UNIVERSITI TEKNOLOGI MARA

COLLEGE OF COMPUTING, INFORMATICS AND MATHEMATICS

UNIVERSITI TEKNOLOGI MARA (UiTM)

CAWANGAN KEDAH

DIPLOMA IN LIBRARY INFORMATICS

(CDIM144)

PROGRAMMING FOR LIBRARY

(IML 208)

TITLE REPORT: PATIENT CLINIC REGISTRATION

PREPARED BY:

MUHAMMAD HAFIZUL ZAHEEN BIN MULIADI (2022661936)
GROUP: CDIM1443B

PREPARED FOR:

SIR AIRUL SHAZWAN BIN NORSHAHIMI

SUBMISSION DATE: WEEK 12

TITLE PAGE

TITLE REPORT: PATIENT CLINIC REGISTRATION

NAME: MUHAMMAD HAFIZUL ZAHEEN BIN MULIADI

MATRIC NO: 2022661936

CLASS: CDIM1443B

DIPLOMA IN LIBRARY INFORMATICS

(CDIM144)

UNIVERSITI TEKNOLOGI MARA (UiTM)

CAWANGAN KEDAH

SUBMISSION DATE: WEEK 12

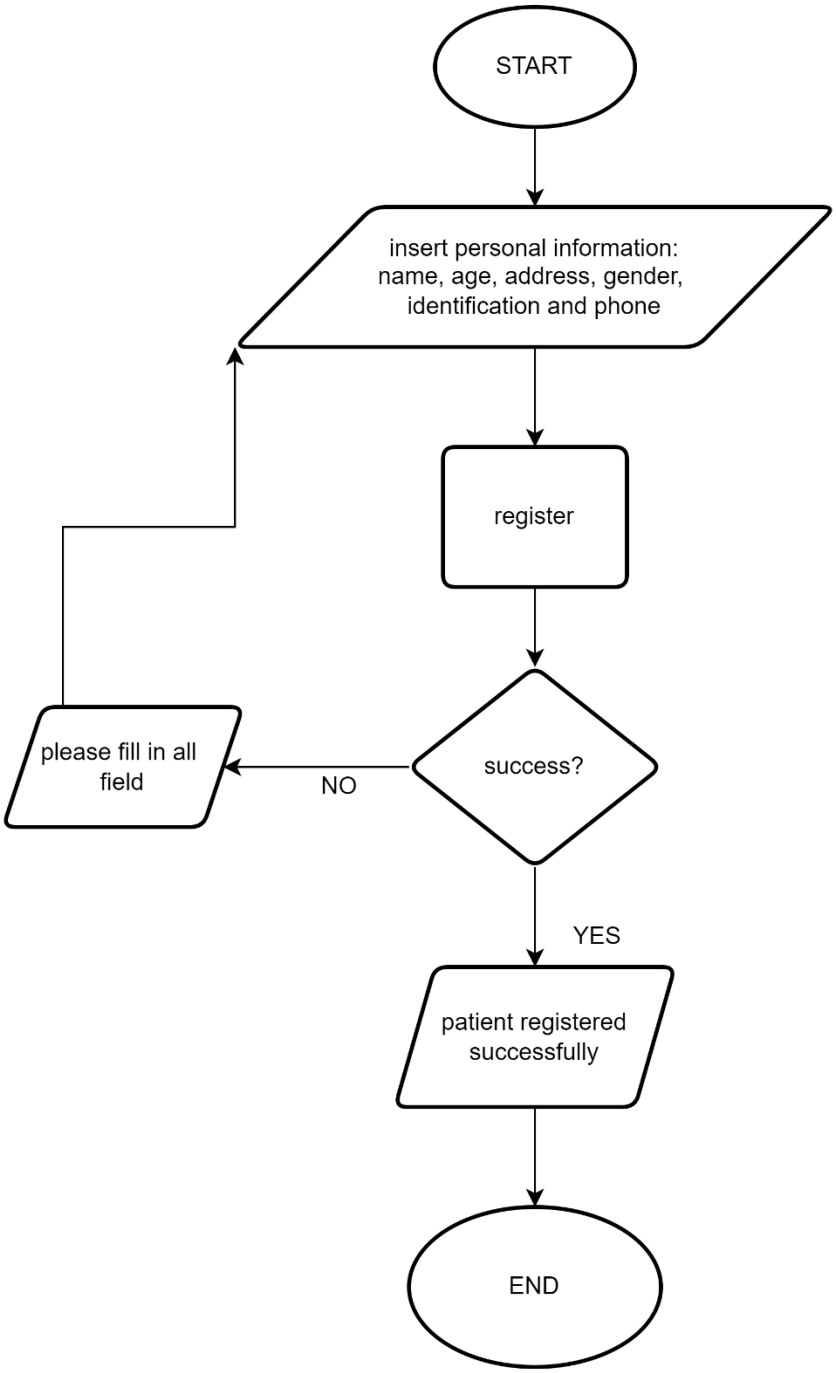
INTRODUCTION

This assignment aims to develop a Patient Clinic Registration System using programming concepts to enhance healthcare accessibility, efficacy, and usability. The system will provide a user-friendly interface for patients and clinic administrators, ensuring seamless registration processes. The project will transcend geographical barriers and prioritize scalability and security to ensure data privacy. The interface will be adaptive to patient-clinic interactions and accommodate evolving healthcare needs. The system will also be fortified with stringent security measures to protect patient information confidentiality and privacy. This project aims to bridge the gap between technology and humanitarian service, enhancing healthcare services for countless individuals in need.

HOW THE SYSTEM WORK

The system going to work with when a user fills up their information in the GUI and then the information is going to appear in the database linked. The project is mean to be for some people who want get check up in clinic early. This project also have a benefit for the clinic organization because they will have a organize work.

FLOW CHART


START
insert personal information:
name, age, address, gender,
identification and phone
register
success?
NO
please fill in all
field
YES
patient registered
successfully
END

PYTHON CODE

assingment.py > create_patient

```python
import tkinter as tk
from tkinter import messagebox
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="patient_clinic_registration"
)

# Create a cursor object to execute SQL queries
mycursor = mydb.cursor()

# Temporary patient data store
patients = []

def create_patient():
    name = name_entry.get()
