



Advanced Data Bases TC3041 – ITESM Puebla  
Professor: Dr. Daniel Pérez Rojas

## Homework 2.4 – Library DB

---

20/03/2018

Rafael Antonio Comonfort Viveros  
Angel Roberto Ruíz Mendoza

A01324276  
A01324489

This homework required a front-end for our previously developed database of a library. However, only 2 CRUD scenarios were asked from us and so, we chose to elaborate on Loans and Returns, which are the main action one can do on a library. We soon realized that to make a loan, one would need to see what books are available, to return a book, fines would need to be displayed in case it wasn't returned on time, etc. This meant we would have to do some extra functions and files to make the system work.

## Implementation


We followed extremely closely the latest PHP example project provided in class. The idea behind it, is using a class corresponding to the desired entity, and write a method for retrieving, updating, deleting and inserting from and into the database. Said PHP class has the same attributes as its database counterpart, with setters for external files to modify them. We also declared interfaces for each class, specifying all methods to implement (save, get, delete and update). We generated the interface and class for the books, returns, fines and loans, although a class like that of books, only has a function to get the tuples from the database, as we didn't want to let the user add new books, update them or delete them. You can find the interfaces in the "interfaces" folder, and the class files in the "models" folder.

The Database.php file is where the connection to Postgres occurs. It is included and used in all models to perform their database methods. We took the file provided in the sample project and simply changed the user, database name and password to match that of The Server (where this homework is to be mounted).

In the "app" folder, all PHP files concerning the front-end are placed. We have a list file for returns, fines, loans and loans to return. The latter being used when making a new return, and we wish to know for which loan we are making a return. We have an add, delete PHP file for books and loans. The former makes use of a save file to actually create an instance of that object, reads the form declared with the add file, and calls the save function. The most complex files are the list ones, where dynamically, all tuples are retrieved from the database using the "get" model's function, and then displayed to the user with edit and delete buttons if it applies. Said buttons are bound to call other PHP files with the ID of the tuple or with the entire instance as parameter. Notice that all files that include HTML code in them, also have the left side navbar imposed on them.

At index.php, the server is told to go to the home landing page, which includes an image. A bootstrap template was used, to provide a better visual appeal to our homework. You will notice too, that it is the same utilized for the last front-end homework.

## Screenshots on Server

A screenshot of a terminal window titled 'usuario4@servi: ~'. The terminal shows a series of PostgreSQL commands and their outputs. The commands include dropping cascades, creating an enum type 'lan', dropping and creating tables 'books', 'authors\_books', and 'books\_loans' with various constraints and foreign keys. The output shows the success of these operations and the specific details of the cascades being dropped.

```
usuario4@servi: ~
NOTICE: drop cascades to 3 other objects
DETAIL: drop cascades to table books column language
drop cascades to view allbooks
drop cascades to view allbooks column language
DROP TYPE
usuario4=> CREATE TYPE lan AS ENUM (
usuario4(>   'En',
usuario4(>   'Es',
usuario4(>   'Fr');
CREATE TYPE
usuario4=>
usuario4=> DROP TABLE IF EXISTS books CASCADE;
NOTICE: drop cascades to 2 other objects
DETAIL: drop cascades to constraint authors_books_bookid_fkey on table authors_books
drop cascades to constraint books_loans_bookid_fkey on table books_loans
DROP TABLE
usuario4=> CREATE TABLE books(
usuario4(>   bookID SERIAL,
usuario4(>   title VARCHAR(150) NOT NULL,
usuario4(>   editorialID INT REFERENCES editorials(editorialID),
usuario4(>   edition INT NOT NULL,
usuario4(>   translator VARCHAR(150),
usuario4(>   language lan NOT NULL,
usuario4(>   daily_fine_amount NUMERIC(10,2) NOT NULL,
usuario4(>   stock INT NOT NULL,
usuario4(>   pages INT,
usuario4(>   publishing_date DATE,
usuario4(>   PRIMARY KEY (bookID)
usuario4(> );
CREATE TABLE
usuario4=>
usuario4=> DROP TABLE IF EXISTS authors_books CASCADE;
DROP TABLE
usuario4=> CREATE TABLE authors_books(
usuario4(>   authorID INT REFERENCES authors(authorID),
usuario4(>   bookID INT REFERENCES books(bookID),
usuario4(>   PRIMARY KEY (authorID, bookID)
usuario4(> );
CREATE TABLE
usuario4=>
usuario4=> DROP TABLE IF EXISTS books_loans CASCADE;
DROP TABLE
usuario4=> CREATE TABLE books_loans(
usuario4(>   bookID INT REFERENCES books(bookID),
usuario4(>   loanID INT REFERENCES loans(loanID),
usuario4(>   PRIMARY KEY (bookID, loanID)
usuario4(> );
```

