LIBRARY MANAGEMENT SYSTEM IN C ++

Data extraction and processing from the CSV
CST 2550 Coursework 1
by Suba Senthilnathan
M00863517

An overview

- **The Library Management System**
- This application functions as a library management system, arranging data about books and library users. The system creates a library database by reading data from a CSV file, giving users effective management and access to knowledge.
- **The command line input:**
  - When entering data, the programme demands a CSV file as a command line parameter.
  - **Data processing:**
  - It scans the CSV file, interpreting each line to retrieve member or book info while ignoring the header line.
  - **The library database:**
  - A thorough library database is created by classifying members and books separately.
- **Display:**
  - The application shows a list of all the books and users that are currently in the library after processing the file.
- **Easy to use**
  - The system's goal is to offer a simple user experience for effective library data management.

The Overview of Classes

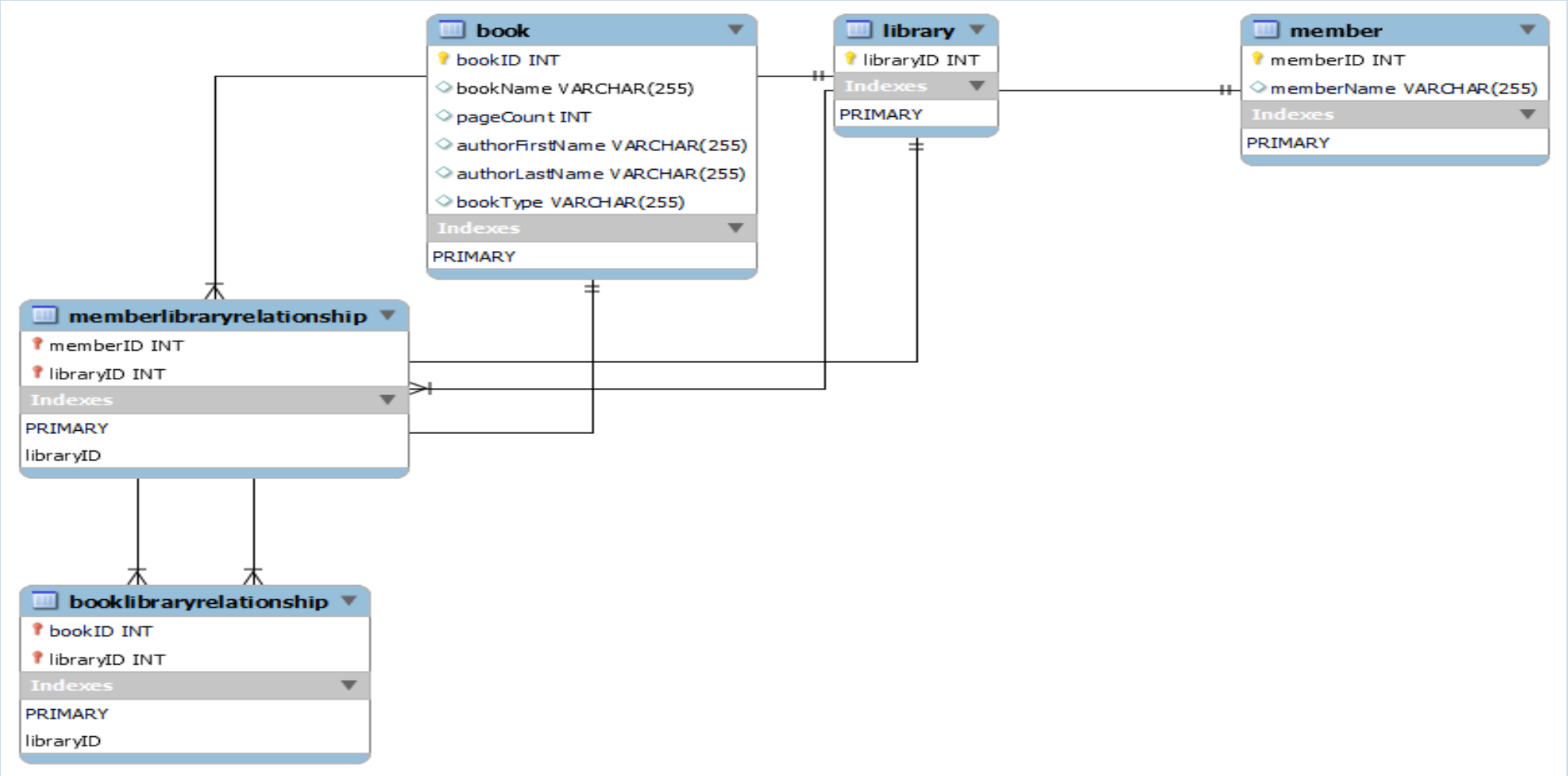
- **Member Class:**
  - Represents a library member with unique **memberID** and **memberName**.
  - Created using a constructor that accepts a name and an ID.
  - **Book Class:**
  - Depicts a book with an individual **bookID**, **bookName**, **pageCount**, **authorFirstName**, **authorLastName**, and **bookType**.
  - Constructed with a constructor that accepts these parameters.
  - **Library Class:**
  - Uses private vectors for handling the library's books and users.
  - Offers ways to add members (addMember) and books (addBook).
  - Shows all members (displayMembers) and books (displayBooks).
  - **Interactions:**
  - The appropriate add methods are used to add books and individuals to the library.
  - Present methods can be used to display material from libraries.
  - **Relationships:**
  - Members from the Member and Book classes are combined into the Library class.
  - Facilitates the effective arrangement and retrieval of library information.

