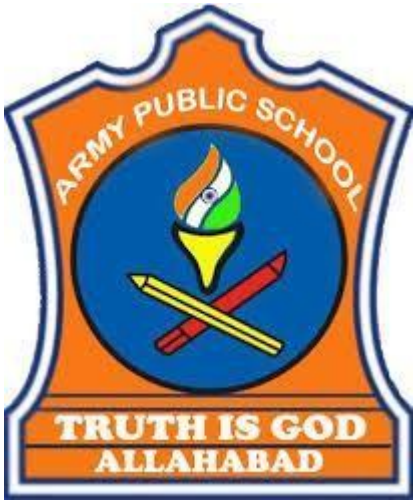
ARMY PUBLIC SCHOOL NEW CANTT



ACADEMIC YEAR : 2022-23

PROJECT REPORT ON

LIBRARY MANAGEMENT SYSTEM

ROLL NO :

NAME : AYUSH SINGH

CLASS : XII E

SUBJECT : COMPUTER SCIENCE

SUB CODE : 083

PROJECT GUIDE : **Mrs. SMITA VERMA**

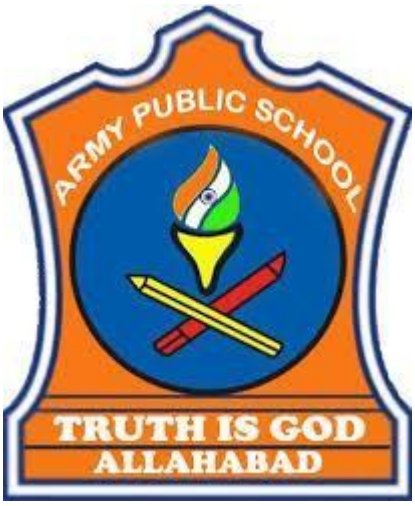
PGT (CS)

ARMY PUBLIC SCHOOL, NEW CANTT

PRAYGRAJ

UTTAR PRADESH

ARMY PUBLIC SCHOOL



CERTIFICATE

This is to certify that Ayush Singh CBSE Roll No : _____ have successfully completed the project work entitled **LIBRARY MANAGEMENT SYSTEM** in the subject Computer Science (083) laid down in the regulations of CBSE for the purpose of Practical Examination in Class XII to be held in Army Public School, New Cantt Praygraj on _____ .

Signature of External Examiner

Name: _____

Examiner Number: _____

(SMITA VERMA)
PGT Computer Science

TABLE OF CONTENTS [T O C]

ACKNOWLEDGEMENT

Apart from the efforts of me, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

I express deep sense of gratitude to almighty God for giving me strength for the successful completion of the project.

I express my heartfelt gratitude to my computer science teacher **Mrs. Smita Verma** for constant encouragement while carrying out this project.

I gratefully acknowledge the contribution of the individuals who contributed in bringing this project up to this level, who continues to look after me despite my flaws,

I express my deep sense of gratitude to the luminary The Principal, Army Public School New Cantt ,Prayagraj who has been continuously motivating and extending their helping hand to us.

I express my sincere thanks to the academician The Vice Principal, Army Pubic School New Cantt ,Prayagraj, for constant encouragement and the guidance provided during this project

I am overwhelmed to express my thanks to The Administrative Officer for providing me an infrastructure and moral support while carrying out this project in the school.

The guidance and support received from all the members who contributed and who are contributing to this project, was vital for the success of the project. I am grateful for their constant support and help.

PROJECT ON LIBRARY MANAGEMENT SYSTEM

INTRODUCTION

The "**LIBRARY MANAGEMENT SYSTEM**" has been developed to override the problems prevailing in practicing manual system. This software is supposed to reduce the hardships faced by this existing system. Moreover, this system is designed for the particular need of organizations to carry out operations in a smooth and efficient manner.

Instead of building a **CLI(Command Line Interface)** for this project which could have been quite simple to build, we have chosen to build a descent looking **GUI(Graphical User Interface)** for this "**LIBRARY MANAGEMENT SYSTEM**" which itself proves that being user-friendly it can be used in solving some real-world problems faced by many of the organizations. No formal knowledge is needed for the user to use this system. Thus, it proves to be user-friendly.

Along with all these features, this system works on **client-server architecture** i.e., the students (clients) can access some of the information from the server (if running).

OBJECTIVES OF THE PROJECT

The objective of the project is to let the students apply the programming knowledge into a real-world situation/problem and exposed the students how programming skills helps in developing a good software.

1. Write programs utilizing modern software tools.
2. Apply object oriented programming principles effectively when developing small to medium sized projects.
3. Write effective procedural code to solve small to medium sized problems.
4. Students will demonstrate a breadth of knowledge in computer science, as exemplified in the areas of systems, theory and software development.
5. Students will demonstrate ability to conduct a research or applied Computer Science project, requiring writing and presentation skills which exemplify scholarly style in computer science.

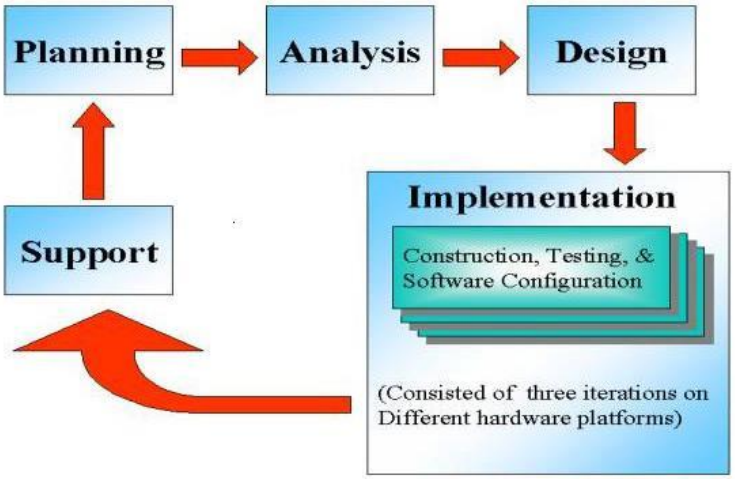
PROPOSED SYSTEM

Today one cannot afford to rely on the fallible human beings of be really wants to stand against today's merciless competition where not to wise saying "to err is human" no longer valid, it's outdated to rationalize your mistake. So, to keep pace with time, to bring about the best result without malfunctioning and greater efficiency so to replace the unending heaps of flies with a much sophisticated hard disk of the computer.

One has to use the data management software. Software has been an ascent in atomization various organization. Many software products working are now in markets, which have helped in making the organizations work easier and efficiently. Data management initially had to maintain a lot of ledgers and a lot of paper work has to be done but now software product on this organization has made their work faster and easier. Now only this software has to be loaded on the computer and work can be done.

This prevents a lot of time and money. The work becomes fully automated and any information regarding the organization can be obtained by clicking the button. Moreover, now it's an age of computers of and automating such an organization gives the better look.

SYSTEM DEVELOPMENT LIFE CYCLE (SDLC)



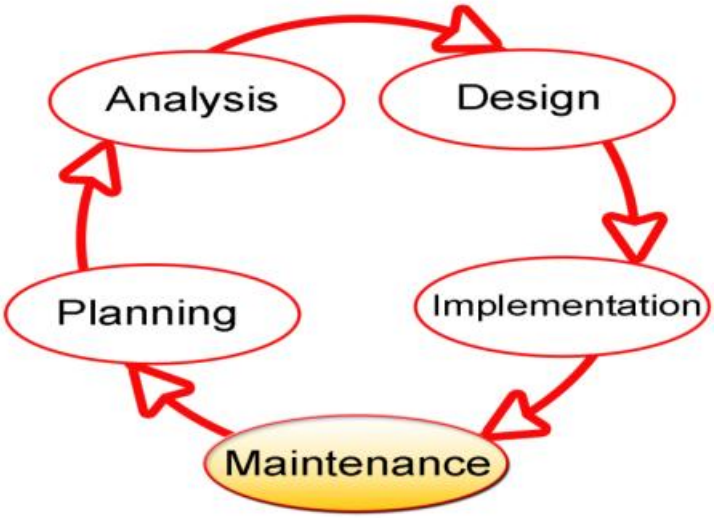
The systems development life cycle is a project management technique that divides complex projects into smaller, more easily managed segments or phases. Segmenting projects allows managers to verify the successful completion of project phases before allocating resources to subsequent phases.

Software development projects typically include initiation, planning, design, development, testing, implementation, and maintenance phases. However, the phases may be divided differently depending on the organization involved.

