CHANDIGARH UNIVERSITY, GHAURAN MOHALI

Subject Name-Programming in Python

Subject Code-24CAH-605

Submitted by-

Name- Tanisha Jain

UID- 24MCI10047

Branch- MCA(AI&ML)

Name: Tanisha Jain **Subject Code:** 24CAH-605

Branch: MCA(AI&ML) **Date of Submission:** 25-10-24

UID: 24MCI10047 **Semester:** 1st

Subject Name: Programming in Python **Section/Group-** 1/B

PROJECT NAME:

ONLINE LIBRARY MANAGEMENT SYSTEM

AIM:

The aim of this project is to design and implement a user-friendly **Online Library Management System** using Python's Tkinter library to manage books and provide basic authentication functionality. The system allows users to:

1. Authenticate via a login system: Ensure that only authorized users can access the library system.

2. Add and remove books: Maintain a list of books in the library, allowing the user to add new books or remove existing ones.

3. Display the current collection: Provide an interface where the list of books is visible, reflecting any changes made by the user.

Task to be done:

1. Design and Setup Login System:

   - Implement a login screen with input fields for username and password.
   - Use predefined credentials for authentication.
   - Display error messages for incorrect login attempts and success messages for valid login.

2. Create Main Dashboard (Library Management Interface) :

   - Design the main library interface with a blue background appropriate layout using the Tkinter library.
   - Set the title and use **Times New Roman** font for a professional look.

3. Implement Book Management Features:

   - Create an add book functionality:
     - Provide an input field where users can enter book names.
     - Add the entered book name to the library (stored in a list and displayed in a Listbox).
   - Implement remove book functionality:
     - Allow users to select a book from the displayed list and remove it.

4. Set Up Book Display:

   - Use a Listbox widget to display the current list of books in the library.
   - Ensure that any additions or deletions reflect in the Listbox immediately.

5. Implement Exit Option:

   - Add an "Exit" button to close the application.

6. Testing:

   - Test the login functionality with both valid and invalid credentials.
   - Test adding and removing books to ensure correct behavior.
   - Ensure the system functions properly across different user actions.

7. Documentation:

- Provide comments and documentation for the code, explaining the purpose of each function and key sections.

- Create a project report detailing the aim, tasks, features, and future improvements.

Process:

1. Planning and Design:

- Define Requirements: Identify the core functionalities needed for the library management system, such as user authentication, book management (adding and removing books), and user interface design.

2. Setting Up the Development Environment:

- Install Python: Ensure Python is installed on your system. You can download it from [python.org](https://www.python.org/).

- Install Tkinter: Tkinter comes pre-installed with Python. If it's not available, you may need to install it based on your operating system.

- Create Project Folder: Set up a dedicated folder for the project files.

3. Implementing the Login Functionality:

-Create Login Window:

- Use Tkinter to create a login window with fields for username and password.

- Add a button to handle login attempts.

- Validate Credentials:

- Store predefined login credentials in a dictionary.

- Implement a function to check user input against the stored credentials and provide feedback.

4. Creating the Main Library Dashboard:

- Initialize Dashboard Window:

- After successful login, create a new window for the library dashboard.

- Set the background color to blue and configure the layout.

- Add Components:

  - Use labels, buttons, and an Entry widget for user input and interaction.

  - Create a Listbox to display the list of books.

5. Implementing Book Management:

- Add Book Feature:

  - Create a function that retrieves the input from the Entry widget when the user wants to add a book.

  - Append the new book to a list and update the Listbox to reflect this addition.

- Remove Book Feature:

  - Implement functionality to select a book from the Listbox and remove it from the list.

  - Update the Listbox after the book is removed.

6. Finalizing the User Interface:

- Organize Layout: Use Tkinter's layout managers (`pack`, `grid`, or `place`) to arrange the components neatly.

- Set Fonts and Colors: Ensure all text uses the Times New Roman font, and maintain a cohesive color scheme throughout the application.

7. Testing the Application:

-Run the Program: Test the application to ensure all features work correctly.

- Login Testing: Verify both successful and unsuccessful login attempts

8. Documentation**:**

- Comment the Code: Provide comments explaining each function and key sections of the code.

- Create User Guide: Draft a simple user guide that explains how to use the application.

