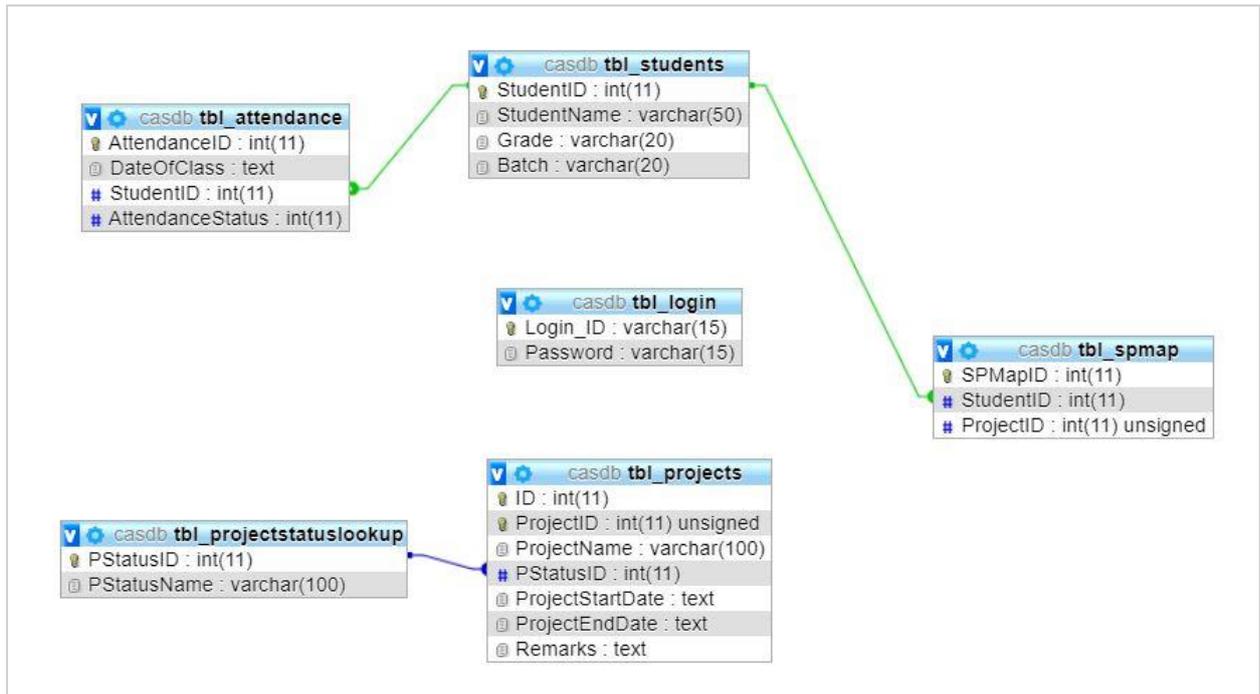


Criterion C

Features: -

- Login
- Student credential storage, display and editing
- Project display storage, display, and editing
- Attendance of students
- Project status allotment
- Dashboard

Database Design:



Software used: MySQL Server

Purpose: The relational database is used with normalization to provide referential integrity for the management of the data.

Tables:

Name: tbl_Login

The length of Username and Password should not exceed 15 characters. Two users can not have same LoginID for their account. Primary Key will be set to ensure the condition mentioned above.

This table is used to store the Login ID and Password of all registered users

The length of Username and Password should not exceed 15 characters

Two users can not have same LoginID for their account

Primary Key will be set to ensure the condition mentioned above.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Login_ID	varchar(15)	latin1_swedish_ci		No	None			Change Drop More
2	Password	varchar(15)	latin1_swedish_ci		No	None			Change Drop More

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	Login_ID	1	A	No	

Name: tbl_Projects

This table is used to store the details of projects. Each project will be identified using an unique key called "Project ID". The status of the project can set to any of three statuses such as "1 -To Confirm", "2-In Process" and "3- Completed".

