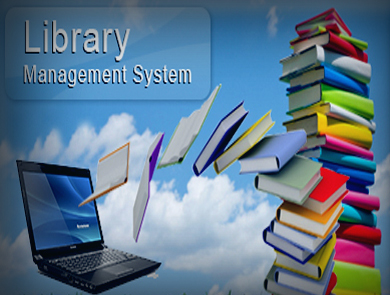
**Library Management ------------------------------**

**Language used:** C++

**Code Reference:**

<https://github.com/suyog-k/Library-Management.git>

****

**Introduction to GitHub**

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.

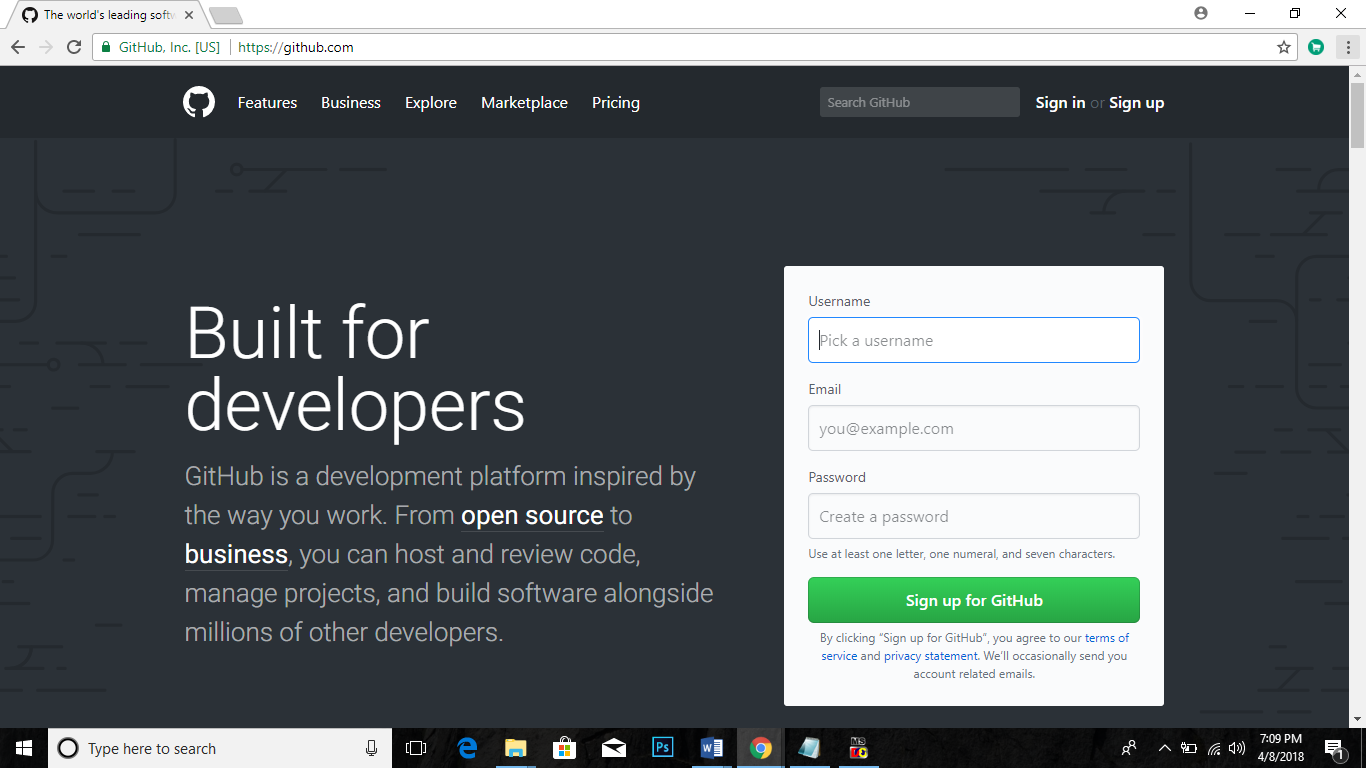
This tutorial teaches you GitHub essentials like repositories*,*branches*,*commits*,* and Pull Requests. You’ll create your own Hello World repository and learn GitHub’s Pull Request workflow, a popular way to create and review code.