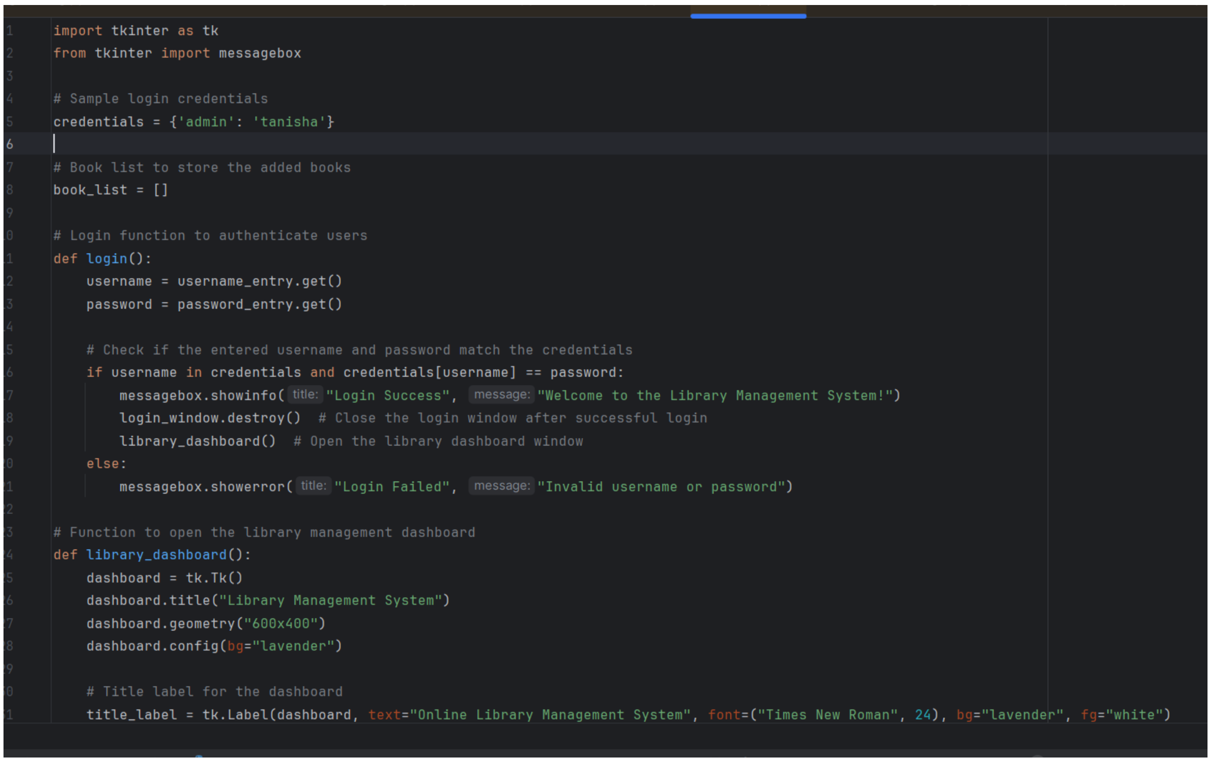
- Write a Project Report: Summarize the project, including its aim, functionalities, and potential future enhancements.

