**Submittedby:**

1. PragyaBoral–23MCB1018
2. VinushaCV-23MAI1006

## Introduction:

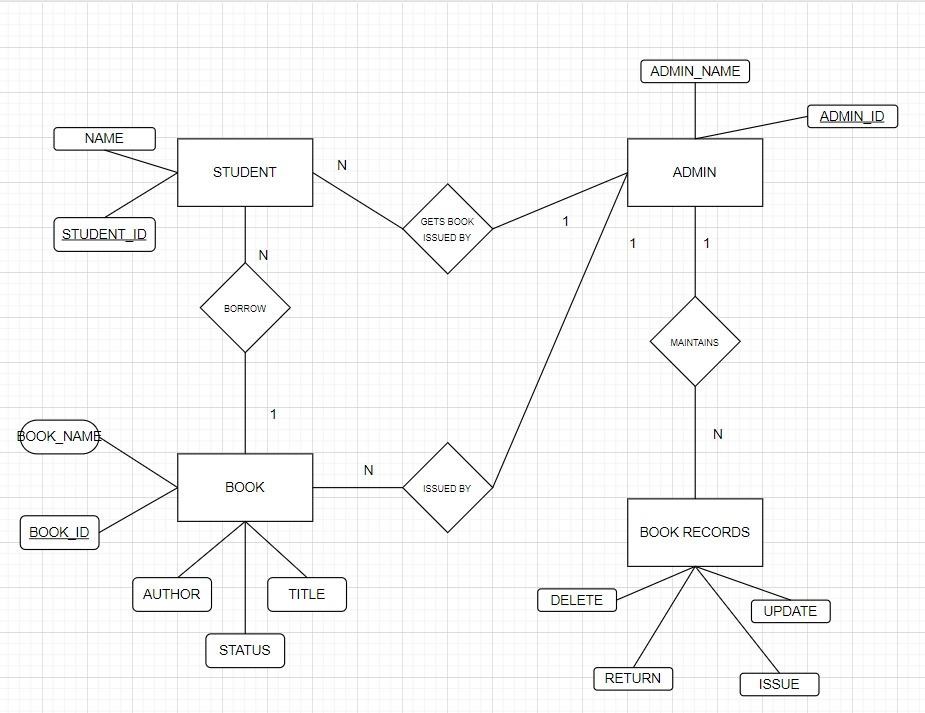
The Library Management System is a sophisticated software application designed to revolutionize library operations. It provides librarians and library users with a user- friendly platform for managing and accessing resources. This project leverages Python and MySQL to create a dynamic system. Librarians can efficiently catalog books, manage user accounts, and monitor book availability. Users benefit from a robust search interface to locate books, borrow and return them seamlessly. The system supports multiple user roles, ensuring an inclusive experience. With an intuitive graphicaluserinterface,theLibraryManagementSystemenhancesthemanagementof library resources and offers a modern, efficient solution for libraries of all sizes.

## ER Diagram:

Inalibrarymanagementsystem,variousentitiesandtheirattributescould include:

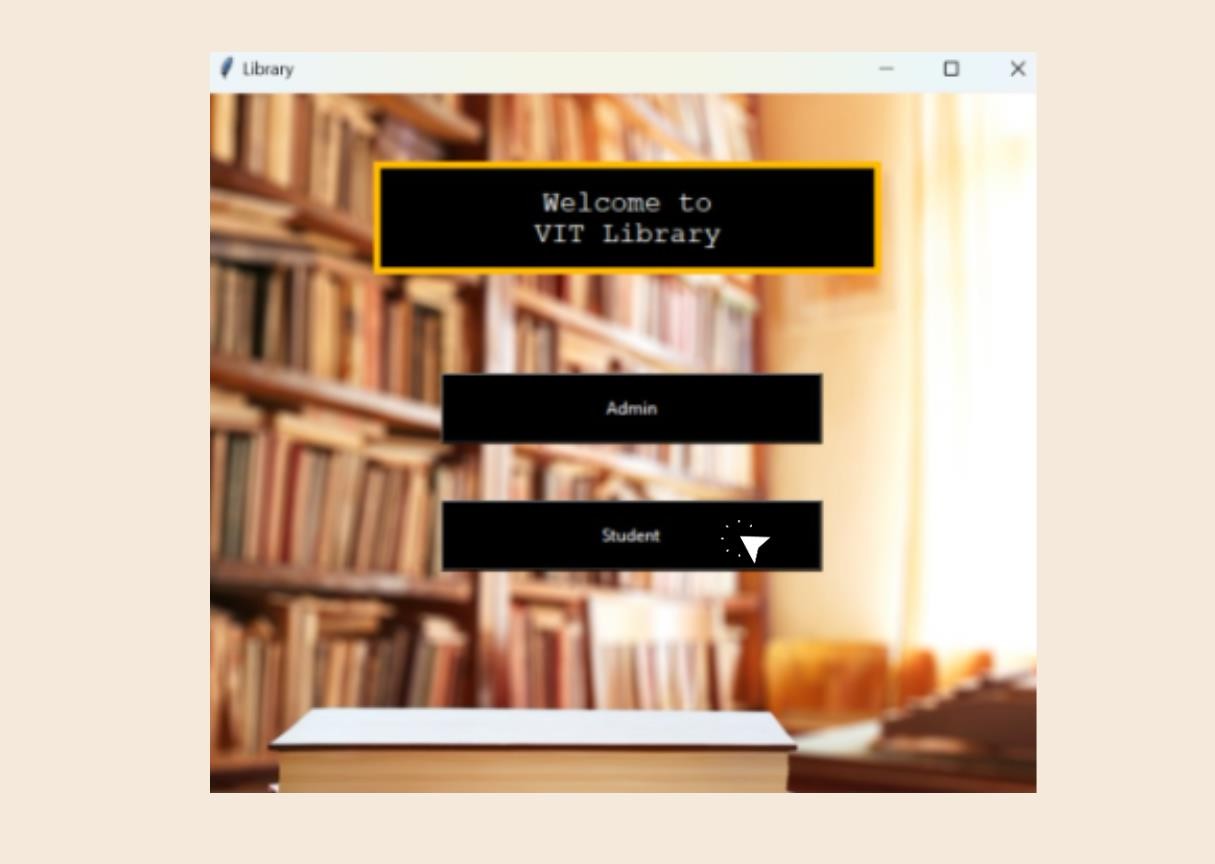
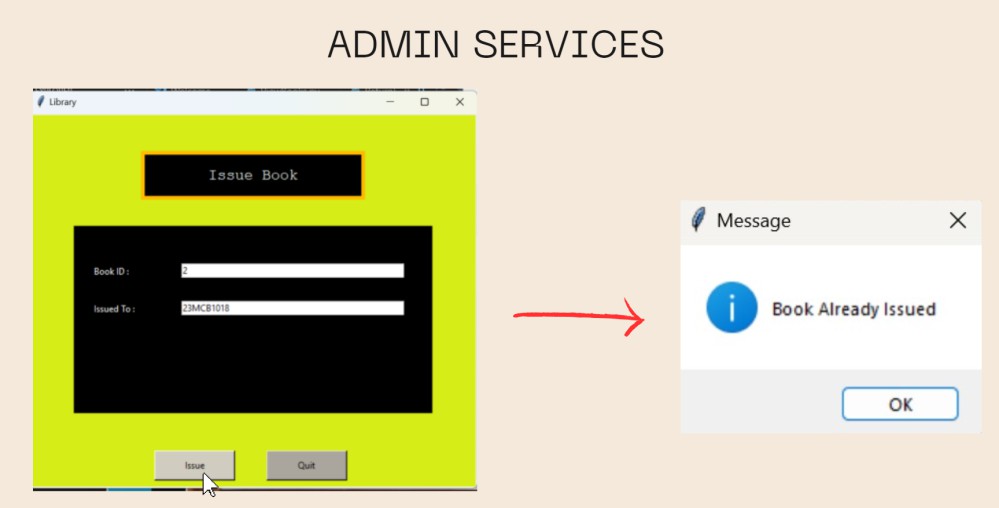
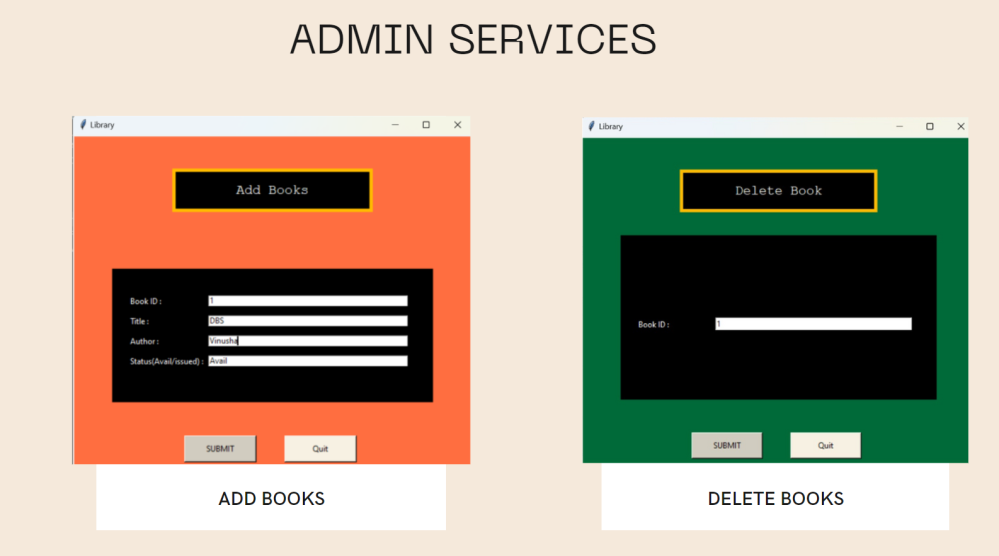
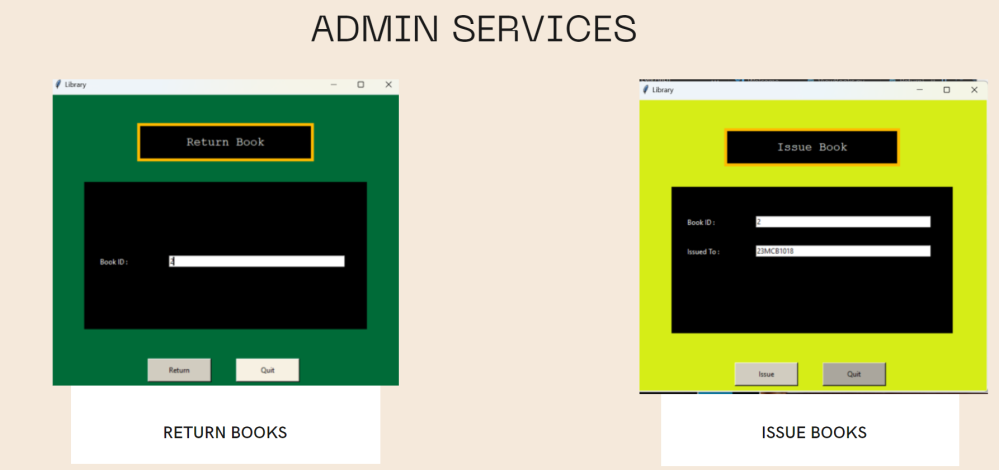
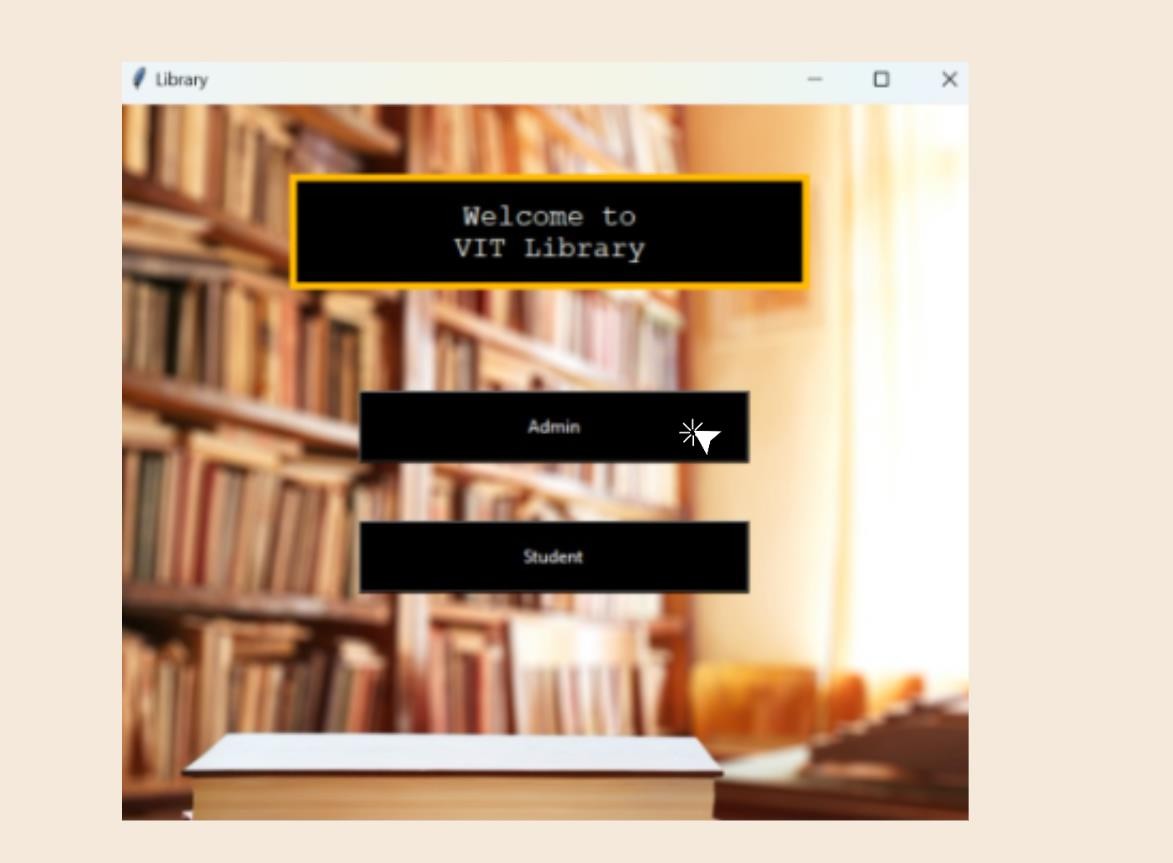
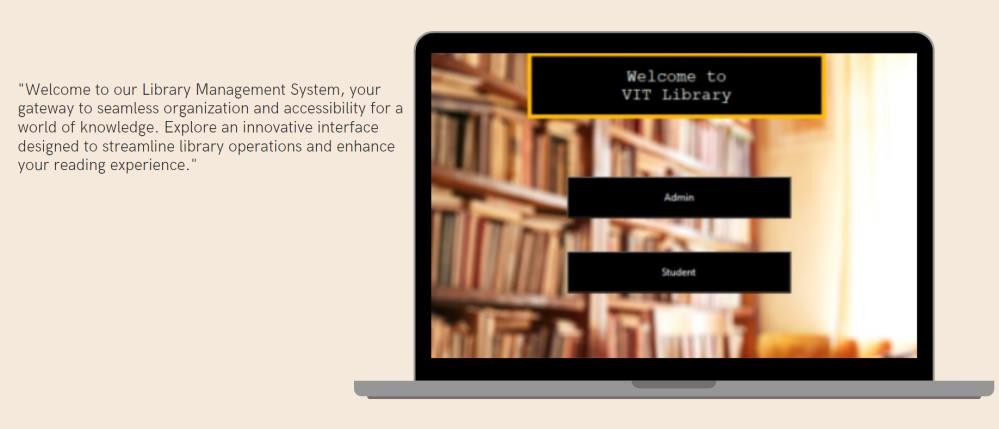
1. ENTITY:Book ATTRIBUTES
   * Title
   * Author
   * Book\_id(primarykey)
   * Book\_name
   * Status
2. ENTITY:Student ATTRIBUTES
   * Name
   * Student\_ID(primarykey)
   * ContactInformation
3. ENTITY:Admin ATTRIBUTES
   * Name
   * Admin\_ID
4. ENTITY:BookRecords ATTRIBUTES
   * Issue
   * Update
   * Return

