Library Kiosk

By- Charan Kamal Singh

Library Kiosk

- What if there was a Kiosk setup in our library instead of a librarian? A simple machine with a dedicated software for our own college.
- This project is made by keeping in mind all the necessary functions and the duties of a human librarian in the library.
- It has been made with the purpose of automating the tasks in library and making them more efficient, faster and leaving lesser possibilities for errors.

Key features of this project:

- This project follows an Object Oriented Approach with the use of classes, multi level inheritance, polymorphism and restricted access and flow of data.
- The users are Students, Teacher and an admin. All the functions have been made by keeping in mind the different actions of these users in the library.
- All the data is stored and accessed through files.
- The project is personalized to our college. There is a roll number check during the registration process that follows the format of our college roll numbers.

Key features of this project:

- There is a limit to the number of books a user can issue at one time. For students its three and for teachers it is five.
- Use of time functionality
  - During the issue of book, the day of the year is also stored so that the computer can keep a check on how many days its been since the book has been issued.
  - The time limit for book return is 10 days.
- Only admin can approve late book returns.
- For late book return, there is a fine calculator in the program.
- Every time someone issues a new book, a new entry is made in a csv file that stores the data of all books issued. This csv file can only be accessed by the admin.

Key features of this project:

- Only an admin can add a new admin. There is a default admin in the program with the username - 123456 and password - 123456.
- The admin has a username of 6 digits unlike the students and teachers.
- When you run the program, it creates directories in the 'C' disk drive in the computer.
- These directories include:
  - Library Kiosk(*main directory*)
    - bookdata *(For storing all information of the books)*
    - adminData *(admin information)*
    - studentData *(student information)*
    - teacherData(*teacher information)*

Key features of this project:

- All operations in this program are handled on files. There are dedicated directories for different entities and those directories contain the files that have all the data in them.
- Functions such as LOGIN and REGISTER are overloaded in all the three classes and are called using static allocation as a student class object is required only to call its function and not from the base class.
- The data of all users is stored in their individual directories thus making data management more organised and efficient.

Key features of this project:

- Click the below links to jump to slides:
  - More possibilities for this project
  - Code Structure
  - How it works?
  - Function deconstruction
  - Student
  - Teacher
  - Admin
  - Database Entry
  - Credits

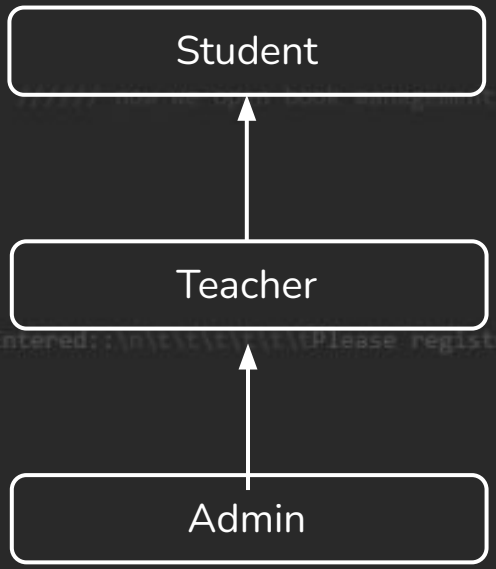
More possibilities for this project

- We can add an opt verification system through email. This would work as a check for validity of the user.
- Instead of entering stock position, we can add a qr scanner in the kiosk to scan the qr code of the book.
- The kiosk can have a camera that identifies face using Computer Vision.
- Because every college personnel has been issued a college id, thus sending an email on that id would confirm the validity of the user.
- The admin can have more functionalities such as viewing all database, checking which users are beyond due date etc.
- The other users could have an option to view all the books in the database.