CODING

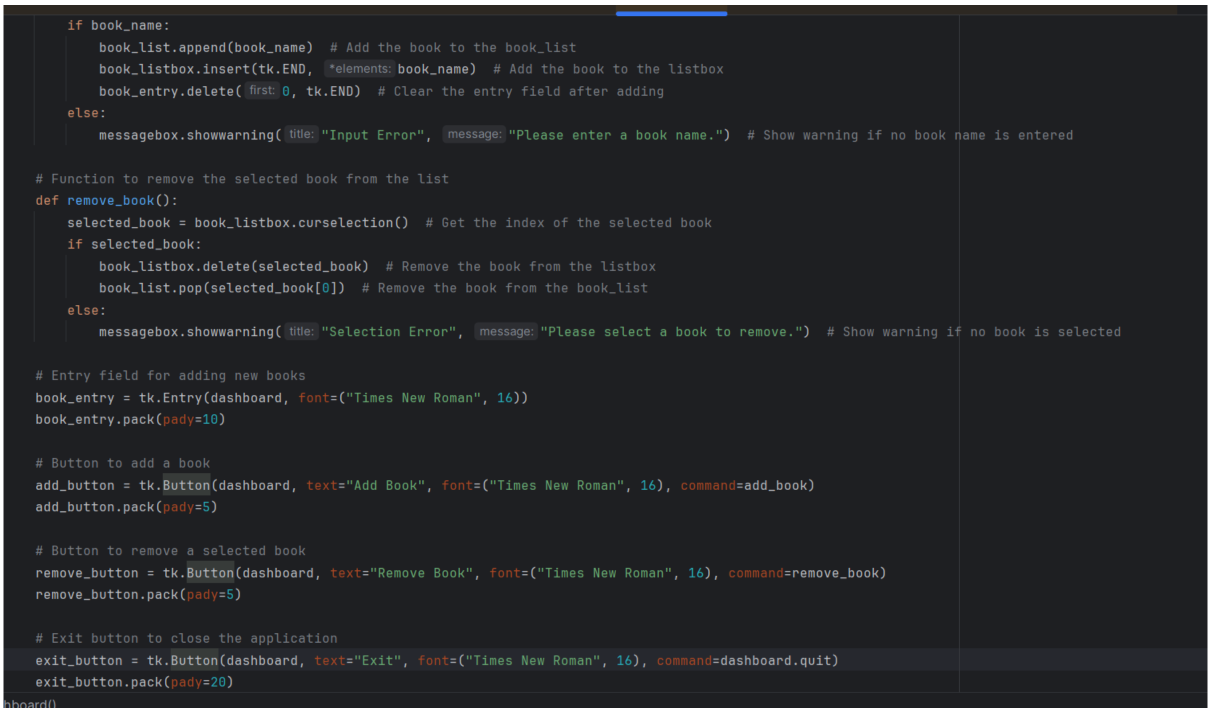
```
import tkinter as tk
from tkinter import messagebox

# Sample login credentials
credentials = {'admin': 'admin123'}  # Predefined login details

# Book list to store the added books
book_list = []

# Login function to authenticate users
def login():
    username = username_entry.get()
    password = password_entry.get()

    # Check if the entered username and password match the credentials
    if username in credentials and credentials[username] == password:
        messagebox.showinfo("Login Success", "Welcome to the Library
Management System!")
        login_window.destroy()  # Close the login window after successful login
        library_dashboard()  # Open the library dashboard window
    else:
        messagebox.showerror("Login Failed", "Invalid username or password")

# Function to open the library management dashboard
def library_dashboard():
    dashboard = tk.Tk()
    dashboard.title("Library Management System")
    dashboard.geometry("600x400")
    dashboard.config(bg="blue")

    # Title label for the dashboard
    title_label = tk.Label(dashboard, text="Online Library Management
System", font=("Times New Roman", 24), bg="blue", fg="white")
    title_label.pack(pady=20)

    # Frame to hold the book list components
    book_frame = tk.Frame(dashboard, bg="blue")
    book_frame.pack(pady=10)

    # Label for the book list
    book_list_label = tk.Label(book_frame, text="Book List", font=("Times New
Roman", 18), bg="blue", fg="white")
    book_list_label.pack()

    # Listbox to display the list of books
    book_listbox = tk.Listbox(book_frame, height=8, width=50, font=("Times
New Roman", 14))
    book_listbox.pack(pady=10)

    # Populate the listbox with books from the book_list
    for book in book_list:
        book_listbox.insert(tk.END, book)

    # Function to add a new book to the list
    def add_book():
        book_name = book_entry.get()  # Get the book name from the entry field
        if book_name:
            book_list.append(book_name)  # Add the book to the book_list
            book_listbox.insert(tk.END, book_name)  # Add the book to the listbox
            book_entry.delete(0, tk.END)  # Clear the entry field after adding
        else:
            messagebox.showwarning("Input Error", "Please enter a book name.")
# Show warning if no book name is entered

    # Function to remove the selected book from the list
    def remove_book():
        selected_book = book_listbox.curselection()  # Get the index of the
selected book
        if selected_book:
            book_listbox.delete(selected_book)  # Remove the book from the listbox
            book_list.pop(selected_book[0])  # Remove the book from the book_list
        else:
            messagebox.showwarning("Selection Error", "Please select a book to
remove.")  # Show warning if no book is selected

    # Entry field for adding new books
    book_entry = tk.Entry(dashboard, font=("Times New Roman", 16))
    book_entry.pack(pady=10)

  # Button to add a book
    add_button = tk.Button(dashboard, text="Add Book", font=("Times New
Roman", 16), command=add_book)
    add_button.pack(pady=5)
    # Button to remove a selected book
    remove_button = tk.Button(dashboard, text="Remove Book", font=("Times New Roman", 16), command=remove_book)
    remove_button.pack(pady=5)
 # Exit button to close the application
    exit_button = tk.Button(dashboard, text="Exit", font=("Times New Roman", 16), command=dashboard.quit)
    exit_button.pack(pady=20)

   dashboard.mainloop()  # Start the dashboard main loop
# Main login window
login_window = tk.Tk()
login_window.title("Library Management System - Login")
login_window.geometry("400x300")
login_window.config(bg="blue")
# Login form title label
login_label = tk.Label(login_window, text="Login", font=("Times New Roman", 20), bg="blue", fg="white")
login_label.pack(pady=20)
# Username label and entry field
username_label = tk.Label(login_window, text="Username:", font=("Times New Roman", 16), bg="blue", fg="white")
username_label.pack(pady=5)
username_entry = tk.Entry(login_window, font=("Times New Roman", 16))
username_entry.pack(pady=5)
# Password label and entry field
password_label = tk.Label(login_window, text="Password:", font=("Times New Roman", 16), bg="blue", fg="white")
password_label.pack(pady=5)
```

```
password_entry = tk.Entry(login_window, font=("Times New Roman", 16), show="*")  # Password field with hidden characters

password_entry.pack(pady=5)

login_button = tk.Button(login_window, text="Login", font=("Times New Roman", 16), command=login)

login_button.pack(pady=20)

login_window.mainloop()  # Start the login window main loop
```

