DESIGN

1 DIAGRAMMI DELLE CLASSI

Le classi del progetto possono essere suddivise in diversi diagrammi in base al loro ruolo:

- CONTROLLER
- EXCEPTION
- MAPPER
- MODEL
- SERVICE

1.1 PRINCIPI DI BUONA PROGETTAZIONE

1.1.1 FRONTEND

Il front-end comprende la gestione delle interfacce grafiche e delle interazioni dell'utente attraverso l'uso di JavaFX e di FXML; la sua struttura interna segue il pattern architetturale dell'MVC. L'uso di questo pattern garantisce una separazione delle responsabilità (SoC) delle classi che lo riguardano grazie alla suddivisione in model, view e controller.
E' stata inoltre realizzata una classe astratta estesa dai vari controller del progetto; questa classe permette la condivisione di funzionalità più articolate (es. apertura di popUp) tra le diverse scene del progetto usando lo stesso codice (DRY).

1.1.2 BACKEND

Il backend comprende la gestione di utenti, prestiti, libri e dell'autenticazione del bibliotecario. Esso è formato dall'implementazione delle interfacce descritte precedentemente attraverso l'uso di un database relazionale.
La scelta dell'uso di un database relazionale garantisce modularità alla persistenza dei dati: si può optare per una soluzione su file (con SQLite) o si può spostare la persistenza su un DBMS (PostgreSQL, MySQL…); inoltre il formato dei dati salvati sul database NON dipende dalla serializzazione di oggetti Java e quindi la persistenza è anche modulare rispetto al linguaggio di programmazione utilizzato.

1.1.3 COMUNICAZIONE FRONTEND/BACKEND

La comunicazione tra front-end e back-end avviene con l'uso di diverse, generiche interfacce chiamate "Service" che hanno lo scopo di isolare le funzionalità richieste dal front-end (gestione utenti, prestiti…) con la loro effettiva implementazione.
Questa architettura ha come principi cardine:

- Principio della Singola Responsabilita': ogni interfaccia espone metodi relativi ad una singolo aspetto del progetto
- Principio Aperto-Chiuso: se un cliente volesse modificare o estendere le funzionalità fornite da un'interfaccia può estendere una delle implementazioni fornite dal back-end senza dover modificare in alcun modo i controller presenti.
- Principio di Sostituzione di Liskov: in corrispondenza di ogni interfaccia si può sostituire una qualsiasi sua implementazione e il programma continuerà a funzionare correttamente

- Principio della Segregazione delle Interfacce: sono state realizzate diverse interfacce che permettono ad un client (es. il front-end) di dipendere solo dalla funzionalità realmente richiesta
- Principio dell'Inversione delle Dipendenze: la comunicazione tra i moduli di basso livello (es. backend) e quelli di alto livello (es. frontend) dipende esclusivamente dall'uso di queste interfacce

Inoltre il progetto comprende numerose eccezioni descrittive (es DuplicateBookByIsbnException) che permettono la gestione di errori e la comunicazione chiara con il bibliotecario.

1.2 COESIONE

1.2.1 FRONTEND

Le classi Controller presentano un livello di coesione funzionale. Ogni controller è legato a una singola scena FXML e gestisce solo gli eventi e la logica di presentazione che riguardano quella specifica schermata (es. `AddBookSceneController` gestisce solo l'aggiunta dei libri).

1.2.2 BACKEND

I servizi implementati dal backend presentano un livello di coesione funzionale. Le classi (es. `DatabaseUserService`) raggruppano metodi che operano sulla stessa entità di dominio e contribuiscono ad un unico compito logico (es. la gestione della persistenza per quell'entità), senza includere logiche estranee.

1.3 ACCOPPIAMENTO

1.3.1 FRONTEND

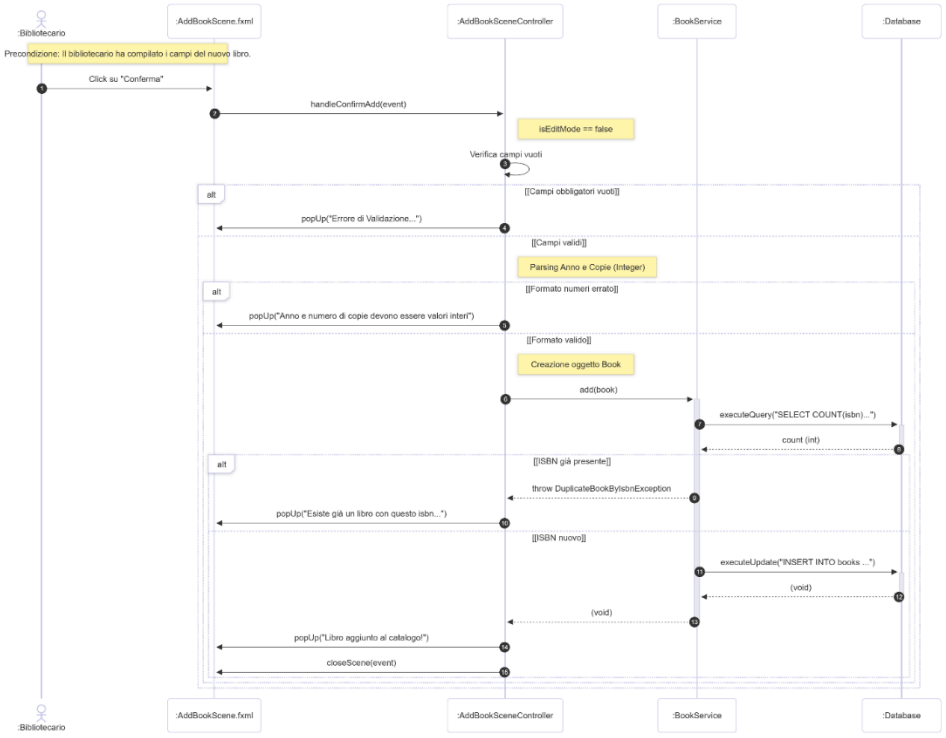
Il frontend mantiene un basso accoppiamento con il backend grazie all'uso delle interfacce dei Service. I controller dipendono dall'astrazione (`BookService`) e non dalla concretezza (`DatabaseBookService`).

1.3.2 BACKEND

I servizi implementati presentano un accoppiamento di tipo stamp, in quanto le implementazioni richiedono l'uso di un oggetto di tipo `Database` (wrapper di JDBI) per funzionare.

2 DIAGRAMMI DI SEQUENZA

2.1 Inserimento di un libro



Questo diagramma di sequenza illustra il flusso di aggiunta di un nuovo libro al catalogo da parte di un Bibliotecario.

Il processo segue questi passaggi principali:

A. Il bibliotecario clicca su "Conferma" nell'interfaccia (AddBookScene).

B. Il Controller verifica che i campi non siano vuoti e che i numeri (anno, copie) siano nel formato corretto. Se ci sono errori, mostra un pop-up.