For example, initial project activities might be designated as request, requirements-definition, and planning phases, or initiation, concept-development, and planning phases. End users of the system under development should be involved in reviewing the output of each phase to ensure the system is being built to deliver the needed functionality.

PHASES OF SYSTEM DEVELOPMENT LIFE CYCLE

PICTORIAL REPRESENTATION OF SDLC:



INITIAL PHASE

The Initiation Phase begins when a business sponsor identifies a need or an opportunity.

The purpose of the Initiation Phase is to:

- Identify and validate an opportunity to improve business accomplishments of the organization or a deficiency related to a business need.
- Identify significant assumptions and constraints on solutions to that need.
- Recommend the exploration of alternative concepts and methods to satisfy the need including questioning the need for technology, i.e., will a change in the business process offer a solution?
- Assure executive business and executive technical sponsorship. The Sponsor designates a Project Manager and the business need is documented in a

Concept Proposal. The Concept Proposal includes information about the business process and the relationship to the Agency/Organization.

- Infrastructure and the Strategic Plan. A successful Concept Proposal results in a Project Management Charter which outlines the authority of the project manager to begin the project

SYSTEM CONCEPT DEVELOPMENT PHASE

The System Concept Development Phase begins after a business need or opportunity is validated by the Agency/Organization Program Leadership and the Agency/Organization CIO.

The purpose of the System Concept Development Phase is to:

- Determine the feasibility and appropriateness of the alternatives.
- Identify system interfaces.
- Identify basic functional and data requirements to satisfy the business need.
- Establish system boundaries; identify goals, objectives, critical success factors, and performance measures.
- Evaluate costs and benefits of alternative approaches to satisfy the basic functional requirements
- Assess project risks
- Identify and initiate risk mitigation actions, and Develop high-level technical architecture, process models, data models, and a concept of operations. This phase explores potential technical solutions within the context of the business need.
- The ITPR must be approved by the State CIO before the project can move forward.

PLANNING PHASE

The planning phase is the most critical step in completing development, acquisition, and maintenance projects. Careful planning, particularly in the early stages of a project, is necessary to coordinate activities and manage project risks effectively. The depth and formality of project plans should be commensurate with the characteristics and risks of a given project. Project plans refine the information gathered during the initiation phase by further identifying the specific activities and resources required to complete a project.

A critical part of a project manager's job is to coordinate discussions between user, audit, security, design, development, and network personnel to identify and document as many functional, security, and network requirements as possible. During this phase, a plan is developed that documents the approach to be used and includes a discussion of methods, tools, tasks, resources, project schedules, and user input. Personnel assignments, costs, project schedule, and target dates are established.

A Project Management Plan is created with components related to acquisition planning, configuration management planning, quality assurance planning, concept of operations, system security, verification and validation, and systems engineering management planning.

REQUIREMENTS ANALYSIS PHASE

This phase formally defines the detailed functional user requirements using high-level requirements identified in the Initiation, System Concept, and Planning phases. It also delineates the requirements in terms of data, system performance, security, and maintainability requirements for the system. The requirements are defined in this phase to a level of detail sufficient for systems design to proceed. They need to be measurable, testable, and relate to the business need or

opportunity identified in the Initiation Phase. The requirements that will be used to determine acceptance of the system are captured in the Test and Evaluation Master Plan.

The purposes of this phase are to:

- Further define and refine the functional and data requirements and document them in the Requirements Document,
- Complete business process re-engineering of the functions to be supported (i.e., verify what information drives the business process, what information is generated, who generates it, where does the information go, and who processes it),
- Develop detailed data and process models (system inputs, outputs, and the process.
- Develop the test and evaluation requirements that will be used to determine acceptable system performance.

DESIGN PHASE

The design phase involves converting the informational, functional, and network requirements identified during the initiation and planning phases into unified design specifications that developers use to script programs during the development phase. Program designs are constructed in various ways. Using a top-down approach, designers first identify and link major program components and interfaces, then expand design layouts as they identify and link smaller subsystems and connections. Using a bottom-up approach, designers first identify and link minor program components and interfaces, then expand design layouts as they identify and link larger systems and connections. Contemporary design techniques often use prototyping tools that build mock-up designs of items such as application screens, database layouts, and system architectures. End users, designers, developers, database managers, and network administrators should review and refine the prototyped designs in an iterative process until they agree on an acceptable design. Audit, security, and quality assurance personnel should be

involved in the review and approval process. During this phase, the system is designed to satisfy the functional requirements identified in the previous phase. Since problems in the design phase could be very expensive to solve in the later stage of the software development, a variety of elements are considered in the design to mitigate risk. These include:

- Identifying potential risks and defining mitigating design features.
- Performing a security risk assessment.
- Developing a conversion plan to migrate current data to the new system.
- Determining the operating environment.
- Defining major subsystems and their inputs and outputs.
- Allocating processes to resources.
- Preparing detailed logic specifications for each software module. The result is a draft System Design Document which captures the preliminary design for the system.
- This document receives a rigorous review by Agency technical and functional representatives to ensure that it satisfies the business requirements. Concurrent with the development of the system design, the Agency Project Manager begins development of the Implementation Plan, Operations and Maintenance Manual, and the Training Plan.

DEVELOPMENT PHASE

The development phase involves converting design specifications into executable programs. Effective development standards include requirements that programmers and other project participants discuss design specifications before programming begins. The procedures help ensure programmers clearly

understand program designs and functional requirements. Programmers use various techniques to develop computer programs. The large transaction oriented programs associated with financial institutions have traditionally been developed using procedural programming techniques. Procedural programming involves the line-by-line scripting of logical instructions that are combined to form a program. Effective completion of the previous stages is a key factor in the success of the Development phase. The Development phase consists of:

- Translating the detailed requirements and design into system components.
- Testing individual elements (units) for usability.
- Preparing for integration and testing of the IT system

INTEGRATION AND TEST PHASE

- Subsystem integration, system, security, and user acceptance testing is conducted during the integration and test phase. The user, with those responsible for quality assurance, validates that the functional requirements, as defined in the functional requirements document, are satisfied by the developed or modified system. OIT Security staff assess the system security and issue a security certification and accreditation prior to installation/implementation.

Multiple levels of testing are performed, including:

- Testing as a deployed system with end users working together with contract personnel
- Operational testing by the end user alone performing all functions. Requirements are traced throughout testing,a final Independent Verification & Validation evaluation is performed and all documentation is reviewed and accepted prior to acceptance of the system.

IMPLEMENTATION PHASE

This phase is initiated after the system has been tested and accepted by the user. In this phase, the system is installed to support the intended business functions. System performance is compared to performance objectives established during the planning phase. Implementation includes user notification, user training, installation of hardware, installation of software onto production computers, and integration of the system into daily work processes. This phase continues until the system is operating in production in accordance with the defined user requirements.