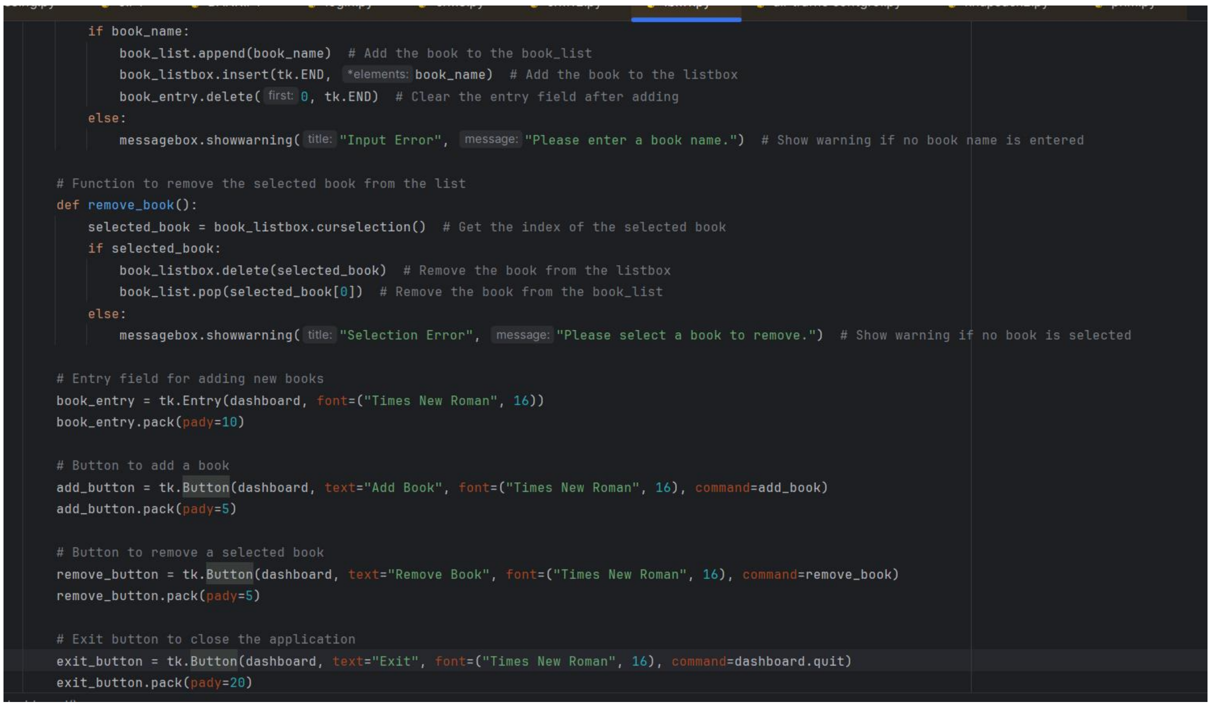
Screenshots



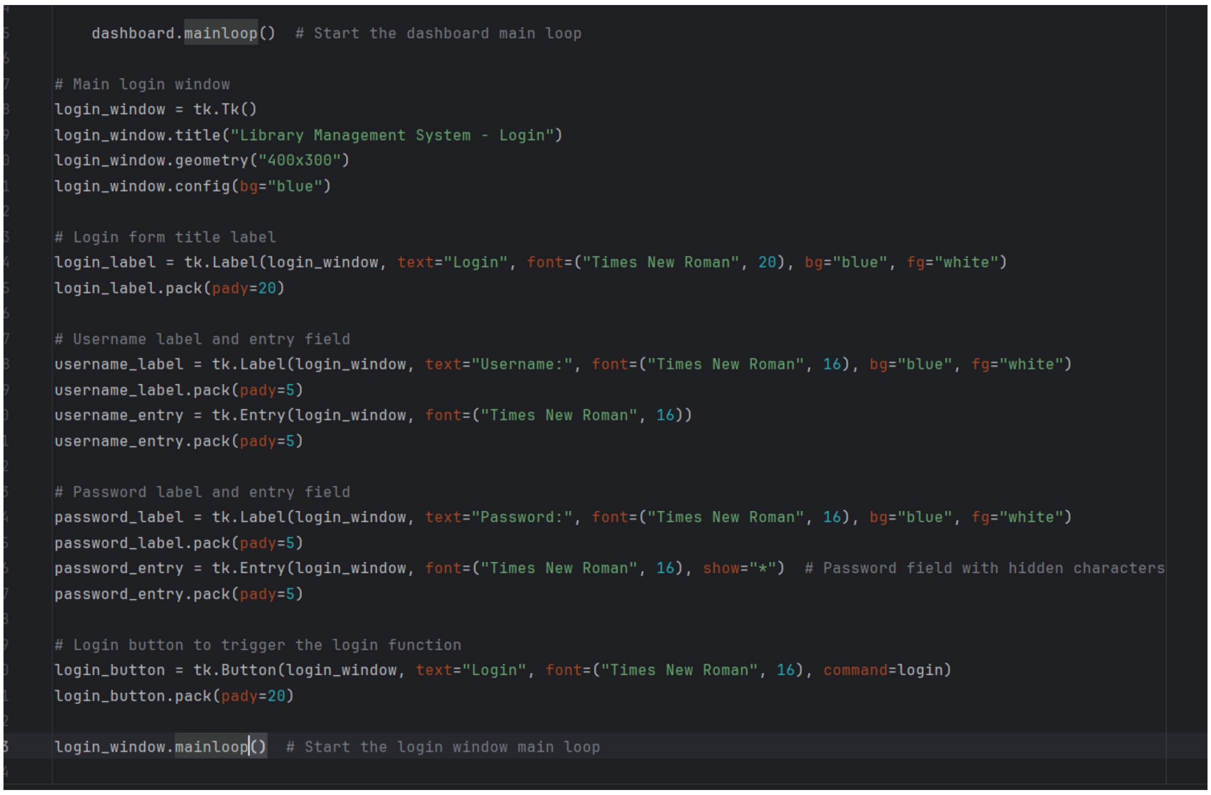
```
import tkinter as tk
from tkinter import messagebox

# Sample login credentials
credentials = {'admin': 'tanisha'}

# Book list to store the added books
book_list = []

# Login function to authenticate users
def login():
    username = username_entry.get()
    password = password_entry.get()

    # Check if the entered username and password match the credentials
    if username in credentials and credentials[username] == password:
        messagebox.showinfo( title: "Login Success",  message: "Welcome to the Library Management System!")
        login_window.destroy()  # Close the login window after successful login
        library_dashboard()  # Open the library dashboard window
    else:
        messagebox.showerror( title: "Login Failed",  message: "Invalid username or password")

# Function to open the library management dashboard
def library_dashboard():
    dashboard = tk.Tk()
    dashboard.title("Library Management System")
    dashboard.geometry("600x400")
    dashboard.config(bg="lavender")

    # Title label for the dashboard
    title_label = tk.Label(dashboard, text="Online Library Management System", font=("Times New Roman", 24), bg="lavender", fg="white")
```



```python
    if book_name:
        book_list.append(book_name)  # Add the book to the book_list
