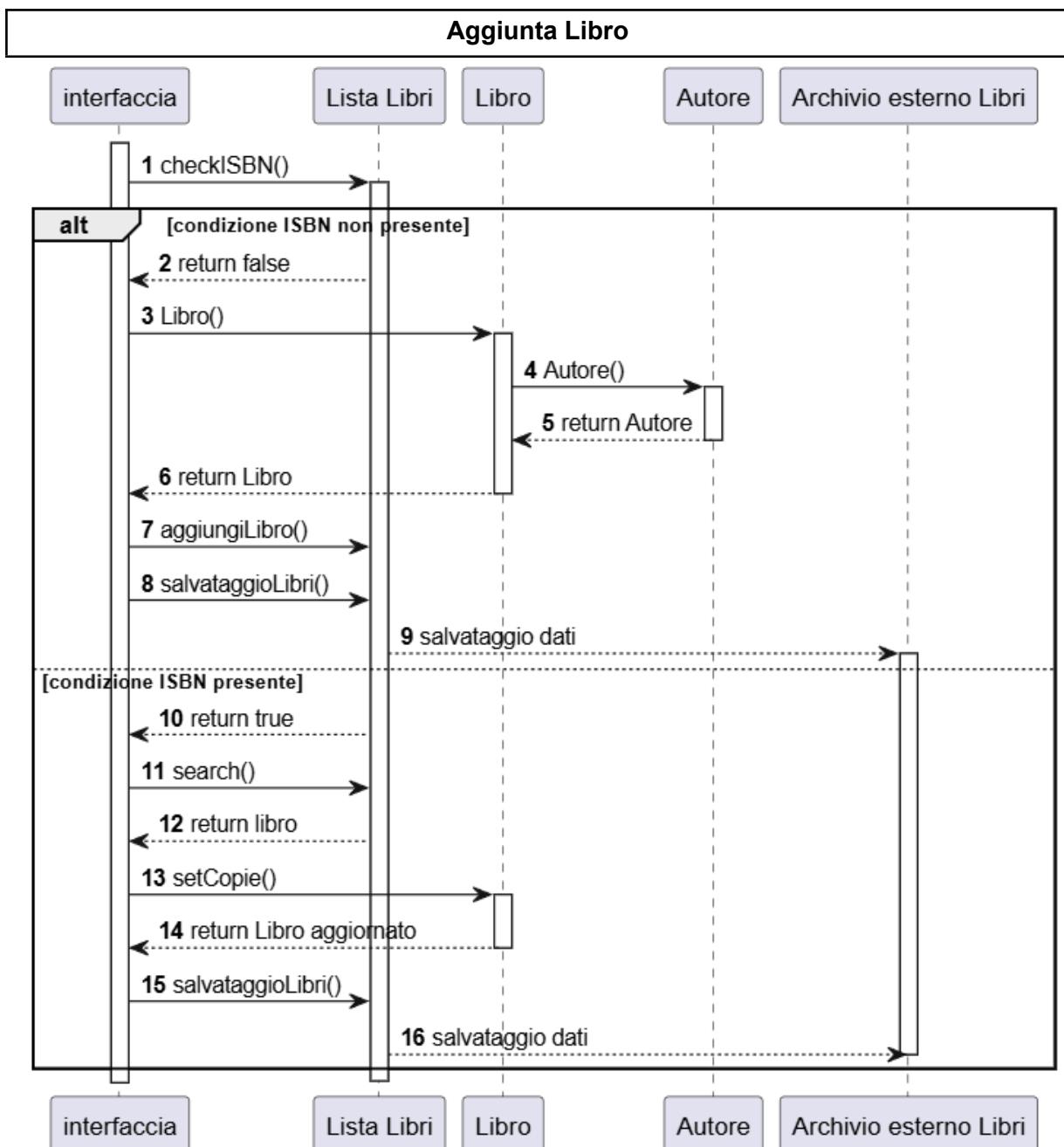
CONTROLLERS DIAGRAM



CONTROLLERS DIAGRAM (ESPLICITO)