This table is used to store the details of projects

Each project will be identified using an unique key called "Project ID".

The status of the project can set to any of three statuses such as "1 -To Confirm", "2-In Process" and "3- Completed"

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(11)			No	None		AUTO_INCREMENT	
2	ProjectID	int(11)		UNSIGNED	No	None			
3	ProjectName	varchar(100)	latin1_swedish_ci		No	None			
4	PStatusID	int(11)			No	None			
5	ProjectStartDate	text	latin1_swedish_ci		No	None			
6	ProjectEndDate	text	latin1_swedish_ci		No	None			
7	Remarks	text	latin1_swedish_ci		No	None			

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	ID	2	A	No	
Edit Drop	PK_ProjectID	BTREE	No	No	ProjectID	2	A	No	
Edit Drop	PK_PStatus_Project	BTREE	No	No	PStatusID	2	A	No	

Name: tbl_ProjectStatusLookUp

This table stores the project status with the PStatusID being the lookup value.

Server: 127.0.0.1 > Database: casdb > Table: tbl_projectstatuslookup

Browse Structure SQL Search Insert Export

Showing rows 0 - 2 (3 total, Query took 0.0015 seconds.)

```
SELECT * FROM `tbl_projectstatuslookup`
```

This table stores the project status with the PStatusID being the lookup value.

Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

	PStatusID	PStatusName
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	To Confirm
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	In Progress
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	Completed

Name: tbl_Students

This table is used to store the student details such as Name, ID, Batch and Grade. The StudentID field will act as a unique field to identify the students.

This table is used to store the student details such as Name, ID, Batch and Grade.

Server: 127.0.0.1 > Database: casdb > Table: tbl_students

Browse Structure SQL Search Insert Export Import Privileges

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
<input type="checkbox"/>	StudentID	int(11)			No	None		AUTO_INCREMENT
<input type="checkbox"/>	StudentName	varchar(50)	latin1_swedish_ci		No	None		
<input type="checkbox"/>	Grade	varchar(20)	latin1_swedish_ci		No	None		
<input type="checkbox"/>	Batch	varchar(20)	latin1_swedish_ci		No	None		

The StudentID field will act as a unique field to identify the students.

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
<input type="checkbox"/> Edit <input type="checkbox"/> Drop	PRIMARY	BTREE	Yes	No	StudentID	1	A	No	

Name: tbl_SPMaP

This table is used to map the Projects and Students. The projectID and studentID are the primary keys for the Student and Project table respectively.

Server: 127.0.0.1 » Database: casdb » Table: tbl_smap

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	SPMaPID			No	None		AUTO_INCREMENT	
<input type="checkbox"/>	2	StudentID			No	None			
<input type="checkbox"/>	3	ProjectID		UNSIGNED	No	None			

Check all With selected: Browse Change Drop Primary Unique Index

Remove from central columns

Print | Propose table structure | Track table | Move columns | Normalize

Add 1 column(s) after ProjectID

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
	PRIMARY	BTREE	Yes	No	SPMaPID	2	A	No	
	FK_StudentID_SPMAP	BTREE	No	No	StudentID	2	A	No	

Name: tbl_Attendance

This table is used to store the daily attendance record for the students for the CAS class.

Server: 127.0.0.1 » Database: casdb » Table: tbl_attendance

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	AttendanceID			No	None		AUTO_INCREMENT	
<input type="checkbox"/>	2	DateOfClass	latin1_swedish_ci		No	None			
<input type="checkbox"/>	3	StudentID			No	None			
<input type="checkbox"/>	4	AttendanceStatus			No	None			

Check all With selected: Browse Change Drop Primary Unique Index

Remove from central columns

Print | Propose table structure | Track table | Move columns | Normalize

Add 1 column(s) after AttendanceStatus

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
	PRIMARY	BTREE	Yes	No	AttendanceID	1	A	No	
	FK_StudentID	BTREE	No	No	StudentID	1	A	No	

