DISPEA
DIPARTIMENTO DI
SCIENZE PURE E APPLICATE

Università degli Studi di Urbino "Carlo Bo"

Corso di Laurea in Informatica Applicata

1506
UNIVERSITÀ
DEGLI STUDI
DI URBINO
CARLO BO

Progetto d'esame di

PROGRAMMAZIONE E MODELLAZIONE A OGGETTI

A.A. 2020/2021

Docente: Saverio Delpriori

Nicolae Cociu

Matricola N. 298301

1506
UNIVERSITÀ
DEGLI STUDI
DI URBINO
CARLO BO

DISPEA
DIPARTIMENTO DI
SCIENZE PURE E APPLICATE

Indice

1506
UNIVERSITÀ DEGLI STUDI DI URBINO CARLO BO
DISPEA
DIPARTIMENTO DI SCIENZE PURE E APPLICATE

Specifica del software

*Il fine principale del software è la **gestione di una biblioteca**, agevolando la fase di amministrazione e quella di prestito dei libri da parte degli utenti della biblioteca.*

Si farà riferimento a una singola biblioteca, contenente un numero variabile di libri e di utenti, ogni libro avrà a disposizione un numero limitato di copie disponibili.

*Vi saranno **utenti amministratori** (che avranno privilegi e accesso ai pannelli di gestione della biblioteca) e **clienti** in grado di consultare i libri disponibili e gestirne il prelievo e la restituzione in modo autonomo.*

*Tutto sarà gestito indirettamente da un **Database** realizzato in conformità al progetto realizzato.*

Studio del problema

L'obiettivo del nostro progetto consiste nel realizzare un software in grado di gestire una ***Biblioteca*** *attraverso il* ***linguaggio di programmazione C#.***

L'IDE utilizzato è ***Microsoft Visual Studio Community 2019*** *(App Windows Form (.NET Frameworks 4.7.2), su macchina Windows 10.*

Il pacchetto si suddivide in due ramificazioni principali:

*La " **Dashboard Amministrativa** " a uso dell'utente amministratore;*

*La " **Dashboard Utente** " a disposizione dell'utente cliente della biblioteca.*

Amministratore :

*L'amministratore è un particolare utente che gode dei **privilegi amministrativi**:*

- *può inserire nuovi utenti (Inserimento Utente)*
- *può inserire nuovi libri (Inserimento Libri)*
- *visualizza tutte le informazioni degli utenti (Gestione Utenti, Elenco utenti)*
- *visualizza tutte le informazioni dei libri (Gestione Genere, Gestione Libri, Elenco Libri)*
- *visualizza i prestiti (Gestione Prestiti)*

Ha il completo accesso al database tramite schermate del pannello di amministrazione.

Utenti:

- *Catalogo*
- *Libri Consigliati*
- *Prestito di un libro*
- *Ricerca nel catalogo*
- *Prestiti*
- *Elenco prestiti in corso*
- *Cronologia prestiti passati*
- *Restituzione libro*

Vincoli :

Per quanto concerne i vincoli dell'amministratore consistono nell'immettere nel catalogo della biblioteca nuovi libri nel caso lo ritengano opportuno;
Si suppone che la biblioteca disponga di un numero limitato di copie per ogni libro archiviato nel sistema, quindi quando un cliente preleva un libro questo decade automaticamente dalle copie disponibili per gli altri utenti.

Il prestito avviene secondo le seguenti modalità: Un utente può prendere in prestito un numero non precisato di libri e al momento dell'inserimento del prestito l'utente ha tempo 30 giorni per restituire il libro.
Inoltre sono stati inseriti dei vincoli di correttezza e sicurezza nel login/registrazione in modo tale da preservare la sicurezza dei dati dell'utente che intende accedere al software della Biblioteca.

Dipendenze:

Non sono necessarie conoscenze particolari per utilizzare il software.
Gli amministratori devono essere solamente istruiti per l'utilizzo del software in maniera adeguata, in maniera tale da poter risolvere ogni possibile dubbio derivante dall'applicazione di esso.