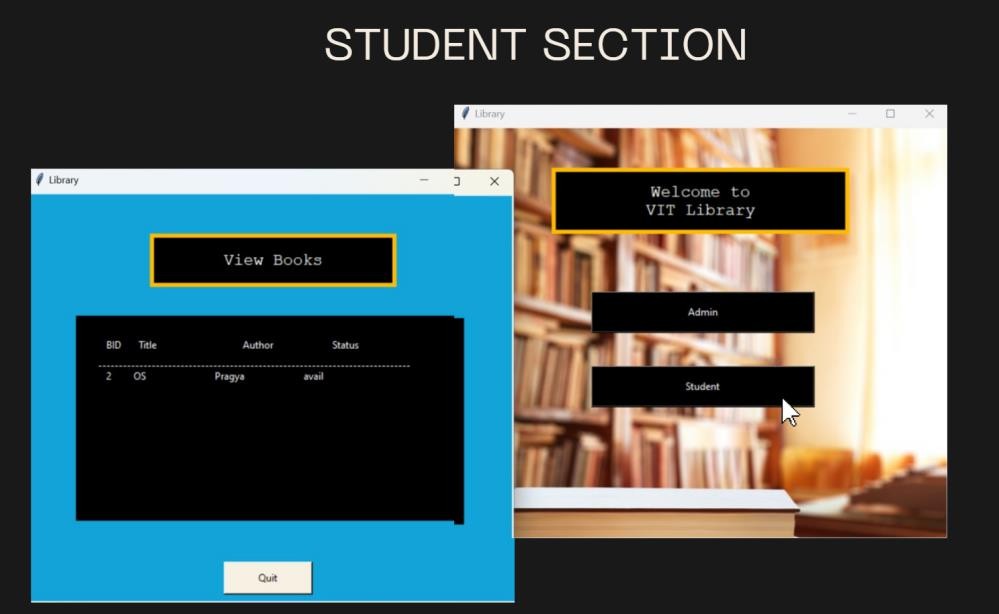
Cardinalitiesbasedontheassumedrelationshipsfromtheprovidedentities: One-to-Many (1:N)