**Introduction to Repository**

A **repository** is usually used to organize a single project. Repositories can contain folders and files, images, videos, spreadsheets, and data sets – anything your project needs. We recommend including a README, or a file with information about your project. GitHub makes it easy to add one at the same time you create your new repository. It also offers other common options such as a license file.

**Steps for Creating account:**

# **Step 1:** Visit the GitHub web site [www.github.com](http://www.github.com)



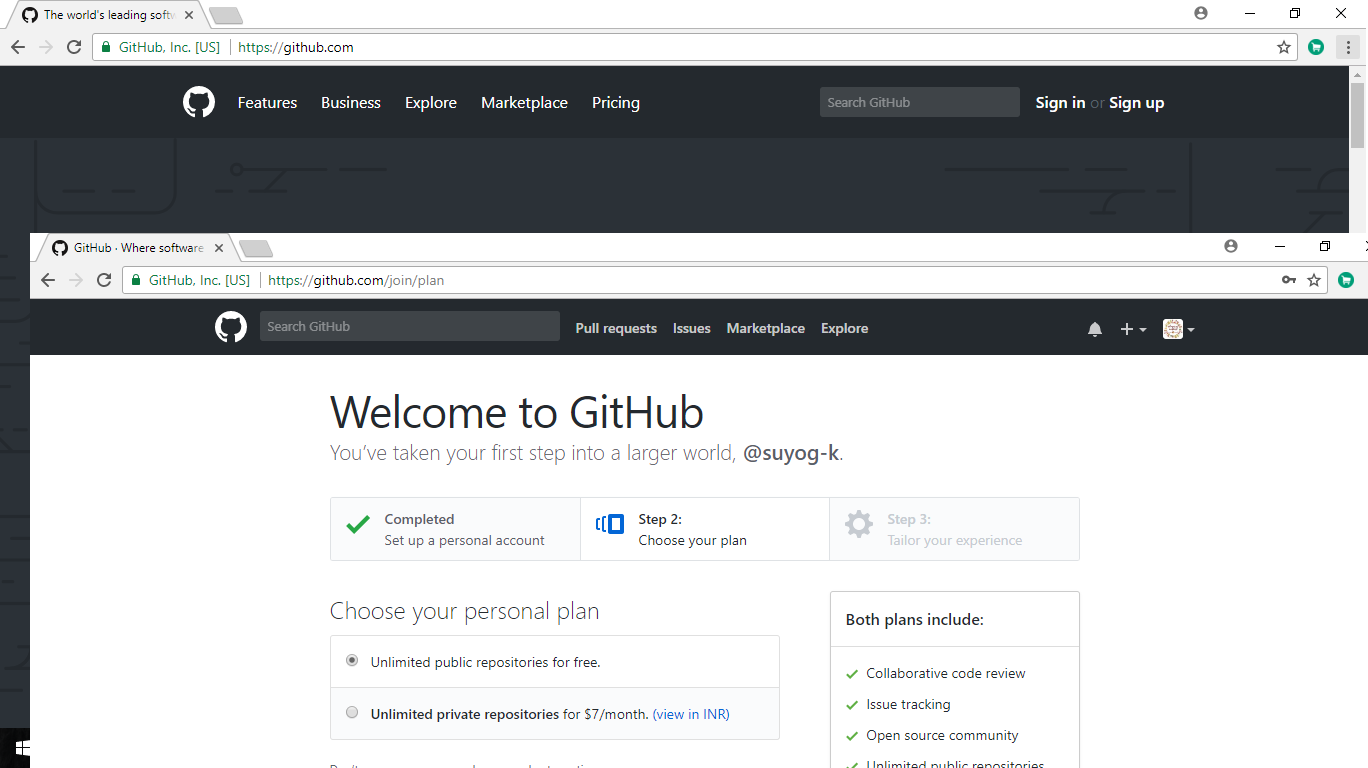
# **Step 2:** Create an account

Click the green **Sign Up for GitHub** button as seen in the image.

Enter a username, your email address and a password.