Application Development:

For connectivity of Python with MySQL, the following modules to be installed using pip:

```
pip install mysql-connector-python
```

Login Window : mainScreen()

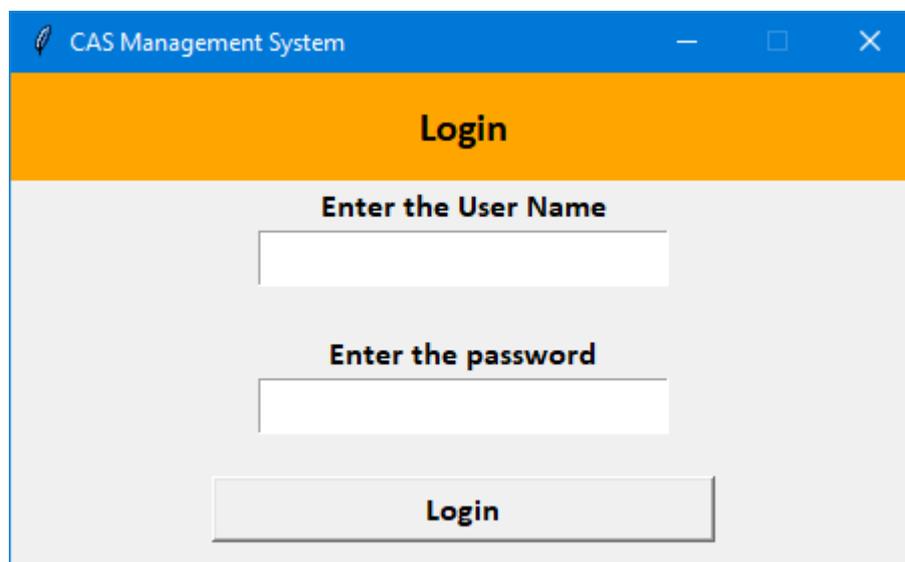
```
CASMgmt -OOP.py - E:\CASMgmt -OOP.py (3.6.7)
File Edit Format Run Options Window Help
def main_account_screen():
    global main_screen
    main_screen = Tk() # create a GUI window
    main_screen.geometry("450x250") # set the configuration of GUI window
    main_screen.title("CAS Management System") # set the title of GUI window
    main_screen.resizable(height = 0, width = 0)
    global username_verify
    global password_verify
    username_verify = StringVar()
    password_verify = StringVar()

    Label(text="Login", bg="Orange", width="450", height="2", font=("Calibri", 15, "bold")).pack()

    # Set username entry
    Label(text="Enter the User Name", width="450", font=("Calibri", 12, "bold")).pack()
    global username_entry
    username_entry = Entry(main_screen, font=("Calibri", 15, "bold"), textvariable=username_verify)
    username_entry.pack()
    Label(text="").pack()

    # Set password entry
    Label(text="Enter the password", width="450", font=("Calibri", 12, "bold")).pack()
    global password_entry
    password_entry = Entry(main_screen, font=("Calibri", 15, "bold"), textvariable=password_verify, show="*")
    password_entry.pack()
    Label(text="").pack()

    # create Login Button
    Button(text="Login", height="1", width="30", font=("Calibri", 12, "bold"), command=login_verify).pack()
```