Class Information - Member

- **Overview of the Member Class:**
- Identifies a specific member of the library.
    - **Features:**
    - Individual memberID; memberName represents the member's name.

    The attributes:
- **memberID: A number that is given to every member.**
- **memberName: The member's name at the library.**
- **Constructor:**
    - Starting **memberID** and **memberName** during object creation.
- **Purpose:**
    - Effectively storing and obtaining member data within the library system.
    - **Application:**
    - Library members are represented by objects of the Member class.
    - Allows the library to manage member data in an organised way.
    - **Relations:** The Library class oversees the whole library system, which includes examples of the Member class.

UML Diagram


book
bookID INT
bookName VARCHAR(255)
pageCount INT
authorFirstName VARCHAR(255)
authorLastName VARCHAR(255)
bookType VARCHAR(255)
Indexes
PRIMARY
library
libraryID INT
Indexes
PRIMARY
member
memberID INT
memberName VARCHAR(255)
Indexes
PRIMARY
memberlibraryrelationship
memberID INT
libraryID INT
Indexes
PRIMARY
libraryID
booklibraryrelationship
bookID INT
libraryID INT
Indexes
PRIMARY
libraryID

The details of the Class: Book

- **Book Class Overview:**
  - **Shows a library book with features.**
  - **bookID, bookName, authorFirstName, authorLastName, bookType, and pageCount**.
  - Allows for the structured storing and retrieval of book data.
  - **Skills:**
  - **bookID: a particular classification.**
  - **bookName: The book's title.**
  - **pageCount: Pages in the document.**
  - **First and last names of the authors.**
  - **bookType: Kind or category.**
  - **The purpose:**
  - Thorough management of the book data in the library.
  - **The usage:**
  - Objects of the **Book** class represent individual books in the library.

Description of the Class: Library

- **Library Class Overview:**
  - Uses two private vectors to administer the library system.
  - Books to keep book-related items.
  - Members to keep member objects in storage.
  - **Techniques:**
  - addBook: Enables the library to hold a book.
  - addMember: Enables a user to join the library.
  - displayBooks: Shows every book in the collection.
  - displayMembers: Shows the library's entire membership.
  - **The purpose:**
  - Effective arrangement and retrieving of library information.
  - **The usage:**
  - The library class makes it easier to add and show books and members.