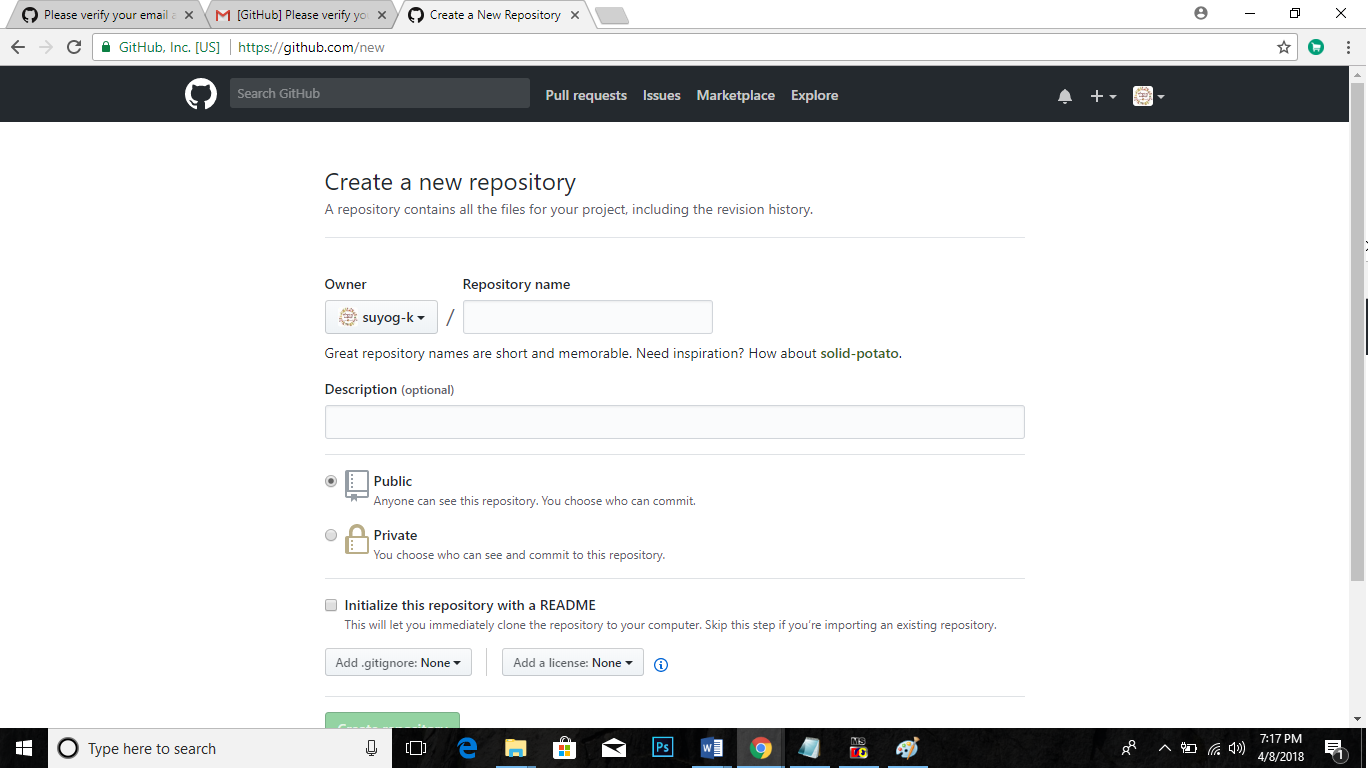
# **Step 3:** Select a plan

GitHub provides several levels of account plans but you can create unlimited public repositories with a free plan so for now, you can select that one. Hit **Finish sign up**.



**Creating a Repository:**

1. In the upper right corner, next to your avatar and then select New repository.
2. Name your repository according to your project.
3. Write a short description.
4. Select initialize this repository with a README.
5. Press create repository, and your repository will be created.

****

**Library management code**

#include<fstream.h>

#include<conio.h>

#include<stdio.h>

#include<process.h>

#include<string.h>

#include<iomanip.h>

class book

{

char bno[6];

char bname[50];

char aname[20];

public:

void create\_book()

{

cout<<"\nNEW BOOK ENTRY...\n";

cout<<"\nEnter The book no.";

cin>>bno;

cout<<"\n\nEnter The Name of The Book ";

gets(bname);

cout<<"\n\nEnter The Author's Name ";

gets(aname);

cout<<"\n\n\nBook Created..";

}

void show\_book()

{

cout<<"\nBook no. : "<<bno;

cout<<"\nBook Name : ";

puts(bname);

cout<<"Author Name : ";

puts(aname);

}

void modify\_book()

