Online Library Management System

Submitted in partial fulfillment of the requirements

for the award of the Degree of B.C.A.

Submitted by:

(Student Name)

Reg.No:

Submitted to

GULBARGA UNIVERSITY, KALABURAGI

College/Department

College Name and City

ACKNOWLEDGEMENT

I would like to pay my praise and humblest thanks to God, the most merciful and beneficent, who bestowed upon me the ability and strength to complete this project.

I am highly indebted to **(Name of your Mentor)** for their valuable guidance and constant supervision, as well as for providing necessary information regarding the project. I am extremely thankful for their support in completing this work, despite their busy schedule managing corporate responsibilities.

I would like to express my heartfelt gratitude towards my parents and members of **(College Name)** for their kind cooperation and encouragement, which helped me immensely in the successful completion of this project.

I am also thankful to and feel fortunate enough to have received constant encouragement, support, and guidance from all the teaching staff of the **[Department Name]**, which played a vital role in the successful completion of my project work.

INDEX

Abstract

Library management system is a project which aims in developing a computerized system to maintain all the daily work of library .This project has many features which are generally not available in normal library management systems like facility of user login and a facility of admin login .It also has a facility of admin login through which the admin can monitor the whole system. It has also a facility where student after logging in their accounts can see list of books issued and its issue date and return date.

Overall, this project of ours is being developed to help the students as well as staff of library to maintain the library in the best way possible and also reduce the human efforts.

1. Main Report

1.1. Introduction

The **Library Management System** is a web-based or desktop application designed to modernize and automate the daily operations of a traditional library. The primary goal of this system is to facilitate the efficient management of library resources, streamline the process of issuing and returning books, and reduce manual workload for librarians and staff.

This system offers two types of user access: **Admin Login** and **User (Student) Login**. The **Admin Module** allows library staff to add new books, manage book records, issue or return books, and monitor the overall system activity. On the other hand, the **Student Module** enables students to log in to their accounts, view the list of books they have issued, check issue and return dates, and search for book availability.

By transitioning to a computerized system, the library can ensure accuracy, improve record-keeping, and reduce the chances of human error. Additionally, the system enhances user experience by providing transparency and self-service functionalities for students.

This project not only aims to digitalize the entire library process but also to create a user-friendly environment where both students and library staff can perform their respective tasks more efficiently and conveniently.

1.2. Problem Statement

The problem occurred before having computerized system includes:

- File lost
- When computerized system is not implemented file is always lost because of human environment. Sometimes due to some human error there may be a loss of records.
- File damaged when a computerized system is not there file is always lost due to some accident like spilling of water by some member on file accidentally. Besides some natural disaster like floods or fires may also damage the files.
- Difficult to search record

- When there is no computerized system there is always a difficulty in searching of records if the records are large in number.

- Space consuming

- After the number of records become large the space for physical storage of file and records also increases if no computerized system is implemented.

- Cost consuming

- As there is no computerized system to add each record paper will be needed which will increase the cost for the management of library.

1.3 Literature Survey

A literature survey is essential to understand the existing work and technological advancements in the domain of library management systems. Several systems have been developed in the past, each aiming to improve the efficiency, accessibility, and accuracy of library operations. Here is a review of some notable efforts:

1. **Traditional Manual Systems:**

    - In conventional libraries, book records were maintained manually using registers and logbooks.
    - These systems were prone to errors, time-consuming, and lacked real-time tracking of issued or returned books.
    - Manual processes often resulted in misplacement of books and difficulties in inventory management.

2. **Basic Digital Library Systems:**

    - Early computerized systems were developed using Microsoft Access or spreadsheet tools.
    - These provided basic functionalities like book record storage and search.
    - However, they lacked user authentication, multi-user access, and real-time updates.

3. **Modern Web-Based Library Systems:**

    - Recent developments use programming languages like PHP, Java, Python, and frameworks like Django and Laravel.
    - These systems support multi-user roles, real-time updates, book search, issuing, and return tracking.
    - Some systems also integrate barcode scanning and SMS/email notification features.

4. **Related Research Works:**

    - Various studies have suggested the implementation of RFID (Radio Frequency Identification) for book tracking.
    - Researchers have explored the use of cloud-based databases to enhance data security and remote access.
    - Machine learning models are also being explored for predicting book demand and automating categorization.

1.4 System analysis & design

System Analysis

EXISTING SYSTEM

In the existing library systems, most processes are either **manual** or based on **basic digital tools** with limited functionalities. The traditional library setup generally involves librarians maintaining handwritten records of books issued and returned, and students manually searching for books in catalogs or shelves.

Some institutions have adopted **basic digital systems**, often created using spreadsheet tools or outdated software that support only single-user environments and lack flexibility and scalability.

Limitations of the Existing System:

- Manual entry increases the chances of **human error**.
- **Time-consuming** processes for book search, issue, and return.
- No **real-time tracking** of book availability.
- Lack of **role-based access control** (no clear division between admin and student access).
- Inadequate report generation and **inefficient record management**.
- No online access for students to check book status or manage their accounts.
- Often requires physical presence for every interaction, causing **inconvenience**.

Due to these limitations, there is a clear need for a more **advanced**, **automated**, and **user-friendly** system that can enhance the efficiency of library operations and provide a better experience to both students and staff.

Proposed System

The proposed **Library Management System** is a **computerized web-based application** designed to overcome the limitations of the existing manual or basic systems. It aims to provide a **fully automated** and **user-friendly platform** that enhances the efficiency of library operations for both administrators and students.

This system will include two main types of users: **Admin** and **Student**. The admin will have full control over the system, including the ability to add/remove books, issue/return books, view transaction history, and manage student records. Students will have their own login accounts to check book availability, view issued books, and see due dates.

Key Features of the Proposed System:

- **Admin Login:** Secure login for administrators to manage all library records and activities.
- **Student Login:** Personalized student accounts to access book-related information.
- **Book Search Module:** Easy and fast book search by title, author, or category.
- **Book Issue/Return Management:** Track issue dates, return dates, and overdue records.
- **Automatic Fine Calculation:** System can calculate fines based on return delays.
- **Report Generation:** Generate reports for issued/returned books, user history, and inventory status.
- **Online Accessibility:** Students and staff can access the system from anywhere (if deployed online).

Advantages Over the Existing System:

- Reduces manual workload and human errors.
- Provides quick and easy access to book information.
- Improves the efficiency and accuracy of library transactions.
- Offers better data management and security.
- Increases transparency and accessibility for students.

The proposed system is intended to **streamline the entire library process**, ensure proper **resource utilization**, and create a **modern, digital library experience**.

Objective

- Improvement in control and performance
- The system is developed to cope up with the current issues and problems of library
- .The system can add user, validate user and is also bug free.
- Save cost
- After computerized system is implemented less human force will be required to maintain the library thus reducing the overall cost.
- Save time
- Librarian is able to search record by using few clicks of mouse and few search keywords thus saving his valuable time.
- Option of online Notice board
- Librarian will be able to provide a detailed description of workshops going in the college as well as in nearby colleges
- Lecture Notes
- Teacher have a facility to upload lectures notes in a pdf file having size not more than 10mb

1.4 System Design

1. NORMAL USER

1.1 USER LOGIN

Description of feature

This feature used by the user to login into system. They are required to enter user id and password before they are allowed to enter the system .The user id and password will be verified and if invalid id is there user is allowed to not enter the system.

Functional requirements

-user id is provided when they register

-The system must only allow user with valid id and password to enter the system

-The system performs authorization process which decides what user level can access to.

-The user must be able to logout after they finished using system.

1.2 REGISTER NEW USER

Description of feature

This feature can be performed by all users to register new user to create account.

Functional requirements

-System must be able to verify information

-System must be able to delete information if information is wrong

Admin Features

- Admin Dashboard
- Admin can add/update/ delete category
- Admin can add/update/ delete author
- Admin can add/update/ delete books
- Admin can issue a new book to student and also update the details when student return book
- Admin can search student by using their student ID
- Admin can also view student details
- Admin can change own password

Students-

- Student can register yourself and after registration they will get studentid
- After login student can view own dashboard.
- Student can update own profile.
- Student can view issued book and book return date-time.
- Student can also view the available books in the library.
- Student can also change own password.
- Student can also recover own password.

1.5 System Requirements

Software Requirements:

- Technology: Python Django
- IDE : Pycharm/Atom
- Client Side Technologies: HTML, CSS, JavaScript , Bootstrap
- Server Side Technologies: Python
- Data Base Server: Sqlite
- Operating System: Microsoft Windows/Linux

Hardware Requirements:

- Processor: Pentium-III (or) Higher
- Ram: 64MB (or) Higher
- Hard disk: 80GB (or) Higher

Software Features

1. Django Framework Features

The backbone of the system is the Django framework, which offers a rich set of features to facilitate rapid, secure, and scalable web application development.

- **Rapid Development & DRY Principle:**
  Django follows the "Don't Repeat Yourself" (DRY) principle, which reduces redundancy by enabling developers to write reusable code. The framework's modular structure allows fast development cycles, enabling quick implementation of new features and easy maintenance.
- **Model-View-Template (MVT) Architecture:**
  Django's MVT pattern separates data models, user interface, and business logic, improving code clarity and manageability.
  - *Models* handle database structure and data representation.
  - *Views* control application logic and respond to user requests.
  - *Templates* render HTML content dynamically, allowing for clean UI separation.
- **Built-in Admin Interface:**
  Django automatically generates an admin dashboard for managing database content. This interface allows administrators to perform CRUD operations on users, books, authors, and categories without additional coding.
- **Robust Authentication and Authorization:**
  Django provides built-in authentication features including user registration, login/logout, password hashing, password reset, and session management. Role-based permissions ensure that only authorized users (admins vs. students) can access or modify specific data.
- **Object-Relational Mapping (ORM):**
  The Django ORM abstracts SQL queries into Python code. This allows seamless interaction with the MySQL database without writing complex SQL queries manually, simplifying database operations such as joins, filters, and transactions.
- **Form Handling and Validation:**
  Django provides powerful form handling with automatic rendering and validation of input data. This feature prevents invalid or malicious data entry, enhancing system security and user experience.
- **Security Features:**
  Django incorporates protection against common web vulnerabilities, such as:
  - *Cross-Site Request Forgery (CSRF)* protection using tokens on forms.
  - *Cross-Site Scripting (XSS)* mitigation via auto-escaping of HTML in templates.
  - *SQL Injection* prevention by using parameterized queries through the ORM.
  - Secure password storage using salted hashing algorithms.

- **URL Routing and Middleware Support:**
  Flexible URL routing allows clean, human-readable URLs improving SEO and user-friendliness. Middleware components facilitate handling requests/responses for logging, authentication checks, and error handling.

2. MySQL Database Features

MySQL serves as the relational database management system (RDBMS) for storing and managing all persistent data in the system.

- **Structured Data Storage:**
  Data entities such as books, authors, categories, students, and transactions are stored in normalized tables to eliminate redundancy and improve data integrity.
- **ACID Compliance:**
  MySQL ensures Atomicity, Consistency, Isolation, and Durability in transactions, maintaining accurate data even during concurrent operations or failures.
- **Efficient Querying and Indexing:**
  Complex queries, including joins across multiple tables, filters, and sorts, are optimized for speed with indexing on frequently accessed columns like student IDs and book ISBNs.
- **Scalability:**
  MySQL supports growth in data size and user concurrency, allowing the library system to serve increasing numbers of students and expanding book collections efficiently.
- **Integration with Django ORM:**
  Django's ORM layer automatically converts Python model queries into MySQL commands, providing seamless database interaction without direct SQL programming.

3. Frontend Technologies

The frontend ensures a user-friendly and interactive experience for both students and administrators.

- **HTML5 & CSS3:**
  Structure and style the web pages to provide a clean, intuitive interface that is easy to navigate and visually appealing.
- **JavaScript & jQuery:**
  Add dynamic behaviors like interactive forms, dropdown menus, modals, and validation messages without requiring full page reloads.
- **AJAX (Asynchronous JavaScript and XML):**
  Enables asynchronous data fetching, allowing parts of the page (like book search results or student info) to update dynamically without refreshing the entire page. This improves responsiveness and user experience.

- **Responsive Design:**
  Use of CSS media queries and flexible layouts to ensure the application works well on different screen sizes, including desktops, tablets, and mobile devices.
- **Cross-Browser Compatibility:**
  Ensures the system functions smoothly across modern browsers such as Google Chrome, Mozilla Firefox, Opera, and Internet Explorer
- Security settings to ensure account protection.

5. Additional Software Features

- **Real-time Notifications (Optional):**
  Automated email or system alerts can remind students of upcoming return dates or overdue books.
- **Logging and Audit Trails:**
  The system logs user actions, such as book issues and returns, providing accountability and traceability.
- **Error Handling:**
  Friendly error pages and messages guide users when something goes wrong, improving usability.
- **Backup and Restore:**
  Support for regular backups of the database to prevent data loss and enable quick recovery.

Analysis and Design

Feasibility Study

A preliminary investigation examines project feasibility, and the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational, and Economical feasibility of adding new modules and debugging old running systems. All system is feasible if they are unlimited resources and infinite time. The feasibility study is a management-oriented activity. The objective of a feasibility study is to find out if an information system project can be done and to suggest possible alternative solutions.

There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operational Feasibility
- Economical Feasibility

Technical Feasibility

It refers to whether the software that is available in the market fully supports the present application. It studies the pros and cons of using a particular software for the development and its feasibility. It also studies the additional training needed to be given to the people to make the application work. The technical requirements are then compared to the technical capability of the organization. The systems project is considered technically feasible if the internal technical capability is sufficient to support the project requirements. The analyst must find out whether current technical resources can be upgraded or added to in a manner that fulfills the request under consideration.

Operational Feasibility

It refers to the feasibility of the product to be operational. Some products may work very well in design and implementation but may fail in the real-time environment. It includes the study of additional human resources required and their Technical expertise. It is dependent on human resources available for the project and involves projecting whether the system will be used if it is developed and implemented. It measures how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the analysis phase of system development. It reviews the willingness of the organization to support the proposed system. In order to determine this feasibility, it is important to understand the management’s commitment to the proposed project.

Economic Feasibility

It refers to the benefits or outcomes we are deriving from the product as compared to the total cost we are spending for developing the product. It the more or less same as the older system, then it is not feasible to develop the product. Economic analysis could also be referred to as cost/benefit analysis. It is the most frequently used method for evaluating the effectiveness of a new system. In economic analysis the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. An entrepreneur must accurately weigh the cost versus benefits before takin

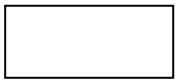
1.6 ERD, DFD

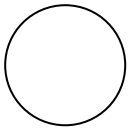
DATA FLOW DIAGRAMS

A DFD does not show a sequence of steps. A DFD only shows what the different process in a system is and what data flows between them.

The following are some DFD symbols used in the project

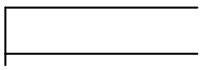
External entities





Process: A transaction of information that resides within the bounds of the system to be module.

DATAFLOWS



DATASTORE: A repository of data that is to be stored for use by one or more processes, may be as simple as buffer of queue or as a relational database

RULES FOR DFD:

- Fix the scope of the system by means of context diagrams.
- Organize the DFD so that the main sequence of the actions reads left to right and top to bottom.
- Identify all inputs and outputs.
- Identify and label each process internal to the system with rounded circles.
- A process is required for all the data transformation and transfers. Therefore, never connect a data store to a data source or the destinations or another data store with just a data flow arrow.
- Do not indicate hardware and ignore control information.

- Make sure the names of the processes accurately convey everything the process is done.
- There must not be unnamed process.
- Indicate external sources and destinations of the data, with squares.
- Number each occurrence of repeated external entities.
- Identify all data flows for each process step, except simple Record retrievals.
- Label data flow on each arrow.
- Use details flow on each arrow.
- Use the details flow arrow to indicate data movements.
- There can't be unnamed data flow.
- A data flow can't connect two external entities.

LEVELS OF DFD:

The complexity of the system means that it is a responsible to represent the operations of any system of single data flow diagram. At the top level, an Overview of the different systems in an organization is shown by the way of context analysis diagram. When exploded into DFD

They are represented by:

- LEVEL-0 : SYSTEM INPUT/OUTPUT
- LEVEL-1:SUBSYSTEM LEVEL DATAFLOW FUNCTIONAL
- LEVEL-2: FILE LEVEL DETAIL DATA FLOW.

The input and output data shown should be consistent from one level to the next.

LEVEL-0: SYSTEM INPUT/OUTPUT LEVEL

A level-0 DFD describes the system-wide boundaries, dealing inputs to and outputs from the system and major processes. This diagram is similar to the combined user-level context diagram.

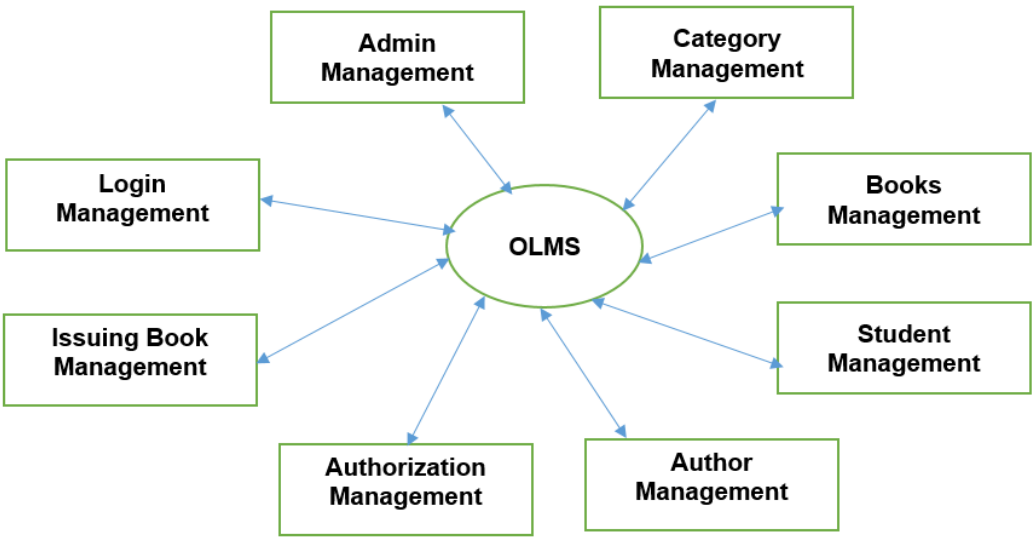
LEVEL-1: SUBSYSTEM LEVEL DATA FLOW

A level-1 DFD describes the next level of details within the system, detailing the data flows between subsystems, which makeup the whole.

