PYTHON PROJECT

ON

“LIBRARY MANAGEMENT SYSTEM”

NAME: KANISHK ADHITHYA

SAP ID: 500120967

ENROLLMENT NUMBER: R2142230973

BATCH: 29

Library Management System Documentation

1. Introduction:

The Library Management System is a software application designed to manage the operations of a library effectively. It provides functionalities for both users and administrators to handle tasks such as borrowing books, returning books, adding and deleting members, adding and deleting books, and modifying member and book details.

2. Features:

User Operations:

Borrowing Books: Users can borrow books by providing book details like name, code, and number of copies required.

```
def borrowbook():
    global df5
    L=[]
    booknam=input("Enter your book name:")
    bookcod=int(input("Enter the book code:"))
    copi=int(input("Enter the number of Copies you want:"))
    g = df3.loc[df3['Book Code'] == bookcod].index
    if(bookcod in df3['Book Code'][g].values and df3['Available'][g].values[0] >=
copi):
        while(copi):
            j=random.randint(100,999)
            bookid=f"DDN{bookcod}.{j}"
            df3.loc[g,'Available'] = df3['Available'][g]-1
            f = df5.loc[df5['Book ID'] == bookid].index
            status=df5.loc[f,'Status'] = "Borrowed"
            d={'Student Name':name,'Sap ID':sapid,'Book Name':df3['Book
Name'][g].values[0],'Author Name':df3['Author Name'][g].values[0],
                'Genre':df3['Genre'][g].values[0],'Book Code':bookcod,'Book
ID':bookid,'Status':status}
            L.append(d)
            copi -= 1
        print(df3)
        df5 = pd.DataFrame(L, columns=['Student Name','Sap ID','Book
Name','Author Name','Genre','Book Code','Book ID','Status'])
        df5.to_csv('Borrow_Details.csv', index=False)
        df3.to_csv('book_details.csv',index=False)
    else:
        print("Entered book code or Copies doesn't exist")
```

Returning Books: Users can return books by providing book details like name, code, and number of copies to be returned.

```
def returnbook():
    global df5
    booknam=input("Enter your book name:")
    bookcod=int(input("Enter the Book Code:"))
    copi=int(input("Enter the number of Copies you need to return:"))
    count=len(df5[(df5['Book Code']==bookcod) & (df5['Student Name']==name)])
    if(copi<=count):
        for i in range(copi):
            f = df3.loc[df3['Book Code'] == bookcod].index
            bookid=input("Enter your book ID:")
            a = df5.loc[df5['Book ID'] == bookid].index
            df5.loc[a,'Status'] = "Returned"
            df3.loc[f,'Available'] = df3['Available'][f]+1
            print(df3)
```

Displaying Available Books: Users can view the available books in the library.

```
def displaybook():
    print(df3)
```

Administrator Operations:

Adding Members: Administrators can add members to the library system by providing details such as name, SAP ID, and password.

```
def addmember():
    global df1
    #Getting inputs from the user
    print("ENTER YOUR DETAILS TO GET ADDED IN OUR LIBRARY")
    name=input("Enter your name:")
    name=name.lower()
    sapid=int(input("Enter your Sap ID:"))
    if len(str(sapid)) != 9:
        print("Enter your correct sapid")
        exit()
    password=input("Enter your Password:")

    if(sapid not in df1['Sap ID'].values):
        #Storing the user inputed values in a Dictionary
        dict2={'Student Name':name,'Sap ID':sapid,'Password':password}

        #Creating a DataFrame for the given dictionary
```

```
        df=pd.DataFrame(dict2, index=[0])

        #Adding the user inputed values to the CSV file by using concat method
        df1=pd.concat([df1,df],ignore_index=True)

        #Writing the added values to the CSV file
        df1.to_csv("student_details.csv", index=False)
    else:
        print("Name already exist")
```

Displaying Members: Administrators can view the details of all library members.

```
def displaymember():
    global df1
    #Displaying the Student details
    print(df1[['Student Name','Sap ID']])
```

Deleting Members: Administrators can delete members from the library system by providing their name and SAP ID.

```
def deletemember():
    global df1
    print("ENTER THE NAME & SAP WHICH YOU WANT TO DELETE")
    name=input("Enter the name:")
    name = name.lower()
    sapid=int(input("Enter the sapid:"))

    #Input name and sapid from the user and check whether the Name exist in the
data
    if(name in df1['Student Name'].values and sapid in df1['Sap ID'].values):
        #Getting the index of the name entered by the user
        x = df1.loc[df1['Student Name'] == name].index

        #Deleting the record from the data
        df1 = df1.drop(x, axis=0)
        df1.to_csv("student_details.csv", index=False)
        print(df1)
    else:
        print("Your Student Name or Sap ID is wrong")
```

Modifying Member Details: Administrators can modify member details such as name and password.

```
def modifymember():
    global df1
    print("ENTER THE DETAILS TO MODIFY THE DETAILS OF STUDENT IN OUR LIBRARY")
    name=input("Enter the Name to be modified:")
    name=name.lower()
    pas=input("Enter the Password to be modified:")
    if(name in df1['Student Name'].values):
        x = df1.loc[df1['Student Name'] == name].index
        df1.loc[x, "Student Name"] = name
        df1.loc[x, "Password"] = pas
    df1.to_csv("student_details.csv", index=False)
```

