Library Management System

**Your Name**

**Class XII Computer Science Project**

**11/25/2020**

It is a small yet feature rich Library Management System Project Developed using python programming language and MySQL database.

# Contents

[Acknowledgement 2](#_Toc57238777)

[Introduction 3](#_Toc57238778)

[Objectives Of the Project 4](#_Toc57238779)

[Proposed System 5](#_Toc57238780)

[System Development Life Cycle 6](#_Toc57238781)

[Phases Of SDLC 7](#_Toc57238782)

[Flow Chart 8](#_Toc57238783)

[Source Code 9](#_Toc57238784)

[Output 23](#_Toc57238785)

[Hardware and Software Requirements 27](#_Toc57238786)

[Bibliography 28](#_Toc57238787)

# Acknowledgement

I would like to express my heartfelt thanks of gratitude to my computer teacher and project guide (Name of teacher) as well as our principal (Name of principal) who offered me this precious opportunity to work on this “Library Management System”, which also helped me learn so many new and useful things while doing research on the internet and library.

I am thankful to many web sources such as Wikipedia, Apka Guruji for being handy in small yet critical parts of the project such as preparing this report file, Software Installation and Configuration etc.

I would also like to thank my parents and friends who helped me immensely by providing help and support in completing the project on time.

Your Name

# Introduction

Books are very important part of our life, whether we are student or grown up, we never stop learning. A library is the place that comes to our mind whenever we are in mood to dive into the universe of knowledge.

This simple python project is an implementation of a system to manage a library, its books and its members. It is implemented using Python programming language and MySQL database, The program runs in any computer that supports Python Interpreter and MySQL server Installation.

# Objectives Of the Project

Here are the prime objectives of this project

* Use a computer to handle all kinds of activities in a library
* Implement a library management software which is small yet powerful
* Use mysql databases for working with library data
  + Keeping records of books
  + Keeping records of members
  + Keeping records of issue and return transactions
  + Easily Search books
  + Easily Search Library Members
  + Easily Issue/Return Books
  + Automatically Calculate the fine if a book is returned late

In addition to above primary objectives we aim to keep the project’s complexity to the lowest level possible, so that the operator of the software can carry out all the operators with ease.

# Proposed System

The proposed system will be a console based (CUI: Character User Interface) menu driven python program that uses MySQL databases to store the different set of data related to a library and can run in all computers and laptop. We will implement the system using following components:

* myprint.py: It will be used to create alternative versions of the print() and input() functions that can draw horizontal separators, move the cursor to the center of the screen etc.
* book.py: It will define the structure of a book record, such as book id, title, author etc.,
* books.py: It will be responsible to make available the set of operations that can be done with books
* member.py: It will define what different components will combine to create a member record.
* members.py: It will handle all the operations that can be executed with the member records.
* issue\_return.py: It defines what information is recorded when a book is issued to a member or returned to the library. such as book id, member id, issue date etc.
* issue\_returns.py: It provides all the options to work with the issue return register, such as issue a book, return a book, see records of issued books etc.
* database.py: It will help in creating connection with the mysql database, it will store the database name, password, host etc.
* main.py: It will the starting point of execution of our software, it will use the database module to connect to the database and pass control to other components of the system.

# System Development Life Cycle

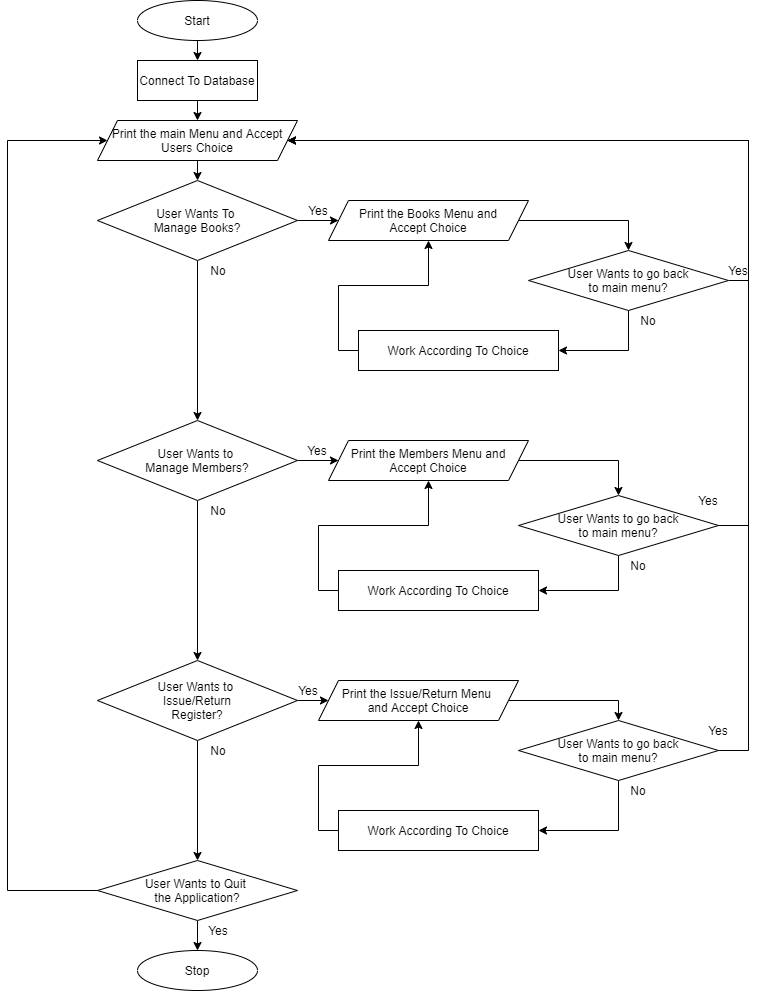
Systems development life cycle (SDLC), also referred to as the application development life-cycle, is a process for planning, creating, testing, and deploying an information system. SDLC includes following activities:

1. Requirement analysis: Obtain end user requirements through documentation, client interviews, observation, and questionnaires.
2. Design: desired features and operations are described in detail, including screen layouts, business rules, process diagrams, pseudocode, and other documentation.
3. Implementation: The real code is written here
4. Testing: All the modules are brought together into a special testing environment, then checked for errors, bugs, and interoperability.
5. Documentation: It helps the end user understand how to use the different components of the system.
6. Deployment: This is the final stage of initial development, where the software is put into production and runs actual business
7. Evaluation: In this stage we check if the newly implemented system meets the initial business requirements and objectives, if the system is reliable and fault-tolerant, and if it functions according to the approved functional requirements.

# Phases Of SDLC

SDLC can be divided into following phases:

# Flow Chart



# Source Code

## myprint.py

SCREEN\_WIDTH **=** 100

**def** print\_center**(**s**):**

x\_pos **=** SCREEN\_WIDTH **//** 2

**print((**" " **\*** x\_pos**),** s**)**

**def** print\_bar**():**

**print(**"=" **\*** 100**)**

**def** print\_bar\_ln**():**

print\_bar**()**

**print()**

**def** input\_center**(**s**):**

x\_pos **=** SCREEN\_WIDTH **//** 2

**print((**" " **\*** x\_pos**),** s**,** end**=**''**)**

**return** **input()**

## book.py

**from** myprint **import** print\_bar

TABLE\_NAME **=** "books"

**class** **Book:**

**def** \_\_init\_\_**(**self**):**

self**.**book\_id **=** 0

self**.**title **=** 0

self**.**author **=** ""

self**.**topic **=** ""

self**.**available **=** **True**

**def** create**(**self**,** book\_id**,** title**,** author**,** topic**,** available**):**

self**.**book\_id **=** book\_id

self**.**title **=** title

self**.**author **=** author

self**.**topic **=** topic

self**.**available **=** available

**return** self

**def** create\_from\_record**(**self**,** record**):**

self**.**book\_id **=** record**[**'id'**]**

self**.**title **=** record**[**'title'**]**

self**.**author **=** record**[**'author'**]**

self**.**topic **=** record**[**'topic'**]**

self**.**available **=** record**[**'available'**]**

**return** self

**def** print\_all**(**self**):**

**print(str(**self**.**book\_id**).**ljust**(**3**),**

self**.**title**.**ljust**(**15**),**

self**.**author**.**ljust**(**15**),**

self**.**topic**.**ljust**(**15**),**

**(**"YES" **if** self**.**available **else** "NO"**).**ljust**(**15**))**

**def** print\_full**(**self**):**

print\_bar**()**

**print(**"Record #"**,** self**.**book\_id**)**

**print(**"Title: "**,** self**.**title**)**

**print(**"Author: "**,** self**.**author**)**

**print(**"Topic: "**,** self**.**topic**)**

**print(**"available: "**,** "YES" **if** self**.**available **else** "NO"**)**

print\_bar**()**

**def** create\_book**():**

book\_id **=** **None**

title **=** **input(**"Enter the title: "**)**

author **=** **input(**"Enter the author name: "**)**

topic **=** **input(**"Enter the topic: "**)**

available **=** **True**

**return** Book**().**create**(**book\_id**,** title**,** author**,** topic**,** available**)**

**def** print\_header**():**

**print(**"="**\***100**)**

**print(**"id"**.**ljust**(**3**),**

"Title"**.**ljust**(**15**),**

"Author"**.**ljust**(**15**),**

"Topic"**.**ljust**(**15**),**

"Available"**.**ljust**(**15**)**

**)**

**print(**"="**\***100**)**

**def** create\_book\_table**(**database**):**

cursor **=** database**.**cursor**()**

cursor**.**execute**(**"DROP table if exists {0}"**.format(**TABLE\_NAME**))**

cursor**.**execute**(**"create table {0} ("

"id int primary key auto\_increment,"

"title varchar(50),"

"author varchar(50),"

"topic varchar(50),"

"available bool)"**.format(**TABLE\_NAME**))**

## books.py

**import** mysql

**from** book **import** Book**,** create\_book**,** TABLE\_NAME**,** create\_book\_table**,** print\_header

NUMBER\_OF\_RECORDS\_PER\_PAGE **=** 10

**def** add\_book**(**database**,** cursor**):**

book **=** create\_book**()**

query **=** "insert into {0}(title,author,topic,available) values('{1}','{2}','{3}',{4})"**.**\

