NAME: ***HABEEB RAHMAN K T***

VTU NO.: ***22336***

SECTION: ***SoC – A4***

PROJET NAME: ***STUDENT MANAGEMENT SYSTEM***

COURSE TITLE: ***PROGRAMMING USING PYTHON***

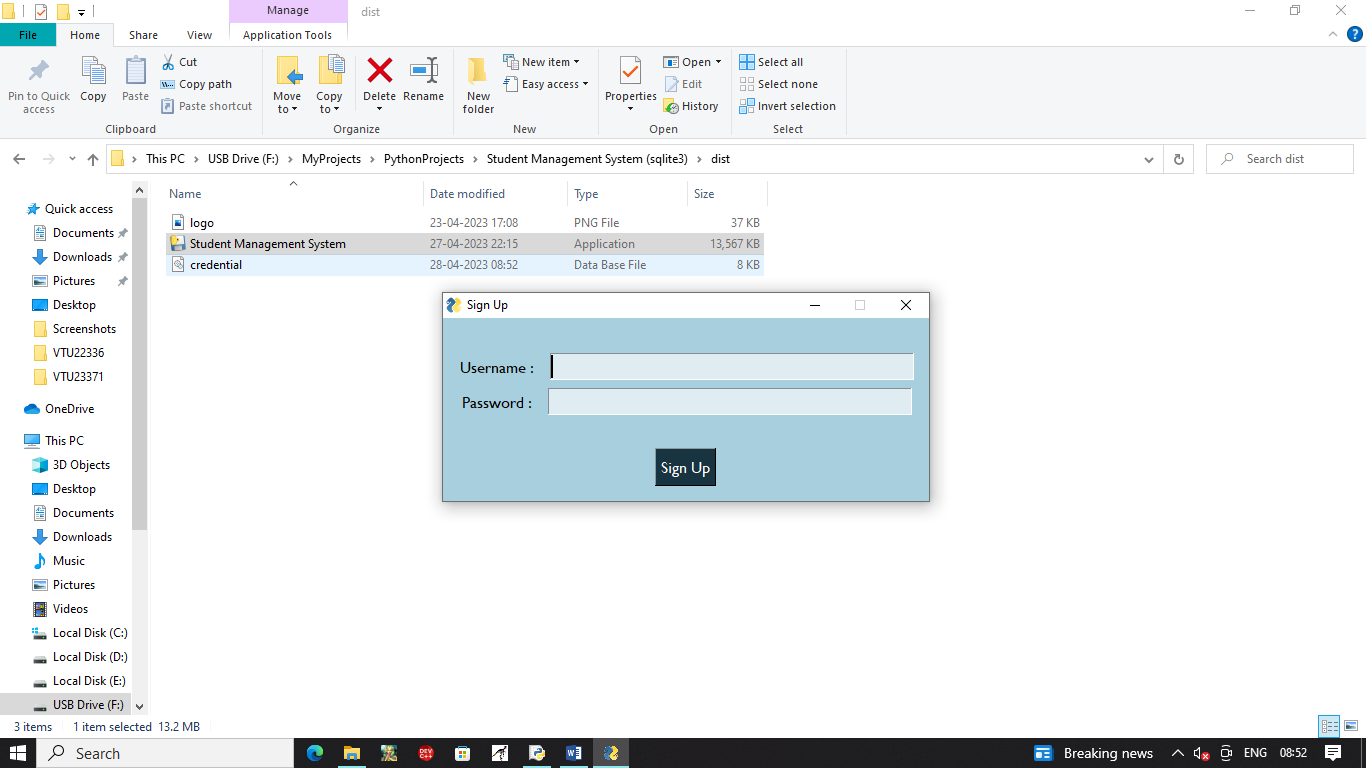
COURSE CODE: ***10210CS104***

DATE OF SUBMISSION: ***28 APRIL 2023***

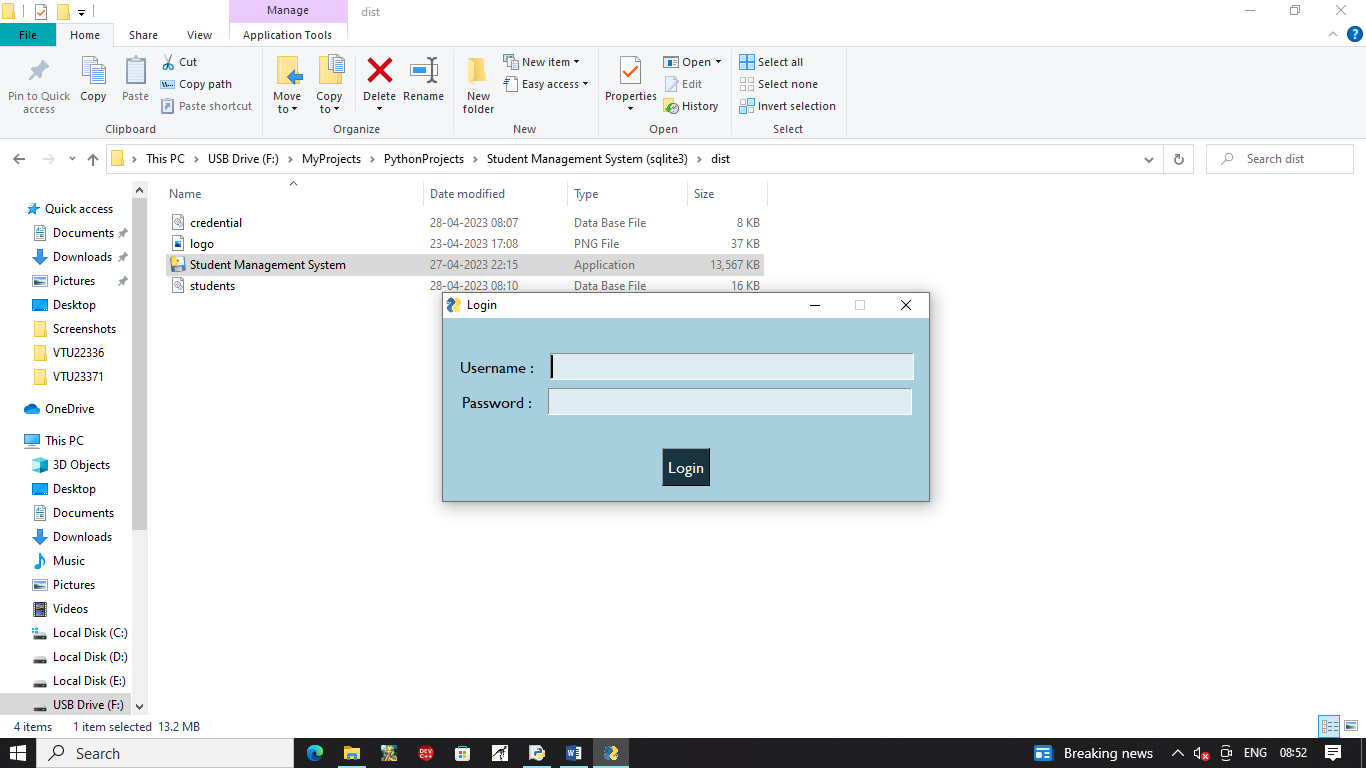
**STUDENT MANAGEMENT SYSTEM**

***Abstract :***

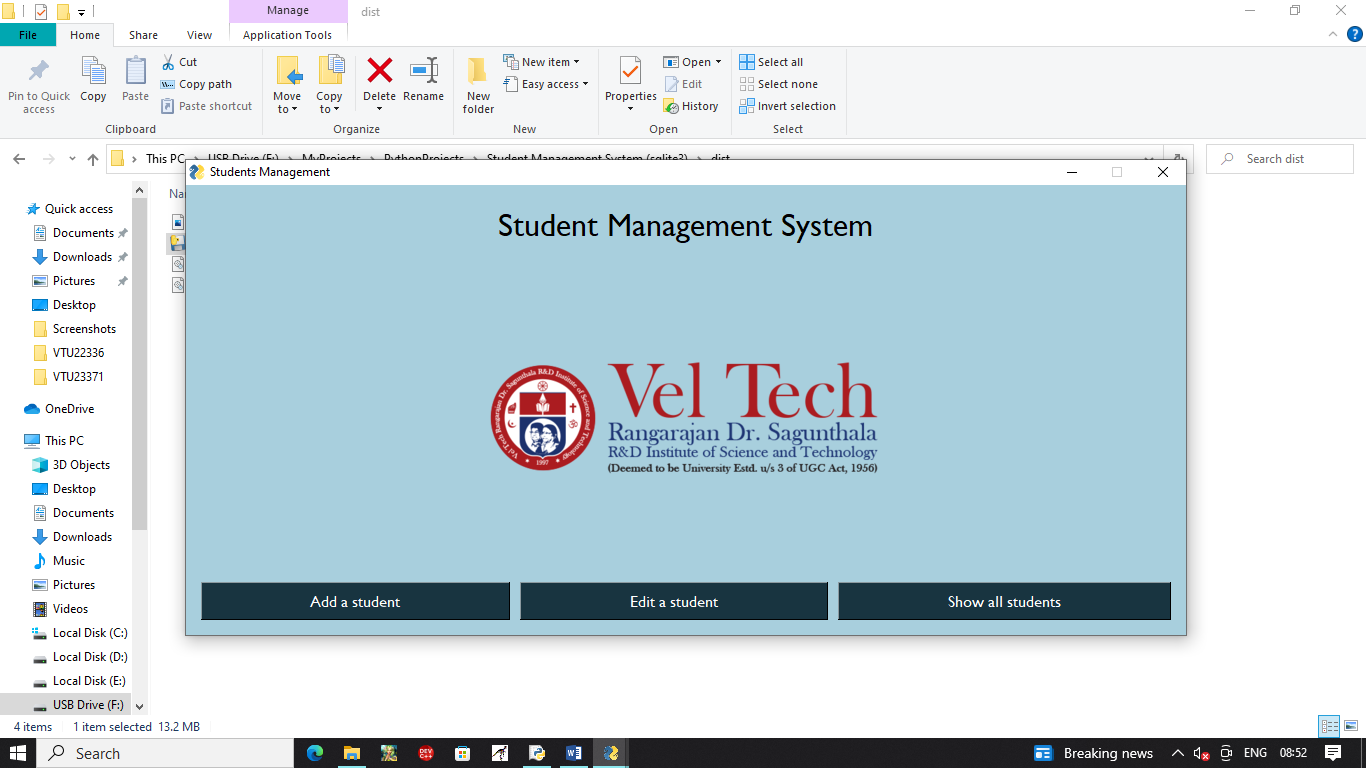
A student management system is a software application designed to streamline the management of student-related data in educational institutions. This helps educators and administrators to efficiently manage student records, including