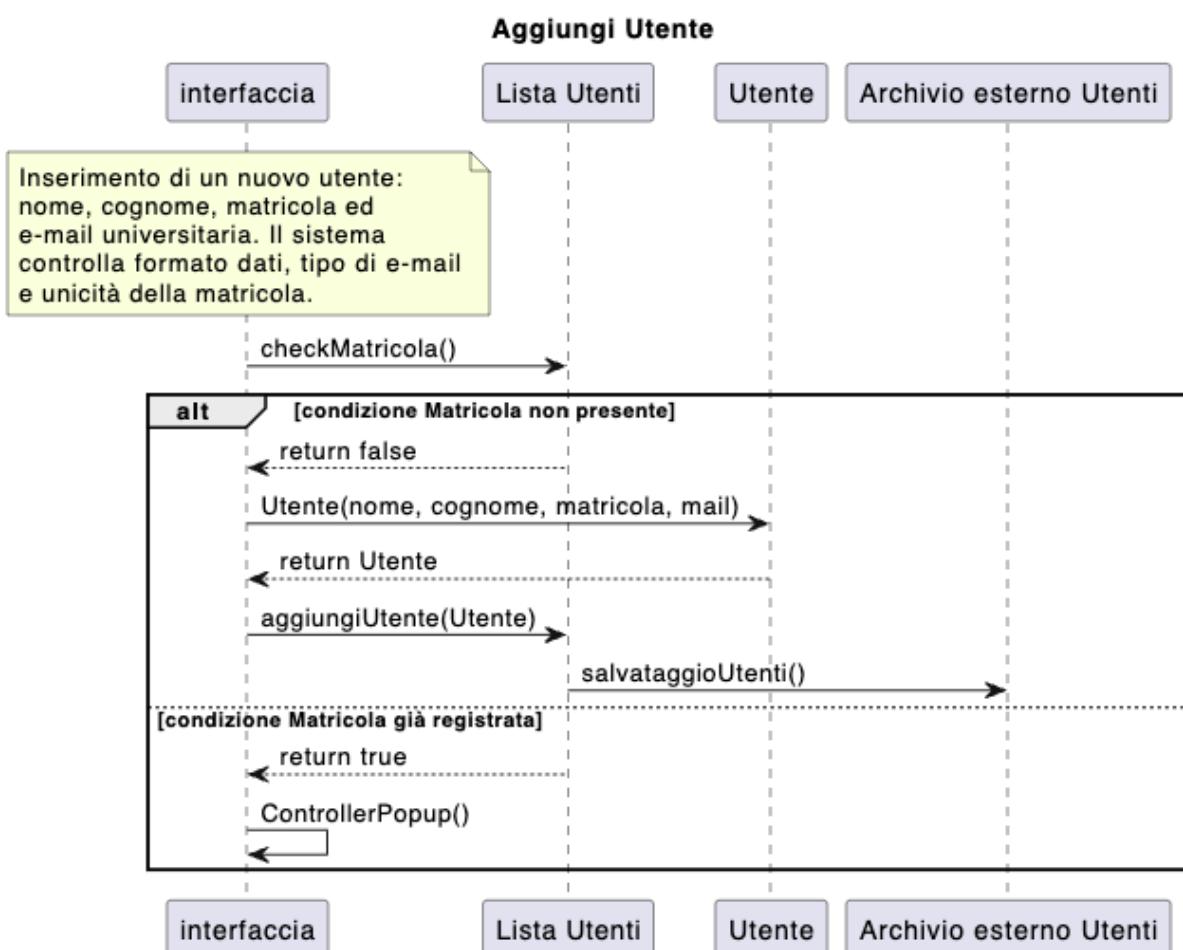


MODELLO DINAMICO



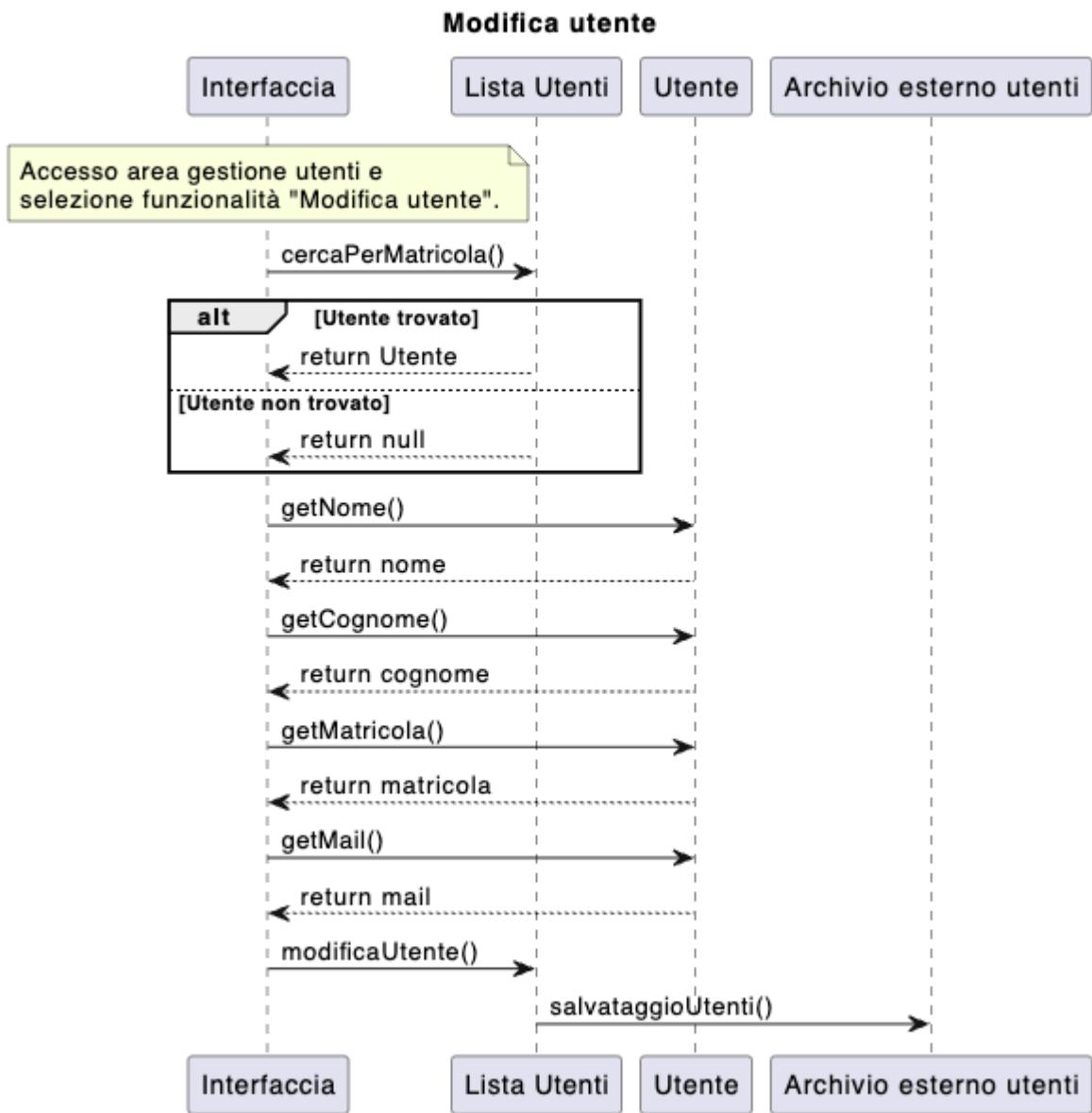
Il grafico rappresenta una delle funzionalità principali del modulo Biblioteca: l'inserimento di un libro in archivio locale con relativa copia nell'archivio esterno. Il diagramma esegue un controllo sull' ISBN, in caso di ISBN presente viene fatta una modifica sul numero di copie mentre se l'ISBN non è presente viene creato un nuovo oggetto Libro e aggiunto alla lista.

