**SALESFORCE PROJECT FOR LIBRARY MANAGEMENT SYSTEM (BookiBeam)**

**🔹 Step 1: Define the Core Objects (Entities)**

We want about **6–7 objects** total (standard + custom). Here’s a solid lineup for BookieBeam:

1. **Student (or “Library Member”)** – core users/consumers.
   * Custom Object: Student\_\_c (you can rename the tab to *Students/Members*).
   * Why: Salesforce already treats Contacts as “people,” saves you effort.
2. **Librarian** – manages circulation, approvals, inventory.
   * Custom Object: Librarian\_\_c.
   * Fields: Name, Employee ID, Shift, Email, Phone.
3. **Book** – catalog of books.
   * Custom Object: Book\_\_c.
   * Fields: ISBN, Title, Author, Genre, Publisher, Status (Available/Issued/Lost), Rack No.
4. **Book Issue/Return** – circulation tracker.
   * Junction Object: Transaction\_\_c between Book and Student.
   * Fields: Transaction ID, Book (Lookup), Student (Lookup), Issue Date, Due Date, Return Date, Status (Issued/Returned/Overdue).
   * For the first time when book is issued the status will be Issued. If due date crossed the deadline then the status will changed to Overdue.
5. **Book Request** – for students to request unavailable books.
   * Custom Object: Book\_Request\_\_c.
   * Fields: Student (Lookup), Book (Lookup), Request Date, Status (Pending/Approved/Rejected).
6. **Operator (Data Entry Staff)** – Salesforce Platform User
   * Fields: Name, Employee ID, Assigned Librarian (Lookup).
7. **Library Inventory/Stock** (optional but cool).
   * Custom Object: Inventory\_\_c.
   * Fields: Book (Lookup), Total Copies, Available Copies, Damaged Copies.

**🔹 Step 2: Relationships**

Use **Lookups & Master-Detail** for structure: S

* Student and Transaction -> Create a lookup field in the Transaction object for Student.
* Book and Transaction -> Create a lookup field in the Transaction object for Book.
* Librarian and Transaction -> Create a lookup field in the Transaction object for Librarian .
* Book and Inventory (Master-Detail) -> Master-Detailed field in Book.
* Book and Request (Master-Detail)-> Master- Detailed field in Book Request.

**🔹 Step 3: Tabs & Navigation**

Create tabs for:

* Students
* Librarians
* Books
* Transactions
* Requests
* Inventory

Also pin these tabs to your BookieBeam app for quick access.

**🔹 Step 4: Fields (must-haves per object)**

**Book\_\_c**

* Title (Text)
* Book code(unique number)
* ISBN (Text)
* Author (Text)
* Publisher *(Text)*
* Edition *(Text)*
* Genre (Picklist)
* Rack Number (Text)
* Language *(Picklist)*
* Published Year *(Number or Date)*
* Status (Picklist: Available, Issued, Damaged, Lost)
* Price*(Currency)*
* Date Added to Library *(Date)*
* Tags/Keywords *(Text or Multi-Select Picklist)*
* Summary/Description *(Long Text Area)*
* Inventory(Master-Detail)
* Roll-up Summary fields on Book can show:
  + Total Copies across all inventory records
  + Total Available Copies
  + Total Damaged / Lost Copies

If you only track aggregated inventory per book, then focus on Total Copies, Available, Damaged, Lost, Times Borrowed.

Use Roll-up Summary fields on Book to calculate:

* + Total Copies = SUM of Total Copies from Inventory
  + Total Available = SUM of Available Copies
  + Total Damaged = SUM of Damaged Copies
  + Total Lost = SUM of Lost Copies

**Student and Librarian as Contact Object:-**

**Student:**

* First Name *(Text)*
* Last Name *(Text)*
* Department*(Picklist: Btech,Bsc,BCA)*
* Semester(Picklist: 1st, 2nd , …..)
* Stream *(Picklist:Computer Sc, CSE, ECE,IT….)*
* Roll Number *(Text, unique)*
* Use **Roll-up Summary fields** on Student and Book to track **currently borrowed books, overdue books, total fines, etc.**