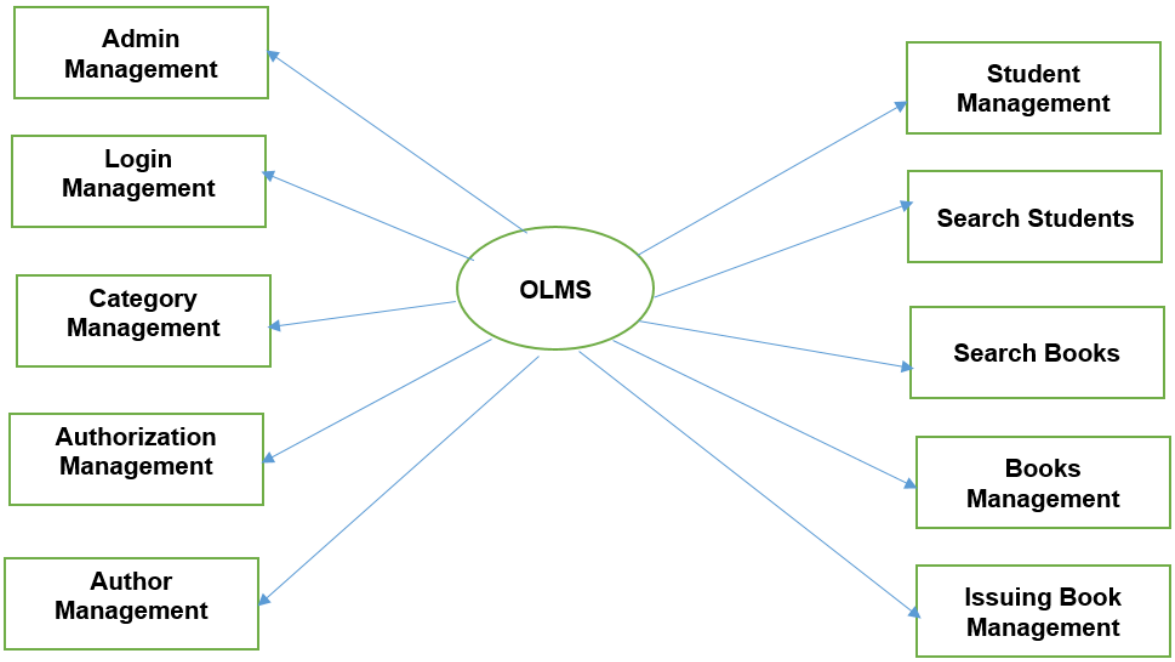
LEVEL-2: FILE LEVEL DETAIL DATA FLOW

All the projects are feasible given unlimited resources and infinite time. It is both necessary and prudent to evaluate the feasibility of the project at the earliest possible time. Feasibility and the risk analysis are pertained in many ways. If project risk is great.

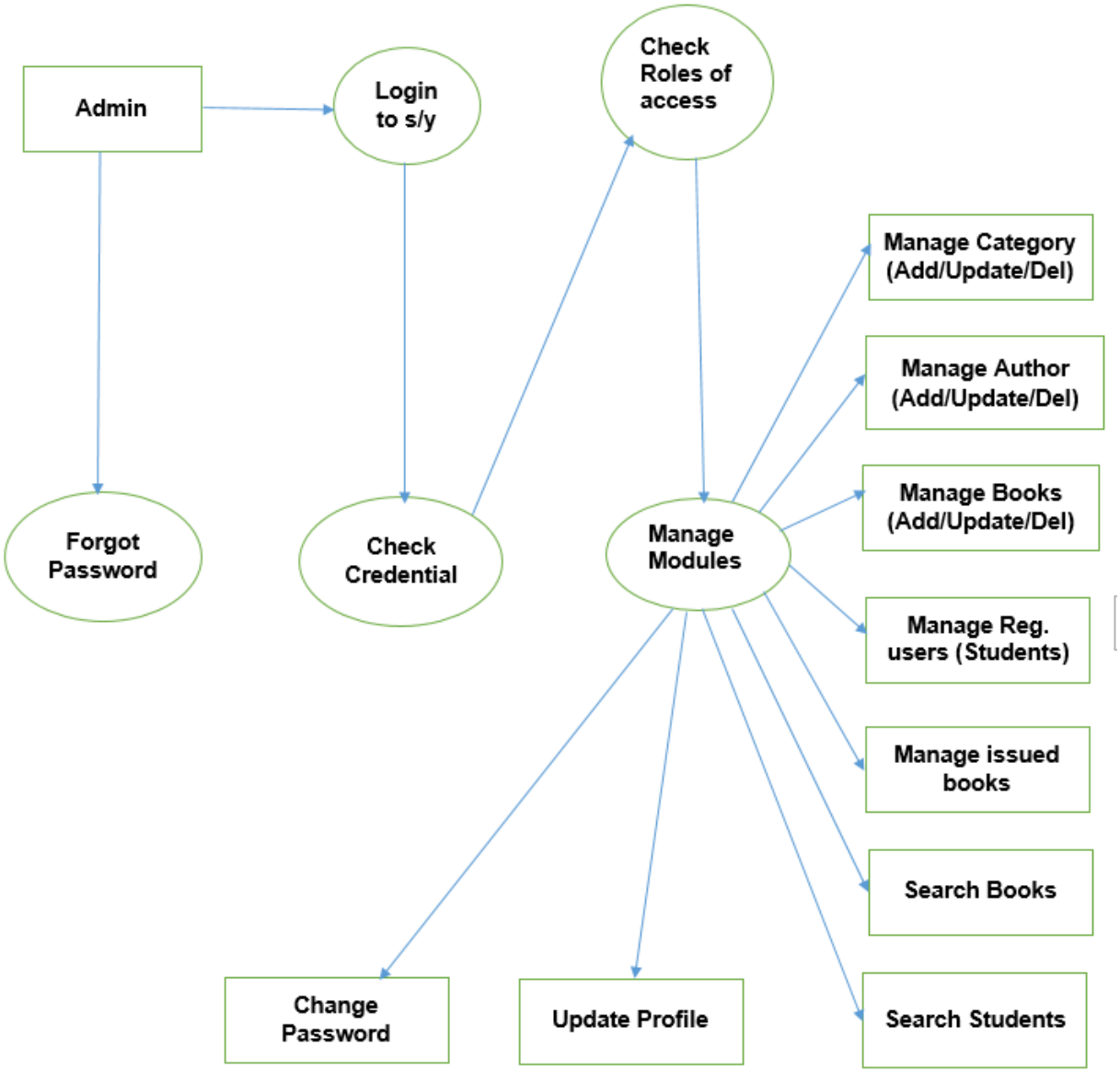
Zero Level DFD

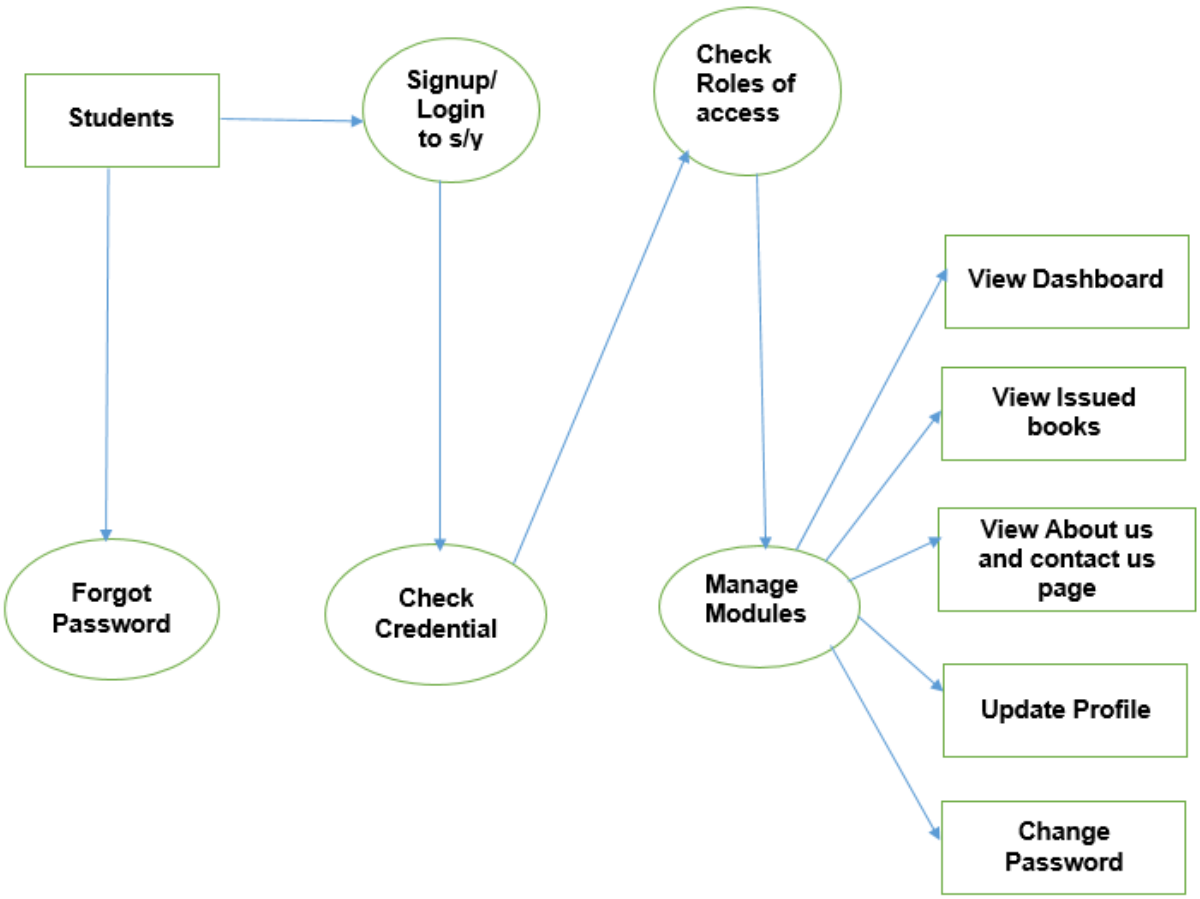


First Level DFD



Second Level DFD




Students
Signup/
Login
to s/y
Check
Roles of
access
View Dashboard
View Issued
books
View About us
and contact us
page
Forgot
Password
Check
Credential
Manage
Modules
Update Profile
Change
Password

ER Diagram

An **Entity-Relationship (E-R) Diagram** is a **graphical representation** used in database design to illustrate the relationships between **entities** (objects, people, or concepts) and **attributes** (properties or details) within a system.

It is a key step in the **conceptual design phase** of database development and is used by database designers to **communicate** the structure of the database with stakeholders, programmers, and system architects.

What is an Entity?

An **entity** is anything about which we want to store data. For example:

- In a college database: **Student**, **Course**, and **Faculty** are entities.

What is a Relationship?

A **relationship** describes how two or more entities are related. For example:

- A **Student** "enrolls" in a **Course**
- A **Faculty** "teaches" a **Course**

What is an Attribute?

An **attribute** is a piece of information that **describes an entity**. For example:

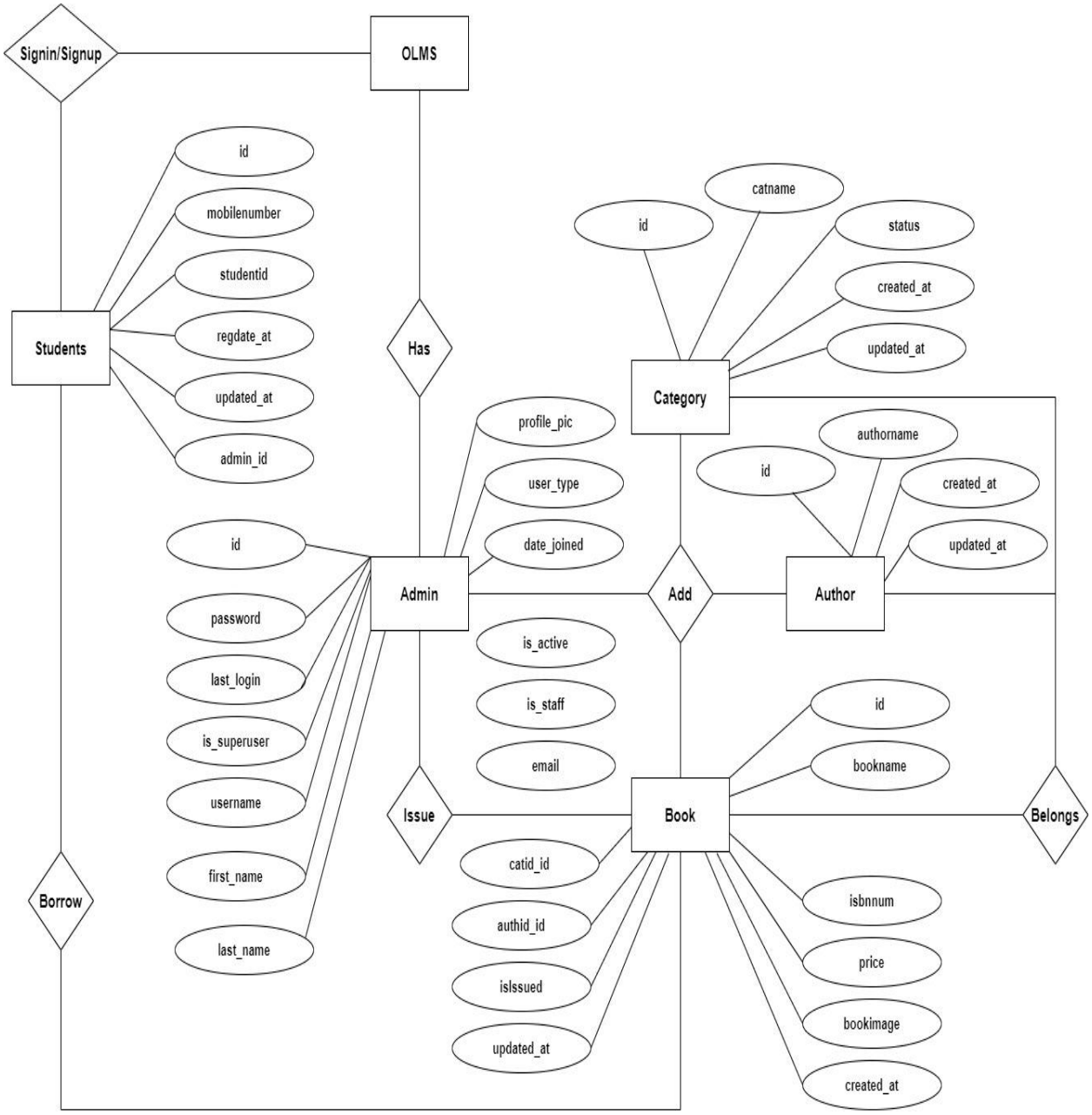
- A **Student** entity may have attributes like: `Student_ID`, `Name`, `Email`.

2. Symbols Used in E-R Diagrams

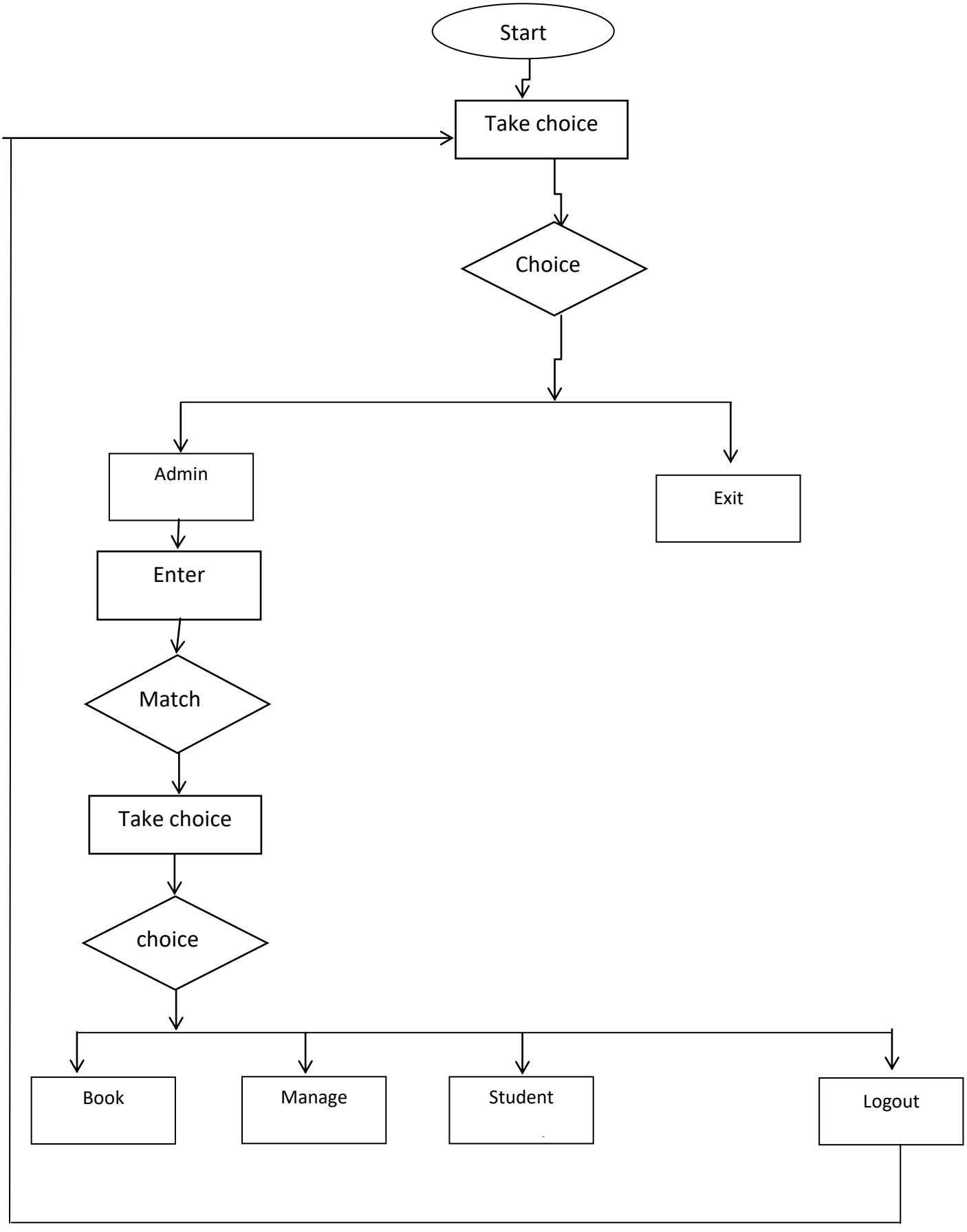
E-R Diagrams use a **standardized set of symbols** to represent entities, relationships, and attributes. Here's a detailed explanation of each:

Symbol	Representation	Purpose
□ Rectangle	**Entity**	Represents an object or concept in the system
□ Oval	**Attribute**	Describes properties of an entity or relation
□ Diamond	**Relationship**	Describes how entities interact
□ Line	**Connector**	Connects entities to relationships or attributes
□ Double Rectangle	**Weak Entity**	Entity that depends on another entity
□ Multivalued Oval	**Multivalued Attribute**	Attributes that can have multiple values
□ Underlined Oval	**Key Attribute**	Uniquely identifies an entity
□ Dashed Oval	**Derived Attribute**	Computed from other attributes

Entity Relationship Diagram



Testing


Start
Take choice
Choice
Admin
Exit
Enter
Match
Take choice
choice
Book
Manage
Student
Logout

1.7 System Plannning

USECASE DIAGRAMS:

Use case diagrams model behavior within a system and helps the developers understand of what the user require. The stick man represents what's called an actor.