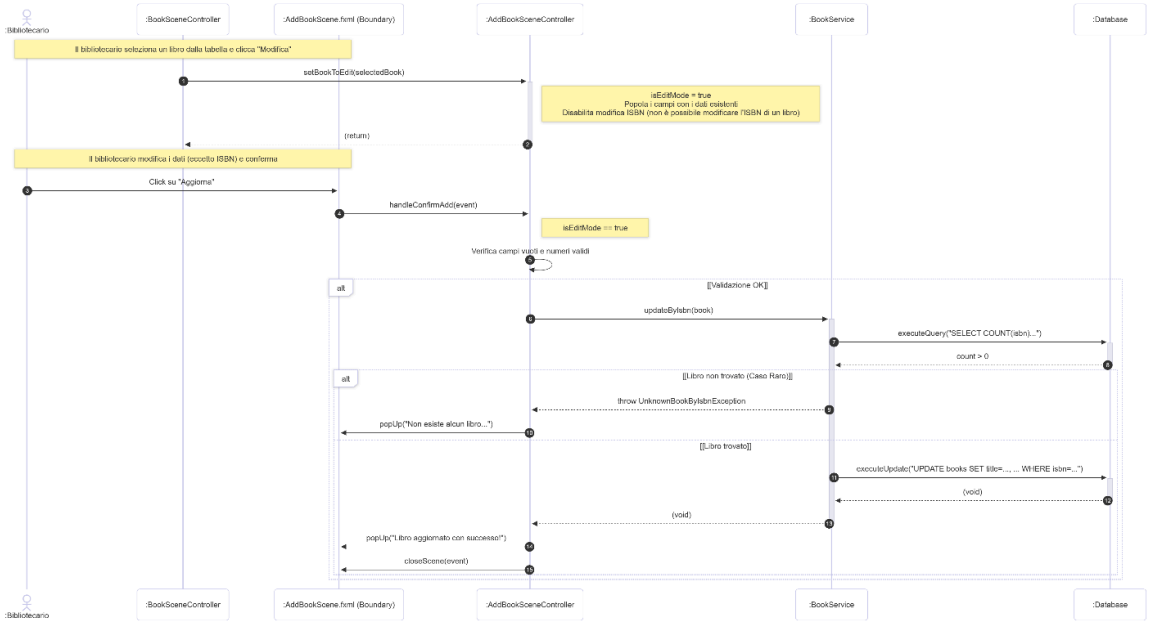
C. Se i dati sono validi, viene creato l'oggetto Book e passato al BookService.

D. Il Service controlla nel Database se esiste già un libro con lo stesso ISBN.

a. Se esiste (duplicato), viene sollevata un'eccezione e mostrato un errore.

b. Se non esiste, il libro viene inserito (INSERT), viene mostrato un messaggio di successo e la finestra si chiude.

2.2 Modifica di un libro



Questo diagramma di sequenza illustra il flusso di modifica (aggiornamento) di un libro esistente nel catalogo da parte di un Bibliotecario.

I punti chiave del processo sono:

A. Modalità Modifica: Quando il bibliotecario clicca su "Modifica", il sistema pre-compila i campi con i dati esistenti e disabilita l'ISBN (che non può essere cambiato). Viene impostato un flag isEditMode = true.

B. Invio e Validazione: L'utente modifica gli altri dati (titolo, autore, ecc.) e conferma. Il sistema valida nuovamente i dati.

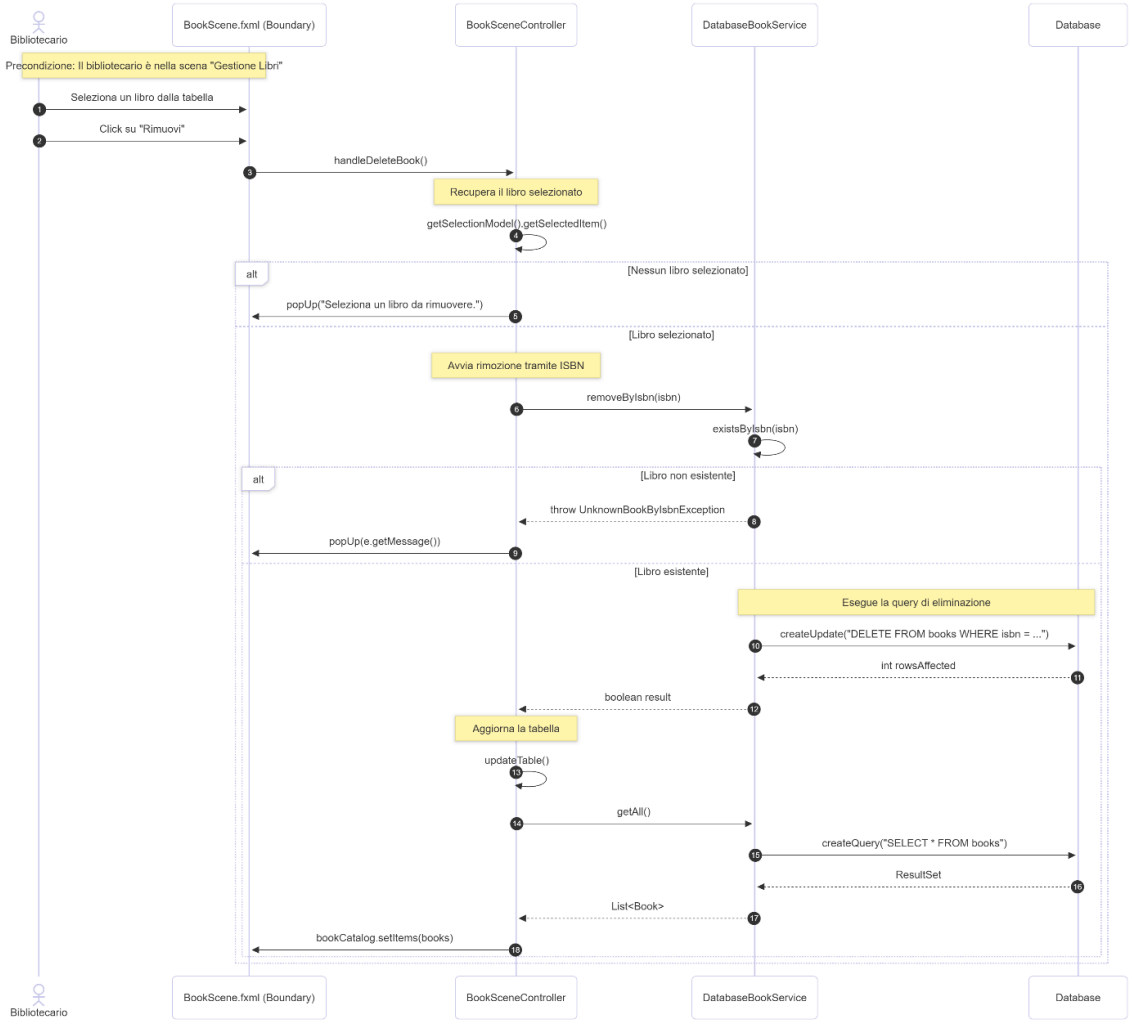
C. Aggiornamento (Database):

a. Il BookService verifica che il libro esista ancora nel database (cercando per ISBN).

b. Se il libro viene trovato, viene eseguita una query UPDATE per sovrascrivere i dati.

c. Viene mostrato un messaggio di conferma ("Libro aggiornato con successo") e la finestra si chiude.