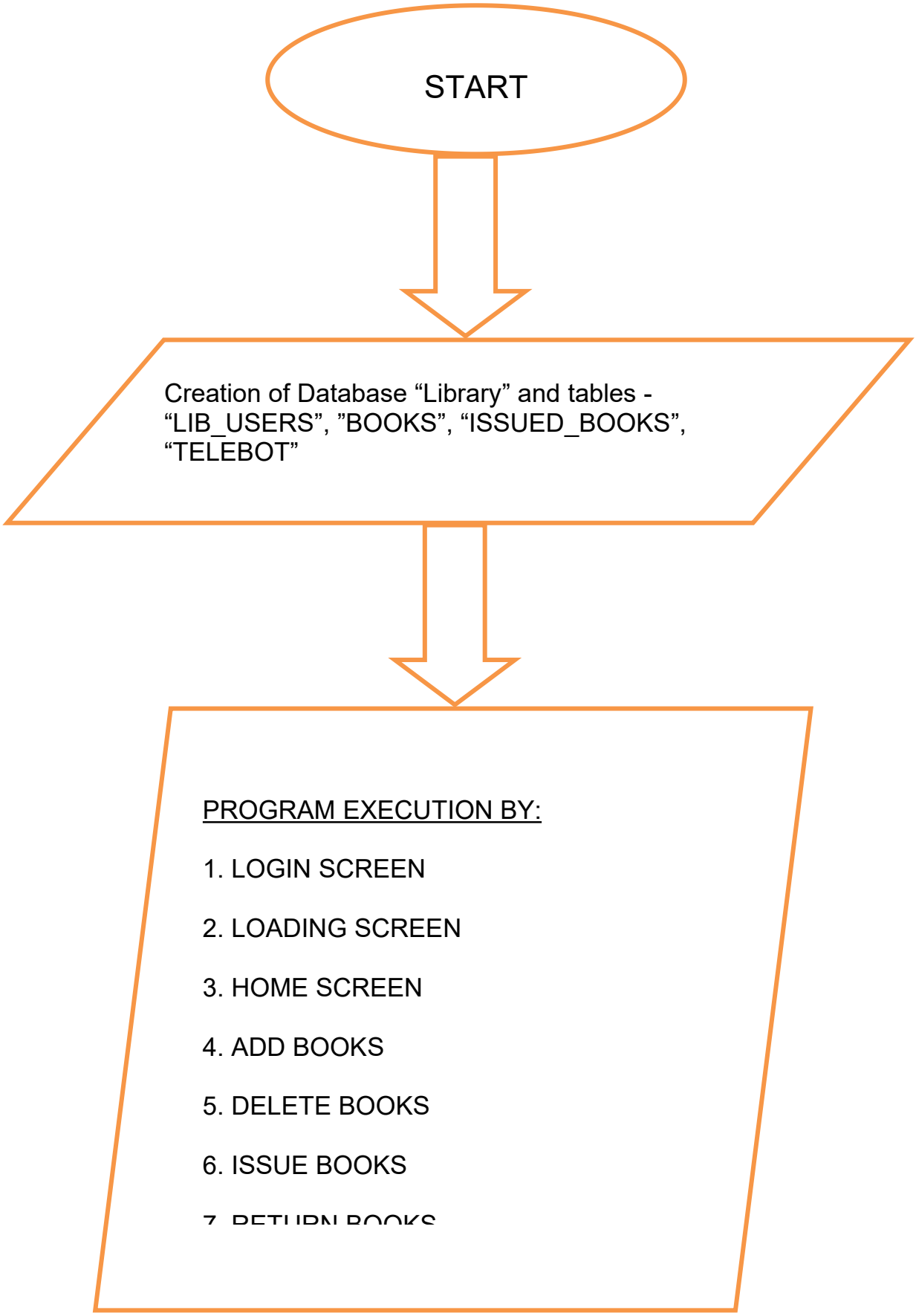
OPERATIONS AND MAINTENANCE PHASE

The system operation is ongoing. The system is monitored for continued performance in accordance with user requirements and needed system modifications are incorporated. Operations continue as long as the system can be effectively adapted to respond to the organization's needs. When modifications or changes are identified, the system may reenter the planning phase.

The purpose of this phase is to:

- Operate, maintain, and enhance the system.
- Certify that the system can process sensitive information.
- Conduct periodic assessments of the system to ensure the functional requirements continue to be satisfied.

FLOW CHART


START
Creation of Database “Library” and tables - “LIB_USERS”, ”BOOKS”, “ISSUED_BOOKS”, “TELEBOT”
PROGRAM EXECUTION BY:
1. LOGIN SCREEN
2. LOADING SCREEN
3. HOME SCREEN
4. ADD BOOKS
5. DELETE BOOKS
6. ISSUE BOOKS

