CS3200-LibraryDB

CS 3200 Final Project by Trang Do, Ren Corr and Morgan Spencer

Building Specifications

Required Programming Languages

* Python (**needs to be at least version 3.10**)
* SQL

Required Libraries

* PyMySQL
* Pandas
* bs4
* requests
* time

Required Software

* MySQL Workbench for reading the SQL codebase
* Visual Studio Code/IntelliJ or similar developing tools for viewing and building the application (Refer to section *Instructions on Building*)
* Terminal/Command Line for running the application

Expected Files

* **libraryDQ.sql** : contains the code for building the library database (backend)
* **main.py** : contains the code for running the application overall (frontend)
* **method.py** : contains the code for the controller of the application along with the supported commands (frontend)
* **data.py** : example of data scraping from the web for book's author and title

Instructions on building

* Build the database *libraryDB.sql* in MySQL Workbench with your personal account
* Change the log in credentials in the file *method.py* to the credentials of the MySQL server which you used to build the database
  + This can be found in the method *connectDB* towards the top of the file, within the assignment of the variable *cnx*
  + We intentionally hidden the server login in the code instead of making it available to the user as in real life, the user would not hook the server up on their own

Current UML Diagram of the Project

A picture containing diagram, text, plan, technical drawing

Description automatically generated

User's Work Flow using this Application

A picture containing diagram, plan, line, yellow

Description automatically generated

Supported Commands for Patron

* **exit()** : exits the application
* **booksAvailable()** : shows all books available for immediate checking out from the library database
* **createHold()** : creates a hold on a currently checked-out book give the book's ID
* **bookCheckout()** : checks out an available book given its ID
* **returnBook()** : returns a currently checked out book given its ID
* **booksBorrowed()** : shows all books which are currently borrowed by the user
* **bookInfo()** : shows all info on the desired book given its ID
* **showComms()** : shows all supported commands in the application
* **createUser()** : sign up as a new user to the application
* **deleteUser()** : remove a user from the application (to be implemented)

Supported Commands for Librarian

* **addBook()** : add a new book to the application given all necessary information regarding the book (to be implemented)
* **removeBook()** : remove a book from the application given the book's ID (to be implemented)

Hidden Supported Commands for User

* **login()** : log into the application given the user's id (patron only) or username and password (librarian only)

Lessons Learned

Technical Expertise Gained

* Coding in Python and SQL while being able to integrate both to create an interactable database program
* We were able to have a more in-depth experience of how front-end and back-end coding ties together in a project, especially through how we delegate our work plus how our code interacts with one another

Insights

* What worked for this project was setting aside a chunk of hours in the day to build the database and frontend
* The group worked in small teams to complete each task and we continually updated and requested more functionality to be added
* This process is what led to a relatively stress free project.

Contemplations/Considerations on the Approach of the Project

* Considering the timeline of our project we could not be as complex as we would have liked but an alternative approach would have been to implement more complex tables and filtering options for finding books

Code not Working in the Project

* All code functions as expected, to our knowledge

Future Work

Current uses:

* Book management: Tracking book availability, managing loans and returns, and getting book information.
* Patron management: Storing patron information, managing holds, and tracking overdue fees.
* Librarian managment: Adding and removing books, checking availabe books.

Future uses:

* Recommendation system: Implementing a recommendation engine based on patron preferences and borrowing history to suggest relevant books.
* Interlibrary loan management: Facilitating the borrowing and lending of books between different library branches or even external libraries.
* Integration with external systems: Integrating the library database with online catalog systems, self-checkout kiosks, or mobile applications.

Planned Uses of the Database

* Example work to demonstrate understanding and application of Python and SQL

Potential Future Functionalities

* Working GUI for a more visual experience
* Functional data scrapping to populate the database with more meaningful data in a more effective way
* Sample reviews online can be collected via online queries through the application