I punti critici del software sono legati alle principali funzionalità del programma:

Criticità	01
Descrizione	Inserimento di un utente
Input	In input sono previsti i campi: -- Nome -- Cognome -- Email -- Città -- Genere Preferito -- Password -- Conferma Password
Processo	L'amministratore inserisce i dati dell'utente dichiarati in input, il sistema controlla che l'utente non esista già (tramite controllo email) e ne approva l'inserimento. In caso di record duplicato avvisa con messaggio di errore.
Visibilità	Amministratore / Cliente
Output	Inserimento di un nuovo utente nel database, messaggio di conferma o messaggio di errore a seconda dei casi. Nel caso in cui l'utente si registra correttamente può immediatamente accedere ai servizi di login tramite il modulo di Login.

Criticità	02
Descrizione	Gestione utenti
Input	Non è previsto nessun input in quanto è un modulo di sola visualizzazione dati.
Processo	L'amministratore entra nel pannello e visualizza una griglia di utenti registrati nel sistema. Tramite la TabControl1 è possibile Modificare o Eliminare uno o più utenti.
Visibilità	Amministratore
Output	Se non si verificano anomalie viene mostrata una tabella con le informazioni riguardanti tutti gli utenti presenti nel sistema. Le informazioni della tabella riportano: -- ID Utente -- Nome -- Cognome -- Email -- Nome Città -- Nome Genere

Criticità	03
Descrizione	Gestione Libri
Input	In input sono previsti i campi: -- Elenco dei Libri -- Elenco dei Generi -- Data di uscita -- Nome del Libro -- Nome del Libro da modificare
Processo	L'amministratore inserisce in input mediante una Combobox un libro dal catalogo e l'amministratore ha la possibilità di Eliminare o Modificare quel determinato libro oppure inserirne uno nuovo specificandone il genere di appartenenza.
Visibilità	Amministratore
Output	Se non si verificano eccezioni viene visualizzato un messaggio di avvenuto inserimento, altrimenti viene mostrato un messaggio di errore (Errore nel inserimento del libro ecc...).

Criticità	04
Descrizione	Prenotazione libro
Input	In Input vengono visualizzati Username e Genere Preferito dell'utente con la possibilità di visionare interamente il catalogo dei libri oppure esclusivamente del genere preferito
Processo	Cliccando sul pulsante "Prenota" viene notificata la conferma della prenotazione del libro e automaticamente il Database detrae una copia della disponibilità di quest'ultimo.
Visibilità	Cliente
Output	Se non si verificano anomalie viene mostrato un messaggio di prenotazione avvenuta, altrimenti si visualizza un messaggio di errore.

Criticità	05
Descrizione	Restituzione libro
Input	Inserimento, tramite ComboBox, del libro da restituire.
Processo	Dopo che il cliente ha selezionato il libro, e cliccando sul pulsante "Restituisci" il sistema inserisce la data di restituzione del libro nel sistema e il libro è restituito.
Visibilità	Cliente
Output	Se non si verificano errori viene visualizzato un messaggio di reso avvenuto.

Nei precedenti paragrafi è stato descritto cosa il sistema deve fare, ovvero i punti critici del sistema e che ne costituiscono quindi i requisiti funzionali;

Vi sono inoltre ulteriori requisiti che devono essere soddisfatti al fine di rendere il software più completo e funzionale possibile:

Manutenibilità

Deve essere possibile effettuare operazioni di modifica al sistema quali la correzione di eventuali errori e modifiche per migliorare la qualità del software.

Le varie funzioni sono modulari ed è di facile comprensione l'inserimento di nuove funzioni.

Usabilità

Il sistema è pensato per una vasta fascia di utenti, quindi il suo utilizzo deve essere semplice anche a coloro che non hanno un'elevata dimestichezza con le tecnologie informatiche.

Robustezza

Nel caso in cui si verifichino situazioni non previste dalle specifiche (dati di input non corretti, carico di lavoro elevato, etc) il sistema deve essere in grado di tornare in uno stato consistente.

Sicurezza

Devono essere stabilite politiche di controllo degli accessi, in modo che l'utente possa accedere soli ai servizi di sua competenza. Poiché questo sistema manipola dati personali è stato deciso di consentire l'accesso al sistema tramite un username ed una password per tutti gli attori.

Tramite la procedura di login è anche possibile determinare se l'utente è un cliente o un amministratore del sistema e proporgli dunque una maschera di amministrazione o semplice visualizzazione del catalogo.

Scelte architetturali :

Molte applicazioni mostrano una schermata iniziale per far sapere all'utente che il programma si sta caricando e per aggiungere un tocco di stile al loro programma.