Basic Purpose

- **Overview of the primary function:**
  - Acts as the program's entrance point.
  - Verifies that the command-line options (filename) are accurate.
  - For storing data, a Library class instance is created.
  - **Managing handlings:**
  - Opens the CSV file that the command line specifies.
  - Verify that the file opened successfully.
  - **Knowledge Handling:**
  - Reads and manipulates information line by line from the CSV file.
  - Utilising a comma as the delimiter, each line is tokenized.
- **Item Production:**
- Determines the number of tokens to create Member or Book objects.
- **Error Solution:**
  - Responds to unpredictable situations and shows the relevant error messages.

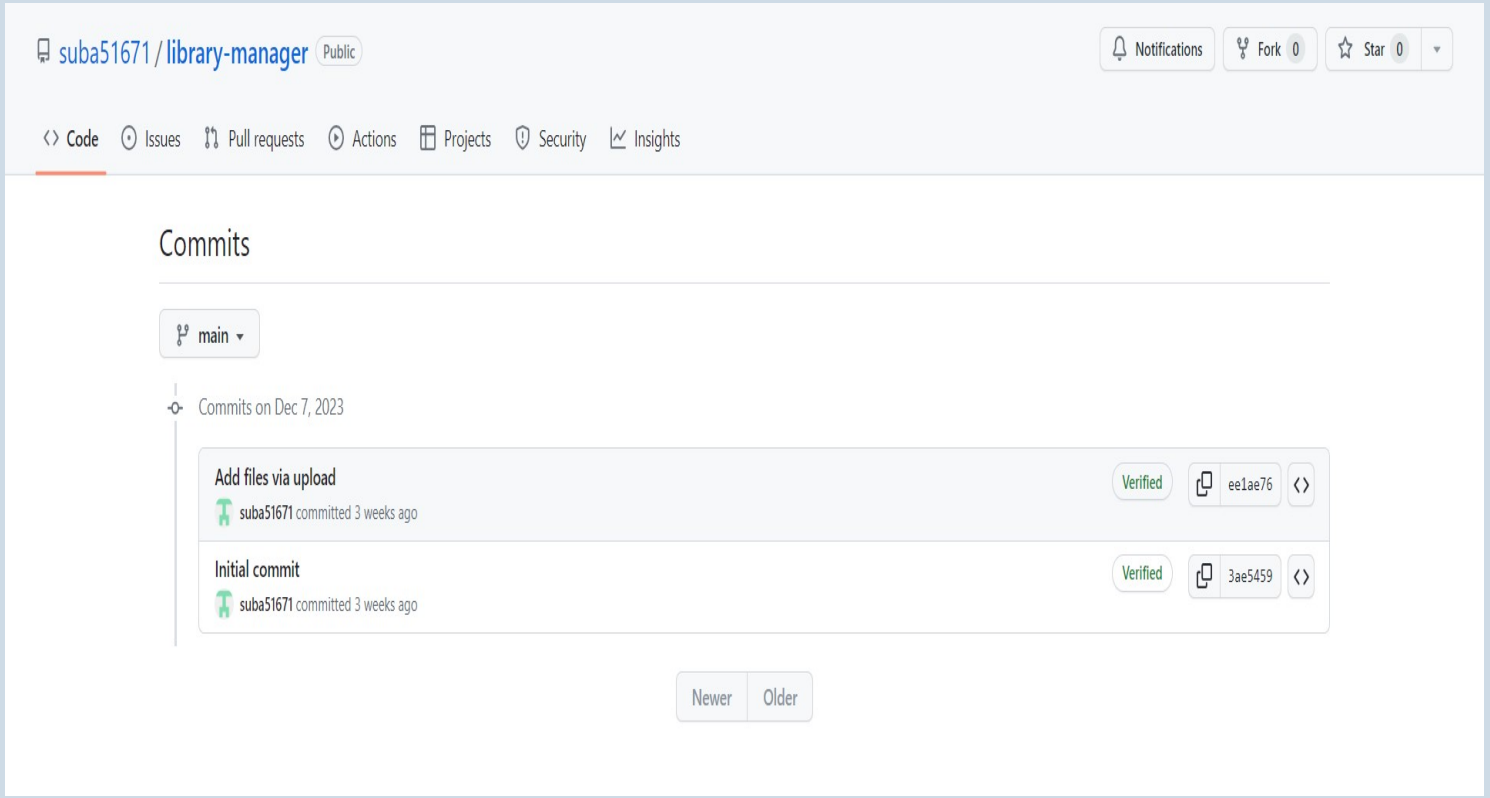
Reading CSV Data

- **Processing CSV Data:**
- Use the opened CSV file to read and analyse data.
    - Ignore the column names in the header line.
- **Tokenization:**
    - Utilising a comma as the delimiter, each line is tokenized.
    - Make use of **std::istringstream** and **std::getline** to extract tokens.
- **Contingent logic counts the tokens in a line to determine if it represents a book or a member.**
- **Dynamic Object Creation:**
- Generates Member or Book objects based on the context.
- addMember or addBook to add the generated objects to the library.
- **Error management**
- Gives unusual line format error messages.

Displaying the Library Information

- **Displaying Books and Members:**
  - The application shows the library's contents after processing the CSV data.
  - **Library Content Overview:**
  - Shows information about every book by using the displayBooks technique.
  - Make use of the **displayMembers** method to showcase information about all members.
- **Example of Output:**
- Gives a sample output to show how member and book details are shown.
  - **User Interaction:**
  - Explains how to see the contents of the library system through user or developer interaction.
  - **Conclusion:**
  - Ends the main programme, demonstrating how the data from the CSV file was successfully read, processed, and displayed.

Implementation