Aggiungi Utente



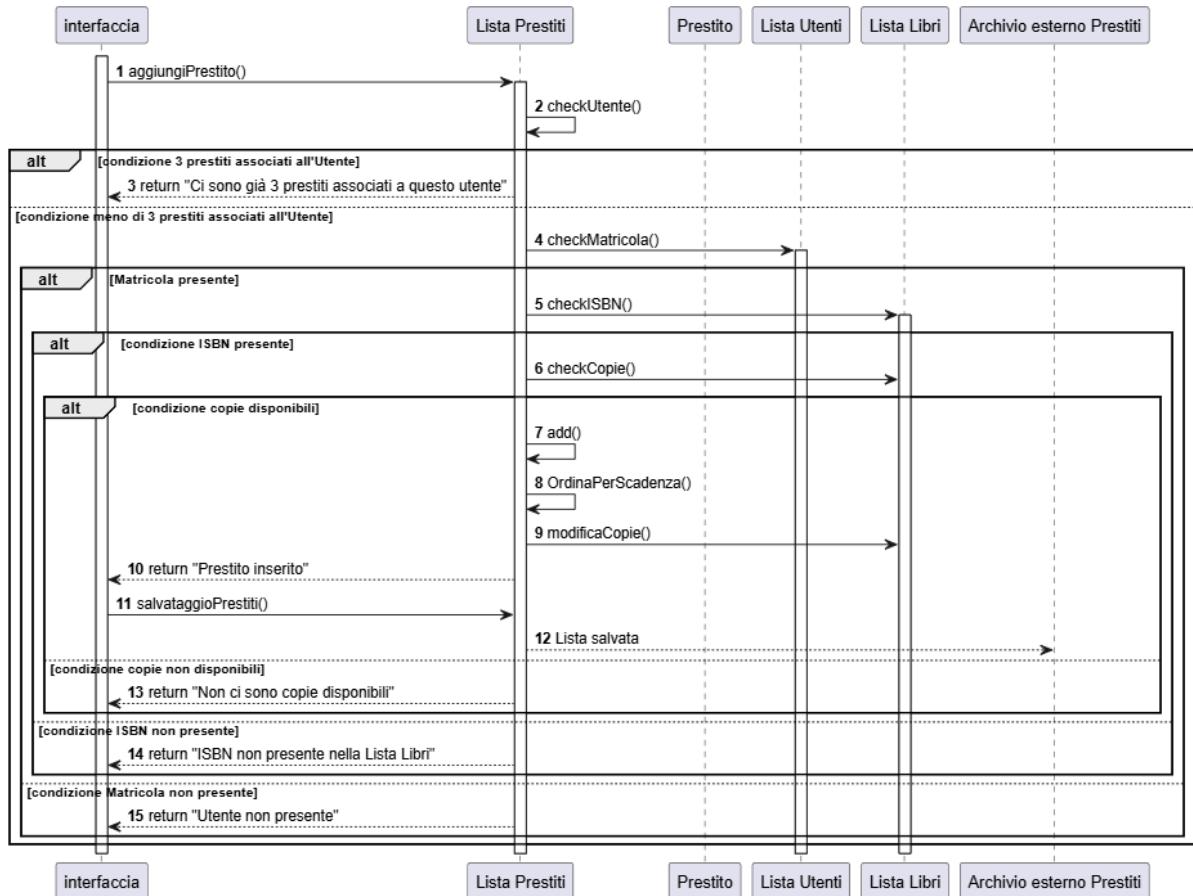
L'aggiunta di un utente avviene in maniera similare all'aggiunta di un libro.
Il check in questo caso avviene sulla matricola ed in caso di matricola presente viene notificato che l'utente è già presente in archivio.

Modifica utente



La modifica di un utente si divide in 2 parti principali la ricerca la ricerca dell'utente e i relativi metodi setter per modificare i dati inseriti