enrolment, editing student details and grades. The system provides a centralized platform for monitoring and analysing a student. Additionally the details of the students are organised and can be accessed, modified or deleted easily. Overall, this software allows an educational institute to organise its students.

**Sign Up / Login:**

When a user is running the application for the first time, the user has to first Sign Up with the username and password of user’s choice.



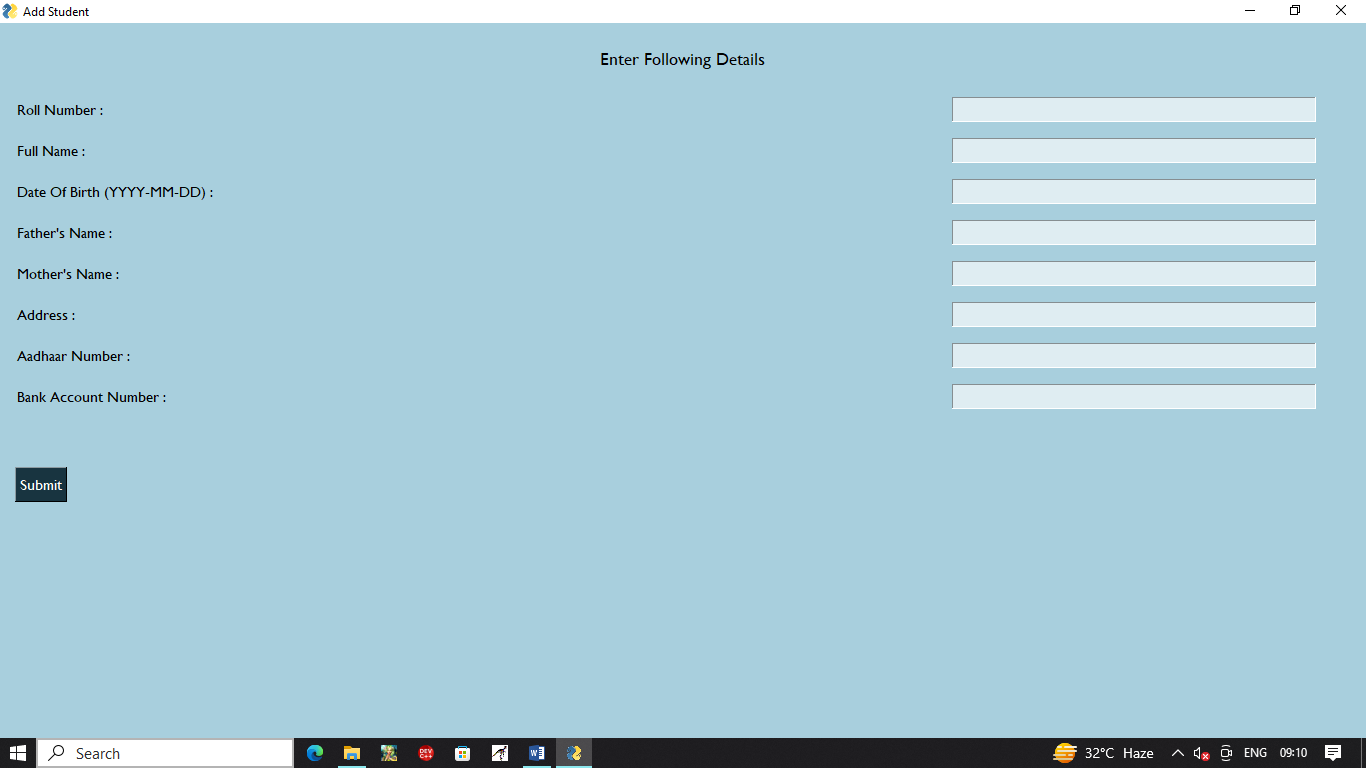
When the user had already signed up, user needs to give the same username and password to get access into the software