CODE

```
import tkinter as tk
from time import sleep
import mysql.connector as conn
from tkinter import messagebox
import os
import csv
import requests
import pickle
import base64
from datetime import date
import customtkinter as ck
import hashlib
from tkinter import filedialog as fd
from os.path import exists


class main:
    def __init__(self) -> None:
        ck.set_appearance_mode("dark")
        ck.set_default_color_theme("blue")

        self.mysql_password = ''
        self.mysql_username = ''
        self.db = ''
        self.cursor = ''

        self.root = ck.CTk()
        self.root.title("Library Management System (APS New Cantt, Prayagraj)")
        self.root.bind("<Key>", lambda key: self.listener(key,self.root))
        self.root.withdraw()

        self.win_del = ck.CTk()
        self.win_del.title("Delete Books")
        self.win_del.bind("<Key>", lambda key: self.listener(key,self.win_del))
        self.win_del.withdraw()

        self.win_add = ck.CTk()
        self.win_add.title("Add Books")
        self.win_add.bind("<Key>", lambda key: self.listener(key,self.win_add))
        self.win_add.withdraw()

        self.win_return = ck.CTk()
        self.win_return.title("Return Books")
        self.win_return.bind("<Key>", lambda key: self.listener(key,self.win_return))
        self.win_return.withdraw()

        self.issue = ck.CTk()
        self.issue.title("Issue Books")
        self.issue.bind("<Key>", lambda key: self.listener(key,self.issue))
        self.issue.withdraw()

        self.reset = ck.CTk()
        self.reset.title("Reset Password")
        self.reset.bind("<Key>", lambda key: self.listener(key,self.reset))
        self.reset.withdraw()

        self.fine = 50
        self.fields = ["BOOK_ID",'BOOK_NAME','AUTHOR_NAME','BOOK_PRICE','QTY']
```

```python
# ---------------------------------------MAIN FUNCTIONS---------------------------------------

def listener(self, key, window):
    win_title = str(window.wm_title())

    if key.keysym == "F11":
        if window.wm_attributes()[7] == 0:
            window.attributes("-fullscreen", True)
        else:
            window.attributes("-fullscreen", False)

    elif key.keysym == "F1":
        self.reset_fine_gui()

    elif key.keysym == "F2":
        self.reset_password_gui()

    elif key.keysym == "F3":
        if window.appearance_mode == 1:
            window.set_appearance_mode("light")
        else:
            window.set_appearance_mode("dark")

    elif key.keysym == "F5":
        self.add_books_csv()


def on_closing(self):
    confirm = messagebox.askquestion("Confirm", "Do you really want to quit?")
    if confirm.lower() == "yes":
        exit()

def win_issue_on_closing(self):
    self.issue.withdraw()

def reset_on_closing(self):
    self.reset.withdraw()

def reset_fine_on_closing(self):
    self.reset_fine.withdraw()

def win_return_on_closing(self):
    self.win_return.withdraw()

def win_del_on_closing(self):
    self.win_del.withdraw()

def win_add_on_closing(self):
    self.win_add.withdraw()

def abt_on_closing(self):
    self.about.withdraw()

def login_on_closing(self):
    exit()
```

```
    def add_books(self, book_id, book_name, author, bookprice, qty):
        sql = 'INSERT INTO BOOKS(BOOK_ID, BOOK_NAME, AUTHOR_NAME, BOOK_PRICE, QTY) VALUES(%s, %s, %s,
%s, %s)'
        val = (book_id.get(), book_name.get(), author.get(), bookprice.get(), qty.get())

        try:
            self.cursor.execute(sql, val)
            self.db.commit()
            messagebox.showinfo("Success", "Book added successfully!")
        except:
            messagebox.showerror("Error", "Something Went Wrong!")

    def delete_book(self, del_book_id):
        try:
            self.cursor.execute('delete from books where BOOK_ID='+del_book_id.get())
            self.db.commit()
            messagebox.showinfo("Success", 'Book deleted successfully!')
        except:
            messagebox.showerror("Error", "Something Went Wrong!")

    def issue_books(self, std_name, class_, section, roll, issue_bid, issue_date):
        self.cursor.execute("SELECT QTY FROM BOOKS WHERE BOOK_ID="+issue_bid.get())
        result = self.cursor.fetchall()
        if (len(result) != 0) and (result[0][-1] != 0):
            self.cursor.execute("select STUDENT_ID from ISSUED_BOOKS;")

            try:
                std_id = self.cursor.fetchall()[-1][0]
            except:
                std_id = 0

            try:
                self.cursor.execute("SELECT BOOK_PRICE FROM BOOKS WHERE BOOK_ID="+issue_bid.get())
                bprice = self.cursor.fetchall()[0][0]
            except IndexError:
                messagebox.showerror("Error", "Book Not Available. Check the list of books again.")

            try:
                sql = 'INSERT INTO ISSUED_BOOKS(STUDENT_ID, STUDENT_NAME, CLASS, SECTION, ROLL_NO,
BOOK_ID, BOOK_PRICE, ISSUE_DATE) VALUES(%s, %s, %s, %s, %s, %s, %s, %s)'
                val = ((std_id+1), std_name.get(), class_.get().upper(), section.get().upper(),
int(roll.get()), int(issue_bid.get()), int(bprice), issue_date.get())

                self.cursor.execute(sql, val)
                self.db.commit()

                self.cursor.execute("UPDATE BOOKS SET QTY=QTY-1 WHERE BOOK_ID="+issue_bid.get())
                self.db.commit()

                messagebox.showinfo("Success", "Book Issued Successfully!")
            except:
                messagebox.showerror("Error", "Something went wrong.")

        else:
            messagebox.showinfo("Info", "Sorry, the book is not currently available!")

    def return_books(self, return_date, return_stdid):
        try:
            return_date_splitted = return_date.get().split('-')
            date1 = date(int(return_date_splitted[0]), int(return_date_splitted[1]),
int(return_date_splitted[2]))

            self.cursor.execute("SELECT * FROM ISSUED_BOOKS")
            rows = self.cursor.fetchall()

            for i in rows:
                if int(i[0]) == int(return_stdid.get()):
                    date2 = i[7]

            days = (date1 - date2).days
            if days > 7:
                self.cursor.execute("UPDATE ISSUED_BOOKS SET FINE=%s WHERE STUDENT_ID=%s", ((days-
7)*self.fine, return_stdid.get()))
                self.db.commit()

            self.cursor.execute("SELECT BOOK_ID FROM ISSUED_BOOKS WHERE
STUDENT_ID="+return_stdid.get())
            return_bid = self.cursor.fetchall()[-1][0]

            self.cursor.execute("UPDATE ISSUED_BOOKS SET RETURN_DATE=%s WHERE STUDENT_ID=%s",
(return_date.get(), return_stdid.get()))
            self.db.commit()

            self.cursor.execute("UPDATE BOOKS SET QTY=QTY+1 WHERE BOOK_ID="+str(return_bid))
            self.db.commit()

            messagebox.showinfo("Info", "Success!")

        except Exception as e:
            messagebox.showerror("Error", f"Something Went Wrong!\n{e}")
```

```
    def view_books(self):
        self.cursor.execute("DESC BOOKS")
        fields = []

        for i in self.cursor.fetchall():
            fields.append(i[0])

        self.cursor.execute("SELECT * FROM BOOKS")
        rows = self.cursor.fetchall()

        filepath = fd.asksaveasfilename(title="Save as")

        try:
            with open(filepath, 'w') as file:
                writer = csv.writer(file)
                writer.writerow(fields)
                writer.writerows(rows)

            messagebox.showinfo(title="Successfull", message="CSV file has been successfully created!")

            sleep(1)
            os.startfile(filepath)
        except:
            messagebox.showerror("Error", "Something Went Wrong!")

    def view_issued_books(self):
        self.cursor.execute('DESC ISSUED_BOOKS')
        fields = []

        for i in self.cursor.fetchall():
            fields.append(i[0])

        self.cursor.execute("SELECT * FROM ISSUED_BOOKS")
        rows = self.cursor.fetchall()

        try:
            path = fd.asksaveasfile().name
        except:
            messagebox.showerror("Error","Please try to save the file with different name (if it is
running in background)!")

        try:
            with open(path, 'w') as file:
                writer = csv.writer(file)
                writer.writerow(fields)
                writer.writerows(rows)

            messagebox.showinfo(title="Successfull", message="CSV file made!!")
            sleep(1)
            os.startfile(path)

        except:
            messagebox.showerror("Error", "Something Went Wrong!")

    def reset_password(self, new_passwrd, username, old_passwrd):
        self.cursor.execute("SELECT * FROM LIB_USERS")
        query1 = self.cursor.fetchall()[0]
        new_passwrd_hash = hashlib.sha256(new_passwrd.encode()).hexdigest()
        old_passwrd_hash = hashlib.sha256(old_passwrd.encode()).hexdigest()

        try:
            self.cursor.execute("UPDATE LIB_USERS SET PASSWORD=%s WHERE USERNAME=%s AND PASSWORD=%s",
(new_passwrd_hash,username,old_passwrd_hash))
            self.db.commit()
        except:
            messagebox.showerror("Error", "Something went wrong.")

        self.cursor.execute("SELECT * FROM LIB_USERS")
        query2 = self.cursor.fetchall()[0]

        if new_passwrd_hash == old_passwrd_hash:
            messagebox.showinfo("Info", "Old Password and New Password are same!")

        elif query1 == query2:
            messagebox.showerror("Error", "Username or Old Password is/are incorrect.")

        else:
            messagebox.showinfo("Info", "Password Changed Successfully.")
```

```
    def add_books_csv(self):
        with open("csv_books.csv","r") as file:
            reader = csv.reader(file)
            fields = next(reader)
            data = [i for i in reader]

            if data == []:
                messagebox.showinfo('Info',"Please enter some data in "csv_books.csv file!"')

            elif data[0][0] == 'BOOK_ID':
                messagebox.showinfo("Info","Please enter column names in the very first row of the csv
file!")

            else:
                try:
                    for i in data:
                        self.cursor.execute("INSERT INTO BOOKS VALUES(%s, %s, %s, %s, %s)",
(i[0],i[1],i[2],i[3],i[4]))
                        self.db.commit()
                    messagebox.showinfo("Success","Data inserted successfully!!")

                except:
                    messagebox.showerror("Error","Something went wrong!")


    # ----------------------------------------MAIN FUNCTIONS END-------------------------------



    # ----------------------------------------GUI START----------------------------------------

    def main_window(self):
        self.root.deiconify()
        self.root.state("zoomed")
        self.root.protocol("WM_DELETE_WINDOW", self.on_closing)

        menu = tk.Menu(self.root)

        help = tk.Menu(menu, tearoff = 0)
        menu.add_cascade(label ='Help', menu = help)
        help.add_command(label ='Change Fine Amount    F1', command = self.reset_fine_gui)
        help.add_command(label ='Reset Password        F2', command = self.reset_password_gui)
        help.add_command(label="Add Books (via csv) F5", command=self.add_books_csv)

        ck.CTkLabel(self.root, text="Welcome", text_font=("Roboto",25)).place(relx=0.25, rely=0.1,
relheight=0.1, relwidth=0.5)

        add_btn = ck.CTkButton(self.root, text_font=("Roboto",16), text="Add Book",
command=self.add_books_gui)
        add_btn.place(relx=0.28, rely=0.26, relwidth=0.45, relheight=0.1)

        delete_btn = ck.CTkButton(self.root, text_font=("Roboto",16), text="Delete Book",
command=self.delete_book_gui)
        delete_btn.place(relx=0.28, rely=0.38, relwidth=0.45, relheight=0.1)

        view_btn = ck.CTkButton(self.root, text_font=("Roboto",16), text="Download List of Books",
command=self.view_books)
        view_btn.place(relx=0.28, rely=0.50, relwidth=0.45, relheight=0.1)

        issue_btn = ck.CTkButton(self.root, text_font=("Roboto",16), text="Issue Book to Student",
command=self.issue_books_gui)
        issue_btn.place(relx=0.28, rely=0.62, relwidth=0.45, relheight=0.1)

        return_btn = ck.CTkButton(self.root, text_font=("Roboto",16), text="Return Book",
command=self.return_books_gui)
        return_btn.place(relx=0.28, rely=0.74, relwidth=0.45, relheight=0.1)

        view_issued = ck.CTkButton(self.root, text_font=("Roboto",16), text="View Issued Books",
command=self.view_issued_books)
        view_issued.place(relx=0.28, rely=0.86, relwidth=0.45, relheight=0.1)

        self.root.config(menu = menu)
        self.root.mainloop()
```