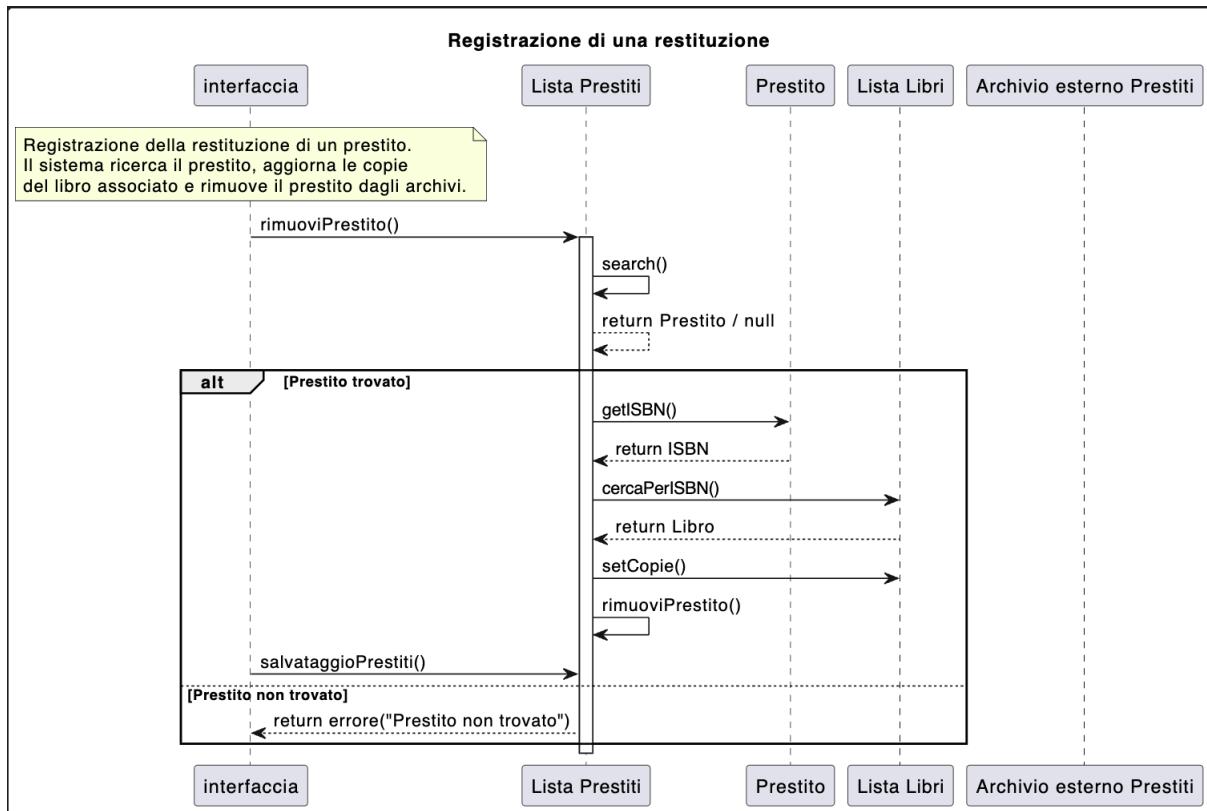
Aggiungi prestito



I prestiti uniscono un ISBN ad un utente ed una data quindi come prima cosa vengono eseguiti i controlli per avere la certezza che i dati inseriti siano presenti. Poi vengono eseguiti i controlli sull' Utente per assicurarsi che non abbia più di 3 prestiti attivi. Nel caso in cui tutti i controlli siano andati a buon fine viene aggiunto il prestito sia in locale che nel archivio esterno.

Elimina prestito

L'eliminazione di un prestito prevede la ricerca del prestito e poi la rimozione da entrambi gli archivi(locale e esterno).



DESIGN INTERFACCIA UTENTE

Accesso al sistema

Biblioteca Universitaria - Portale

The screenshot shows the homepage of the "Biblioteca Universitaria - Portale". At the top, there is a header bar with three colored circles (red, yellow, green) on the left and the text "Biblioteca Universitaria - Portale" in the center. Below the header is a large banner featuring a cartoon owl wearing a graduation cap, sitting on a stack of books and reading a book. The owl has a speech bubble above it that says "BIBLIOTECA UNIVERSITARIA 10". Above the owl, the text "ENTRA NELLA NOSTRA BIBLIOTECA" is displayed. In the bottom right corner of the banner, there is a blue button with the word "ENTRA" in white. At the very bottom of the page, there is a dark footer bar with the text "Progetto Biblioteca Universitaria "10" - a.a 2025/2026" in white.