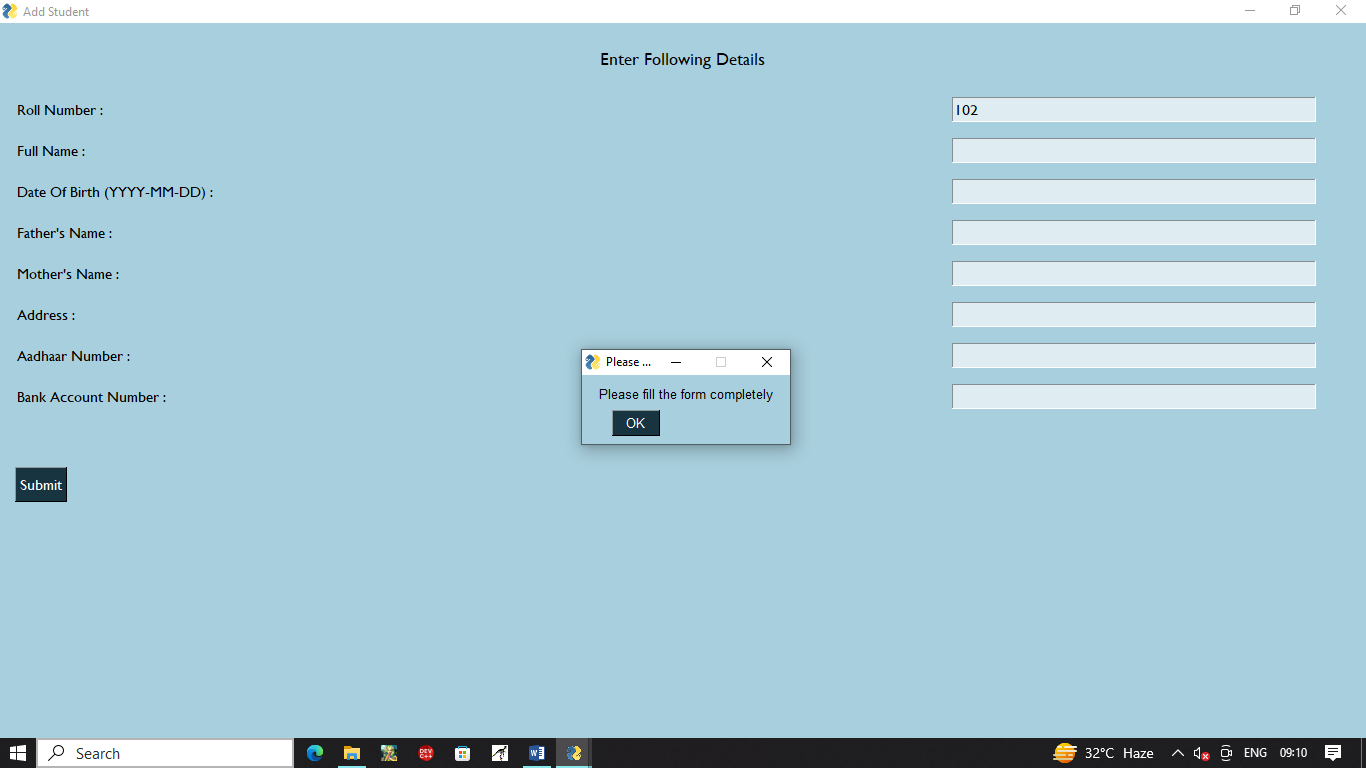
**Home Window:**

This window has three options:

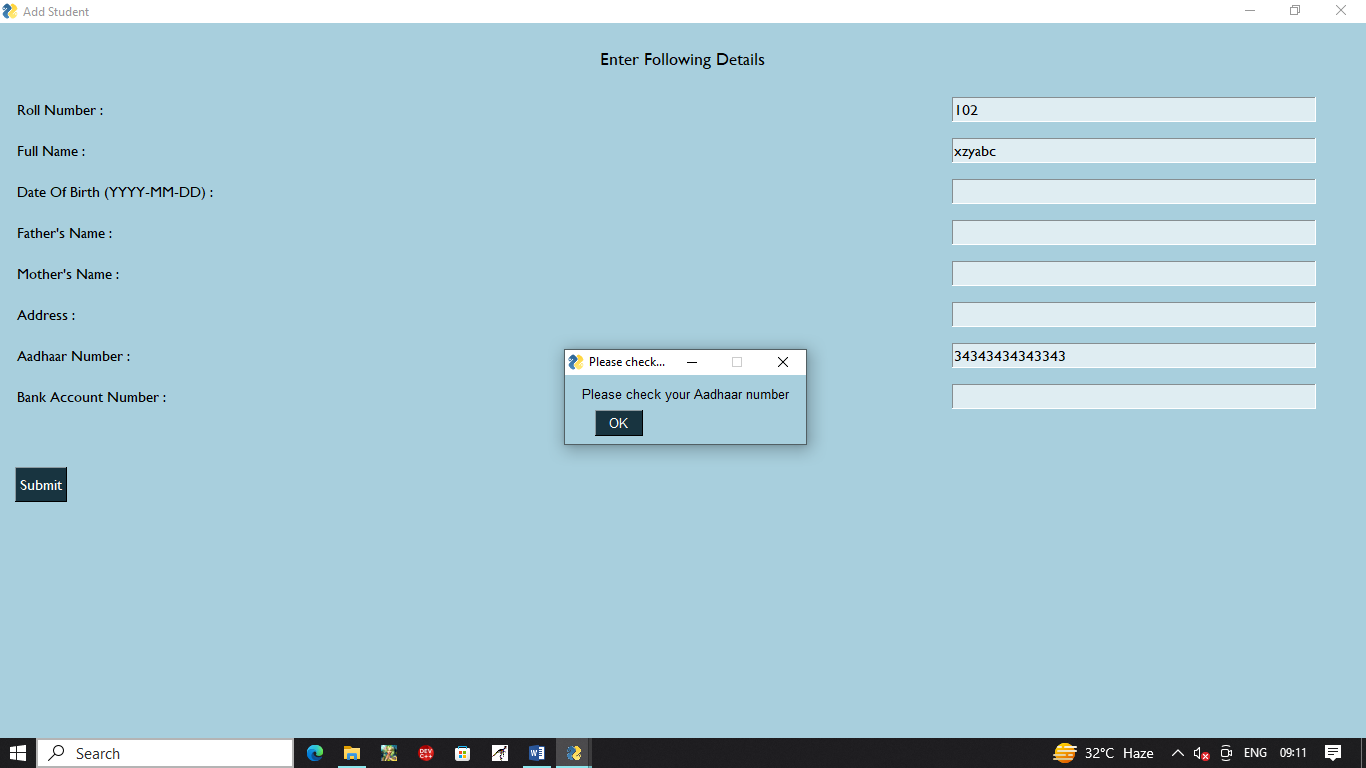
* Add a student
* Edit a student
* Show all students

