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***

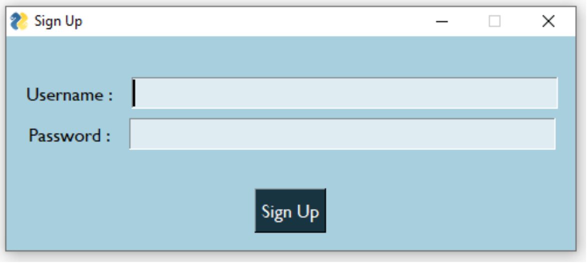
STUDENT MANAGEMENT SYSTEM

Abstract :

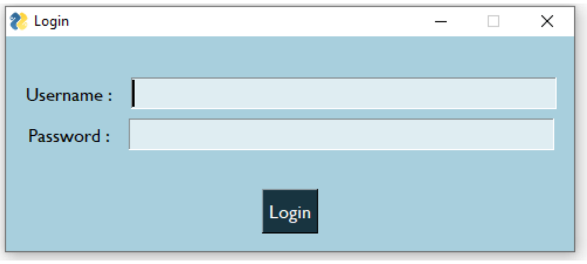
A student management system is a software application designed to streamline the management of student-related data in educational institutions. This helps educators and administrators to efficiently manage student records, including enrolment, editing student details and grades. The system provides 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:

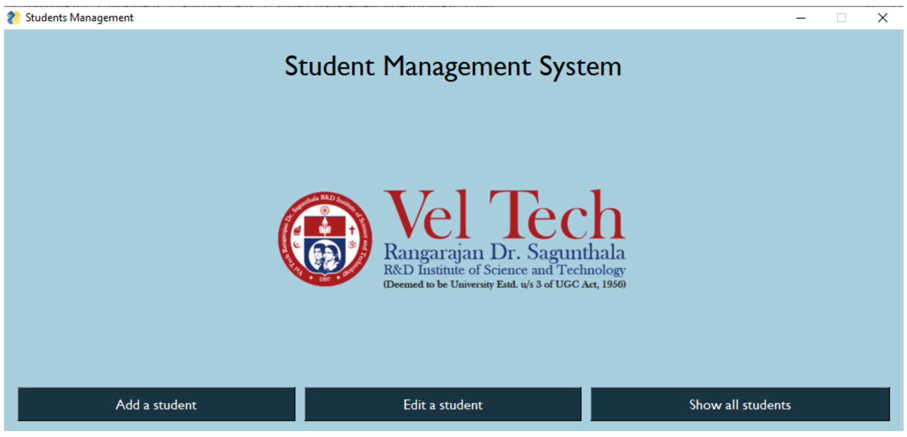
When a user is running the application for the first time, the user has to first Sign Up with the username and password of user's choice.



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



Home Window:

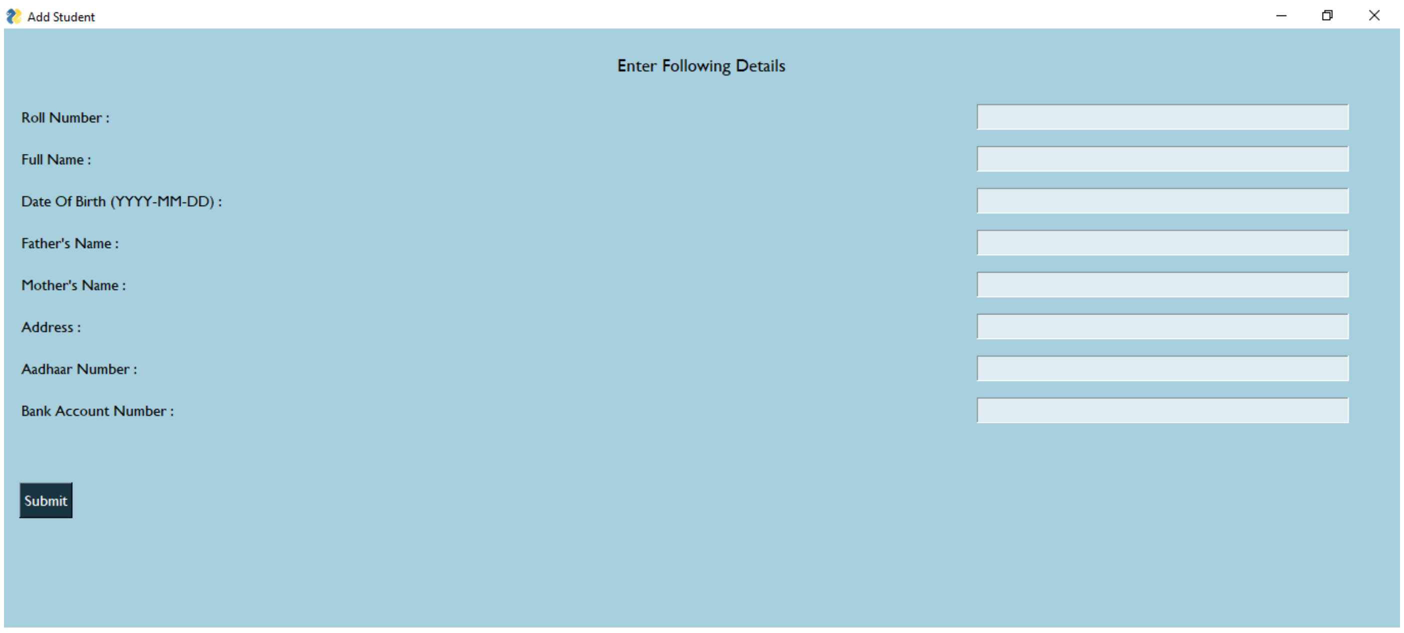

Students Management
Student Management System
Vel Tech
Rangarajan Dr. Sagunthala
R&D Institute of Science and Technology
(Deemed to be University Estd. u/s 3 of UGC Act, 1956)
Add a student
Edit a student
Show all students

This window has three options:

- Add a student
- Edit a student
- Show all students

Add Student Window:

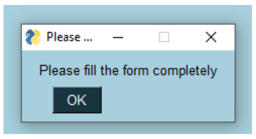
This window will allow the user to add a student to the database


Add Student

Enter Following Details

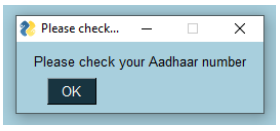
Roll Number :	
Full Name :	
Date Of Birth (YYYY-MM-DD) :	
Father's Name :	
Mother's Name :	
Address :	
Aadhaar Number :	
Bank Account Number :	

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


Please ...

Please fill the form 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

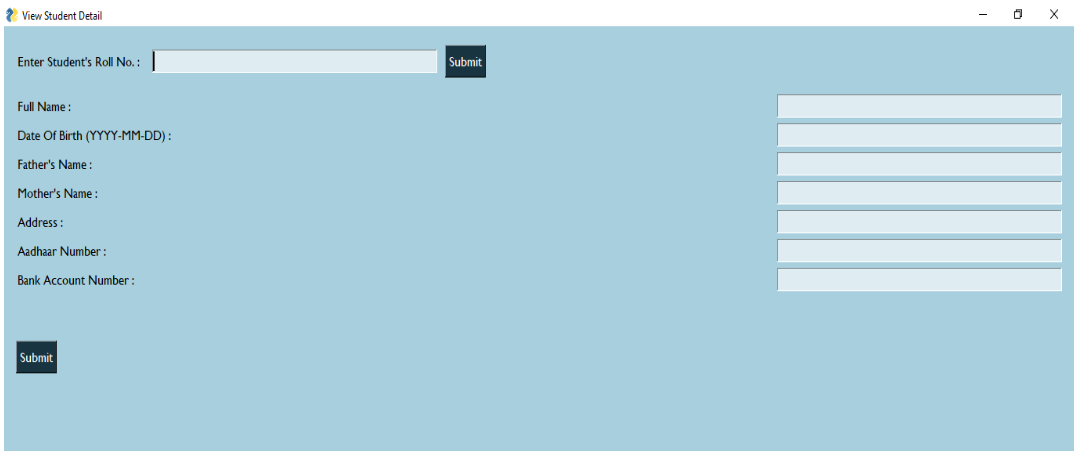

Please check...