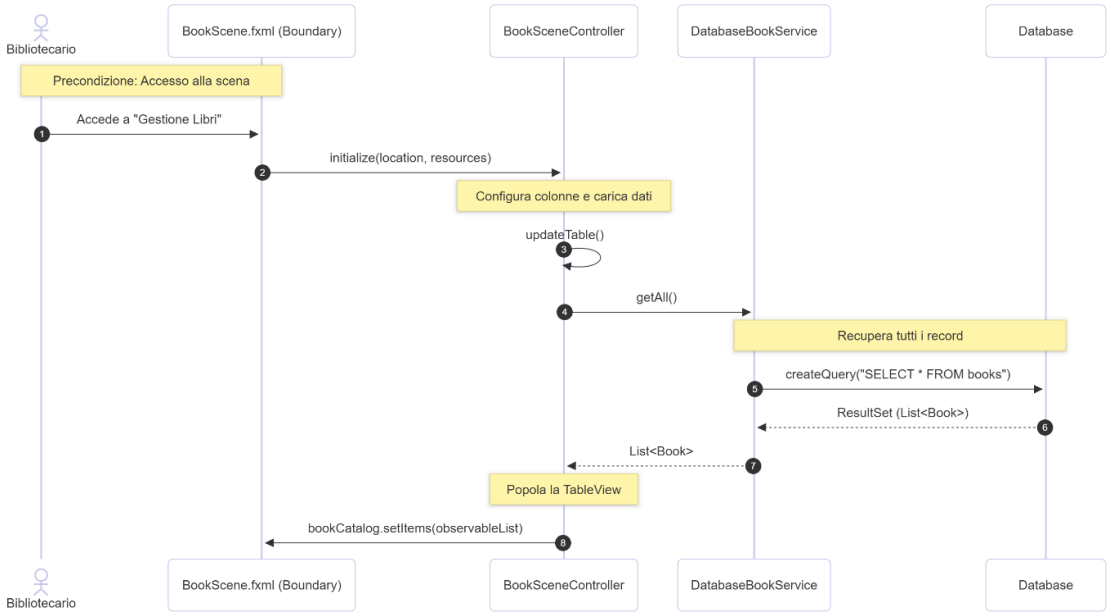
2.3 Rimozione di un libro



Questo caso d'uso descrive come il bibliotecario può rimuovere un libro dal catalogo dopo essere entrati nella sezione "Gestione libri".

- **A.** Il *BookSceneController* verifica la selezione di un libro dalla tabella. Se la selezione è stata effettuata, recupera l'ISBN del libro e lo invia al *DatabaseBookService* per la richiesta di rimozione; altrimenti, mostra un pop-up di errore.
- **B.** Il servizio controlla la presenza del libro nel database e, in caso positivo, esegue la query che ne elimina il record.
- **C.** Alla fine, il controller aggiorna la vista per mostrare il catalogo privo del libro eliminato.

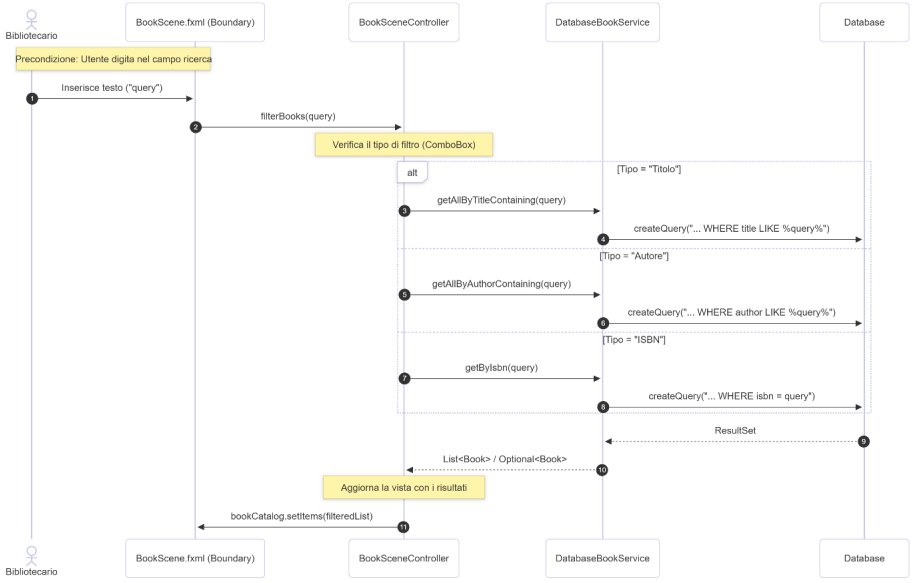
2.4 Visualizzazione catalogo libri



Questo caso d'uso descrive come viene visualizzato il catalogo dei libri dopo essere entrati nella sezione "Gestione libri".

- **A.** Il *BookSceneController* inizializza la tabella e chiama il metodo di aggiornamento per visualizzare i dati più recenti.
- **B.** Il *DatabaseBookService* esegue una query al database per recuperare la lista completa dei libri presenti nel catalogo.
- **C.** La lista di libri ottenuta viene convertita in una lista osservabile e inserita nella *TableView*, consentendo al bibliotecario di visualizzare a schermo il catalogo dei libri.

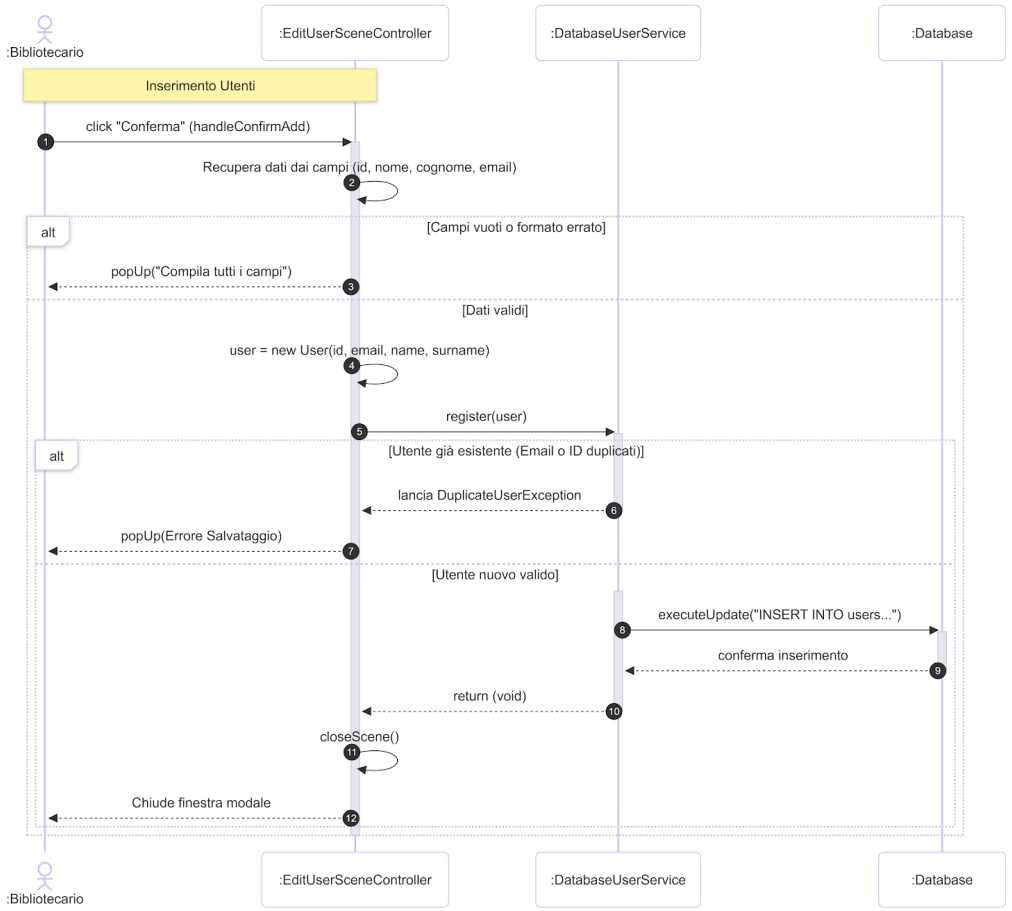
2.5 Ricerca libri



Questo caso d'uso descrive come il sistema filtra dinamicamente il catalogo in base ai criteri di ricerca inseriti dal bibliotecario.