Use case diagram can be useful for getting an overall view of the system and clarifying who can do and more importantly what they can't do.

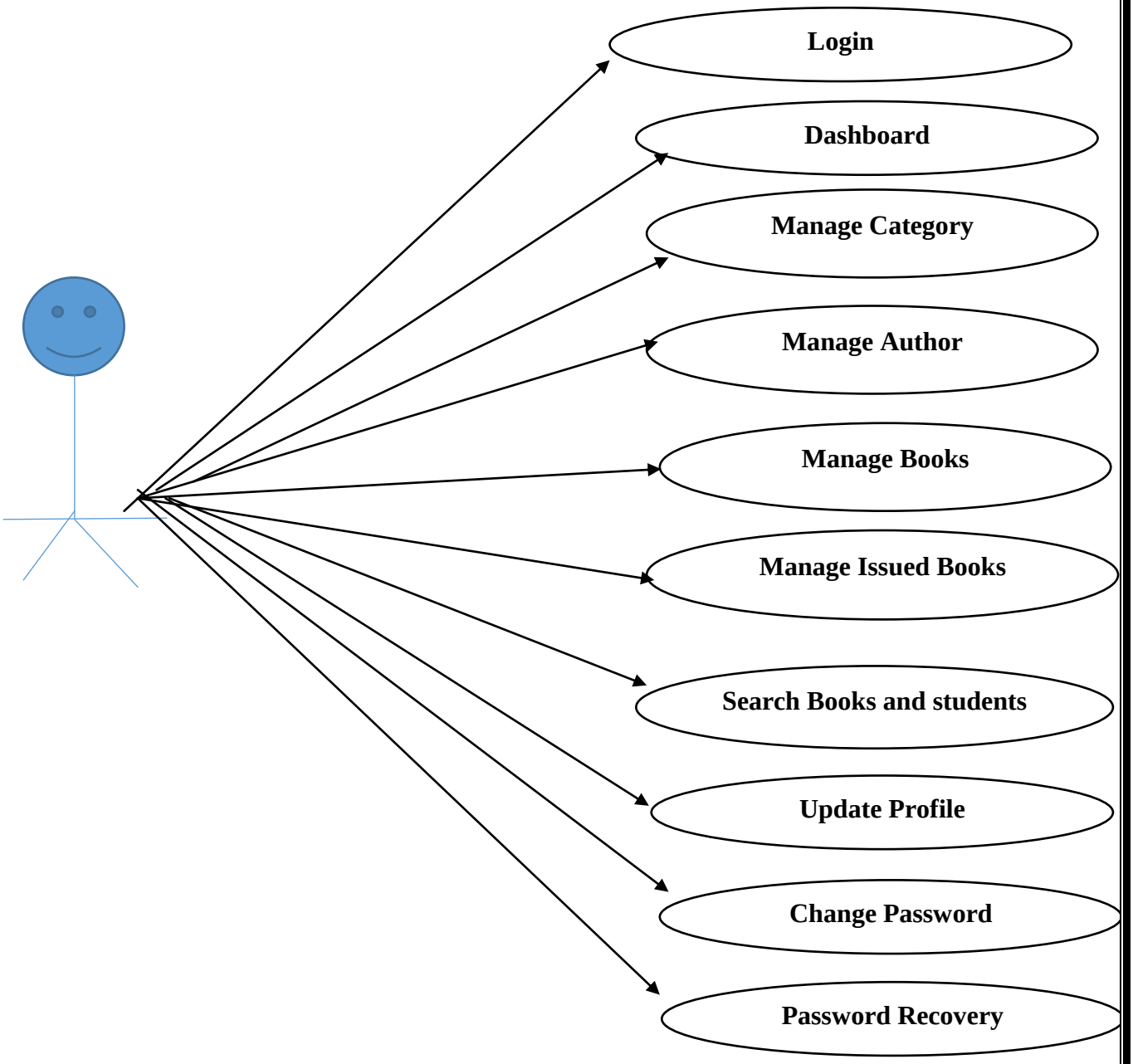
Use case diagram consists of use cases and actors and shows the interaction between the use case and actors.

- The purpose is to show the interactions between the use case and actor.
- To represent the system requirements from user's perspective.
- An actor could be the end-user of the system or an external system.

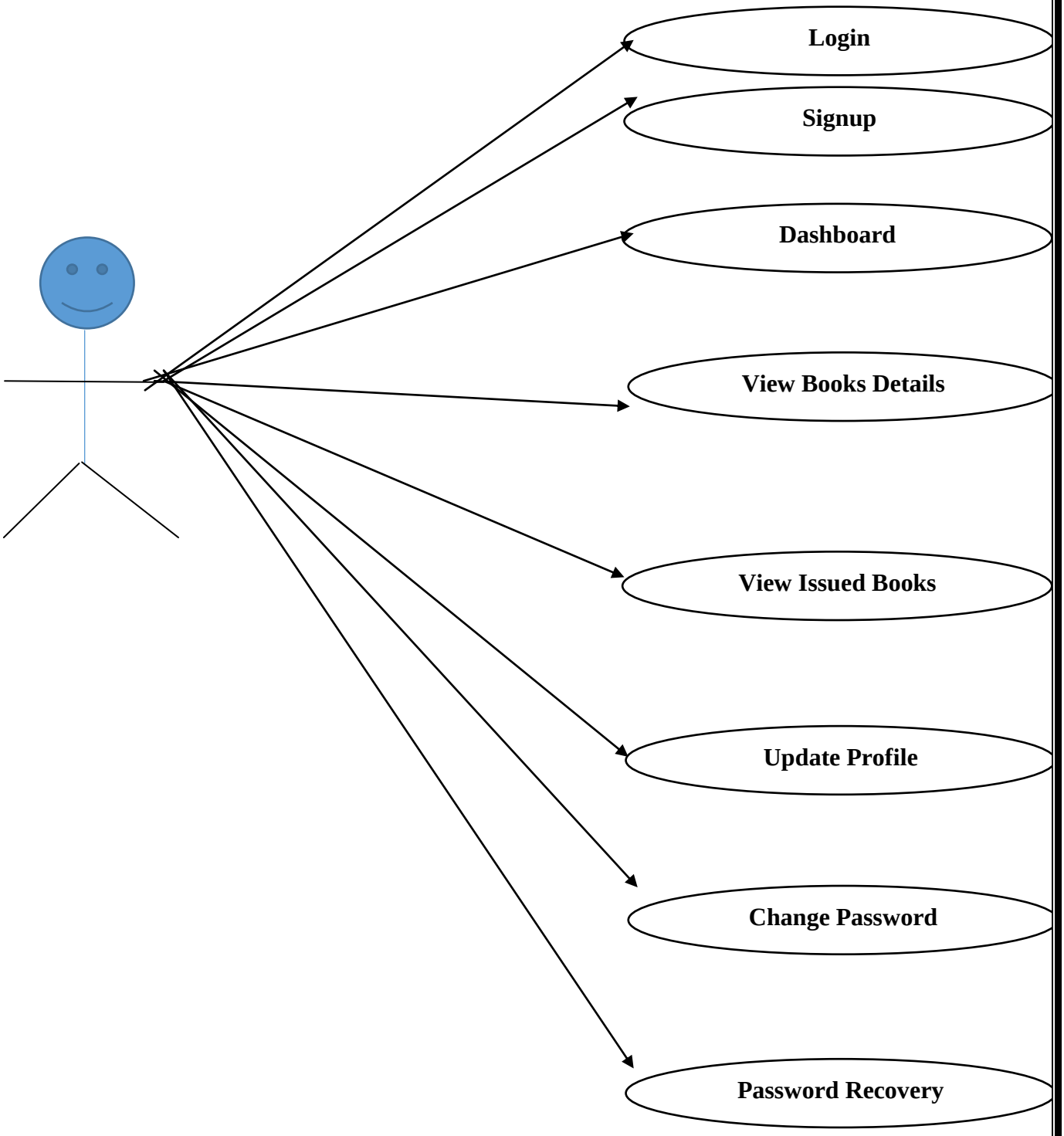
USECASE DIAGRAM:A Use case is a description of set of sequence of actions. Graphically it is rendered as an ellipse with solid line including only its name. Use case diagram is a behavioral diagram that shows a set of use cases and actors and their relationship. It is an association between the use cases and actors. An actor represents a real-world object. Primary Actor – Sender, Secondary Actor Receiver.

Use Case Diagrams:

Admin



User



Relationship between tables (Class Diagram)

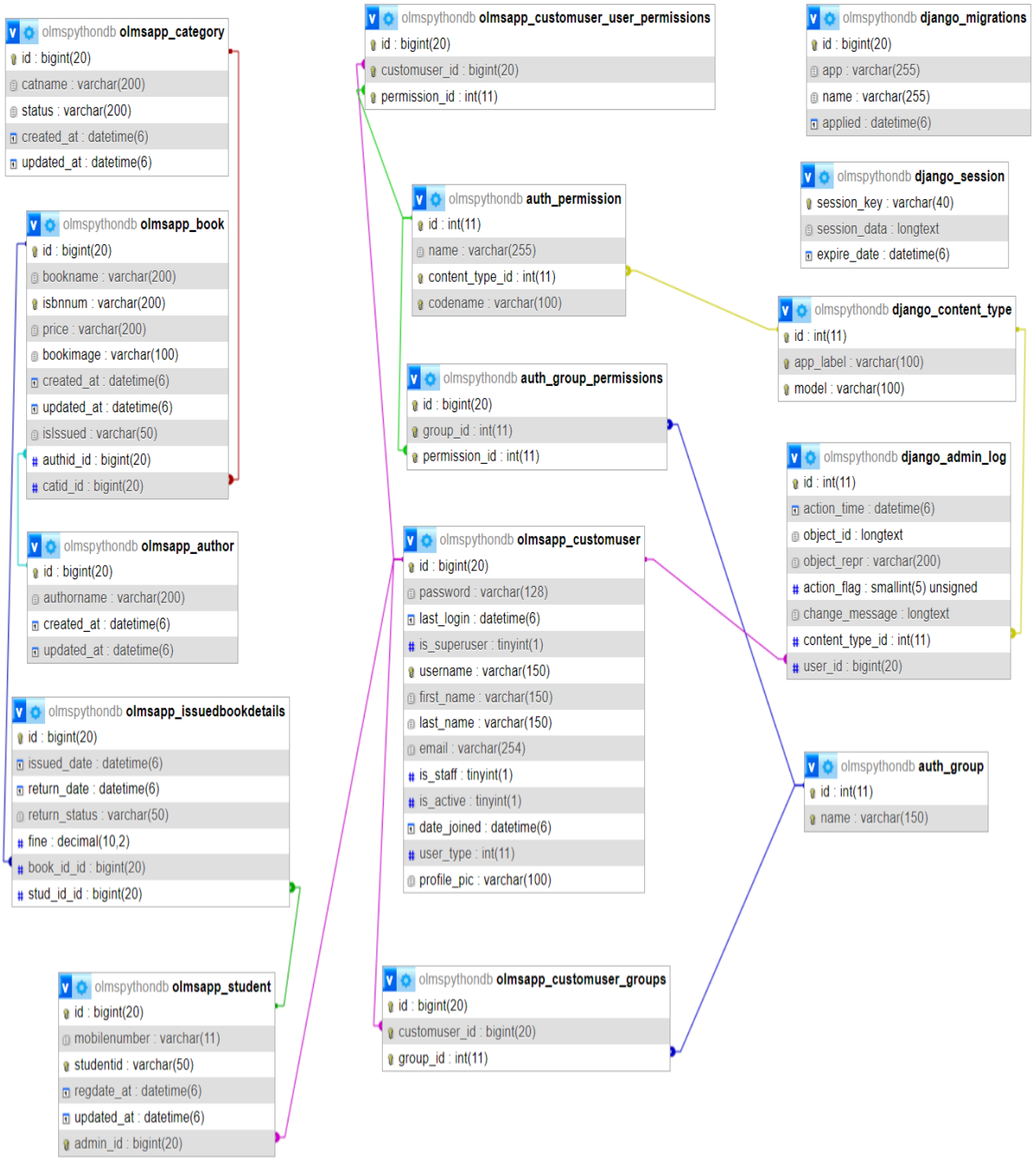


TABLE DESIGN

Online Library Management System" contains 15 MySQL tables(In this MySQL 6 table is customized and 9 tables made by default in Django)

olmsapp_customuser: This table store personal and login details of admin and students.

olmsapp_customuser

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	None		AUTO_INCREMENT
2	password	varchar(128)	utf8mb4_general_ci		No	None		
3	last_login	datetime(6)			Yes	NULL		
4	is_superuser	tinyint(1)			No	None		
5	username	varchar(150)	utf8mb4_general_ci		No	None		
6	first_name	varchar(150)	utf8mb4_general_ci		No	None		
7	last_name	varchar(150)	utf8mb4_general_ci		No	None		
8	email	varchar(254)	utf8mb4_general_ci		No	None		
9	is_staff	tinyint(1)			No	None		
10	is_active	tinyint(1)			No	None		
11	date_joined	datetime(6)			No	None		
12	user_type	int(11)			No	None		
13	profile_pic	varchar(100)	utf8mb4_general_ci		Yes	NULL		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	6	A	No	
username	BTREE	Yes	No	username	6	A	No	

olmsapp_author: This table track the details of authors.

olmsapp_author

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	None		AUTO_INCREMENT
2	authorname	varchar(200)	utf8mb4_general_ci		No	None		
3	created_at	datetime(6)			No	None		
4	updated_at	datetime(6)			No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	13	A	No	

olmsapp_category: This table track the record of category

olmsapp_category

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	None		AUTO_INCREMENT
2	catname	varchar(200)	utf8mb4_general_ci		No	None		
3	status	varchar(200)	utf8mb4_general_ci		No	None		
4	created_at	datetime(6)			No	None		
5	updated_at	datetime(6)			No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	8	A	No	

olmsapp_book: This table for books records.

olmsapp_book

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	None		AUTO_INCREMENT
2	bookname	varchar(200)	utf8mb4_general_ci		No	None		
3	isbnnum	varchar(200)	utf8mb4_general_ci		No	None		
4	price	varchar(200)	utf8mb4_general_ci		No	None		
5	bookimage	varchar(100)	utf8mb4_general_ci		Yes	NULL		
6	created_at	datetime(6)			No	None		
7	updated_at	datetime(6)			No	None		
8	islssued	varchar(50)	utf8mb4_general_ci		No	None		
9	authid_id	bigint(20)			No	None		
10	catid_id	bigint(20)			No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	10	A	No	
isbnnum	BTREE	Yes	No	isbnnum	10	A	No	
olmsapp_book_authid_id_92576cf6_fk_olmsapp_author_id	BTREE	No	No	authid_id	10	A	No	
olmsapp_book_catid_id_7aa39b2d_fk_olmsapp_category_id	BTREE	No	No	catid_id	10	A	No	

olmsapp_student: This table store student details.

olmsapp_student

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	None		AUTO_INCREMENT
2	mobilenumber	varchar(11)	utf8mb4_general_ci		No	None		
3	studentid	varchar(50)	utf8mb4_general_ci		No	None		
4	regdate_at	datetime(6)			No	None		
5	updated_at	datetime(6)			No	None		
6	admin_id	bigint(20)			No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	5	A	No	
studentid	BTREE	Yes	No	studentid	5	A	No	
admin_id	BTREE	Yes	No	admin_id	5	A	No	

olmsapp_issuedbookdetails: This table for maintain issued book and returned book record.

olmsapp_issuedbookdetails

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	None		AUTO_INCREMENT
2	issued_date	datetime(6)			No	None		
3	return_date	datetime(6)			No	None		
4	return_status	varchar(50)	utf8mb4_general_ci		No	None		
5	fine	decimal(10,2)			No	None		
6	book_id_id	bigint(20)			No	None		
7	stud_id_id	bigint(20)			No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	6	A	No	
olmsapp_issuedbookdetails_book_id_id_756580fa_fk_olmsapp_book_id	BTREE	No	No	book_id_id	6	A	No	
olmsapp_issuedbookde_stud_id_id_ea19652e_fk_olmsapp_s	BTREE	No	No	stud_id_id	6	A	No	

Default Tables Details

authentication_customuser_groups table Structure : This table by default table.

authentication_customuser_groups

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	None		AUTO_INCREMENT
2	customuser_id	bigint(20)			No	None		
3	group_id	int(11)			No	None		

authentication_customuser_user_permissions table Structure : This table by default table.

authentication_customuser_user_permissions

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	None		AUTO_INCREMENT
2	customuser_id	bigint(20)			No	None		
3	permission_id	int(11)			No	None		

auth_group table Structure : This table in Django is part of the Django authentication system and is used to represent groups of users.

auth_group

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		AUTO_INCREMENT
2	name	varchar(150)	utf8mb4_general_ci		No	None		

auth_group_permissions table Structure : This table in Django is a part of the permission system and is used to manage the relationship between user groups and permissions.

auth_group_permissions

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	None		AUTO_INCREMENT
2	group_id	int(11)			No	None		
3	permission_id	int(11)			No	None		

auth_permission table Structure : This table in Django is used to store information about the permissions defined.

auth_permission

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		AUTO_INCREMENT
2	name	varchar(255)	utf8mb4_general_ci		No	None		
3	content_type_id	int(11)			No	None		
4	codename	varchar(100)	utf8mb4_general_ci		No	None		

django_admin_log table Structure : This table is used to store records of actions taken by administrators or users through the Django admin interface.

django_admin_log

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	None		AUTO_INCREMENT
2	action_time	datetime(6)			No	None		
3	object_id	longtext	utf8mb4_general_ci		Yes	NULL		
4	object_repr	varchar(200)	utf8mb4_general_ci		No	None		
5	action_flag	smallint(5)		UNSIGNED	No	None		
6	change_message	longtext	utf8mb4_general_ci		No	None		
7	content_type_id	int(11)			Yes	NULL		
8	user_id	bigint(20)			No	None		

django_content_type table Structure : This table in Django is a system table that is used to store information about each model (database table) in your Django project.

django_content_type

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(11)			No	*None*		AUTO_INCREMENT
2	app_label	varchar(100)	utf8mb4_general_ci		No	*None*		
3	model	varchar(100)	utf8mb4_general_ci		No	*None*		

django_migrations table Structure : This table table is part of the database schema and is used to keep track of which migrations have been applied to the database.