Please check your Aadhaar 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.

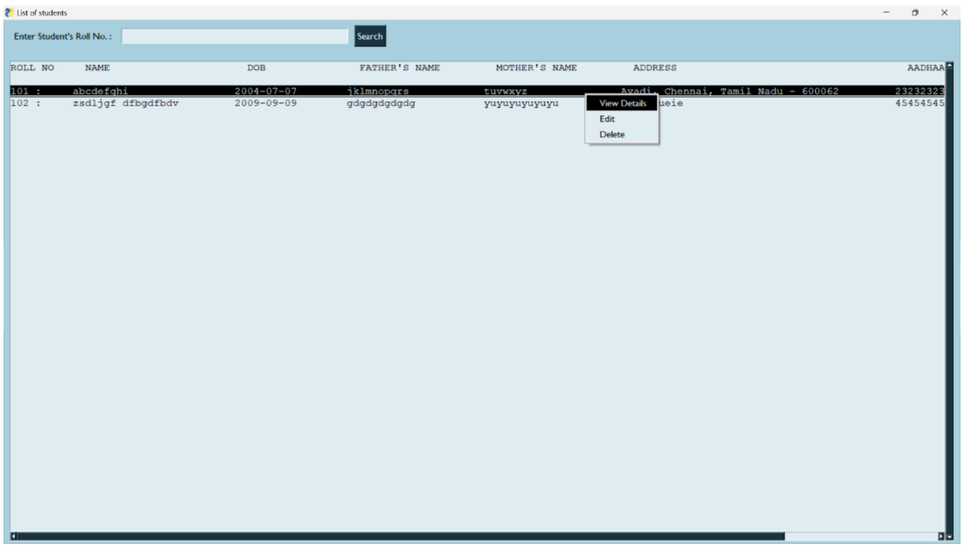

View Student Detail

Enter Student's Roll No. :

Full Name :	
Date Of Birth (YYYY-MM-DD) :	
Father's Name :	
Mother's Name :	
Address :	
Aadhaar Number :	
Bank Account Number :	

Show All Students Window:

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

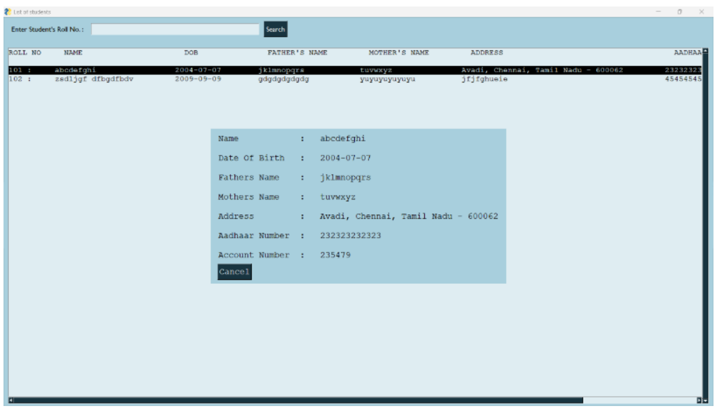

List of students

Enter Student's Roll No. :

ROLL NO	NAME	DOB	FATHER'S NAME	MOTHER'S NAME	ADDRESS	AADHAA
101 :	abcdefghi	2004-07-07	jklmnopqrs	tuvwxyz	Avadi, Chennai, Tamil Nadu - 600062	23232323
102 :	zsdljgf dfbgdfbdv	2009-09-09	gdgdgdgdgdg	yuyuyuyuyuyu	ueie	45454545

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.