*Tables creation on server's Postgres*

```

usuario4@servi: ~
INSERT 0 4
usuario4=>
usuario4=> INSERT INTO books (title, editorialID, edition, translator, language, daily_fine_amount, stock, pages, publishing_date) VALUES ('A Pattern Language', 1, 1, NULL, 'En', 34.99, 3, 1171, '1977-01-01'), ('A Game of Thrones', 2, 5, NULL, 'En', 50.00, 15, 694, '1996-08-01'), ('A Clash of Kings', 2, 5, NULL, 'En', 55.00, 17, 768, '1998-01-01'), ('A Storm of Swords', 2, 4, NULL, 'En', 60.00, 20, 973, '2000-01-01'), ('A Feast for Crows', 2, 3, NULL, 'En', 40.00, 7, 976, '2005-01-01'), ('A Dance with Dragons', 2, 2, NULL, 'En', 40.00, 7, 1040, '2011-07-12'), ('The Name of The Wind', 3, 2, NULL, 'En', 50.00, 15, 662, '2007-03-27'), ('Fahrenheit 451', 4, 10, NULL, 'En', 55.50, 20, 358, '1953-01-01');
INSERT 0 8
usuario4=>
usuario4=> INSERT INTO authors (name, nationality) VALUES ('Christopher Alexander', 'AUT'), ('Sara Ishikawa', 'JPN'), ('Murray Silverstein', 'USA'), ('George R. R. Martin', 'USA'), ('Patrick Rothfuss', 'USA'), ('Ray Bradbury', 'USA');
INSERT 0 6
usuario4=>
usuario4=> INSERT INTO authors_books (authorID, bookID) VALUES (1, 1), (2, 1), (3, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (5, 7), (6, 8);
INSERT 0 10
usuario4=>
usuario4=> INSERT INTO clients (name, telephone, address, email, debt) VALUES ('Angel Bob Menendez', '2227564899', 'Avenida Siempre Viva #2000', 'mrbbob@gmail.com', 4321.00), ('Carlos McLovin Manilla', '2224564788', 'Blvd. Gutenberg, #4 Int. 20', 'mclov@domains.com', 0.00);
INSERT 0 2
usuario4=>
usuario4=> INSERT INTO librarians (name, telephone, address, shift_begin, shift_end) VALUES ('Pepe Toño Pantufla', '22245453467', 'Evergreen Terrace Road 48', '07:00:00', '20:00:00'), ('Daniel P. Irri', '22245453667', NULL, '20:00:01', '06:00:59');
INSERT 0 2
usuario4=>
usuario4=> INSERT INTO loans (clientID, loan_date, return_date, librarianID) VALUES (1, '2018-03-01', '2018-03-08', 1), (2, '2018-02-18', '2018-02-25', 2), (1, '2018-03-01', '2018-03-08', 1);
INSERT 0 3
usuario4=>
usuario4=> INSERT INTO books_loans (bookID, loanID) VALUES (2, 1), (3, 1), (4, 1), (2, 2), (3, 2), (4, 2), (5, 2), (6, 2), (8, 3);
INSERT 0 9
usuario4=>
usuario4=> INSERT INTO breturns (loanID, actual_return_date) VALUES (2, CURRENT_DATE);
INSERT 0 1
usuario4=>
usuario4=>