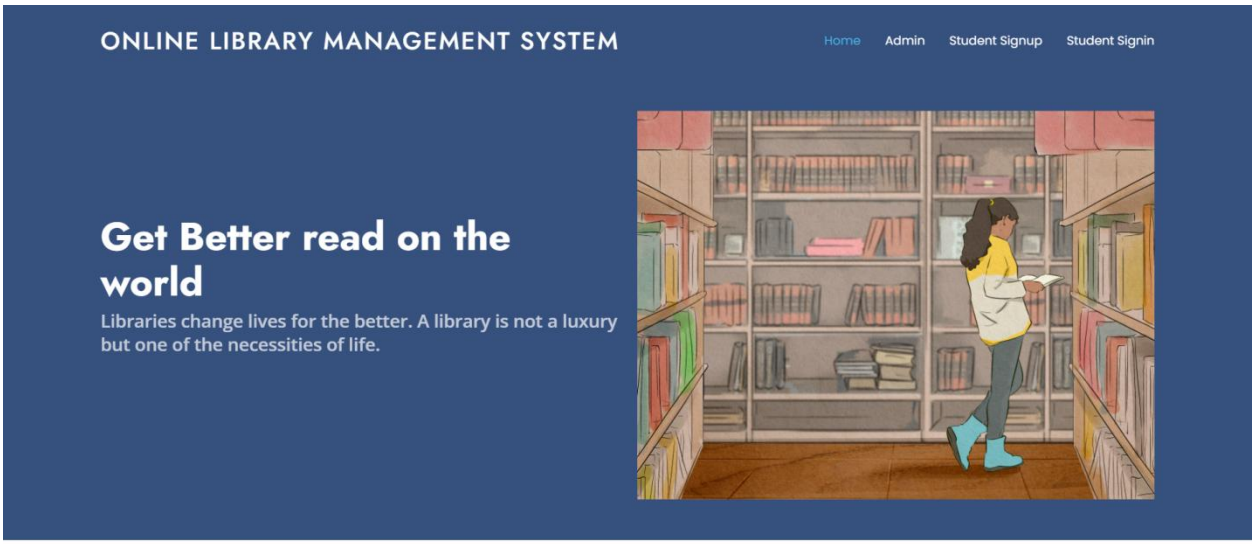
django_migrations

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	bigint(20)			No	*None*		AUTO_INCREMENT
2	app	varchar(255)	utf8mb4_general_ci		No	*None*		
3	name	varchar(255)	utf8mb4_general_ci		No	*None*		
4	applied	datetime(6)			No	*None*		

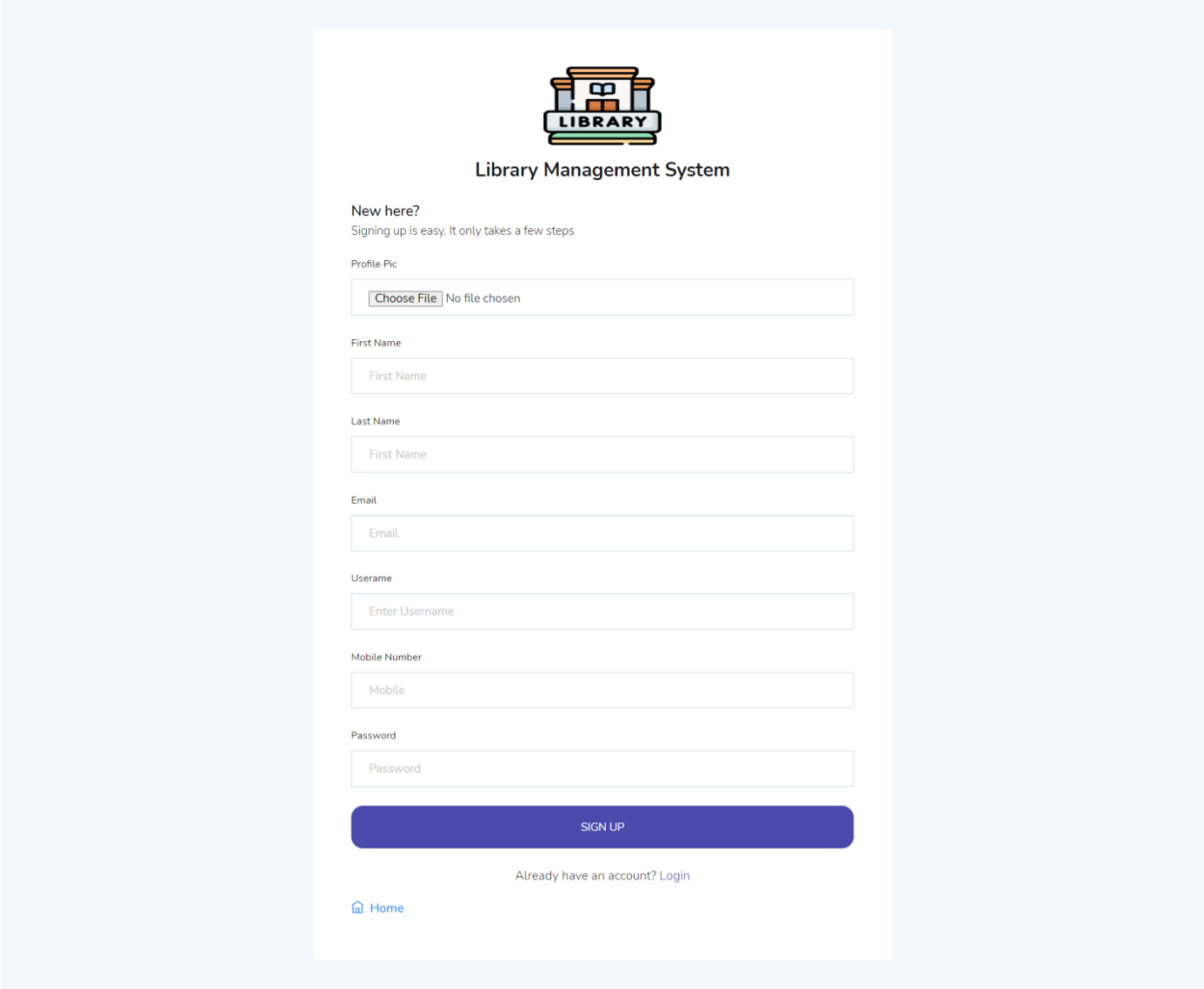
django_session table Structure : This table store the session data for user.

