# Important Instructions / Guidelines for Sprint-1 (Java) – SD11

* Use proper indentation, comments, naming conventions and self-explanatory names if you want to secure better marks.
* **This project will be** **team based or solo and you can have a maximum of 3 people in a group. If solo is chosen, you will not have to implement the working menu with the scanner.**
* All classes should be in different files
* Submit all files using a GitHub Repository. Submit the link to the assignment page **ZIP Files Will Not Be Marked!**

Create a library Management system that manages library items, authors, and patrons. The system should have the following features:

1. Item management: Libraries can have Periodicals and books. Periodicals can be Printed or electronic. Books can be printed, electronic and audio. The system should allow librarians to add new library item, edit existing items, and delete items from the library. Each item should have an ID, title, author, ISBN, publisher, and number of copies and other attributes specific to their type.
2. Author management: The system should allow librarians to add new authors, edit existing authors, and delete authors from the library. Each author should have a name, date of birth, and a list of items they have written.
3. Patron management: The system should allow librarians to add new patrons, edit existing patrons, and delete patrons from the library. Each patron should have a name, address, phone number, and a list of Library Items they have borrowed. Patron can be a student or an employee.
4. Library Item borrowing: The system should allow patrons to borrow Library items (books, periodicals) from the library.

Patrons should be able to search for items by title, author, or ISBN, and then borrow a specific number of copies of the item. If the item is not available, the system should display a message indicating the item is currently checked out.

1. Library Item Returning: The system should allow patrons to return items they have borrowed. Patrons should be able to search for the items they have borrowed, and then return a specific number of copies of the item.
2. **(TEAM BASED REQUIRMENT)** You must implement a working menu system that allows users to navigate through the different functionalities of your library management system. The menu system should utilize the **Scanner class** to capture user input. This interactive console-based menu will facilitate operations such as adding, editing, deleting, borrowing, and returning items and all other required features. **An example of a menu will be provided in the assignment portal!**

# Deliverables:

## **Code files: [30 marks]**

Design and implement the code using all the requirements mentioned above. A complete running code file for each of the following is required.

1. LibraryItem: A class hierarchy that represents library items. Libraries can have Periodicals and Books
2. `Author`: A class that represents an author. The `Author` class should have properties such as name, data of birth and a list of items they have written.
3. `Patron`: A class hierarchy that represents a patron. The `Patron` class should have properties such as name, address, phone number, and list of items they have borrowed.
4. `Library`: A class that manages library items, authors, and patrons. The `Library` class should have methods to search for library items by title, author, or ISBN, and to borrow and return those items.
5. A Demo Class that will be the main point of entry for your users to the application. Your menu could live here? The purpose of this class is to showcase all the required features and them working.

**Challenges(!!Optional!!)**

**Implement The Following Classes Into Your Project**

* `Status`: An Enum that represents the status of a book. The `Status` Enum should have values such as `AVAILABLE`, `CHECKED\_OUT`, and `OVERDUE`.

**Explore these task**

* After you have completed your project in Visual Studio Code. Try to get you code working in **Intellij IDE Community Edition.** (Getting familiar with this IDE will benefit you in the next term!) **Provide a screenshot of your code working in Intellij to complete this task!**

<https://www.jetbrains.com/products/compare/?product=idea&product=idea-ce>

* Organize your project structure so it is all not in one giant directory Try and separate things in different packages to make it easier to navigate. i.e All code related to the ‘Library’ class should be in the same folder(package).

## **Documentation: [ 10 marks]**

You need to document your java project above.

**Please make one document for the project include sections**

1. User Documentation This include a document stating what the application is about, explanation of all the classes and their working, and how to start it/access it. Also include the class diagram with the associations between them. [6]
2. Development Documentation This includes at least the Javadocs, a description of the source code directory structure, the build process (i.e., how to compile the project), compiler time dependencies, development standards, how a database would look in theory for this project (**It Does Not Have To Be Actually Setup!)**. Include the entity relationships with your database design, and how to get the source code from the **GitHub repository.** [2]
3. Deployment Documentation This is basically the installation manual of the application, describing any steps needed to make it run. [2]
4. **Please submit a document separately stating the contributions of each team member. This is important! (We want to ensure that each team member is contributing) Contributions will be based off GitHub commits if this document is not submitted**

## **3. To demonstrate the project [10 marks]**

1. Create a main class to test the library management system and its features. Add appropriate data (at least 10 records) to show the working. [5]

Hint: you can write a method, to load the data every time you run your code.

1. Record a small video to demonstrate the working of your library management system. Explain all the methods, run the code and explain the functionality and the outputs. [5]

Note: Video must not exceed 5 minutes and must not be less than 2 minutes.

**Good Luck! Have Fun!**