**format(**TABLE\_NAME**,** book**.**title**,** book**.**author**,** book**.**topic**,** book**.**available**)**

**try:**

cursor**.**execute**(**query**)**

database**.**commit**()**

**except** mysql**.**connector**.**Error **as** err**:**

create\_book\_table**(**database**)**

cursor**.**execute**(**query**)**

database**.**commit**()**

**print(**"Operation Successful"**)**

**def** show\_record**(**cursor**,** query**):**

**try:**

cursor**.**execute**(**query**)**

records **=** cursor**.**fetchall**()**

**if** cursor**.**rowcount **==** 0**:**

**print(**"No Matching Records"**)**

**return**

record **=** records**[**0**]**

book **=** Book**().**create\_from\_record**(**record**)**

book**.**print\_full**()**

**return** book

**except** mysql**.**connector**.**Error **as** err**:**

**print(**err**)**

**def** show\_records**(**cursor**,** query**):**

**try:**

cursor**.**execute**(**query**)**

records **=** cursor**.**fetchall**()**

**if** cursor**.**rowcount **==** 0**:**

**print(**"No Matching Records"**)**

**return**

print\_header**()**

**for** record **in** records**:**

book **=** Book**().**create\_from\_record**(**record**)**

book**.**print\_all**()**

**return** records

**except** mysql**.**connector**.**Error **as** err**:**

**print(**err**)**

**def** get\_and\_print\_book\_by\_id**(**cursor**):**

book\_id **=** **int(input(**"Enter the book id: "**))**

query **=** "select \* from {0} where id={1}"**.format(**TABLE\_NAME**,** book\_id**)**

book **=** show\_record**(**cursor**,** query**)**

**return** book

**def** edit\_by\_book\_id**(**database**,** cursor**):**

book **=** get\_and\_print\_book\_by\_id**(**cursor**)**

**if** book **is** **not** **None:**

query **=** "update {0} set"**.format(**TABLE\_NAME**)**

**print(**"Input new values (leave blank to keep previous value)"**)**

title **=** **input(**"Enter new book title: "**)**

**if** **len(**title**)** **>** 0**:**

query **+=** " title='{0}',"**.format(**title**)**

author **=** **input(**"Enter new author: "**)**

**if** **len(**author**)** **>** 0**:**

query **+=** " author='{0}',"**.format(**author**)**

topic **=** **input(**"Enter new topic: "**)**

**if** **len(**topic**)** **>** 0**:**

query **+=** " topic='{0}',"**.format(**topic**)**

query **=** query**[**0**:-**1**]** **+** " where id={0}"**.format(**book**.**book\_id**)**

confirm **=** **input(**"Confirm Update (Y/N): "**).**lower**()**

**if** confirm **==** 'y'**:**

cursor**.**execute**(**query**)**

database**.**commit**()**

**print(**"Operation Successful"**)**

**else:**

**print(**"Operation Cancelled"**)**

**def** change\_book\_status**(**database**,** cursor**,** book\_id**,** available**):**

query **=** "update {0} set available={1} where id={2}"**.format(**TABLE\_NAME**,** available**,** book\_id**)**

cursor**.**execute**(**query**)**

database**.**commit**()**

**def** delete\_by\_book\_id**(**database**,** cursor**):**

book **=** get\_and\_print\_book\_by\_id**(**cursor**)**

**if** book **is** **not** **None:**

confirm **=** **input(**"Confirm Deletion (Y/N): "**).**lower**()**

**if** confirm **==** 'y'**:**

query **=** "delete from {0} where id={1}"**.format(**TABLE\_NAME**,** book**.**book\_id**)**

cursor**.**execute**(**query**)**

database**.**commit**()**

**print(**"Operation Successful"**)**

**else:**

**print(**"Operation Cancelled"**)**

**def** book\_menu**(**database**,** cursor**):**

**while** **True:**

**print()**

**print(**"============================"**)**

**print(**"==========Books Menu========="**)**

**print(**"============================"**)**

**print()**

**print(**"1. Add new book"**)**

**print(**"2. Get book details by book id"**)**

**print(**"3. Search a book by title or topic"**)**

**print(**"4. Edit Book details"**)**

**print(**"5. Delete Book"**)**

**print(**"6. View all Books"**)**

**print(**"0. Go Back"**)**

choice **=** **int(input(**"Enter your choice: "**))**

**if** choice **==** 1**:**

add\_book**(**database**,** cursor**)**

**elif** choice **==** 2**:**

get\_and\_print\_book\_by\_id**(**cursor**)**

**elif** choice **==** 3**:**

topic **=** **input(**"Enter a part of the book title or topic: "**)**

query **=** "select \* from {0} where title like '%{1}%' or topic like '%{1}%'"**.format(**TABLE\_NAME**,** topic**)**

show\_records**(**cursor**,** query**)**

**elif** choice **==** 4**:**

edit\_by\_book\_id**(**database**,** cursor**)**

**elif** choice **==** 5**:**

delete\_by\_book\_id**(**database**,** cursor**)**

**elif** choice **==** 6**:**

query **=** "select \* from {0}"**.format(**TABLE\_NAME**)**

show\_records**(**cursor**,** query**)**

**elif** choice **==** 0**:**

**break**

**else:**

**print(**"Invalid choice (Press 0 to go back)"**)**

## member.py

**from** datetime **import** datetime

**from** dateutil**.**relativedelta **import** relativedelta

**from** myprint **import** print\_bar

TABLE\_NAME **=** "members"