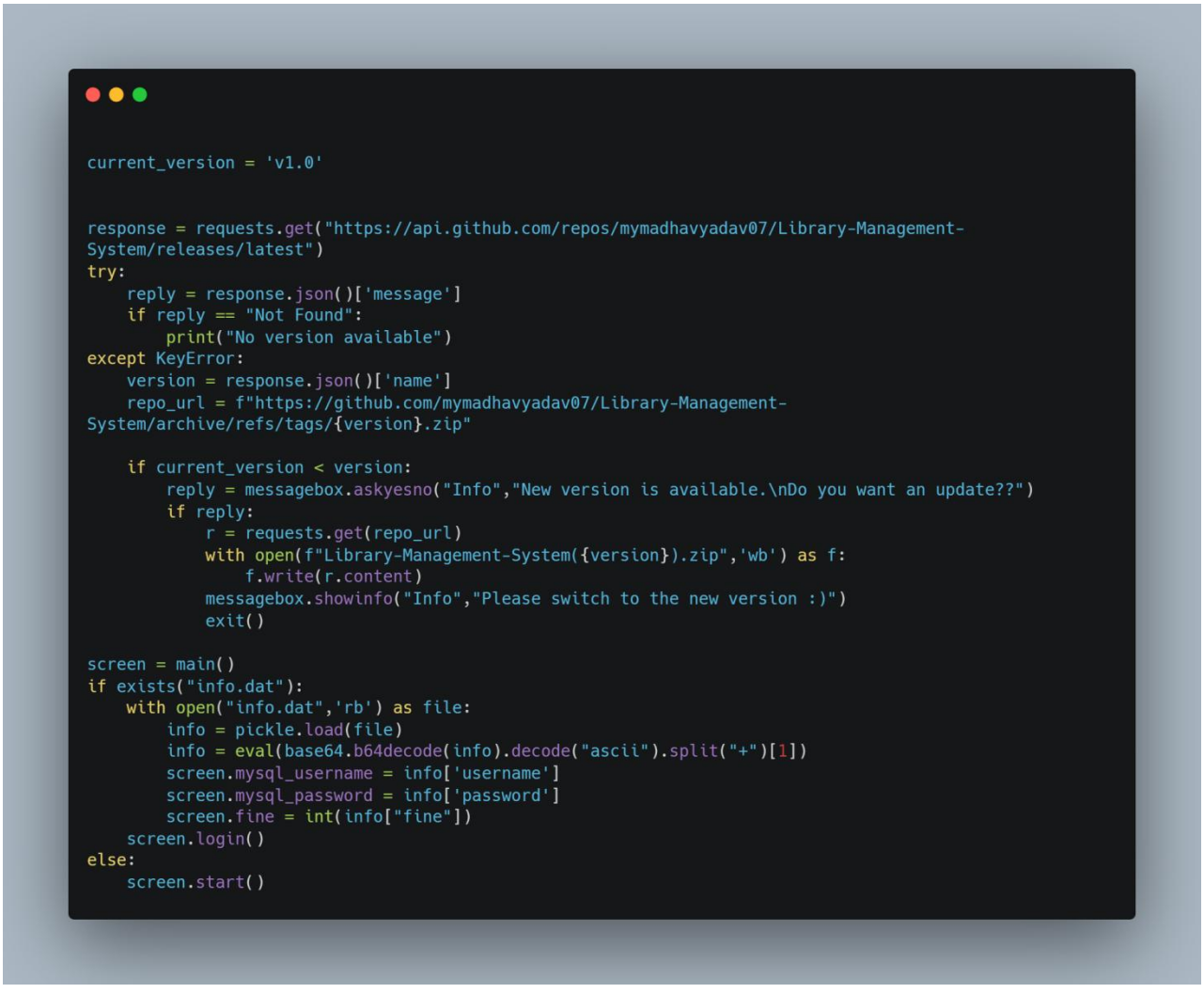
```
    def add_books_gui(self):
        self.win_add.protocol("WM_DELETE_WINDOW", self.win_add_on_closing)

        self.win_add.deiconify()
        self.win_add.state("zoomed")

        ck.CTkLabel(self.win_add, text="Add Books", text_font=("Roboto",30)).place(relx=0.21, rely=0.1,
relheight=0.1, relwidth=0.6)

        add_frame = ck.CTkFrame(self.win_add, bg='black')
        add_frame.place(relx=0.27, rely=0.4, relheight=0.5, relwidth=0.5)

        ck.CTkLabel(add_frame, text="Book ID:", text_font=("Roboto",18)).place(relx=0.05, rely=0.03)

        book_id = ck.CTkEntry(add_frame, width=480, text_font=("Roboto",18))
        book_id.place(relx=0.3, rely=0.03)

        ck.CTkLabel(add_frame, text="Book Name:", text_font=("Roboto",18)).place(relx=0.05, rely=0.16)

        book_name = ck.CTkEntry(add_frame, width=480, text_font=("Roboto",18))
        book_name.place(relx=0.3, rely=0.16)

        ck.CTkLabel(add_frame, text="Author Name:", text_font=("Roboto",18)).place(relx=0.05,
rely=0.29)

        author = ck.CTkEntry(add_frame, width=480, text_font=("Roboto",18))
        author.place(relx=0.3, rely=0.29)

        ck.CTkLabel(add_frame, text="Book Price:", text_font=("Roboto",18)).place(relx=0.05, rely=0.42)

        bookprice = ck.CTkEntry(add_frame, width=480, text_font=("Roboto",18))
        bookprice.place(relx=0.3, rely=0.42)

        ck.CTkLabel(add_frame, text="Book Qty:", text_font=("Roboto",18)).place(relx=0.05, rely=0.55)

        qty = ck.CTkEntry(add_frame, width=480, text_font=("Roboto",18))
        qty.place(relx=0.3, rely=0.55)

        submit = ck.CTkButton(add_frame, text="Submit", text_font=("Roboto",18), command=lambda:
self.add_books(book_id, book_name, author, bookprice, qty))
        submit.place(relx=0.40, rely=0.75, relwidth=0.4)

        self.win_add.mainloop()


    def delete_book_gui(self):
        self.win_del.protocol("WM_DELETE_WINDOW", self.win_del_on_closing)

        self.win_del.deiconify()
        self.win_del.state("zoomed")

        ck.CTkLabel(self.win_del, text="Delete Books", text_font=("Roboto",30)).place(relx=0.21,
rely=0.1, relheight=0.1, relwidth=0.6)

        del_frame = ck.CTkFrame(self.win_del, bg='black')
        del_frame.place(relx=0.27, rely=0.4, relheight=0.5, relwidth=0.5)

        ck.CTkLabel(del_frame, text="Book ID:", text_font=("Roboto",18)).place(relx=0.1, rely=0.4)

        del_book_id = ck.CTkEntry(del_frame, width=400, text_font=("Roboto",18))
        del_book_id.place(relx=0.36, rely=0.4)

        submit = ck.CTkButton(del_frame, text="Submit", text_font=("Roboto",18), command=lambda:
self.delete_book(del_book_id))
        submit.place(relx=0.3,rely=0.75, relwidth=0.5)

        self.win_del.mainloop()
```

```
    def issue_books_gui(self):
        self.issue.protocol("WM_DELETE_WINDOW", self.win_issue_on_closing)
        self.issue.deiconify()
        self.issue.state("zoomed")

        issue_frame = ck.CTkFrame(self.issue)
        issue_frame.place(relx=0.16, rely=0.28, relheight=0.65, relwidth=0.7)

        ck.CTkLabel(self.issue, text="Issue Books", text_font=("Roboto",25)).place(relx=0.21, rely=0.1,
relheight=0.1, relwidth=0.6)
        ck.CTkLabel(issue_frame, text="Student Name -", text_font=("Roboto",18)).place(relx=0.05,
rely=0.07)

        std_name = ck.CTkEntry(issue_frame, width=480, text_font=("Roboto",18))
        std_name.place(relx=0.43, rely=0.07)

        ck.CTkLabel(issue_frame, text="Class -", text_font=("Roboto",20)).place(relx=0.05, rely=0.18)

        class_ = ck.CTkEntry(issue_frame, width=480, text_font=("Roboto",18))
        class_.place(relx=0.43, rely=0.18)

        ck.CTkLabel(issue_frame, text="Section -", text_font=("Roboto",18)).place(relx=0.05, rely=0.31)

        section = ck.CTkEntry(issue_frame, width=480, text_font=("Roboto",18))
        section.place(relx=0.43, rely=0.31)

        ck.CTkLabel(issue_frame, text="Roll Number -", text_font=("Roboto",18)).place(relx=0.05,
rely=0.44)

        roll = ck.CTkEntry(issue_frame, width=480, text_font=("Roboto",18))
        roll.place(relx=0.43, rely=0.44)

        ck.CTkLabel(issue_frame, text="Book ID -", text_font=("Roboto",18)).place(relx=0.05, rely=0.57)

        issue_bid = ck.CTkEntry(issue_frame, width=480, text_font=("Roboto",18))
        issue_bid.place(relx=0.43, rely=0.57)

        ck.CTkLabel(issue_frame, text="Issue Date (YYYY-MM-DD) -", text_font=
("Roboto",18)).place(relx=0.05, rely=0.70)

        issue_date = ck.CTkEntry(issue_frame, width=480, text_font=("Roboto",18))
        issue_date.place(relx=0.43, rely=0.70)

        submit = ck.CTkButton(issue_frame, text="Submit", text_font=("Roboto",18), command=lambda:
self.issue_books(std_name, class_, section, roll, issue_bid, issue_date))
        submit.place(relx=0.3, rely=0.85, relwidth=0.5)

        self.issue.mainloop()

    def return_books_gui(self):
        self.win_return.protocol("WM_DELETE_WINDOW", self.win_return_on_closing)
        self.win_return.deiconify()
        self.win_return.state("zoomed")

        return_frame = ck.CTkFrame(self.win_return, bg='black')
        return_frame.place(relx=0.16, rely=0.28, relheight=0.65, relwidth=0.7)

        ck.CTkLabel(self.win_return, text="Return Books", text_font=("Roboto",25)).place(relx=0.21,
rely=0.1, relheight=0.1, relwidth=0.6)

        ck.CTkLabel(return_frame, bg='black', fg='lime', text="Student ID -", text_font=
("Roboto",20)).place(relx=0.05, rely=0.35)

        return_stdid = ck.CTkEntry(return_frame, width=480, text_font=("Roboto",18))
        return_stdid.place(relx=0.37, rely=0.35)

        ck.CTkLabel(return_frame, text="Return Date -", text_font=("Roboto",20)).place(relx=0.05,
rely=0.48)

        return_date = ck.CTkEntry(return_frame, width=480, text_font=("Roboto",18))
        return_date.place(relx=0.37, rely=0.48)

        submit = ck.CTkButton(return_frame, text="Submit", text_font=("Roboto",22), command=lambda:
self.return_books(return_date, return_stdid))
        submit.place(relx=0.3, rely=0.8, relwidth=0.5)

        self.win_return.mainloop()
```

```
    def reset_fine_gui(self):
        my_fine = ck.CTkInputDialog(self.root, title="Fine", text="Enter your desired fine.")
        my_fine = my_fine.get_input()

        try:
            self.fine = int(my_fine)
            messagebox.showinfo("Info","Fine Updated Successfully!")
            with open("info.dat",'wb') as file:
                data = {"username":self.mysql_username,"password":self.mysql_password,"fine":self.fine}
                data = f'hafhahgha[adauhaicnacb[]a[afag=+{str(data)}+ajkfhacuagugxuabjrbjakajajda=-=-
===[]agag'.encode("ascii")
                data = base64.b64encode(data)
                pickle.dump(data,file)
        except:
            messagebox.showerror("Error", "Something went wrong!")

    def reset_password_gui(self):
        self.reset.protocol("WM_DELETE_WINDOW", self.reset_on_closing)
        self.reset.deiconify()
        self.reset.state("zoomed")

        ck.CTkLabel(self.reset, text="Reset Password", text_font=("Roboto",25)).place(relx=0.45,
rely=0.1)

        ck.CTkLabel(self.reset, text="Username: ", text_font=("Roboto",20)).place(relx=0.26, rely=0.35)
        username = ck.CTkEntry(self.reset, width=480)
        username.place(relx=0.43, rely=0.35)

        ck.CTkLabel(self.reset, text="Old Password: ", text_font=("Roboto",20),).place(relx=0.26,
rely=0.45)

        old_passwrd = ck.CTkEntry(self.reset, width=480, show="*")
        old_passwrd.place(relx=0.43, rely=0.45)

        hide_var1 = ck.StringVar(value="on")

        def switch_event1():
            if hide_var1.get() == "on":
                old_passwrd.configure(show = "*")
            else:
                old_passwrd.configure(show = "")

        hide_switch = ck.CTkSwitch(master=self.reset, text="Hide Password", command=switch_event1,
                                   variable=hide_var1, onvalue="on", offvalue="off", text_font=
("Roboto",13))
        hide_switch.place(relx=0.76, rely=0.46)


        ck.CTkLabel(self.reset, text="New Password: ", text_font=("Roboto",20)).place(relx=0.26,
rely=0.55)
        new_passwrd = ck.CTkEntry(self.reset, width=480, show="*")
        new_passwrd.place(relx=0.43,rely=0.55)

        hide_var2 = ck.StringVar(value="on")

        def switch_event2():
            if hide_var2.get() == "on":
                new_passwrd.configure(show = "*")
            else:
                new_passwrd.configure(show = "")

        hide_switch2 = ck.CTkSwitch(master=self.reset, text="Hide Password", command=switch_event2,
                                   variable=hide_var2, onvalue="on", offvalue="off", text_font=
("Roboto",13))
        hide_switch2.place(relx=0.76, rely=0.55)


        reset_btn = ck.CTkButton(self.reset, text="Submit", width=240, text_font=("Roboto",18),
command=lambda: self.reset_password(new_passwrd.get(), username.get(), old_passwrd.get()))
        reset_btn.place(relx=0.46, rely=0.7)

        self.reset.mainloop()
```