        book_listbox.insert(tk.END, *elements: book_name)  # Add the book to the listbox
        book_entry.delete( first: 0, tk.END)  # Clear the entry field after adding
    else:
        messagebox.showwarning( title: "Input Error",  message: "Please enter a book name.")  # Show warning if no book name is entered

# Function to remove the selected book from the list
def remove_book():
    selected_book = book_listbox.curselection()  # Get the index of the selected book
    if selected_book:
        book_listbox.delete(selected_book)  # Remove the book from the listbox
        book_list.pop(selected_book[0])  # Remove the book from the book_list
    else:
        messagebox.showwarning( title: "Selection Error",  message: "Please select a book to remove.")  # Show warning if no book is selected

# Entry field for adding new books
book_entry = tk.Entry(dashboard, font=("Times New Roman", 16))
book_entry.pack(pady=10)

# Button to add a book
add_button = tk.Button(dashboard, text="Add Book", font=("Times New Roman", 16), command=add_book)
add_button.pack(pady=5)

# Button to remove a selected book
remove_button = tk.Button(dashboard, text="Remove Book", font=("Times New Roman", 16), command=remove_book)
remove_button.pack(pady=5)

# Exit button to close the application
exit_button = tk.Button(dashboard, text="Exit", font=("Times New Roman", 16), command=dashboard.quit)
exit_button.pack(pady=20)
```