Questo approccio ci consente di utilizzare qualsiasi modulo di Windows come splashscreen. Nasconde automaticamente lo schermo quando il primo Windows Form dell'applicazione termina il caricamento e viene visualizzato.

Inoltre mostra la schermata iniziale all'uscita dell'applicazione.

Appena avviata l'applicazione ci troveremo innanzi a questa breve schermata mostrata di seguito:

Questo metodo utilizza un timer che da come si può notare è stato implementato con una if condition (una volta che il pannello raggiunge la lunghezza del form automaticamente essa si interrompe passando alla schermata successiva di Login)

```
1 riferimento
private void timer1_Tick(object sender, EventArgs e)
{
    panel2.Width += 3;
    if (panel2.Width >= 750)
    {
        timer1.Stop();
        formLogin login = new formLogin();
        login.Show();
        this.Hide();
    }
}
```

Una volta caricata la schermata, l'utente si troverà davanti al form di Login e in base alle credenziali inserite il software riconoscerà l'utente e lo distinguerà in cliente o amministratore.

```
// FUNZIONE LOGIN - UTENTE//

1 riferimento
public bool AccediUtente(Utenti ut)
{
    bool check;
    DataTable tb = new DataTable();
    cmd = new SqlCommand("SELECT email,password FROM utenti WHERE utenti.email=@email AND utenti.password=@password AND utenti.IsAdministrator=0", connection);
    cmd.Connection = connection;
    cmd.Parameters.Add("@email", SqlDbType.VarChar, 255).Value = ut.email;
    cmd.Parameters.Add("@password", SqlDbType.VarChar, 255).Value = ut.password;

    cmd.ExecuteNonQuery();
    SqlDataAdapter adp = new SqlDataAdapter();
    adp.SelectCommand = cmd;
    adp.Fill(tb);
    if (tb.Rows.Count == 1)
    {
        check = true;
    }
    else
    {
        check = false;
    }

    return check;
}
```

Come possiamo notare nella quasi totalità dei form sono state stanziate delle variabili pubbliche necessarie alla connessione con il Database in particolare SqlConnection la quale rappresenta una connessione aperta al nostro database di SQL Server.

SqlConnection rappresenta una sessione univoca per un'origine dati SQL Server.

Con un sistema di database client/server, equivale a una connessione di rete al server.

SqlConnection viene in particolare utilizzato insieme a

SqlDataAdapter e SqlCommand per migliorare le prestazioni durante la connessione a un database di Microsoft SQL Server.

Per quanto riguarda SqlDataAdapter esso Funge da ponte tra DataSet (che Rappresenta una cache in memoria dei dati) e SQL Server per il recupero e il salvataggio dei dati.

SqlDataAdapter Fornisce questo Bridge per mapping Fill , che modifica i dati in DataSet in modo che corrispondano ai dati nell'origine.

L'aggiornamento viene eseguito in base alle righe. Per ogni riga inserita, modificata ed eliminata, il metodo Update determina il tipo di modifica che è stata eseguita (Insert , Update o Delete).

Quando l'oggetto SqlDataAdapter Compila un oggetto DataSet , vengono create le tabelle e le colonne necessarie per i dati restituiti, se non esistono già.

Per quanto riguarda invece SqlCommand Quando viene creata un'istanza di SqlCommand , le proprietà di lettura/scrittura vengono impostate sui rispettivi valori iniziali.

Essa rappresenta un'istruzione Transact-SQL da eseguire in relazione a un database SQL Server, in quanto tutti gli strumenti e le applicazioni che comunicano con un database SQL inviano comandi T-SQL.

Pertanto nel mio caso ho stanziato una variabile pubblica stringa "connectionString" e un valore che ho chiamato "cmd" come SqlCommand utilizzato nella costruzione dei vari metodi implementativi mediante costruzione di Query.

In particolare ho utilizzato ExecuteNonQuery per eseguire operazioni di catalogo, ad esempio l'esecuzione di query sulla struttura di un database o la creazione di oggetti di database quali tabelle per esempio nel metodo "Accedi Amministratore":

```
cmd = new SqlCommand("SELECT email,password FROM utenti WHERE utenti.email=@email AND
utenti.password=@password AND utenti.IsAdministrator=1", connection);
```

Ci permetteva di accedere alla Dashboard Amministratore grazie alla parte della Query finale ovvero utenti.IsAdministrator=1 dove è stato assegnato 1 come valore vero e 0 come valore falso ovvero cliente.