Output Screens

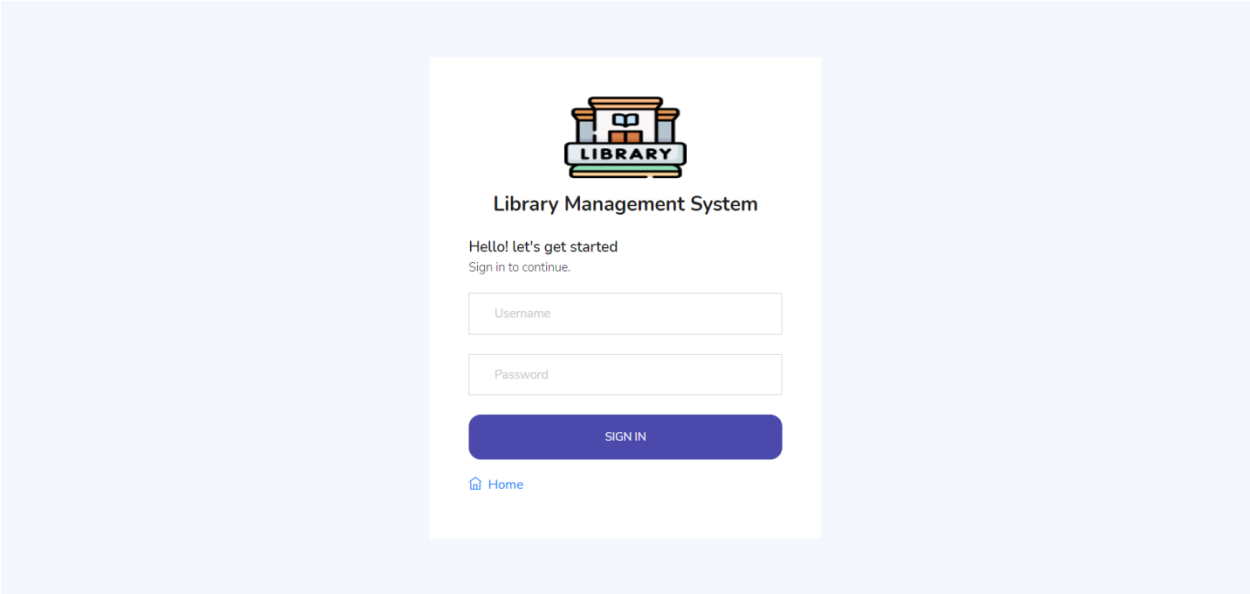
Home Page



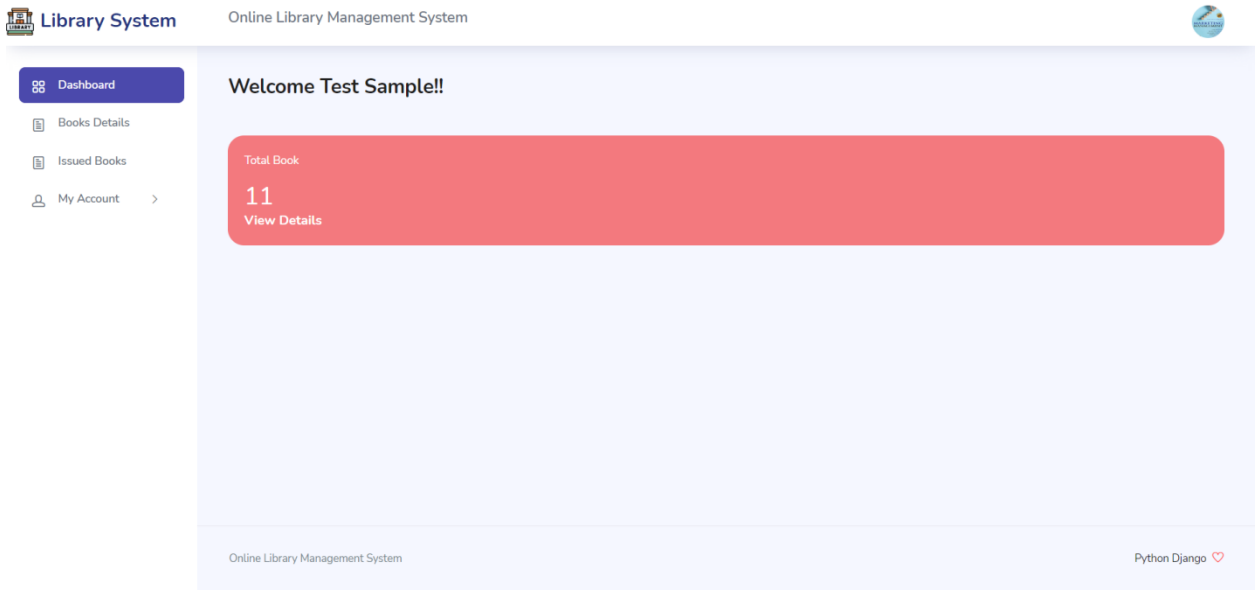
Student Signup



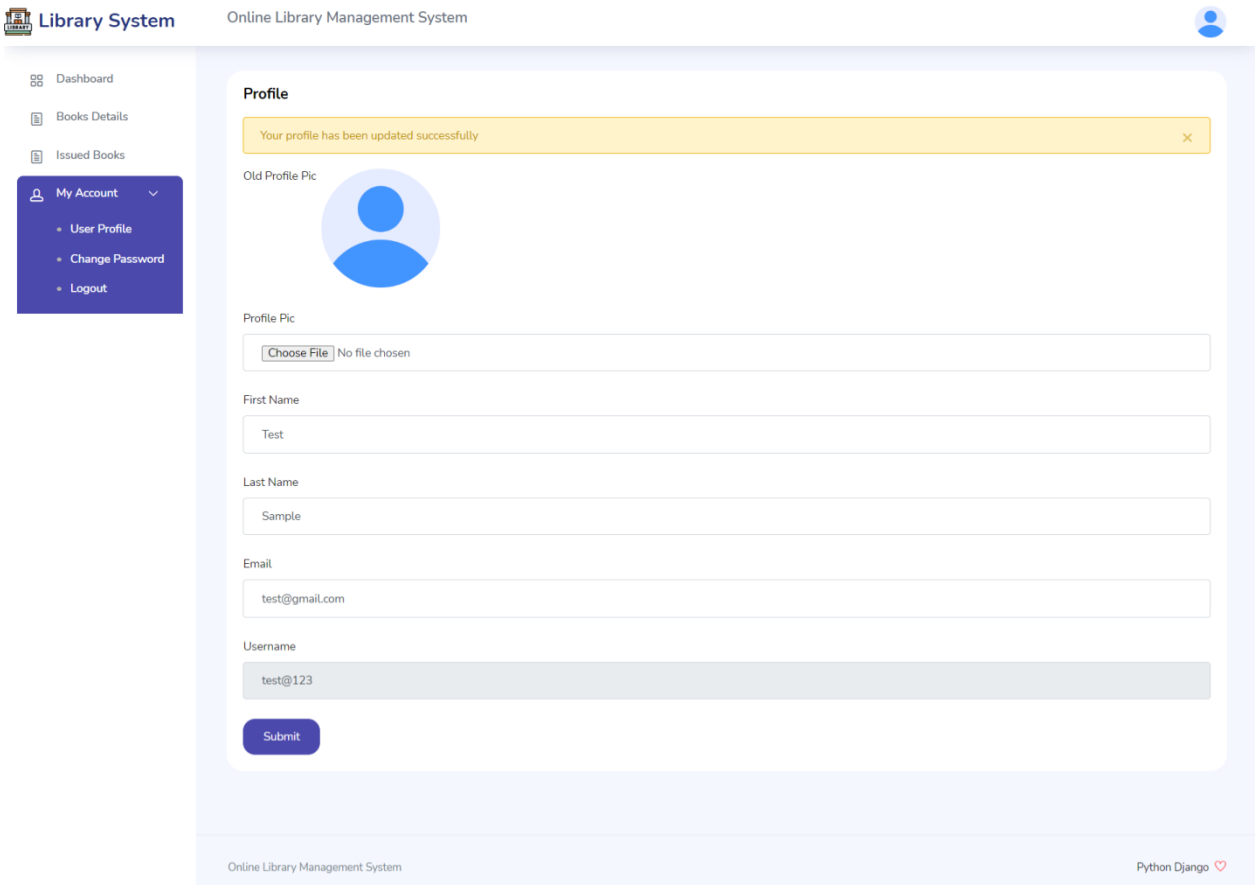
Student Login



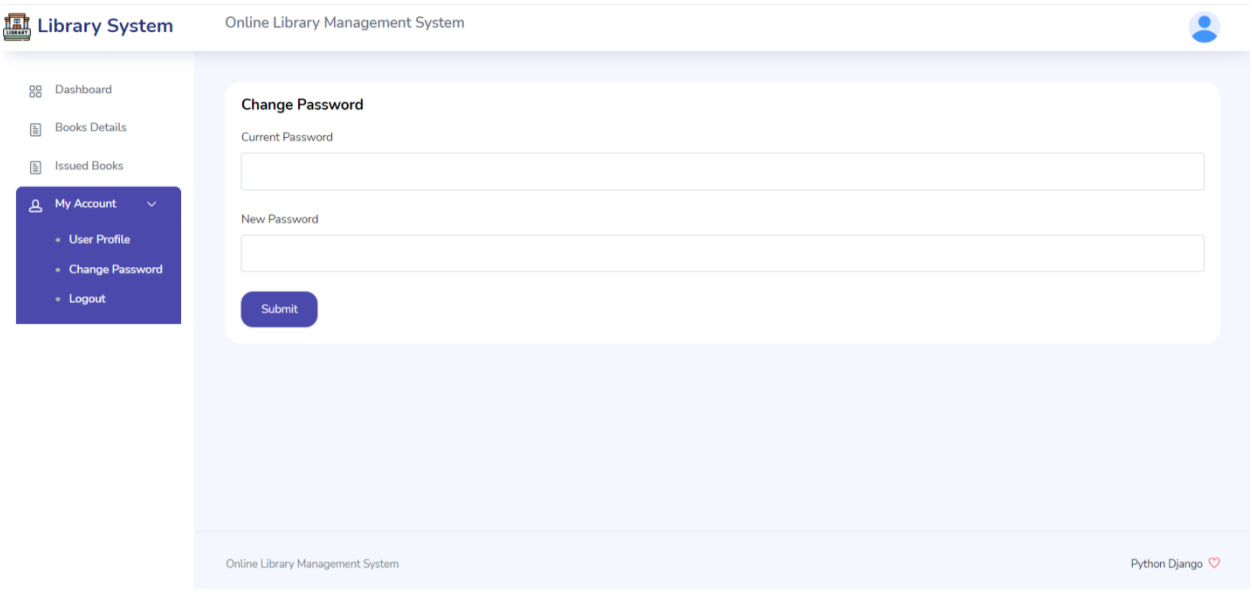
Student Dashboard



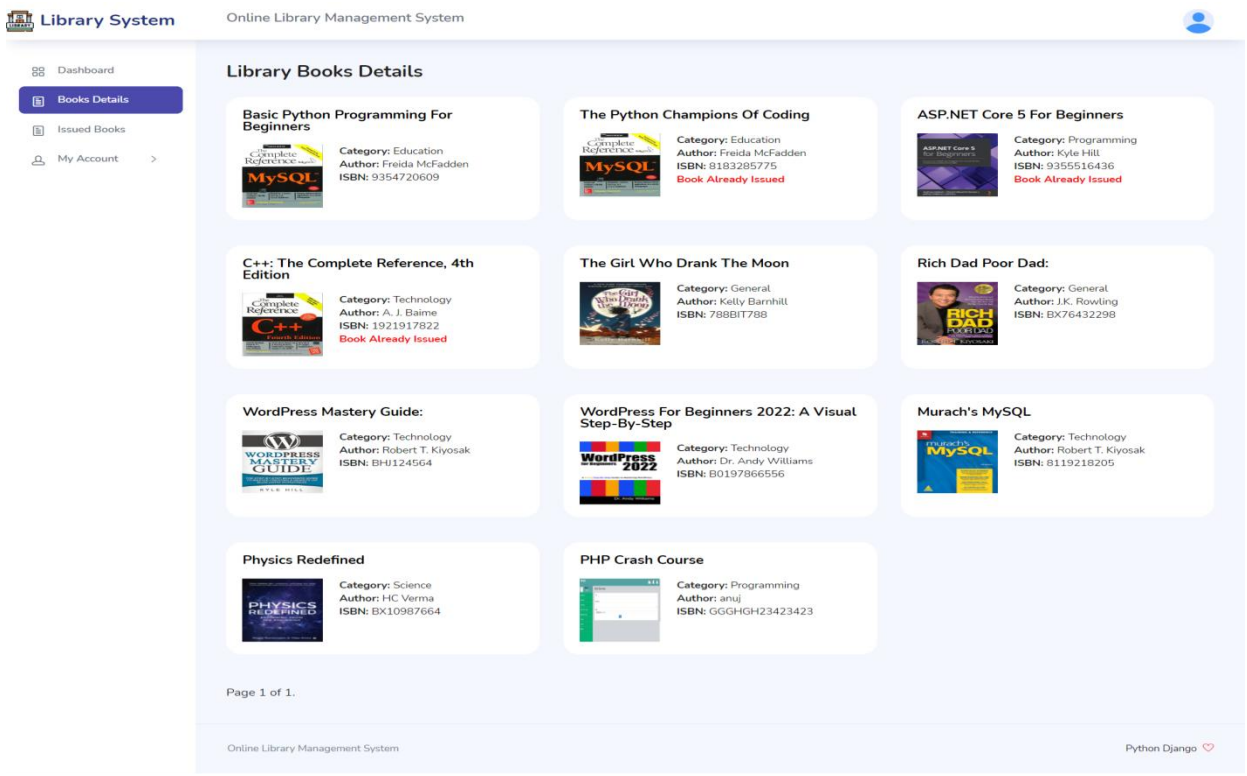
Student Profile



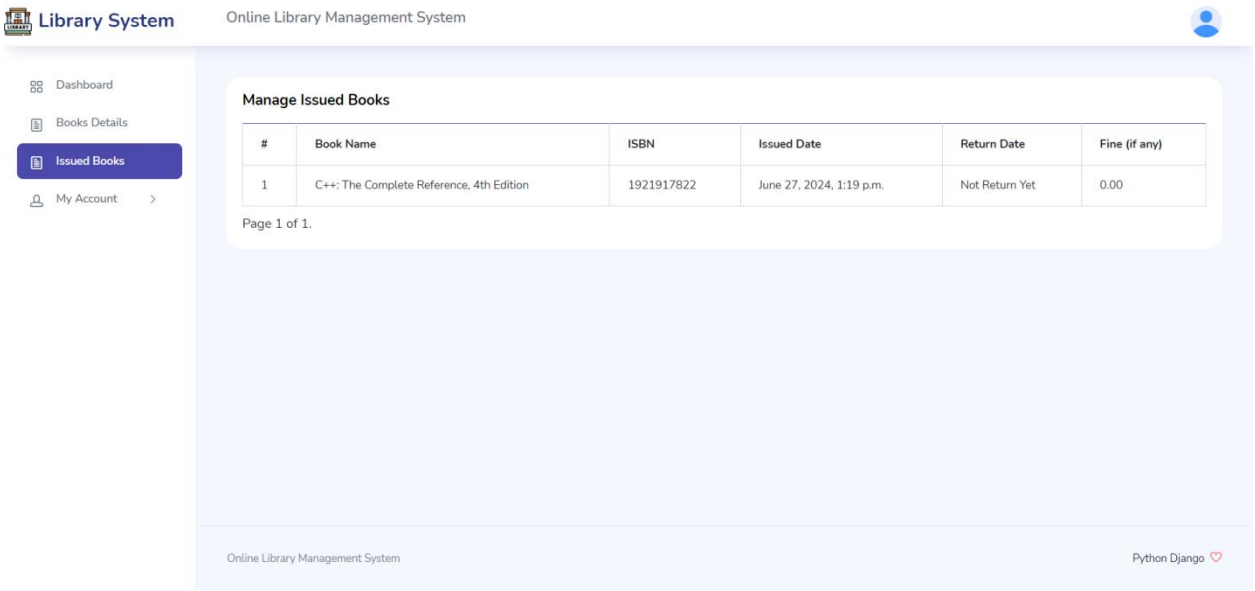
Change Password



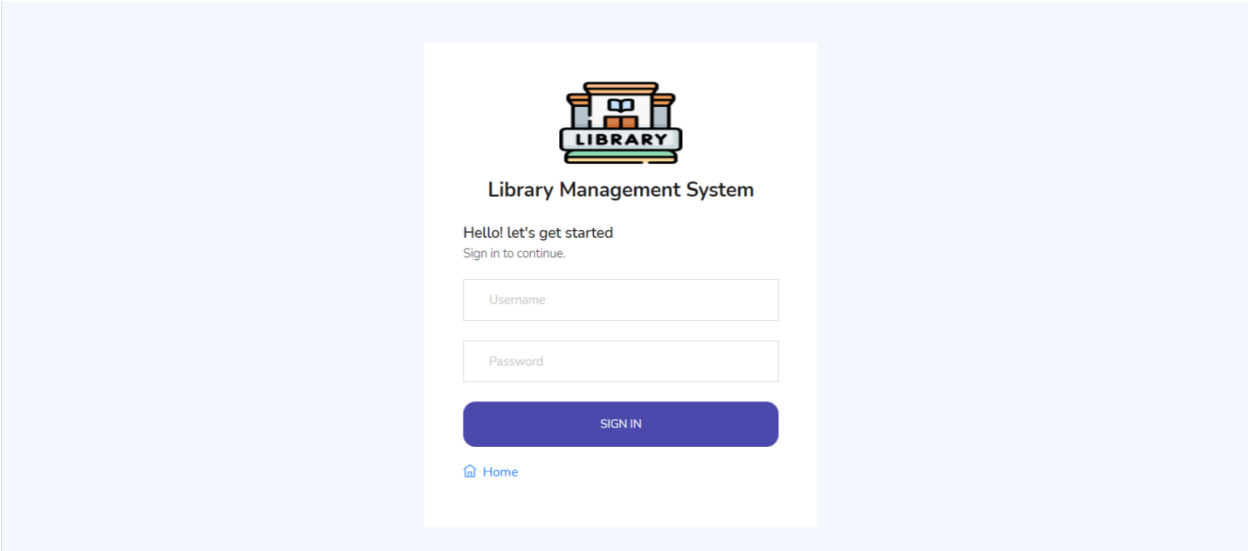
Listed Books



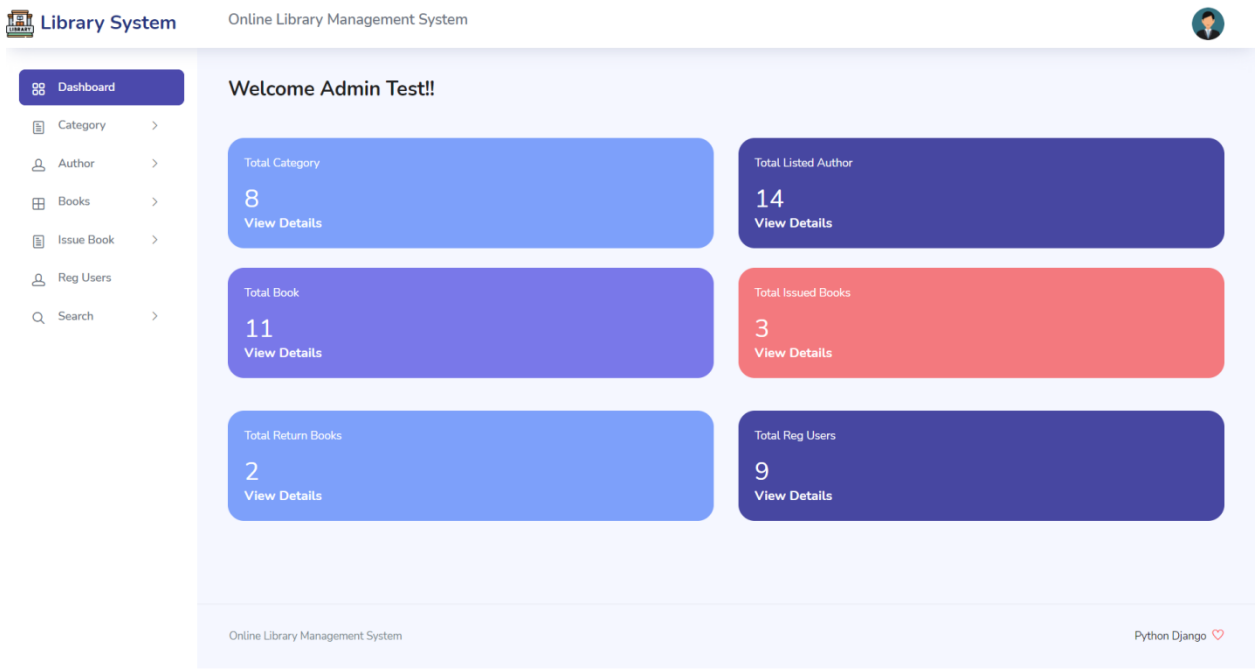
Issued Books



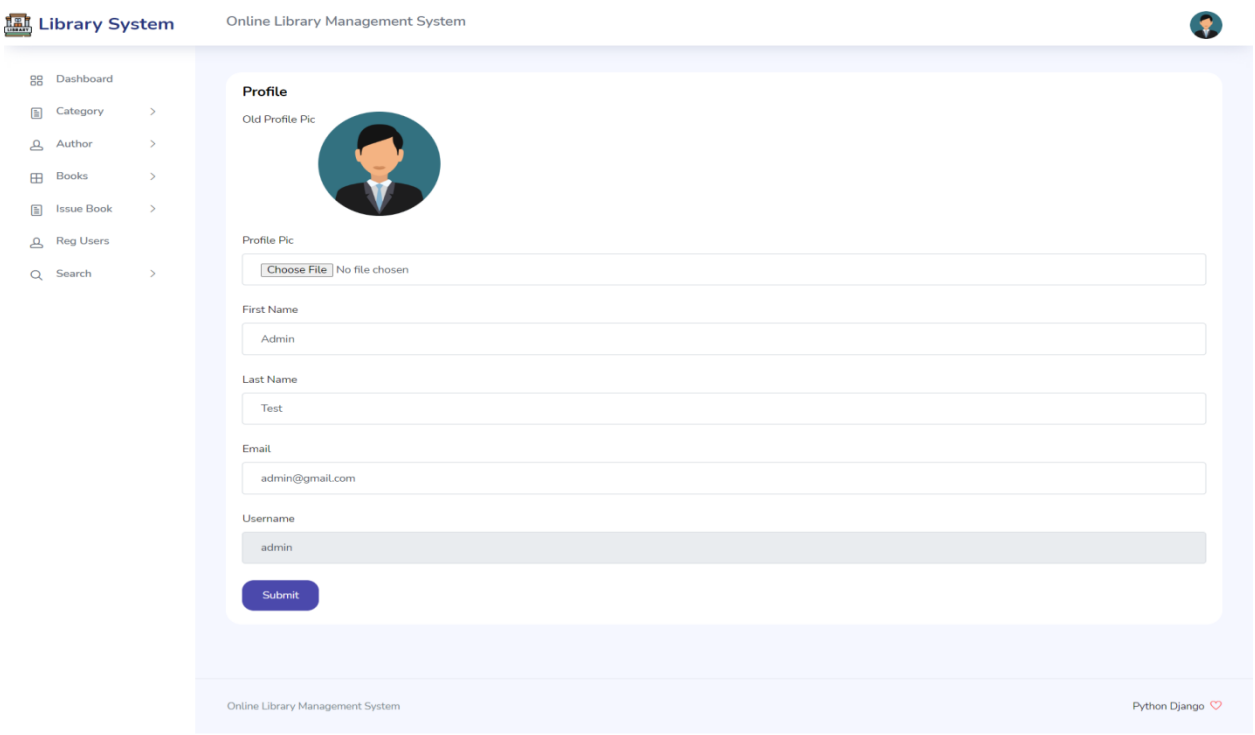
Admin Login


LIBRARY
Library Management System
Hello! let's get started
Sign in to continue.
Username
Password
SIGN IN
Home