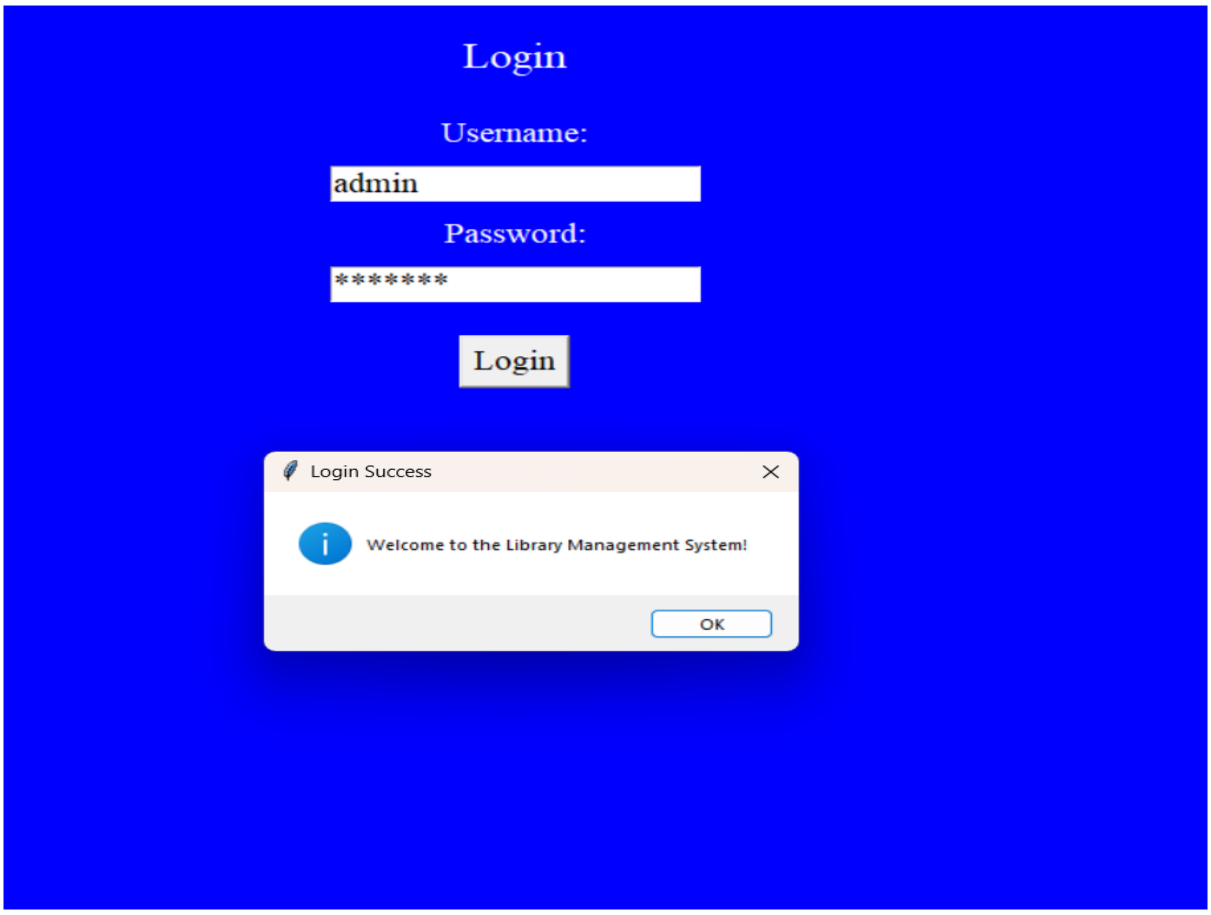


```python
    if book_name:
        book_list.append(book_name)  # Add the book to the book_list
        book_listbox.insert(tk.END, *elements: book_name)  # Add the book to the listbox
        book_entry.delete( first: 0, tk.END)  # Clear the entry field after adding
    else:
        messagebox.showwarning( title: "Input Error",  message: "Please enter a book name.")  # Show warning if no book name is entered

# Function to remove the selected book from the list
def remove_book():
    selected_book = book_listbox.curselection()  # Get the index of the selected book
    if selected_book:
        book_listbox.delete(selected_book)  # Remove the book from the listbox
        book_list.pop(selected_book[0])  # Remove the book from the book_list
    else:
        messagebox.showwarning( title: "Selection Error",  message: "Please select a book to remove.")  # Show warning if no book is selected

# Entry field for adding new books
book_entry = tk.Entry(dashboard, font=("Times New Roman", 16))
book_entry.pack(pady=10)

# Button to add a book
add_button = tk.Button(dashboard, text="Add Book", font=("Times New Roman", 16), command=add_book)
add_button.pack(pady=5)

# Button to remove a selected book
remove_button = tk.Button(dashboard, text="Remove Book", font=("Times New Roman", 16), command=remove_book)
remove_button.pack(pady=5)

# Exit button to close the application
exit_button = tk.Button(dashboard, text="Exit", font=("Times New Roman", 16), command=dashboard.quit)
exit_button.pack(pady=20)
```



