PROJECT REPORT

Registration Form for Symposium:Python

A Mini Project submitted in partial Fulfilment for the Award of

MASTER OF SCIENCE IN DATA SCIENCE

Submitted by

PAVITHIRAN V

Register number:235229122

Under the guidance of

G.Rajalakshmi, B.sc., M.sc.,

Assistant Professor

DEPARTMENT OF DATA SCIENCE

BISHOP HEBER COLLEGE(AUTONOMOUS)

(Nationally Reaccredited at the 'A' Grade by NAAC with the CGPA of 3.69 out of 4)
(Recognized by UGC as "college of Excellence")(Affiliated to Bharathidasan University)

TIRUCHIRAPALLI-620017

OCTOBER-2023

DECLARATION

I here by declare that the project work presented is originally done by me under the guidance of **G.Rajalakshmi, B.sc., M.sc., Assistant Professor, Department of Data Science , Bishop Heber College (Autonomous), Tiruchirapalli-620 017** and has not been included in any other thesis/project submitted for any other degree

Name of the candidate : PAVITHIRAN

Register Number : 235229128

Batch : 2023-2025

Signature of the Candidate

DEPARTMENT OF DATA SCIENCE
BISHOP HEBER COLLEGE (AUTONOMOUS)

(Nationally Reaccredited at the 'A' Grade by NAAC with the CGPA of 3.58 out of 4)
(Recognized by UGC as "College with Potential for Excellence")
(Affiliated to Bharathidasan University)

TIRUCHIRAPPALLI 620017

Date:

Course Title: Programming and Data Structure with Python

Course Code: P23DS101

BONAFIDE CERTIFICATE

This is to certify that the project work titled “TITLE” is a bonafide record of the project work done by Name of the student, Register Number, in partial fulfillment of the requirements for the award of the degree of MASTER OF SCIENCE IN DATA SCIENCE during the period 2021-2023.

The Viva-Voce examination for the candidate Name of the Student, Register Number, was held on _____.

Signature of the HOD **Signature of the Guide**

Examiners:

1.

2.

Acknowledgement

The satisfaction that accompanies the successful completion of this project would be in complete without the mention of the people who made it possible, without whose constant guidance and encouragement would have made efforts go in vain. I consider myself privileged to express gratitude and respect towards all those who guided us through the completion of this project

I convey thanks to my project guide **G.Rajalakshmi Assistant professor Department of Data Science** for providing encouragement, constant support and guidance which was of a great help to complete this project successfully.

Last but not the least, we wish to thank our **parents** for financing our studies in this college as well as for constantly encouraging us to learn Datascience. Their personal sacrifice in providing this opportunity to learn engineering is gratefully acknowledged

Abstract:

The "Symposium Application Form Management System" is a Python-based project designed to automate and streamline the application submission and management process for symposium organizers. This project leverages a user-friendly GUI (Graphical User Interface) and integrates with a SQL (Structured Query Language) database to enhance efficiency, reduce errors, and centralize applicant data. It aims to improve the overall experience for applicants and provide valuable insights through data analysis.

Registration Form for Symposium

Introduction:

This Python script is designed to create a graphical user interface (GUI) application for a symposium registration form. It uses the Tkinter library for building the GUI, SQLite for database management, and the Pillow (PIL) library for image processing. The registration form collects information from participants, including their personal details, college information, team leader details, and event preferences.

Dependencies:

- Tkinter: The standard Python interface to the Tk GUI toolkit.
- Pillow (PIL): A Python Imaging Library to work with images.
- sqlite3: A built-in library for SQLite database management.

Database:

The application uses an SQLite database to store participant information. It creates several tables to store data, including college_info, event1_data, event2_data, event3_data, and event4_data. Additionally, there's a user_data table for storing login credentials. Database connections are managed throughout the script.

GUI Pages and Functions:

The application consists of several pages, each with its own set of functionalities. Below is a brief overview of the pages and their associated functions:

1. Front Page (front_page):
   - Displays a background image.
   - Provides an option to register for the symposium.

2. Login Page (login_page):
   - Allows users to log in with their phone number and password.
   - Provides the option to sign up for new users.
   - Validates user credentials and grants access to the next page.

3. Registration Page (register_page):
   - Allows users to sign up with a phone number and password.
   - Checks for existing phone numbers to prevent duplicates.
   - Stores new user data in the user_data table.

4. Rules Page (rules_page):
   - Displays the rules and guidelines of the symposium.
   - Provides options to navigate to the next page or sign out.

5. Student Coordinator Page (stud_coord_page):
   - Collects college information, including college name, department, shift, address, team leader name, team leader phone number, and the number of participants.
   - Allows users to scroll if there are multiple participants.
   - Data is stored in the college_info table.

6. Event Information Page (event_info_page):
   - Collects information about participants' event preferences.
   - Includes four events with details for two participants per event.
   - Data is stored in event1_data, event2_data, event3_data, and event4_data tables.

7. Payment Page (payment_page):
   - Displays payment-related information (not fully implemented in the provided code).

Database Integration:

- The application creates a view called integrated_data that combines data from different tables to display a comprehensive view of participant information.
- It also allows viewing the integrated data in a Treeview widget.

Navigation:

- Navigation between pages is managed using functions like switch_to_login_page(), switch_to_register_page(), etc.
- Users can navigate between pages using "Back" and "Next" buttons.

Image Handling:

- The application allows users to upload and store an image.

User Authentication:

- Users can log in with their phone number and password.
- Admin login is also available using the username "admin" and a password.
- Passwords are stored securely in the database.

Data Insertion:

- User-provided data is inserted into the appropriate database tables.

Resetting Entry Fields:

- A function reset_entry_fields() is available to clear input fields.

Sign Out:

- Users can sign out of their accounts.

Final Page:

- The final page displays an integrated view of participant data stored in the database.

Limitations:

- Some features, such as payment processing, are not fully implemented in the provided code.
- Proper error handling and validation for user inputs should be added.

Conclusion:

This Python script creates a symposium registration form with a user-friendly GUI. Users can log in, register, and provide their information, including college details and event preferences. The data is stored in an SQLite database and can be viewed in an integrated format on the final page.

Please note that this documentation provides an overview of the script's structure and functionality. Additional details, such as widget placements and visual elements, can be found by reviewing the code itself.