Pertanto ho stanziato una variabile pubblica stringa "connectionString" e un valore che ho chiamato "cmd" come SqlCommand utilizzato nella costruzione dei vari metodi implementativi mediante costruzione di Query.

```
1 riferimento
private void formLogin_Load(object sender, EventArgs e)
{
    try
    {
        // CONNESSIONE DATABASE //
        connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=" + Application.StartupPath + @"\Database\BibliotecaDB.mdf;Integrated Security=True;Connect Timeout=30";
        connection = new SqlConnection(connectionString);
        connection.Open();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Errore nel caricamento: " + ex.Message);
    }
}
```

```
1 riferimento
private void txtEmail_Leave(object sender, EventArgs e)
{
    string pattern = @"^([a-zA-Z0-9_\-\.]+)@([a-zA-Z0-9_\-\.]+)\.([a-zA-Z]{2,5})$";
    Regex reg = new Regex(pattern, RegexOptions.None);
    if (reg.IsMatch(Convert.ToString(txtEmail.Text)))
    {
        checkEmail = true;
    }
    else
    {
        checkEmail = false;
    }
}
```

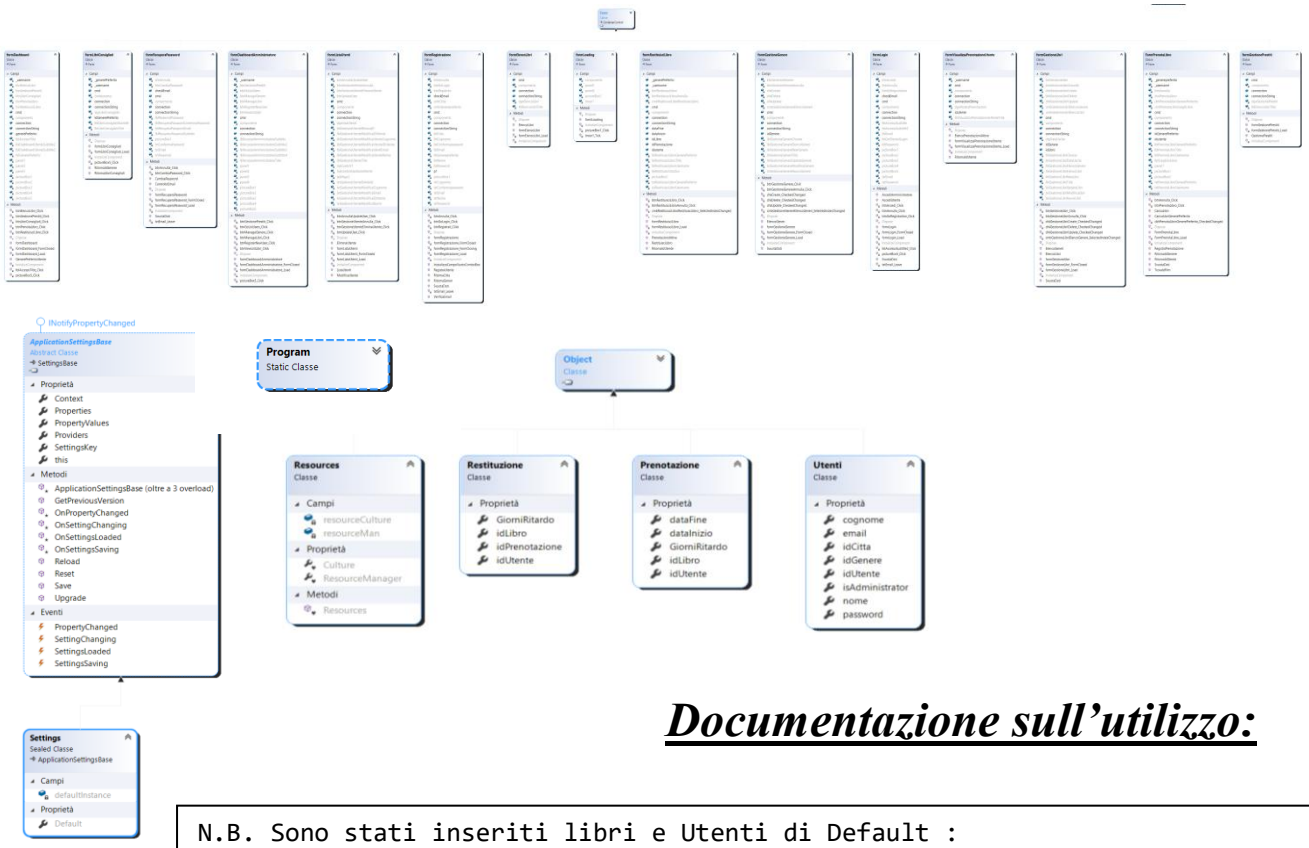
Un ulteriore funzione aggiunta riguarda il metodo Regex.Match ("string pattern") che cerca in una stringa di input una sottostringa che corrisponda al criterio di espressione regolare e restituisce la prima ricorrenza come singolo oggetto Match nel nostro caso.

```
string pattern = @"^([a-zA-Z0-9_\-\.]+)@([a-zA-Z0-9_\-\.]+)\.([a-zA-Z]{2,5})$";
```