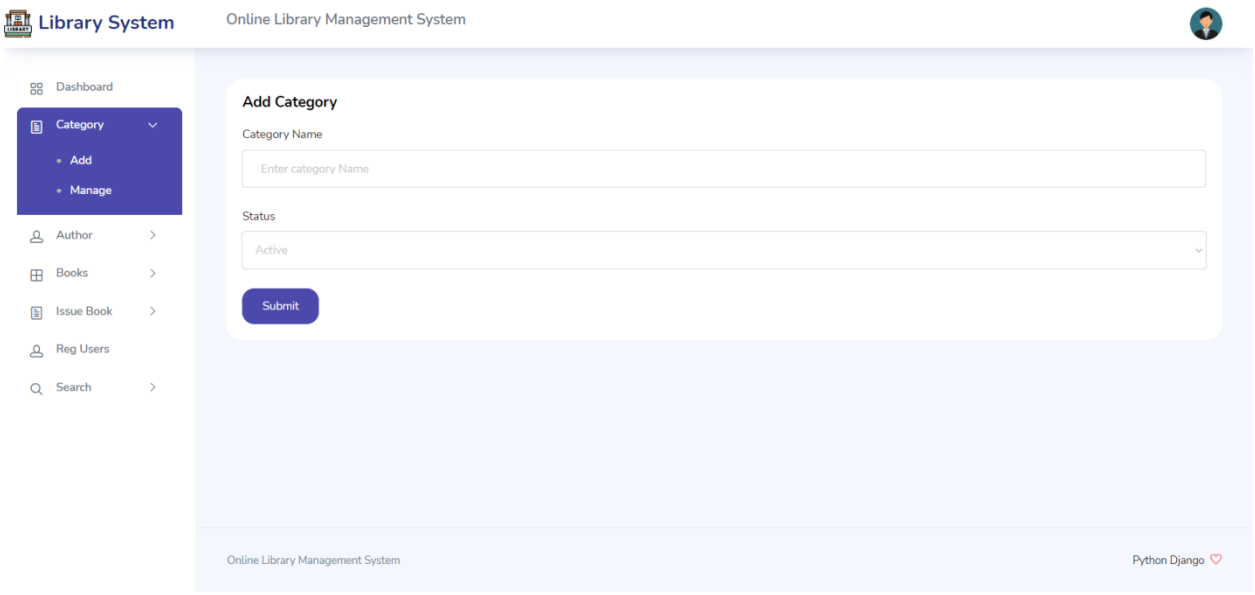
Admin Dashboard



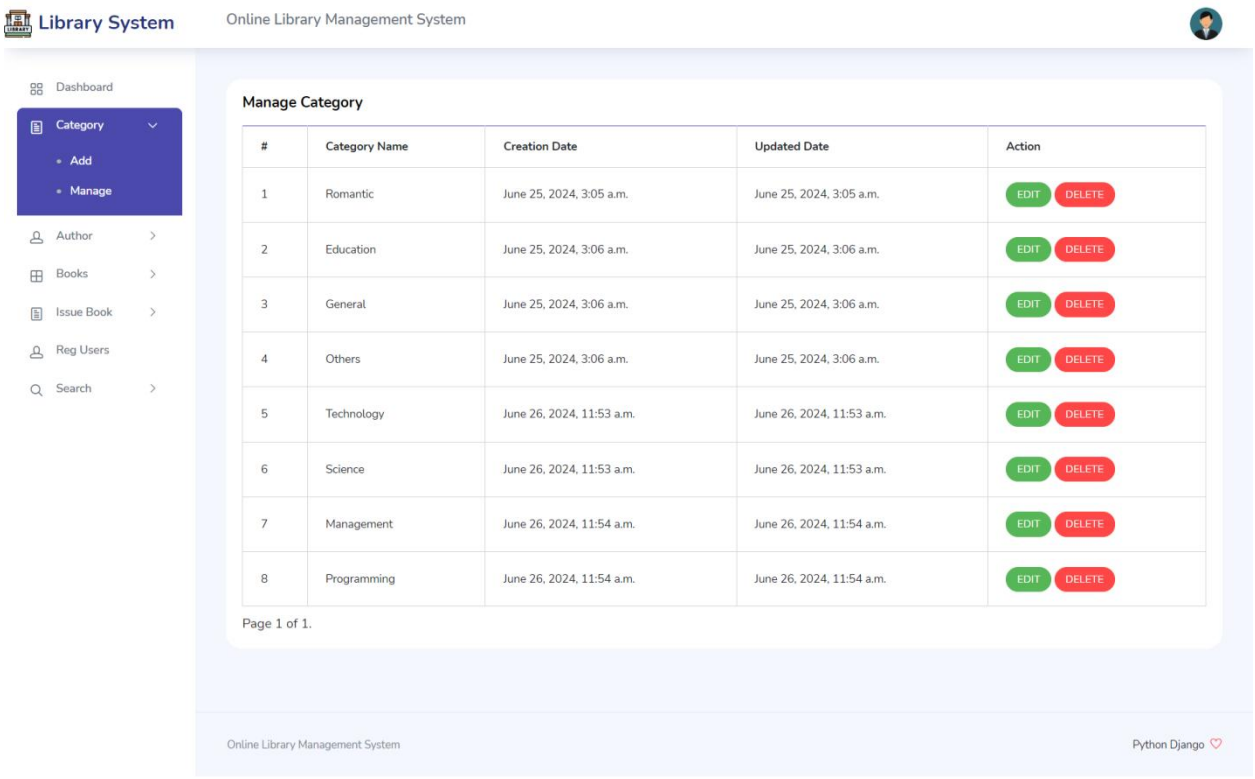
Admin Profile



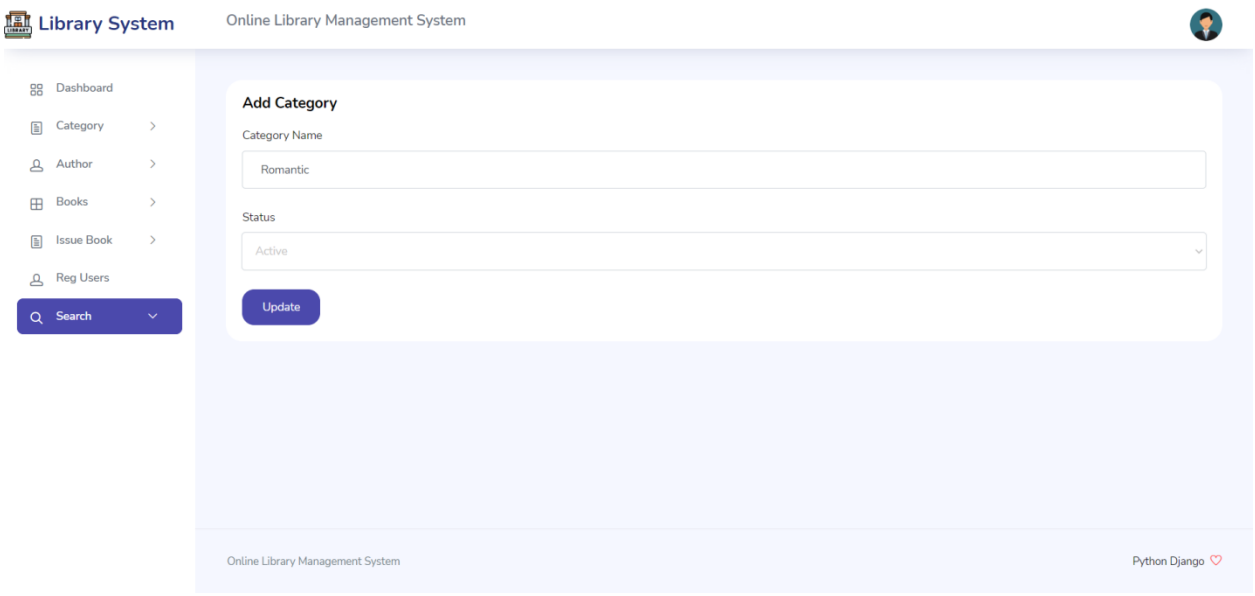
Add Category



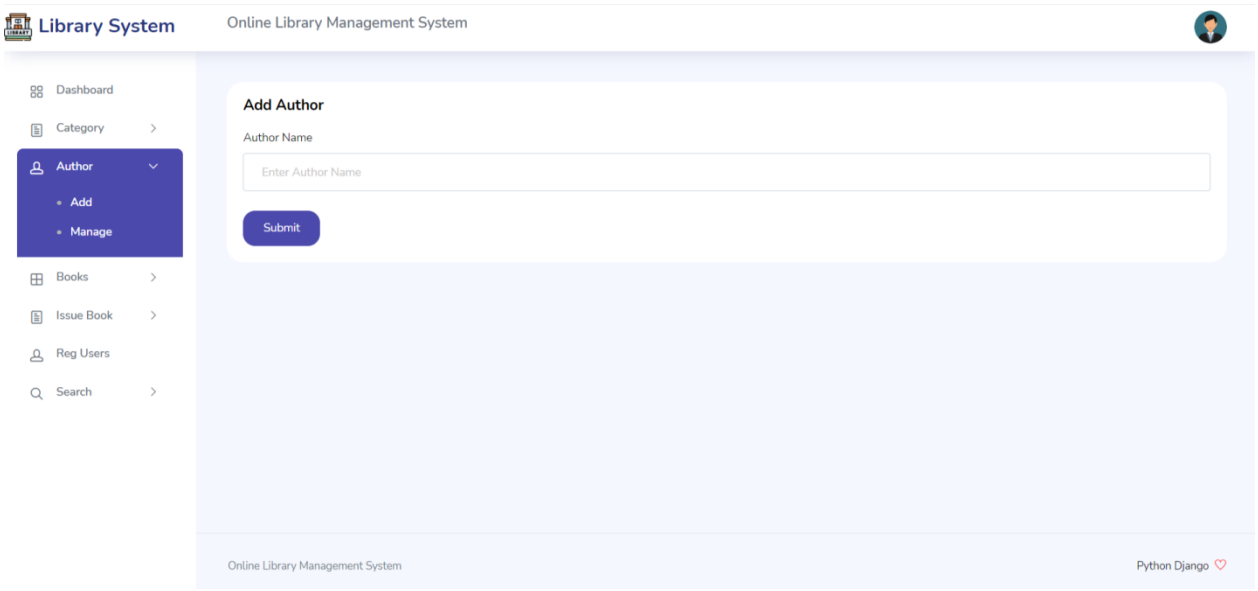
Manage Category



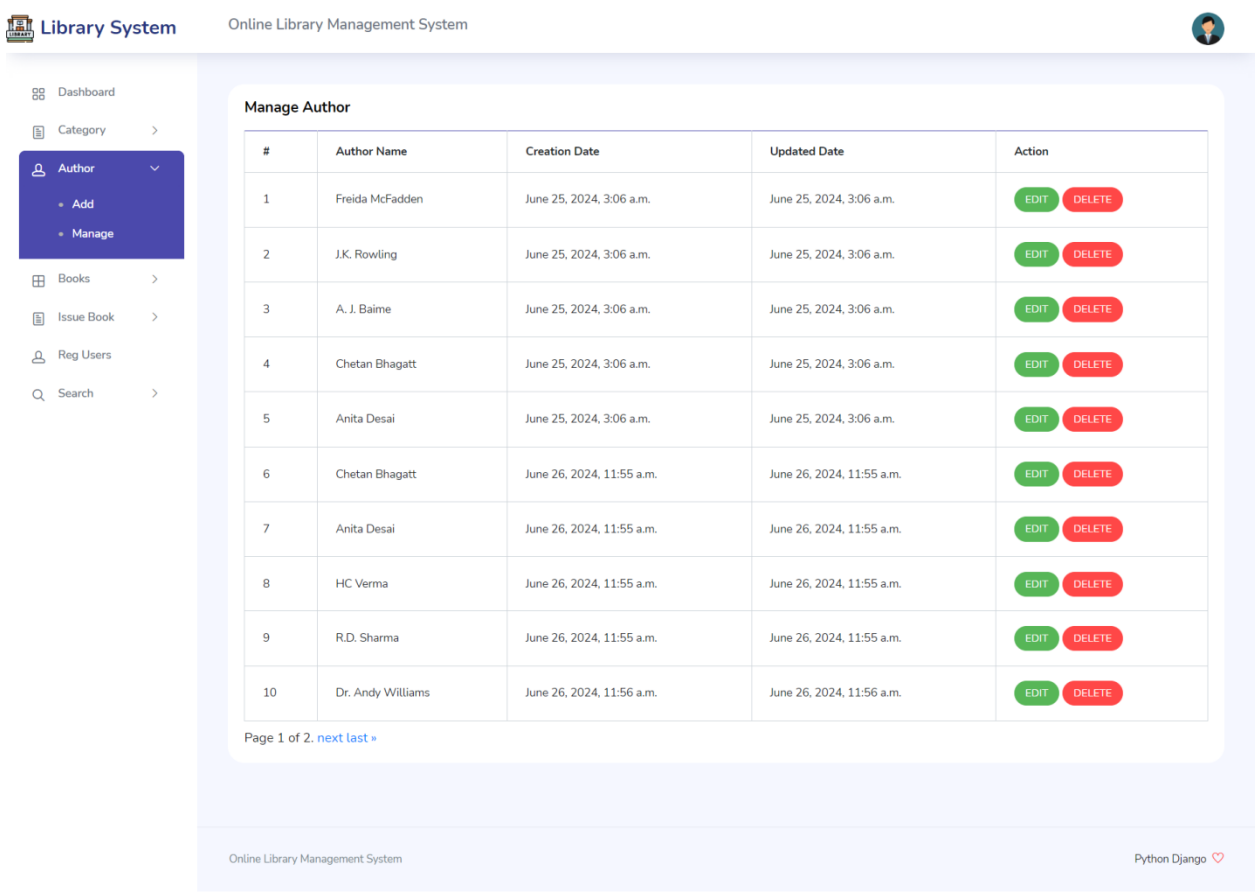
Update Category



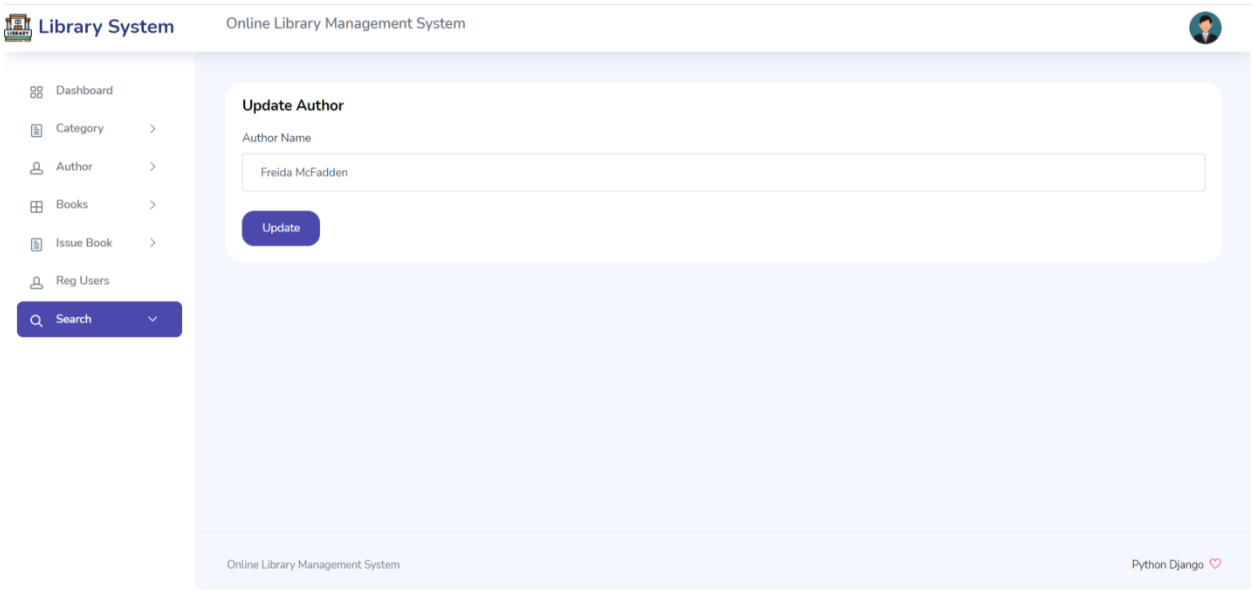
Add Author


Library System
Online Library Management System
Dashboard
Category
Author
Add
Manage
Books
Issue Book
Reg Users
Search
Add Author
Author Name
Enter Author Name
Submit
Online Library Management System
Python Django