MEMBERSHIP\_PERIOD **=** 3

**class** **Member:**

**def** \_\_init\_\_**(**self**):**

self**.**member\_id **=** 0

self**.**name **=** ""

self**.**address **=** ""

self**.**phone **=** ""

self**.**joining\_date **=** ""

self**.**valid\_till **=** ""

**def** create**(**self**,** member\_id**,** name**,** address**,** phone**,** joining\_date**,** valid\_till**):**

self**.**member\_id **=** member\_id

self**.**name **=** name

self**.**address **=** address

self**.**phone **=** phone

self**.**joining\_date **=** joining\_date

self**.**valid\_till **=** valid\_till

**return** self

**def** create\_from\_record**(**self**,** record**):**

self**.**member\_id **=** record**[**'id'**]**

self**.**name **=** record**[**'name'**]**

self**.**address **=** record**[**'address'**]**

self**.**phone **=** record**[**'phone'**]**

self**.**joining\_date **=** record**[**'joining'**]**

self**.**valid\_till **=** record**[**'valid\_till'**]**

**return** self

**def** print\_all**(**self**):**

**print(str(**self**.**member\_id**).**ljust**(**3**),**

self**.**name**[**0**:**15**].**ljust**(**15**),**

self**.**address**[**0**:**15**].**ljust**(**15**),**

self**.**phone**.**ljust**(**15**),**

self**.**joining\_date**.**strftime**(**"%d-%b-%y"**).**ljust**(**15**),**

**(**self**.**valid\_till**.**strftime**(**"%d %b %y"**)** **if** self**.**valid\_till **is** **not** **None** **else** "None"**).**ljust**(**15**))**

**def** print\_full**(**self**):**

print\_bar**()**

**print(**"Member #"**,** self**.**member\_id**)**

**print(**"Name: "**,** self**.**name**)**

**print(**"Address: "**,** self**.**address**)**

**print(**"Phone: "**,** self**.**phone**)**

**print(**"Joined on "**,** self**.**joining\_date**.**strftime**(**"%d %b %y"**))**

**print(**"Membership expires on : "**,** self**.**valid\_till**.**strftime**(**"%d %b %y"**)** **if** self**.**valid\_till **is** **not** **None** **else** **None)**

print\_bar**()**

**def** create\_member**():**

member\_id **=** **None**

name **=** **input(**"Enter the name: "**)**

address **=** **input(**"Enter the address: "**)**

phone **=** **input(**"Enter the phone: "**)**

joining\_date **=** datetime**.**now**()**

valid\_till **=** joining\_date **+** relativedelta**(**months**=**MEMBERSHIP\_PERIOD**)**

**return** Member**().**create**(**member\_id**,** name**,** address**,** phone**,** joining\_date**,** valid\_till**)**

**def** print\_header**():**

**print(**"="**\***100**)**

**print(**"id"**.**ljust**(**3**),**

"name"**.**ljust**(**15**),**

"address"**.**ljust**(**15**),**

"phone"**.**ljust**(**15**),**

"joining"**.**ljust**(**15**),**

"expiry"**.**ljust**(**15**))**

**print(**"="**\***100**)**

**def** create\_members\_table**(**database**):**

cursor **=** database**.**cursor**()**

cursor**.**execute**(**"DROP table if exists {0}"**.format(**TABLE\_NAME**))**

cursor**.**execute**(**"create table {0} ("

"id int primary key auto\_increment,"

"name varchar(20),"

"address varchar(50),"

"phone varchar(10),"

"joining datetime,"

"valid\_till datetime)"**.format(**TABLE\_NAME**))**

## members.py

**from** datetime **import** datetime

**import** mysql

**from** member **import** Member**,** create\_member**,** TABLE\_NAME**,** create\_members\_table**,** print\_header**,** MEMBERSHIP\_PERIOD

NUMBER\_OF\_RECORDS\_PER\_PAGE **=** 10

**def** add\_member**(**database**,** cursor**):**

member **=** create\_member**()**

confirm **=** **input(**"Complete the operation? (Y/N) "**).**lower**()**

**if** confirm **==** 'y'**:**

query **=** "insert into {0}(name, address, phone, joining,valid\_till) values('{1}','{2}','{3}','{4}','{5}')"**.** \

**format(**TABLE\_NAME**,** member**.**name**,** member**.**address**,** member**.**phone**,**

member**.**joining\_date**.**strftime**(**"%Y-%m-%d %H:%M:%S"**),**

member**.**valid\_till**.**strftime**(**"%Y-%m-%d %H:%M:%S"**))**

**try:**

cursor**.**execute**(**query**)**

database**.**commit**()**

**except** mysql**.**connector**.**Error**:**

create\_members\_table**(**database**)**

cursor**.**execute**(**query**)**

database**.**commit**()**

**print(**"Operation Successful"**)**

**else:**

**print(**"Operation Canceled"**)**

**def** show\_records**(**cursor**,** query**):**

**try:**

cursor**.**execute**(**query**)**

records **=** cursor**.**fetchall**()**

**if** cursor**.**rowcount **==** 0**:**

**print(**"No Matching Records"**)**

**return**

print\_header**()**

**for** record **in** records**:**

member **=** Member**().**create\_from\_record**(**record**)**

member**.**print\_all**()**

**return** records

**except** mysql**.**connector**.**Error **as** err**:**

**print(**err**)**

**def** show\_record**(**cursor**,** query**):**

**try:**

cursor**.**execute**(**query**)**

records **=** cursor**.**fetchall**()**

**if** cursor**.**rowcount **==** 0**:**

**print(**"No Matching Records"**)**

**return**

record **=** records**[**0**]**

member **=** Member**().**create\_from\_record**(**record**)**

member**.**print\_full**()**

**return** member

**except** mysql**.**connector**.**Error **as** err**:**

**print(**err**)**

**def** get\_and\_print\_member\_by\_id**(**cursor**):**

member\_id **=** **input(**"Enter the member id: "**)**

query **=** "select \* from {0} where id = {1}"**.format(**TABLE\_NAME**,** member\_id**)**