ENTRA NELLA NOSTRA BIBLIOTECA

BIBLIOTECA
UNIVERSITARIA 10

ENTRA

Progetto Biblioteca Universitaria "10" - a.a 2025/2026

Home

Biblioteca Universitaria - Home

Ritorna al Portale

**BENVENUTO ALL'INTERNO DELLA
BIBLIOTECA UNIVERSITARIA 10**

Seleziona un'area di gestione dal menu a sinistra

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI

USERNAME: admin

LIBRI

UTENTI

PRESTITI

RITARDI

Sistema gestione biblioteca – versione 1.0

UNIVERSITÀ DEGLI STUDI DI SALERNO

Università degli Studi di Salerno • Progetto Biblioteca A.A 2025/2026



Interfaccia generale per la gestione libri

Biblioteca Universitaria - Home

Libri

Cerca libro

Cerca per titolo, autore o ISBN

Cerca Reset

Titolo Autore/ri Anno ISBN Copie

Nessun contenuto nella tabella

Modifica Libro Rimuovi Libro

Visualizza Lista Libri

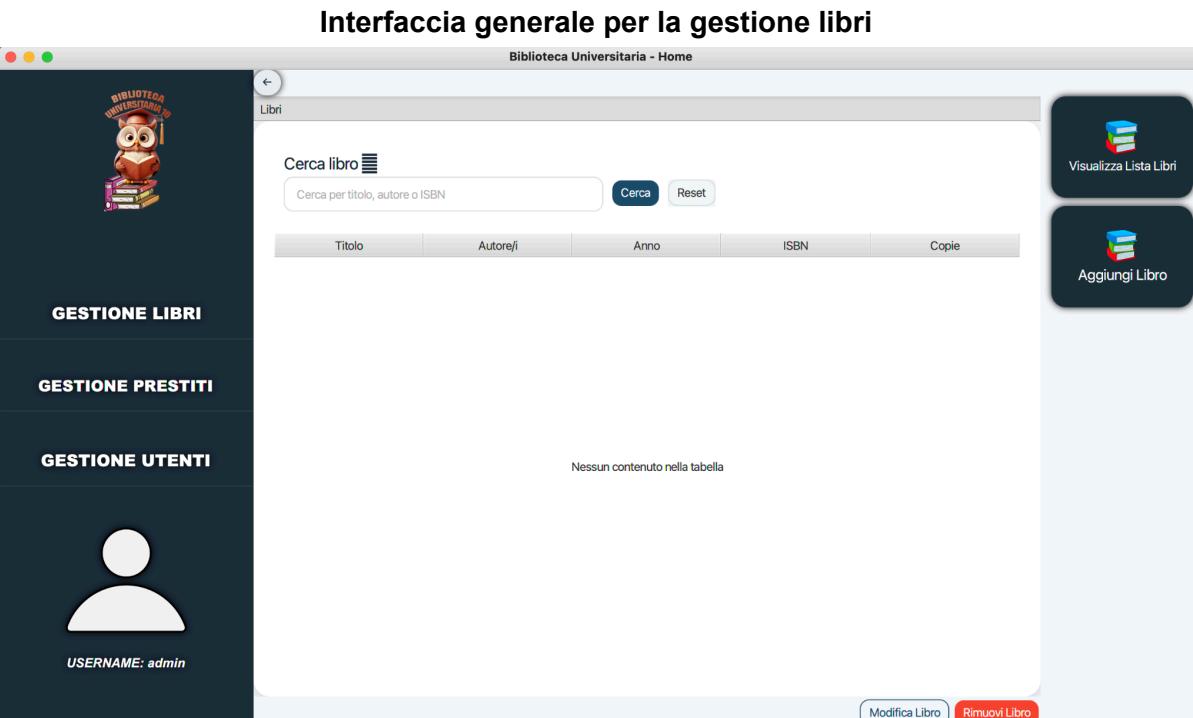
Aggiungi Libro

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI

USERNAME: admin



Visualizza libri

Biblioteca Universitaria - Home

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI

USERNAME: admin

Visualizza libri

Biblioteca Universitaria - Home

Cerca

Cerca

LISTA LIBRI

Titolo	Autore/i	Anno	ISBN	Copie
Elettrotecnica	Walter Zamboni	2015	134444444444	0