In modo tale che l'email soddisfi dei parametri particolari e una composizione corretta (ovvero la presenza della @).

In definitiva il progetto si basa principalmente su un Database di fondamentale importanza dalla quale vengono estrapolati i dati e nel quale sono frequenti i passaggi di input e di output.

Questi passaggi avvengono mediante la formulazione di Query che evidenziano che sono alla base del progetto. Ogni eventuale modifica per quanto riguarda i dati inseriti può essere effettuata sia tramite comandi impostati nel software (Inserimento, Modifica, Eliminazione Libri, Genere, Utenti) sia tramite il Database principale modificando le varie tabelle.



Documentazione sull'utilizzo:

N.B. Sono stati inseriti libri e Utenti di Default :
1) Nel caso in cui si voglia accedere come amministratore le credenziali sono:

E-mail : verdi.matteo@email.it Password: 09876

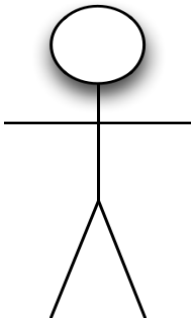
2) Nel caso in cui si voglia accedere come cliente le credenziali sono:

E-mail : rossi.mario@email.it Password: 12345

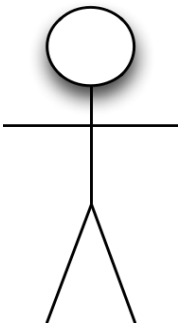
Use Case – Casi D'uso

Questo schema serve per identificare le operazioni che ogni utente può effettuare nel sistema.

-- Non viene specificato come vengono implementate le funzioni.
-- Cattura le funzionalità del sistema dal punto di vista degli utenti --
Raccoglie un insieme di use case ed attori e le relazioni tra essi.
-- Rappresenta le modalità di utilizzo del sistema da parte di uno o più utilizzatori (attori).
-- Descrive l'interazione tra attori e sistema, non la "logica interna" della funzione.
-- Descrive l'organizzazione ed il modello del comportamento di un sistema.



Gli attori del sistema sono pertanto due:



Un attore del sistema:

-- *Solitamente è una classe di persone fisiche (es. Amministratore, Cliente)*
-- *Controlla le funzionalità del software*
-- *Fornisce input o riceve output dal sistema*
-- *Può partecipare a più use case*
-- *Può rivestire più ruoli nel medesimo use case*

Di seguito riportiamo il legame tra gli attori e le macrofunzionalità del sistema.
Si è deciso di dividere le funzionalità in base ai due attori che partecipano all'interazione con il software.
Cliente ed Amministratore condividono gli use case:

-- Elenco libri

Le schermate sono del tutto identiche per i due attori.
Il primo grafico che si propone è una visione d'insieme degli use case dei due attori: nel sistema bibliotecario abbiamo infatti sia i pannelli per l'amministratore che per il cliente.

Successivamente si riportano due grafici, il primo rappresenta i casi d'uso dell'amministratore, mentre il secondo quelli del cliente.

Per quanto riguarda l'amministratore si è deciso di suddividere i casi d'uso in tre sottogruppi principali:

-- Gestione dei libri
-- Gestione degli utenti
-- Gestione dei generi
-- Gestione dei Prestiti

Si descrivono alcune delle situazioni d'uso tipiche, tramite scenari d'utilizzo da parte dell'amministratore e successivamente del cliente.