- **A.** Il bibliotecario sceglie un filtro di ricerca dal menù a tendina e digita nel campo di ricerca una query, il *BookSceneController* cattura l'input e verifica quale tipo di filtro sia stato selezionato (Titolo, Autore, ISBN ecc.).
- **B.** Il *DatabaseBookService*, interroga il database con la query inserita per recuperare la lista di libri corrispondente.
- **C.** La lista di libri ottenuta viene inserita nella *TableView*, consentendo al bibliotecario di visualizzare a schermo solo i libri corrispondenti al criterio di ricerca specificato.

2.6 Inserimento di un utente



Questo diagramma di sequenza illustra il flusso di registrazione di un nuovo Utente nel sistema da parte del Bibliotecario.

A. Avvio e Validazione: Il bibliotecario compila i dati (Matricola, Nome, Email, ecc.) e conferma. Il Controller verifica che tutti i campi obbligatori siano pieni.

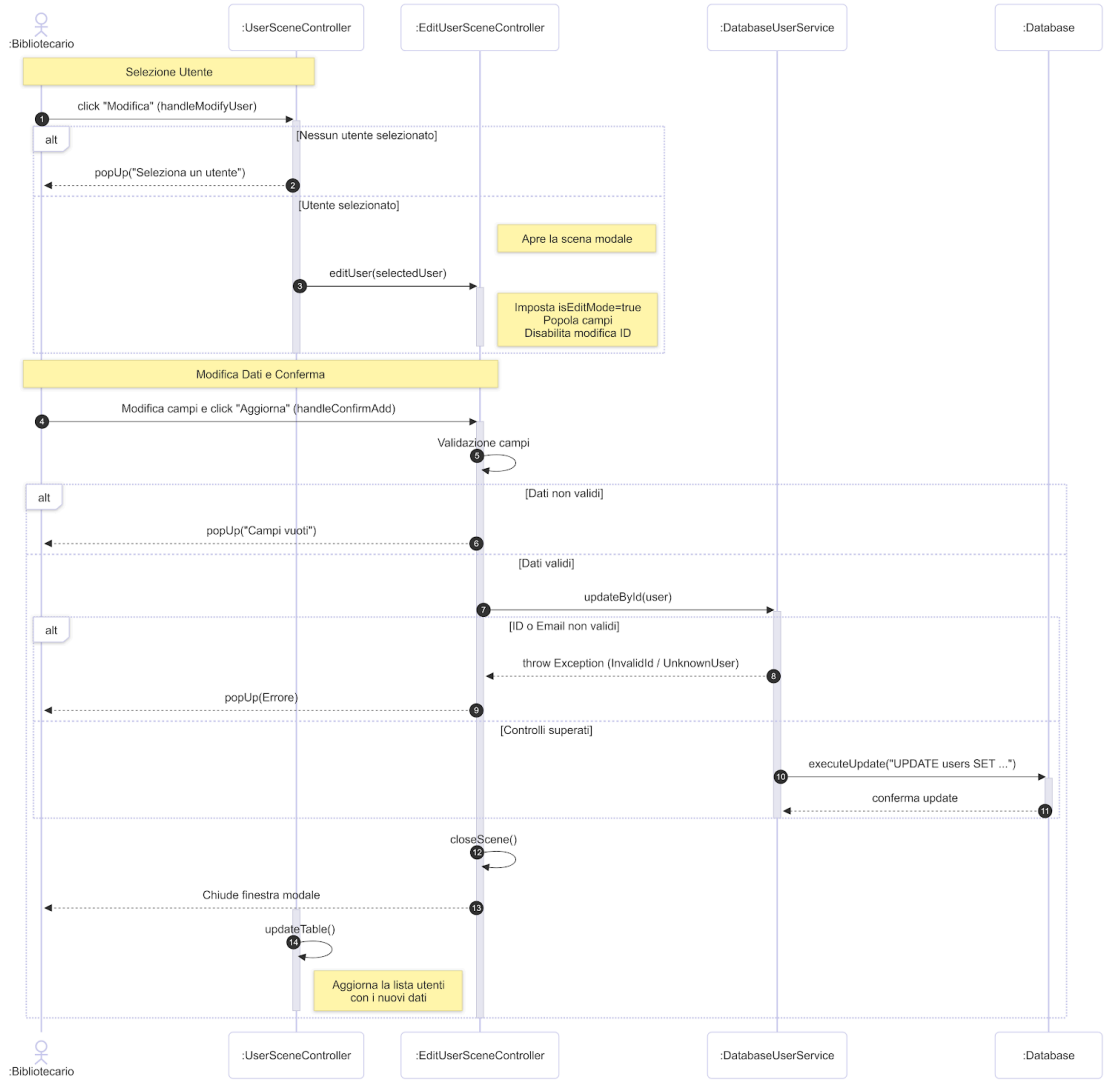
B. Lo UserService esegue due controlli distinti nel Database:

- Verifica se l'Email è già presente.
- Verifica se la Matricola è già presente.

C. Risultato:

- Se viene trovato un duplicato (Email o Matricola), viene sollevata un'eccezione specifica e mostrato l'errore all'utente.
- Se i dati sono nuovi, l'utente viene inserito nel database (INSERT) e la finestra si chiude.

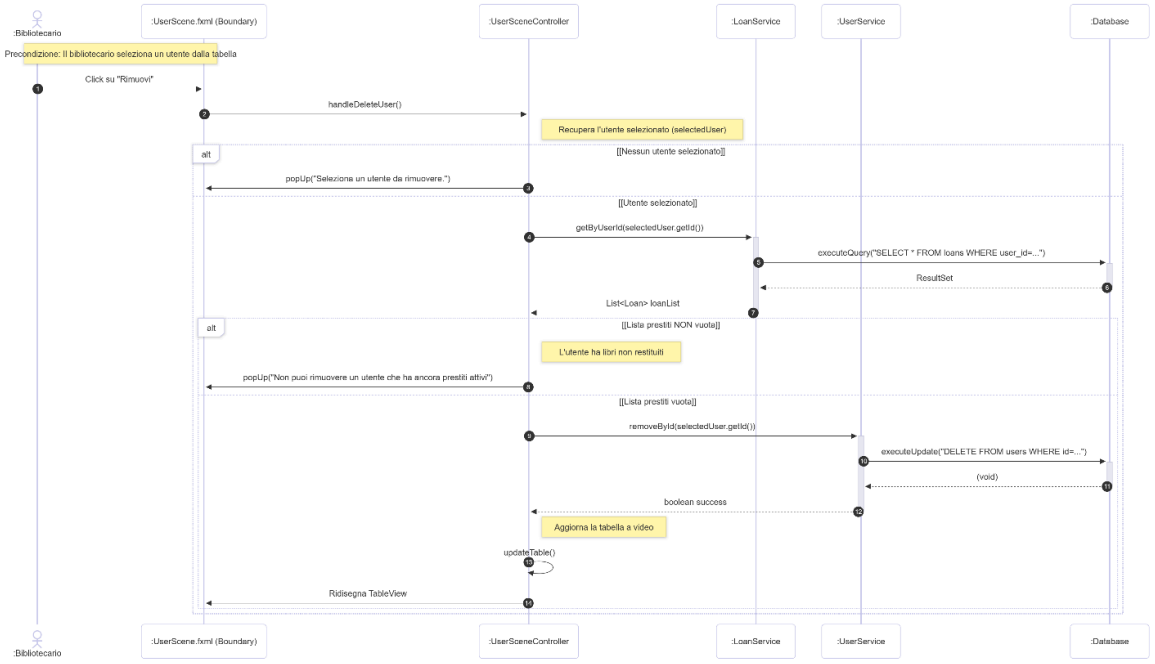
2.7 Modifica di un utente



Questo diagramma di sequenza illustra il flusso di modifica di un utente nel sistema da parte del Bibliotecario.