Code Structure

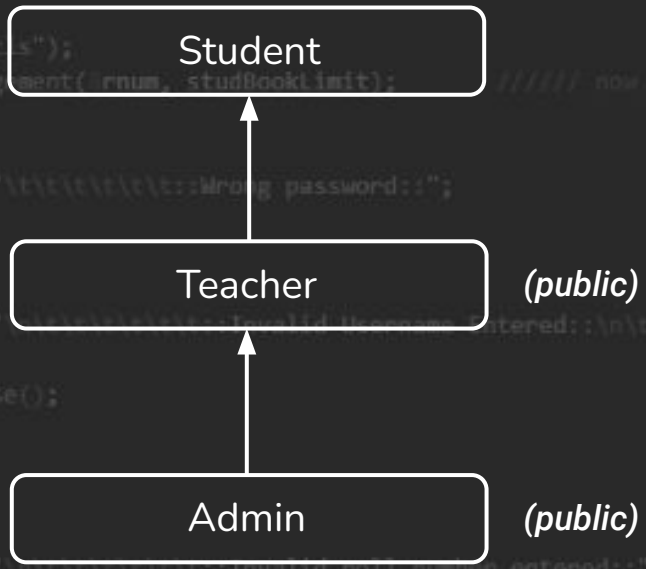
- It has three main classes, Student, Teacher and the Admin, which represent the different entities that will be interacting with the program.
- The three classes exhibit multilevel inheritance, with student being the base class, teacher class inherits from student class and admin class inherits from teacher class.

Code Structure



Hence the admin has access to all data and methods in the program

Code Structure



Book

This book class is defined in the header file “bookManagement.h”

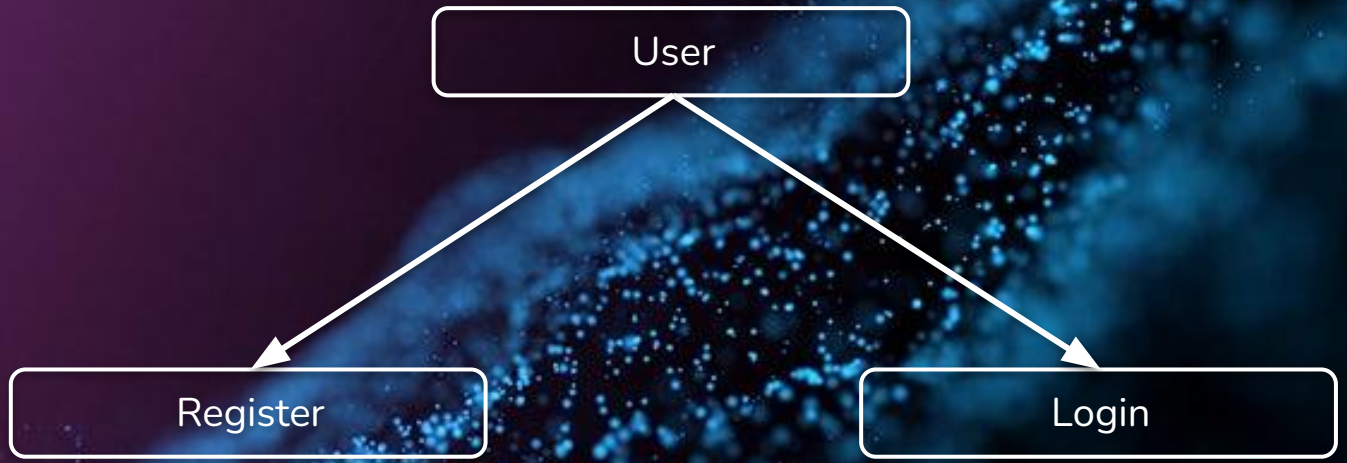
Code Structure

- Thus our admin can access all public/private attributes and methods of both student and teacher class and vice versa is not possible.
- The program makes directories in C drive of the computer and stores all data in the form of files.
- Also this program works with a custom header file "bookManagement.h" which contains all the operations that we can perform on books and the database.

How it works?