Caso d'uso: Login

ID: *UC1*

Attori: *Amministratore / Cliente*

Pre-Condizioni:

• Caricamento Form di Loading

• Focus sulla schermata principale

Corso degli eventi:

1) L'attore inserisce E-mail e Password
2) Viene notificato tramite messaggio l'avvenuto Login tramite Amministratore / Cliente

Post-condizione: UC2 o UC4

Percorsi Alternativi:
1) Avviso: Email o Password errati
2) Avviso: Email non del formato corretto
3) Avviso: Lunghezza password non valida
4) Hai dimenticato la Password? Clicca qui per recuperarla → indirizzamento sul Form Recupera Password

5) Button: Registrati → indirizzamento sul Form Registrazione

Caso d'uso: Inserimento nuovo libro

ID: *UC2*

Attori*: Amministratore*

Pre-Condizioni:

• Caricamento Form di Loading

• UC1 (LOGIN)

Corso degli eventi:

1) L' Amministratore tramite la Dashboard principale seleziona il servizio Gestione Libri
2) Seleziona la spunta su "Crea nuovo Libro"
3) Imposta Genere del Libro e Data di Uscita
4) Imposta il Nome del nuovo Libro
5) Salva

Post-condizione: *Il Nuovo Libro viene automaticamente inserito nel Database della Biblioteca.*

Percorsi Alternativi:

Avviso: Errore nell'effettuare la modifica sul libro:

Caso d'uso: Gestione Utenti

ID: *UC3*

Attori: *Amministratore*

Pre-Condizioni:

• Caricamento Form di Loading

• UC1 (LOGIN)

Corso degli eventi:

1) L' Amministratore tramite la Dashboard principale seleziona il servizio Utenti
2) Visualizza tramite il DataGridView la Lista degli Utenti
3) Mediante il TabControl Seleziona IdUtente che vuole eliminare
4) Button: Elimina

Post-condizione: *Viene visualizzato in contemporanea l'eliminazione dell'utente dal Database della Biblioteca*

Percorsi Alternativi:

Button: Annulla (Annulla l'operazione, quindi svuota la TextBox (IdUtente)
Tramite la 2° sezione l'Amministratore può modificare E-mail, Nome e Cognome dell'Utente selezionato dal DataGridView.

Caso d'uso: Prenotazione Libro

ID: *UC4*

Attori: *Cliente*

Pre-Condizioni:

• Caricamento Form di Loading

• UC1 (LOGIN)

Corso degli eventi:

1) L' Utente tramite la "Dashboard Utente" seleziona il servizio "Prenota"
2) Viene visualizzato Username e Genere Preferito dell'Utente
3) Utente Sceglie un Libro consigliato appositamente per lui in base al genere preferito
4) Seleziona il Libro desiderato
5) Prenota
6) Viene notificato tramite messaggio l'avvenuta Prenotazione del Libro

Post-condizione: Il libro prenotato appare nella schermata dei Prestiti Attivi con valore 0 (in possesso dell'utente) e inizano a decorrere i 30 giorni di prestito.

Percorsi Alternativi:

Avviso: Errore nell'effettuare la prenotazione del libro:
L'Utente intende Prenotare un qualsiasi libro disponibile nel catalogo pertanto seleziona la spunta su "Mostra Tutti i Libri" e una volta selezionato lo Prenota.

Caso d'uso: Restituzione Libro

ID: *UC5*

Attori: *Cliente*

Pre-Condizioni:

• Caricamento Form di Loading

• UC1 (LOGIN)

Corso degli eventi:

1) L' Utente tramite la "Dashboard Utente" seleziona il servizio "Restituisci Libro"
2) Viene visualizzato Username e Genere Preferito dell'Utente
3) Utente seleziona il Libro che intende restituire
4) Button: Restituisci
6) Viene notificato tramite messaggio l'avvenuta Restituzione del Libro

Post-condizione: *Il libro viene restituito e viene visualizzato nella schermata Prestiti Attivi con valore 1 (riconsegnato alla Biblioteca)*

Percorsi Alternativi:

Avviso: Errore nell'effettuare la restituzione del libro:
Button: Annulla (Svuota le TextBox → possibilità di una nuova restituzione)