```

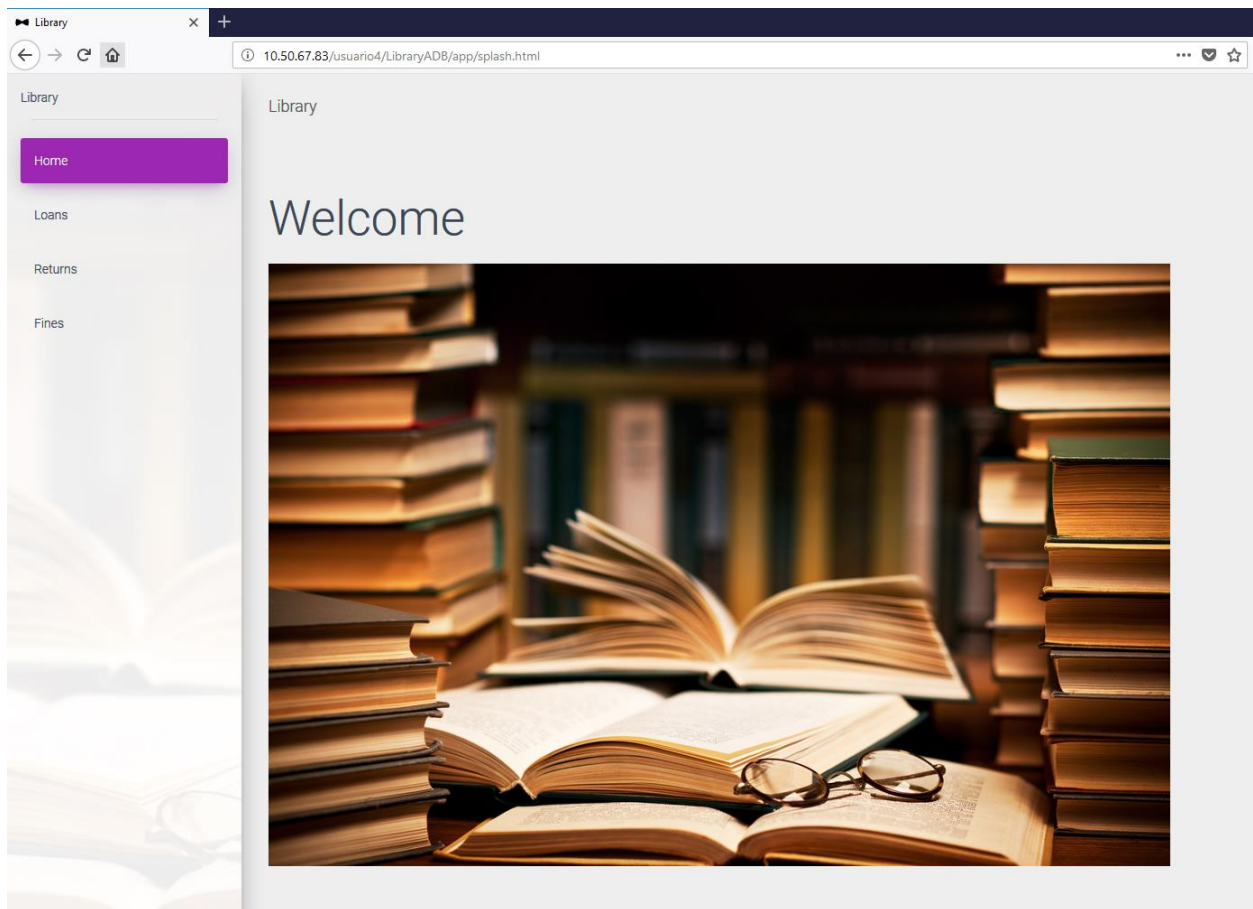
*Database seeding for all tables*

```

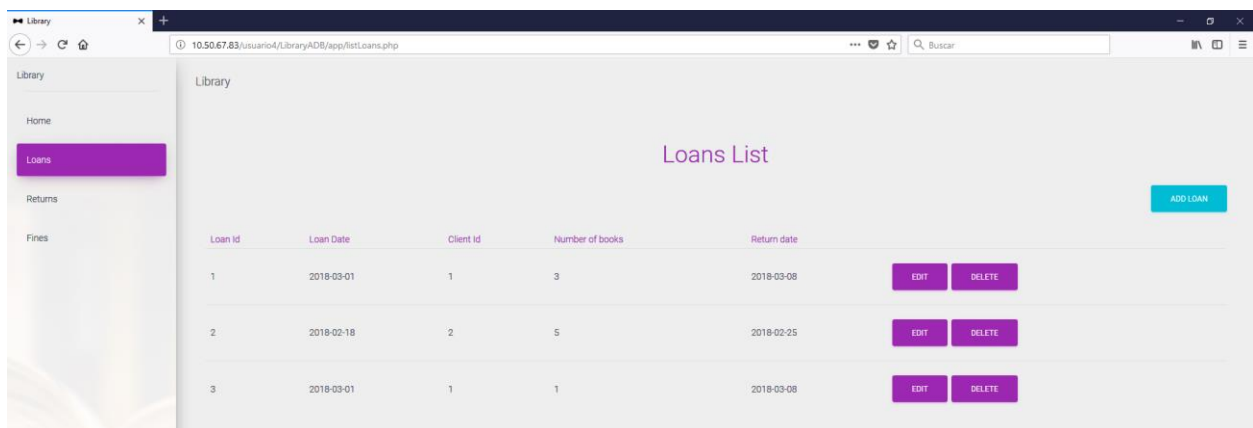
usuario4@servi: ~
usuario4$>         RETURN  finePerBook;
usuario4$>         END;
usuario4$> $finePerBook$ LANGUAGE plpgsql;
CREATE FUNCTION
usuario4=>
usuario4=> CREATE OR REPLACE FUNCTION CalculateTotalFine (lID INT)
usuario4-> RETURNS NUMERIC (10, 2) AS $total$
usuario4$> DECLARE
usuario4$>   total NUMERIC (10, 2);
usuario4$>   originalReturnDate DATE;
usuario4$> BEGIN
usuario4$>   SELECT return_date INTO originalReturnDate
usuario4$>   FROM loans l WHERE l.loanID = lID;
usuario4$>
usuario4$>   SELECT SUM(FinePerBook(bl.bookID, originalReturnDate)) INTO total FROM bo
oks_loans bl WHERE bl.loanID = lID;
usuario4$>
usuario4$>   RETURN total;
usuario4$> END;
usuario4$> $total$ LANGUAGE plpgsql;
CREATE FUNCTION
usuario4=>
usuario4=> CREATE OR REPLACE FUNCTION CreateFine(loID INT)
usuario4->   RETURNS INT AS $fID$
usuario4$>   DECLARE
usuario4$>     fID INT;
usuario4$>     amount NUMERIC(10,2);
usuario4$>   BEGIN
usuario4$>     SELECT CalculateTotalFine(loID) INTO amount;
usuario4$>     INSERT INTO fines (loanID, total_amount) VALUES (loID, amount) RET
URNING fineID INTO fID;
usuario4$>     RETURN fID;
usuario4$>   END;
usuario4$> $fID$ LANGUAGE plpgsql;
CREATE FUNCTION
usuario4=>
usuario4=> CREATE OR REPLACE FUNCTION DeleteLoan(loID INT)
usuario4->   RETURNS VOID AS $$
usuario4$>   BEGIN
usuario4$>     DELETE FROM breturns WHERE loanID = loID;
usuario4$>     DELETE FROM fines WHERE loanID = loID;
usuario4$>     DELETE FROM books_loans WHERE loanID = loID;
usuario4$>     DELETE FROM loans WHERE loanID = loID;
usuario4$>   END;
usuario4$> $$ LANGUAGE plpgsql;
CREATE FUNCTION
usuario4=>