member **=** show\_record**(**cursor**,** query**)**

**return** member

**def** extend\_membership\_by\_id**(**database**,** cursor**):**

member **=** get\_and\_print\_member\_by\_id**(**cursor**)**

**if** member **is** **not** **None:**

query **=** "update {0} set valid\_till = date\_add(valid\_till, INTERVAL {1} MONTH) where id={2}"\

**.format(**TABLE\_NAME**,**MEMBERSHIP\_PERIOD**,**member**.**member\_id**)**

confirm **=** **input(**"Confirm Update (Y/N): "**).**lower**()**

**if** confirm **==** 'y'**:**

cursor**.**execute**(**query**)**

database**.**commit**()**

**print(**"Operation Successful"**)**

**else:**

**print(**"Operation Cancelled"**)**

**def** edit\_by\_member\_by\_id**(**database**,** cursor**):**

member **=** get\_and\_print\_member\_by\_id**(**cursor**)**

**if** member **is** **not** **None:**

query **=** "update {0} set"**.format(**TABLE\_NAME**)**

**print(**"Input new values (leave blank to keep previous value)"**)**

name **=** **input(**"Enter new name: "**)**

**if** **len(**name**)** **>** 0**:**

query **+=** " name='{0}',"**.format(**name**)**

address **=** **input(**"Enter new address: "**)**

**if** **len(**address**)** **>** 0**:**

query **+=** " address='{0}',"**.format(**address**)**

phone **=** **input(**"Enter phone number: "**)**

**if** **len(**phone**)** **>** 0**:**

query **+=** " phone='{0}',"**.format(**phone**)**

query **=** query**[**0**:-**1**]** **+** " where id={0}"**.format(**member**.**member\_id**)**

confirm **=** **input(**"Confirm Update (Y/N): "**).**lower**()**

**if** confirm **==** 'y'**:**

cursor**.**execute**(**query**)**

database**.**commit**()**

**print(**"Operation Successful"**)**

**else:**

**print(**"Operation Cancelled"**)**

**def** delete\_by\_member\_id**(**database**,** cursor**):**

member **=** get\_and\_print\_member\_by\_id**(**cursor**)**

**if** member **is** **not** **None:**

confirm **=** **input(**"Confirm Deletion (Y/N): "**).**lower**()**

**if** confirm **==** 'y'**:**

query **=** "delete from {0} where id={1}"**.format(**TABLE\_NAME**,** member**.**member\_id**)**

cursor**.**execute**(**query**)**

database**.**commit**()**

**print(**"Operation Successful"**)**

**else:**

**print(**"Operation Cancelled"**)**

**def** member\_menu**(**database**,** cursor**):**

**while** **True:**

**print()**

**print(**"=============================="**)**

**print(**"==========Member Menu========="**)**

**print(**"=============================="**)**

**print()**

**print(**"1. New Member"**)**

**print(**"2. Show Member Details by name"**)**

**print(**"3. Show Member details by member id"**)**

**print(**"4. Show Member details by address"**)**

**print(**"5. Show Member details by phone number"**)**

**print(**"6. Extend Membership"**)**

**print(**"7. Edit Member Details"**)**

**print(**"8. Delete Member record"**)**

**print(**"9. View all customers"**)**

**print(**"0. Go Back"**)**

choice **=** **int(input(**"Enter your choice: "**))**

**if** choice **==** 1**:**

add\_member**(**database**,** cursor**)**

**elif** choice **==** 2**:**

name **=** **input(**"Enter the name: "**).**lower**()**

query **=** "select \* from {0} where name like '%{1}%'"**.format(**TABLE\_NAME**,** name**)**

show\_records**(**cursor**,** query**)**

**elif** choice **==** 3**:**

get\_and\_print\_member\_by\_id**(**cursor**)**

**elif** choice **==** 4**:**

address **=** **input(**"Enter the address: "**).**lower**()**

query **=** "select \* from {0} where address like '%{1}%'"**.format(**TABLE\_NAME**,** address**)**

show\_records**(**cursor**,** query**)**

**elif** choice **==** 5**:**

phone **=** **input(**"Enter the phone number: "**)**

query **=** "select \* from {0} where phone like '%{1}%'"**.format(**TABLE\_NAME**,** phone**)**

show\_records**(**cursor**,** query**)**

**elif** choice **==** 6**:**

extend\_membership\_by\_id**(**database**,** cursor**)**

**elif** choice **==** 7**:**

edit\_by\_member\_by\_id**(**database**,** cursor**)**

**elif** choice **==** 8**:**

delete\_by\_member\_id**(**database**,** cursor**)**

**elif** choice **==** 9**:**

query **=** "select \* from {0}"**.format(**TABLE\_NAME**)**

show\_records**(**cursor**,** query**)**

**elif** choice **==** 0**:**

**break**

**else:**

**print(**"Invalid choice (Press 0 to go back)"**)**

## issue\_return.py

**from** datetime **import** datetime

**from** myprint **import** print\_bar

TABLE\_NAME **=** "issue\_return"