**Add Student Window:**

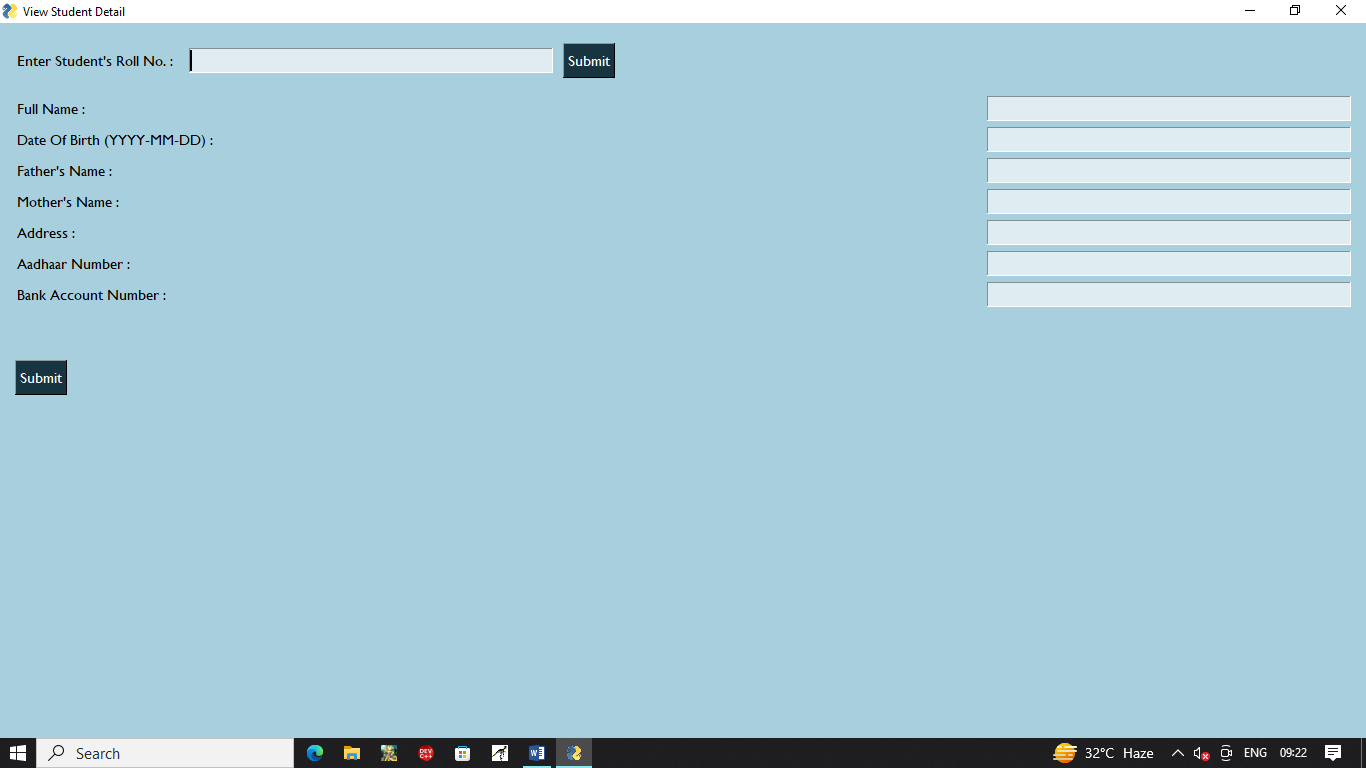
This window will allow the user to add a student to the database



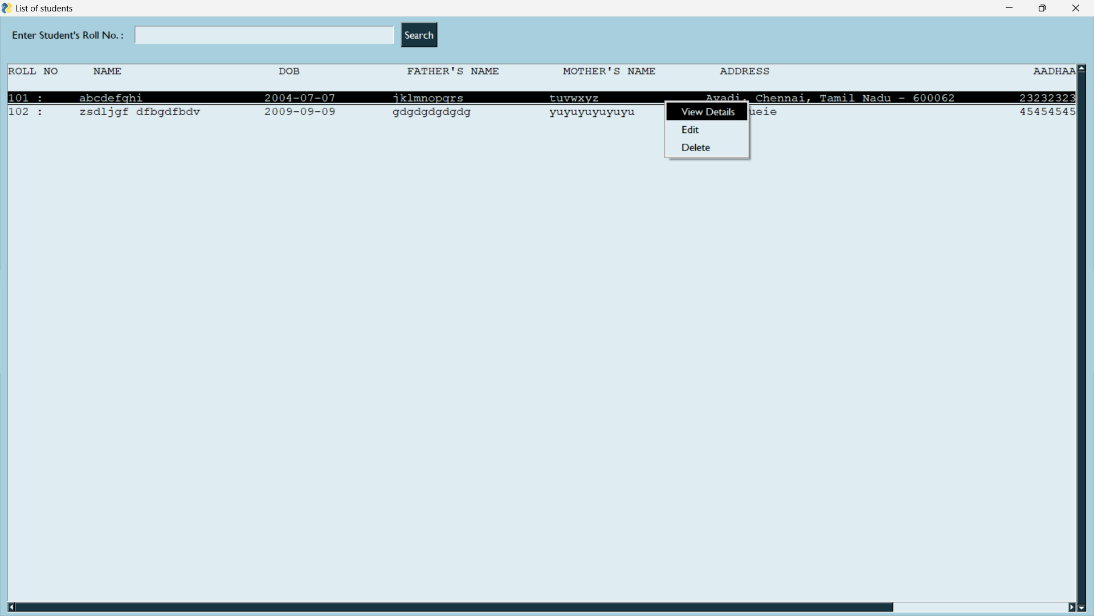
If the form is not filled completely, it will show a pop up window to instruct the use to fill the details completely

Since an Aadhaar number is a group of 12 digits, the software will make sure that the user has entered 12 digits. Otherwise it will show a pop up window instructing the user to check the Aaadhaar number

Similarly, many other errors or features has been added.