**Librarian:**

* Roll no *(Auto Number/Text, unique)*
* First Name *(Text)*
* Last Name *(Text)*
* Designation/Role *(Picklist: Librarian, Assistant Librarian, Admin)*
* Date of Joining *(Date)*
* Work Shift/Timings *(Text or Picklist)*

**Transaction\_\_c**

* Transaction ID (Auto Number)
* Student (Lookup to Contact)
* Book (Lookup to Book)
* Librarian(Lookup)
* Issue Date (Date)
* Renewal Date(Date)(Formula field)Issue date + 7 days.
* Due Date (Date)(Formula field) Renewal date + 7 days.
* Return Date (Date)
* IsRenewed?(Checkbox)-> formula field will be set to true when Return date is blank and current date exceeded the renewal date.
* TransactionStatus (Picklist: Issued, Returned, Overdue, Renewal,Lost)
* Through flow we need to update the status to Overdue.
* Note:- When status is overdue then Fine Amount,Fine Amount Paid,Overdue days, Payment Status and Transaction Completed checkbox will be updated.
* Overdue Days *(Formula → Due Date – Current Date if returned late)*
* Fine Amount *(Currency → calculated based on overdue days or system logic*
* Fine Amount Paid (Currency)
* Payment Status *(Picklist: Paid, Unpaid, exempted)*
* Transaction Completed? *(Checkbox → TRUE if book returned and fine cleared)*
* Book Condition (On Issue) *(Picklist: New, Good, Worn, Damaged)*
* Book Condition (On Return) *(Picklist: New, Good, Worn, Damaged)*

Note:- When the Fine Amount paid is equal to the Fine Amount, then the Payment Status will be paid, and the Transaction Completed will be True. But the Transaction status will remain Overdue unless and until the student returns the book.

**Inventory:-**

* Total Copies *(Number, Required)* → total copies of the book in the library
* Available Copies *(Number, Formula or manually updated)* → total copies currently available for issue
* Issued Copies will be calculated through the transaction object.

**🔹 Step 5: Automations (Flows/Validation Rules)**

Here’s where you flex:

* **Flow:** Auto-update Book status to *Issued* when Transaction is created.
* **Validation Rule:** Prevent issue if Book status = “Issued.”
* **Flow:** Auto-calculate “Overdue” if Return Date > Due Date.
* **Flow:** When Request is approved, auto-create a Transaction.
* **Roll-Up Summary:** On Inventory, calculate Available Copies = Total – Issued.

**🔹 Step 6: Security & Sharing**

* Create **Profiles**: Librarian, Operator, Student.
* Control CRUD: e.g., Students can only create Requests, not Transactions.
* **Role Hierarchy:** Librarian > Operator > Student.
* Use **Field-Level Security** to hide internal fields from Students.

**🔹 Step 7: Reports & Dashboards**

Reports to impress:

* “Books Issued by Genre” (Matrix).
* “Overdue Books by Student.”
* “Top 5 Most Issued Books.”
* “Pending Requests Report.”

Dashboard:

* Pie chart: Issued vs Available Books.
* Bar chart: Issued Books by Genre.
* Table: Students with max overdue.

**🔹 Step 8: UI/UX Polish**

* Use **Lightning App Builder** to customize Home Page: show KPIs (Total Books, Active Issues, Overdues).
* Create **Compact Layouts**: Book record shows Title, Author, Status.
* Quick Actions: “Issue Book” button directly on Student record.

**🔹 Step 9: Apex (Extra Credit)**

* Write a **Trigger** on Transaction\_\_c: If book returned late, auto-calculate Fine.
* SOQL query to fetch “Top 10 most requested books.”

**🔹 Step 10: Project Storyline (for your viva/demo)**

* BookieBeam manages end-to-end library operations.
* Students request, Librarians approve, Operators maintain catalog, Transactions track flow.
* Reports & Dashboards provide insights.
* Automations reduce manual effort.
* Apex adds smart logic like fines and auto-approvals.

**🔹 Step 11: Scenario to be added**

* If book a book is out of stock through automation create a record on book request object.
* If a student issues more than 5 books a week create a validation rule for the next issue.