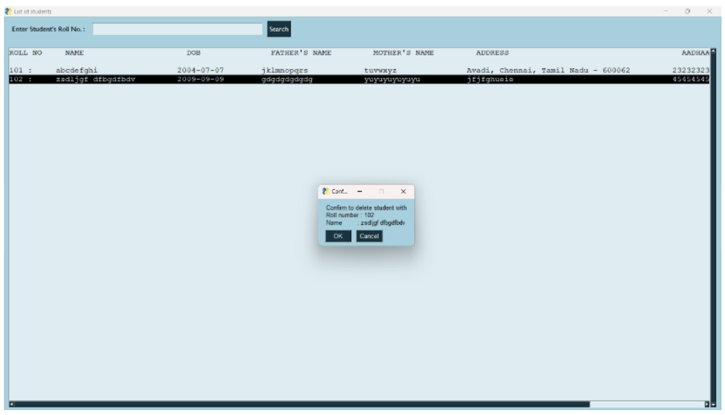

List of students

Enter Student's Roll No. :

ROLL NO	NAME	DOB	FATHER'S NAME	MOTHER'S NAME	ADDRESS	AADHAA
101 :	abcdefghi	2004-07-07	jklmnopqrs	tuvwxyz	Avadi, Chennai, Tamil Nadu - 600062	23232323
102 :	zsdljgf dfbgdfbdv	2009-09-09	gdgdgdgdgdg	yuyuyuyuyuyu	jfjfghueie	45454545

```
Name            :   abcdefghi
Date Of Birth   :   2004-07-07
Fathers Name    :   jklmnopqrs
Mothers Name    :   tuvwxyz
Address         :   Avadi, Chennai, Tamil Nadu - 600062
Aadhaar Number  :   232323232323
Account Number  :   235479
```

- **Edit:** This option allows the user to directly edit the student details without going to the Edit Student Window.


List of students

Enter Student's Roll No. :

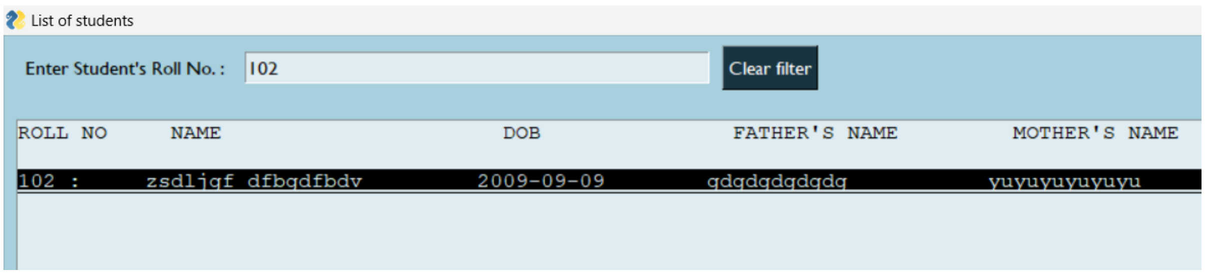
ROLL NO	NAME	DOB	FATHER'S NAME	MOTHER'S NAME	ADDRESS	AADHAA
101 :	abcdefghi	2004-07-07	jklmnopqrs	tuvwxyz	Avadi, Chennai, Tamil Nadu - 600062	23232323
102 :	zsdljgf dfbgdfbdv	2009-09-09	gdgdgdgdgdg	yuyuyuyuyuyu	jfjfghueie	45454545

Conf...

Confirm to delete student with
Roll number : 102
Name : zsdljgf dfbgdfbdv

- **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.


List of students

Enter Student's Roll No. : 102

ROLL NO	NAME	DOB	FATHER'S NAME	MOTHER'S NAME
102 :	zsdljgf dfbgdfbdv	2009-09-09	gdgdgdgdgdg	yuyuyuyuyuyu

CODE :

```python
"""
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
```

```python
# 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()
```