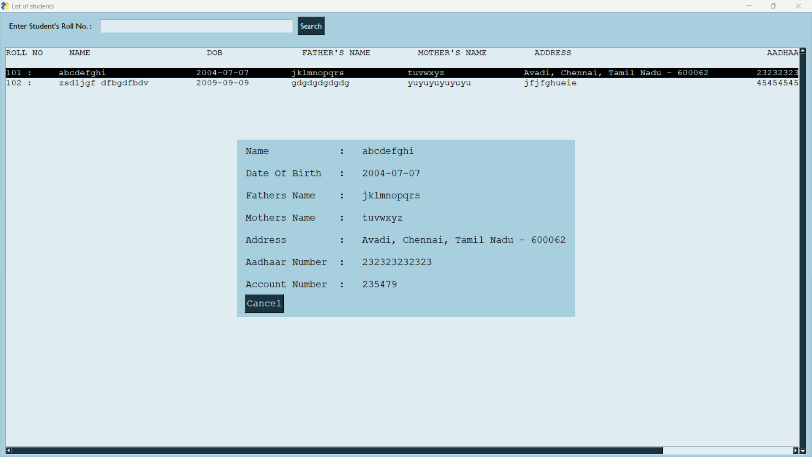
**Edit Student Window:**

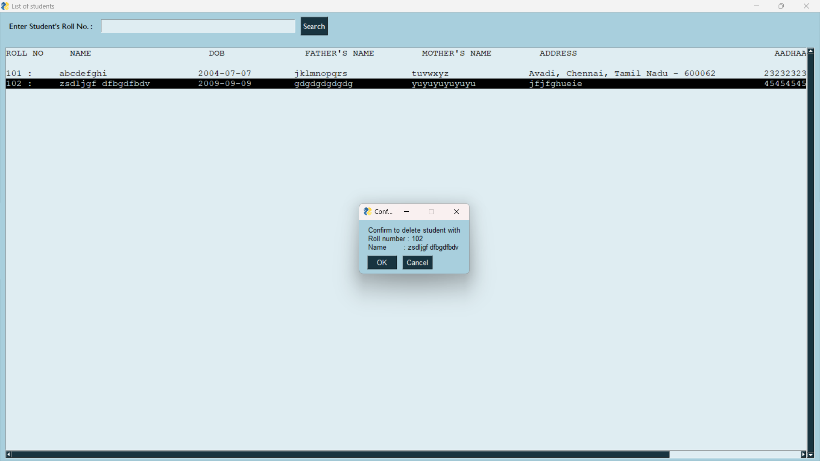
This window will allow the user to edit any of the details of the student. It searches by the roll number.

**Show All Students Window:**

This window allows the user to view all the students present in the database.

There are three options when the user right clicks on a particular student: -



* **View Details :** This will show all the details of a student in a banner.
* **Edit:** This option allows the user to directly edit the student details without going to the Edit Student Window.
* **Delete:** This option is used to delete a student from the database.

The user can also search for a particular student in the database using the roll number.



***CODE :***

*"""  
This program is developed by Habeeb Rahman K T (VTU22336)  
"""*

import sqlite3  
import PySimpleGUI as Sg  
import sys  
import os  
  
BUNDLE\_DIRECTORY = getattr(sys, '\_MEIPASS', os.path.abspath(os.path.dirname(\_\_file\_\_)))  
CREDENTIAL\_PATH = "credential.db"  
STUDENTS\_DATA\_PATH = "students.db"  
LOGO\_PATH = BUNDLE\_DIRECTORY + "\\logo.png"  
  
# Hiding files  
# subprocess.check\_call(["attrib", "+H", CREDENTIAL\_PATH])  
# subprocess.check\_call(["attrib", "+H", STUDENTS\_DATA\_PATH])  
  
WINDOW\_CLOSED = 'Window is closed'  
ACCESS\_GRANTED = 'Username and password is correct'  
ACCESS\_DENIED = 'incorrect password or username'  
NOT\_FOUND = 'NO\_DATA\_FOUND'  
  
  
def extract\_roll\_number\_from\_data(data):  
 roll\_number = NOT\_FOUND  
 try:  
 roll\_number = int(data[0][0:3]) # Instead of 3, enter the number of digits in roll number {Bug to be fixed}  
 except ValueError:  
 print('Value Error!')  
 return roll\_number  
  
  
# This function gets all the details of a student from database  
# by taking student roll number as argument  
# and returns a tuple of details ONLY if the roll number is present in the table  
def get\_details\_with\_roll\_no(\_roll\_number):  
 try:  
 cursor.execute(f"SELECT \* FROM students\_details WHERE Roll\_No = {\_roll\_number}")  
 details = cursor.fetchone()  
  
 except sqlite3.OperationalError:  
 Sg.popup\_ok("Enter a valid roll number to continue")  
 else:  
 return details  
 return None  
  
  
# Beautifying data for the user  
def justify\_details(details):  
 longest\_student\_name\_length = 4  
 longest\_father\_name\_length = 13  
 longest\_mother\_name\_length = 13  
 longest\_address\_length = 7  
 for student\_details in details:  
 if longest\_student\_name\_length < len(student\_details[1]):  
 longest\_student\_name\_length = len(student\_details[1])  
 if longest\_father\_name\_length < len(student\_details[3]):  
 longest\_father\_name\_length = len(student\_details[3])  
 if longest\_mother\_name\_length < len(student\_details[4]):  
 longest\_mother\_name\_length = len(student\_details[4])  
 if longest\_address\_length < len(student\_details[5]):  
 longest\_address\_length = len(student\_details[5])  
  