    age = age_entry.get()
    gender = gender_entry.get()
    address_label = address_entry.get()
    identification = identification_entry.get()
    phone = phone_entry.get()

    print("name:", name)
    print("age:", age)
    print("gender:", gender)
    print("address :", address_label)
    print("id:", identification)
    print("phone:", phone)

    sql = "INSERT INTO patient_registration (name, age, gender, address_label, identification, phone) VALUES (%s, %s, %s, %s
    val = (name, age, gender, address_label, identification, phone)
    mycursor.execute(sql, val)
    mydb.commit()

    # Basic validation
    if name and age and gender and identification and phone:
        patients.append({"Name": name, "Age": age, "Gender": gender, "ID": identification, "Phone": phone})
        messagebox.showinfo("Success", "Patient Registered Successfully!")
        clear_entries()
        update_patient_list()
    else:
        messagebox.showerror("Error", "Please fill in all fields.")

def clear_entries():
    name_entry.delete(0, tk.END)
    age_entry.delete(0, tk.END)
    gender_entry.delete(0, tk.END)
    address_entry.delete(0, tk.END)
    identification_entry.delete(0, tk.END)
    phone_entry.delete(0, tk.END)

def show_patients():
    if patients:
        patient_info = "\n".join([f"Name: {patient['Name']}, Age: {patient['Age']}, Gender: {patient['Gender']}, Address: {p
        messagebox.showinfo("Patient List", patient_info)
```

```
    else:
        messagebox.showinfo("Patient List", "No Patients Registered Yet.")

def clear_patients():
    global patients
    patients = []
    update_patient_list()
    messagebox.showinfo("Success", "All Patients Cleared.")

def edit_patient():
    selection = patient_listbox.curselection()
    if selection:
        selected_patient = patients[selection[0]]
        edit_window = tk.Tk()
        edit_window.title("Edit Patient")
        edit_window.geometry('300x300')

        edit_frame = tk.Frame(edit_window)
        edit_frame.pack()

        name_label = tk.Label(edit_frame, text="Name")
        name_label.grid(row=0, column=0)
        name_entry = tk.Entry(edit_frame)
        name_entry.grid(row=0, column=1)
        name_entry.insert(tk.END, selected_patient['Name'])

        age_label = tk.Label(edit_frame, text="Age")
        age_label.grid(row=1, column=0)
        age_entry = tk.Entry(edit_frame)
```