- **Technique:** Creating code based on the specs and design is the first step in converting the design into functional software. It entails comprehending the instructions, decomposing them into tasks, and eventually coding each one.
- **The Makefile**: The Makefile is utilised to manage the software's building process. It lists the actions and variables needed to connect and build the programme.
- **Version Control:** GitHub, and other version control systems are necessary for managing shifts coordinating with others and guaranteeing that the source code remains in a functional condition. Commit messages include an outline of modifications and an explanation of their causes.

Testing approach

- **Unit testing** – I carried out unit testing to make sure every part of the library management system worked and was right, to ensure that every part and function produces the intended result for a given input, this required testing each one separately. Test cases, which included boundary conditions and incorrect inputs, were created to cover a variety of situations and edge cases.
- **Integration testing** – I examined the integration and interconnection of the various library management system modules once each of their components had been tested, to make that the modules function as a cohesive one, this required evaluating the data flow and communication between them. Test cases were created to cover various integration situations and spot any problems or contradictions.
- **System testing** – I carried out system testing to assess the library management system's general operation and performance. This entailed verifying that the system satisfies the criteria by evaluating the system, which includes every module and how they interact. Test cases were created to simulate various user scenarios and encourage system usage in the actual world.

Conclusion

- **Features of the System:**
- The C++ program reads and processes the data from a CSV file to manage library data.
- Make use of the **Member**, **Book**, and **Library** classes, the program ensures organised and modular data storage.

The user interaction:

- Browsing the library items is made simple for users by techniques like displayBooks and displayMembers.
- The application offers enlightening error messages, improving system resilience and user experience.