Source code:

```
import os
import shutil
import tkinter as tk
from tkinter import ttk, filedialog, messagebox
from PIL import Image, ImageTk
import sqlite3

conn = sqlite3.connect('registration_form.db')
cursor = conn.cursor()
conn3 = sqlite3.connect("user_data.db")
cursor3 = conn3.cursor()

cursor3.execute('''CREATE TABLE IF NOT EXISTS user_data (
    phone_number TEXT PRIMARY KEY,
    password TEXT
)''')
conn3.commit()
```

```
conn3.close()

cursor.execute('''CREATE TABLE IF NOT EXISTS college_info (
   text_box_ph_no_pg2 TEXT PRIMARY KEY,
   clg_entry TEXT,
   dept_entry TEXT,
   shift_entry TEXT,
   clg_add_text TEXT,
   stud_coord_name_entry TEXT,
   stud_coord_ph_entry TEXT,
   selected_value TEXT)''')

cursor.execute('''CREATE TABLE IF NOT EXISTS event1_data (
   text_box_ph_no_pg2 TEXT PRIMARY KEY,
   event1_par1_name_entry TEXT,
   event1_par1_ph_entry TEXT,
   event1_par2_name_entry TEXT,
   event1_par2_ph_entry TEXT)''')

cursor.execute('''CREATE TABLE IF NOT EXISTS event2_data (
   text_box_ph_no_pg2 TEXT PRIMARY KEY,
   event2_par1_name_entry TEXT,
   event2_par1_ph_entry TEXT,
   event2_par2_name_entry TEXT,
   event2_par2_ph_entry TEXT
)''')

cursor.execute('''CREATE TABLE IF NOT EXISTS event3_data (
   text_box_ph_no_pg2 TEXT PRIMARY KEY,
   event3_par1_name_entry TEXT,
   event3_par1_ph_entry TEXT,
   event3_par2_name_entry TEXT,
   event3_par2_ph_entry TEXT
)''')

cursor.execute('''CREATE TABLE IF NOT EXISTS event4_data (
   text_box_ph_no_pg2 TEXT PRIMARY KEY,
   event4_par1_name_entry TEXT,
   event4_par1_ph_entry TEXT,
   event4_par2_name_entry TEXT,
   event4_par2_ph_entry TEXT
)''')


conn.commit()
conn.close()
```

```
def switch_to_login_page():
    front_page.pack_forget()
    register_page.pack_forget()
    rules_page.pack_forget()
    stud_coord_page.pack_forget()
    event_info_page.pack_forget()
    payment_page.pack_forget()
    login_page.pack(fill="both", expand=True)


def switch_to_front_page():
    login_page.pack_forget()
    register_page.pack_forget()
    rules_page.pack_forget()
    stud_coord_page.pack_forget()
    event_info_page.pack_forget()
    payment_page.pack_forget()
    front_page.pack(fill="both", expand=True)


def switch_to_register_page():
    front_page.pack_forget()
    login_page.pack_forget()
    rules_page.pack_forget()
    stud_coord_page.pack_forget()
    event_info_page.pack_forget()
    payment_page.pack_forget()
    register_page.pack(fill="both", expand=True)


def switch_to_rules_page():
    front_page.pack_forget()
    login_page.pack_forget()
    register_page.pack_forget()
    stud_coord_page.pack_forget()
    event_info_page.pack_forget()
    payment_page.pack_forget()
    rules_page.pack(fill="both", expand=True)


def switch_to_stud_coord_page():
    front_page.pack_forget()
    login_page.pack_forget()
    register_page.pack_forget()
    rules_page.pack_forget()
    event_info_page.pack_forget()
    payment_page.pack_forget()
    stud_coord_page.pack(fill="both", expand=True)
```

```
def switch_to_event_info_page():
    front_page.pack_forget()
    login_page.pack_forget()
    register_page.pack_forget()
    rules_page.pack_forget()
    stud_coord_page.pack_forget()
    payment_page.pack_forget()
    event_info_page.pack(fill="both", expand=True)


def switch_to_payment_page():
    front_page.pack_forget()
    login_page.pack_forget()
    register_page.pack_forget()
    rules_page.pack_forget()
    stud_coord_page.pack_forget()
    event_info_page.pack_forget()
    payment_page.pack(fill="both", expand=True)


def on_select(event):
    selected_option = selected_value.get()
    print(f"Selected option: {selected_option}")


def upload_and_store_image():
    file_path = filedialog.askopenfilename(filetypes=[("Image Files", "*.png *.jpg *.jpeg *.gif
*.bmp")])
    if file_path:
        filename = os.path.basename(file_path)
        destination_path = os.path.join(os.getcwd(), filename)
        try:
            shutil.copy(file_path, destination_path)
            status_label.config(text=f"Image '{filename}' saved in project files.")
        except Exception as e:
            status_label.config(text=f"Error: {str(e)}")


def insert_data():
    phone_number = text_box_ph_no_pg3.get()
    password = text_box_password_pg3.get()

    if not phone_number or not password:
        status_label.config(text="Please provide both phone number and password.",
foreground="red")
        return
```

```
    try:
        conn1 = sqlite3.connect("user_data.db")
        cursor1 = conn1.cursor()


        cursor1.execute("SELECT * FROM user_data WHERE phone_number=?", (phone_number,))
        existing_record = cursor1.fetchone()

        if existing_record:
            messagebox.showwarning("Warning", "Phone number already exists.")
            text_box_ph_no_pg3.delete(0, tk.END)
            text_box_password_pg3.delete(0, tk.END)
            return

        cursor1.execute("INSERT INTO user_data (phone_number, password) VALUES (?, ?)",
(phone_number, password))
        conn1.commit()
        conn1.close()
        text_box_ph_no_pg3.delete(0, tk.END)
        text_box_password_pg3.delete(0, tk.END)
        status_label.config(text="Data inserted successfully!", foreground="orange")
        messagebox.showinfo("Success", "Registration successful!")
        switch_to_login_page()

    except sqlite3.Error as e:
        status_label.config(text=f"Database error: {str(e)}", foreground="red")


def login_admin_check():
    if text_box_ph_no_pg2.get() == 'admin' and text_box_password_pg2.get():
        final_page()
        pass
    else:
        check_credentials()


def check_credentials():
    phone_number = text_box_ph_no_pg2.get()
    password = text_box_password_pg2.get()
    if not phone_number or not password:
        messagebox.showwarning("Warning", "Please provide both phone number and password.")
        return
    try:
        conn2 = sqlite3.connect("user_data.db")
        cursor2 = conn2.cursor()
        cursor2.execute("SELECT * FROM user_data WHERE phone_number=?", (phone_number,))
        existing_record = cursor2.fetchone()
```

```
        if existing_record:
            if password == existing_record[1]:
                messagebox.showinfo("Success", "Login successful!")
                switch_to_rules_page()
            else:
                messagebox.showerror("Error", "Incorrect password.")
        else:
            messagebox.showerror("Error", "Phone number not found.")
        conn2.close()
    except sqlite3.Error as e:
        messagebox.showerror("Error", f"Database error: {str(e)}")


def reset_entry_fields():
    text_box_ph_no_pg3.delete(0, tk.END)
    text_box_ph_no_pg2.delete(0, tk.END)
    text_box_password_pg2.delete(0, tk.END)
    text_box_password_pg3.delete(0, tk.END)
    clg_entry.delete(0, tk.END)
    dept_entry.delete(0, tk.END)