- **A.** Il Bibliotecario preme sulla riga dell'utente da modificare.
- **B.** Dopo la modifica dei dati, il Controller invia i dati aggiornati al Servizio UserService
- **C.** Il Servizio esegue la query UPDATE sul Database.

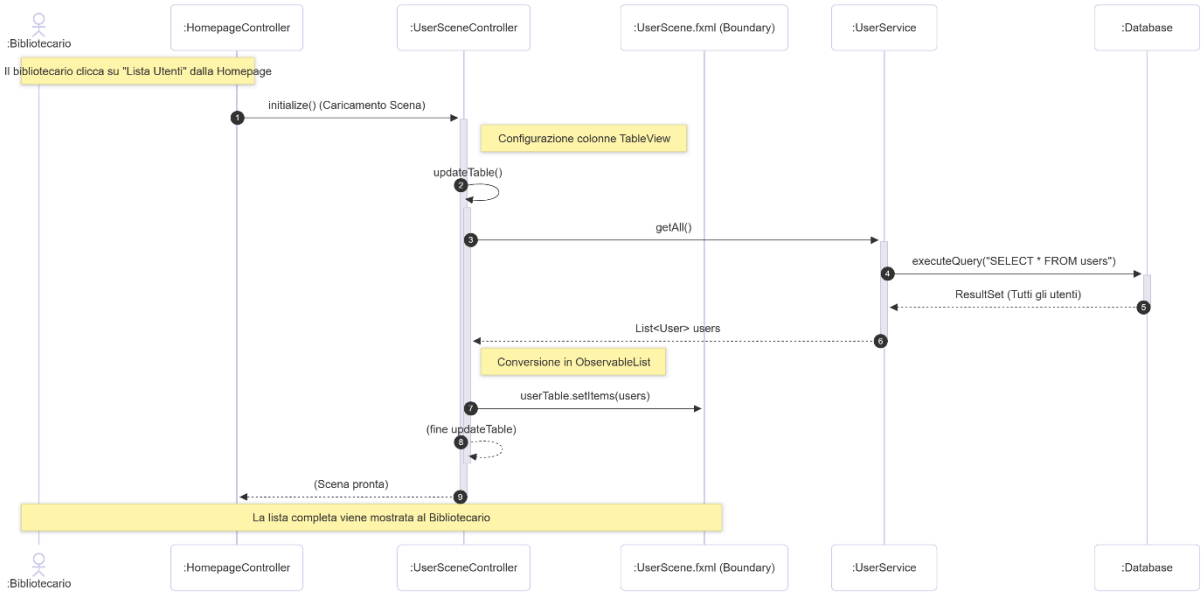
2.8 Rimozione di un utente



Questo diagramma di sequenza illustra il flusso di rimozione di un utente nel sistema da parte del Bibliotecario.

- **A.** Il Controller Utente (UC) verifica prima, tramite il Servizio Prestiti (LS), l'assenza di prestiti attivi per l'utente dal Database (DB).
- **B.** Se vengono trovati prestiti attivi, il processo viene bloccato con un'eccezione; in caso contrario, il Controller delega la rimozione al Servizio Utente (US).
- **C.** Il Servizio Utente esegue la query DELETE sul Database; una volta completata la rimozione, il successo viene notificato e l'account è considerato rimosso.

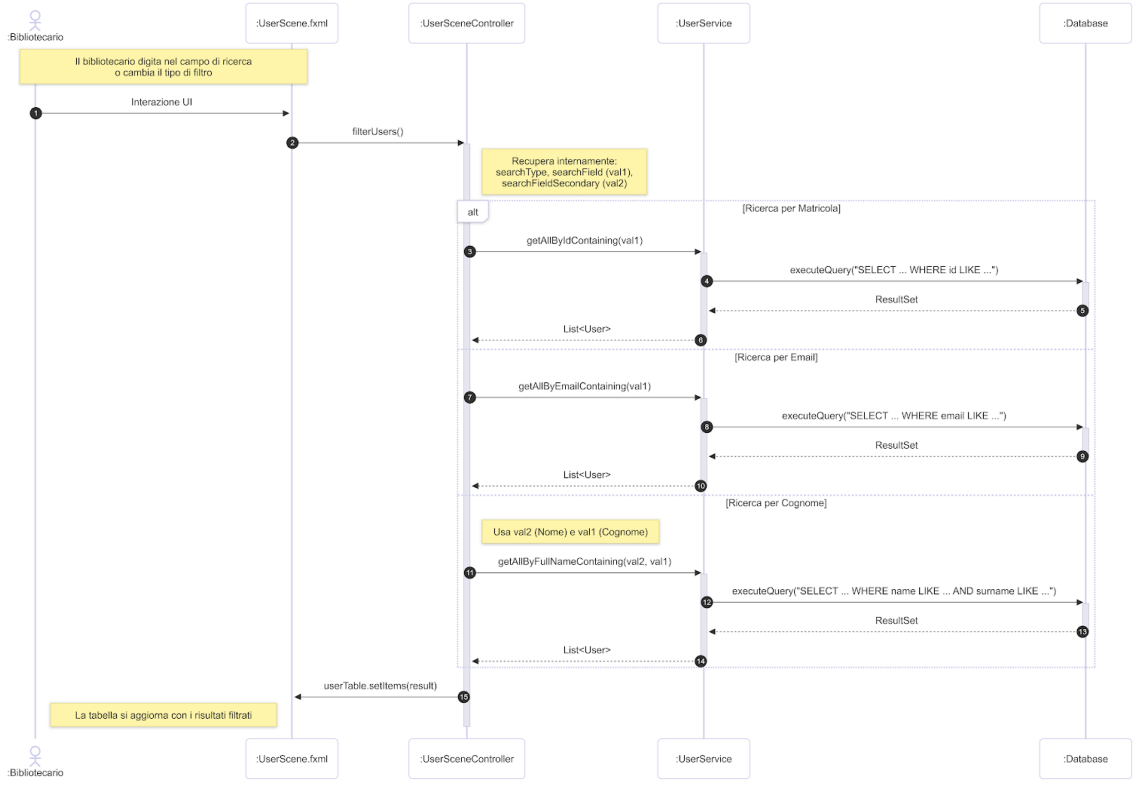
2.9 Visualizzazione lista utenti



Questo diagramma di sequenza illustra il flusso per visualizzare una lista di utenti dal sistema da parte del Bibliotecario.