- When you run the program, it creates directories in the ‘C’ disk drive in the computer.
- These directories include:
  - Library Kiosk(*main directory*)
    - bookdata *(For storing all information of the books)*
    - adminData *(admin information)*
    - studentData *(student information)*
    - teacherData(*teacher information)*

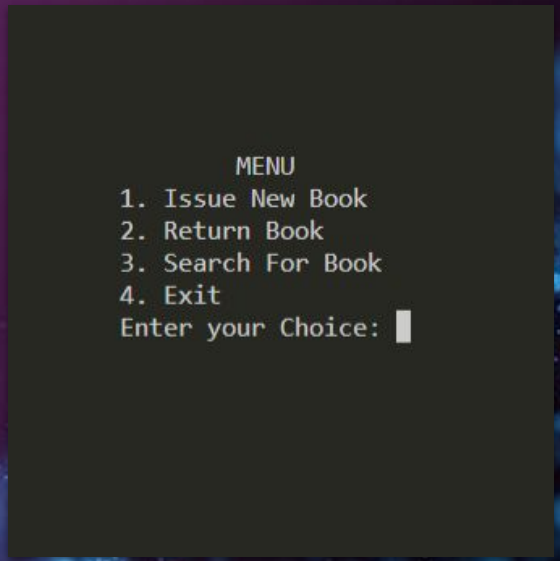
How it works?



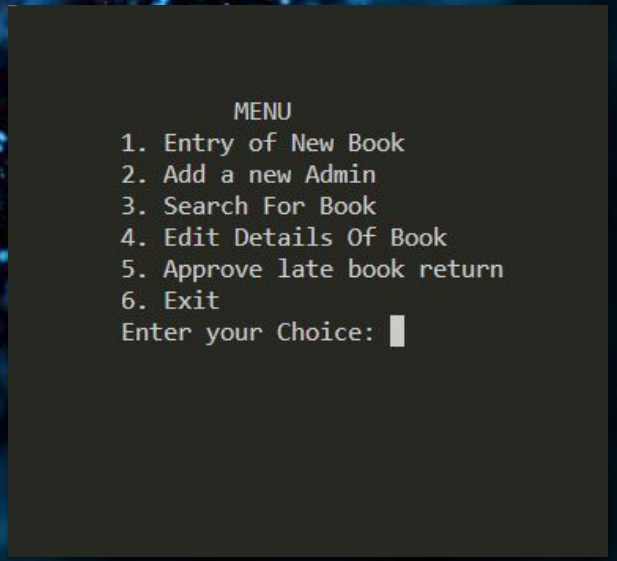
The User get the option to login with their credentials or register if they are a new member.

After the user is logged in, different users get a different set of functionalities.

Student/Teacher

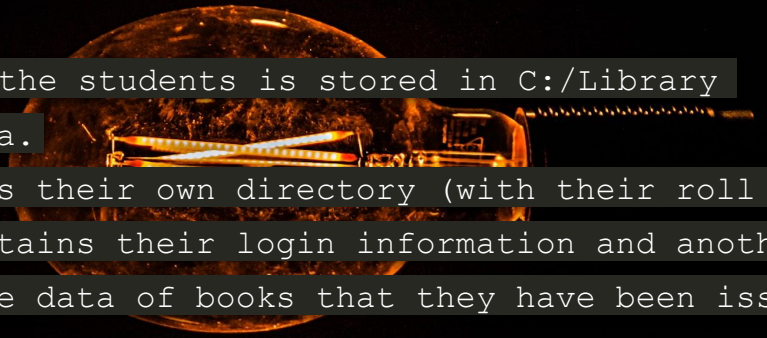


Admin



- Functions such as LOGIN and REGISTER are overloaded in all the three classes and are called using static allocation as a student class object is required only to call its function and not from the base class.
- All operations in this program are handled on files. There are dedicated directories for different entities and those directories contain the files that have all the data in them.

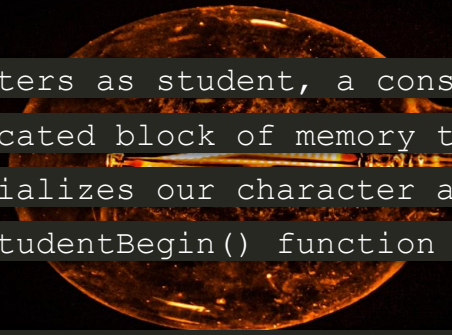
Student



- All the data of the students is stored in C:/Library Kiosk/studentData.
- Every student has their own directory (with their roll number as name), which contains their login information and another file that contains the data of books that they have been issued.