    shift_entry.delete(0, tk.END)
    clg_add_text.delete("1.0", tk.END)
    stud_coord_name_entry.delete(0, tk.END)
    stud_coord_ph_entry.delete(0, tk.END)
    selected_value.set(values[0])
    event1_par1_name_entry.delete(0, tk.END)
    event1_par1_ph_entry.delete(0, tk.END)
    event1_par2_name_entry.delete(0, tk.END)
    event1_par2_ph_entry.delete(0, tk.END)
    event2_par1_name_entry.delete(0, tk.END)
    event2_par1_ph_entry.delete(0, tk.END)
    event2_par2_name_entry.delete(0, tk.END)
    event2_par2_ph_entry.delete(0, tk.END)
    event3_par1_name_entry.delete(0, tk.END)
    event3_par1_ph_entry.delete(0, tk.END)
    event3_par2_name_entry.delete(0, tk.END)
    event3_par2_ph_entry.delete(0, tk.END)
    event4_par1_name_entry.delete(0, tk.END)
    event4_par1_ph_entry.delete(0, tk.END)
    event4_par2_name_entry.delete(0, tk.END)
    event4_par2_ph_entry.delete(0, tk.END)


def submit_function():
    submit_button_event()
    switch_to_login_page()
    reset_entry_fields()
    pass
```

```
def sign_out_function():
    switch_to_login_page()
    text_box_ph_no_pg2.delete(0, tk.END)
    text_box_password_pg2.delete(0, tk.END)


def submit_button_event():
    print(text_box_ph_no_pg2.get(),
          clg_entry.get(),
          dept_entry.get(),
          shift_entry.get(),
          clg_add_text.get("1.0", "end-1c"),
          stud_coord_name_entry.get(),
          stud_coord_ph_entry.get(),
          selected_value.get())
    print(text_box_ph_no_pg3.get(),
          event1_par1_name_entry.get(),
          event1_par1_ph_entry.get(),
          event1_par2_name_entry.get(),
          event1_par2_ph_entry.get())
    print(text_box_ph_no_pg3.get(),
          event2_par1_name_entry.get(),
          event2_par1_ph_entry.get(),
          event2_par2_name_entry.get(),
          event2_par2_ph_entry.get())
    print(text_box_ph_no_pg3.get(),
          event4_par1_name_entry.get(),
          event4_par1_ph_entry.get(),
          event4_par2_name_entry.get(),
          event4_par2_ph_entry.get())
    conn2 = sqlite3.connect('registration_form.db')
    cursor2 = conn2.cursor()
    cursor2.execute('''INSERT INTO college_info (
                        text_box_ph_no_pg2,
                        clg_entry,
                        dept_entry,
                        shift_entry,
                        clg_add_text,
                        stud_coord_name_entry,
                        stud_coord_ph_entry,
                        selected_value)VALUES (?, ?, ?, ?, ?, ?, ?, ?)''',
                    (text_box_ph_no_pg2.get(),
                     clg_entry.get(),
                     dept_entry.get(),
                     shift_entry.get(),
                     clg_add_text.get("1.0", "end-1c"),
```

```
                stud_coord_name_entry.get(),
                stud_coord_ph_entry.get(),
                selected_value.get()))

cursor2.execute('''INSERT INTO event1_data (
                    text_box_ph_no_pg2,
                    event1_par1_name_entry,
                    event1_par1_ph_entry,
                    event1_par2_name_entry,
                    event1_par2_ph_entry)VALUES (?, ?, ?, ?, ?)''',
                (text_box_ph_no_pg2.get(),
                event1_par1_name_entry.get(),
                event1_par1_ph_entry.get(),
                event1_par2_name_entry.get(),
                event1_par2_ph_entry.get()))

cursor2.execute('''INSERT INTO event2_data (
                    text_box_ph_no_pg2,
                    event2_par1_name_entry,
                    event2_par1_ph_entry,
                    event2_par2_name_entry,
                    event2_par2_ph_entry)VALUES (?, ?, ?, ?, ?)''',
                (text_box_ph_no_pg2.get(),
                event2_par1_name_entry.get(),
                event2_par1_ph_entry.get(),
                event2_par2_name_entry.get(),
                event2_par2_ph_entry.get()))

cursor2.execute('''INSERT INTO event3_data (
                    text_box_ph_no_pg2,
                    event3_par1_name_entry,
                    event3_par1_ph_entry,
                    event3_par2_name_entry,
                    event3_par2_ph_entry)VALUES (?, ?, ?, ?, ?)''',
                (text_box_ph_no_pg2.get(),
                event3_par1_name_entry.get(),
                event3_par1_ph_entry.get(),
                event3_par2_name_entry.get(),
                event3_par2_ph_entry.get()))

cursor2.execute('''INSERT INTO event4_data (
                    text_box_ph_no_pg2,
                    event4_par1_name_entry,
                    event4_par1_ph_entry,
                    event4_par2_name_entry,
                    event4_par2_ph_entry)VALUES (?, ?, ?, ?, ?)''',
                (text_box_ph_no_pg2.get(),
                event4_par1_name_entry.get(),
```

```
                event4_par1_ph_entry.get(),
                event4_par2_name_entry.get(),
                event4_par2_ph_entry.get()))
    messagebox.showinfo("Success", "Data inserted successfully!")
    conn2.commit()
    conn2.close()
    print("value saved")


def final_page():

    root = tk.Tk()
    root.title("SQLite Table Viewer")
    conn4 = sqlite3.connect('registration_form.db')
    cursor4 = conn4.cursor()
    view_sql = """
        CREATE VIEW IF NOT EXISTS integrated_data AS
        SELECT
        ci.text_box_ph_no_pg2 as student_leader_number,
        ci.clg_entry as college_name,
        ci.dept_entry as department_name,
        ci.shift_entry as shift,
        ci.clg_add_text as address,
        ci.stud_coord_name_entry as team_leader_name,
        ci.stud_coord_ph_entry as team_leader_ph,
        ci.selected_value as no_of_participant,
        e1.event1_par1_name_entry as event1_participant1_name,
        e1.event1_par1_ph_entry as event1_participant1_ph,
        e1.event1_par2_name_entry as event1_participant2_name,
        e1.event1_par2_ph_entry as event1_participant2_ph,
        e2.event2_par1_name_entry as event2_participant1_name,
        e2.event2_par1_ph_entry as event2_participant1_ph,
        e2.event2_par2_name_entry as event2_participant2_name,
        e2.event2_par2_ph_entry as event2_participant2_ph,
        e3.event3_par1_name_entry as event3_participant1_name,
        e3.event3_par1_ph_entry as event3_participant1_ph,
        e3.event3_par2_name_entry as event3_participant2_name,
        e3.event3_par2_ph_entry as event3_participant2_ph,
        e4.event4_par1_name_entry as event4_participant1_name,
        e4.event4_par1_ph_entry as event4_participant1_ph,
        e4.event4_par2_name_entry as event4_participant2_name,
        e4.event4_par2_ph_entry as event4_participant2_ph
        FROM college_info ci
        LEFT JOIN event1_data e1 ON ci.text_box_ph_no_pg2 = e1.text_box_ph_no_pg2
        LEFT JOIN event2_data e2 ON ci.text_box_ph_no_pg2 = e2.text_box_ph_no_pg2
        LEFT JOIN event3_data e3 ON ci.text_box_ph_no_pg2 = e3.text_box_ph_no_pg2
        LEFT JOIN event4_data e4 ON ci.text_box_ph_no_pg2 = e4.text_box_ph_no_pg2;
        """
```

```
    cursor4.execute(view_sql)

    query = "SELECT * FROM integrated_data"
    cursor4.execute(query)
    data = cursor4.fetchall()
    columns = [desc[0] for desc in cursor4.description]
    tree = ttk.Treeview(root, columns=columns)
    column_alignments = {
        col: "center"  # Default to center alignment for all columns
        for col in columns
    }

    for col in columns:
        heading_text = col.replace("_", " ").title()
        alignment = column_alignments.get(col, "center")
        tree.heading(col, text=heading_text, anchor=alignment)

    for col in columns:
        tree.column(col, width=100, anchor=column_alignments.get(col, "center"))

    vsb = ttk.Scrollbar(root, orient="vertical", command=tree.yview)
    tree.configure(yscrollcommand=vsb.set)
    vsb.pack(side="right", fill="y")
    hsb = ttk.Scrollbar(root, orient="horizontal", command=tree.xview)
    tree.configure(xscrollcommand=hsb.set)
    hsb.pack(side="bottom", fill="x")

    for row in data:
        tree.insert("", "end", values=row)
    tree.pack(fill="both", expand=True)

    conn4.close()
    root.geometry("800x600")
    root.resizable(False, False)
    root.mainloop()


def on_treeview_scroll(*args):
    tree.yview(*args)


def on_treeview_hscroll(*args):
    tree.xview(*args)


window = tk.Tk()
window.title("Registration Form For Symposium")
front_page = tk.Frame(window, width=200, height=100, bg="purple")
```

```
login_page = tk.Frame(window, width=200, height=100, bg="green")
register_page = tk.Frame(window, width=300, height=200, bg="blue")
rules_page = tk.Frame(window, width=200, height=100, bg="red")
stud_coord_page = tk.Frame(window, width=300, height=200, bg="blue")
event_info_page = tk.Frame(window, width=200, height=100, bg="red")
payment_page = tk.Frame(window, width=300, height=200, bg="blue")
front_page.pack(fill="both", expand=True)

# page1 front page
dummy_frame_page3 = tk.Frame(front_page, width=200, height=100, bg="green")
image_4 = Image.open("background_image1.jpeg")
image_tk_4 = ImageTk.PhotoImage(image_4)
imagelabel_4 = ttk.Label(dummy_frame_page3, image=image_tk_4)
imagelabel_4.pack()
dummy_frame_page3.pack()


page1_reg_now_button = tk.Button(front_page, text="Register Now",
command=switch_to_login_page)
page1_reg_now_button.place(x=325, y=440, width=100, height=40)


dummy_frame_page = tk.Frame(dummy_frame_page3, width=200, height=100, bg="green")
image_2 = Image.open("frontPage.jpg")
image_tk_2 = ImageTk.PhotoImage(image_2)
imagelabel_2 = ttk.Label(dummy_frame_page, image=image_tk_2)
imagelabel_2.pack()
dummy_frame_page.place(x=135, y=60, width=510, height=360)

# page2 login page

image_3 = Image.open("background_image1.jpeg")
image_tk_3 = ImageTk.PhotoImage(image_3)
imagelabel_3 = ttk.Label(login_page, image=image_tk_3)
imagelabel_3.pack()

sign_up_button_page = tk.Button(login_page, text="Sign Up",
command=switch_to_register_page)
sign_up_button_page.place(x=425, y=440, width=110, height=40)

login_up_button_page = tk.Button(login_page, text="Login", command=login_admin_check)
login_up_button_page.place(x=545, y=440, width=110, height=40)

page2_back_button = tk.Button(login_page, text="Back", command=switch_to_front_page)
page2_back_button.place(x=150, y=440, width=90, height=40)

ph_no_pg1_lbl_pg2 = tk.Label(login_page, text="Phone Number:")
ph_no_pg1_lbl_pg2.place(x=225, y=200, width=100, height=40)  # Add padding for spacing
```

```
text_box_ph_no_pg2 = tk.Entry(login_page)
text_box_ph_no_pg2.place(x=325, y=200, width=200, height=40)

password_pg1_lbl_pg2 = tk.Label(login_page, text="Password:")
password_pg1_lbl_pg2.place(x=225, y=300, width=100, height=40)  # Add padding for spacing

text_box_password_pg2 = tk.Entry(login_page)
text_box_password_pg2.place(x=325, y=300, width=200, height=40)



# page3

image_5 = Image.open("background_image1.jpeg")
image_tk_5 = ImageTk.PhotoImage(image_5)
imagelabel_5 = ttk.Label(register_page, image=image_tk_5)
imagelabel_5.pack()

submit_button_page3 = tk.Button(register_page, text="Submit", command=insert_data)
submit_button_page3.place(x=425, y=440, width=110, height=40)

ph_no_pg1_lbl_pg3 = tk.Label(register_page, text="Phone Number:")
ph_no_pg1_lbl_pg3.place(x=225, y=200, width=100, height=40)  # Add padding for spacing

text_box_ph_no_pg3 = tk.Entry(register_page)
text_box_ph_no_pg3.place(x=325, y=200, width=200, height=40)

password_pg1_lbl_pg3 = tk.Label(register_page, text="Password:")
password_pg1_lbl_pg3.place(x=225, y=300, width=100, height=40)  # Add padding for spacing

text_box_password_pg3 = tk.Entry(register_page)
text_box_password_pg3.place(x=325, y=300, width=200, height=40)

page3_back_button = tk.Button(register_page, text="Back", command=switch_to_login_page)
page3_back_button.place(x=150, y=440, width=90, height=40)

# page4


image_6 = Image.open("background_image1.jpeg")
image_tk_6 = ImageTk.PhotoImage(image_6)
imagelabel_6 = ttk.Label(rules_page, image=image_tk_6)
imagelabel_6.pack()


page4_back = tk.Button(rules_page, text="Back", command=switch_to_login_page)
page4_back.place(x=425, y=440, width=110, height=40)
```

```
page4_next = tk.Button(rules_page, text="Next", command=switch_to_stud_coord_page)
page4_next.place(x=545, y=440, width=110, height=40)

page4_sign_out = tk.Button(rules_page, text="Sign Out", command=sign_out_function)
page4_sign_out.place(x=545, y=40, width=110, height=40)


image_7 = Image.open("rules_page.jpg")
image_tk_7 = ImageTk.PhotoImage(image_7)
imagelabel_7 = ttk.Label(rules_page, image=image_tk_7)
imagelabel_7.place(x=90, y=30, width=310, height=540)


# page5


image_8 = Image.open("background_image1.jpeg")
image_tk_8 = ImageTk.PhotoImage(image_8)
imagelabel_8 = ttk.Label(stud_coord_page, image=image_tk_8)
imagelabel_8.pack()

page5_back = tk.Button(stud_coord_page, text="Back", command=switch_to_rules_page)
page5_back.place(x=425, y=440, width=110, height=40)

page5_next = tk.Button(stud_coord_page, text="Next", command=switch_to_event_info_page)
page5_next.place(x=545, y=440, width=110, height=40)

page5_sign_out = tk.Button(stud_coord_page, text="Sign Out", command=sign_out_function)
page5_sign_out.place(x=545, y=40, width=110, height=40)

canvas1 = tk.Canvas(stud_coord_page)
canvas1.place(x=100, y=125, width=550, height=265)

scrollbar = ttk.Scrollbar(canvas1, orient=tk.VERTICAL, command=canvas1.yview)
scrollbar.pack(side=tk.RIGHT, fill=tk.Y)
canvas1.configure(yscrollcommand=scrollbar.set)
frame = ttk.Frame(canvas1)
canvas1.create_window((0, 0), window=frame, anchor='nw')

clg_label = tk.Label(canvas1, text="College Name:")
clg_label.place(x=20, y=20)

clg_entry = tk.Entry(canvas1)
clg_entry.place(x=150, y=20, width=290)

dept_label = tk.Label(canvas1, text="Department Name:")
dept_label.place(x=20, y=45)
```

```
dept_entry = tk.Entry(canvas1)
dept_entry.place(x=150, y=45, width=290)

shift_lbl = tk.Label(canvas1, text="Shift:")
shift_lbl.place(x=20, y=70)

shift_entry = tk.Entry(canvas1)
shift_entry.place(x=150, y=70, width=290)

clg_add_label = tk.Label(canvas1, text="College Address")
clg_add_label.place(x=20, y=95)
clg_add_text = tk.Text(canvas1, wrap=tk.WORD, width=40, height=5)
clg_add_text.place(x=150, y=95)

stud_coord_name_lbl = tk.Label(canvas1, text="Team leader Name:")
stud_coord_name_lbl.place(x=20, y=170)

stud_coord_name_entry = tk.Entry(canvas1)
stud_coord_name_entry.place(x=150, y=170, width=290)

stud_coord_ph_lbl = tk.Label(canvas1, text="Team leader Contact:")
stud_coord_ph_lbl.place(x=20, y=195)

stud_coord_ph_entry = tk.Entry(canvas1)
stud_coord_ph_entry.place(x=150, y=195, width=290)

participants_count = tk.Label(canvas1, text="Participants Count:")
participants_count.place(x=20, y=220)
selected_value = tk.StringVar()
combo = ttk.Combobox(canvas1, textvariable=selected_value, state='readonly')
combo.place(x=150, y=220, width=290)
values = [str(i) for i in range(1, 9)]
combo['values'] = values
combo.set(values[0])

combo.bind("<<ComboboxSelected>>", on_select)

# page 6

image_9 = Image.open("background_image1.jpeg")
image_tk_9 = ImageTk.PhotoImage(image_9)
imagelabel_9 = ttk.Label(event_info_page, image=image_tk_9)
imagelabel_9.pack()

page6_back = tk.Button(event_info_page, text="Back", command=switch_to_stud_coord_page)
page6_back.place(x=425, y=440, width=110, height=40)
```

```
page6_next = tk.Button(event_info_page, text="Next", command=switch_to_payment_page)
page6_next.place(x=545, y=440, width=110, height=40)

page6_sign_out = tk.Button(event_info_page, text="Sign Out", command=sign_out_function)
page6_sign_out.place(x=545, y=40, width=110, height=40)

canvas2 = tk.Canvas(event_info_page)
canvas2.place(x=100, y=85, width=550, height=340)

scrollbar = ttk.Scrollbar(canvas2, orient=tk.VERTICAL, command=canvas2.yview)
scrollbar.pack(side=tk.RIGHT, fill=tk.Y)
canvas2.configure(yscrollcommand=scrollbar.set)
frame = ttk.Frame(canvas2)
canvas2.create_window((0, 0), window=frame, anchor='nw')

event1_lbl = tk.Label(canvas2, text="Event 1")
event1_lbl.place(x=20, y=20)

event1_par1_name_lbl = tk.Label(canvas2, text="Participant 1:")
event1_par1_name_lbl.place(x=20, y=45)

event1_par1_ph_lbl = tk.Label(canvas2, text="Ph.no:")
event1_par1_ph_lbl.place(x=300, y=45)

event1_par1_name_entry = tk.Entry(canvas2)
event1_par1_name_entry.place(x=110, y=45, width=180)

event1_par1_ph_entry = tk.Entry(canvas2)
event1_par1_ph_entry.place(x=350, y=45, width=180)

event1_par2_name_lbl = tk.Label(canvas2, text="Participant 2:")
event1_par2_name_lbl.place(x=20, y=70)

event1_par2_ph_lbl = tk.Label(canvas2, text="Ph.no:")
event1_par2_ph_lbl.place(x=300, y=70)

event1_par2_name_entry = tk.Entry(canvas2)
event1_par2_name_entry.place(x=110, y=70, width=180)

event1_par2_ph_entry = tk.Entry(canvas2)
event1_par2_ph_entry.place(x=350, y=70, width=180)

event2_lbl = tk.Label(canvas2, text="Event 2")
event2_lbl.place(x=20, y=95)

event2_par1_name_lbl = tk.Label(canvas2, text="Participant 1:")
event2_par1_name_lbl.place(x=20, y=120)
```

```
event2_par1_name_entry = tk.Entry(canvas2)
event2_par1_name_entry.place(x=110, y=120, width=180)

event2_par1_ph_lbl = tk.Label(canvas2, text="Ph.no:")
event2_par1_ph_lbl.place(x=300, y=120)

event2_par1_ph_entry = tk.Entry(canvas2)
event2_par1_ph_entry.place(x=350, y=120, width=180)

event2_par2_name_lbl = tk.Label(canvas2, text="Participant 2:")
event2_par2_name_lbl.place(x=20, y=145)

event2_par2_name_entry = tk.Entry(canvas2)
event2_par2_name_entry.place(x=110, y=145, width=180)

event2_par2_ph_lbl = tk.Label(canvas2, text="Ph.no:")
event2_par2_ph_lbl.place(x=300, y=145)

event2_par2_ph_entry = tk.Entry(canvas2)
event2_par2_ph_entry.place(x=350, y=145, width=180)

event3_lbl = tk.Label(canvas2, text="Event 3")
event3_lbl.place(x=20, y=170)

event3_par1_name_lbl = tk.Label(canvas2, text="Participant 1:")
event3_par1_name_lbl.place(x=20, y=195)

event3_par1_name_entry = tk.Entry(canvas2)
event3_par1_name_entry.place(x=110, y=195, width=180)

event3_par1_ph_lbl = tk.Label(canvas2, text="Ph.no:")
event3_par1_ph_lbl.place(x=300, y=195)

event3_par1_ph_entry = tk.Entry(canvas2)
event3_par1_ph_entry.place(x=350, y=195, width=180)

event3_par2_name_lbl = tk.Label(canvas2, text="Participant 2:")
event3_par2_name_lbl.place(x=20, y=220)

event3_par2_name_entry = tk.Entry(canvas2)
event3_par2_name_entry.place(x=110, y=220, width=180)

event3_par2_ph_lbl = tk.Label(canvas2, text="Ph.no:")
event3_par2_ph_lbl.place(x=300, y=220)

event3_par2_ph_entry = tk.Entry(canvas2)
event3_par2_ph_entry.place(x=350, y=220, width=180)
```

```
event4_lbl = tk.Label(canvas2, text="Event 4")
event4_lbl.place(x=20, y=245)

event4_par1_name_lbl = tk.Label(canvas2, text="Participant 1:")
event4_par1_name_lbl.place(x=20, y=270)

event4_par1_name_entry = tk.Entry(canvas2)
event4_par1_name_entry.place(x=110, y=270, width=180)

event4_par1_ph_lbl = tk.Label(canvas2, text="Ph.no:")
event4_par1_ph_lbl.place(x=300, y=270)

event4_par1_ph_entry = tk.Entry(canvas2)
event4_par1_ph_entry.place(x=350, y=270, width=180)

event4_par2_name_lbl = tk.Label(canvas2, text="Participant 2:")
event4_par2_name_lbl.place(x=20, y=295)

event4_par2_name_entry = tk.Entry(canvas2)
event4_par2_name_entry.place(x=110, y=295, width=180)

event4_par2_ph_lbl = tk.Label(canvas2, text="Ph.no:")
event4_par2_ph_lbl.place(x=300, y=295)

event4_par2_ph_entry = tk.Entry(canvas2)
event4_par2_ph_entry.place(x=350, y=295, width=180)

# page 7

image_10 = Image.open("background_image1.jpeg")
image_tk_10 = ImageTk.PhotoImage(image_10)
imagelabel_10 = ttk.Label(payment_page, image=image_tk_10)
imagelabel_10.pack()

page7_back = tk.Button(payment_page, text="Back", command=switch_to_event_info_page)
page7_back.place(x=150, y=440, width=90, height=40)

page7_label_payment = tk.Label(payment_page, text="Payment")
page7_label_payment.place(x=20, y=40)

page7_sign_out = tk.Button(payment_page, text="Sign Out", command=sign_out_function)
page7_sign_out.place(x=545, y=40, width=110, height=40)

image = Image.open("Qrcode.png")
image_tk = ImageTk.PhotoImage(image)
imagelabel = ttk.Label(payment_page, image=image_tk)
imagelabel.place(x=200, y=85, width=290, height=290)
```

```
upload_button = ttk.Button(payment_page, text="Upload File",
command=upload_and_store_image)
upload_button.place(x=350, y=440, width=125, height=40)

status_label = ttk.Label(frame, text="")
status_label.grid(row=1, column=0, padx=10, pady=5)

page7_submit = tk.Button(payment_page, text="Submit", command=submit_function)  #
page7_submit.place(x=545, y=440, width=110, height=40)

window.resizable(False, False)
window.geometry("800x600")
window.mainloop()
```