Chiudi Modifica Libro Rimuovi Libro

Copie

Visualizza Lista Libri

Aggiungi Libro

Modifica Libro Rimuovi Libro

Aggiungi libro

Biblioteca Universitaria - Home

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI

USERNAME: admin

Aggiungi libro

Biblioteca Universitaria - Home

Cerca

Cerca

Aggiungi Libro

Titolo

Autori Nome Cognome

Anno di Pubblicazione YYYY

Codice ISBN 13 cifre

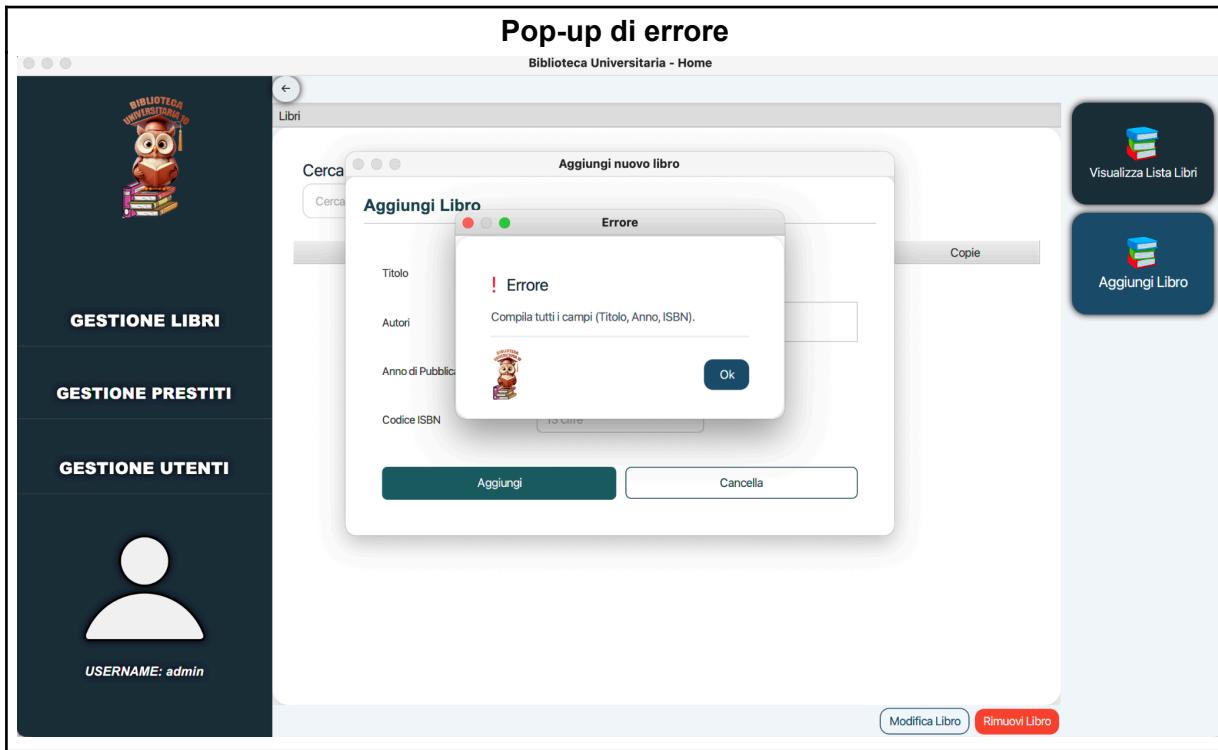
Aggiungi Cancella

Copie

Visualizza Lista Libri

Aggiungi Libro

Modifica Libro Rimuovi Libro



Interfaccia generale per la gestione prestiti

Biblioteca Universitaria - Home

Prestiti

Cerca Prestito

Cerca per cognome, matricola o ISBN del libro...

