XII

COMPUTER SCIENCE

PROJECT

Aadhil Nandan

TABLE OF CONTENTS

INTRODUCTION

STUDENT MANAGEMENT SYSTEM

The Student Management System (SMS) is a software application designed to effectively manage student records.

This system streamlines student data management by handling admissions, personal information, and academic records. Built with Python and a MySQL database, the SMS project simplifies manual processes, ensuring quick and accurate data handling for users.

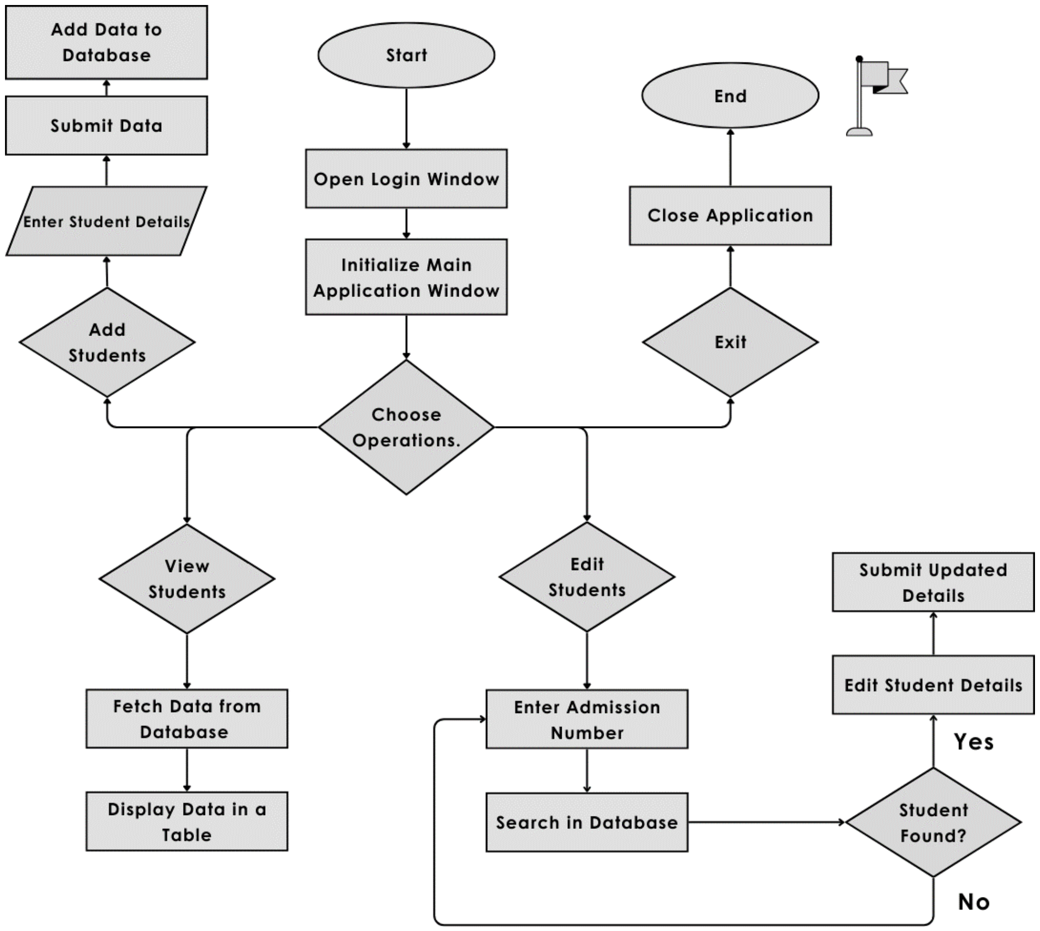
The system offers an intuitive interface that enables staff to navigate and manage tasks with ease. It also features secure storage in a MySQL database, ensuring the protection and privacy of sensitive student information.

Implementing an SMS in educational institutions enhances data management by centralizing student information, reducing administrative time and human error. This efficiency improves communication among staff, students, and parents, allowing educators to focus more on teaching. Additionally, an organized database facilitates report generation, academic tracking, and data-driven decision-making, leading to better educational outcomes.

FLOWCHART

FLOW OF OPERATIONS

1) *Start*
2) *The user selects the desired operation: Add, Edit, Search.*
3) *The system interacts with MySQL database.*
4) *Based on the user's input, the corresponding operation is executed.*
5) *The output is displayed, or the record is updated.*
6) *End*



DATABASE STRUCTURE

DATABASE NAME: SMS
TABLE NAME: STUDENTS

FIELD NAME	**DATA TYPE**	**CONSTRAINTS**
ADMN_NO	INT	PRIMARY KEY
NAME	VARCHAR (50)	NOT NULL
CLASS	VACHAR (4)	NOT NULL
AGE	INT	-
FATHER'S NAME	VARCHAR (50)	-
GENDER	VARCHAR (10)	-
PHONE NUMBER	BIGINT (10)	-
EMAIL	VARCHAR (255)	-
ADDRESS	VARCHAR (255)	-