Dashboard Window : Dashboard()

```
CASMgmt - OOP.py - E:\CASMgmt - OOP.py (3.6.7)
File Edit Format Run Options Window Help

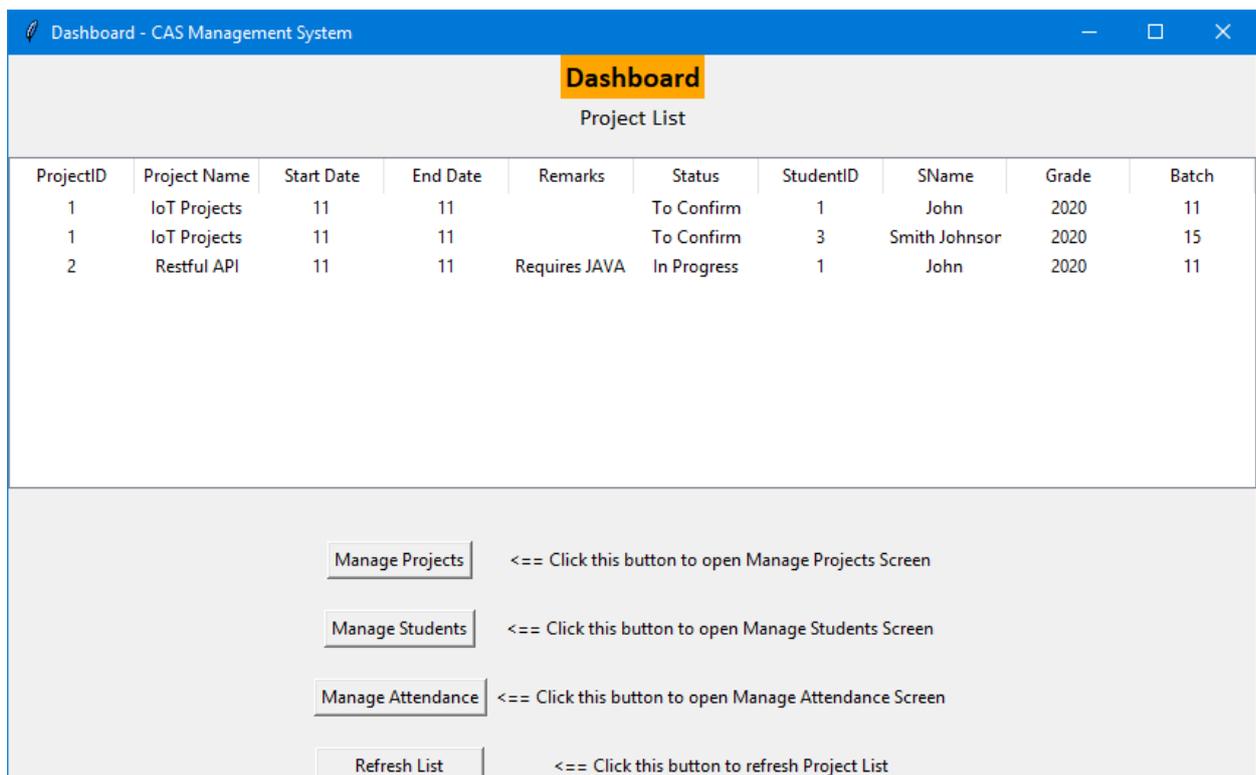
class Dashboard:
    def __init__(self, master):
        self.master = master
        self.master.geometry("850x500")
        self.master.title("Dashboard - CAS Management System") # set the title of GUI window
        self.master.resizable(height = 0, width = 1000)
        self.project_status_dict = {}
        self.fetch_project_status()
        self.show_dashboard()

    def show_dashboard(self):
        self.frame1 = tk.Frame(self.master)
        # create a Form label
        Label(self.frame1, text="Dashboard", bg="Orange", font=("Calibri", 15, "bold")).grid(row=0, column=0)
        Label(self.frame1, text="Project List", font=("Calibri", 12,)).grid(row=1, column=0, columnspan=10)
        self.frame1.pack()

        self.frame2 = tk.Frame(self.master)
        cols = ('ProjectID', 'Project Name', 'Start Date', 'End Date', 'Remarks', 'Status', 'StudentID', 'SName', 'Grade', 'Batch')
        self.listBox = ttk.Treeview(self.frame2, columns=cols, show='headings', selectmode='browse')

        for col in cols:
            self.listBox.column(col, width = 10, anchor='c')
            self.listBox.heading(col, text=col)
        self.listBox.pack(fill='x', expand=True)

        tempList = self.fetch_project_details()
        for i in tempList:
            self.listBox.insert("", "end", values=i)
```