{

cout<<"\nBook no. : "<<bno;

cout<<"\nModify Book Name : ";

gets(bname);

cout<<"\nModify Author's Name of Book : ";

gets(aname);

}

char\* retbno()

{

return bno;

}

void report()

{cout<<bno<<setw(30)<<bname<<setw(30)<<aname<<endl;}

};

class student

{

char admno[6];

char name[20];

char stbno[6];

int token;

public:

void create\_student()

{

clrscr();

cout<<"\nNEW STUDENT ENTRY...\n";

cout<<"\nEnter The admission no. ";

cin>>admno;

cout<<"\n\nEnter The Name of The Student ";

gets(name);

token=0;

stbno[0]='/0';

cout<<"\n\nStudent Record Created..";

}

void show\_student()

{

cout<<"\nAdmission no. : "<<admno;

cout<<"\nStudent Name : ";

puts(name);

cout<<"\nNo of Book issued : "<<token;

if(token==1)

cout<<"\nBook No "<<stbno;

}

void modify\_student()

{

cout<<"\nAdmission no. : "<<admno;

cout<<"\nModify Student Name : ";

gets(name);

}

char\* retadmno()

{

return admno;

}

char\* retstbno()

{

return stbno;

}

int rettoken()

{

return token;

}

void addtoken()

{token=1;}

void resettoken()

{token=0;}

void getstbno(char t[])

{

strcpy(stbno,t);

}

void report()

{cout<<"\t"<<admno<<setw(20)<<name<<setw(10)<<token<<endl;}

};

fstream fp,fp1;

book bk;

student st;

void write\_book()

{

char ch;

fp.open("book.dat",ios::out|ios::app);

do

{

clrscr();

bk.create\_book();

fp.write((char\*)&bk,sizeof(book));

cout<<"\n\nDo you want to add more record..(y/n?)";

cin>>ch;

}while(ch=='y'||ch=='Y');

fp.close();

}

void write\_student()

{

char ch;

fp.open("student.dat",ios::out|ios::app);

do

{

st.create\_student();

fp.write((char\*)&st,sizeof(student));

cout<<"\n\ndo you want to add more record..(y/n?)";

cin>>ch;

}while(ch=='y'||ch=='Y');

fp.close();

}

void display\_spb(char n[])

{

cout<<"\nBOOK DETAILS\n";

int flag=0;

fp.open("book.dat",ios::in);

while(fp.read((char\*)&bk,sizeof(book)))

{

if(strcmpi(bk.retbno(),n)==0)

{

bk.show\_book();

flag=1;

}

}

fp.close();

if(flag==0)

cout<<"\n\nBook does not exist";

getch();

}

void display\_sps(char n[])

{

cout<<"\nSTUDENT DETAILS\n";

int flag=0;

fp.open("student.dat",ios::in);

while(fp.read((char\*)&st,sizeof(student)))

{

if((strcmpi(st.retadmno(),n)==0))

{

st.show\_student();

flag=1;

}

}

fp.close();

if(flag==0)

cout<<"\n\nStudent does not exist";

getch();

}

void modify\_book()

{

char n[6];

int found=0;

clrscr();

cout<<"\n\n\tMODIFY BOOK REOCORD";

cout<<"\n\n\tEnter The book no. of The book";

cin>>n;

fp.open("book.dat",ios::in|ios::out);

while(fp.read((char\*)&bk,sizeof(book)) && found==0)

{

if(strcmpi(bk.retbno(),n)==0)

{

bk.show\_book();

cout<<"\nEnter The New Details of book"<<endl;

bk.modify\_book();

int pos=-1\*sizeof(bk);

fp.seekp(pos,ios::cur);

fp.write((char\*)&bk,sizeof(book));

cout<<"\n\n\t Record Updated";

found=1;

}

}

fp.close();

if(found==0)

cout<<"\n\n Record Not Found ";

getch();

}

void modify\_student()

{

char n[6];

int found=0;

clrscr();

cout<<"\n\n\tMODIFY STUDENT RECORD";

cout<<"\n\n\tEnter The admission no. of The student";

cin>>n;

fp.open("student.dat",ios::in|ios::out);

while(fp.read((char\*)&st,sizeof(student)) && found==0)

{

if(strcmpi(st.retadmno(),n)==0)

