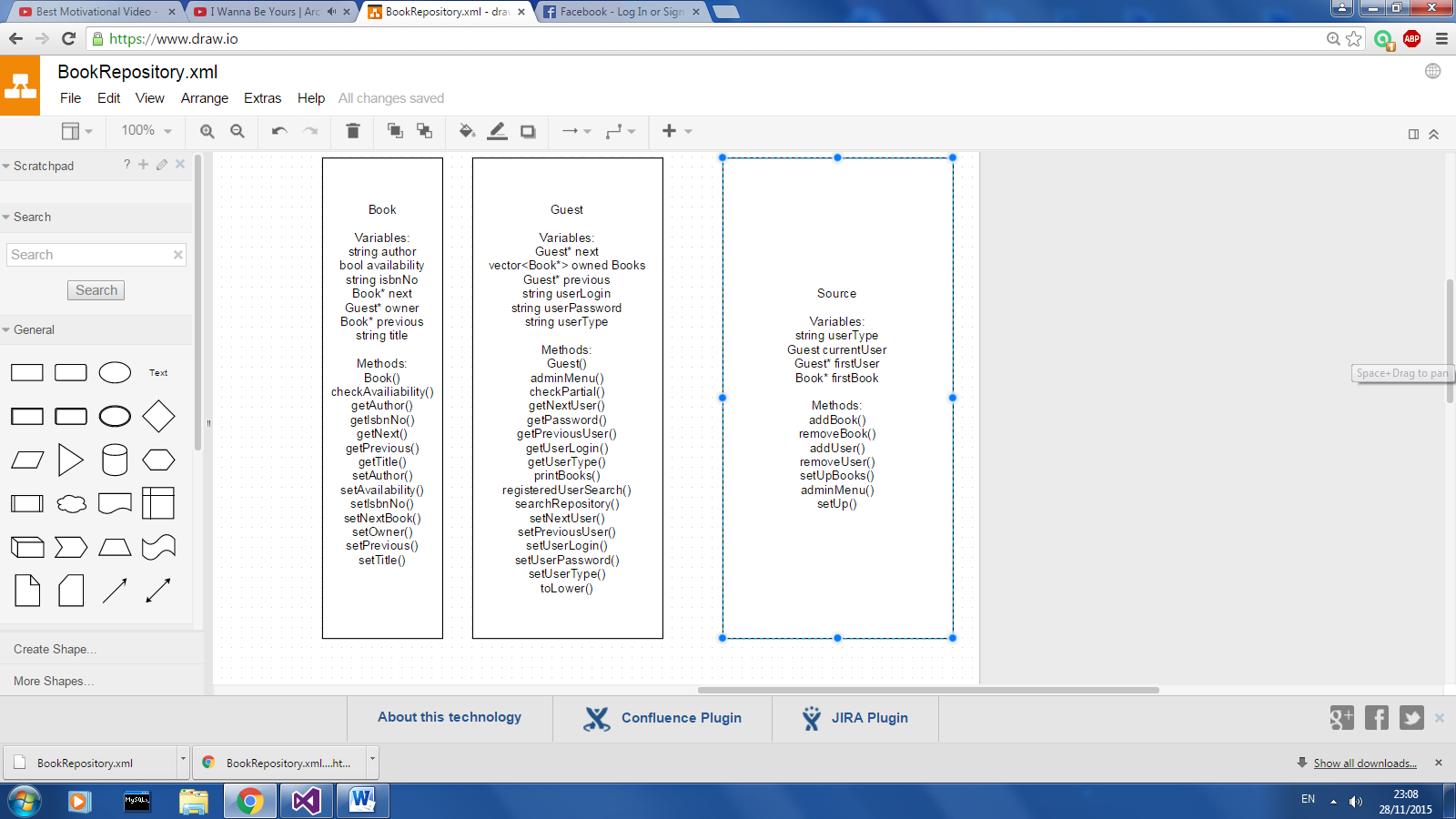
Alan

C++ Book Repository

A00209408

# UML Diagram:



# Class: Book

The basic book class encapsulating the following details: string title, string author, string isbnNo, bool availability.

The class also includes set and get methods for each one of those variables.

# Class: Guest

The guest class serves as a type of a base class through which the users are split up into either guest, registered or admin users.

The variables encapsulated in this class are: string userLogin, string userPassword, string userType, vector<Book\*> ownedBooks, Guest\* next, Guest\* previous.

Besides the get and set methods for these variables the class has some other methods such as:

* toLower(string param)
  + This method iterates through the string and sets the letters to lowercase
* chechPartial(string input, string compare)
  + The method puts the strings into vectors of char type.
  + It iterates through the vectors and compares each character
  + If a matching char is found but the other chars fail it steps back to that position
  + Therefore this method allows for any combination or part of a word to be found no matter where the user input is in the word being compared
* printBooks(vector(book\*) bookVector)
  + A vector from the repository search is passed into the method which then goes on to print out all the books inside of it.
  + It also gives each book a number so that the user can choose the book he wishes to see.
* searchRepository(Book\* firstBook)
  + The firstBook parameter is used as the start of the linked list through which the method iterates and using the checkPartial method compares the input to the linked list book title, author and isbn number respectively.
  + Any books who match this search are then put into the bookVector and sent to the printBooks method.
  + The user is then asked which book he wants to choose and a pointer to that book is then returned from the method.
* registeredUserSearch(Book\* firstBook)
  + The method calls the search repository method and then uses the returned book object.
  + The method prints out whether the book is available or not.
  + If the book is available the user is asked if he wants to check it out.
  + The book is then added to the user’s bookVector and the user is added to the book owner variable.
  + If the book is unavailable the method checks whether user is the owner of the book and if he is he can return the book.

# Source:

The source method includes the basic objects and their set up, as well as some menu methods.

Important variables include: Guest currentUser , Guest\* firstUser , Guest\* firstBook.

The two first variables are the first object in my custom linked lists.

Methods:

* setUpBooks()
  + Populates book objects and sets them in the correct order for the linked list.
* setUp()
  + Creates some of the basic user objects.
  + A switch case is set up and the user can either go straight to searching the repository or log in for more options.
  + Asks the user for input and if he is trying to log in the input is compared to each of the user logins and passwords in the linked lists.
  + The user then goes into either the admin or registered user menu.
* adminMenu()
  + A switch case menu allows the user to either:
    - Return
    - Search Repository
    - Add book
    - Delete book
    - Add user
    - Delete user
  + Since the searchRepository method returns a book object it also has another small switch case which allows the admin to set the book as available or unavailable.
* addBook()
  + Asks the admin for book details and then creates an object based on those.
  + The book linked list is then iterated through and the new object is added to the list
* removeBook()
  + The method calls the searchRepository method and based on the object it gets back it then deletes that object from the linked list.
* addUser()
  + Asks the admin for new user details and then creates that user.
  + It then iterates throught the user linked list and adds the new user object.
* removeUser()
  + Asks the admin for the user name of the user it is supposted to delete.
  + It then iterates through the linked list and if it finds the user the admin is asked to confirm the user is to be delted.