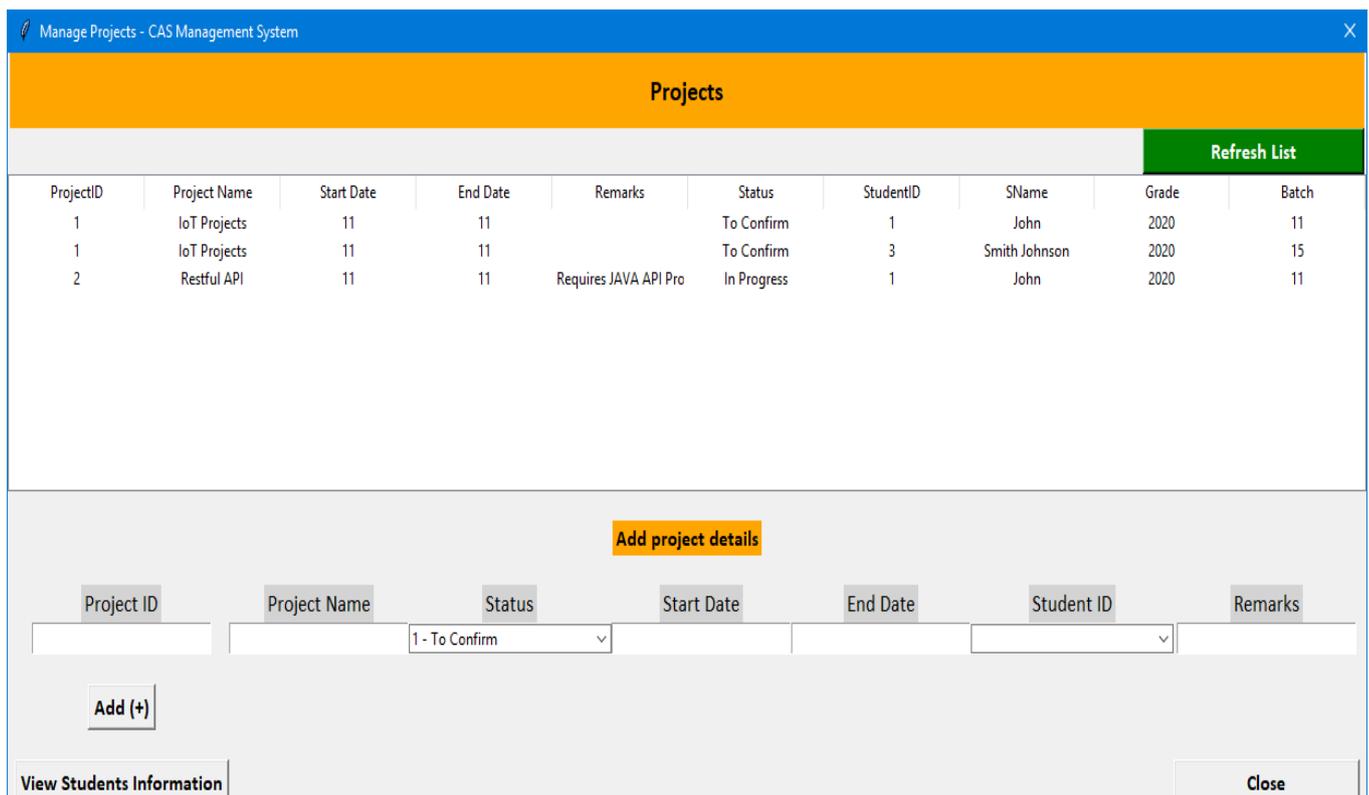
Project Window : Class Projects ()

```
CASMgrt - OOP.py - E:\CASMgrt - OOP.py (3.6.7)
File Edit Format Run Options Window Help
class Projects(Dashboard):
    def __init__(self, master):
        self.master = master
        self.master.title("Manage Projects - CAS Management System") # set the title of GUI window
        self.master.geometry("1240x540")
        self.master.resizable(height = 0, width = 0)
        self.project_status_dict = {}
        self.projectid = StringVar()
        self.projectname = StringVar()
        self.status = StringVar()
        self.startdate = StringVar()
        self.enddate = StringVar()
        self.studentid= StringVar()
        self.remarks = StringVar()
        self.fetch_project_status()
        self.show_projects_window()

    def show_projects_window(self):
        self.frame1 = tk.Frame(self.master)
        # create a Form label
        Label(self.frame1, text="Projects", width=123, height=2, bg="Orange", font=("Calibri", 15, "bold")).g
        refresh_btn = Button(self.frame1, text="Refresh List", width=24, bg='green',fg='white',font=("Calibri"
        self.frame1.pack()

        self.frame2 = tk.Frame(self.master)
        cols = ('ProjectID', 'Project Name', 'Start Date','End Date','Remarks','Status','StudentID','SName','Gra
        self.listBox = ttk.Treeview(self.frame2, columns=cols, show='headings',selectmode='browse')

        for col in cols:
```