{

st.show\_student();

cout<<"\nEnter The New Details of student"<<endl;

st.modify\_student();

int pos=-1\*sizeof(st);

fp.seekp(pos,ios::cur);

fp.write((char\*)&st,sizeof(student));

cout<<"\n\n\t Record Updated";

found=1;

}

}

fp.close();

if(found==0)

cout<<"\n\n Record Not Found ";

getch();

}

void delete\_student()

{

char n[6];

int flag=0;

clrscr();

cout<<"\n\n\n\tDELETE STUDENT...";

cout<<"\n\nEnter The admission no. of the Student You Want To Delete : ";

cin>>n;

fp.open("student.dat",ios::in|ios::out);

fstream fp2;

fp2.open("Temp.dat",ios::out);

fp.seekg(0,ios::beg);

while(fp.read((char\*)&st,sizeof(student)))

{

if(strcmpi(st.retadmno(),n)!=0)

fp2.write((char\*)&st,sizeof(student));

else

flag=1;

}

fp2.close();

fp.close();

remove("student.dat");

rename("Temp.dat","student.dat");

if(flag==1)

cout<<"\n\n\tRecord Deleted ..";

else

cout<<"\n\nRecord not found";

getch();

}

void delete\_book()

{

char n[6];

clrscr();

cout<<"\n\n\n\tDELETE BOOK ";

cout<<"\n\nEnter The Book no. of the Book You Want To Delete : ";

cin>>n;

fp.open("book.dat",ios::in|ios::out);

fstream fp2;

fp2.open("Temp.dat",ios::out);

fp.seekg(0,ios::beg);

while(fp.read((char\*)&bk,sizeof(book)))

{

if(strcmpi(bk.retbno(),n)!=0)

{

fp2.write((char\*)&bk,sizeof(book));

}

}

fp2.close();

fp.close();

remove("book.dat");

rename("Temp.dat","book.dat");

cout<<"\n\n\tRecord Deleted ..";

getch();

}

void display\_alls()

{

clrscr();

fp.open("student.dat",ios::in);

if(!fp)

{

cout<<"ERROR!!! FILE COULD NOT BE OPEN ";

getch();

return;

}

cout<<"\n\n\t\tSTUDENT LIST\n\n";

cout<<"==================================================================\n";

cout<<"\tAdmission No."<<setw(10)<<"Name"<<setw(20)<<"Book Issued\n";

cout<<"==================================================================\n";

while(fp.read((char\*)&st,sizeof(student)))

{

st.report();

}

fp.close();

getch();

}

void display\_allb()

{

clrscr();

fp.open("book.dat",ios::in);

if(!fp)

{

cout<<"ERROR!!! FILE COULD NOT BE OPEN ";

getch();

return;

}

cout<<"\n\n\t\tBook LIST\n\n";

cout<<"=========================================================================\n";

cout<<"Book Number"<<setw(20)<<"Book Name"<<setw(25)<<"Author\n";

cout<<"=========================================================================\n";

while(fp.read((char\*)&bk,sizeof(book)))

{

bk.report();

}

fp.close();

getch();

}

void book\_issue()

{

char sn[6],bn[6];

int found=0,flag=0;

clrscr();

cout<<"\n\nBOOK ISSUE ...";

cout<<"\n\n\tEnter The student's admission no.";

cin>>sn;

fp.open("student.dat",ios::in|ios::out);

fp1.open("book.dat",ios::in|ios::out);

while(fp.read((char\*)&st,sizeof(student)) && found==0)

{

if(strcmpi(st.retadmno(),sn)==0)

{

found=1;

if(st.rettoken()==0)

{

cout<<"\n\n\tEnter the book no. ";

cin>>bn;

while(fp1.read((char\*)&bk,sizeof(book))&& flag==0)

{

if(strcmpi(bk.retbno(),bn)==0)

{

bk.show\_book();

flag=1;

st.addtoken();

st.getstbno(bk.retbno());

int pos=-1\*sizeof(st);

fp.seekp(pos,ios::cur);

fp.write((char\*)&st,sizeof(student));

cout<<"\n\n\t Book issued successfully\n\nPlease Note: Write the current date in backside of your book and submit within 15 days fine Rs. 1 for each day after 15 days period";

}

}

if(flag==0)

cout<<"Book no does not exist";

}

else

cout<<"You have not returned the last book ";

}

}

if(found==0)

cout<<"Student record not exist...";

getch();

fp.close();

fp1.close();