(This program has been designed in a way that all the files and the data is highly organised and is easy to understand by any other programmer.)

Student

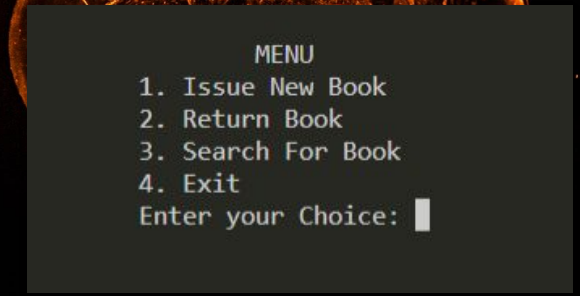


- When the user enters as student, a constructor runs that dynamically allocated block of memory to name attribute in the class. Thus initializes our character array.
- After that the studentBegin() function is called using the student object.
- This constructor only initializes the attributes, after that function is called that takes the program further.

(The structure of the program follows an object oriented approach. The whole flow of the program is controlled by class attributes and functions. We only call the studentBegin() function and the rest of the program runs itself, thus its structure has a bottom up approach)

Student

- The user is asked to login and register after which, a task menu appears that gives all the functionalities available to the student.



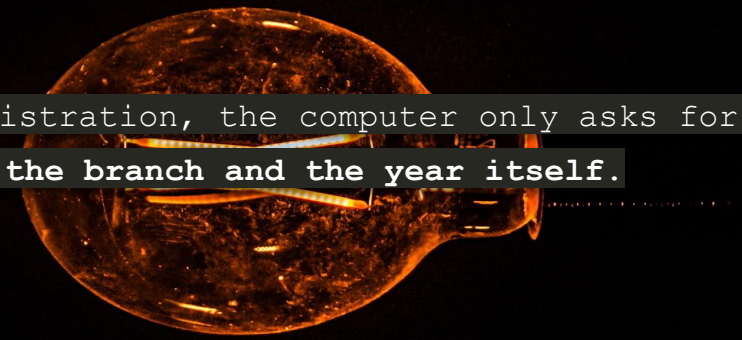
(The login credentials are stored inside a file with the name as their roll number with the address C:/Library Kiosk/studentData/thestudentsrollnumber/rollnumber.txt)

Roll Number Check

- During registration the computer checks the validity of the roll number by checking the format of our college.
  - The roll Number is five digit number
  - The third digit of the roll number signifies our branch. Thus it can only be between 1 to 4.
  - The first two digits are for year, thus it cannot be more than 21.

(This roll number check runs every time during registration so that false users can be eliminated and only college personnel can use the library kiosk)

Roll Number Check



- Thus during registration, the computer only asks for roll number and calculates **the branch and the year itself.**

(This roll number check runs every time during registration so that false users can be eliminated and only college personnel can use the library kiosk, thus making it a more personalized software for our college)

New Book Issue

- This function is defined in the book class in “bookManagement” header file.
- This function is used commonly by the teacher and student objects. Thus promotes reusability of code and makes our work easier and the code efficient
- At one time a teacher can issue only 5 books and for students the limit is 3 books.

(This function creates a file in the database of the user and in that database it inputs the book data)

New Book Issue

- All the operations in this function occur in files.
- The function takes the id and book limit as arguments.
- It checks how many books have been issued to the user, it only issues more books if the user is under the book limit.
- If the user hasn’t been issued any books yet, a new file is created in the student database.
- It also checks if the user is trying to issue the same book again.

(The function creates a file in the database of the user with the name “book + id.txt”)

Introduction of time functionality

- In the file, stock position and the day of the year is added.
- This done using <ctime.h> library.
- The time limit for returning the books is set to 10 days. After that the user is fined accordingly.