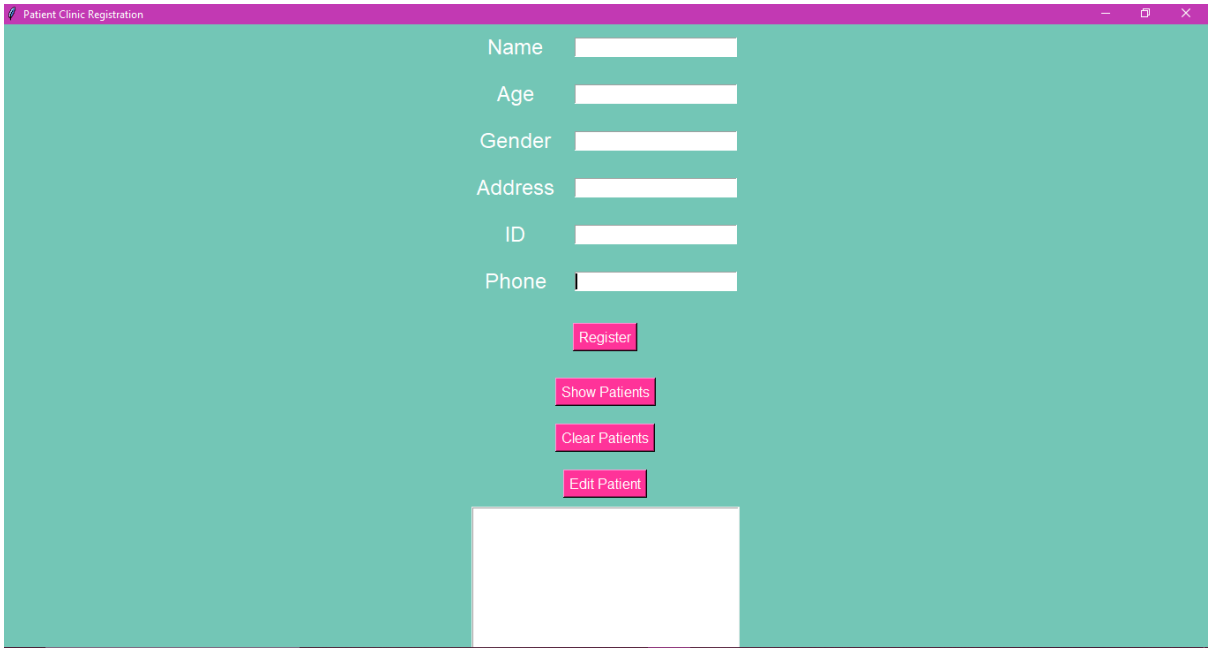
```
        age_entry.grid(row=1, column=1)
        age_entry.insert(tk.END, selected_patient['Age'])

        gender_label = tk.Label(edit_frame, text="Gender")
        gender_label.grid(row=2, column=0)
        gender_entry = tk.Entry(edit_frame)
        gender_entry.grid(row=2, column=1)
        gender_entry.insert(tk.END, selected_patient['Gender'])

        address_label = tk.Label(edit_frame, text="Address")
        address_label.grid(row=3, column=0)
        address_entry = tk.Entry(edit_frame)
        address_entry.grid(row=3, column=1)
        address_entry.insert(tk.END, selected_patient['Address'])

        identification_label = tk.Label(edit_frame, text="ID")
        identification_label.grid(row=4, column=0)
        identification_entry = tk.Entry(edit_frame)
        identification_entry.grid(row=4, column=1)
        identification_entry.insert(tk.END, selected_patient['ID'])

        phone_label = tk.Label(edit_frame, text="Phone")
        phone_label.grid(row=5, column=0)
        phone_entry = tk.Entry(edit_frame)
        phone_entry.grid(row=5, column=1)
        phone_entry.insert(tk.END, selected_patient['Phone'])

        def update_patient():
            patients[selection[0]] = {
```

```
                    "Name": name_entry.get(),
                    "Age": age_entry.get(),
                    "Gender": gender_entry.get(),
                    "Address": address_entry.get(),
                    "ID": identification_entry.get(),
                    "Phone": phone_entry.get()
                }
                messagebox.showinfo("Success", "Patient Details Updated Successfully!")
                edit_window.destroy()
                update_patient_list()

            update_button = tk.Button(edit_window, text="Update", command=update_patient)
            update_button.pack()

        else:
            messagebox.showerror("Error", "Please select a patient to edit.")

    def update_patient_list():
        patient_listbox.delete(0, tk.END)
        for patient in patients:
            patient_listbox.insert(tk.END, f"{patient['Name']}")

    def create_registration_form():
        global name_entry, age_entry, gender_entry, address_entry, identification_entry, phone_entry, patient_listbox

        registration_window = tk.Tk()
        registration_window.title("Patient Clinic Registration")
        registration_window.geometry('450x400')
        registration_window.configure(bg='#73C6B6')
```

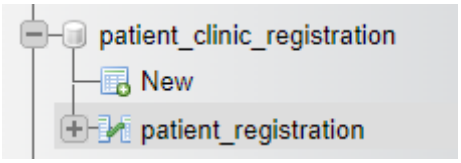
```

        frame = tk.Frame(registration_window, bg='#73C6B6')

        name_label = tk.Label(frame, text="Name", bg='#73C6B6', fg="#FFFFFF", font=("Arial", 18))
        name_entry = tk.Entry(frame, font=("Arial", 12))

        age_label = tk.Label(frame, text="Age", bg='#73C6B6', fg="#FFFFFF", font=("Arial", 18))
        age_entry = tk.Entry(frame, font=("Arial", 12))

        gender_label = tk.Label(frame, text="Gender", bg='#73C6B6', fg="#FFFFFF", font=("Arial", 18))
        gender_entry = tk.Entry(frame, font=("Arial", 12))

        address_label = tk.Label(frame, text="Address", bg='#73C6B6', fg="#FFFFFF", font=("Arial", 18))
        address_entry = tk.Entry(frame, font=("Arial", 12))

        identification_label = tk.Label(frame, text="ID", bg='#73C6B6', fg="#FFFFFF", font=("Arial", 18))
        identification_entry = tk.Entry(frame, font=("Arial", 12))

        phone_label = tk.Label(frame, text="Phone", bg='#73C6B6', fg="#FFFFFF", font=("Arial", 18))
        phone_entry = tk.Entry(frame, font=("Arial", 12))

        register_button = tk.Button(frame, text="Register", bg="#FF3399", fg="#FFFFFF", font=("Arial", 12), command=create_patie
        show_button = tk.Button(frame, text="Show Patients", bg="#FF3399", fg="#FFFFFF", font=("Arial", 12), command=show_patien
        clear_patients_button = tk.Button(frame, text="Clear Patients", bg="#FF3399", fg="#FFFFFF", font=("Arial", 12), command=
        edit_button = tk.Button(frame, text="Edit Patient", bg="#FF3399", fg="#FFFFFF", font=("Arial", 12), command=edit_patient

        name_label.grid(row=0, column=0, padx=10, pady=10)
        name_entry.grid(row=0, column=1, padx=10, pady=10)
        age_label.grid(row=1, column=0, padx=10, pady=10)
```

```
        age_entry.grid(row=1, column=1, padx=10, pady=10)
        gender_label.grid(row=2, column=0, padx=10, pady=10)
        gender_entry.grid(row=2, column=1, padx=10, pady=10)
        address_label.grid(row=3, column=0, padx=10, pady=10)
        address_entry.grid(row=3, column=1, padx=10, pady=10)
        identification_label.grid(row=4, column=0, padx=10, pady=10)
        identification_entry.grid(row=4, column=1, padx=10, pady=10)
        phone_label.grid(row=5, column=0, padx=10, pady=10)
        phone_entry.grid(row=5, column=1, padx=10, pady=10)
        register_button.grid(row=6, column=0, columnspan=2, pady=20)
        show_button.grid(row=7, column=0, columnspan=2, pady=10)
        clear_patients_button.grid(row=8, column=0, columnspan=2, pady=10)
        edit_button.grid(row=9, column=0, columnspan=2, pady=10)

        patient_listbox = tk.Listbox(frame, width=50)
        patient_listbox.grid(row=10, column=0, columnspan=2)
        frame.pack()

        registration_window.mainloop()

    if __name__ == "__main__":
        create_registration_form()
```