**class** **IssueReturn:**

**def** \_\_init\_\_**(**self**):**

self**.**record\_id **=** 0

self**.**book\_id **=** ""

self**.**book\_title **=** ""

self**.**member\_id **=** ""

self**.**member\_name **=** ""

self**.**issue\_date **=** ""

self**.**return\_date **=** ""

**def** create**(**self**,** record\_id**,** book\_id**,** book\_title**,** member\_id**,** member\_name**,** issue\_date**,** return\_date**):**

self**.**record\_id **=** record\_id

self**.**book\_id **=** book\_id

self**.**book\_title **=** book\_title

self**.**member\_id **=** member\_id

self**.**member\_name **=** member\_name

self**.**issue\_date **=** issue\_date

self**.**return\_date **=** return\_date

**return** self

**def** create\_from\_record**(**self**,** record**):**

self**.**record\_id **=** record**[**'id'**]**

self**.**book\_id **=** record**[**'book\_id'**]**

self**.**book\_title **=** record**[**'book\_title'**]**

self**.**member\_id **=** record**[**'member\_id'**]**

self**.**member\_name **=** record**[**'member\_name'**]**

self**.**issue\_date **=** record**[**'issue\_date'**]**

self**.**return\_date **=** record**[**'return\_date'**]**

**return** self

**def** print\_all**(**self**):**

**print(str(**self**.**record\_id**).**ljust**(**3**),**

**str(**self**.**book\_id**).**ljust**(**15**),**

self**.**book\_title**[**0**:**15**].**ljust**(**15**),**

**str(**self**.**member\_id**).**ljust**(**15**),**

self**.**member\_name**[**0**:**15**].**ljust**(**15**),**

self**.**issue\_date**.**strftime**(**"%d-%b-%y"**).**ljust**(**15**),**

**(**self**.**return\_date**.**strftime**(**"%d %b %y"**)** **if** self**.**return\_date **is** **not** **None** **else** "None"**).**ljust**(**15**))**

**def** print\_full**(**self**):**

print\_bar**()**

**print(**"Record #"**,** self**.**record\_id**)**

**print(**"Book id: "**,** self**.**book\_id**)**

**print(**"Book Title: "**,** self**.**book\_title**)**

**print(**"Member id: "**,** self**.**member\_id**)**

**print(**"Member Name: "**,**self**.**member\_name**)**

**print(**"Issue Date: "**,** self**.**issue\_date**.**strftime**(**"%d %b %y"**))**

**print(**"Return date: "**,** self**.**return\_date**.**strftime**(**"%d %b %y"**)** **if** self**.**return\_date **is** **not** **None** **else** **None)**

print\_bar**()**

**def** create\_record**(**book**,** member**):**

record\_id **=** **None**

issue\_date **=** datetime**.**now**()**

return\_date **=** **None**

**return** IssueReturn**().**create**(**record\_id**,**book**.**book\_id**,** book**.**title**,** member**.**member\_id**,** member**.**name**,** issue\_date**,** return\_date**)**

**def** print\_header**():**

**print(**"="**\***100**)**

**print(**"id"**.**ljust**(**3**),**

"bookid"**.**ljust**(**15**),**

"book title"**.**ljust**(**15**),**

"member id"**.**ljust**(**15**),**

"member name"**.**ljust**(**15**),**

"issue date"**.**ljust**(**15**),**

"return date"**.**ljust**(**15**))**

**print(**"="**\***100**)**

**def** create\_issue\_return\_table**(**database**):**

cursor **=** database**.**cursor**()**

cursor**.**execute**(**"DROP table if exists {0}"**.format(**TABLE\_NAME**))**

cursor**.**execute**(**"create table {0} ("

"id int primary key auto\_increment,"

"book\_id int,"

"book\_title varchar(50),"

"member\_id int,"

"member\_name varchar(50),"

"issue\_date datetime,"

"return\_date datetime)"**.format(**TABLE\_NAME**))**

## issue\_returns.py

**from** datetime **import** datetime

**import** mysql

**from** issue\_return **import** create\_record**,** TABLE\_NAME**,** create\_issue\_return\_table**,** print\_header**,**IssueReturn

**from** books **import** get\_and\_print\_book\_by\_id**,** change\_book\_status

**from** members **import** get\_and\_print\_member\_by\_id

NUMBER\_OF\_RECORDS\_PER\_PAGE **=** 10

BOOK\_ISSUE\_PERIOD **=** 30 # DAYS

LATE\_FINE\_PER\_DAY **=** 1 # Rupees

**def** add\_record**(**database**,** cursor**):**

book **=** get\_and\_print\_book\_by\_id**(**cursor**)**

**if** book **is** **not** **None:**

**if** **not** book**.**available**:**

**print(**"The Book Is Not Available"**)**

**return**

member **=** get\_and\_print\_member\_by\_id**(**cursor**)**

**if** member **is** **not** **None:**

record **=** create\_record**(**book**,**member**)**

confirm **=** **input(**"Complete the operation? (Y/N) "**).**lower**()**

**if** confirm **==** 'y'**:**

query **=** "insert into {0}(book\_id, book\_title,member\_id, member\_name,issue\_date) " \