Students Window : Class Students ()

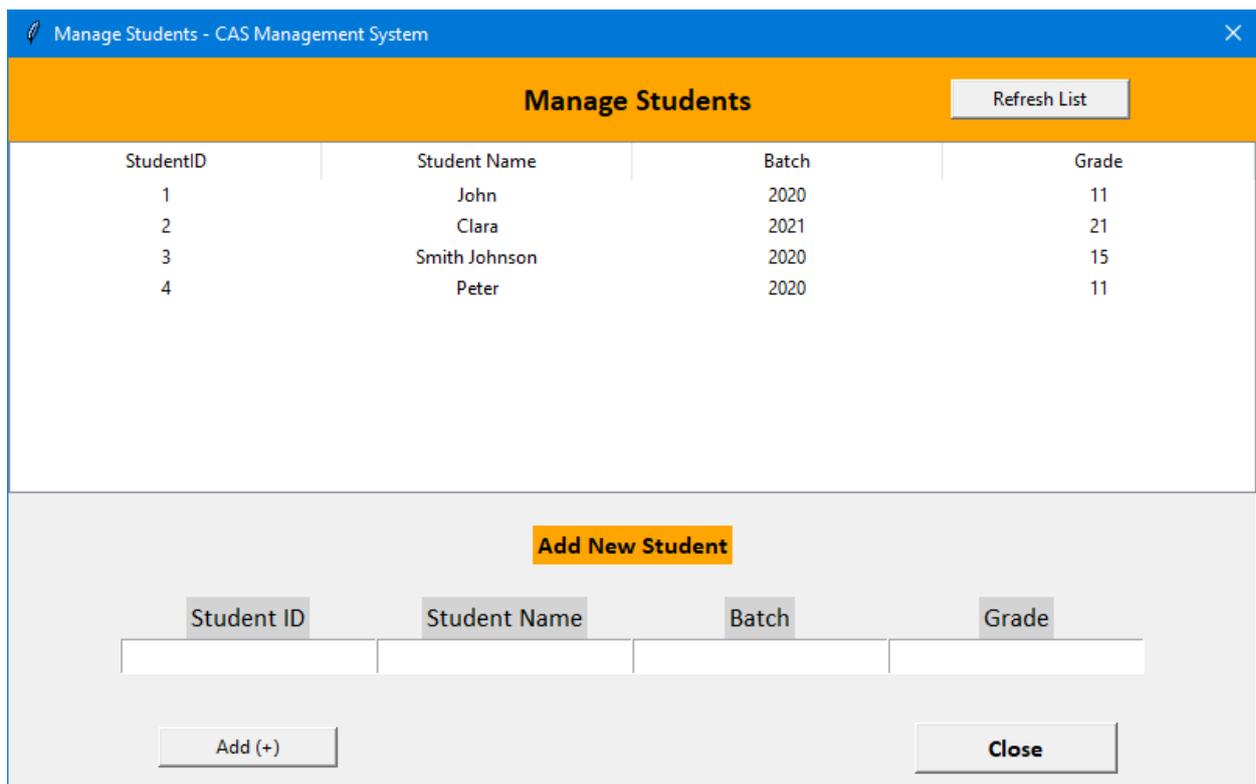
```
CASMgmt - OOP.py - E:\CASMgmt - OOP.py (3.6.7)
File Edit Format Run Options Window Help
class Students(Dashboard):
    def __init__(self, master):
        self.master = master
        self.master.title("Manage Students - CAS Management System") # set the title of GUI window
        self.master.geometry("800x475")
        self.master.resizable(height = 0, width = 0)
        self.show_students_window()

    def show_students_window(self):
        self.frame1 = tk.Frame(self.master)
        # create a Form label
        Label(self.frame1, text="Manage Students", width = "80", bg="Orange", height="2", font=("Calibri",
        refresh_btn = Button(self.frame1, text="Refresh List", width=15, command=self.refresh_students_i
        self.frame1.pack()

        self.frame2 = tk.Frame(self.master)
        cols = ('StudentID','Student Name','Batch','Grade')
        self.listBox = ttk.Treeview(self.frame2, columns=cols, show='headings',selectmode ='browse')
        for col in cols:
            self.listBox.column(col, width = 40, anchor ='c')
            self.listBox.heading(col, text=col)

        self.listBox.pack(fill='x',expand=True)
        tempList = self.fetch_student_details()
        for i in tempList:
            self.listBox.insert("", "end", values=i)

# self.listBox.grid(row=1, column=0, columnspan=4)
self.frame2.pack(fill='x',expand=True)
```