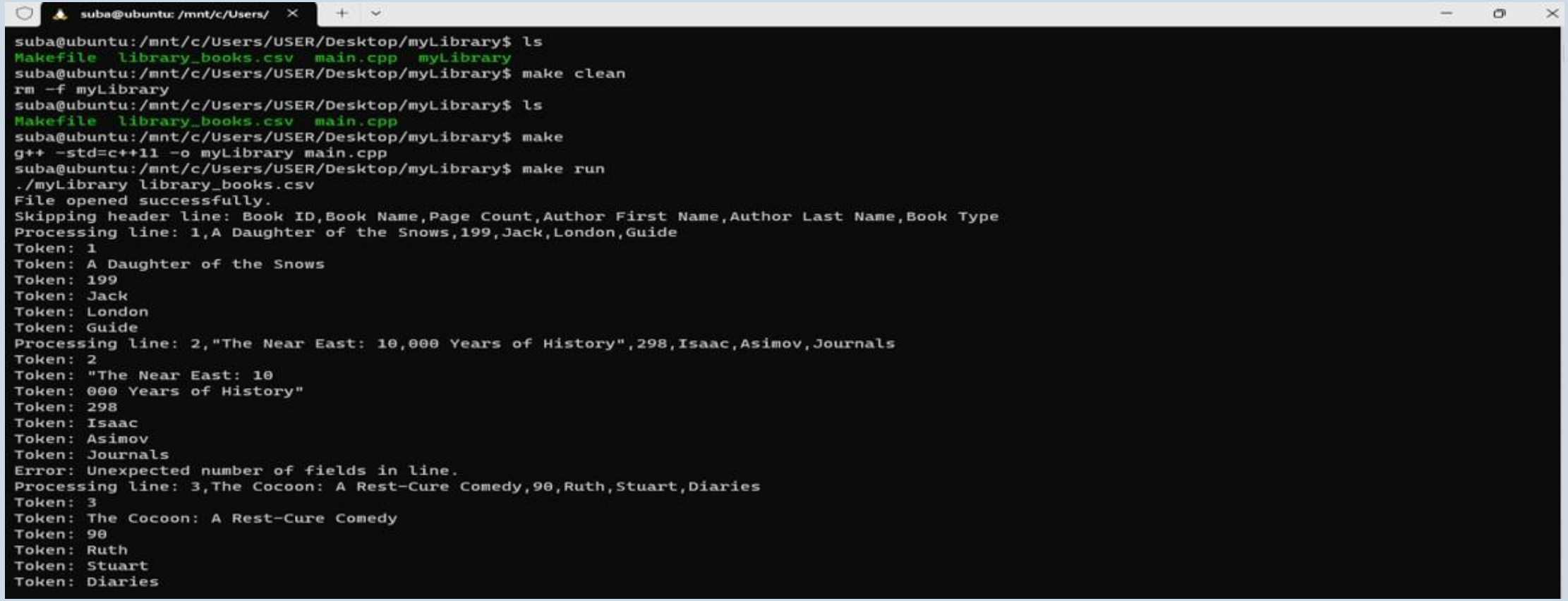
Adaptability and Flexibility:

- The application dynamically manages different CSV line structures, demonstrating object generation versatility.
- Future improvements are made simple by the modular architecture, including the inclusion of new functionality and support for various file types.

