|  |  |
| --- | --- |
|  | **AGH – UNIVERSITY OF SCIENCE AND TECHNOLOGY** |

Project documentation for

**Library Management System**

**Object-oriented programming languages**

Electronics and Telecommunication EN, III year

*Konrad Włodarczyk*

lecturer: Rafał Frączek

23.01.2025

# Project description

This project is a Library Management System developed in C++ that allows students to borrow, return and reserve books efficiently. It maintains records of issued books, all students, all books and calculates overdue fines. This project uses object-oriented programming principles such as inheritance, encapsulation and polymorphism.

# User’s manual

Features: ‘Add Book’, ‘Add Student’ and ‘Display All Students’ required a password that is predefined right below library declarations.

**Features:**

1. Add Book: Enter Title, number of pages, ISBN, whether it is issued and whether it is reserved (if yes, display student’s ID’.
2. Add Student: Enter Student ID, name, age and whether he has book reserved
3. Issue Book: Enter Student ID and ISBN of a book
4. Return Book: Enter Book ISBN to return
5. Reserve Book: Enter Student ID and ISBN of a book
6. Display All Books: Displays all books available in the library
7. Display All Students: Displays all student available in the database
8. Display Overdue Books: Displays Title, ISBN and fine if issued for more than 15 days.
9. Exit: Exits the program.

Invalid inputs (or student/book missing from the database) are handled and appropriate information is displayed.

# Compilation

To compile the project, use a C++ (g++ for example)

**Standard build command:**

g++ -o online-library -Wall online-library.cpp

**Run the project using:**

./online-library

# Source files

The project consists of only one source file:

* online-library.cpp – Contains all class definitions (Person, Student, Book, Library) and the main program logic, including the menu system and handling.

# Dependencies

The project does not use any external libraries beyond the C++ Standard Library.

# Class description

* Person – base class for Student.
  + virtual void display() const – Pure virtual function to display information
  + Person(const string& name, int age) – Constructor
  + Virtual ~Person() – Virtual destructor
* Student (inherited from Person) – Represents a student using the library.
  + const int& getStudentID() const – Returns Student ID
  + Students(consts string& name, int age, int student\_ID) - Constructor
  + bool canIssueBook() const – Checks if a book can be issued.
  + void issueBook() – Marks a book as issued
  + void returnBook() – Unmarks the book as issued
  + void display() const override – Displays student details
  + string serialize() const – Converts student data into a string format for storage
  + string shared\_ptr<student deserialize(const string& data) – Converts stored string data back into a Student object (technically not a method since it does not operate on an existing instance, but I put it here for clarification)
* Book – Represents a book in the library
  + Book(const string& title, int page\_count, int ISBN, bool is\_issued = false) – Constructor,
  + const int& getISBN()– Returns book ISBN
  + const string& getTitle() const – Returns book title
  + book getIssuedStatus() const – Checks if the book is issued
  + int getReservedBy() const – Returns the student ID of the person who reserved the book
  + void issue() – Issues the book
  + void returnBook() – Returns the book
  + void reserve(int student\_ID) – Reserves the book
  + void display() const – Displays book details
  + string serialize() const – Converts book data into a string format for storage
  + static shared\_ptr<Book> deserialize(const string& data) – Converts stored string data back into a Book object (technically not a method since it does not operate on an existing instance, but I put it here for clarification)
* Library – Manages books and students.
  + Library() – Constructor that loads books and students from files
  + ~Library() – Destructor that saves books and students into files.
  + void addBook(const string& title, int page\_count, int ISBN) – Adds a book to the library.
  + void addStudent(const string& name, int age, int student\_ID) – Adds a student
  + void issueBook(int student\_ID, int ISBN) – Issues a book.
  + void returnBook(int ISBN) – Returns a book.
  + void reserve(int student\_ID, int ISBN) – Reserves a book
  + void displayAllBooks() const – Displays all books
  + void displayAllStudent() const – Displays all students
  + void displayOverdueBooks() const – Displays overdue books and fines
  + void addBookUI() – UI function for adding books
  + void addStudentUI() – UI function for adding students
  + void issueBookUI() – UI function for issuing books
  + void returnBookUI() – UI function for returning books
  + void returnBookUI() – UI function for returning books
  + void saveBooks() const – Saves book data to file
  + void saveStudents() const – Saves student data to file
  + void loadBook() – Loads book data from file
  + void loadStudent() – Loads student data from file
  + shared\_ptr<Book> findBook(int) const – Finds a book by ISBN
  + shared\_ptr<Student> findStudent(int) const – Finds a student by ID

There is also one function outside of the classes:

* bool checkPassword() – Checks whether the password inputted is correct.

# Resources

In the project the following resources are used:

* books.txt – a file with all books. The file structure:
  + Books title
  + Number of pages
  + ISBN
  + Is issued?
  + Reserved By

All data is written in a single line, one line per book. The data is divided with a bar symbol (‘|’)

* students.txt – a file with all students. The file structure:
  + Name
  + Age
  + Student ID
  + Has reserved book?

All data is written in a single line, one line per student. The data is divided with a bar symbol (‘|’)

# Future development

Possible future improvements:

* Implementing a graphical user interface (GUI).
* Enhancing the reservation system with waitlists.
* Using a database for better data management.
* Adding a better and more secure authentication for library staff.

# Other

None.