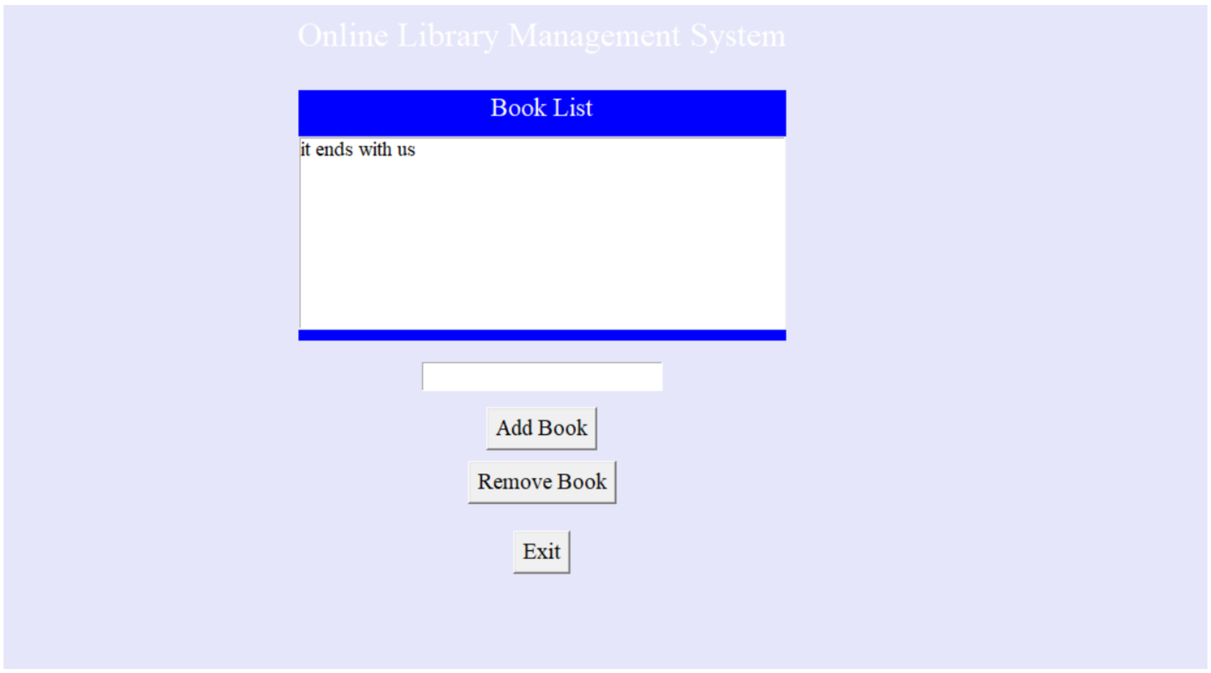
```python
    dashboard.mainloop()  # Start the dashboard main loop

# Main login window
login_window = tk.Tk()
login_window.title("Library Management System - Login")
login_window.geometry("400x300")
login_window.config(bg="blue")

# Login form title label
login_label = tk.Label(login_window, text="Login", font=("Times New Roman", 20), bg="blue", fg="white")
login_label.pack(pady=20)

# Username label and entry field
username_label = tk.Label(login_window, text="Username:", font=("Times New Roman", 16), bg="blue", fg="white")
username_label.pack(pady=5)
username_entry = tk.Entry(login_window, font=("Times New Roman", 16))
username_entry.pack(pady=5)

# Password label and entry field
password_label = tk.Label(login_window, text="Password:", font=("Times New Roman", 16), bg="blue", fg="white")
password_label.pack(pady=5)
password_entry = tk.Entry(login_window, font=("Times New Roman", 16), show="*")  # Password field with hidden characters
password_entry.pack(pady=5)

# Login button to trigger the login function
login_button = tk.Button(login_window, text="Login", font=("Times New Roman", 16), command=login)
login_button.pack(pady=20)

login_window.mainloop()  # Start the login window main loop
```

OUTPUT



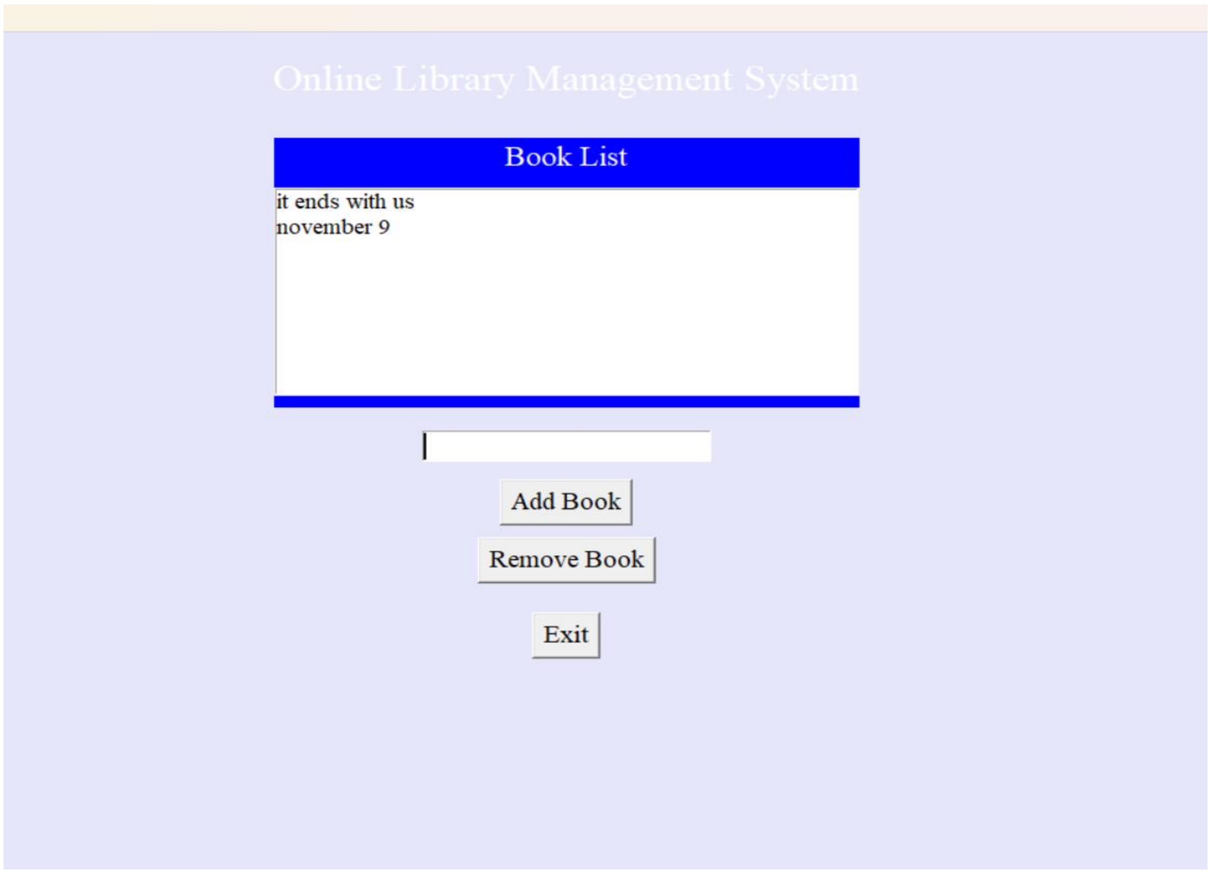
```
Login

Username:
admin
Password:
*******
Login

Login Success
Welcome to the Library Management System!
OK
```