Attendance Window : Class Attendance ()

```
CASMgmt - OOP.py - E:\CASMgmt - OOP.py (3.6.7)
File Edit Format Run Options Window Help
class Attendance(Dashboard):
    def __init__(self, master):
        self.master = master
        self.master.title("Window n.2")
        self.master.geometry("335x250")
        self.master.title("Attendance Entry - CAS Management System") # set the title of GUI window
        self.master.resizable(height = 0, width = 0)
        self.studentid = StringVar()
        self.attendance = StringVar()
        self.timestampStr= StringVar()
        self.show_attendance_window()

    def show_attendance_window(self):
        self.frame = tk.Frame(self.master)
        dateTimeObj = datetime.now()
        self.timestampStr = dateTimeObj.strftime("%a %d-%b-%Y %r")

        # create a Form label
        Label(self.frame, text="Students Attendance Entry", bg="Orange", font=("Calibri", 15,"bold")).grid(r
        Label(self.frame, text=self.timestampStr, font=("Calibri", 11)).grid(row=1, column=0,columnspan=2)

        Label(self.frame, text="").grid(row=2, column=0)
        Label(self.frame, text="Student ID : ").grid(row=3, column=0)

        studentid_combo = ttk.Combobox(self.frame, width = 27, textvariable = self.studentid ,state="reado
        studentid_combo['values'] = self.fetch_student_detail()
        studentid_combo.grid(row = 3,column = 1)
        studentid_combo.current(0)

        Label(self.frame, text="").grid(row=5, column=0)
```

Attendance Entry - CAS Management System

Students Attendance Entry

Fri 04-Sep-2020 11:00:53 AM

Student ID :

Attendance :

Save Attendance

Close

Bibliography:

1. <https://pynative.com/install-mysql-connector-python/>
2. https://www.tutorialspoint.com/python/python_gui_programming.htm
3. <https://www.geeksforgeeks.org/python-tkinter-tutorial/>
4. <https://effbot.org/tkinterbook/>