

# Processo e Sviluppo Software: Assignment 3

Ivo Junior Bettini - 806878, Umberto Cocca - 807191,  
Silvia Traversa - 816435  
GitLab repository

## Applicazione

L'applicazione BooksLoan permette agli utenti di visualizzare il catalogo di una biblioteca. Essi possono visualizzare sia le copie disponibili e non, dei libri ed eventualmente richiederne una in prestito. È inoltre possibile visualizzare le informazioni relative a ogni singolo libro, quali l'autore o la presenza di sequel.

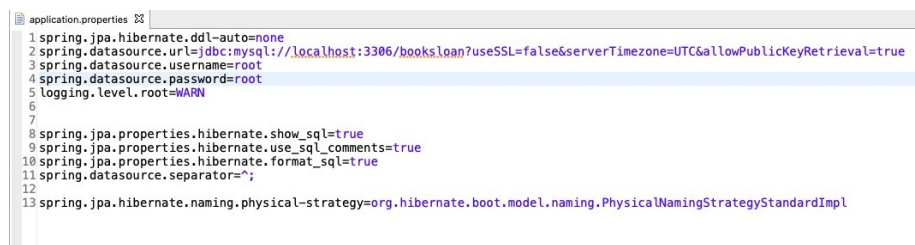
Gli amministratori, che si dividono in amministratori con contratto a tempo indeterminato e amministratori con contratto a tempo determinato, differiscono per la possibilità di eliminare i libri. Possono entrambi interagire con un'area CMS eseguendo il CRUD delle principali entità. Questa differenziazione di operatività nasce con l'idea di dare maggior responsabilità e controllo.

## Esecuzione dell'applicazione

### Preparazione Database

Prima di eseguire l'applicazione, è necessario importare i dati e la struttura del database, creato con MySQL.

Per effettuare questa operazione bisogna effettuare su MySQLWorkbench un Data Import del file *DumpFinale* presente nella cartella *dumps*. Successivamente, per permettere la connessione al database, è richiesta la modifica del file *src/main/resources/application.properties* inserendo le proprie credenziali MySQL nei campi *spring.datasource.username* e *spring.datasource.password*, come illustrato nella figura.



```
application.properties
1 spring.jpa.hibernate.ddl-auto=none
2 spring.datasource.url=jdbc:mysql://localhost:3306/booksloan?useSSL=false&serverTimezone=UTC&allowPublicKeyRetrieval=true
3 spring.datasource.username=root
4 spring.datasource.password=root
5 logging.level.root=WARN
6
7
8 spring.jpa.properties.hibernate.show_sql=true
9 spring.jpa.properties.hibernate.use_sql_comments=true
10 spring.jpa.properties.hibernate.format_sql=true
11 spring.datasource.separator=";";
12
13 spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl
```

## Avvio Applicazione

Per far partire l'applicazione, dopo aver clonato il repository, eseguire le seguenti righe di comando:

```
./mvnw clean package spring-boot:repackage
```

```
java -Djava.security.egd=file:/dev/./urandom -jar target/BooksLoan-1.jar
```

L'applicazione sarà disponibile all'indirizzo <http://localhost:8080>.

Per accedere alle funzionalità dell'applicazione è richiesto un login, che è possibile effettuare con tre tipi diversi di account, rappresentativi di categorie:

- **amministratore con contratto a tempo indeterminato:**

username: 123 password: 1234

- **amministratore con contratto a tempo determinato:**

username: 456 password: 4567

- **cliente:**

username: 321 password: 1234

## Schema E-R

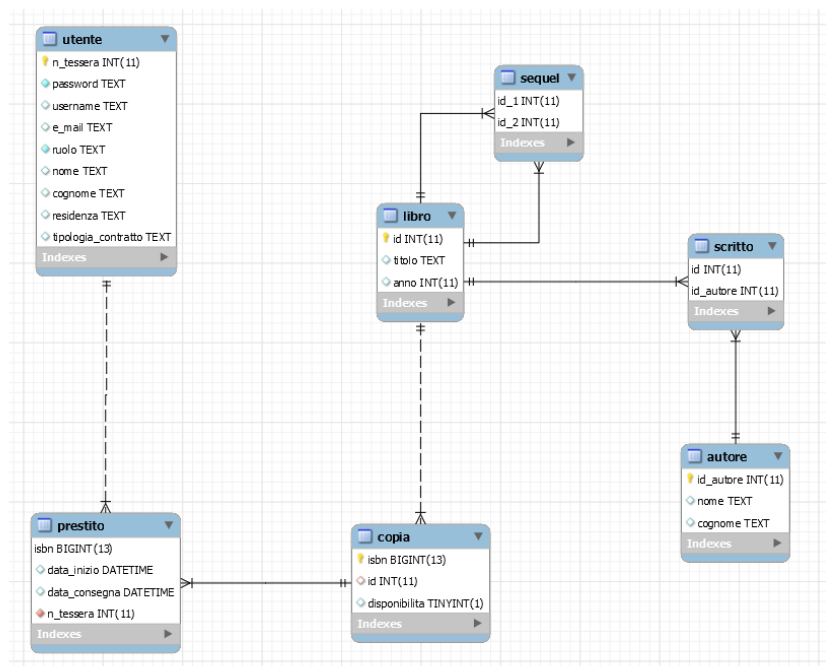


Figure 1: Schema ER

Le entità utilizzate nel nostro progetto sono così descritte:

- Utente: descrive la persona registrata e che utilizza il sito. Gli attributi `n_tessera` e `password` rappresentano le credenziali d'accesso al sito, mentre l'attributo `ruolo` determina se l'utente è un amministratore o un utente.
- Libro: elemento che compone la nostra libreria online.
- Copia: insieme di diverse edizioni di un libro che possiedono l'ISBN come chiave primaria. Una copia può trovarsi in due stati, disponibile o non disponibile (in quest'ultimo caso non può essere presa in prestito).
- Autore: colui che ha scritto uno o più libri.

Esse sono messe in relazione da:

- Prestito: entità che relaziona utente con copia. Un utente può prenotare più copie dello stesso libro.
- Sequel: entità che mette in relazione un libro con se stesso. Un libro può possedere zero o più sequel (nell'applicazione rendiamo visibile solo il diretto successivo).
- Scritto: entità che mette in relazione autore con libro. Un libro può essere scritto da più autori e un autore può scrivere più libri.

## Pattern MVC

Ai fini del progetto si è utilizzato il pattern architetturale Model-view-controller (MVC), per separare la logica di presentazione dei dati dalla logica di business.

Il package **`net.assignment.booksLoan.model`** contiene al suo interno l'implementazione delle classi presenti nello *Schema E-R*. Sono state annotate le classi e i rispettivi membri con le annotazioni della Java Persistence API (JPA), un framework che nasce per garantire la persistenza dei dati.

Per mezzo del Controller viene implementata la logica di controllo dell'applicazione. All'interno delle classi Controller vengono eseguiti i metodi per gestire le richieste Get e Post dell'utente reperendo le informazioni attraverso i layer repository che permettono di dare un'ulteriore livello di astrazione sull'accesso dei dati, potendo eseguire il CRUD, e grazie ai layer service che espongono la logica di business ottenendo così i Model da mostrare in View.

## BooksLoan

Per poter accedere all'applicazione bisogna obbligatoriamente essere registrati. Attraverso l'utilizzo della dipendenza Spring Security viene controllato l'accesso, impedendone la navigazione tra le pagine se non si è registrati e gestendone i livelli d'accesso. Nello specifico un cliente non può accedere alle pagine di un amministratore.

Abbiamo deciso di distinguere l'amministratore in due diverse tipologie, che si differenziano in base al loro tipo di contratto. Questa scelta è stata fatta pensando all'organizzazione interna di una biblioteca, e anche perchè in questo modo abbiamo creato diversi livelli di privilegi e di operazioni eseguibili da un utente. L'amministratore con un contratto a tempo indeterminato, una volta che ha eseguito il login, ha la possibilità di effettuare ogni tipo di operazione: può eliminare, aggiungere, modificare libri e impostare le copie, gli autori e i sequel di un libro. Un amministratore con contratto a tempo determinato, invece, non può cancellare i libri e le sue relative copie.

Il cliente, una volta effettuato il login, ha a disposizione l'elenco dei libri del catalogo della biblioteca. Questi sono riportati in una tabella in cui è possibile filtrare cercando per chiave. Il filtraggio è stato eseguito utilizzando jQuery. Inoltre, il cliente può visualizzare se è presente una copia disponibile di un determinato libro ed eventualmente prenderla in prestito, può visualizzare i suoi prestiti in corso e restituire una copia. La restituzione rende disponibile la copia stessa e quindi accessibile da altri clienti per la prenotazione.