**CPSC 2221 - Database Systems (Spring 2018)**

**Group Project - Implementation of a Relational Database**

|  |  |
| --- | --- |
| **Project Title:** | **Langara Book Trade Club** |
| **Project Milestone:** | **Milestone 5a – Complete Project Files** |

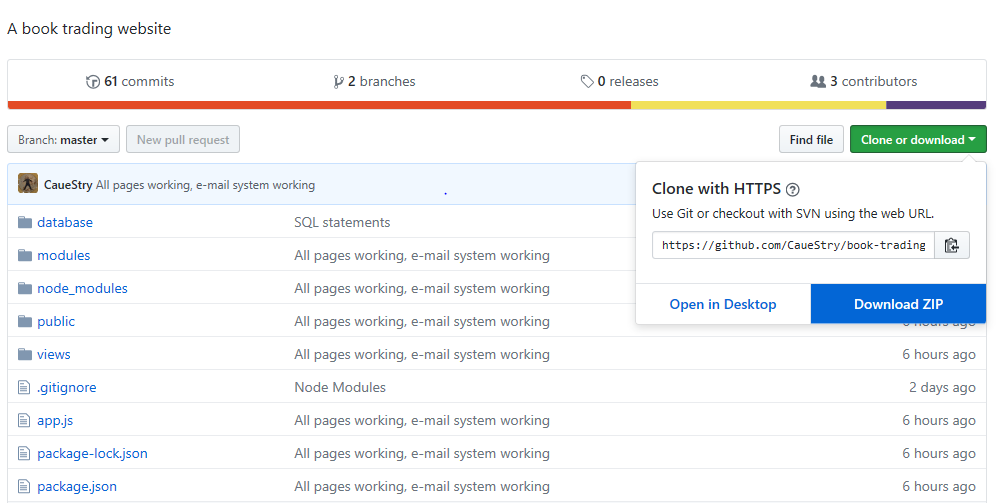
|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Student Name** | **Student ID** | **Email Address** |
| **1** | Elian Figueiredo | 100290512 | elian\_gaspar@hotmail.com |
| **2** | Caue Pinheiro | 100302918 | cauehier@gmail.com |
| **3** | Artyom Lunyakin | 100288803 | artyomluu@gmail.com |

By keying our names and student IDs in the above table, we certify that the work submitted with this cover page was performed solely by those whose names and student IDs are included above.

Also, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Langara College.

**Source code**

The source code can be easily acquired on GitHub, please use the link below:

<https://github.com/CaueStry/book-trading-club>

**How to install your application (step-by-step)**

Once the code has been downloaded and unzipped:

1. Install Node.js
   1. Latest version can be found here: <https://nodejs.org/en/>
2. Make sure you have MySQL installed on localhost, port 3306 (default MySQL port) with the root user properly configured.
3. Create database objects:
   1. Inside repository’s root folder, there’s a folder called “database” as can be seen on image above. All scripts needed to create the database and schema are inside the file “tables.sql”.
      1. There is no need to create the schema, simply execute the commands inside ‘tables.sql’ and the database will be ready to use.  
         To execute the commands on the terminal type:  
         mysql < tables.sql -u root -p   
         [Enter the root password]
4. Now that we have Node.js installed and database ready, we just need to run the server and the application will be ready, here’s how:
   1. Open command prompt, go to the root folder of the project and run the following command: **node app.js**
      1. You’re going to see a message saying “Listening on port 3000...” meaning the server is ready and running.



* 1. Server is running, now we can access the application:
     1. Open your browser and type: <http://localhost:3000>
     2. This will take you to login page, your credentials are: e-mail: ‘jchu24@d2l.langara.bc.ca’, password: ‘123’.
  2. Feel free to explore the application, create more accounts and test the functionalities.
  3. To stop the server, on command line, press ‘Control + C’ or ‘Command + C’ for MacOS.

**Script that could be used to create all tables and data in the database**

The Scripts can be found within the database folder: book-trading-club/database/tables.sql.

**A short description of what the project accomplished**

We managed to implement a fully functional web app, using modern technologies and practices. The Book Trading Club has a secure storage of credentials implementing a Hash and Salt algorithm developed by us to store the password and the salt as a 64byte hash, securing it from brute force attacks using hash lists or database break-ins. We also implemented the entire web server using Node.JS and the Express framework, along with several node modules to back up the functionalities. The database was made using MySQL and normalized to BCNF. The app also implements an e-mail system to keep the users aware of changes on their requests.

Every page loads/writes content from/to the database and is rendered dynamically using a view engine called EJS (Embedded JavaScript).

Every user can see the available featured books, request them, cancel requests, approve requests from other users or reject them. When a request is approved, both users receive an e-mail containing the information to contact their trade partner, allowing them the continue the negotiation.

**A list of the SQL queries used**

**1. [1 Mark] Projection query:**

View the title of all books: SELECT title FROM book;

**2. [1 Mark] Selection query:**

View profile information of current logged user:

SELECT langara\_id, first\_name, last\_name, email FROM sys\_user WHERE langara\_id = (SELECT langara\_id FROM sys\_user WHERE sys\_user.email='elian\_gaspar@hotmail.com');

**3. [1 Mark] Join query:**

View information on books uploaded by current logged user:

SELECT OC.copy\_id AS copy, OC.owner\_langara\_id AS bOwner, OC.book\_price AS bPrice, OC.user\_image\_url AS bUrl, B.title, B.author FROM owned\_copy OC INNER JOIN book B ON OC.book\_id = B.isbn13 WHERE OC.owner\_langara\_id = (SELECT langara\_id FROM sys\_user WHERE sys\_user.email='elian\_gaspar@hotmail.com');

**4. [1 Mark] Division query:**

Select titles for non-requested books owned by a specific user:

SELECT b.title FROM owned\_copy oc INNER JOIN book b ON oc.book\_id = b.isbn13 WHERE oc.owner\_langara\_id = (

SELECT langara\_id FROM sys\_user WHERE sys\_user.email='${email}'

)

AND b.title NOT IN (  
 SELECT b.title FROM owned\_copy oc INNER JOIN book b ON oc.book\_id = b.isbn13 WHERE oc.owner\_langara\_id = (

SELECT langara\_id FROM sys\_user WHERE sys\_user.email='${email}'

)

AND oc.requested\_by\_langara\_id IS NOT NULL

);

**5. [2 Mark] Aggregation query:**

Count how many books have been uploaded:

SELECT COUNT(isbn13) FROM book;

See the average price of all books:

SELECT ROUND(AVG(book\_price), 2) AS AvgAllBooks FROM owned\_copy;

**6. [1 Mark] Nested aggregation with group-by:**

SELECT owner\_langara\_id, COUNT(copy\_id) AS totalBooks, AVG(book\_price) AS avgPrice FROM owned\_copy WHERE owner\_langara\_id = (SELECT langara\_id FROM sys\_user WHERE sys\_user.email='elian\_gaspar@hotmail.com');

GROUP BY owner\_langara\_id;

**7. [2 Marks] Delete operation: Implement a cascade-on-delete situation.**

DELETE FROM sys\_user WHERE email = ‘john.doe@gmail.com’;

By deleting user, all books uploaded by this user will also be deleted as per cascade constrain.

**8. [1 Marks] Update Operation:**

User can select which book they don’t want anymore and cancel the request:

UPDATE owned\_copy SET requested\_by\_langara\_id=NULL WHERE copy\_id=3;

**9. [3 Bonus Marks] Extra features:**

Application has been developed using Bootstrap and EJS.