 # Making the details of the student cleaner to view  
 students\_details\_to\_view = []  
 for student\_details in details:  
 blank\_space\_after\_student\_name = longest\_student\_name\_length - len(student\_details[1]) + 8  
 blank\_space\_after\_father\_name = longest\_father\_name\_length - len(student\_details[3]) + 8  
 blank\_space\_after\_mother\_name = longest\_mother\_name\_length - len(student\_details[4]) + 8  
 blank\_space\_after\_address = longest\_address\_length - len(student\_details[5]) + 8  
  
 edited\_data = f'{student\_details[0]} : ' \  
 f'{student\_details[1]} {blank\_space\_after\_student\_name \* " "}' \  
 f'{student\_details[2]}{" " \* 8}' \  
 f'{student\_details[3]} {blank\_space\_after\_father\_name \* " "}' \  
 f'{student\_details[4]} {blank\_space\_after\_mother\_name \* " "}' \  
 f'{student\_details[5]} {blank\_space\_after\_address \* " "}' \  
 f'{student\_details[6]}{" " \* 8}' \  
 f'{student\_details[7]}'  
  
 students\_details\_to\_view.append(edited\_data)

students\_details\_to\_view.insert(0, f'ROLL NO '  
 f'NAME{(longest\_student\_name\_length + 5) \* " "}'  
 f'DOB{" " \* 15}'  
 f'FATHER\'S NAME{(longest\_father\_name\_length - 4) \* " "}'  
 f'MOTHER\'S NAME{(longest\_mother\_name\_length - 4) \* " "}'  
 f'ADDRESS{(longest\_address\_length + 2) \* " "}'  
 f'AADHAAR NO. {" " \* 8}'  
 f'BANK ACCOUNT NO.')  
 students\_details\_to\_view.insert(1, " ")  
 return students\_details\_to\_view

def get\_all\_students():  
 cursor.execute('SELECT \* FROM students\_details')  
 students = cursor.fetchall()  
  
 if not students:  
 return NOT\_FOUND  
  
 justified\_students\_details = justify\_details(students)  
 return justified\_students\_details  
  
  
def delete\_student(roll\_no):  
 cursor.execute(f'DELETE FROM students\_details WHERE Roll\_No = {roll\_no}')  
 student\_db\_conn.commit()  
  
  
# Window for viewing the list of all students  
def show\_all\_students():  
 students = get\_all\_students()  
 print(students)  
 if students == NOT\_FOUND:  
 students = ['No student added...']  
  
 window\_layout = [  
 [  
 Sg.Text("Enter Student's Roll No. : "),  
 Sg.Input(key='STUDENT\_ON\_ROLL'),  
 Sg.Button(key='SEARCH', button\_text='Search'),  
 Sg.Button(key='CLEAR', button\_text='Clear filter', visible=False)  
 ],  
 [Sg.Listbox(key='STUDENTS\_LIST', values=students, expand\_x=True, pad=(0, (20, 0)),  
 expand\_y=True, font=('Courier', 12), enable\_events=True,  
 right\_click\_menu=['', ['View Details', 'Edit', 'Delete']], horizontal\_scroll=True)]  
 ]  
  
 window = Sg.Window('List of students', layout=window\_layout, font=("Gill Sans MT", 11),  
 finalize=True, resizable=True)  
 window.maximize()  
  
 while True:  
 event, values = window.read()  
  
 if event == 'Exit' or event == Sg.WIN\_CLOSED:  
 window.close()  
 break  
  
 elif event == 'View Details':  
 roll\_number = extract\_roll\_number\_from\_data(values['STUDENTS\_LIST'])  
 print(roll\_number)  
 if roll\_number != NOT\_FOUND:  
 student\_details = get\_details\_with\_roll\_no(roll\_number)  
 edited\_details = f"Name : {student\_details[1]}\n\n" \  
 f"Date Of Birth : {student\_details[2]}\n\n" \  
 f"Fathers Name : {student\_details[3]}\n\n" \  
 f"Mothers Name : {student\_details[4]}\n\n" \  
 f"Address : {student\_details[5]}\n\n" \  
 f"Aadhaar Number : {student\_details[6]}\n\n" \  
 f"Account Number : {student\_details[7]}"  
  
 Sg.popup(edited\_details, custom\_text='Cancel', no\_titlebar=True,  
 font=('Courier', 14), grab\_anywhere=False)  
  
 elif event == 'Edit':  
 roll\_number = extract\_roll\_number\_from\_data(values['STUDENTS\_LIST'])  
  
 if roll\_number != NOT\_FOUND:  
 window.close()  
 edit\_student\_window(roll\_number)  
 show\_all\_students()  
  
 elif event == 'Delete':  
 roll\_number = extract\_roll\_number\_from\_data(values['STUDENTS\_LIST'])  
  
 if roll\_number != NOT\_FOUND:  
 deletion\_details = get\_details\_with\_roll\_no(roll\_number)  
 text\_to\_show = f'Confirm to delete student with\n' \  
 f'Roll number : {roll\_number}\n' \  
 f'Name : {deletion\_details[1]}'  
 user\_conformation = Sg.popup\_ok\_cancel(text\_to\_show)

if user\_conformation == 'OK':  
 delete\_student(roll\_number)  
 students = get\_all\_students()  
 window['STUDENTS\_LIST'].update(values=students)  
  
 elif event == 'SEARCH':  
 \_roll\_number = values['STUDENT\_ON\_ROLL']  
  
 if \_roll\_number == '':  
 window['STUDENTS\_LIST'].update(values=students)  
 continue  
  
 details = get\_details\_with\_roll\_no(\_roll\_number)  
 if details is None:  
 Sg.popup\_ok("No student is registered under this roll number")

else:  
 details = [tuple(details)]  
 justified\_student\_details = justify\_details(details)  
 window['STUDENTS\_LIST'].update(values=justified\_student\_details)  
 window['SEARCH'].update(visible=False)  
 window['CLEAR'].update(visible=True)  
  
 elif event == 'CLEAR':  
 window['STUDENTS\_LIST'].update(values=students)  
 window['CLEAR'].update(visible=False)  
 window['SEARCH'].update(visible=True)  
  
  
# Window for editing details of an existing student  
def edit\_student\_window(roll\_number=NOT\_FOUND):  
 edit\_window\_layout = [  
 [  
 Sg.Text("Enter Student's Roll No. : ", key='ON\_ROLL\_MESSAGE', pad=(5, 20)),  
 Sg.Input(key='STUDENT\_ON\_ROLL'),  
 Sg.Submit(key='ON\_ROLL\_SUBMIT')  
 ],  
 [  
 Sg.Text("Full Name : ", expand\_x=True),  
 Sg.InputText(key="NAME", default\_text="")  
 ],  
 [  
 Sg.Text("Date Of Birth (YYYY-MM-DD) : ", expand\_x=True),  
 Sg.InputText(key="DOB")  
 ],  
 [  
 Sg.Text("Father's Name : ", expand\_x=True),  
 Sg.InputText(key="FATHER\_NAME")  
 ],  
 [  
 Sg.Text("Mother's Name : ", expand\_x=True),  
 Sg.InputText(key="MOTHER\_NAME")  
 ],  
 [  
 Sg.Text("Address : ", expand\_x=True),  
 Sg.InputText(key="ADDRESS")  
 ],  
 [  
 Sg.Text("Aadhaar Number : ", expand\_x=True),  
 Sg.InputText(key="AADHAAR\_NO")  
 ],  
 [  
 Sg.Text("Bank Account Number : ", expand\_x=True),  
 Sg.InputText(key="BANK\_ACCOUNT\_NO")  
 ],  
 [  
 Sg.Submit(key="EDIT\_SUBMIT", pad=(5, 50))  
 ]  
 ]  
  