Matricola	ID	ISBN	Data scadenza
Nessun contenuto nella tabella			


BIBLIOTECA UNIVERSITARIA UD

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI


USERNAME: admin


Visualizza Lista Prestiti


Aggiungi Prestito


Visualizza Ritardi

Visualizza prestiti

Biblioteca Universitaria - Home

Prestiti

Cerca

LISTA PRESTITI

Matricola	ISBN	ID	Data Scadenza
061279574	134444444444	0	2025-12-16


BIBLIOTECA UNIVERSITARIA UD

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI


USERNAME: admin


Visualizza Lista Prestiti


Aggiungi Prestito


Visualizza Ritardi

Aggiungi prestito

Biblioteca Universitaria - Home



Prestiti

Aggiungi Prestito

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI

USERNAME: admin

ISBN:

Matricola: 06127 Inserisci ultime 4 cifre

E-mail: nome.cognome @studenti.unisa.it

Data Scadenza:

Aggiungi Cancella

Visualizza Lista Prestiti

Aggiungi Prestito

Visualizza Ritardi

Visualizza ritardi

Biblioteca Universitaria - Home



Prestiti

Lista completa dei ritardi

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI

USERNAME: admin

Lista Ritardi

Cerca Reset

Matricola	ID	ISBN	Data Scadenza
Nessun ritardo presente.			

Chiudi Modifica Ritardo Rimuovi Ritardo

Visualizza Lista Prestiti

Aggiungi Prestito

Visualizza Ritardi

Interfaccia generale per la gestione Utenti

Biblioteca Universitaria - Home

Utenti

Cerca Utente Cerca per Matricola

Cerca Reset

Nome	Cognome	Matricola	E-mail Universitaria
Nessun contenuto nella tabella			

Visualizza Lista Utenti

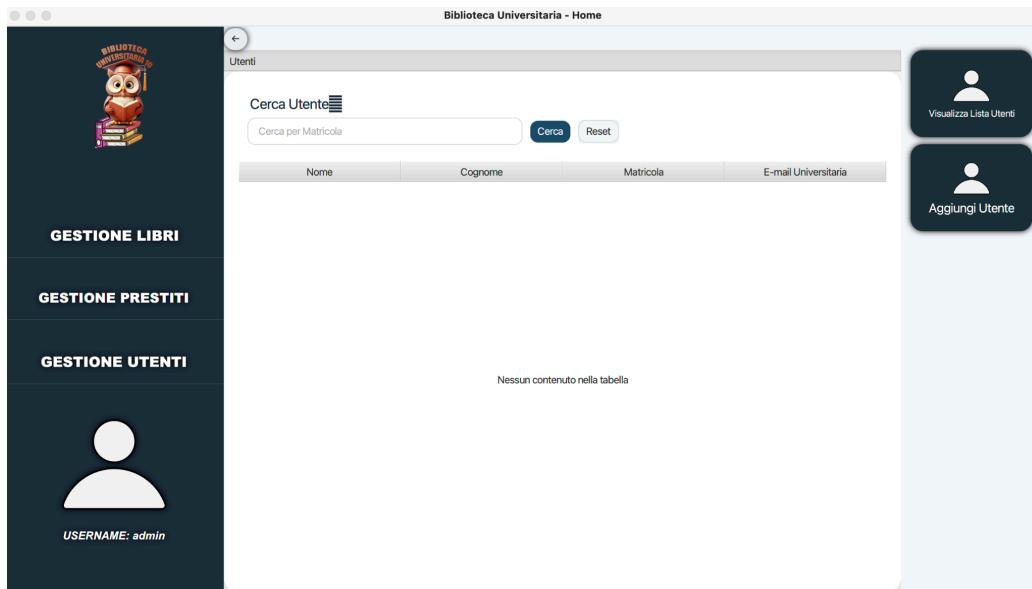
Aggiungi Utente

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI

USERNAME: admin



Visualizza lista Utenti

Biblioteca Universitaria - Home

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI

USERNAME: admin

Biblioteca Universitaria - Home

GESTIONE LIBRI

GESTIONE PRESTITI

GESTIONE UTENTI

USERNAME: admin