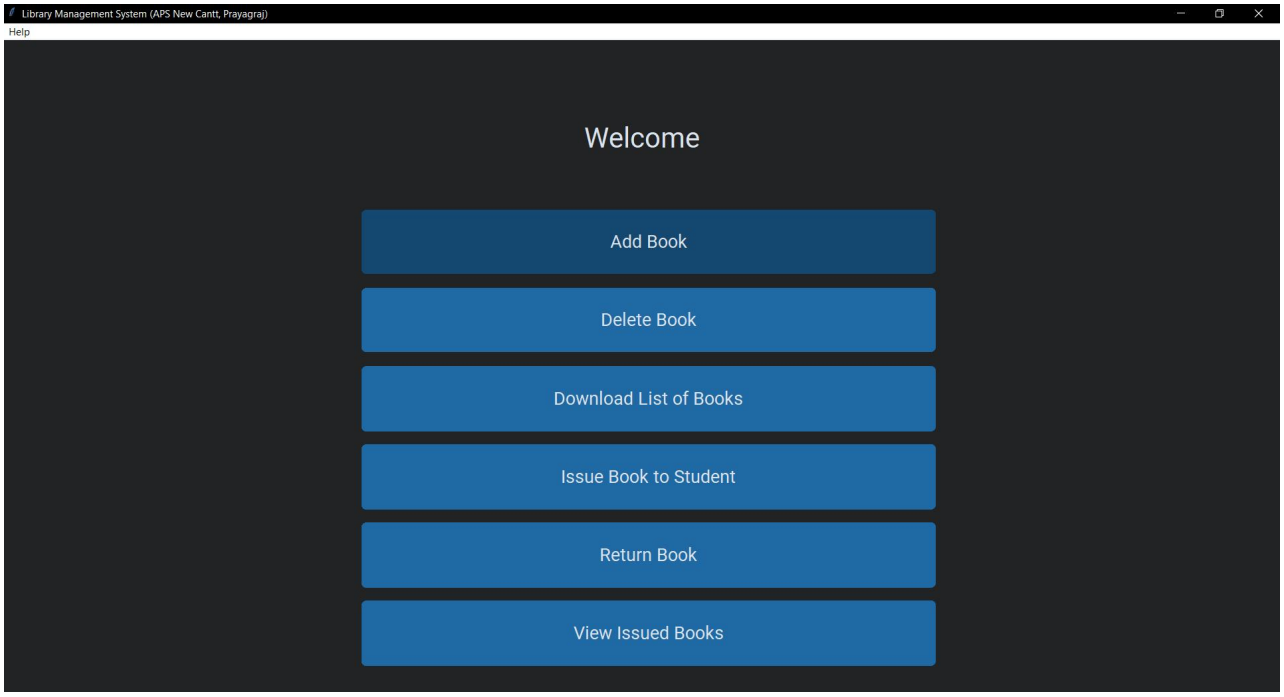
```
    def load(self):
        ck.set_appearance_mode("dark")

        load = ck.CTk()
        load.title("Loading")
        load.attributes("-fullscreen",True)
        load.config(bg = "black")

        ck.CTkLabel(load, text="Army Public School, New Cantt", text_font=("Courier New",25),
text_color="lime").place(relx=0.3, rely=0.05)
        ck.CTkLabel(load, text="Prayagraj", text_font=("Courier New",25),
text_color="lime").place(relx=0.42, rely=0.11)
        ck.CTkLabel(load, text="Library Management System", text_font=("Courier New",25),
text_color="lime").place(relx=0.32, rely=0.17)
        ck.CTkLabel(load, text="Loading...", text_font=("Courier New",25),
text_color="lime").place(relx=0.17, rely=0.43)

        box = 0.15
        for i in range(35):
            ck.CTkLabel(load, text="", bg_color='black', height=20, width=20).place(relx=(box)+0.02,
rely=0.5)
            box += 0.02

        for i in range(3):
            box = 0.15
            for j in range(35):
                ck.CTkLabel(load, text="", bg_color="lime", height=20, width=20).place(relx=(box)+0.02,
rely=0.5)
                sleep(0.02)
                box += 0.02
                load.update()
            box = 0.15

            for j in range(35):
                ck.CTkLabel(load, text="", bg_color='black', height=20, width=20).place(relx=
(box)+0.02, rely=0.5)
                sleep(0.02)
                box += 0.02
                load.update()
        load.destroy()


    def check_creds(self, screen, username, passwrd):
        try:
            self.db = conn.connect(host='localhost', user=self.mysql_username,
password=self.mysql_password, database='library')
            self.cursor = self.db.cursor()
        except conn.Error:
            reply = messagebox.askyesno("Error","Incorrect MySQL Username/Password!!\nDo you want to
update your MySQL credentials?")
            if reply:
                self.start()
            else:
                exit()

        self.cursor.execute("SELECT * FROM lib_users")
        result = self.cursor.fetchall()[0]
        passwrd_hash = hashlib.sha256(passwrd.get().encode()).hexdigest()

        if (result[0] == username.get()) and (result[1] == passwrd_hash):
            screen.destroy()
            self.load()
            self.main_window()
        else:
            messagebox.showerror("Error", "Invalid Credentials.")
```