A. Il Bibliotecario accede alla scena e sceglie il criterio di ordinamento desiderato (es. per nome o cognome).
B. Il Controller Utente (UC) inoltra la richiesta di recupero e ordinamento al DatabaseUserService (US), che esegue una query SELECT con la clausola ORDER BY sul Database (DB).
C. Il Database. restituisce la lista completa degli utenti già ordinata, che viene poi visualizzata dalla scena UI all'attore.

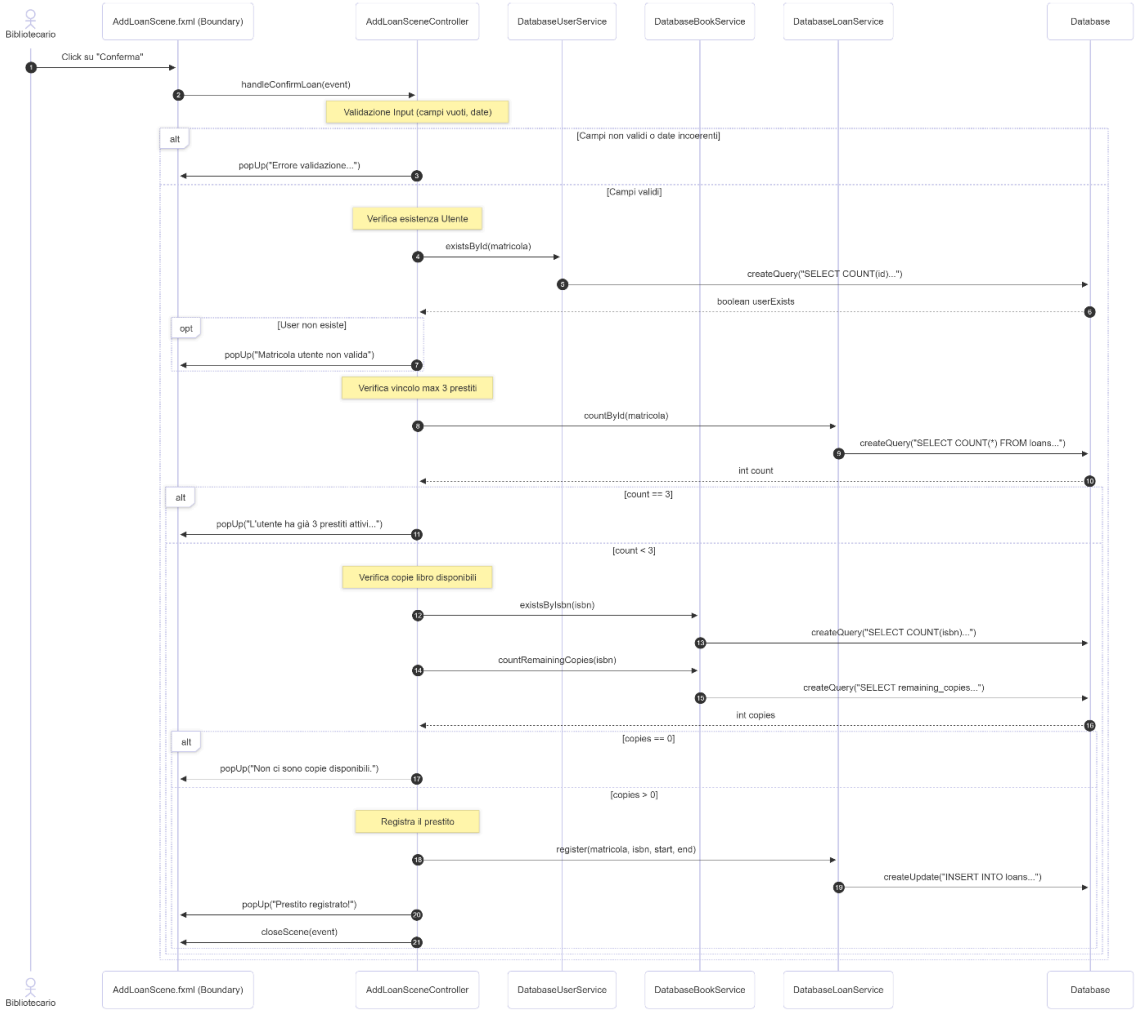
2.10 Ricerca utente



Questo diagramma di sequenza illustra il flusso di ricerca di un utente secondo un criterio nel sistema da parte del Bibliotecario.

- **A.** Il Bibliotecario specifica il criterio di ricerca (es. matricola o email) all'interfaccia.
- **B.** Il Controller delega la richiesta al DatabaseUserService, che esegue una query SELECT sul Database.
- **C.** Il Database restituisce una lista di utenti corrispondenti, che viene visualizzata a schermo per il Bibliotecario.

2.11 Registrazione prestito



Questo caso d'uso descrive come il bibliotecario può registrare un nuovo prestito accedendo alla sezione "Gestione prestiti".

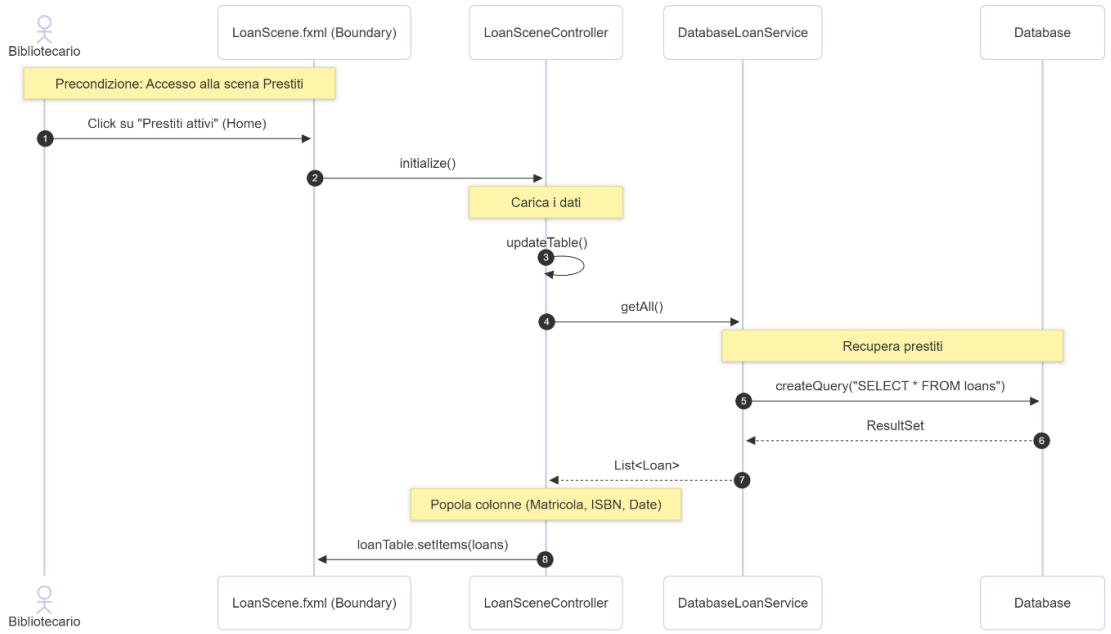
A. Il bibliotecario inserisce nei vari campi di testo i dati relativi al nuovo prestito da registrare.