 edit\_window = Sg.Window(title="View Student Detail", layout=edit\_window\_layout, font=("Gill Sans MT", 11),  
 resizable=True, finalize=True)  
 edit\_window.maximize()  
  
 if roll\_number != NOT\_FOUND:  
 edit\_window['ON\_ROLL\_MESSAGE'].update(visible=False)  
 edit\_window['ON\_ROLL\_SUBMIT'].update(visible=False)  
  
 details = get\_details\_with\_roll\_no(roll\_number)  
 if details is None:  
 Sg.popup\_ok("No Student Is Registered Under This Roll Number")  
  
 else:  
 edit\_window['STUDENT\_ON\_ROLL'].update(roll\_number)  
 edit\_window['NAME'].update(details[1])  
 edit\_window['DOB'].update(details[2])  
 edit\_window['FATHER\_NAME'].update(details[3])  
 edit\_window['MOTHER\_NAME'].update(details[4])  
 edit\_window['ADDRESS'].update(details[5])  
 edit\_window['AADHAAR\_NO'].update(details[6])  
 edit\_window['BANK\_ACCOUNT\_NO'].update(details[7])

while True:  
 event, values = edit\_window.read()  
  
 if event == 'Exit' or event == Sg.WIN\_CLOSED:  
 edit\_window.close()  
 break  
  
 elif event == 'ON\_ROLL\_SUBMIT':  
 roll\_number = values['STUDENT\_ON\_ROLL']

try:  
 details = get\_details\_with\_roll\_no(roll\_number)  
  
 except sqlite3.OperationalError:  
 Sg.popup\_ok("Entered roll number is invalid")  
 continue

else:  
 try:  
 edit\_window['NAME'].update(details[1])  
 edit\_window['DOB'].update(details[2])  
 edit\_window['FATHER\_NAME'].update(details[3])  
 edit\_window['MOTHER\_NAME'].update(details[4])  
 edit\_window['ADDRESS'].update(details[5])  
 edit\_window['AADHAAR\_NO'].update(details[6])  
 edit\_window['BANK\_ACCOUNT\_NO'].update(details[7])  
  
 except TypeError:  
 Sg.popup\_ok("No student is registered under this roll number")  
 continue  
  
 elif event == 'EDIT\_SUBMIT':  
  
 \_roll\_number = values['STUDENT\_ON\_ROLL']  
 details = get\_details\_with\_roll\_no(\_roll\_number)  
  
 if details is not None:  
 new\_details = (int(values['STUDENT\_ON\_ROLL']),  
 values['NAME'],  
 values['DOB'],  
 values['FATHER\_NAME'],  
 values['MOTHER\_NAME'],  
 values['ADDRESS'],  
 values['AADHAAR\_NO'],  
 values['BANK\_ACCOUNT\_NO'])  
  
 if new\_details == details:  
 Sg.popup\_ok("No data changed")  
 continue  
  
 Sg.popup\_ok("Confirm to change data")  
 cursor.execute(f"DELETE FROM students\_details WHERE Roll\_No = {\_roll\_number}")  
  
 add\_student\_query = "INSERT INTO students\_details (Roll\_No, Name, " \  
 "Date\_Of\_Birth, Fathers\_Name, Mothers\_Name, Address, Aadhaar\_No, " \  
 "Bank\_Account\_Number)" \  
 " VALUES (?, ?, date(?), ?, ?, ?, ?, ?) "  
  
 try:  
 cursor.execute(add\_student\_query, (\_roll\_number, # Roll Number  
 new\_details[1], # Name  
 new\_details[2], # Date Of Birth  
 new\_details[3], # Father's Name  
 new\_details[4], # Mother's Name  
 new\_details[5], # Address  
 new\_details[6], # Aadhaar Number  
 new\_details[7])) # Bank Account Number  
 student\_db\_conn.commit()  
  
 except sqlite3.IntegrityError:  
 Sg.popup\_ok("Check the format of date of birth (YYYY-MM-DD)")  
 else:  
 Sg.popup\_ok("Data Changed!")  
  
  
# Window for Adding A Student  
def add\_student\_window():  
 add\_student\_layout = [  
 [  
 Sg.Text("Enter Following Details", justification="center/top",  
 font=("Gill Sans MT", 13), expand\_x=True, pad=(0, 16))  
 ],  
 [  
 Sg.Text("Roll Number : ", expand\_x=True),  
 Sg.InputText(key="ROLL\_NO", pad=(40, 8))  
 ],  
 [  
 Sg.Text("Full Name : ", expand\_x=True),  
 Sg.InputText(key="NAME", pad=(40, 8))  
 ],

[  
 Sg.Text("Date Of Birth (YYYY-MM-DD) : ", expand\_x=True),  
 Sg.InputText(key="DOB", pad=(40, 8))  
 ],  
 [  
 Sg.Text("Father's Name : ", expand\_x=True),  
 Sg.InputText(key="FATHER\_NAME", pad=(40, 8))  
 ],  
 [  
 Sg.Text("Mother's Name : ", expand\_x=True),  
 Sg.InputText(key="MOTHER\_NAME", pad=(40, 8))  
 ],  
 [  
 Sg.Text("Address : ", expand\_x=True),  
 Sg.InputText(key="ADDRESS", pad=(40, 8))  
 ],  
 [  
 Sg.Text("Aadhaar Number : ", expand\_x=True),  
 Sg.InputText(key="AADHAAR\_NO", pad=(40, 8))  
 ],