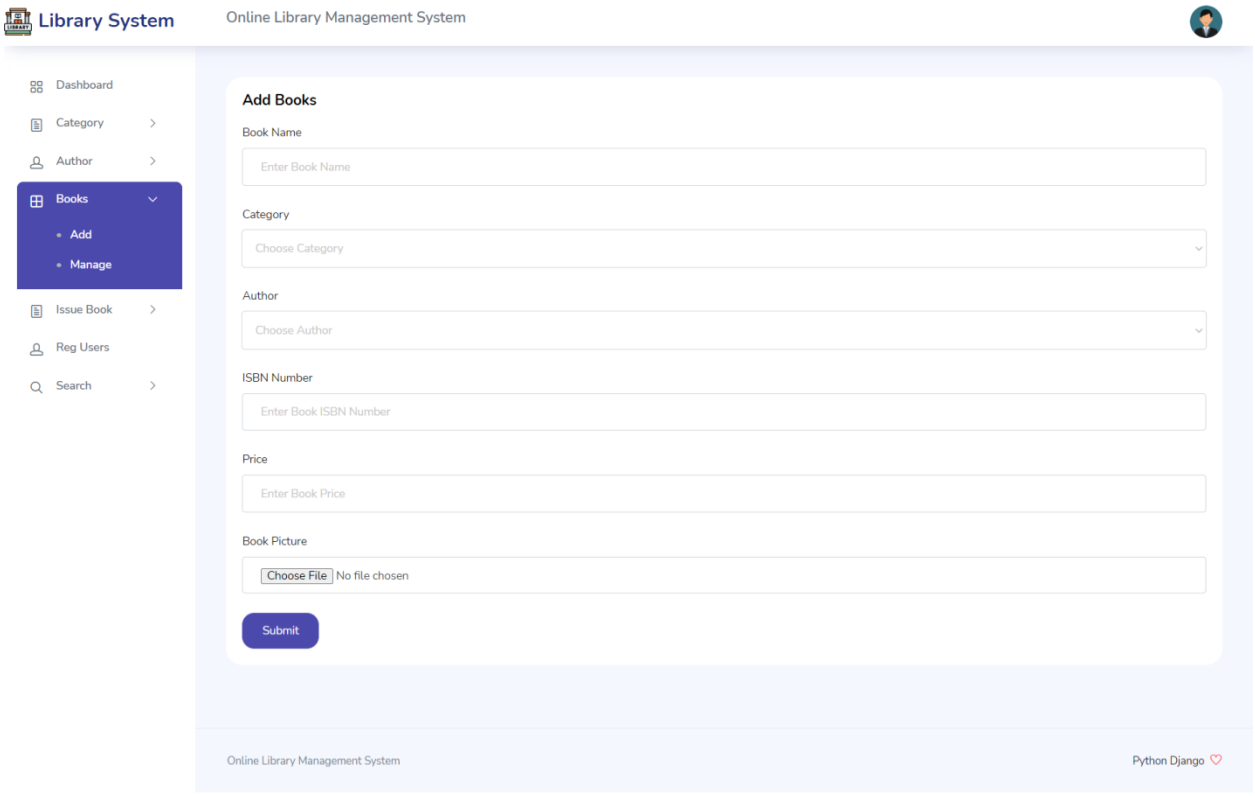
Manage Author



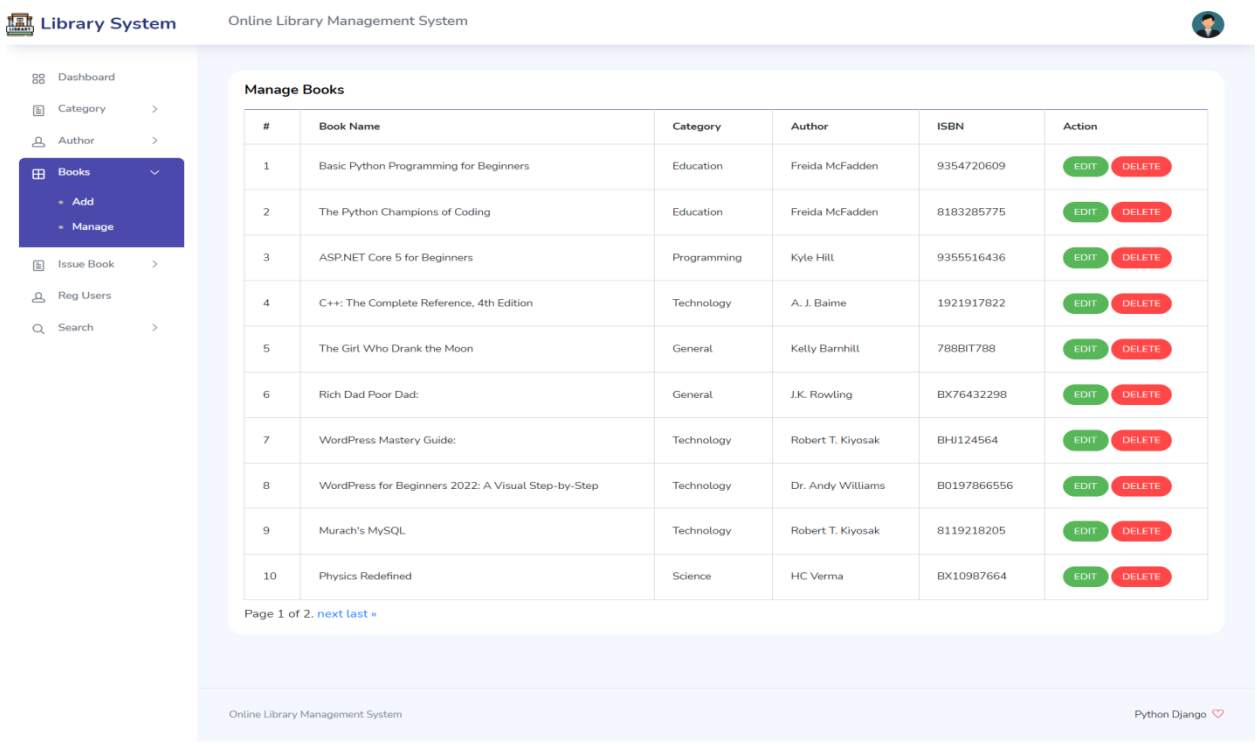
Update Author



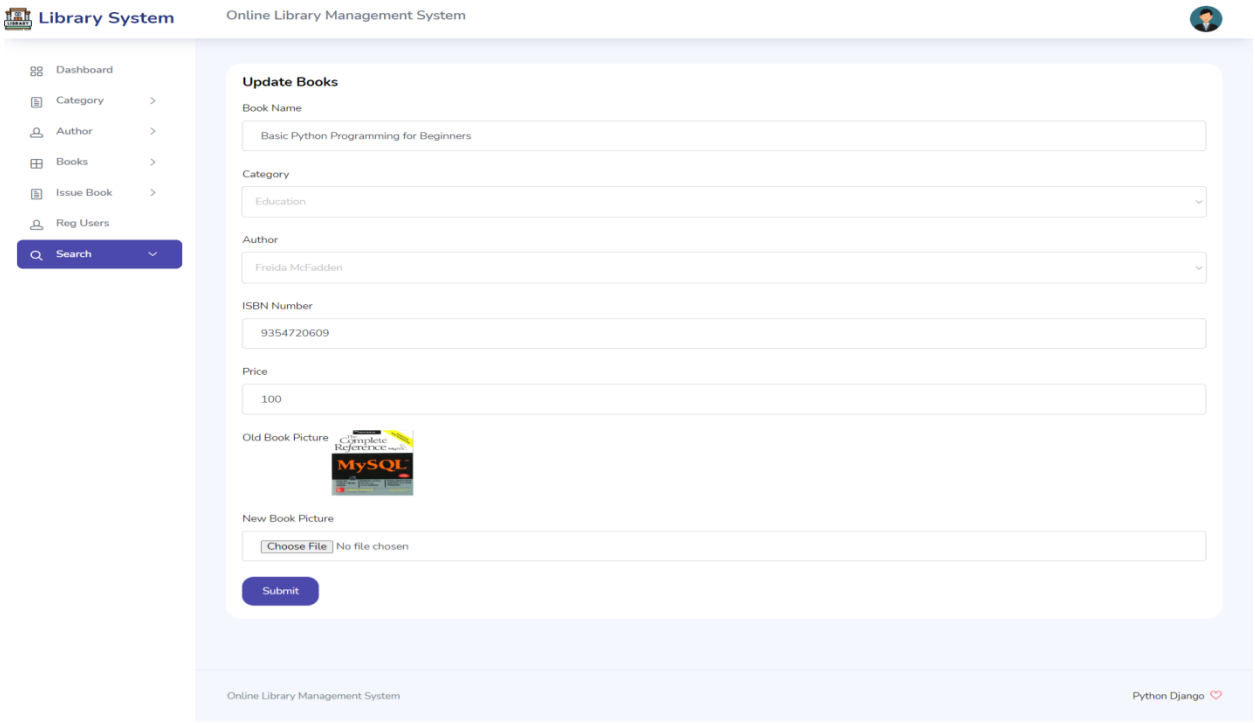
Add Books


Library System
Online Library Management System
Dashboard
Category
Author
Books
Add
Manage
Issue Book
Reg Users
Search
Add Books
Book Name
Enter Book Name
Category
Choose Category
Author
Choose Author
ISBN Number
Enter Book ISBN Number
Price
Enter Book Price
Book Picture
Choose File
No file chosen
Submit
Online Library Management System
Python Django