B. L'*AddLoanSceneController* controlla che i dati siano validi e, tramite BookService, *UserService* e *LoanService*, verifica:

a. che l'utente sia esistente e che non abbia già raggiunto il limite massimo di prestiti;
   che il libro sia disponibile nel catalogo e che ci siano copie libere.

b. Se le verifiche non soddisfano i requisiti, notifica l'errore tramite pop-up. Altrimenti, il prestito viene registrato nel database. Il sistema conferma l'operazione e chiude la finestra di inserimento.

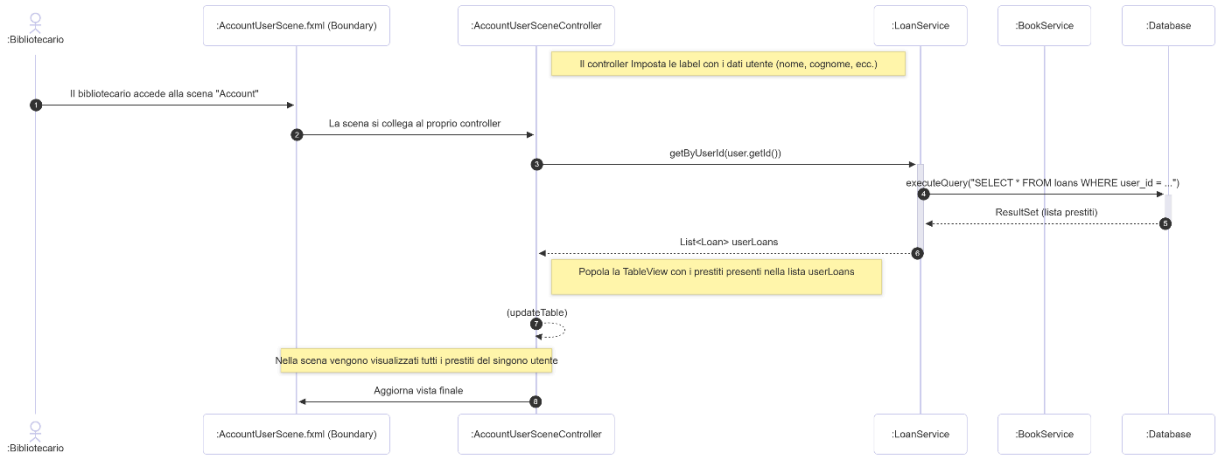
2.12 Visualizzazione lista prestiti attivi



Questo caso d'uso descrive come viene visualizzata la lista dei prestiti attivi dopo essere entrati nella sezione "Gestione prestiti".

- **A.** Il *LoanSceneController* inizializza la tabella richiedendo al service la lista di tutti i prestiti attivi.
- **B.** Il *DatabaseLoanService* esegue una query al database per recuperare la lista dei prestiti attivi.
- **C.** Le informazioni ottenute vengono inserite nella tableView, consentendo al bibliotecario di visualizzare la lista dei prestiti attivi.

2.13 Visualizzazione dei libri presi in prestito da un utente



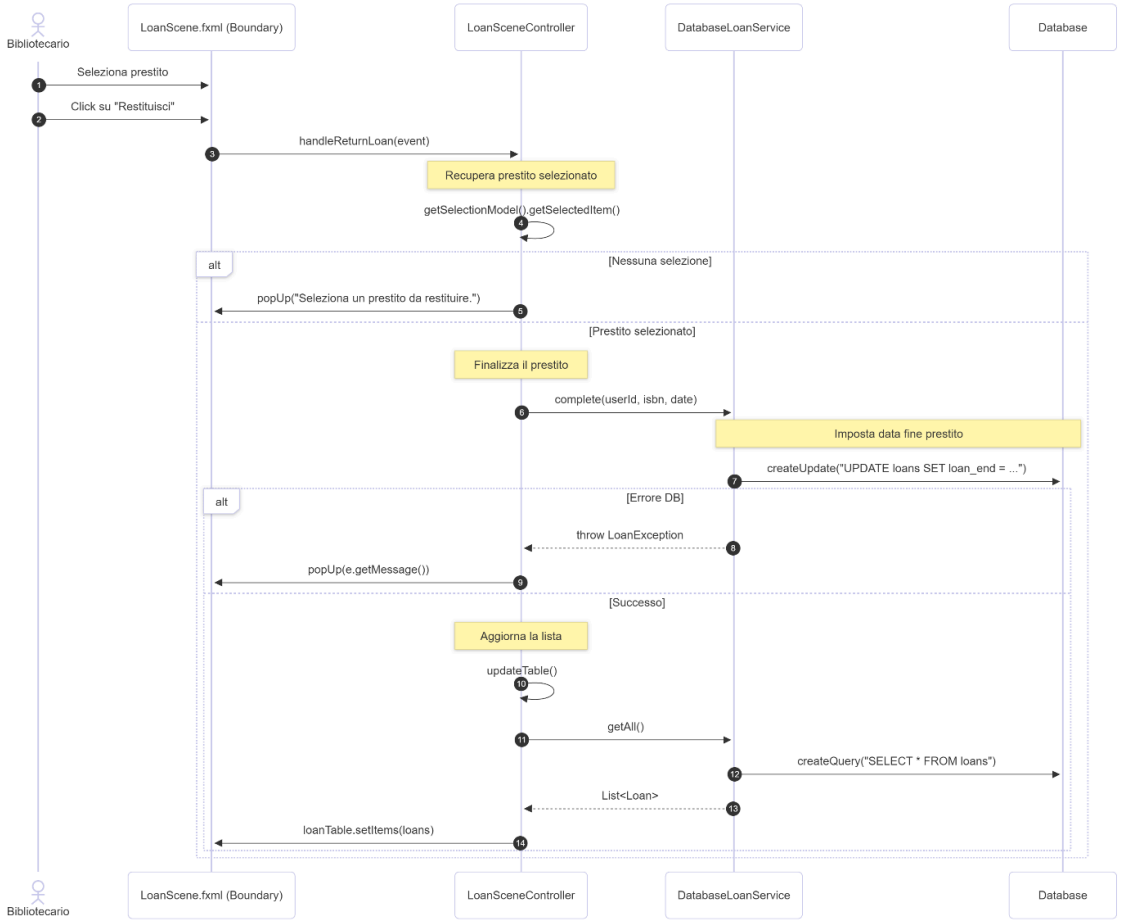
Il diagramma di sequenza illustra il processo con cui il sistema recupera e mostra i prestiti attivi di uno specifico utente:

A. Il Bibliotecario accede alla scena AccountUserScene, attivando il relativo Controller (AccountUserSceneController).

B. Il controller richiede al LoanService di ottenere i prestiti associati all'ID dell'utente (getByUserId); il servizio esegue la query SQL di selezione sul Database e restituisce una lista di oggetti Loan.

C. Il controller utilizza i dati ricevuti per popolare la TableView nell'interfaccia, aggiornando la vista finale per l'attore.