```

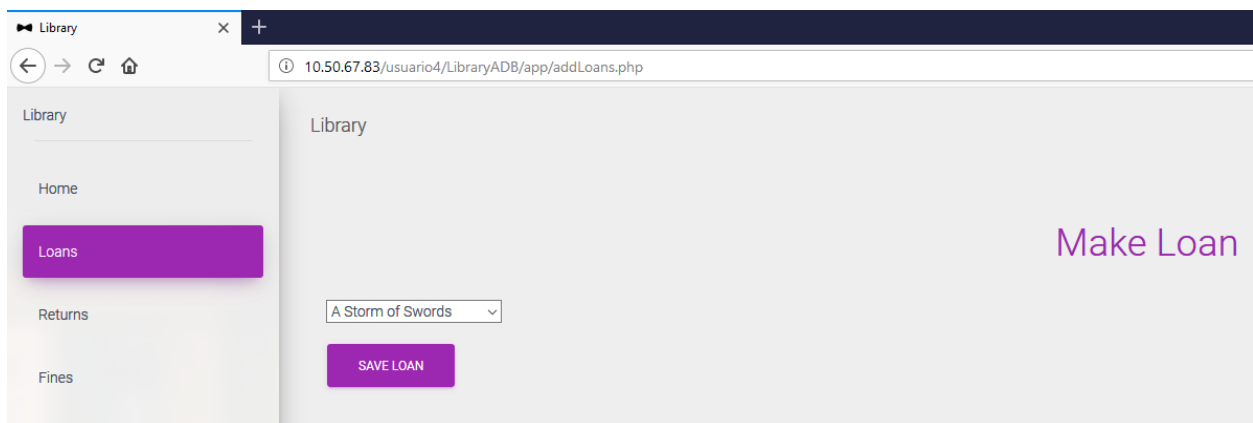
*Creation of required functions, triggers, etc.*