SOURCE CODE

```
import customtkinter as ctk
import tkinter.messagebox as tkmb
import tkinter as tk
import mysql.connector as csql
from customtkinter import CTkFrame, CTkLabel, CTkEntry, CTkRadioButton

root = ctk.CTk()
```

```
# Basic personalization
ctk.set_appearance_mode("System")
ctk.set_default_color_theme("green")
s_width, s_height = root.winfo_screenwidth(), root.winfo_screenheight()
s_size = f"{s_width} x {s_height}"
root.geometry(s_size)
```

```
# Login Function
def login(new_window):
    # Credentials
    username = "Admin"
    password = "pass"

    # User Entry
    input_username = user_entry.get()
    input_password = user_pass.get()

    # For successful login
    if input_username == username and input_password == password:
        tkmb.showinfo(title="Login Successful", message="You have logged in
successfully")
        print(main_window(login_window))

    else:
        tkmb.showerror("Login Failed", "Invalid username and password.")
```

```
# Login window
def open_login_window():
    global login_window, user_entry, user_pass
    root.withdraw()
    login_window = ctk.CTkToplevel(root)
    login_window.title("Login")
    login_window.geometry(s_size)
    login_window.state('zoomed')

    # Frame and Buttons for Login Window
    login_frame = ctk.CTkFrame(login_window, width=500, height=1000,
corner_radius=15)
    login_frame.place(relx=0.5, rely=0.5, anchor="center")

    ctk.CTkLabel(login_frame, text="Login to SMS", font=("Arial", 18,
"bold")).pack(pady=20)

    user_entry = ctk.CTkEntry(login_frame, placeholder_text="Enter
Username")
    user_entry.pack(pady=10, padx=40, ipadx=30, ipady=5)

    user_pass = ctk.CTkEntry(login_frame, placeholder_text="Enter
Password", show="*")
    user_pass.pack(pady=10, padx=40, ipadx=30, ipady=5)

    login_button = ctk.CTkButton(login_frame, text="Login",
command=lambda: login(login_window))
    login_button.pack(pady=20, ipadx=20)

    exit_button = ctk.CTkButton(login_frame, text="Exit", command=exit_fn)
    exit_button.pack(pady=5, ipadx=20)
```

```
def main_window(login_win):
    # Main window
    root.deiconify()
    root.state('zoomed')
    login_win.destroy()
    root.title("Student Management System")

    # ============ Buttons =============== #

    btn_frame: CTkFrame = ctk.CTkFrame(root, corner_radius=10,
width=s_width // 8, height=s_height)
    btn_frame.pack(side=ctk.LEFT)

    fn_frame = ctk.CTkFrame(root, corner_radius=10, width=(s_width -
s_width // 8), height=s_height)
    fn_frame.pack(side=ctk.RIGHT)

    add_btn = ctk.CTkButton(btn_frame, text="Add Student Details",
width=290,
                            height=50, command=lambda:
addinfo_window(fn_frame))
    add_btn.grid(row=2, column=0, padx=10, pady=10)

    remove_btn = ctk.CTkButton(btn_frame, text="Edit Student Details",
width=290,
                               height=50, command=lambda:
edit_student_details(fn_frame))
    remove_btn.grid(row=3, column=0, padx=10, pady=10)

    display_btn = ctk.CTkButton(btn_frame, text="Display Student Details",
width=290,
                                height=50, command=lambda: view_details(fn_frame))
    display_btn.grid(row=4, column=0, padx=10, pady=10)

    exit_btn = ctk.CTkButton(btn_frame, text="Logout", width=290,
                             height=50, command=lambda: exit_fn())
    exit_btn.grid(row=5, column=0, padx=10, pady=10)

    appearance_mode_menu = ctk.CTkOptionMenu(btn_frame,
values=["Light", "Dark", "System"],
                                             command=lambda mode: change_mode(root,
mode))
    appearance_mode_menu.grid(row=8, column=0, padx=10, pady=100,
sticky="s")
```

```
    btn_head = ctk.CTkLabel(btn_frame, text='Welcome to SMS ',
fg_color="gray30",
                            corner_radius=6, font=("Georgia", 15))
    btn_head.grid(row=0, column=0, padx=5, pady=70, ipady=20,
sticky="ew")


def change_mode(main, new_appearance_mode):
    ctk.set_appearance_mode(new_appearance_mode)
```

```
def clear_frame(frame):
    for widget in frame.winfo_children():
        widget.destroy()


# Add student data side frame
def addinfo_window(frame):
    clear_frame(frame)
    root.title("Add Student Data")

    display_frame = ctk.CTkFrame(frame, corner_radius=5, height=(s_height-
500), width=(s_width - s_width//8))
    display_frame.grid(row=0, column=0, padx=5, pady=5, ipadx=5,
ipady=5, sticky="ew")

    add_title = ctk.CTkLabel(display_frame, text='Enter Student Details: ',
fg_color="gray30",
                             corner_radius=6, font=("Georgia", 15))
    add_title.grid(row=0, column=0, padx=5, pady=(10, 0), sticky="ew")

    root.geometry(s_size)

    # Scrollable Frame and widgets
    add_frame = ctk.CTkScrollableFrame(display_frame, width=(s_width -
s_width // 8) - 200,
                                       height=s_height - 100, corner_radius=10)
    add_frame.grid(row=1, column=0, padx=10, pady=10, sticky="nsew")

    admission_label = ctk.CTkLabel(add_frame, text="Admission No")
    admission_label.grid(row=1, column=0, padx=20, pady=20, sticky="ew")
    admission_entry = ctk.CTkEntry(add_frame, placeholder_text="Enter
Admission No")
```

```
    admission_entry.grid(row=1, column=1, columnspan=4, padx=20,
pady=20, sticky="ew")

    name_label = ctk.CTkLabel(add_frame, text="Name")
    name_label.grid(row=2, column=0, padx=20, pady=20, sticky="ew")
    name_entry = ctk.CTkEntry(add_frame, placeholder_text="Enter Name")
    name_entry.grid(row=2, column=1, columnspan=3, padx=20, pady=20,
sticky="ew")

    class_label = ctk.CTkLabel(add_frame, text="Class")
    class_label.grid(row=3, column=0, padx=20, pady=20, sticky="ew")
    classes = ["LKG", "UKG", '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12']
    class_option_menu = ctk.CTkOptionMenu(add_frame, values=classes)
    class_option_menu.grid(row=3, column=1, padx=20, pady=20,
sticky="ew")

    age_label = ctk.CTkLabel(add_frame, text="Age")
    age_label.grid(row=4, column=0, padx=20, pady=20, sticky="ew")
    age_entry = ctk.CTkEntry(add_frame, placeholder_text="Enter Age")
    age_entry.grid(row=4, column=1, columnspan=3, padx=20, pady=20,
sticky="ew")

    fathers_name_label: CTkLabel = ctk.CTkLabel(add_frame, text="Father's
Name")
    fathers_name_label.grid(row=5, column=0, padx=20, pady=20,
sticky="ew")
    fathers_name_entry: CTkEntry = ctk.CTkEntry(add_frame,
placeholder_text="Enter Father's Name")
    fathers_name_entry.grid(row=5, column=1, columnspan=3, padx=20,
pady=20, sticky="ew")

    gender_label = ctk.CTkLabel(add_frame, text="Gender")
    gender_label.grid(row=6, column=0, padx=20, pady=20, sticky="ew")
    gender_var = tk.StringVar(value="Prefer not to say")
    male_button: CTkRadioButton = ctk.CTkRadioButton(add_frame,
text="Male", variable=gender_var, value="Male")
    male_button.grid(row=6, column=1, padx=20, pady=20, sticky="ew")
    female_button: CTkRadioButton = ctk.CTkRadioButton(add_frame,
text="Female", variable=gender_var, value="Female")
    female_button.grid(row=6, column=2, padx=20, pady=20, sticky="ew")
    none_button: CTkRadioButton = ctk.CTkRadioButton(add_frame,
text="Prefer not to say", variable=gender_var, value="None")
    none_button.grid(row=6, column=3, padx=20, pady=20, sticky="ew")
    phone_label = ctk.CTkLabel(add_frame, text="Phone Number")
    phone_label.grid(row=7, column=0, padx=20, pady=20)
    phone_entry = ctk.CTkEntry(add_frame, placeholder_text="Enter Phone
number")
    phone_entry.grid(row=7, column=1, columnspan=3, padx=20, pady=20,
sticky="ew")

    email_label = ctk.CTkLabel(add_frame, text="Email Id")
    email_label.grid(row=8, column=0, padx=20, pady=20)
    email_entry = ctk.CTkEntry(add_frame, placeholder_text="Enter Email
Id")
    email_entry.grid(row=8, column=1, columnspan=3, padx=20, pady=20,
sticky="ew")

    address_label = ctk.CTkLabel(add_frame, text="Address")
    address_label.grid(row=9, column=0, padx=20, pady=20)
    address_entry = ctk.CTkEntry(add_frame, placeholder_text="Enter
Address")
    address_entry.grid(row=9, column=1, columnspan=3, padx=20,
pady=20, sticky="ew")

    def submit_data():
        student_data = (
            int(admission_entry.get()),
            name_entry.get(),
            class_option_menu.get(),
            int(age_entry.get()),
            fathers_name_entry.get(),
            gender_var.get(),
            int(phone_entry.get()),
            email_entry.get(),
            address_entry.get()
        )
        add_data(student_data, frame)

    submit_btn = ctk.CTkButton(add_frame, text="Submit",
command=lambda: submit_data())
    submit_btn.grid(row=10, column=2, columnspan=3, padx=3, pady=3)
```

```
def data():
    my_db = csql.connect(
        host="localhost",
        user="root",
        passwd="NSad*1807",
        database="sms"
    )
    my_cursor = my_db.cursor()


def add_data(x, frame):
    my_database = csql.connect(
        host="localhost",
        user="root",
        passwd="NSad*1807",
        database="sms"
    )

    my_cursor = my_database.cursor()
    add = """
    INSERT INTO students (admn_no, name, class, age, father_name,
gender, ph_no, email, address)
    VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)
    """
    my_cursor.execute(add, x)
    my_database.commit()
    tkmb.showinfo(title="Success", message="Data Added Successfully")
    addinfo_window(frame)
    my_database.close()


def db_connect():
    try:
        database = csql.connect(
            host="localhost",
            user="root",
            password="NSad*1807",
            database="sms"
        )
        return database
    except csql.Error as err:
        tkmb.showerror("Error", "Couldn't connect to database.")
        return None
```

```
def db_data_retriever():
    global data, headers
    extract_db = db_connect()
    cursor = extract_db.cursor()
    cursor.execute("SELECT * FROM students")
    data = cursor.fetchall()
    headers = [desc[0] for desc in cursor.description]
    extract_db.close()
    return data, headers
```

```
def view_details(frame):
    clear_frame(frame)
    root.title("View Student Data")
    view_frame = ctk.CTkFrame(frame, corner_radius=5, height=(s_height-
500), width=(s_width - s_width//8))
    view_frame.grid(row=0, column=0, padx=50, pady=50, ipadx=200,
ipady=50, sticky="ew")

    fn_frame = ctk.CTkScrollableFrame(view_frame, width=(s_width -
s_width // 8) - 200, height=s_height - 100,
                                      corner_radius=10)
    fn_frame.grid(row=1, column=0, padx=10, pady=20, sticky="nsew")

    def resize_canvas(event, table):
        table.configure(scrollregion=table.bbox("both"),
background="#2B2B2B",
                        width=(s_width - s_width // 8) - 250, height=s_height - 150)

    def load_data(table, colums):
        global data, headers
        try:

            data, header = db_data_retriever()
            display_table(table, colums)

        except Exception as e:
            tkmb.showerror("Error", f"Could not load database: {e}")

    def display_table(table, column):
        global data, headers
        headers = ["No", "Names     ", "Class", "Age", "Father's Name", "Gender",
"Phone", f"Email ID {" "*20 }", f"Address {" "*20}" ]
        for widget in column.winfo_children():
            widget.destroy()

        for col, header in enumerate(headers):
            header_label = ctk.CTkLabel(column, text=header, font=("Arial", 14,
"bold"), padx=20, pady=10)
            header_label.grid(row=0, column=col, sticky="nsew")

        for row, row_data in enumerate(data, start=1):
            for col, cell_data in enumerate(row_data):
                entry = ctk.CTkEntry(column, width=40,
textvariable=ctk.StringVar(value=cell_data),
                                    state="readonly")
                entry.grid(row=row, column=col, sticky="nsew")

        for col in range(len(headers)):
            column.grid_columnconfigure(col, weight=5)

        table.update_idletasks()
        resize_canvas(None, table)

    table_frame = ctk.CTkFrame(fn_frame, bg_color="Black")
    table_frame.pack(fill="both", expand=True, padx=10, pady=10)

    # Create canvas for the table
    table_canvas = ctk.CTkCanvas(table_frame)
    table_canvas.grid(row=0, column=0, sticky="nsew")

    # Horizontal scrollbar
    scrollbar_x = ctk.CTkScrollbar(table_frame, orientation="horizontal",
command=table_canvas.xview)
    scrollbar_x.grid(row=1, column=0, sticky="ew")
    table_canvas.configure(xscrollcommand=scrollbar_x.set)

    # Vertical scrollbar
    scrollbar_y = ctk.CTkScrollbar(table_frame, orientation="vertical",
command=table_canvas.yview)
    scrollbar_y.grid(row=0, column=1, sticky="ns", )
    table_canvas.configure(yscrollcommand=scrollbar_y.set)

    # Content inside canvas
    table_content = ctk.CTkFrame(table_canvas)
    table_canvas.create_window((0, 0), window=table_content,
anchor="nw")
    table_canvas.bind("<Configure>", lambda event: resize_canvas(event,
table_canvas))

    load_data(table_canvas, table_content)
```

```
def edit_student_details(frame):
    clear_frame(frame)
    root.title("Edit Student Data")
    def search_student(find_admission_no):
        global mydb
        admission_no = find_admission_no.get()
        if not admission_no.isdigit():
            tkmb.showerror("Error", "Please enter a valid Admission Number.")
            return

        try:
            mydb = db_connect()
            my_cursor = mydb.cursor()
            query = "SELECT * FROM students WHERE admn_no = %s"
            my_cursor.execute(query, (admission_no,))
            student = my_cursor.fetchone()

            if not student:
                tkmb.showinfo("Not Found", "Student Not Found.")
                return

            name_entry.delete(0, ctk.END)
            name_entry.insert(0, student[1])
            class_entry.set(student[2])
            age_entry.delete(0, ctk.END)
            age_entry.insert(0, student[3])
            father_name_entry.delete(0, ctk.END)
            father_name_entry.insert(0, student[4])
            gender_var.set(student[5])
            phone_entry.delete(0, ctk.END)
            phone_entry.insert(0, student[6])
            email_entry.delete(0, ctk.END)
            email_entry.insert(0, student[7])
            address_entry.delete(0, ctk.END)
            address_entry.insert(0, student[8])
        except Exception as e:
            tkmb.showerror("Error", f"An error occurred: {e}")
        finally:
            mydb.close()

    def update_student():
        global db
        admission_no = search_entry.get()
        updated_data = (
            name_entry.get(),
            class_entry.get(),
            age_entry.get(),
            father_name_entry.get(),
            gender_var.get(),
            phone_entry.get(),
            email_entry.get(),
            address_entry.get(),
            admission_no,
        )

        try:
            db = db_connect()
            my_cursor = db.cursor()
            query = """
            UPDATE students
            SET name = %s, class = %s, age = %s, father_name = %s, gender =
%s, ph_no = %s, email = %s, address = %s
            WHERE admn_no = %s
            """
            my_cursor.execute(query, updated_data)
            db.commit()
            tkmb.showinfo("Success", "Student details updated successfully.")
        except Exception as e:
            tkmb.showerror("Error", f"An error occurred: {e}")
        finally:
            db.close()

    main_frame = ctk.CTkFrame(frame, corner_radius=5, height=(s_height-
500), width=(s_width - s_width//8))
    main_frame.grid(row=0, column=0, padx=50, pady=50, ipadx=200,
ipady=50, sticky="ew")

    search_frame = ctk.CTkFrame(main_frame, corner_radius=5,
height=500, width=800)
```

```
    search_frame.grid(row=0, column=0, padx=20, pady=20, ipadx=20,
ipady=20, sticky="ew")

    search_label = ctk.CTkLabel(search_frame, text="Enter Admission
Number:", font=("Arial", 14))
    search_label.grid(row=0, column=0, padx=10, pady=10, sticky="w")

    search_entry = ctk.CTkEntry(search_frame, placeholder_text="Admission
Number")
    search_entry.grid(row=0, column=1, padx=10, pady=10, sticky="ew")

    search_btn = ctk.CTkButton(search_frame, text="Search",
command=lambda: search_student(search_entry))
    search_btn.grid(row=0, column=2, padx=10, pady=10)

    fields_frame = ctk.CTkFrame(search_frame, corner_radius=5)
    fields_frame.grid(row=1, column=0, columnspan=3, padx=10, pady=10,
sticky="ew")

    ctk.CTkLabel(fields_frame, text="Name:").grid(row=0, column=0,
padx=10, pady=5, sticky="w")
    name_entry = ctk.CTkEntry(fields_frame)
    name_entry.grid(row=0, column=1, padx=10, pady=5, sticky="ew")

    ctk.CTkLabel(fields_frame, text="Class:").grid(row=1, column=0,
padx=10, pady=5, sticky="w")
    class_entry = ctk.CTkOptionMenu(fields_frame,
                                values=["LKG", "UKG", "1", "2", "3", "4", "5", "6", "7", "8", "9",
"10", "11",
                                        "12"])
    class_entry.grid(row=1, column=1, padx=10, pady=5, sticky="ew")

    ctk.CTkLabel(fields_frame, text="Age:").grid(row=2, column=0, padx=10,
pady=5, sticky="w")
    age_entry = ctk.CTkEntry(fields_frame)
    age_entry.grid(row=2, column=1, padx=10, pady=5, sticky="ew")

    ctk.CTkLabel(fields_frame, text="Father's Name:").grid(row=3, column=0,
padx=10, pady=5, sticky="w")
    father_name_entry = ctk.CTkEntry(fields_frame)
    father_name_entry.grid(row=3, column=1, padx=10, pady=5,
sticky="ew")
```

```
    ctk.CTkLabel(fields_frame, text="Gender:").grid(row=4, column=0,
padx=10, pady=5, sticky="w")
    gender_var = tk.StringVar(value="Prefer not to say")
    ctk.CTkRadioButton(fields_frame, text="Male", variable=gender_var,
value="Male").grid(row=4, column=1, padx=10,
                                                   pady=5)
    ctk.CTkRadioButton(fields_frame, text="Female", variable=gender_var,
value="Female").grid(row=4, column=2, padx=10,
                                                     pady=5)

    ctk.CTkLabel(fields_frame, text="Phone Number:").grid(row=5,
column=0, padx=10, pady=5, sticky="w")
    phone_entry = ctk.CTkEntry(fields_frame)
    phone_entry.grid(row=5, column=1, padx=10, pady=5, sticky="ew")

    ctk.CTkLabel(fields_frame, text="Email:").grid(row=6, column=0,
padx=10, pady=5, sticky="w")
    email_entry = ctk.CTkEntry(fields_frame)
    email_entry.grid(row=6, column=1, padx=10, pady=5, sticky="ew")

    ctk.CTkLabel(fields_frame, text="Address:").grid(row=7, column=0,
padx=10, pady=5, sticky="w")
    address_entry = ctk.CTkEntry(fields_frame)
    address_entry.grid(row=7, column=1, padx=10, pady=5, sticky="ew")

    update_btn = ctk.CTkButton(fields_frame, text="Update",
command=update_student)
    update_btn.grid(row=8, column=0, columnspan=1, pady=10,
padx=10,sticky="ew")
```