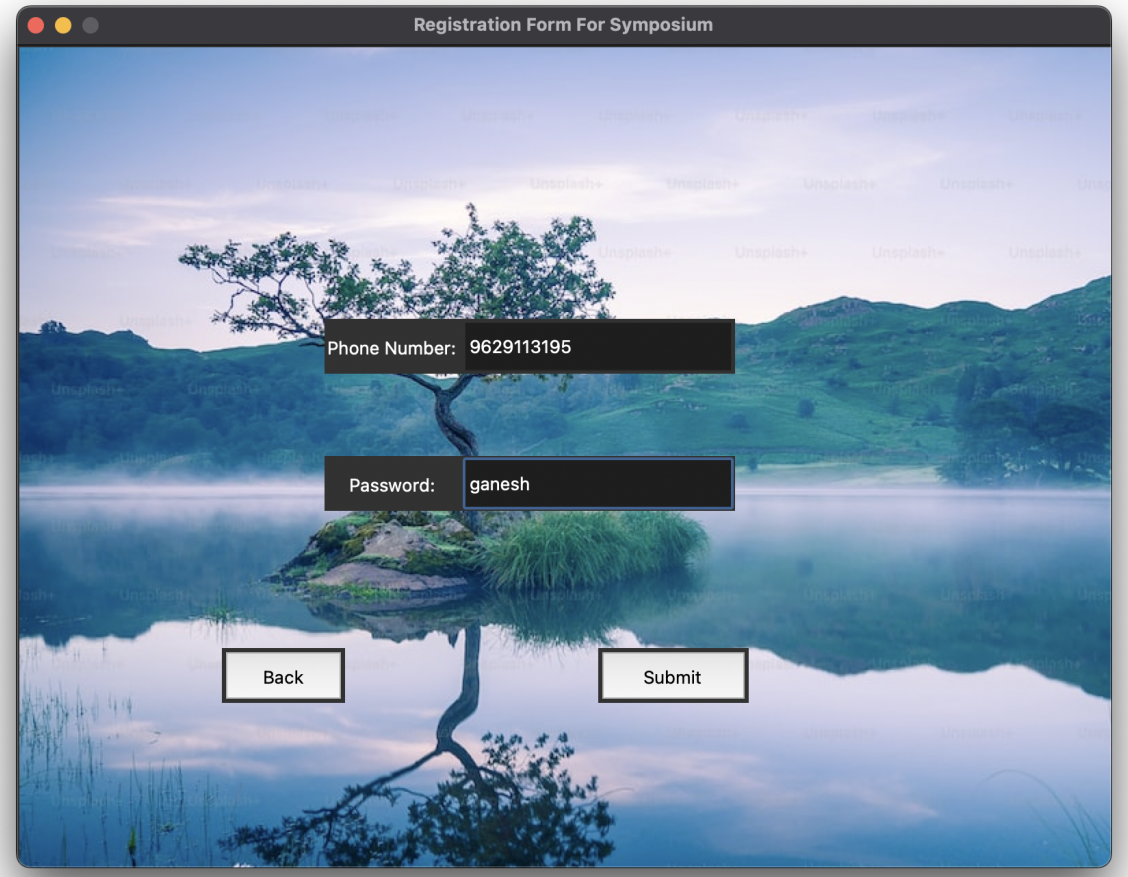
Output Screens:

Front Page:

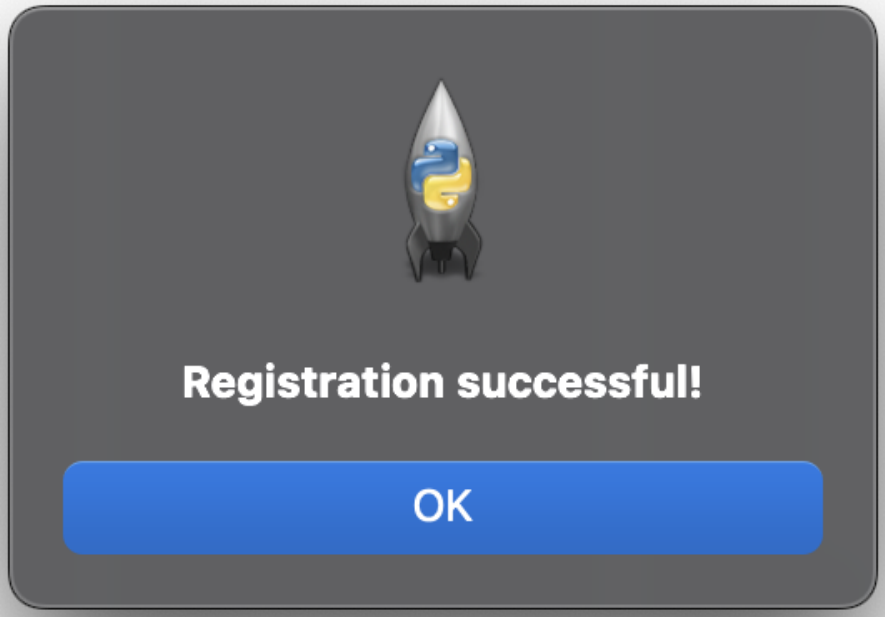
Registration Form For Symposium

Register Now

Registration Page(Signup Page):

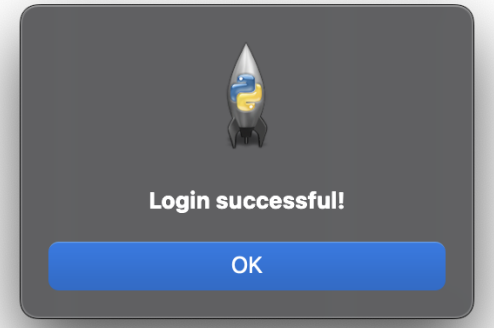


Registration Confirmation message:

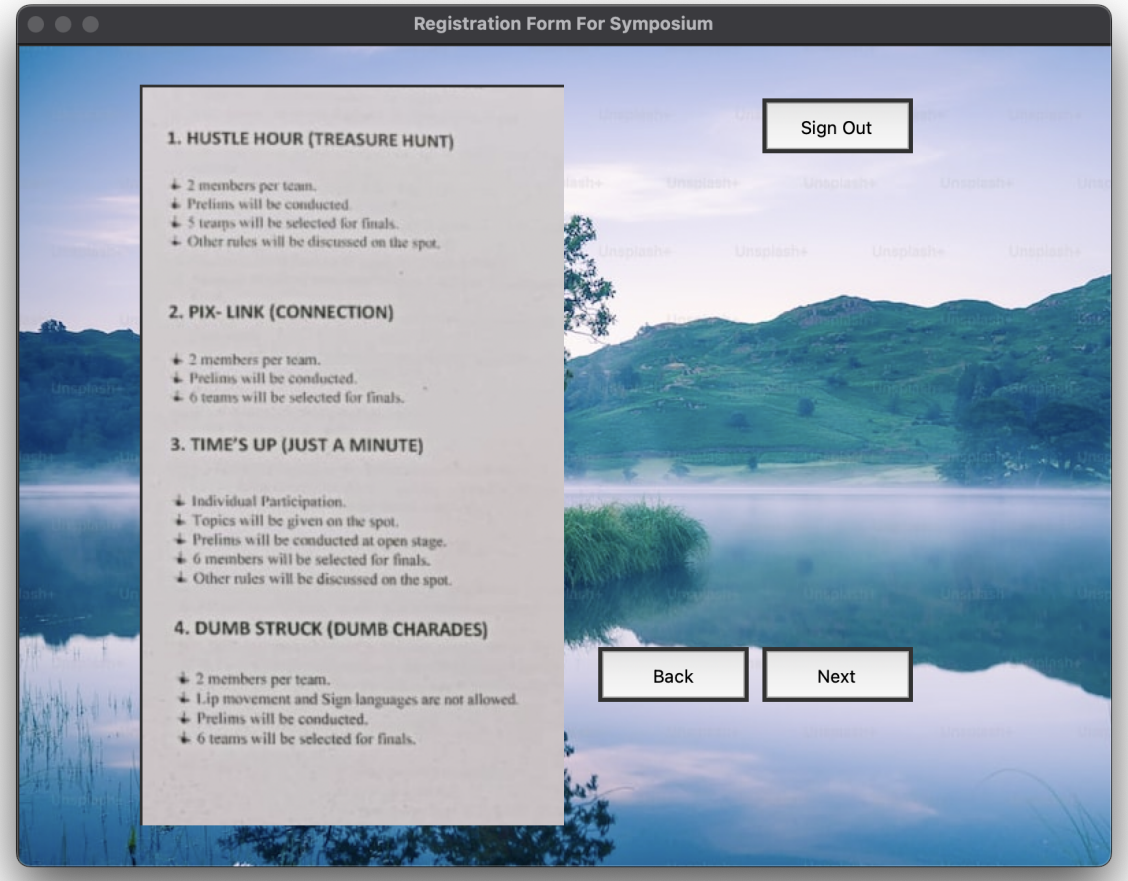


Login Page:

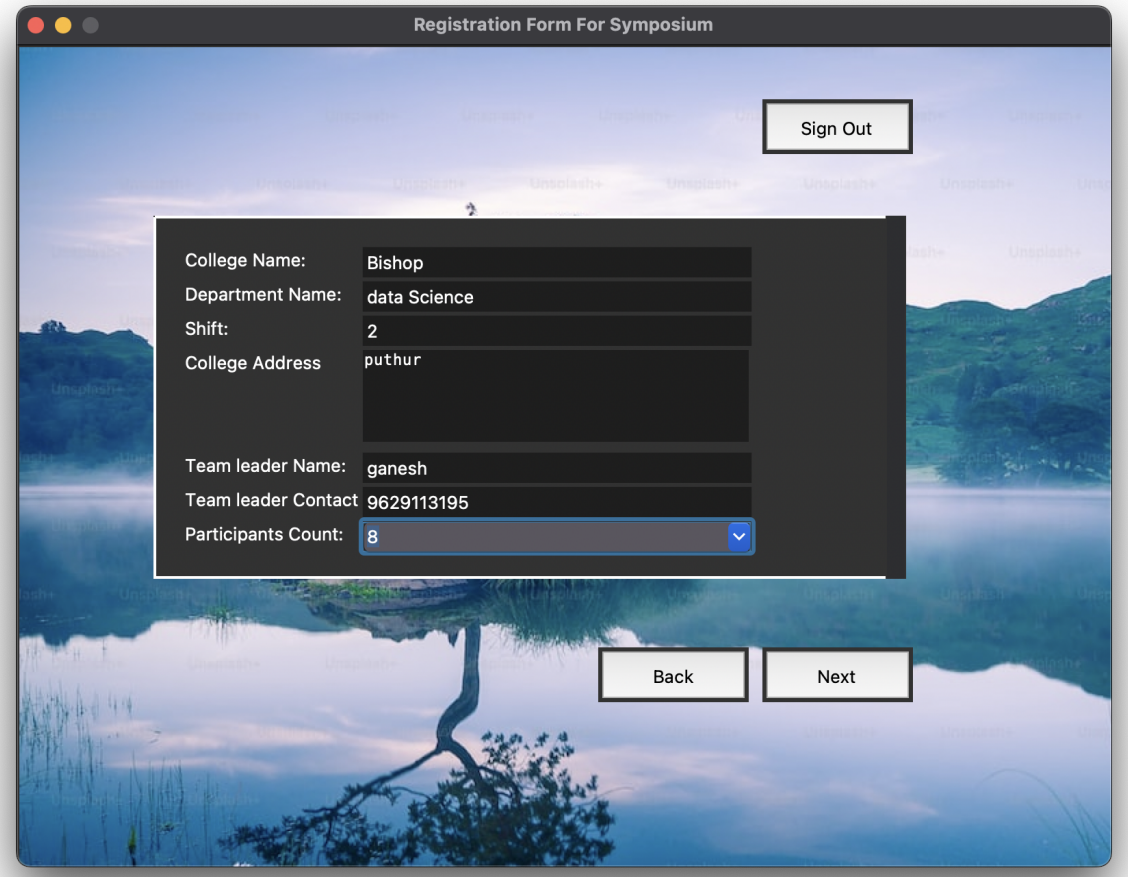
Login Confirm message:



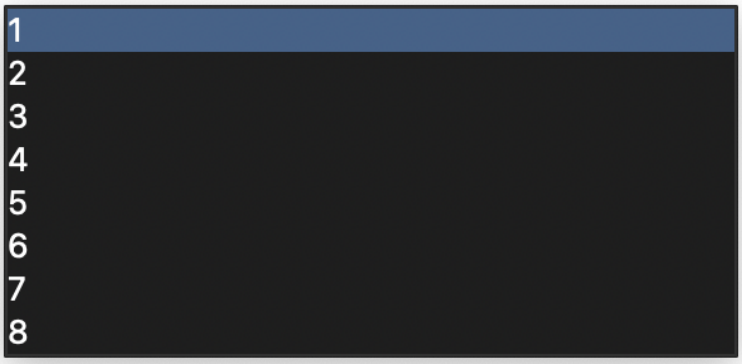
Rules Page:



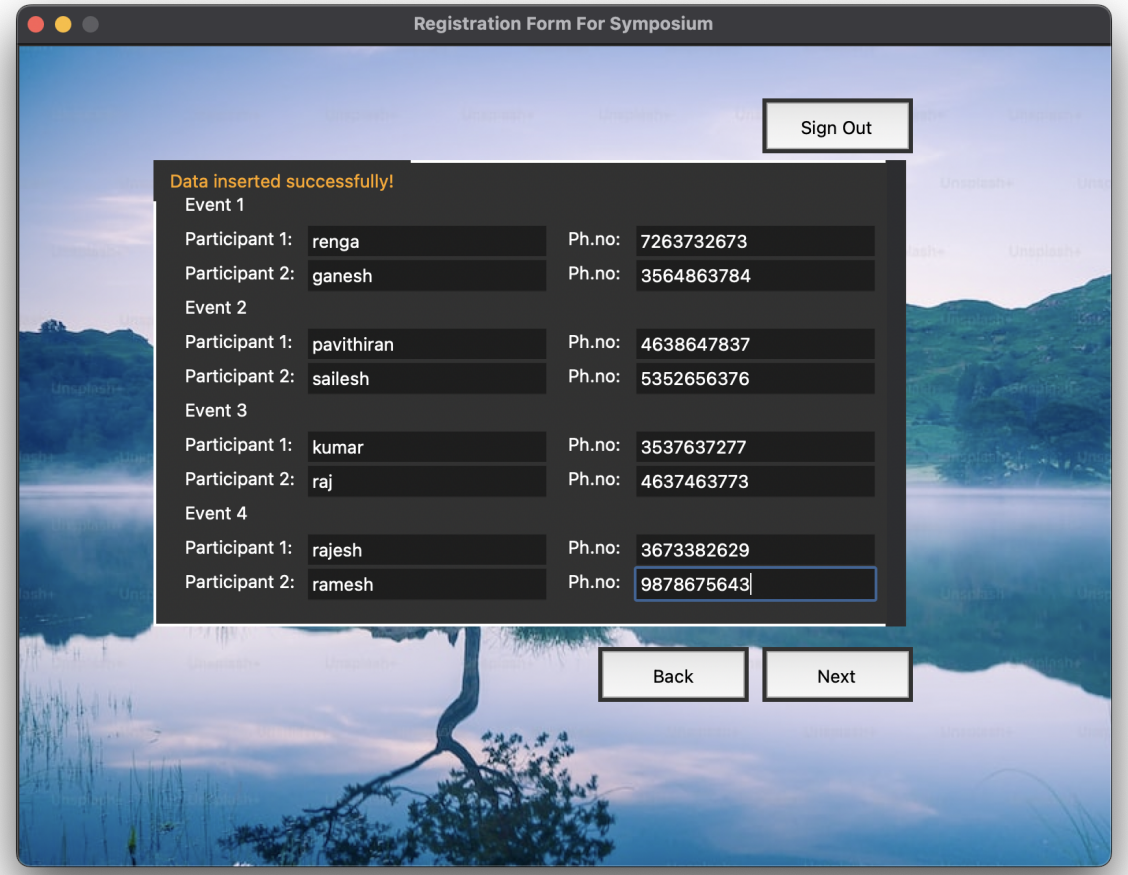
Student Coordinator Page:



Drop down box(Participants Count):

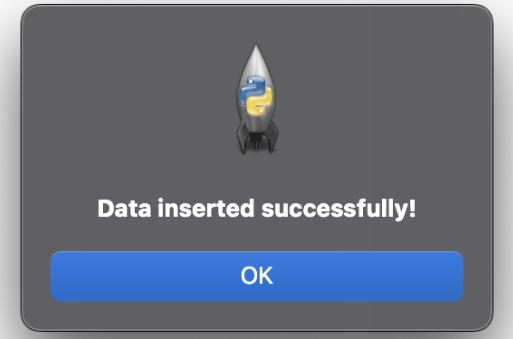


Event Information Page(participant details):



Payment Page:

Registration Confirmation:



Admin login(login Page):

Table Information (admin page):