"values ({1},'{2}',{3},'{4}','{5}')"**.**\

**format(**TABLE\_NAME**,**record**.**book\_id**,**record**.**book\_title**,**record**.**member\_id**,**

record**.**member\_name**,**record**.**issue\_date**.**strftime**(**"%Y-%m-%d %H:%M:%S"**))**

**try:**

cursor**.**execute**(**query**)**

database**.**commit**()**

**except** mysql**.**connector**.**Error **as** err**:**

create\_issue\_return\_table**(**database**)**

cursor**.**execute**(**query**)**

database**.**commit**()**

change\_book\_status**(**database**,**cursor**,**book**.**book\_id**,False)**

**print(**"Operation Successful"**)**

**def** show\_record**(**cursor**,** query**):**

**try:**

cursor**.**execute**(**query**)**

records **=** cursor**.**fetchall**()**

**if** cursor**.**rowcount **==** 0**:**

**print(**"No Matching Records"**)**

**return**

record **=** records**[**0**]**

issue\_record **=** IssueReturn**().**create\_from\_record**(**record**)**

issue\_record**.**print\_full**()**

**return** issue\_record

**except** mysql**.**connector**.**Error **as** err**:**

**print(**err**)**

**def** show\_records**(**cursor**,** query**):**

**try:**

cursor**.**execute**(**query**)**

records **=** cursor**.**fetchall**()**

**if** cursor**.**rowcount **==** 0**:**

**print(**"No Matching Records"**)**

**return**

print\_header**()**

**for** record **in** records**:**

issue\_record **=** IssueReturn**().**create\_from\_record**(**record**)**

issue\_record**.**print\_all**()**

**return** records

**except** mysql**.**connector**.**Error **as** err**:**

**print(**err**)**

**def** get\_and\_print\_record\_by\_book\_id**(**cursor**):**

book\_id **=** **input(**"Enter the book id: "**)**

query **=** "select \* from {0} where book\_id = {1} order by issue\_date desc limit 1"**.format(**TABLE\_NAME**,** book\_id**)**

issue\_record **=** show\_record**(**cursor**,** query**)**

**return** issue\_record

**def** return\_book**(**database**,** cursor**):**

issue\_record **=** get\_and\_print\_record\_by\_book\_id**(**cursor**)**

**if** issue\_record **is** **not** **None:**

late\_delta **=** datetime**.**now**()** **-** issue\_record**.**issue\_date

**if** late\_delta**.**days **>** BOOK\_ISSUE\_PERIOD**:**

**print(**"Calculated Fine: "**,** late\_delta**.**days **\*** LATE\_FINE\_PER\_DAY **,** " Rupees"**)**

confirm **=** **input(**"Complete the operation? (Y/N) "**).**lower**()**

**if** confirm **==** 'y'**:**

return\_date **=** datetime**.**now**()**

query **=** "update {0} set return\_date='{1}' where id={2}"**.**\

**format(**TABLE\_NAME**,**return\_date**.**strftime**(**"%Y-%m-%d %H:%M:%S"**),**issue\_record**.**record\_id**)**

cursor**.**execute**(**query**)**

database**.**commit**()**

change\_book\_status**(**database**,**cursor**,**issue\_record**.**book\_id**,True)**

**def** delete\_record\_by\_book\_id**(**database**,** cursor**):**

issue\_record **=** get\_and\_print\_record\_by\_book\_id**(**cursor**)**

**if** issue\_record **is** **not** **None:**

confirm **=** **input(**"Complete the operation? (Y/N) "**).**lower**()**

**if** confirm **==** 'y'**:**

query **=** "delete from {0} where id={1}"**.format(**TABLE\_NAME**,**issue\_record**.**record\_id**)**

cursor**.**execute**(**query**)**

database**.**commit**()**

**print(**"Operation Successful"**)**

**def** issue\_return\_menu**(**database**,** cursor**):**

**while** **True:**

**print()**

**print(**"============================"**)**

**print(**"==========Books Menu========="**)**

**print(**"============================"**)**

**print()**

**print(**"1. Issue A Book"**)**

**print(**"2. Show Issue Record By Book ID"**)**

**print(**"3. Return A Book"**)**

**print(**"4. Show Books Issued By A Member"**)**

**print(**"5. Show Books Whose Return Date has Passed"**)**

**print(**"6. Delete Record"**)**

**print(**"7. View all Records"**)**

**print(**"0. Go Back"**)**

choice **=** **int(input(**"Enter your choice: "**))**

**if** choice **==** 1**:**

add\_record**(**database**,** cursor**)**

**elif** choice **==** 2**:**

get\_and\_print\_record\_by\_book\_id**(**cursor**)**

**elif** choice **==** 3**:**

return\_book**(**database**,**cursor**)**

**elif** choice **==** 4**:**

member\_id **=** **int(input(**"Enter the member id: "**))**

query **=** "select \* from {0} where member\_id={1} order by issue\_date desc"**.format(**TABLE\_NAME**,** member\_id**)**

show\_records**(**cursor**,** query**)**

**elif** choice **==** 5**:**

present\_date **=** datetime**.**now**()**

query **=** "select \* from {0} where date\_add(issue\_date, INTERVAL {1} DAY) < '{1}'"**.**\