}

void book\_deposit()

{

char sn[6],bn[6];

int found=0,flag=0,day,fine;

clrscr();

cout<<"\n\nBOOK DEPOSIT ...";

cout<<"\n\n\tEnter The student’s admission no.";

cin>>sn;

fp.open("student.dat",ios::in|ios::out);

fp1.open("book.dat",ios::in|ios::out);

while(fp.read((char\*)&st,sizeof(student)) && found==0)

{

if(strcmpi(st.retadmno(),sn)==0)

{

found=1;

if(st.rettoken()==1)

{

while(fp1.read((char\*)&bk,sizeof(book))&& flag==0)

{

if(strcmpi(bk.retbno(),st.retstbno())==0)

{

bk.show\_book();

flag=1;

cout<<"\n\nBook deposited in no. of days";

cin>>day;

if(day>15)

{

fine=(day-15)\*1;

cout<<"\n\nFine has to deposited Rs. "<<fine;

}

st.resettoken();

int pos=-1\*sizeof(st);

fp.seekp(pos,ios::cur);

fp.write((char\*)&st,sizeof(student));

cout<<"\n\n\t Book deposited successfully";

}

}

if(flag==0)

cout<<"Book no does not exist";

}

else

cout<<"No book is issued..please check!!";

}

}

if(found==0)

cout<<"Student record not exist...";

getch();

fp.close();

fp1.close();

}

void intro()

{

clrscr();

gotoxy(30,11);

cout<<"LIBRARY MANAGEMENT SYSTEM";

getch();

}

void admin\_menu()

{

clrscr();

int ch2;

cout<<"\n\n\n\tADMINISTRATOR MENU";

cout<<"\n\n\t1.CREATE STUDENT RECORD";

cout<<"\n\n\t2.DISPLAY ALL STUDENTS RECORD";

cout<<"\n\n\t3.DISPLAY SPECIFIC STUDENT RECORD ";

cout<<"\n\n\t4.MODIFY STUDENT RECORD";

cout<<"\n\n\t5.DELETE STUDENT RECORD";

cout<<"\n\n\t6.CREATE BOOK ";

cout<<"\n\n\t7.DISPLAY ALL BOOKS ";

cout<<"\n\n\t8.DISPLAY SPECIFIC BOOK ";

cout<<"\n\n\t9.MODIFY BOOK ";

cout<<"\n\n\t10.DELETE BOOK ";

cout<<"\n\n\t11.BACK TO MAIN MENU";

cout<<"\n\n\tPlease Enter Your Choice (1-11) ";

cin>>ch2;

switch(ch2)

{

case 1: clrscr();

write\_student();break;

case 2: display\_alls();break;

case 3:

char num[6];

clrscr();

cout<<"\n\n\tPlease Enter The Admission No. ";

cin>>num;

display\_sps(num);

break;

case 4: modify\_student();break;

case 5: delete\_student();break;

case 6: clrscr();

write\_book();break;

case 7: display\_allb();break;

case 8: {

char num[6];

clrscr();

cout<<"\n\n\tPlease Enter The book No. ";

cin>>num;

display\_spb(num);

break;

}

case 9: modify\_book();break;

case 10: delete\_book();break;

case 11: return;

default:cout<<"\a";

}

admin\_menu();

}

void main()

{

char ch;

intro();

do

{

clrscr();

cout<<"\n\n\n\tMAIN MENU";

cout<<"\n\n\t01. BOOK ISSUE";

cout<<"\n\n\t02. BOOK DEPOSIT";

cout<<"\n\n\t03. ADMINISTRATOR MENU";

cout<<"\n\n\t04. EXIT";

cout<<"\n\n\tPlease Select Your Option (1-4) ";

ch=getche();

switch(ch)

{

case '1':clrscr();

book\_issue();

break;

case '2':book\_deposit();

break;

case '3':admin\_menu();

break;

case '4':exit(0);

default :cout<<"\a";

}

}while(ch!='4');

getch();

}

**Library management code**

This code is useful for keeping the records of books inventory and students who issue books.

This code contains file handling where records are stored and maintained. It also uses concept of functions.

* You can find the code which ‘**library\_management.cpp**’ using the link mentioned above on GitHub.
* It is a moderate code for storing data and is useful for maintaining library records.