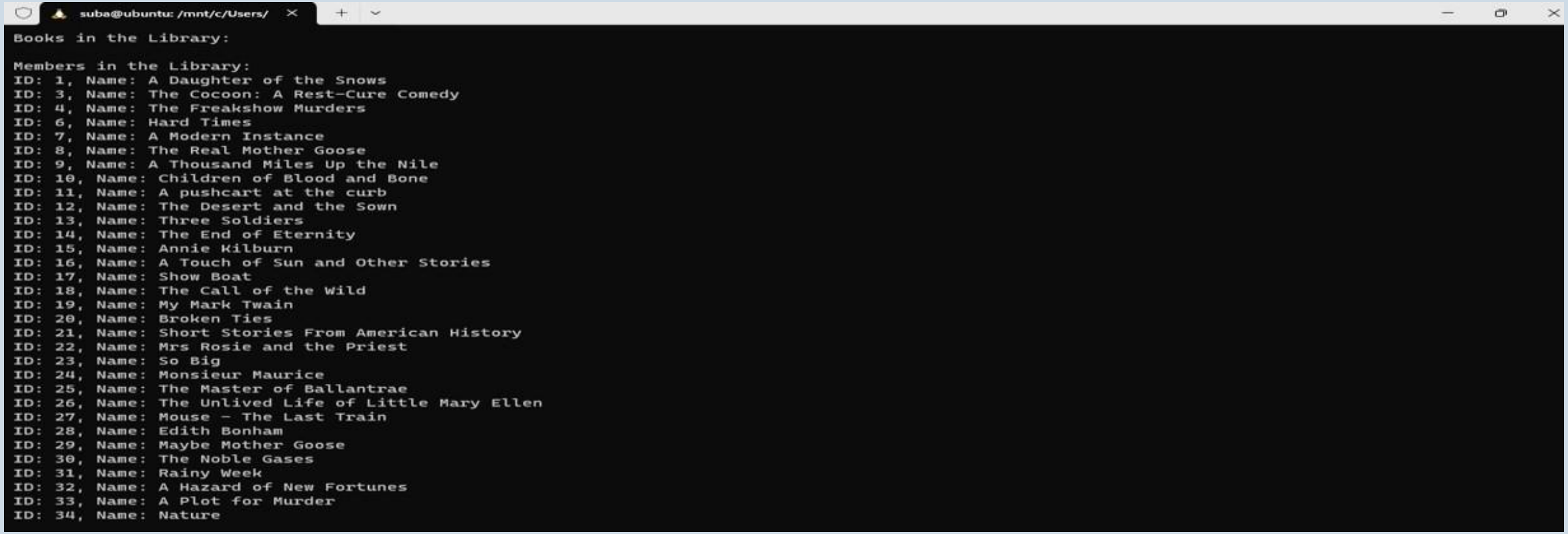
Relevance and Effect:

- Beyond library management, the concepts covered in this programme also shed light on efficient file dealing with, processing of information, and class-based structure in C++.