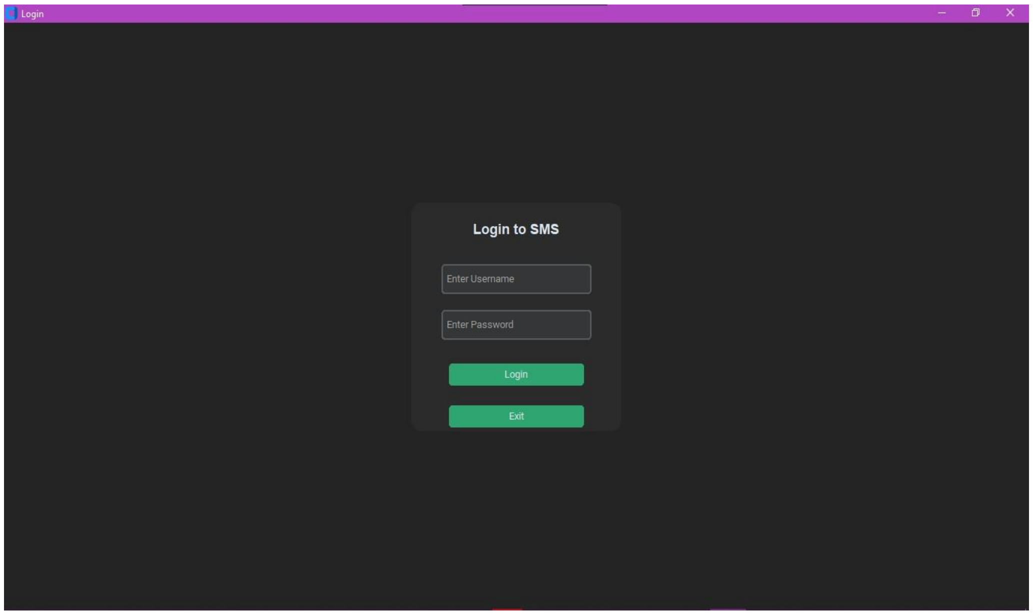
```
def exit_fn():
    if tkmb.askyesno(title="Quit SMS", message="Do you want to exit?"):
        root.destroy()
```

```
open_login_window()
root.mainloop()
```