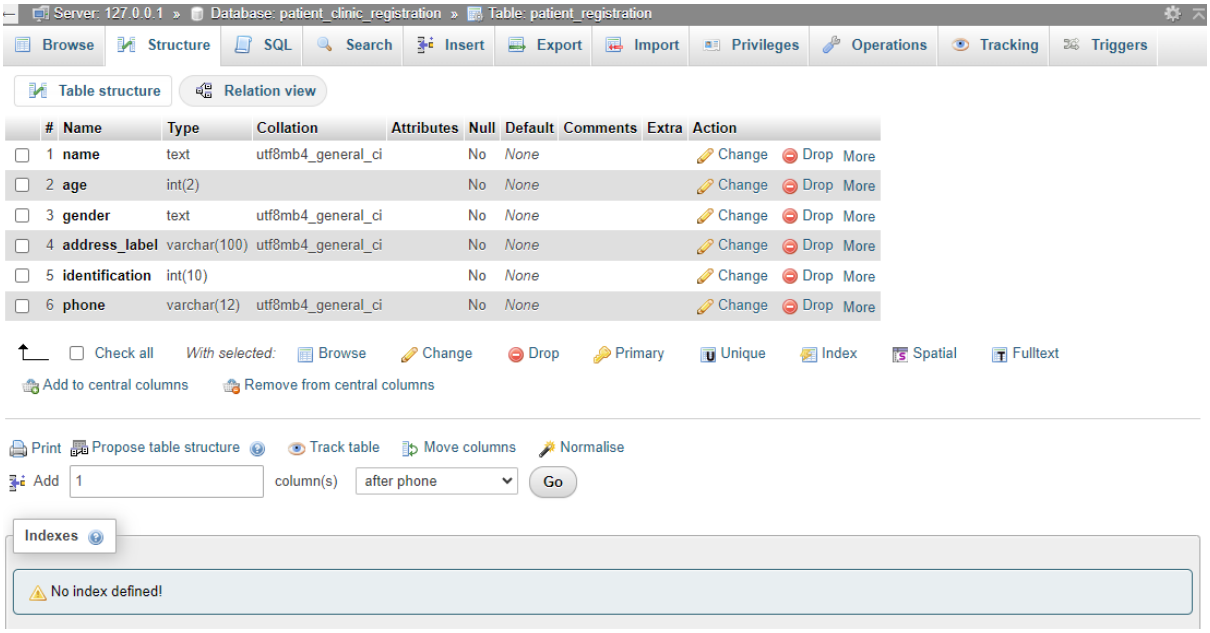
GUI



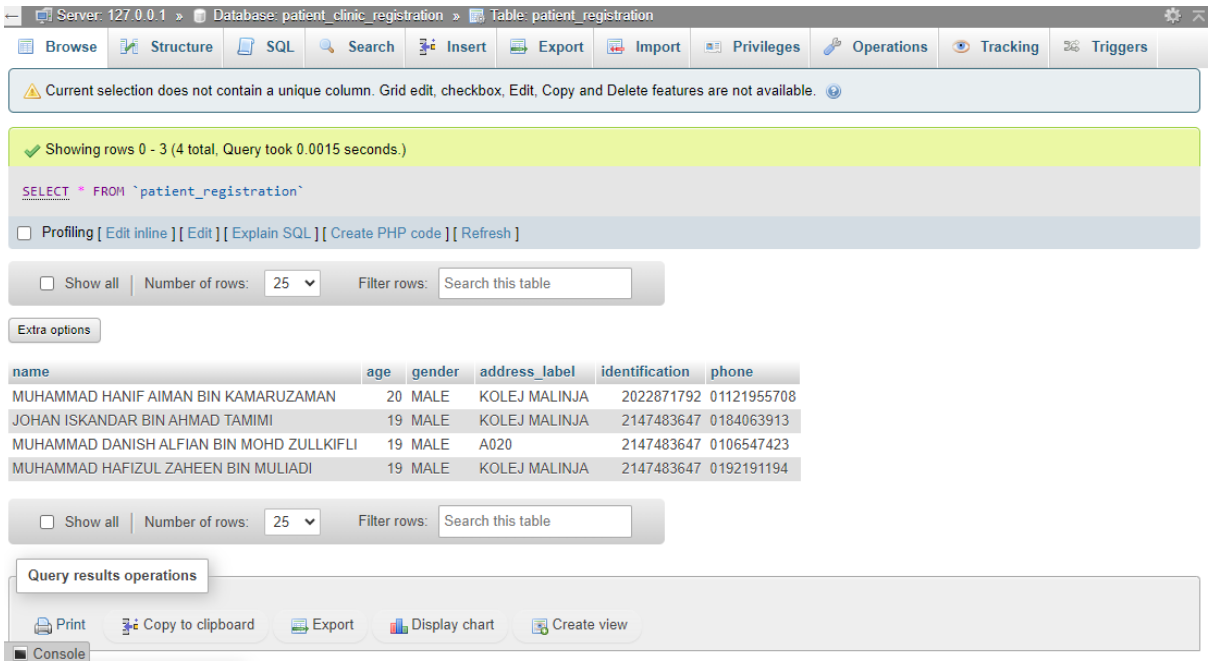
DATABASE



STRUCTURE



BROWSER



CONCLUSION

Project IML208 is a ground-breaking project that optimises patient clinic registration systems using Python programming. As part of this endeavour, an application was developed that integrates technology with human-centric aspects of patient care. Its user-friendly interface improves appointment scheduling and allows for smooth communication between healthcare practitioners and patients. The versatility and robustness of the application provide a smooth connection, improving the overall experience for both patients and healthcare practitioners. Administrators may use the data analysis skills to make educated choices, address emerging patterns, and optimise clinic resources for better patient management. The user-friendly features help to improve operational efficiency and provide a more patient centred healthcare experience. Project IML208 transforms patient clinic registration processes, improving patient care quality and showcasing the beneficial impact of well-designed technology.