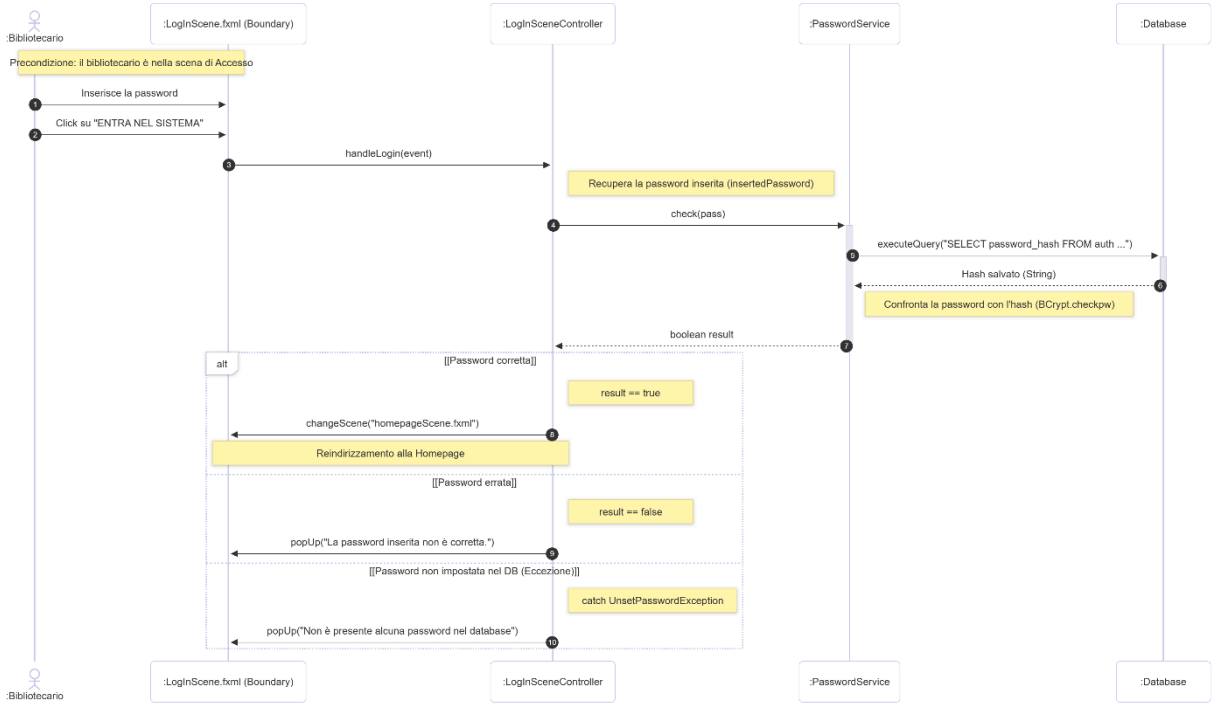
2.14 Registrazione restituzione



Questo caso d'uso descrive come il bibliotecario può estinguere il prestito di un libro dopo essere entrati nella sezione "Gestione prestiti".

- **A.** Il *LoanSceneController* verifica la selezione di un prestito dalla tabella. Se la selezione è stata effettuata, recupera l'ISBN del libro, la matricola dell'utente e la data di fine prestito e invia i dai al *DatabaseLoanService* per la richiesta di aggiornamento sul prestito; altrimenti, mostra un pop-up di errore.
- **B.** Il *DatabaseLoanService* aggiorna il record, registrando la data di restituzione e aumentando il numero di copie disponibili di una unità.
- **C.** Alla fine, il controller ricarica la tabella per mostrare i rimanenti prestiti ancora attivi.

2.15 Accesso con password



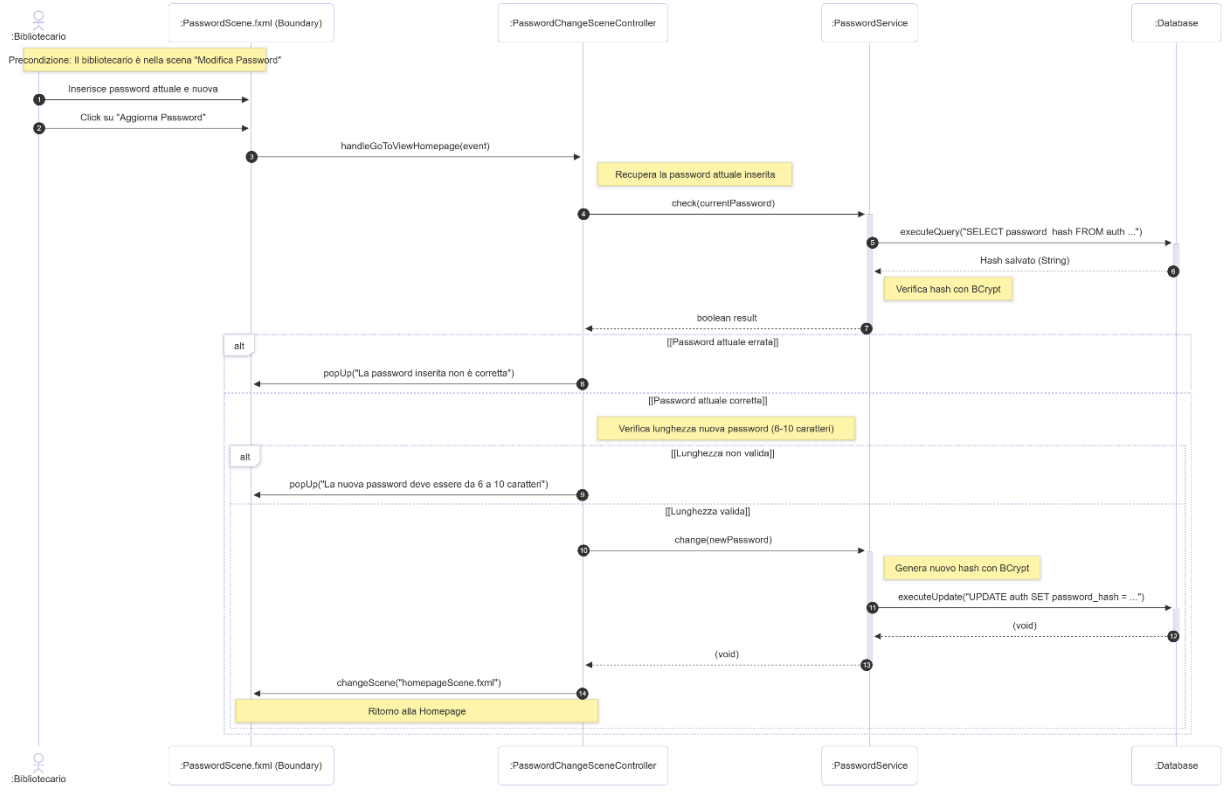
Il diagramma illustra il processo di autenticazione dell'utente:

A. Il Bibliotecario inserisce la password nella LogInScene e avvia il tentativo di accesso.

B. Il controller delega la validazione al PasswordService, che recupera l'hash dal database e lo confronta con la password inserita usando l'algoritmo BCrypt.

C. Il sistema gestisce il risultato attraverso blocchi alternativi: se la password è corretta reindirizza alla Homepage, altrimenti (o in caso di errore tecnico) visualizza un popup di errore.

2.16 Modifica password di accesso



Il diagramma illustra il flusso avviato dal Bibliotecario per aggiornare le proprie credenziali:

- **A.** Il PasswordChangeSceneController verifica la correttezza della password attuale tramite il PasswordService, che controlla l'hash nel database.
- **B.** Se la verifica ha successo, il controller controlla che la nuova password rispetti i vincoli di lunghezza (tra 6 e 10 caratteri).
- **C.** Se i dati sono validi, il servizio aggiorna la password nel database e il sistema reindirizza l'utente alla Homepage; in caso contrario, vengono mostrati messaggi di errore.