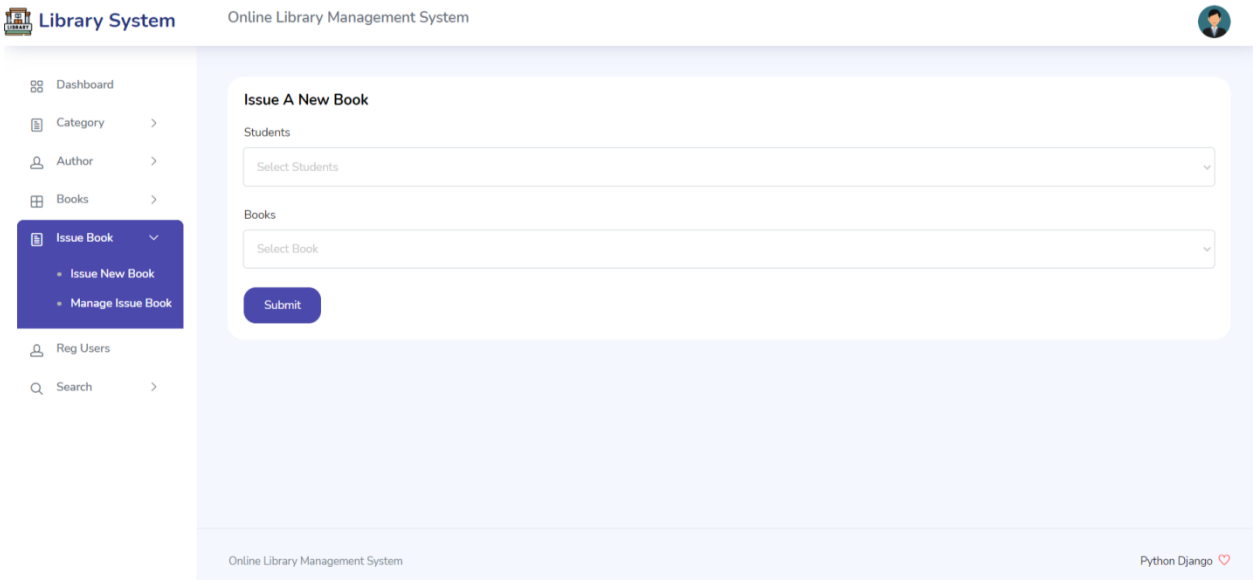
Manage Books



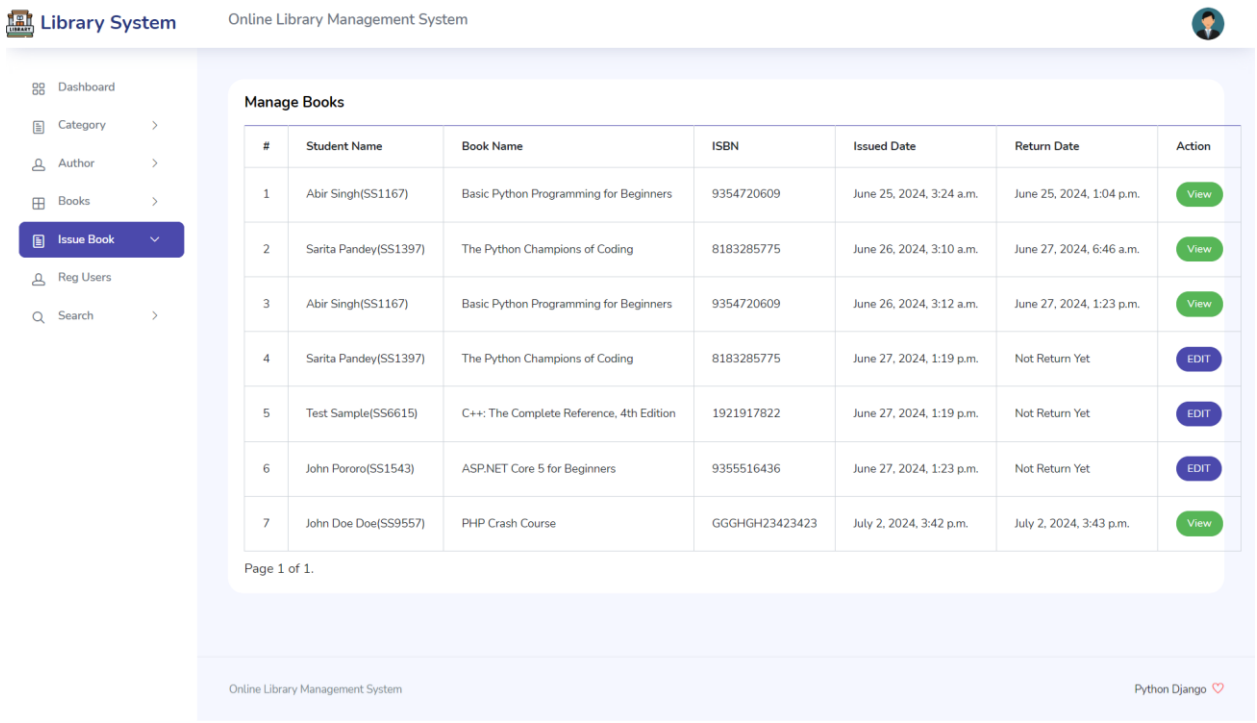
Update Books



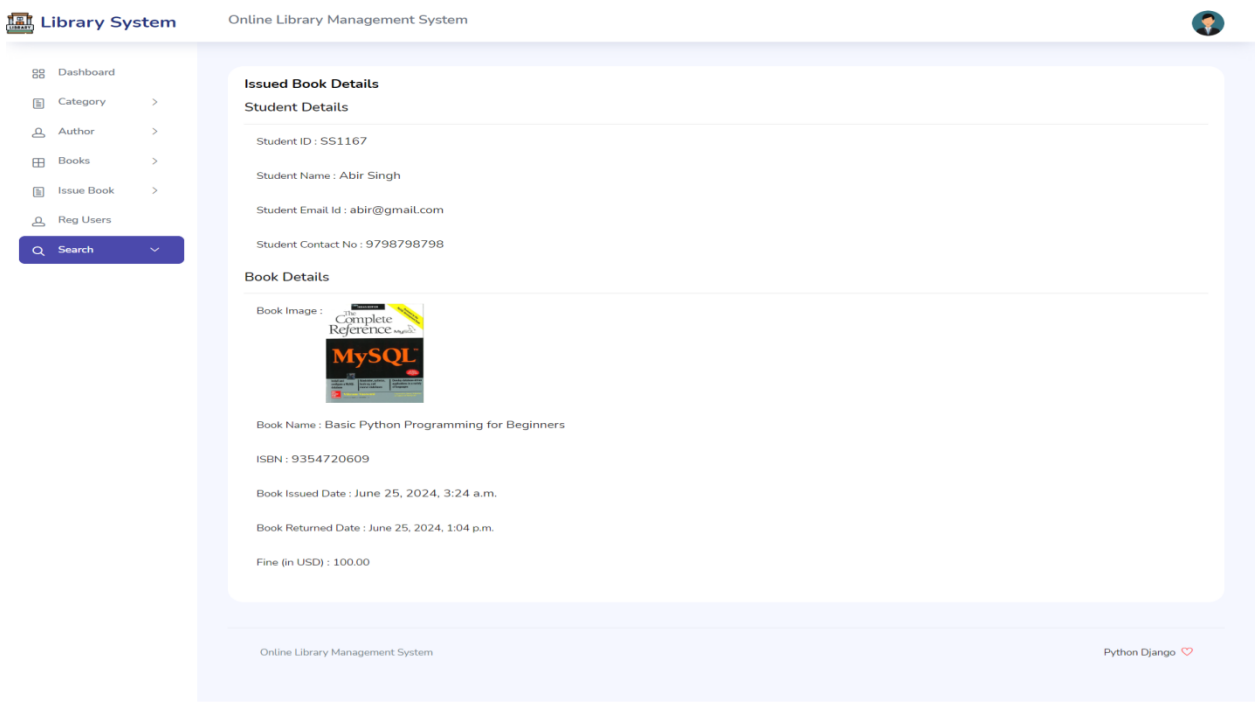
Issue New Books



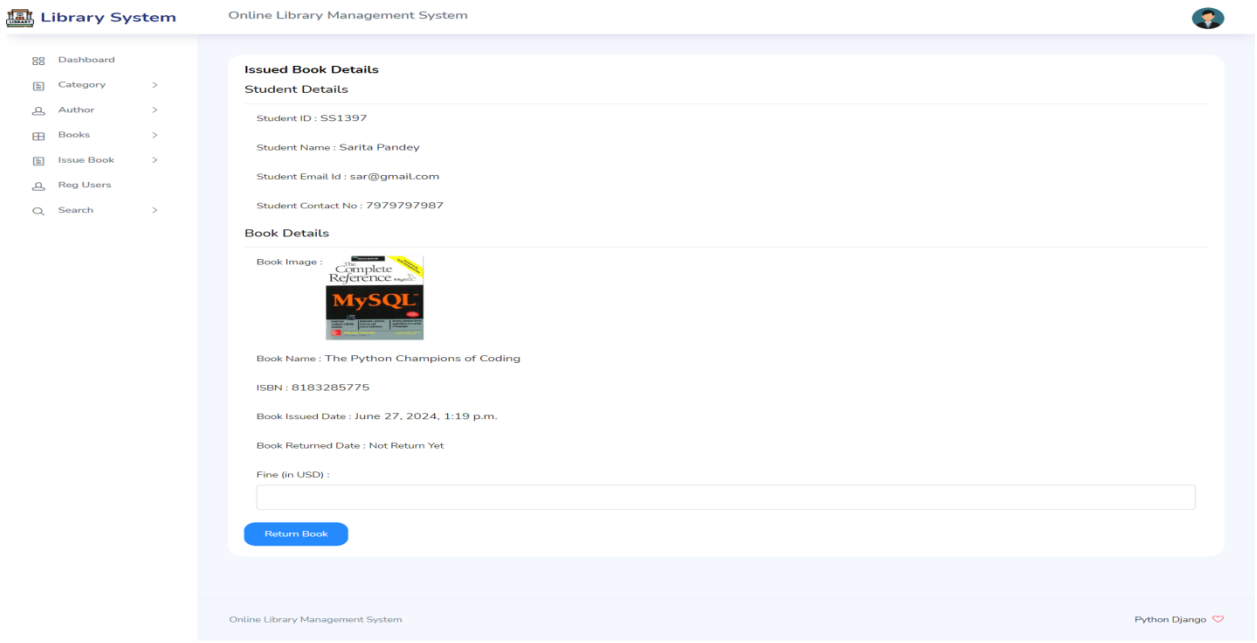
Manage Issue Books



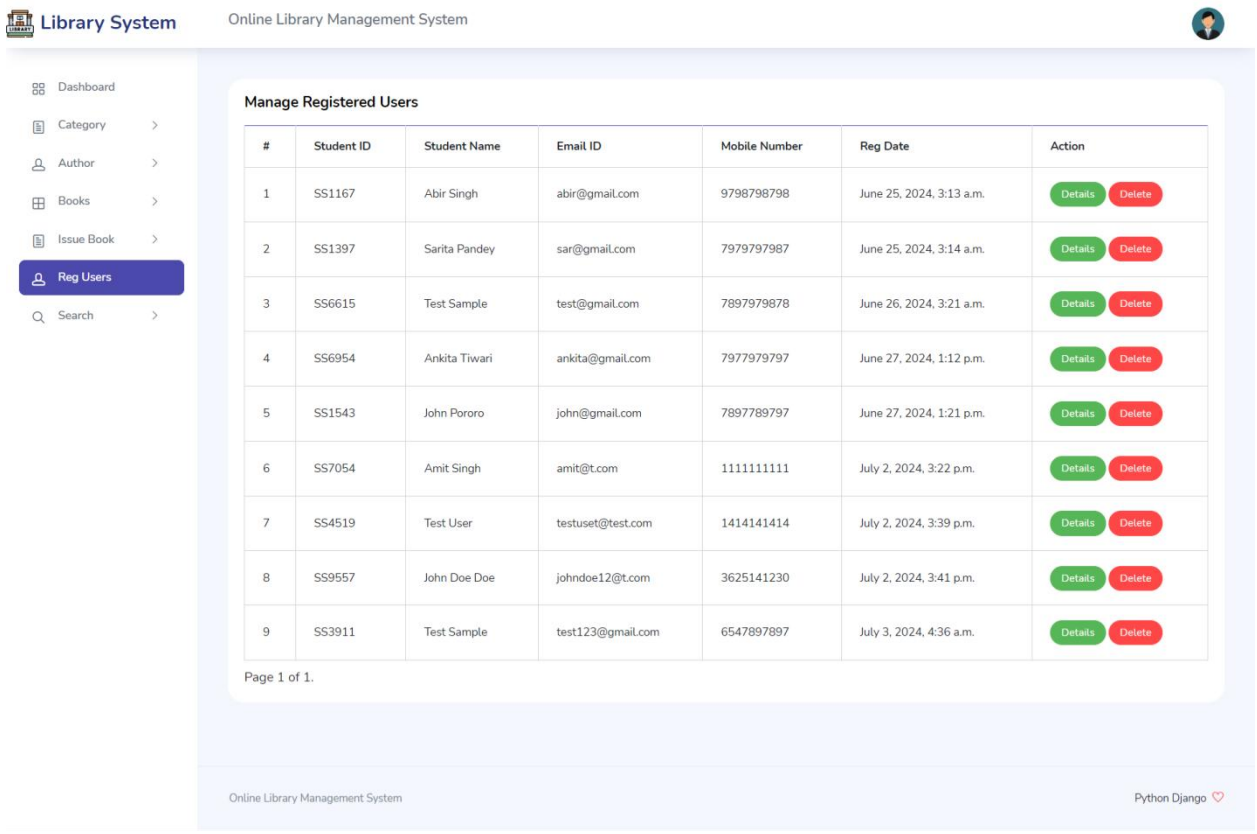
View Issue Books



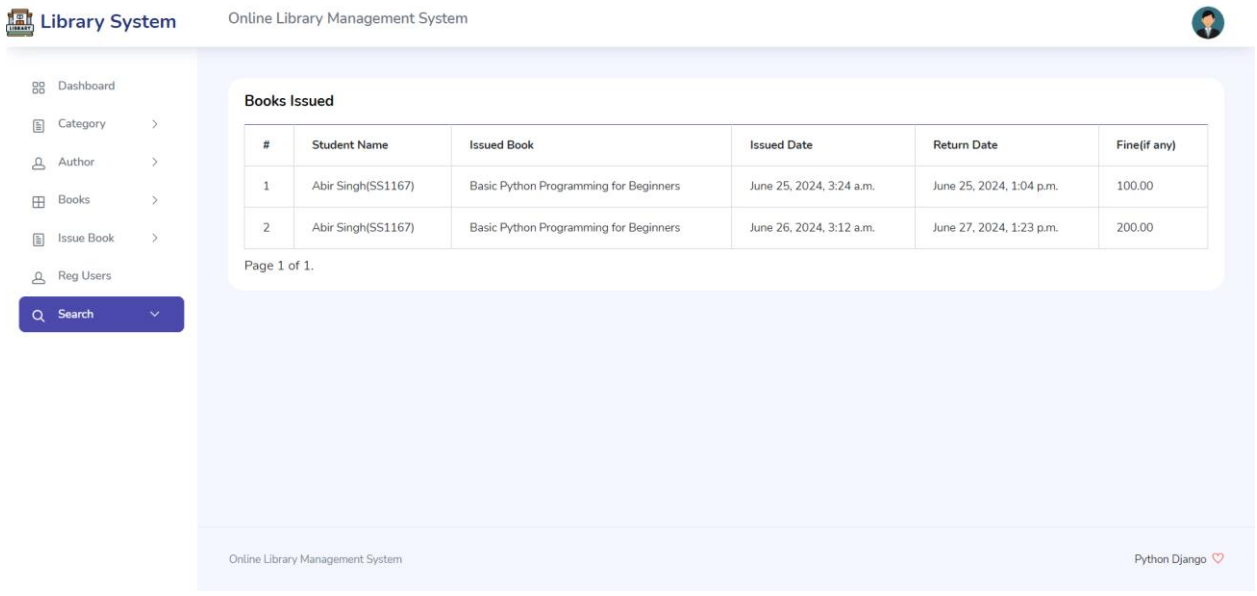
Update Issued Book



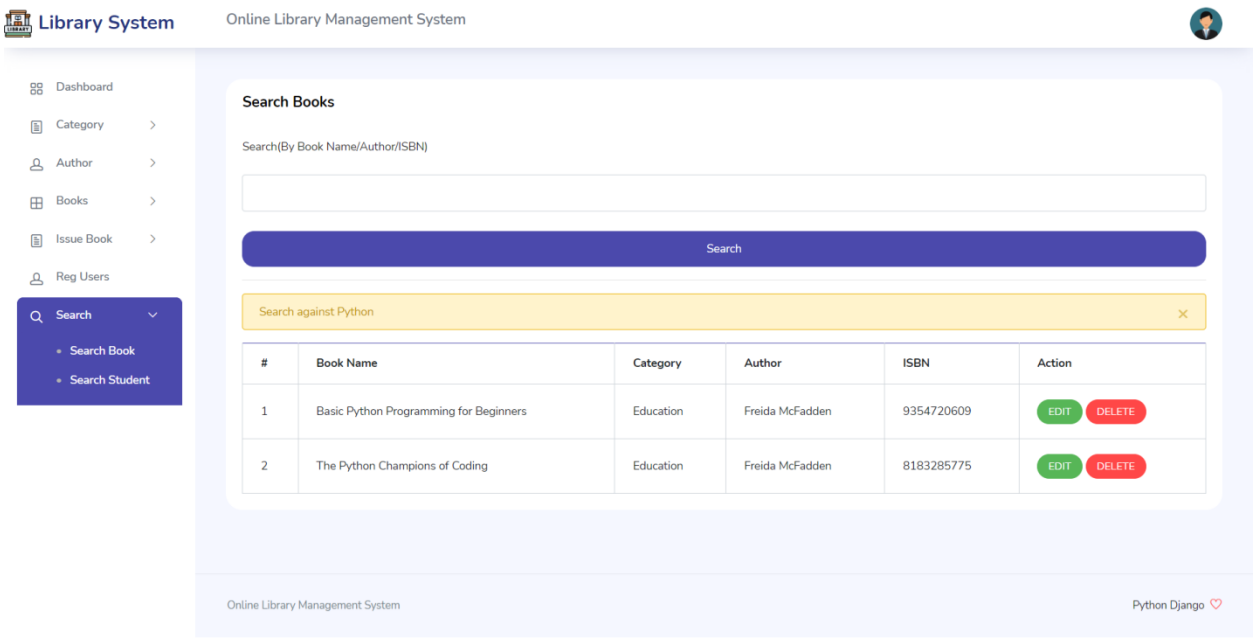
Register Students



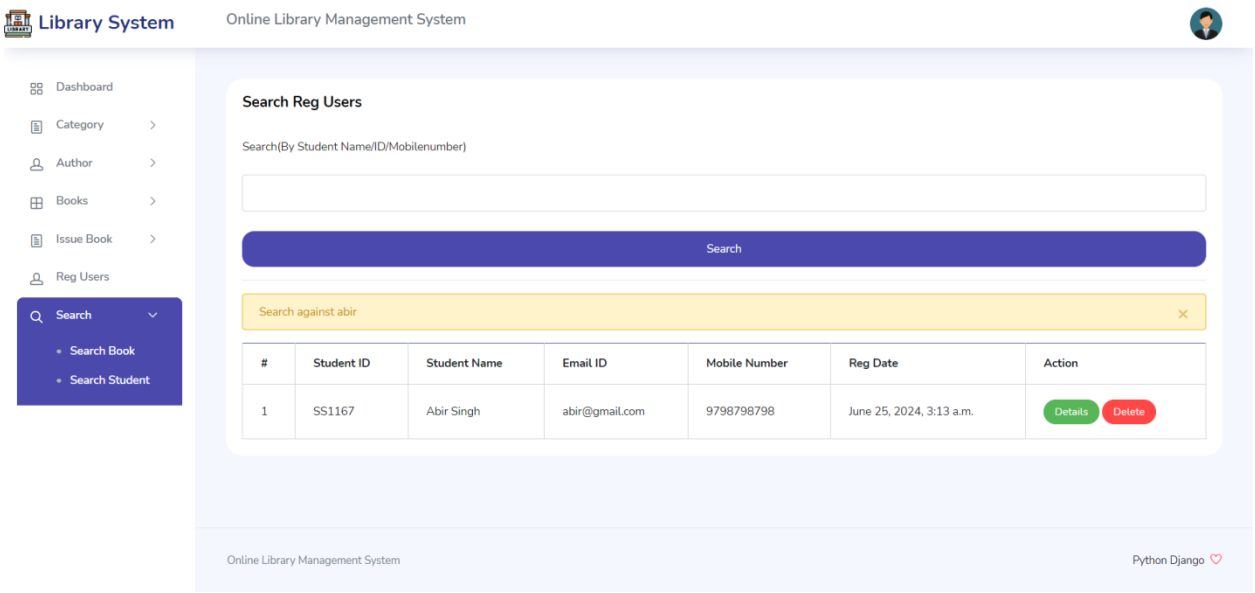
Book Issued Details to Students



Search Book


Library System
Online Library Management System
Dashboard
Category
Author
Books
Issue Book
Reg Users
Search
Search Book
Search Student
Search Books
Search(By Book Name/Author/ISBN)
Search
Search against Python
Book Name Category Author ISBN Action
1 Basic Python Programming for Beginners Education Freida McFadden 9354720609 EDIT DELETE
2 The Python Champions of Coding Education Freida McFadden 8183285775 EDIT DELETE
Online Library Management System
Python Django

Search Student



1.8 Implementation

Python

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.
Python is a programming language that lets you work quickly and integrate systems more efficiently.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

HTML

HTML (Hypertext Markup Language) is the set of markup symbols or codes inserted in a file intended for display on a World Wide Web browser page. The markup tells the Web browser how to display a Web page's words and images for the user. Each individual markup code is referred to as an element (but many people also refer to it as a tag). Some elements come in pairs that indicate when some display effect is to begin and when it is to end.

CASCADING STYLE SHEET (CSS)

Cascading Style Sheets (CSS) are a collection of rules we use to define and modify web pages. CSS are similar to styles in Word. CSS allow Web designers to have much more control over their pages look and layout. For instance, you could create a style that defines the body text to be Verdana, 10 point. Later on, you may easily change the body text to Times New Roman, 12 point by just changing the rule in the CSS. Instead of having to change the font on each page of your website, all you need to do is redefine the style on the style sheet, and it will instantly change on all of the pages that the style sheet has been applied to. With HTML styles, the font change would be applied to each instance of that font and have to be changed in each spot.

CSS can control the placement of text and objects on your pages as well as the look of those objects.

HTML information creates the objects (or gives objects meaning), but styles describe how the objects should appear. The HTML gives your page structure, while the CSS creates the “presentation”. An external CSS is really just a text file with a .css extension. These files can be created with Dreamweaver, a CSS editor, or even Notepad.

The best practice is to design your web page on paper first so you know where you will want to use styles on your page. Then you can create the styles and apply them to your page.

Javascript

JavaScript is a programming language commonly used in web development. It was originally developed by Netscape as a means to add dynamic and interactive elements to websites. While JavaScript is influenced byJava, the syntax is more similar to C and is based on ECMAScript, a scripting language developed by Sun Microsystems.

JavaScript is a client-side scripting language, which means the source code is processed by the client's web browser rather than on the web server. This means JavaScript functions can run after a webpage has loaded without COMMUNICATING with the server. For example, a JavaScript function may check a web form before it is submitted to make sure all the required fields have been filled out. The JavaScript code can produce an error message before any information is actually transmitted to the server.

Like server-side scripting languages, such as PHP and ASP, JavaScript code can be inserted anywhere within the HTML of a webpage. However, only the output of server-side code is displayed in the HTML, while JavaScript code remains fully visible in the source of the webpage. It can also be referenced in a separate .JS file, which may also be viewed in a browser.

Django

Django is a web application framework written in Python programming language. It is based on MVT (Model View Template) design pattern. The Django is very demanding due to its rapid development feature. It takes less time to build application after collecting client requirement.

This framework uses a famous tag line: The web framework for perfectionists with deadlines.

1.9 Integration

Integration is a crucial phase in the development of the **Library Management System**, where all the individual modules are brought together and made to work as a unified system. Each module—such as user authentication, book management, issue/return tracking, and report generation—is developed and tested independently. Once verified, these modules are integrated to ensure smooth communication and functionality across the system.

Modules to be Integrated:

1. **Admin Module:**
    - Manages the book inventory, user records, and transaction logs.
    - Interfaces with the database and report generation module.
2. **Student/User Module:**
    - Provides user login access.
    - Connects with the book search and issue/return tracking modules.
3. **Book Management Module:**
    - Manages data related to books (title, author, category, availability).
    - Integrated with both Admin and Student modules.
4. **Issue/Return Module:**
    - Handles issuing and returning of books.
    - Syncs with book availability and user history.
5. **Report and Notification Module:**
    - Generates reports and may provide notifications (e.g., due dates).
    - Relies on data from all other modules.

Integration Approach:

- **Incremental Integration Testing** is followed to ensure stability after each module is integrated.
- **Database Integration** ensures all modules share and retrieve real-time data accurately.
- **User Interface Integration** guarantees a consistent and intuitive experience for both admin and students.

The successful integration of all components ensures the system operates seamlessly and delivers the intended functionalities to its users.

1.10 Testing

System testing is a series of different test whose primary purpose is to fully exercise computer based system.
We can say that it will run according to its specifications and in the way users expect. Special test data are input for processing, and the results examined. A limited number of users may be allowed to use the system so that analyst can see whether they try to use it in unforeseen ways. It is desirable to discover any surprises before the organization implements the system and depends on it.

- We follow Black Box testing.
- Black box testing attempts to find errors in following
- Incorrect or missing function
- Interface errors
- Errors in data structure
- Initialization and termination errors

UNIT TESTING

Unit testing is commenced when a unit has been created and effectively reviewed .In order to test a single module we need to provide a complete environment i.e. besides the section we would require
The procedures belonging to other units that the unit under test calls

Non local data structures that module accesses

A procedure to call the functions of the unit under test with appropriate parameters

1. Test for the admin module

Testing admin login form-This form is used for log in of administrator of the system. In this form we enter the username and password if both are correct administration page will open otherwise if any of data is wrong it will get redirected back to the login page and again ask the details

Report Generation: admin can generate report from the main database.

INTEGRATION TESTING

In the Integration testing we test various combination of the project module by providing the input.
The primary objective is to test the module interfaces in order to confirm that no errors are occurring when one module invokes the other module.

FUNCTIONAL TESTING

Functional testing is a type of software testing that verifies whether each function of a system operates in conformance with the required specifications. It primarily focuses on the user interface, APIs, databases, security, client/server communication, and other functionalities of the system. This testing involves providing appropriate input and validating the output against the expected results to ensure that the software behaves as intended. Key areas assessed during functional testing include user authentication, data processing, input validation, output generation, and error handling. By simulating real-world scenarios and user interactions, functional testing helps ensure the system delivers accurate, reliable, and consistent results across all modules, thereby enhancing overall software quality and user satisfaction.

1.11 RESULT AND DISCUSSION

The **Library Management System** has been successfully developed and tested to meet the functional requirements set during the design phase. The system performs all core operations such as adding books, issuing and returning books, user management, and generating reports with accuracy and reliability.

Results Achieved:

- **Admin Panel**: Fully functional interface for managing books, users, and transactions.
- **Student Login**: Students can securely log in, check issued books, due dates, and search for available books.
- **Book Inventory Management**: Admins can efficiently add, update, or delete book records.
- **Issue/Return Tracking**: Accurate handling of book transactions, including date tracking and fine calculations (if applicable).
- **Data Storage**: All records are safely stored and retrieved from the database in real time.
- **User-Friendly Interface**: Clean and intuitive navigation for both students and admins.

Discussion:

- The system significantly **reduces manual efforts** in maintaining library records.
- **Error handling and validation** ensure data integrity throughout all operations.
- The modular design allows for **future scalability** (e.g., integrating RFID or mobile notifications).
- Compared to traditional methods, this system delivers a **faster and more transparent** experience to users.
- The system was tested with multiple users and performed well under typical usage scenarios.

This project serves as a **complete solution** for small to medium-scale libraries, offering automation, efficiency, and ease of use.

.

1.12 Advantages and Disadvantages:

Advantages:

1. **Time-Saving:**
   - Automates book issue and return processes, reducing manual work and saving valuable time for both librarians and students.
2. **User-Friendly Interface:**
   - Simple and intuitive design allows users with basic computer knowledge to navigate the system easily.
3. **Real-Time Information:**
   - Students can instantly check book availability, issue dates, and return deadlines.
4. **Data Accuracy and Security:**
   - Reduces human error and ensures secure storage of records in the database.
5. **Role-Based Access Control:**
   - Separate login for admin and students ensures proper data access control.
6. **Efficient Record Management:**
   - Facilitates quick searching, sorting, and retrieval of book and user records.
7. **Report Generation:**
   - Allows the admin to generate various reports related to books, users, and transactions.

Disadvantages:

1. **Initial Setup Cost:**
   - Requires investment in computers, internet, and server/database setup for initial deployment.
2. **Technical Knowledge Required:**
   - Admins may need basic training to manage and operate the system efficiently.
3. **Internet Dependency (for Web-Based Systems):**
   - System functionality may be affected if internet connectivity is poor or unavailable.
4. **System Maintenance:**
   - Requires periodic updates and backups to keep the system secure and efficient.

1.13 APPLICATION

The **Library Management System** can be applied in a wide range of educational and organizational environments where book and resource management is essential. Its flexibility and automation make it suitable for various institutions and organizations.

Key Applications:

1. **Schools and Colleges:**
   - Helps manage student and faculty book records, improve borrowing efficiency, and maintain organized inventory tracking.
2. **Universities:**
   - Useful for managing large volumes of books, research papers, journals, and theses in multiple departments.
3. **Public Libraries:**
   - Ideal for issuing books to citizens, maintaining user memberships, and handling overdue alerts and fines.
4. **Private Institutions and Coaching Centers:**
   - Assists in managing a small to medium-sized book collection efficiently.
5. **Medical and Engineering Colleges:**
   - Manages specialized journals, reference books, and department-wise categorization.
6. **Corporate Offices and Research Centers:**
   - Used to manage technical manuals, internal documentation, and reference materials.
7. **Digital and E-Libraries:**
   - Can be extended to support digital content management for e-books and PDFs.

1.14 Future Scope of the Project

The **Library Management System** has strong potential for future enhancement and scalability. As technology continues to evolve, several advanced features can be integrated to further improve the system's efficiency, accessibility, and user experience.

Future Enhancements:

1. **Mobile Application Integration:**
   - Development of Android/iOS apps to allow users to access the library system on mobile devices.
2. **Notification System:**
   - Integration of email or SMS alerts for due dates, book availability, and return reminders.
3. **AI-Powered Recommendations:**
   - Use of machine learning algorithms to recommend books based on user interests and borrowing history.
4. **E-Book and Digital Library Support:**
   - Integration with e-book platforms to allow online reading and downloading of digital content.
5. **Cloud-Based System:**
   - Migration to cloud infrastructure for better accessibility, scalability, and data backup.
6. **RFID and Barcode Integration:**
   - Automation of book issue/return using RFID or barcode scanners for faster transactions.
7. **Advanced Security Features:**
   - Implementation of two-factor authentication and encryption to improve data security.
8. **Multi-language Support:**
   - Making the system available in multiple languages to cater to diverse user groups.

1.15 Conclusion

This websites mainly focuses on how we can improve the traditional method of working of a library because the traditional method includes doing all the things in manual mode which is slow, less efficient, less secure, and difficult to manage. The solution to this is an online library management system which take care of all the work by automating and digitizing the whole process. Our application is based on Python and is linked to a relational database (MySQL). The frontend part has been coded using Python and its framework Django. The backend is supported and connected with database. With the increase in the workload of the library, new features can be added to the existing application to make it relevant in the future as well.

As we know with the increase in number of students, books, complexity other workloads, there can be a need of shifting the library data from the local database to the cloud.

1.16 REFERENCES

] Django homepage. http://www.djangoproject.com/.

[2] Python documentation. http://www.python.org/doc.

[3] Django(web framework). http://en.wikipedia.org/wiki/Django.

[4] Django documentation. http://docs.djangoproject.com.

[5] Python(programming language). http://en.wikipedia.org/wiki/Python.

[6]Books: Web Development with Django by Ben Shaw, Saurabh,