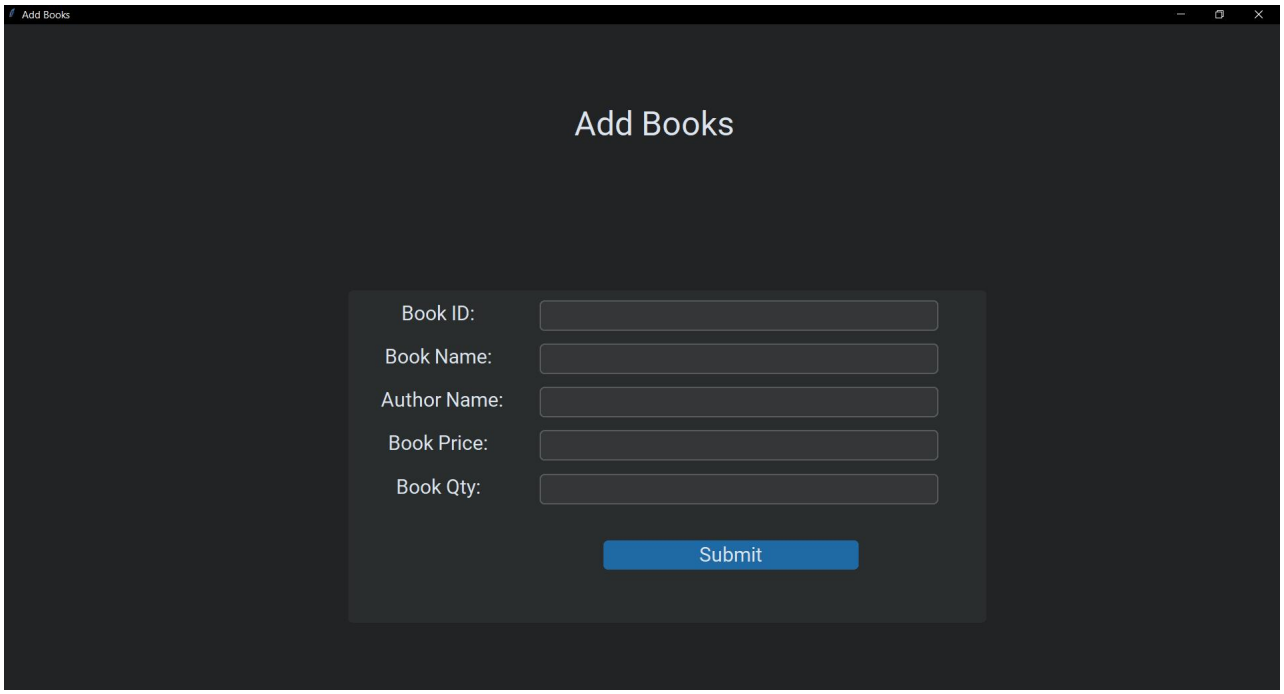
```
    def login(self):
        login_screen =ck.CTk()
        login_screen.title("Login")
        login_screen.geometry("950x400")
        login_screen.resizable(False,False)
        login_screen.protocol("WM_DELETE_WINDOW", self.login_on_closing)


        ck.CTkLabel(login_screen, text="Login", text_font=("Roboto",22,'bold')).place(relx=0.45,
rely=0.05)
        ck.CTkLabel(login_screen, text="Username:", text_font=("Roboto",18)).place(relx=0.15, rely=0.3)

        username = ck.CTkEntry(login_screen, width=360)
        username.place(relx=0.35, rely=0.3)

        ck.CTkLabel(login_screen, text="Password:", text_font=("Roboto",18)).place(relx=0.15, rely=0.6)
        passwrd = ck.CTkEntry(login_screen, width=360, show="*")
        passwrd.place(relx=0.35, rely=0.6)

        hide_var = ck.StringVar(value="on")

        def switch_event():
            if hide_var.get() == "on":
                passwrd.configure(show = "*")
            else:
                passwrd.configure(show = "")

        hide_switch = ck.CTkSwitch(master=login_screen, text="Hide Password", command=switch_event,
                                   variable=hide_var, onvalue="on", offvalue="off", text_font=
("Roboto",13))
        hide_switch.place(relx=0.75, rely=0.61)

        submit = ck.CTkButton(login_screen, text="Submit", text_font=("Roboto",12), command=lambda:
self.check_creds(login_screen, username, passwrd))
        submit.place(relx=0.45, rely=0.8)

        login_screen.mainloop()

    def start(self):
        my_user = ck.CTkInputDialog(self.root, title="Username", text="Enter your mysql username.")
        my_user = my_user.get_input()

        my_pass = ck.CTkInputDialog(self.root, title="Password", text="Enter your mysql password.")
        my_pass = my_pass.get_input()

        fine = ck.CTkInputDialog(self.root, title="Fine", text="Enter desired fine amount.")
        fine = fine.get_input()

        try:
            self.db = conn.connect(host='localhost', user=my_user, password=my_pass, db='library')
            self.cursor = self.db.cursor()

            self.mysql_password = my_pass
            self.mysql_username = my_user

            with open("info.dat",'wb') as file:
                data = {"username":my_user,"password":my_pass,"fine":fine}
                data = f'hafhahgha[adauhaicnacb[]a[afag=+{str(data)}+ajkfhacuagugxuabjrbjakajajda=-=-
===[]agag'.encode("ascii")
                data = base64.b64encode(data)
                pickle.dump(data,file)
        except:
            messagebox.showerror("Error","Something went wrong!!\nTry Checking your MySQL
Credentials!")

        self.login()

    # ------------------------------------------GUI END-----------------------------------------
----
```