[  
 Sg.Text("Bank Account Number : ", expand\_x=True),  
 Sg.InputText(key="BANK\_ACCOUNT\_NO", pad=(40, 8))  
 ],  
 [  
 Sg.Submit(key="SUBMIT", pad=(5, 50))  
 ]  
 ]  
  
 add\_window = Sg.Window(title="Add Student", layout=add\_student\_layout, element\_padding=(5, 5), resizable=True,  
 font=("Gill Sans MT", 11)).finalize()  
 add\_window.maximize()  
  
 while True:  
 event, values = add\_window.read()  
  
 if event == 'Exit' or event == Sg.WIN\_CLOSED:  
 add\_window.close()  
 break  
  
 elif event == "SUBMIT":  
 try:  
 roll\_no = int(values["ROLL\_NO"])  
 except ValueError:  
 Sg.popup\_ok('Enter a valid roll number')  
 continue  
  
 name = values["NAME"].lstrip(' ')  
 fathers\_name = values["FATHER\_NAME"].lstrip(' ')  
 mothers\_name = values["MOTHER\_NAME"].lstrip(' ')  
 address = values["ADDRESS"].lstrip(' ')  
 aadhaar\_no = values["AADHAAR\_NO"].lstrip(' ')  
 bank\_account\_no = values["BANK\_ACCOUNT\_NO"].lstrip(' ')  
  
 if roll\_no == 0:  
 Sg.popup\_ok('Zero cannot be assigned as a roll number')  
 continue  
  
 if (name, fathers\_name, mothers\_name, address) == ('', '', '', ''):  
 Sg.popup\_ok('Please fill the form completely')  
 continue  
  
 if len(aadhaar\_no) != 12:  
 Sg.popup\_ok('Please check your Aadhaar number')  
 continue  
  
 else:  
  
 add\_student\_query = "INSERT INTO students\_details (Roll\_No, Name, Date\_Of\_Birth, " \  
 "Fathers\_Name, Mothers\_Name, Address, Aadhaar\_No, Bank\_Account\_Number)" \  
 " VALUES (?,?,date(?),?,?,?,?,?) "  
  
 try:  
 cursor.execute(add\_student\_query, (roll\_no, name, values['DOB'], fathers\_name,  
 mothers\_name, address, aadhaar\_no, bank\_account\_no))  
 student\_db\_conn.commit()  
  
 except sqlite3.IntegrityError:  
 Sg.popup\_ok("hAnother student is already registered under this roll number!")  
  
 else:  
 add\_window.close()  
 break

# Main window of the application  
def home\_window():  
 layout = [  
 [  
 Sg.Text("Student Management System", font=('Gill Sans MT', 24), expand\_x=True, justification="centre"),  
 ],  
 [Sg.Image(source=LOGO\_PATH, expand\_y=True, expand\_x=True)],  
 [  
 Sg.Button(key="ADD\_BUTTON", button\_text="Add a student", enable\_events=True, expand\_x=True),  
 Sg.Button(key="EDIT\_BUTTON", button\_text="Edit a student", enable\_events=True, expand\_x=True),  
 Sg.Button(key='STUDENTS\_LIST', button\_text='Show all students', enable\_events=True, expand\_x=True)  
 ],  
 ]

window = Sg.Window(title="Students Management", layout=layout, size=(1000, 450), element\_padding=(5, 10),  
 default\_button\_element\_size=(10, 1), font=('Gill Sans MT', 12))  
  
 while True:  
 event, value = window.read()  
  
 if event == 'Exit' or event == Sg.WIN\_CLOSED:  
 break  
  
 elif event == "ADD\_BUTTON":  
 window.close() # Main window is closed -> clicks 'Add Student' button for a new window  
 add\_student\_window() # Calling to show window to ADD A STUDENT  
 home\_window() # Recalling the main window, to prevent whole software from closing

elif event == "EDIT\_BUTTON":  
 window.close() # Main window is closed -> clicks 'Add Student' button for a new window  
 edit\_student\_window() # Calling to show window to EDIT STUDENT DETAILS  
 home\_window() # Recalling the main window, to prevent whole software from closing  
  
 elif event == "STUDENTS\_LIST":  
 window.close() # Main window is closed -> clicks 'Add Student' button for a new window  
 show\_all\_students() # Calling to show window to EDIT STUDENT DETAILS  
 home\_window() # Recalling the main window, to prevent whole software from closing  
  
 window.close()  
  
  
# Password Window For Confirming The User  
def get\_authenticated():  
 title = "Sign Up"  
 login\_conn = sqlite3.connect(CREDENTIAL\_PATH)  
 login\_cursor = login\_conn.cursor()  
  
 try:  
 login\_cursor.execute("""CREATE TABLE credentials (  
 username TEXT,  
 password TEXT)""")  
 except sqlite3.OperationalError:  
 print("Credential database already exist")  
 title = "Login"  
  
 finally:  
 password\_window\_layout = [  
 [  
 Sg.Text('Username : ', pad=(5, (30, 5))),  
 Sg.Input(key='USERNAME', pad=(5, (30, 5)))  
 ],  
 [  
 Sg.Text("Password : "),  
 Sg.Input(key="PASSWORD")  
 ],  
 [Sg.Submit(key='PASSWORD\_SUBMIT', button\_text=title, pad=(5, (30, 10)))]  
 ]  
  
 password\_window = Sg.Window(title=title, layout=password\_window\_layout, font=('Gill Sans MT', 12),  
 element\_justification='center')  
  
 while True:  
 event, values = password\_window.read()  
  
 if event == 'Exit' or event == Sg.WIN\_CLOSED:  
 password\_window.close()  
 return WINDOW\_CLOSED  
  
 elif event == 'PASSWORD\_SUBMIT':  
 username = values['USERNAME']  
 password = values['PASSWORD']  
  
 # If the user has already entered the credentials  
 if title == "Login":  
 login\_cursor.execute(f"""SELECT password FROM credentials  
 WHERE username = ?""", (username,))  
 saved\_pass = login\_cursor.fetchone()

if saved\_pass is not None and password == saved\_pass[0]:  
 login\_conn.close()  
 password\_window.close()  
 return True  
 else:  
 Sg.popup\_ok("Please enter a valid Username and Password")  
 else:  
 login\_cursor.execute("""INSERT INTO CREDENTIALS VALUES(?, ?)""", (username, password))  
 login\_conn.commit()  
 login\_conn.close()  
 password\_window.close()  
 return True  
  
  
Sg.theme("LightBlue3")  
if \_\_name\_\_ == "\_\_main\_\_":  
  
 access\_granted = get\_authenticated()  
 if access\_granted and access\_granted != WINDOW\_CLOSED:  
 student\_db\_conn = sqlite3.connect(STUDENTS\_DATA\_PATH)  
 cursor = student\_db\_conn.cursor()  
  
 try:  
 cursor.execute("""CREATE TABLE students\_details (  
 Roll\_No TEXT PRIMARY KEY ,  
 Name TEXT NOT NULL,   
 Date\_Of\_Birth TEXT NOT NULL,   
 Fathers\_Name TEXT,   
 Mothers\_Name TEXT,   
 Address TEXT,   
 Aadhaar\_No VARCHAR(12) NOT NULL UNIQUE,   
 Bank\_Account\_Number TEXT)""")

except sqlite3.OperationalError:  
 print("Database already exist")  
  
 finally:  
 home\_window()