ADD BOOK

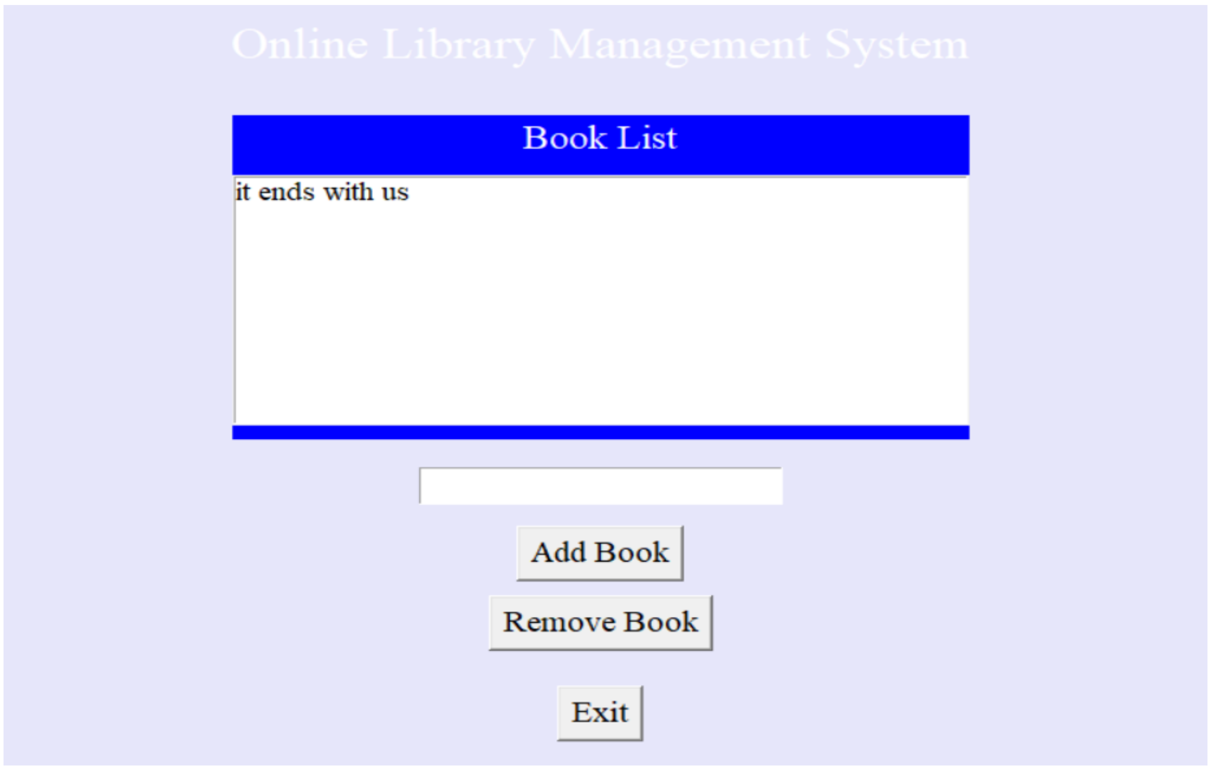


it ends with us

REMOVE BOOK



it ends with us
november 9



it ends with us

Learning Outcomes

1. Proficiency in Python Programming
2. Familiarity with tkinter
3. User authentication techniques
4. Book management logic
5. Software design principles