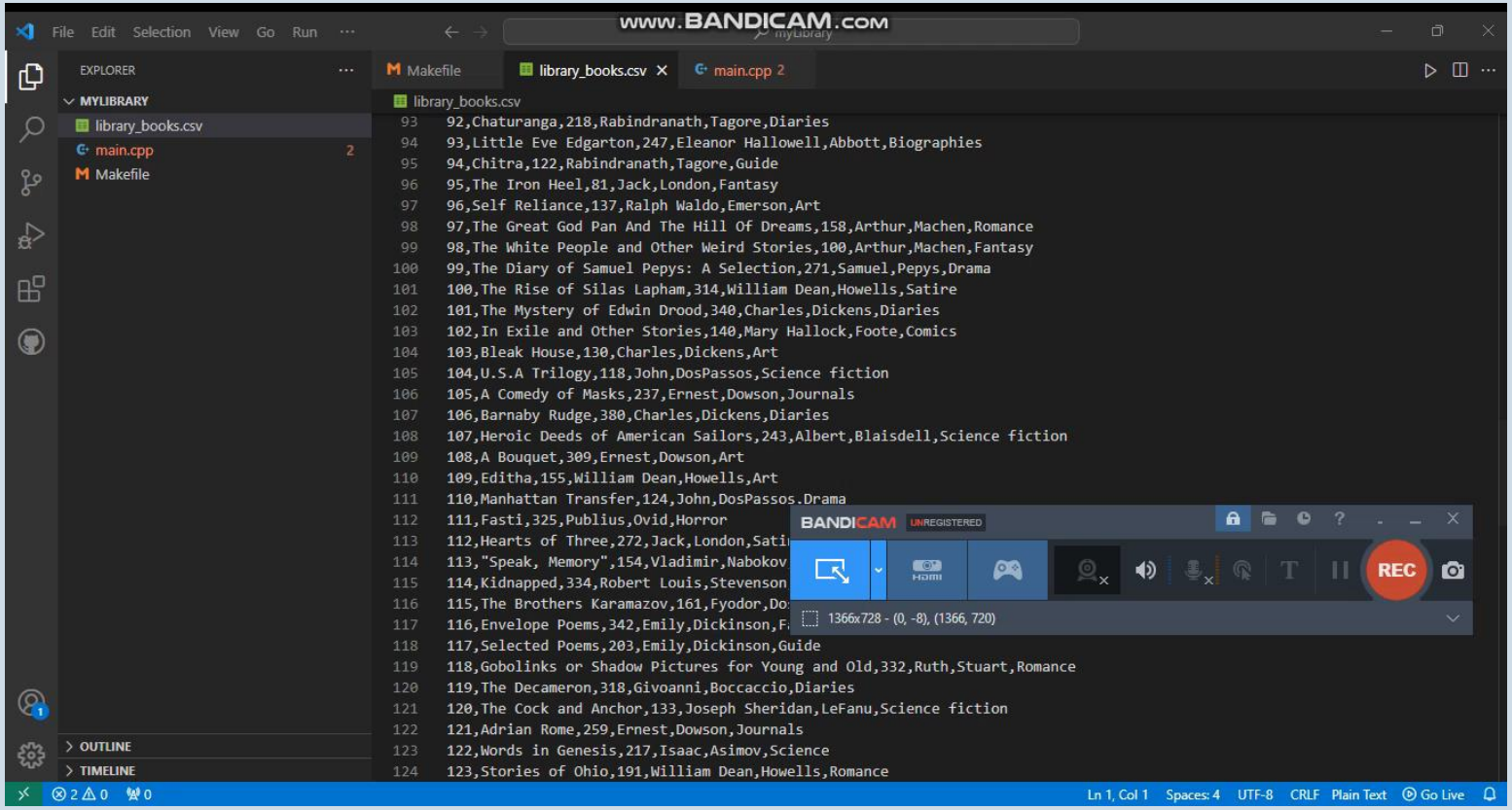
Outputs


suba@ubuntu: /mnt/c/Users/
suba@ubuntu:/mnt/c/Users/USER/Desktop/myLibrary$ ls
Makefile library_books.csv main.cpp myLibrary
suba@ubuntu:/mnt/c/Users/USER/Desktop/myLibrary$ make clean
rm -f myLibrary
suba@ubuntu:/mnt/c/Users/USER/Desktop/myLibrary$ ls
Makefile library_books.csv main.cpp
suba@ubuntu:/mnt/c/Users/USER/Desktop/myLibrary$ make
g++ -std=c++11 -o myLibrary main.cpp
suba@ubuntu:/mnt/c/Users/USER/Desktop/myLibrary$ make run
./myLibrary library_books.csv
File opened successfully.
Skipping header line: Book ID,Book Name,Page Count,Author First Name,Author Last Name,Book Type
Processing line: 1,A Daughter of the Snows,199,Jack,London,Guide
Token: 1
Token: A Daughter of the Snows
Token: 199
Token: Jack
Token: London
Token: Guide
Processing line: 2,"The Near East: 10,000 Years of History",298,Isaac,Asimov,Journals
Token: 2
Token: "The Near East: 10
Token: 000 Years of History"
Token: 298
Token: Isaac
Token: Asimov
Token: Journals
Error: Unexpected number of fields in line.
Processing line: 3,The Cocoon: A Rest-Cure Comedy,90,Ruth,Stuart,Diaries
Token: 3
Token: The Cocoon: A Rest-Cure Comedy
Token: 90
Token: Ruth
Token: Stuart
Token: Diaries