* studenttobook:OnestudentcanhavemanyBook.
* admintoBookRecords:OneadmincanaccessmanyBookRecords.
* admintoStudent:OneadmincanaccessissueformanyStudents The ER diagram for the project was created using Draw.io.

## Canva:

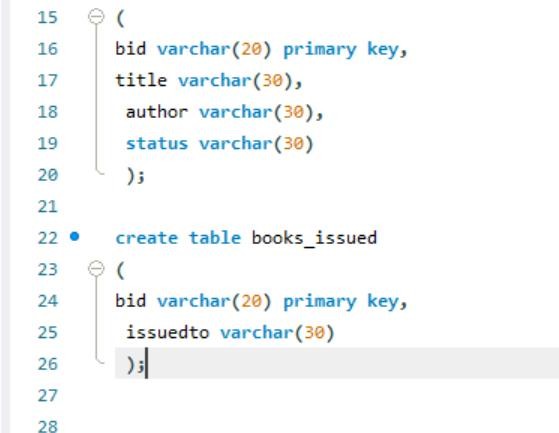
Samplescreenandlinkforthecanvaisattached.





Link:<https://vit-library2023.my.canva.site/mini-project>

**Tablecreated:**



Normalization:

First Normal Form (1NF):

- In the context of the Library Management System, achieving 1NF means ensuring that each table contains only atomic (indivisible) values.

- For example, consider the "Books" table. To satisfy 1NF, you would avoid storing multiple authors in a single column. Instead, create a separate "Authors" table, with each author having a unique identifier. The "Books" table can then reference the "Authors" table using foreign keys.

- Similarly, avoid storing multiple categories or genres in a single column. Instead, create a "Genres" or "Categories" table to store these values and reference them from the "Books" table.

Second Normal Form (2NF):

- 2NF builds on 1NF and focuses on ensuring that non-key attributes are fully functionally dependent on the entire primary key. In simpler terms, it means that if a table has a composite primary key, each non-key attribute should depend on the entire composite key.

- For instance, let's say you have a "BorrowedBooks" table with a composite primary key consisting of "UserID" and "BookID." The "DueDate" should depend on both "UserID" and "BookID" as it represents the due date for a specific user and a specific book.

Third Normal Form (3NF):

- 3NF goes further by addressing transitive dependencies. It means that non-key attributes should not depend on other non-key attributes.

- In the Library Management System, consider the "Users" table. Let's say it contains both user information and their contact details. If user contact information (like email, phone) is dependent on a user's address, you should create a separate "User Details" table with its own primary key and include a foreign key in the "Users" table to establish the relationship. This separation ensures that user details are not dependent on the address.

In addition to 1NF, 2NF, and 3NF, you may also consider other normal forms depending on the complexity of your data model. For example, if you have multi-valued dependencies, you would need to address Fourth Normal Form (4NF) and Fifth Normal Form (5NF). This approach enables better data management and ensures that the data remains consistent and reliable. Normalization also simplifies data retrieval and maintenance, improving the overall performance of the system. However, the specific application of these principles should be based on the complexity and specific requirements of your database schema.