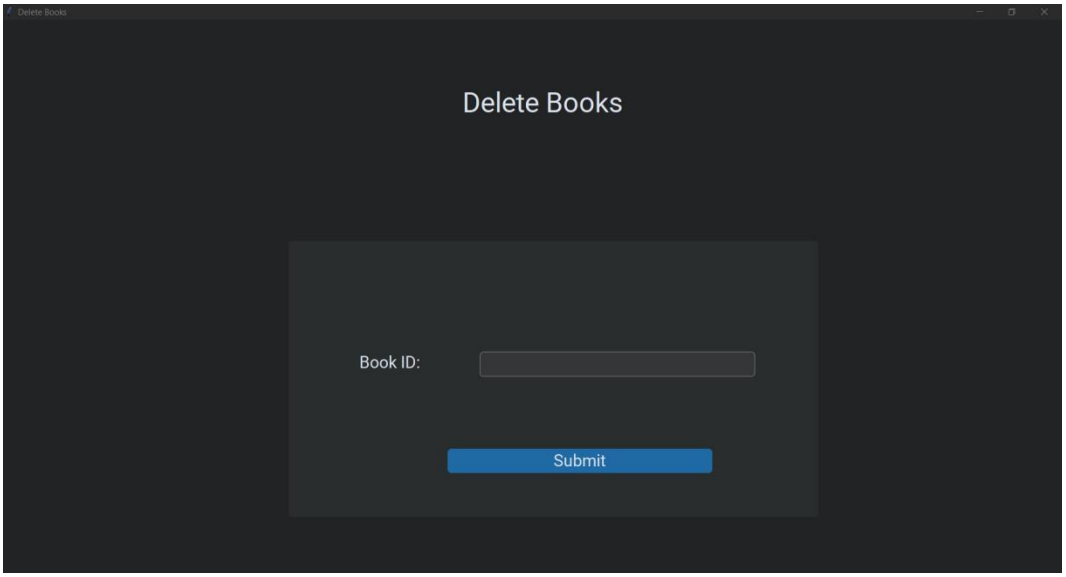


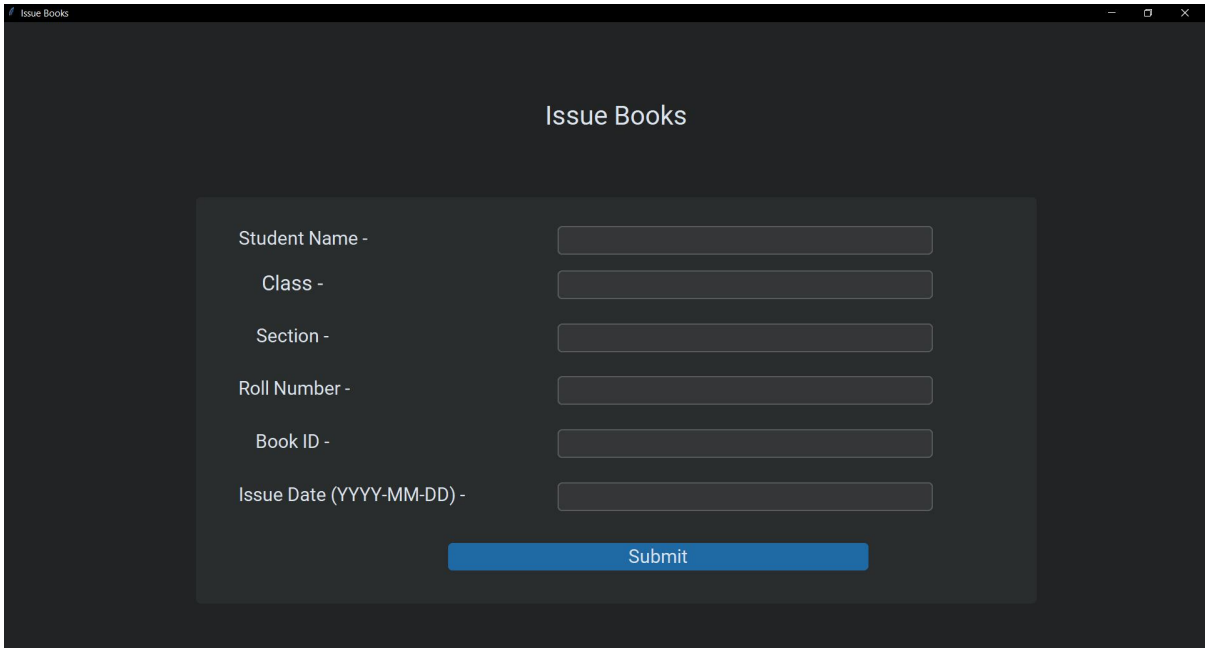
```
current_version = 'v1.0'


response = requests.get("https://api.github.com/repos/mymadhavyadav07/Library-Management-
System/releases/latest")
try:
    reply = response.json()['message']
    if reply == "Not Found":
        print("No version available")
except KeyError:
    version = response.json()['name']
    repo_url = f"https://github.com/mymadhavyadav07/Library-Management-
System/archive/refs/tags/{version}.zip"

    if current_version < version:
        reply = messagebox.askyesno("Info","New version is available.\nDo you want an update??")
        if reply:
            r = requests.get(repo_url)
            with open(f"Library-Management-System({version}).zip",'wb') as f:
                f.write(r.content)
            messagebox.showinfo("Info","Please switch to the new version :)")
            exit()

screen = main()
if exists("info.dat"):
    with open("info.dat",'rb') as file:
        info = pickle.load(file)
        info = eval(base64.b64decode(info).decode("ascii").split("+")[1])
        screen.mysql_username = info['username']
        screen.mysql_password = info['password']
        screen.fine = int(info["fine"])
    screen.login()
else:
    screen.start()
```

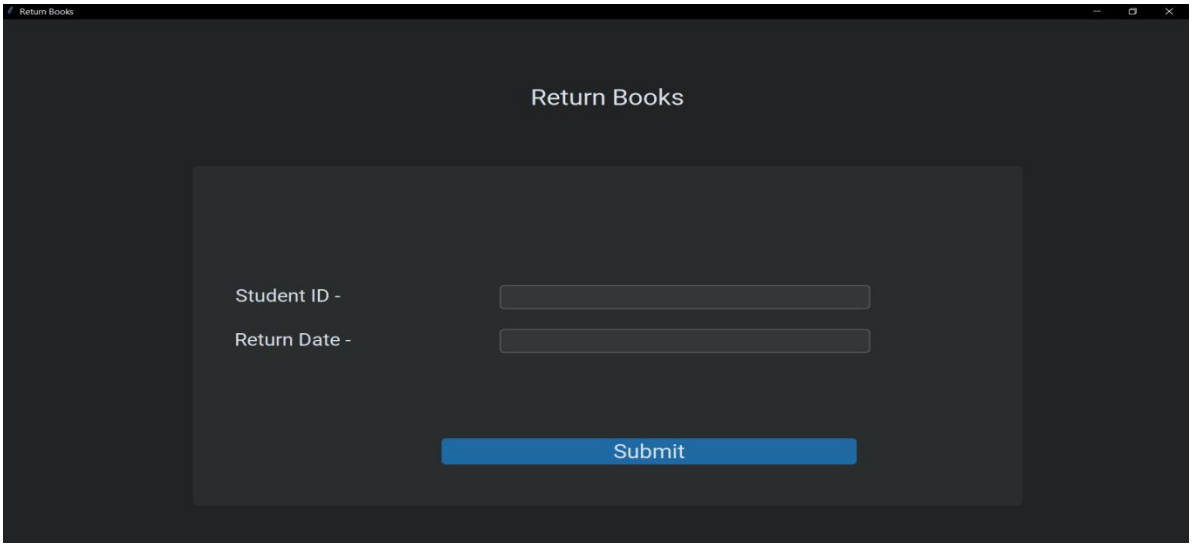
OUTPUT


Library Management System (APS New Cantt, Prayagraj)
Help
Welcome
Add Book
Delete Book
Download List of Books
Issue Book to Student
Return Book
View Issued Books


Add Books
Add Books
Book ID:
Book Name:
Author Name:
Book Price:
Book Qty:
Submit


Delete Books
Delete Books
Book ID:
Submit


Issue Books
Issue Books
Student Name -
Class -
Section -
Roll Number -
Book ID -
Issue Date (YYYY-MM-DD) -
Submit


Return Books
Return Books
Student ID -
Return Date -
Submit

INSTALLATION PROCEDURE

Library Management System :-

Pre-Requisites :-

---

1. You have to have the following softwares for the successful running of this software; which are

I) Python (Only for the First time), it is downloadable from 'www.python.org'.

II) MySQL (Only for the First time), it is downloadable from 'www.mysql.org'.

Installation :-

For Main Interface:

1. Run “source config.sql” command in MySQL command line client, it will setup the database for the project automatically.
2. Run “python3 -m pip install -r requirements.txt” to install the required python modules.
3. Run “python3 main.py” in your command prompt.
4. It would ask for the MySQL credentials.
5. After entering MySQL credentials, it would ask you to login default credentials (You can change them later) are -

Username: root

Password: toor

Telegram bot setup(*optional):

1. Use Telegram BotFather to create a telegram bot.
2. Copy the Bot’s API-token.
3. Now open “tele_creds.txt” and enter your MySQL Username, MySQL Password and bot API-Token there.
4. Finally, put the “tele_creds.txt” and “tele_server.py” file on some server class machine to keep your Telegram Bot server running forever.

TESTING

Software Testing is an empirical investigation conducted to provide stakeholders with information about the quality of the product or service under test, with respect to the context in which it is intended to operate. Software Testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs. It can also be stated as the process of validating and verifying that a software program/application/product meets the business and technical requirements that guided its design and development, so that it works as expected and can be implemented with the same characteristics. Software Testing, depending on the testing method employed, can be implemented at any time in the development process, however the most test effort is employed after the requirements have been defined and coding process has been completed. We have used some basic testing by passing our project through multiple test cases and have fixed the bugs found during the testing.

HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENTS:

I.OPERATING SYSTEM : WINDOWS 7 AND ABOVE

II. PROCESSOR : PENTIUM(ANY) OR AMD ATHALON(3800+ 4200+ DUAL CORE)

III. MOTHERBOARD : 1.845 OR 915,995 FOR PENTIUM 0R MSI K9M VVIA K8M800+8237R PLUS CHIPSET FOR AMD ATHALON

IV. RAM : 512 MB+

V. Hard disk : SATA 40 GB OR ABOVE

VI. CD/DVD r/w multi drive combo : (If back up required)

VII. FLOPPY DRIVE 1.44 MB : (If Backup required)

VIII. MONITOR 14.1 or 15 -17 inch

IX. Keyboard and mouse

SOFTWARE REQUIREMENTS:

I. Windows OS

II. Python

III. MySQL

BIBLIOGRAPHY

1. *Computer science With Python - Class XII By : Sumita Arora*
2. *Website:* **https://www.google.com**
3. *Website:* **https://github.com/TomSchimansky/CustomTkinter**
4. *Website:* **https://github.com/mymadhavyadav07/Library-Management-System**

***