OUTPUT

STEPS

A. Open the application to see the login interface.
B. Enter the username and password.
C. Validate credentials and confirm successful login.
D. Click "Add Student Details," input the required information, and submit.
E. Display a confirmation message for successful addition.
F. Click "Display Student Details" to view records.
G. Click "Edit Student Details," search for the student, update the information, and submit.
H. Display a confirmation message for successful updates.
I. Click "Delete Student Details," search for the student, and confirm deletion.
J. Display a confirmation message for successful deletion.
K. Click "Logout" to end the session.


Login

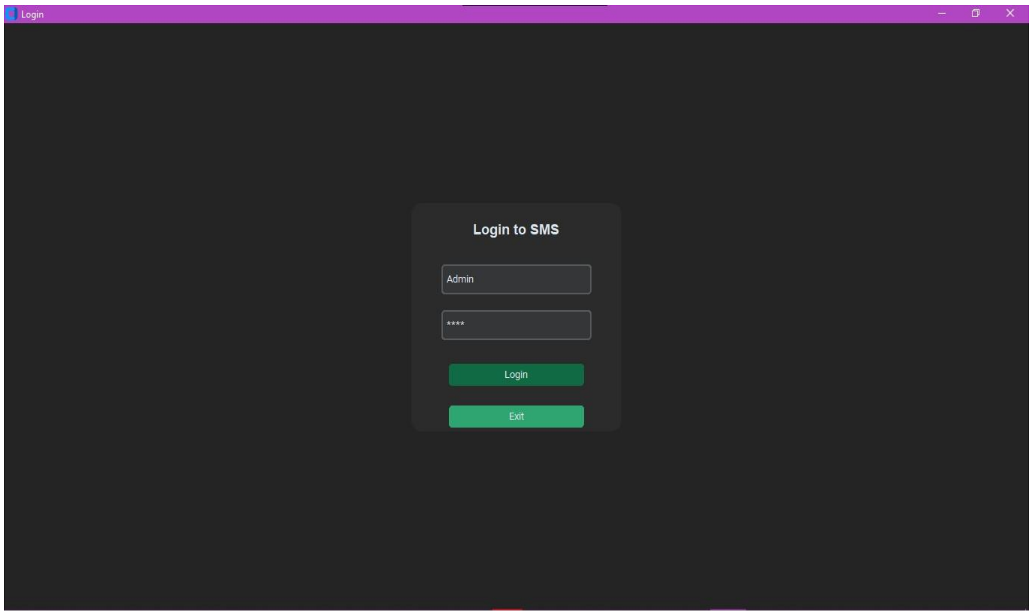
Login to SMS

Enter Username

Enter Password

Login

Exit


Login

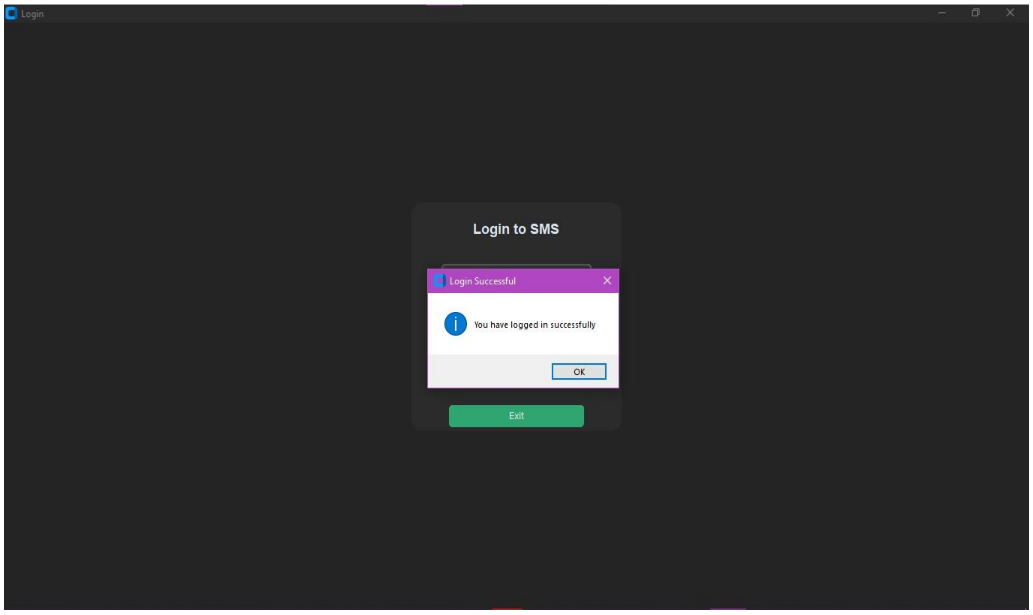
Login to SMS

Admin

****

Login

Exit


Login