Adding Books: Administrators can add books to the library system by providing details such as code, name, author, genre, and availability.

```
def addbook():
    global df2
    print("ENTER THE DETAILS TO ADD THE BOOK IN OUR LIBRARY")
    bookcod=int(input("Enter the Book Code:"))
    booknam=input("Enter the Book Name:")
    authnam=input("Enter the Author Name:")
    genre=input("Enter the Genre:")
    avail=int(input("Enter the Total number of books:"))
    if(bookcod not in df2['Book Code'].values):
        dict1={'Book Code':bookcod,'Book Name':booknam,'Author
Name':authnam,'Genre':genre,'Available':avail}
        df=pd.DataFrame(dict1, index=[0])
        df2=pd.concat([df2,df],ignore_index=True)
        df2.to_csv('book_details.csv',index=False)
    else:
        print("Book already exist")
```

Modifying Book Details: Administrators can modify book details such as name, author, genre, and availability.

```
def modifybook():
    global df2
    print("ENTER THE DETAILS TO MODIFY THE BOOK IN OUR LIBRARY")

    ch=int(input("Enter which details to be modified\n1.Book Name\n2.Author
Name\n3.Available\n4.Genre\n"))
```

```
    bookcod=int(input("Enter the Book Code to be modified:"))
    x = df2.loc[df2['Book Code'] == bookcod].index
    if(bookcod in df2['Book Code'].values):
        if ch==1:
            booknam=input("Enter the Book Name to be modified:")
            df2.loc[x, "Book Name"] = booknam
        elif ch==2:
            authname=input("Enter the Author Name to be modified:")
            df2.loc[x, "Author Name"] = authname
        elif ch==3:
            avail=int(input("Enter the Available books to be modified:"))
            df2.loc[x, "Available"] = avail
        elif ch==4:
            genre=input("Enter the genre to be modified:")
            df2.loc[x, "Genre"] = genre
    df2.to_csv('book_details.csv',index=False)
```

Deleting Books: Administrators can delete books from the library system by providing their code.

```
def deletebook():
    global df2
    print("ENTER THE BOOK CODE AND BOOK NAME WHICH YOU WANT TO DELETE")
    bookcod=int(input("Enter the Book Code:"))
    if(bookcod in df2['Book Code'].values):
        x = df2.loc[df2['Book Code'] == bookcod].index
        df2 = df2.drop(x, axis=0)
        df2.to_csv('book_details.csv',index=False)
    else:
        print("Entered Book Code doesn't exist")
```

3. Implementation:

Programming Language: Python

Libraries Used: Pandas, Random

Data Storage: CSV files (student_details.csv, book_details.csv, Borrow_Details.csv)

```
import pandas as pd
import random

df1=pd.read_csv('student_details.csv')
df5=pd.read_csv("Borrow_Details.csv")
df3=pd.read_csv("book_details.csv")
```

4. Functionality Overview:

The program uses CSV files to store data related to students, books, and borrowing details.

Users are required to log in using their SAP ID and password.

Users can perform operations like borrowing books, returning books, and viewing available books.

Administrators have additional functionalities like adding members, deleting members, modifying member details, adding books, deleting books, and modifying book details.

All operations are implemented using functions to ensure modularity and maintainability.

book_details.csv

```
Book Code,Book Name,Author Name,Genre,Available
100,The Alchemist,Paulo Coelho,Fiction,5.0
101,The Great Gatsby,F.Scott Fitzgerald,Fiction,10.0
102,Nineteen Eighty-Four,George Orwell,Fiction,10.0
103,To Kill a Mockingbird,Harper Lee,Novel,10.0
104,Three Men in a Boat,Jerome K. Jerome,Novel,10.0
105,Harry Potter and the Philosophers Stone,J.K.Rowling,Novel,10.0
106,The Perfect Murder,Ruskin Bond,Mystery,10.0
107,The Complete Sherlock Holmes,Conan Doyle,Mystery,10.0
108,And Then There Were None,Agatha Christie,Mystery,10.0
109,The Silent Patient,Chetan Bhagat,Thriller,10.0
110,One Arranged Murder,Alex Michaelides,Thriller,10.0
111,Lock Every Door,Todd Ritter,Thriller,10.0
```

Borrow_Details.csv

```
Student Name,Sap ID,Book Name,Author Name,Genre,Book Code,Book ID,Status
Kanishk,500120967,The Alchemist,Paulo Coelho,Fiction,100,DDN100.859,Borrowed
```

student_details.csv

```
Student Name,Sap ID,Password
Kanishk,500120967,Kanishk123
```

5. Future Improvements:

Implementation of graphical user interface (GUI) for better user interaction.

Integration with a database management system for efficient data handling.

Implementation of advanced search and filtering functionalities for books and members.

Addition of email notifications for overdue books and membership renewals.

Implementation of user roles and access control for enhanced security.

6. Conclusion:

The Library Management System provides an efficient way to manage library operations and improve user experience. With functionalities for both users and administrators, it offers a comprehensive solution for managing library resources effectively. Further enhancements and improvements can be made to extend its capabilities and meet the evolving needs of library management.