Outputs


suba@ubuntu: /mnt/c/Users/
Books in the Library:
Members in the Library:
ID: 1, Name: A Daughter of the Snows
ID: 3, Name: The Cocoon: A Rest-Cure Comedy
ID: 4, Name: The Freakshow Murders
ID: 6, Name: Hard Times
ID: 7, Name: A Modern Instance
ID: 8, Name: The Real Mother Goose
ID: 9, Name: A Thousand Miles Up the Nile
ID: 10, Name: Children of Blood and Bone
ID: 11, Name: A pushcart at the curb
ID: 12, Name: The Desert and the Sown
ID: 13, Name: Three Soldiers
ID: 14, Name: The End of Eternity
ID: 15, Name: Annie Kilburn
ID: 16, Name: A Touch of Sun and Other Stories
ID: 17, Name: Show Boat
ID: 18, Name: The Call of the Wild
ID: 19, Name: My Mark Twain
ID: 20, Name: Broken Ties
ID: 21, Name: Short Stories From American History
ID: 22, Name: Mrs Rosie and the Priest
ID: 23, Name: So Big
ID: 24, Name: Monsieur Maurice
ID: 25, Name: The Master of Ballantrae
ID: 26, Name: The Unlived Life of Little Mary Ellen
ID: 27, Name: Mouse - The Last Train
ID: 28, Name: Edith Bonham
ID: 29, Name: Maybe Mother Goose
ID: 30, Name: The Noble Gases
ID: 31, Name: Rainy Week
ID: 32, Name: A Hazard of New Fortunes
ID: 33, Name: A Plot for Murder
ID: 34, Name: Nature

Video demo


www.BANDICAM.com
File Edit Selection View Go Run
EXPLORER
MYLIBRARY
library_books.csv
main.cpp 2
Makefile
OUTLINE
TIMELINE
Makefile
library_books.csv
main.cpp 2
library_books.csv
92,Chaturanga,218,Rabindranath,Tagore,Diaries
93,Little Eve Edgarton,247,Eleanor Hallowell,Abbott,Biographies
94,Chitra,122,Rabindranath,Tagore,Guide
95,The Iron Heel,81,Jack,London,Fantasy
96,Self Reliance,137,Ralph Waldo,Emerson,Art
97,The Great God Pan And The Hill Of Dreams,158,Arthur,Machen,Romance
98,The White People and Other Weird Stories,100,Arthur,Machen,Fantasy
99,The Diary of Samuel Pepys: A Selection,271,Samuel,Pepys,Drama
100,The Rise of Silas Lapham,314,William Dean,Howells,Satire
101,The Mystery of Edwin Drood,340,Charles,Dickens,Diaries
102,In Exile and Other Stories,140,Mary Hallock,Foote,Comics
103,Bleak House,130,Charles,Dickens,Art
104,U.S.A Trilogy,118,John,DosPassos,Science fiction
105,A Comedy of Masks,237,Ernest,Dowson,Journals
106,Barnaby Rudge,380,Charles,Dickens,Diaries
107,Heroic Deeds of American Sailors,243,Albert,Blaisdell,Science fiction
108,A Bouquet,309,Ernest,Dowson,Art
109,Editha,155,William Dean,Howells,Art
110,Manhattan Transfer,124,John,DosPassos,Drama
111,Fasti,325,Publius,Ovid,Horror
112,Hearts of Three,272,Jack,London,Sati
113,"Speak, Memory",154,Vladimir,Nabokov
114,Kidnapped,334,Robert Louis,Stevenson
115,The Brothers Karamazov,161,Fyodor,Do
116,Envelope Poems,342,Emily,Dickinson,F
117,Selected Poems,203,Emily,Dickinson,Guide
118,Gobolinks or Shadow Pictures for Young and Old,332,Ruth,Stuart,Romance
119,The Decameron,318,Givoanni,Boccaccio,Diaries
120,The Cock and Anchor,133,Joseph Sheridan,LeFanu,Science fiction
121,Adrian Rome,259,Ernest,Dowson,Journals
122,Words in Genesis,217,Isaac,Asimov,Science
123,Stories of Ohio,191,William Dean,Howells,Romance
BANDICAM UNREGISTERED
REC
1366x728 - (0, -8), (1366, 720)
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Plain Text Go Live