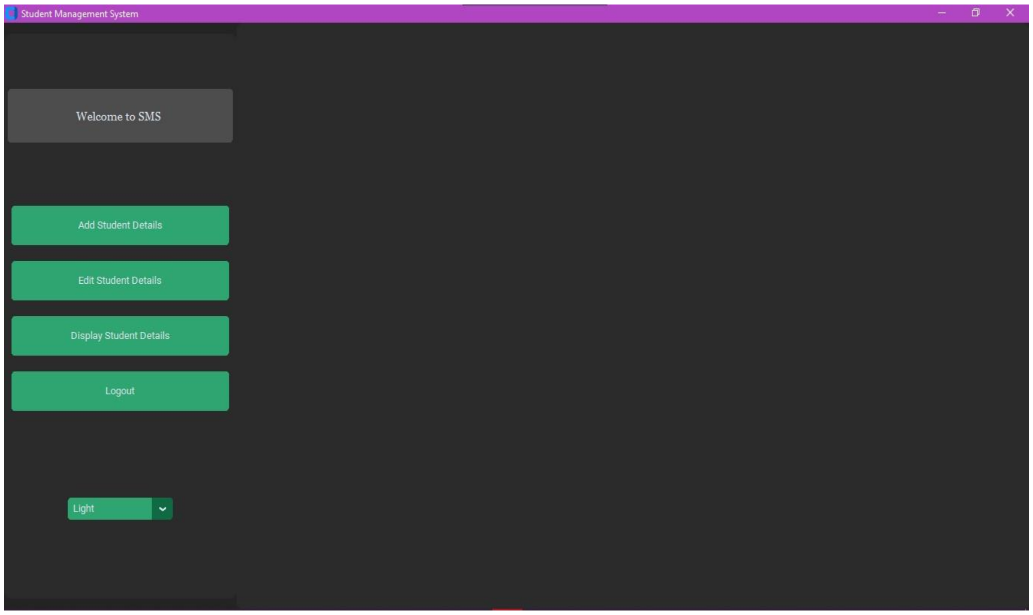
Login to SMS

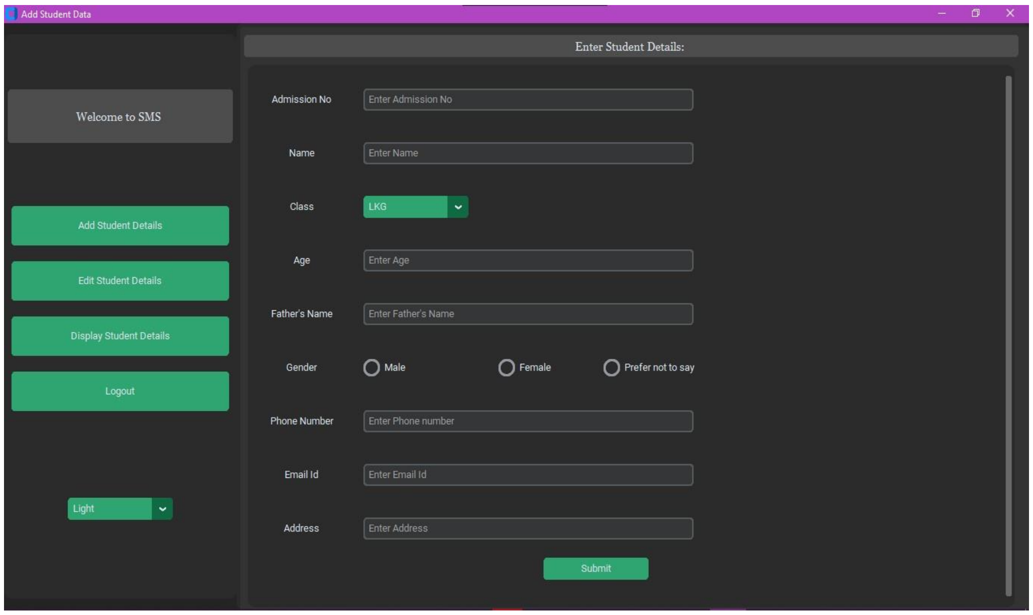
Login Successful

You have logged in successfully

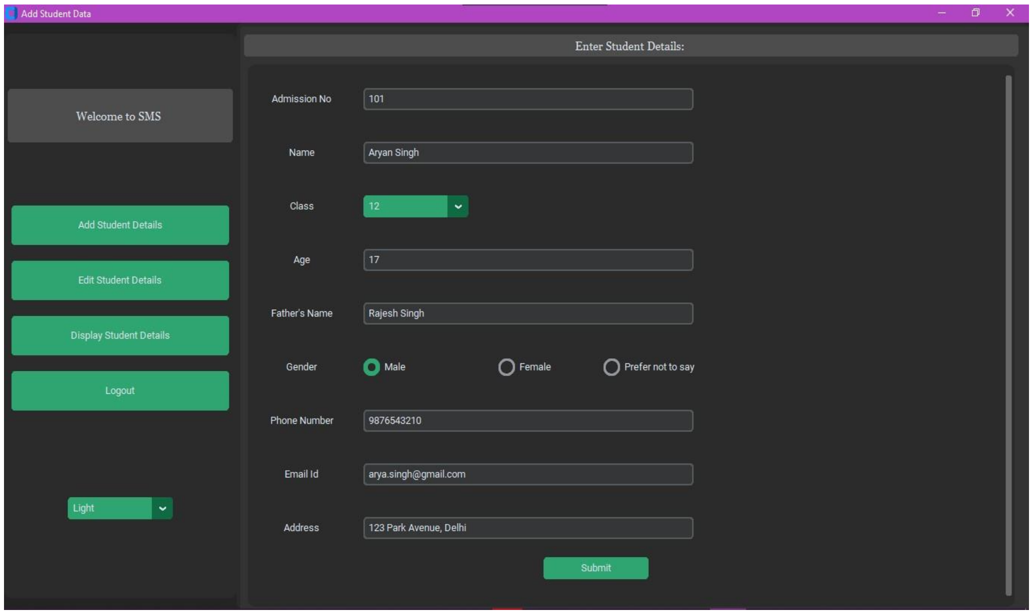
OK

Exit

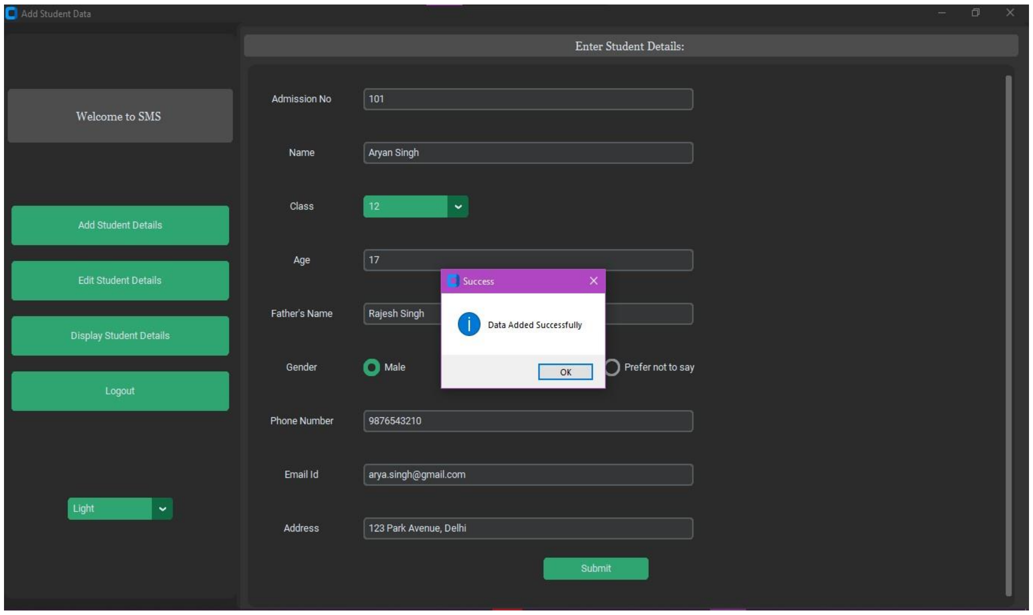




Admission No	Enter Admission No
Name	Enter Name
Class	LKG
Age	Enter Age
Father's Name	Enter Father's Name
Gender	Male Female Prefer not to say
Phone Number	Enter Phone number
Email Id	Enter Email Id
Address	Enter Address



Admission No	101
Name	Aryan Singh
Class	12
Age	17
Father's Name	Rajesh Singh
Gender	[✓ Male] Female Prefer not to say
Phone Number	9876543210
Email Id	arya.singh@gmail.com
Address	123 Park Avenue, Delhi

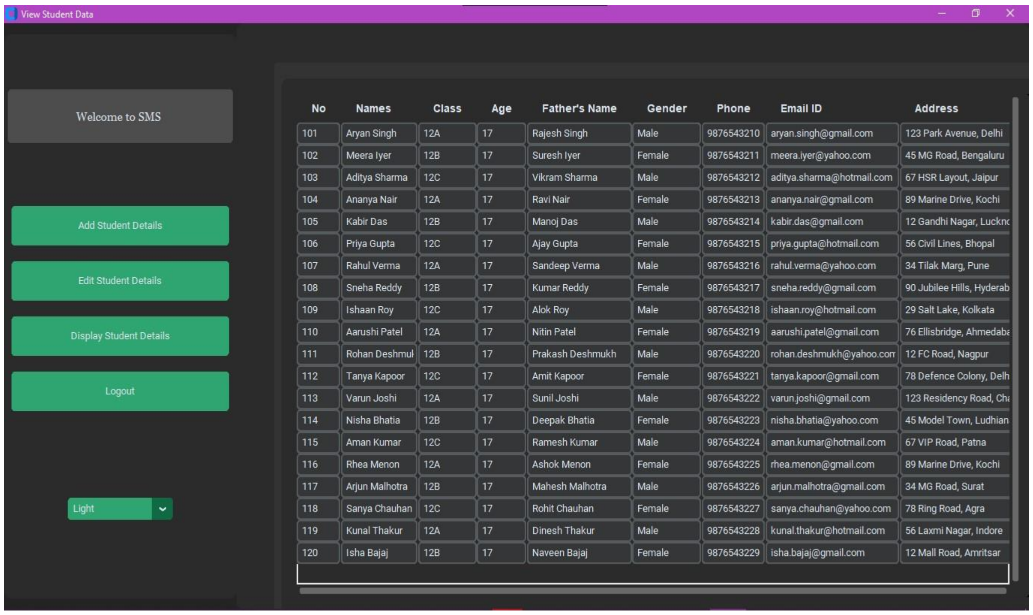


Admission No	101
Name	Aryan Singh
Class	12
Age	17
Father's Name	Rajesh Singh
Gender	[✓ Male] … Prefer not to say
Phone Number	9876543210
Email Id	arya.singh@gmail.com
Address	123 Park Avenue, Delhi