**format(**TABLE\_NAME**,**BOOK\_ISSUE\_PERIOD**,**present\_date**.**strftime**(**"%Y-%m-%d"**))**

show\_records**(**cursor**,** query**)**

**elif** choice **==** 6**:**

delete\_record\_by\_book\_id**(**database**,** cursor**)**

**elif** choice **==** 7**:**

query **=** "select \* from {0}"**.format(**TABLE\_NAME**)**

show\_records**(**cursor**,** query**)**

**elif** choice **==** 0**:**

**break**

**else:**

**print(**"Invalid choice (Press 0 to go back)"**)**

## database.py

**import** mysql**.**connector

HOST **=** "localhost"

USER **=** "scott"

PASSWORD **=** "scotttiger"

DATABASE **=** "library"

**def** get\_database**():**

**try:**

database **=** mysql**.**connector**.**connect**(**

host**=**HOST**,**

user**=**USER**,**

password**=**PASSWORD**,**

database**=**DATABASE

**)**

cursor **=** database**.**cursor**(**dictionary**=True)**

**return** database**,** cursor

**except** mysql**.**connector**.**Error**:**

**return** **None,** **None**

## main.py

**import** sys

**from** books **import** book\_menu

**from** issue\_returns **import** issue\_return\_menu

**from** members **import** member\_menu

**from** myprint **import** print\_center**,** input\_center

**from** database **import** get\_database

**if** \_\_name\_\_ **==** '\_\_main\_\_'**:**

database**,** cursor **=** get\_database**()**

**if** database **is** **None:**

**print(**"The Database does not exist or not accessible."**)**

sys**.exit(**1**)**

**while** **True:**

**print()**

print\_center**(**"=============================="**)**

print\_center**(**"=====Apka Guruji Library====="**)**

print\_center**(**"=============================="**)**

print\_center**(**"1. Issue/Return Register"**)**

print\_center**(**"2. Manage Books"**)**

print\_center**(**"3. Manage Members"**)**

print\_center**(**"0. Exit"**)**

**print()**

choice **=** **int(**input\_center**(**"Enter your choice: "**))**

**if** choice **==** 1**:**

issue\_return\_menu**(**database**,**cursor**)**

**elif** choice **==** 2**:**

book\_menu**(**database**,** cursor**)**

**elif** choice **==** 3**:**

member\_menu**(**database**,** cursor**)**

**elif** choice **==** 0**:**

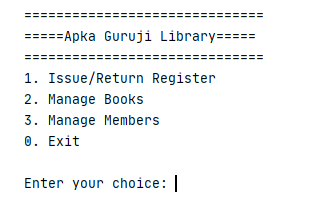
**break**

**else:**

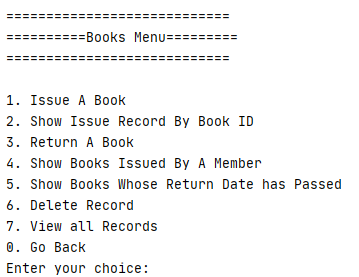
**print(**"Invalid choice (Press 0 to exit)"**)**

print\_center**(**"GoodBye"**)**

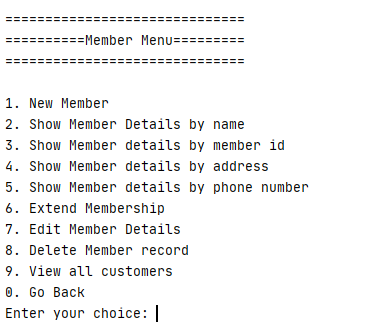
# Output



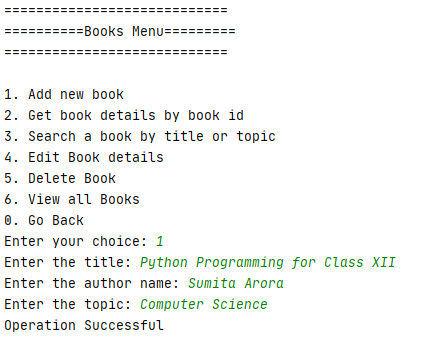
Main Menu



Books Menu



Member Menu



Adding A New Book

# Hardware and Software Requirements

The software being developed in Python programming language scores high in portability, i.e. runs on almost all different types of computers with any operating system.

## Software Requirements

* Operating Systems: Windows, Linux, Macintosh
* MySQL Server
* Latest Python Interpreter along with following modules: dateutil, mysql,

## Hardware requirements

Any computer capable of running above mentioned software can run our program. We tested our program in following hardware configurations:

* Processor: Intel, AMD
* Ram: 2GB, 4GB, 8GB
* Hard Disk: 80 GB, 160 GB, 500 GB, 1 TB
* Keyboard, Mouse: Wired or Wireless
* Monitor: 14 inch or above VGA or HDMI

# Bibliography

ApkaGuruji.com. (n.d.). *Preparing Project Report.* Retrieved from apkaguruji..com: apkaguruji.com

Wikipedia. (n.d.). *System Development Life Cycle.* Retrieved from Wikepedia.com: https://en.wikipedia.org/wiki/Systems\_development\_life\_cycle