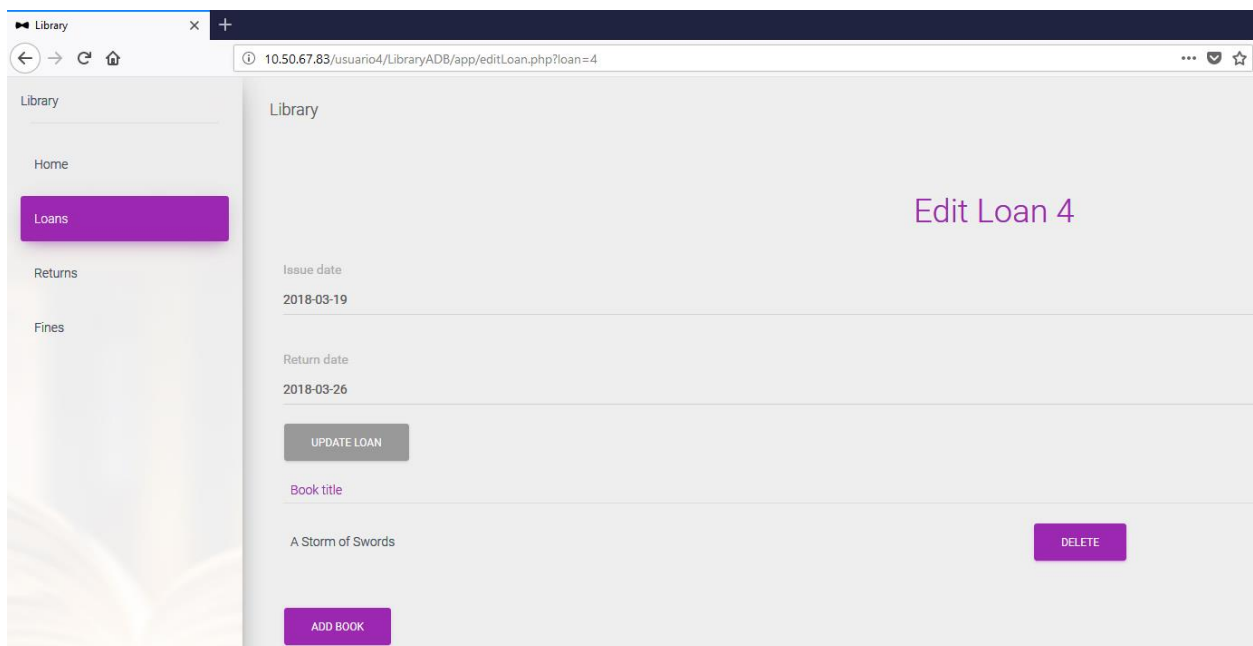
*Homepage for the Library site*



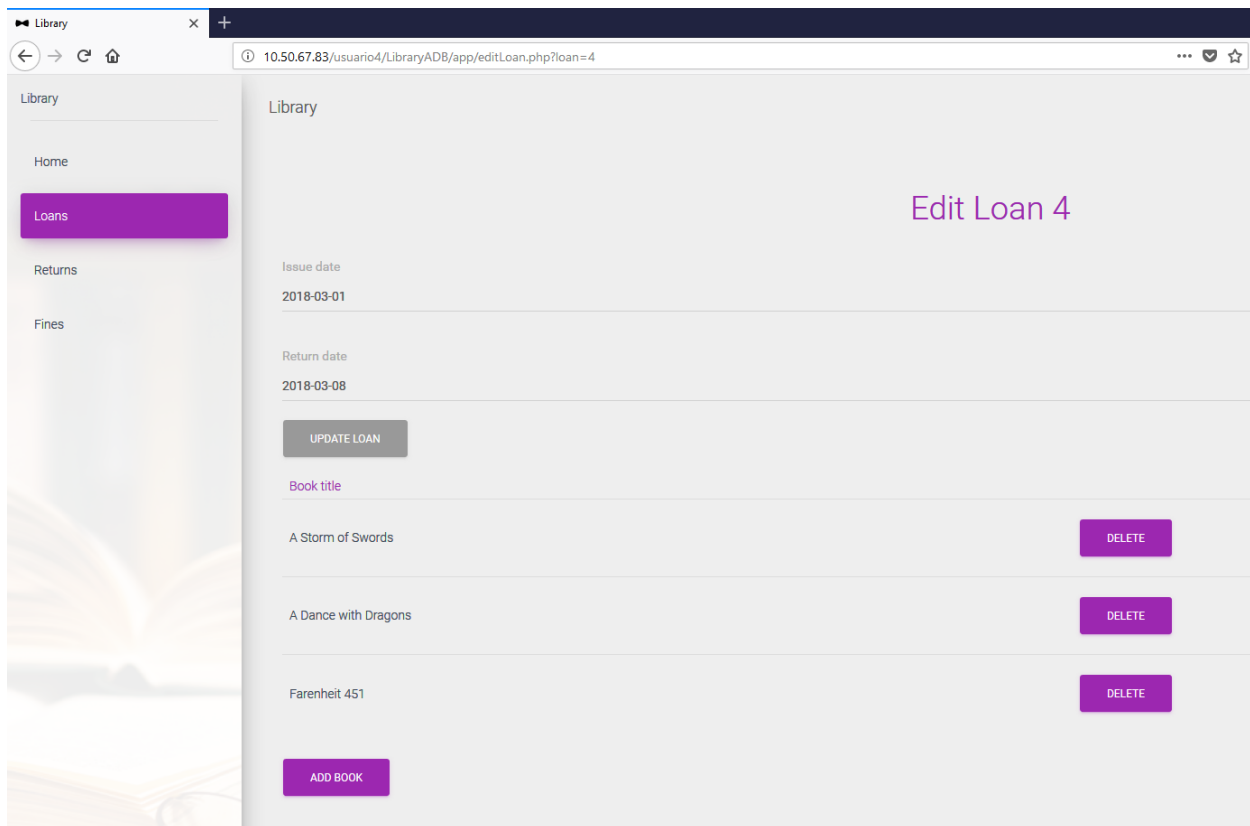
*Listing of all registered loans, from here you can edit or delete any one of them, or add a new one*



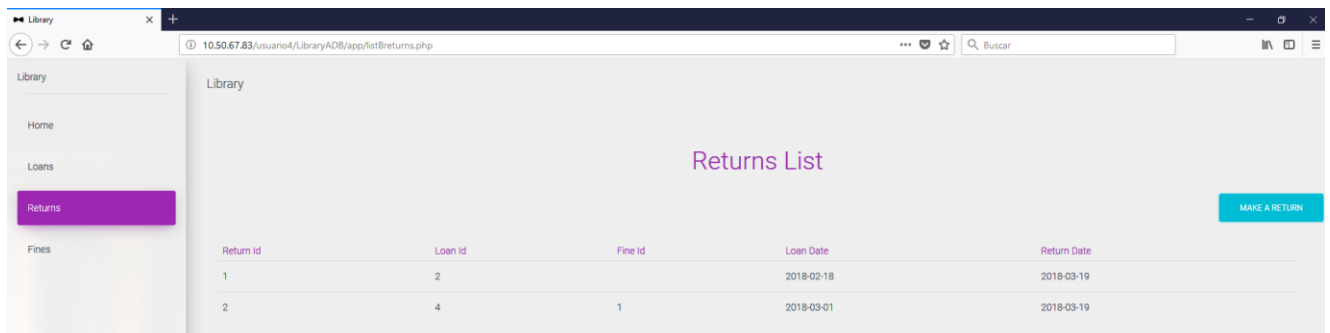
*The new loan screen lets you select 1 book to start off a new loan. Its issue date is the current day, the planned returned date is 7 days later.*



*Edit page for a loan. You are able to change the issue date in the past or future and modify the lending time (by extending the return date) as you see fit, for demonstration purposes.*



*Added a couple books to loan with ID 4, this is the selected way to have one of this entities with more than 1 book lent.*



*Page that displays all returns. A return cannot be deleted or edited.*



Library

Home

Loans

Returns

Fines

### Loans List

Loan Id	Loan Date	Client Id	Return date	
1	2018-03-01	1	2018-03-08	RETURN
3	2018-03-01	1	2018-03-08	RETURN
4	2018-03-01	1	2018-03-08	RETURN

*A list of pending loans is displayed, the returned is inserted automatically, with its date taken from the day it was added.*

Library

Home

Loans

Returns

Fines

### Fines List

Fine Id	Loan Id	Fine amount
1	4	\$1100.00

*If the return inserted has a later return date than that specified by the loan, a fine is added by a trigger. All fines are displayed here.*