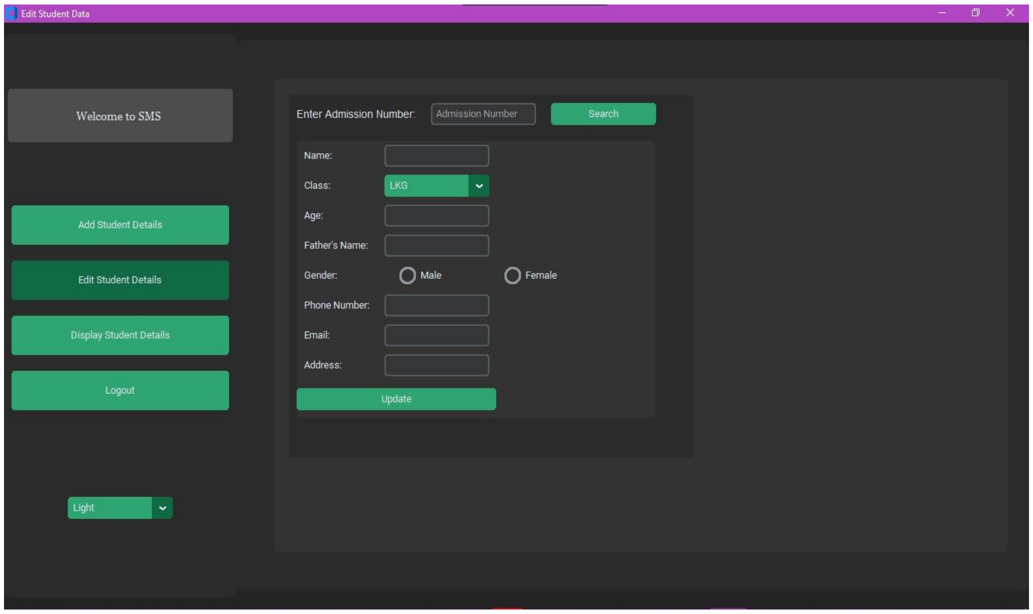
Success

Data Added Successfully

OK

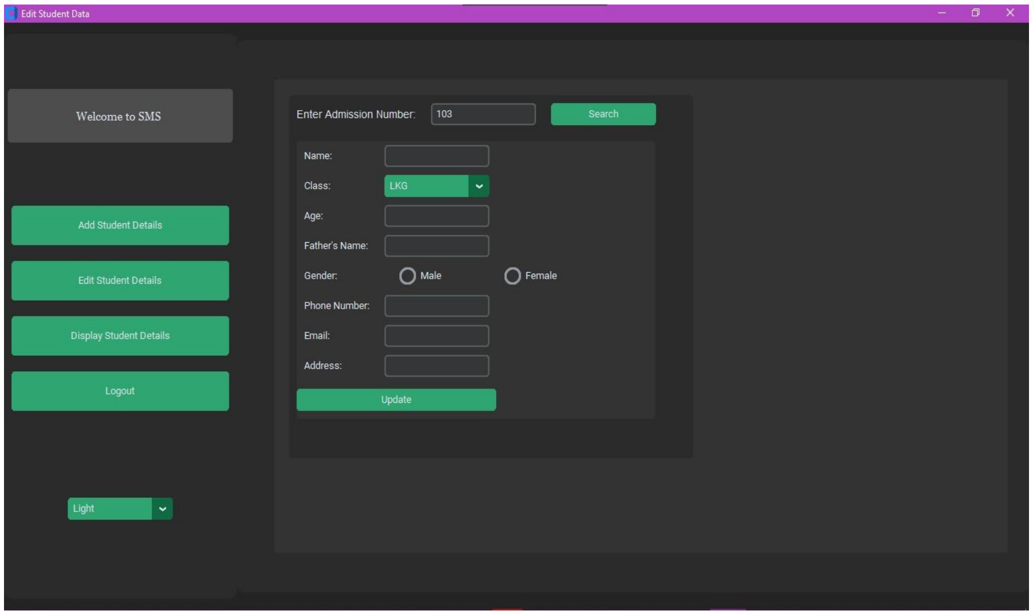


No	Names	Class	Age	Father's Name	Gender	Phone	Email ID	Address
101	Aryan Singh	12A	17	Rajesh Singh	Male	9876543210	aryan.singh@gmail.com	123 Park Avenue, Delhi
102	Meera Iyer	12B	17	Suresh Iyer	Female	9876543211	meera.iyer@yahoo.com	45 MG Road, Bengaluru
103	Aditya Sharma	12C	17	Vikram Sharma	Male	9876543212	aditya.sharma@hotmail.com	67 HSR Layout, Jaipur
104	Ananya Nair	12A	17	Ravi Nair	Female	9876543213	ananya.nair@gmail.com	89 Marine Drive, Kochi
105	Kabir Das	12B	17	Manoj Das	Male	9876543214	kabir.das@gmail.com	12 Gandhi Nagar, Luckno
106	Priya Gupta	12C	17	Ajay Gupta	Female	9876543215	priya.gupta@hotmail.com	56 Civil Lines, Bhopal
107	Rahul Verma	12A	17	Sandeep Verma	Male	9876543216	rahul.verma@yahoo.com	34 Tilak Marg, Pune
108	Sneha Reddy	12B	17	Kumar Reddy	Female	9876543217	sneha.reddy@gmail.com	90 Jubilee Hills, Hyderab
109	Ishaan Roy	12C	17	Alok Roy	Male	9876543218	ishaan.roy@hotmail.com	29 Salt Lake, Kolkata
110	Aarushi Patel	12A	17	Nitin Patel	Female	9876543219	aarushi.patel@gmail.com	76 Ellisbridge, Ahmedaba
111	Rohan Deshmul	12B	17	Prakash Deshmukh	Male	9876543220	rohan.deshmukh@yahoo.con	12 FC Road, Nagpur
112	Tanya Kapoor	12C	17	Amit Kapoor	Female	9876543221	tanya.kapoor@gmail.com	78 Defence Colony, Delh
113	Varun Joshi	12A	17	Sunil Joshi	Male	9876543222	varun.joshi@gmail.com	123 Residency Road, Cha
114	Nisha Bhatia	12B	17	Deepak Bhatia	Female	9876543223	nisha.bhatia@yahoo.com	45 Model Town, Ludhian
115	Aman Kumar	12C	17	Ramesh Kumar	Male	9876543224	aman.kumar@hotmail.com	67 VIP Road, Patna
116	Rhea Menon	12A	17	Ashok Menon	Female	9876543225	rhea.menon@gmail.com	89 Marine Drive, Kochi
117	Arjun Malhotra	12B	17	Mahesh Malhotra	Male	9876543226	arjun.malhotra@gmail.com	34 MG Road, Surat
118	Sanya Chauhan	12C	17	Rohit Chauhan	Female	9876543227	sanya.chauhan@yahoo.com	78 Ring Road, Agra
119	Kunal Thakur	12A	17	Dinesh Thakur	Male	9876543228	kunal.thakur@hotmail.com	56 Laxmi Nagar, Indore
120	Isha Bajaj	12B	17	Naveen Bajaj	Female	9876543229	isha.bajaj@gmail.com	12 Mall Road, Amritsar



Enter Admission Number: Admission Number

Name:	
Class:	LKG
Age:	
Father's Name:	
Gender:	Male Female
Phone Number:	
Email:	
Address:	



Enter Admission Number: 103

Name:

Class: LKG

Age:

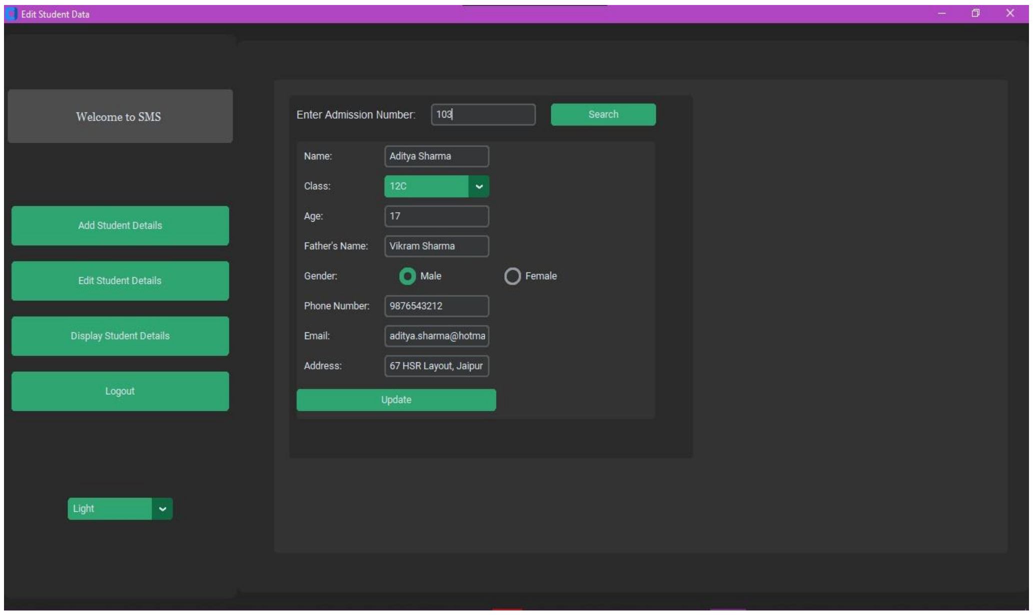
Father's Name:

Gender: ○ Male ○ Female

Phone Number:

Email:

Address:



Enter Admission Number: 103

Name: Aditya Sharma

Class: 12C

Age: 17

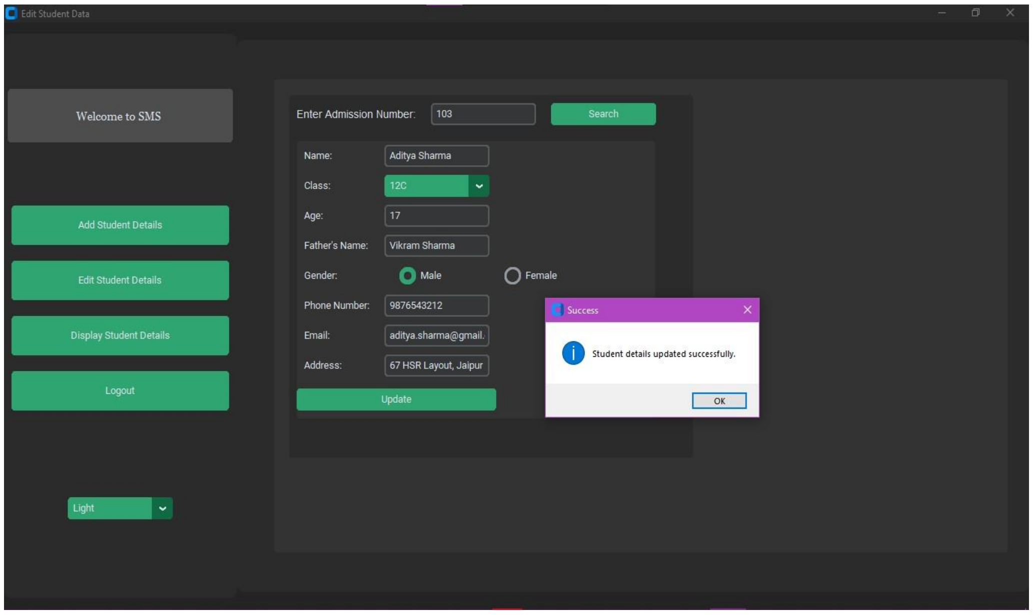
Father's Name: Vikram Sharma

Gender: [✓ Male] Female

Phone Number: 9876543212

Email: aditya.sharma@hotma

Address: 67 HSR Layout, Jaipur



Enter Admission Number: 103

Name: Aditya Sharma

Class: 12C

Age: 17

Father's Name: Vikram Sharma

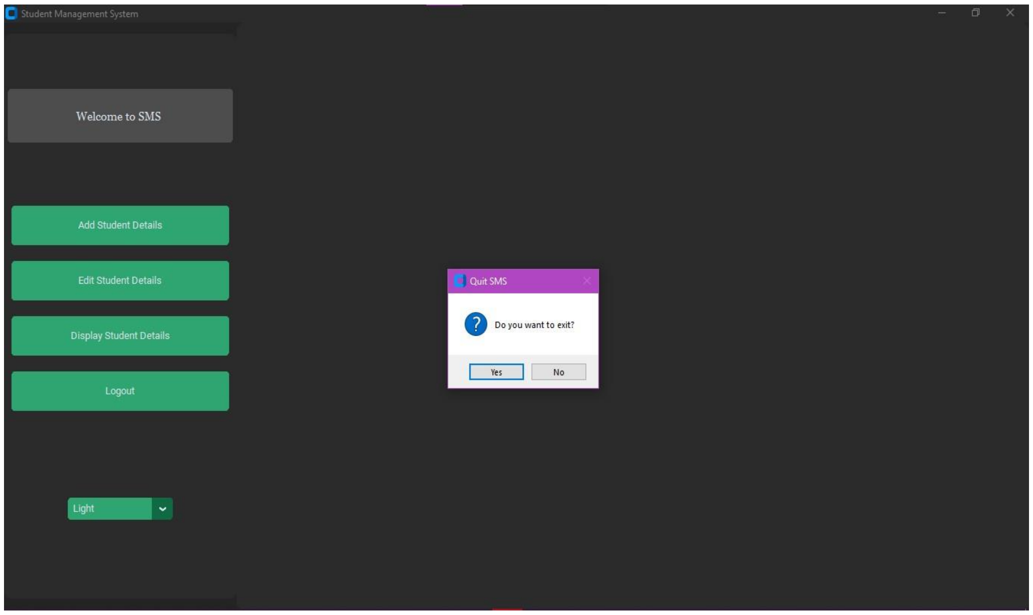
Gender: [✓ Male] Female

Phone Number: 9876543212

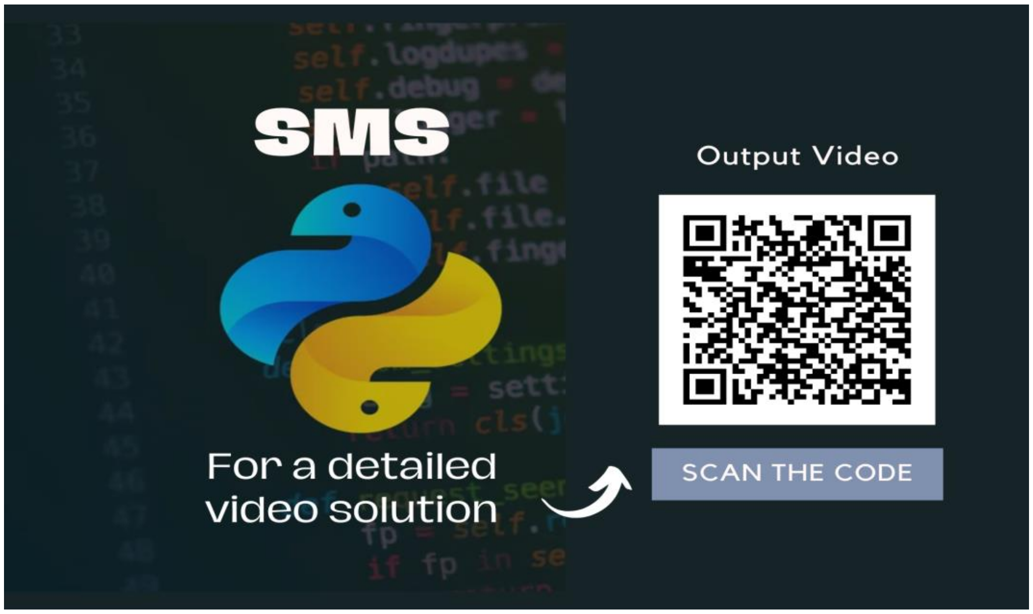
Email: aditya.sharma@gmail.

Address: 67 HSR Layout, Jaipur

Student details updated successfully.



Do you want to exit?



PROS & CONS

"The table below outlines the key pros and cons of the Student Management System (SMS), offering a comprehensive overview of its strengths and areas that may require further enhancement."

PROS

- ✓ *Simplifies student record management.*
- ✓ *User-friendly interface.*
- ✓ *Ensures accurate and consistent data.*
- ✓ *Enables quick retrieval and updates.*
- ✓ *Scalable for large student datasets.*

CONS

- × *Needs Python and MySQL setup.*
- × *Limited to local use without server hosting.*
- × *Error handling can be improved.*

CONCLUSION

STUDENT MANAGEMENT SYSTEM

In summary, the Student Management System (SMS) marks a significant step forward in managing student data for educational institutions.

By offering a robust and efficient platform, the SMS digitizes records, reduces manual workloads, and streamlines administrative processes. Its intuitive design ensures easy adoption, enabling staff to focus on core educational tasks while minimizing errors.

Future enhancements, such as web-based interfaces and advanced reporting and analytics capabilities, would further enhance accessibility and provide valuable insights. These improvements would empower institutions to make data-driven decisions, boosting overall performance and effectiveness.

LIST OF REFERENCES

01 PYTHON DOCUMENTATION

02 MYSQL DOCUMENTATION

03 GEEKSFORGEEKS

04 CUSTOMTKINTER

Aadhil Nandan

Check out my Github for code ⟶