(In the file, stock <space> dayOfYear is inputted and every other entry is done in the next line)

Book Return

- For book return, the first thing that is done is checking if the requested book is in the database of the user.
- Second, it checks whether the user is late for returning the book. If yes only admin can approve the book return after due fine is paid.
- If the user is not late, the book is deleted from the user's database.

(The function counts how many days the user is late and using that data it calculates the fine)

Book Return

HOW IT WORKS?

- The system creates a temporary file, and then reads from the main file and writes in the temporary file.
- All entries are read from the main file except for the one requested to be deleted.
- Then the system reads all data from temporary file and enters in the main file.

(The program counts which line is not to be read and that line number is used during book return .The same function is used for both teachers and students as it is defined in the header file)

Book Return

- If the user is late for returning the book, Only the admin can approve the book return.
- Thus it ensures that the fine has been paid.
- An internal function “calculateFine()” automatically calculates the fine based on how many days the user is late.

(The day of issue is read from the file, and the current day is calculated using the ctime.h library, if the difference is greater than 10 days then the student is late)

Search for book

- The stock position of the book is used as a key.
- All the book data is stored in files in the C:/Library Kiosk/bookdata directory.
- The name of each book file is its Stock position.
- Thus, the user enters the stock position of the file, and the system opens that file and prints the data.

(If no such file exists then that means that there is no such book in the database.)

Teacher Class

- All the functionality for teachers and students is same.
- The functions used for login and registration are overloaded functions and the rest of the functions are defined in the header file for common use.
- The systems tells the difference between student and teacher object by the book limit argument given in all functions.

(The directory for the teachers is different, but the format of storing all data is same. Teachers data is saved in C:/Library Kiosk/teacherData directory)

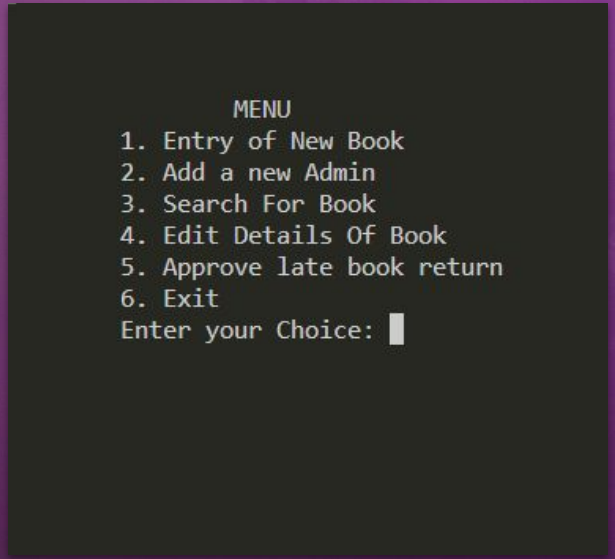
Admin Class

- The admin has the access to all data.
- Only an admin can register other admins.
- The admin has a 6 digit id unlike the student and teachers.
- Admin has the control over all data and can choose to remove users also.

(The directory for the admin is different, but the format of storing all data is same. Teachers data is saved in C:/Library Kiosk/adminData directory)

Admin Class

- Given below are the functions admin is capable of:



Admin Functionalities

- The admin can add new books in the database. The function just creates a file in the bookdata directory with all the information of the book.
- Only the admin can register other admins, the register function is again overloaded from the base classes.
- The admin can also edit existing book data, by just overriding in the existing book file.
- The admin can also approve late book return. This assures that fine has been paid by the user.

Database Entry

One essential function of the Admin is,

- It automatically creates a csv database file and enters the details of every book issued.
- Whenever a book is issued by any user, a function called databaseEntry() is called that creates or appends to a csv file.
- The file has data like,
  - stock position of the book,
  - Roll Number/ID of the user
  - User identity(student or teacher)
  - Time of book issue

(This creates a single database where all data is stored in one place about books issued and returned)

Credits:
Name- Charan Kamal Singh
Roll Number - CO21314
Branch- CSE

To access resources:
https://drive.google.com/drive/folders/1Ms66heBdJ3HcA9ilXsf75xU41X5otChT?usp=sharing

You can also scan this QR: