UNIVERSITI TEKNOLOGI MARA

KEDAH BRANCH

SCHOOL OF INFORMATION SCIENCE

COLLEGE OF COMPUTING, INFORMATICS AND MATHEMATICS

DIPLOMA IN LIBRARY INFORMATIC (IM144)

IML 208: PROGRAMMING FOR LIBRARIES

ASSESSMENT 2: GROUP ASSIGNMENT

REPORT: CAT HOTEL REGISTRATION

PREPARED BY:

NAME	MATRIX NO
FATEHAH FARHANA BINTI KAMARUDIN	2022454566
NOOR JANNAH AFIFAH BINTI MOHD ZULKIFLI	2022869892
NURUL FITRIYAH BINTI MOHD SHOKRI	2022858108
NURUL SOFIYYAH BINTI AZHAR	2022810368

CLASS: KCDIM1443E

PREPARED FOR:

MR. AIRUL SHAZWAN BIN NORSHAHIMI

SUBMISSION DATE:

16 JANUARY 2024

REPORT: CAT HOTEL REGISTRATION

PREPARED BY:

NAME	MATRIX NO
FATEHAH FARHANA BINTI KAMARUDIN	2022454566
NOOR JANNAH AFIFAH BINTI MOHD ZULKIFLI	2022869892
NURUL FITRIYAH BINTI MOHD SHOKRI	2022858108
NURUL SOFIYYAH BINTI AZHAR	2022810368

CLASS: KCDIM1443E

IM144 – DIPLOMA IN INFORMATIC LIBRARY

SCHOOL OF INFORMATION SCIENCE

COLLEGE OF COMPUTING, INFORMATICS AND MATHEMATICS

UNIVESITI TEKNOLOGI MARA (UITM)

KEDAH BRANCH

TABLE OF CONTENT

1.0 INTRODUCTION

The need for effective and user-friendly registration systems for specialized services, like cat hotels, has grown in the fast-paced pet care industry. We have created a python program about cat hotel registration in response to this necessity. The goal of this system is to completely transform the way cat hotels handle registration procedures by offering automation, personalization, and improved user experiences for both hotel employees and cat owners.

This database system also recognizes that cats have special needs, and it offers customization options for the registration procedure. Cat owners can provide particular information about their animals, such as food preferences, health history, and any special needs for accommodations.

Cat hotel registration system's ultimate goal is to provide a satisfying user experience. It presents a contemporary and effective method of cat hotel registrations by utilizing python's capabilities, satisfying the needs of both cat owners and hotel personnel.

2.0 PROBLEM STATEMENT

In any company, there will be some problem in the database which we call a problem statement.

1. The database may be corrupted which probably switches within the cats.
   For example, the veterinarian almost gives false medicine to the cat because the data has been switched accidentally.

2. Data breaches and concerns about confidentiality are more likely when sensitive data about pets and their owners is manually entered. The privacy of client information must be safeguarded by a compliant and safe data management system.

3. The overall income potential of cat hotels is impacted by lost possibilities to upsell extra services like grooming, veterinary treatment, or luxurious lodging due to the lack of a simplified registration system.

4. A single typo in Python code, such as a misplaced letter or missing punctuation, can cause the system to throw a syntax error, preventing the code from running correctly.

4.0 OBJECTIVES

1. To make sure the data in the system will not corrupt and that it might be harmful to the cats.

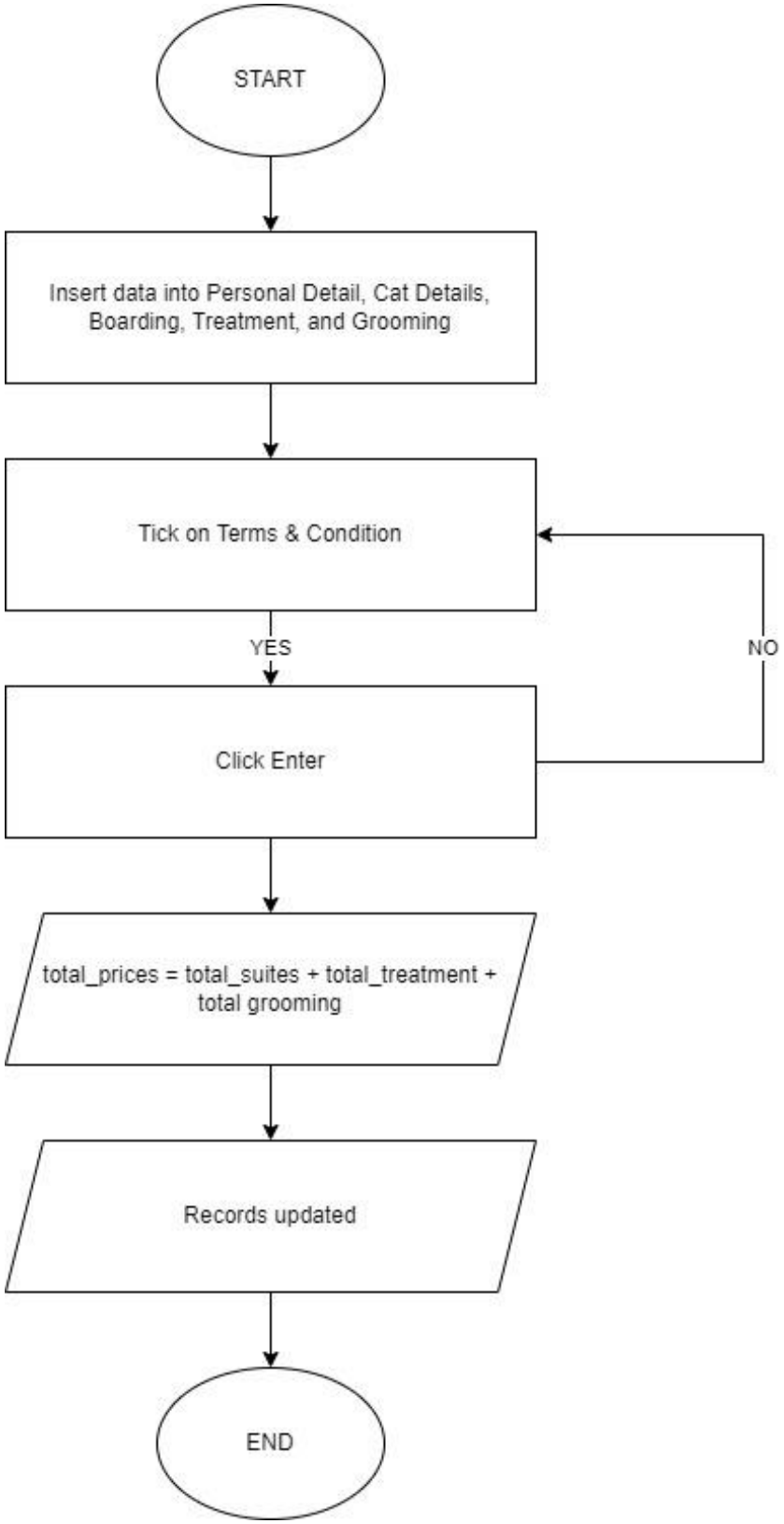
2. Create a scalable registration system to handle the cat hotel's expansion. It is to make sure the system has the flexibility to easily integrate new features and adjust to changing requirements.

3. To enhance the overall user experience by implementing a seamlessly integrated system with a user-friendly graphical user interface (GUI) featuring intuitive design and customization, coupled with a user-friendly database equipped with easy data management, robust authentication, security measures, scalability, and efficient query tools. And being able to rearrange the data.

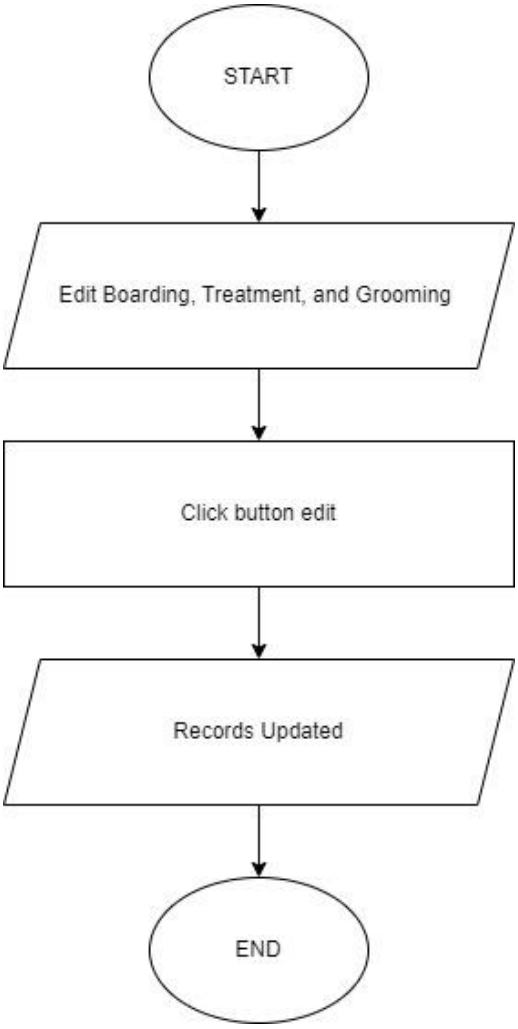
4. A system that makes it easy for staff to fill in data without confusing the services that will be provided to customers

4.0 FLOWCHART

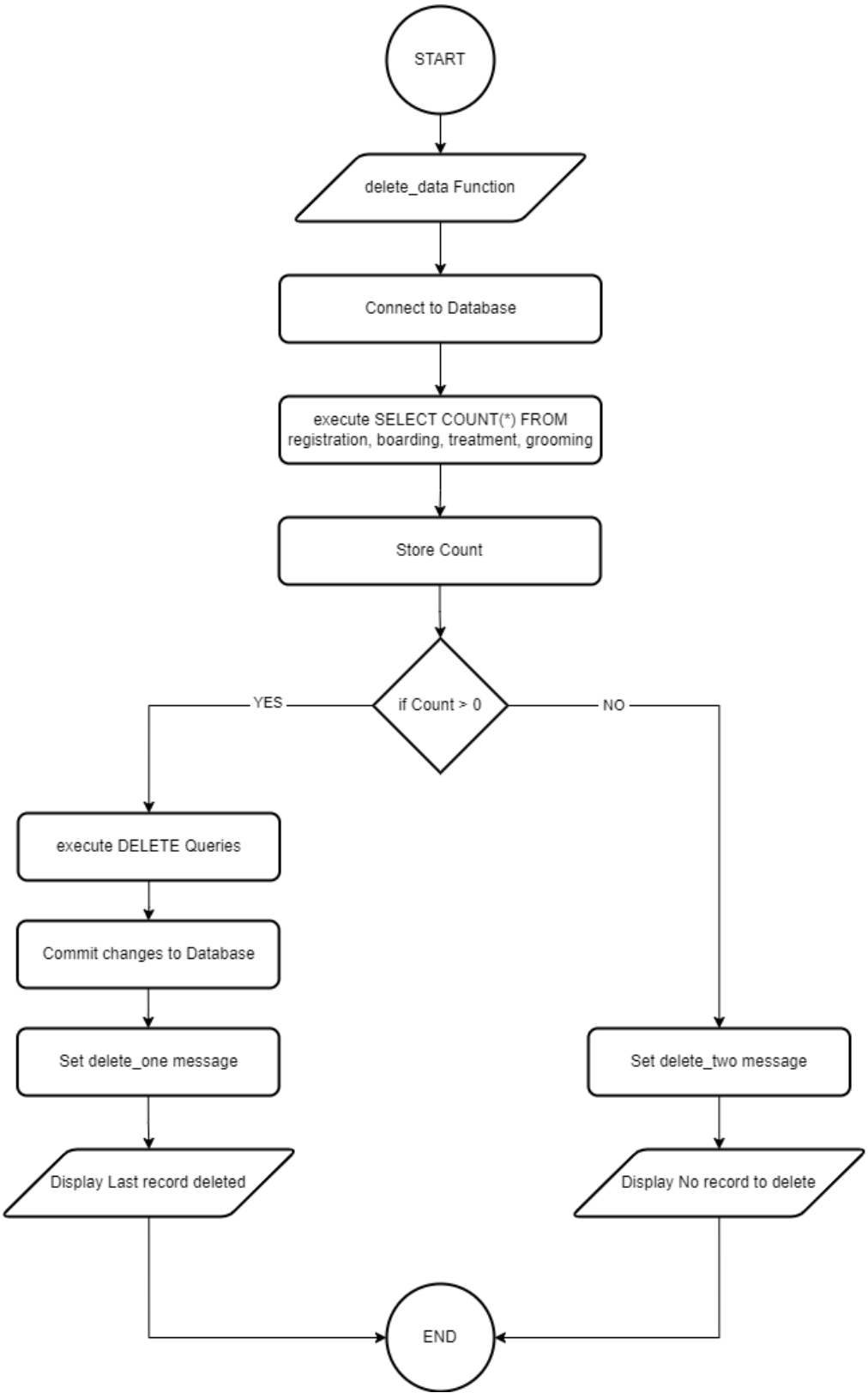
4.1 FLOWCHART OF INSERT DATA



4.2 FLOWCHART OF EDIT DATA



4.3 FLOWCHART OF DELETE FUNCTION



5.0 SNAPSHOT OF CODE

grp_pjc_cat_hotel.py | cat_hotel.sql

grp_pjc_cat_hotel.py > ...

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

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

mycursor = mydb.cursor()

#mainwindow
window = tk.Tk()
window.geometry("870x560")
window['background'] = '#836953'
window.title("Cat Owner Registration")
```

```python
#mainwindow
window = tk.Tk()
window.geometry("870x560")
window['background'] = '#836953'
window.title("Cat Owner Registration")

labelmenu = tk.Label(window, text="\nWelcome MATE !!", font = ('Comic Sans MS bold', 25), fg="white", bg="#836953")
labelmenu.pack(padx=20, pady=1)

labelmenu = tk.Label(window, text="* Make sure to insert the data carefully, \n so that there will not have any mistake *", font = ('Comic Sans M
labelmenu.pack(padx=20, pady=1)

labelmenu = tk.Label(window, text="* Only boarding date, grooming package, and haricut can be edited *", font = ('Comic Sans MS bold', 18), fg="w
labelmenu.pack(padx=20, pady=1)

labelmenu = tk.Label(window, text="* Please treat our customer carefully and helpfull \n so that we get a good review *", font = ('Comic Sans MS
labelmenu.pack(padx=20, pady=1)

labelmenu = tk.Label(window, text="\n ~~Happy Working~~", font = ('Comic Sans MS bold', 18), fg="white", bg="#836953")
labelmenu.pack(padx=20, pady=1)

labelmenu = tk.Label(window, text=" V[redacted]y", font = ('Comic Sans MS bold', 18), fg="white", bg="#836953")
labelmenu.place(x=380, y=470)
```

```
def register_page():

    def collect_data():

        accepted = accept_var.get()

        suite_type        = suite_type_combobox.get()
        board_type        = board_type_combobox.get()
        board_day         = int(board_day_combobox.get())
        medicine_type     = medicine_type_combobox.get()
        treatment_session = int(treatment_session_combobox.get())
        grooming_package  = grooming_package_combobox.get()
        grooming_cut      = grooming_cut_combobox.get()

        prices = {"Rabies FVRCP" : 90, "Self Check-in & out"    : 0,   "lion"        : 35,
                  "FelV VACCINE" : 85, "Self Check-in & Drop"   : 15,  "Asian Lion"  : 35,
                  "FHV-1"        : 96, "Pickup & Self Check-out": 15,  "Natural Look": 20,
                  "ITRACONAZOLE" : 55, "Pickup & Drop"          : 30,  "Panther"     : 35,
                  "TERBINAFINE"  : 61, "Basic"                  : 80,  "Belly Shave" : 20,
                  "FLUCONAZOLE"  : 58, "Full"                   : 150, "Butt Shave"  : 20,
                  "Gold"         : 50, "Deluxe"                 : 200, "None"        : 0,
                  "Platinum"     : 100,"Teddy Bear"             : 40,
                  "Diamond"      : 150,"tiger"                  : 35,}

        total_suite     = (prices[suite_type] * board_day + prices[board_type])
        total_treatment = (prices[medicine_type] * treatment_session)
        total_grooming  = (prices[grooming_package] + prices[grooming_cut])
        total_prices    = total_suite + total_treatment + total_grooming


        output_total = tk.Label(output_frame, text="", fg="white", bg="#836953")
        output_total.grid(row=0, column=1)
        output_total.config(text=f"RM{total_prices}")
```

```
        if accepted == "Accepted":
            id       = owner_id_entry.get()
            name     = owner_name_entry.get()
            dbirth   = d_b_entry.get()
            mbirth   = m_b_entry.get()
            ybirth   = y_b_entry.get()
            address  = owner_address_entry.get()
            phone    = owner_phone_entry.get()
            email    = owner_email_address_entry.get()
```

```
            if name and address and phone and email:
                cat             = cat_name_entry.get()
                cat_age         = cat_age_combobox.get()
                cat_breed       = cat_breed_combobox.get()
                cat_noted       = cat_noted_entry.get()
                board_date      = board_date_entry.get()
                treatment_type  = treatment_type_combobox.get()

                print("Name: ", name)
                print("Birth Date:", dbirth, "/", mbirth, "/", ybirth)
                print("Address:", address)
                print("Contact No:", phone)
                print("Email:", email)
                print("............")
                print("Cat Name:", cat)
                print("Age:", cat_age)
                print("Breed:", cat_breed)
                print("Notes:", cat_noted)
                print("Suite Type:", suite_type)
                print("Board Type:", board_type)
                print("Board Day:", board_day,)
                print("Board Date:", board_date)
                print("Treatment Type:", treatment_type)
                print("Treatment Session:", treatment_session)
                print("Medicine Type:", medicine_type)
                print("Grooming Package:", grooming_package)
                print("Grooming Cut:", grooming_cut)
                print("----------------------------------------------------------------")
            else:
                tk.messagebox.showwarning(title= "Error", message= "There are items that require your attention!")
        else:
            tk.messagebox.showwarning(title= "Error", message= "You have not accepted the terms!")
```

```
        sql  = "INSERT INTO registration (ID, NAME, ADDRESS, PHONE_NO, EMAIL, CAT_NAME, CAT_AGE, CAT_BREED, NOTES) VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%
        val  = (id, name, address, phone, email, cat, cat_age, cat_breed, cat_noted)
        sql2 = "INSERT INTO boarding (SUITE,BOARDING_TYPE,DAY,DAY_CHECK_IN,TOTAL) VALUES (%s,%s,%s,%s,%s)"
        val2 = (suite_type, board_type, board_day, board_date, total_suite)
        sql3 = "INSERT INTO treatment (TREATMENT_TYPE, SESSION,MEDICINE_TYPE,TOTAL) VALUES (%s,%s,%s,%s)"
        val3 = (treatment_type, treatment_session, medicine_type, total_treatment)
        sql4 = "INSERT INTO grooming (PACKAGE,HAIRCUT,TOTAL) VALUES (%s,%s,%s)"
        val4 = (grooming_package, grooming_cut, total_grooming)

        mycursor.execute(sql, val)
        mycursor.execute(sql2, val2)
        mycursor.execute(sql3, val3)
        mycursor.execute(sql4, val4)
        mydb.commit()


    def view_data():
        mydb     = mysql.connector.connect(
        host     = "localhost",
        user     = "root",
        password = "",
        database = "cat_hotel")

        mycursor = mydb.cursor()
        mycursor.execute("SELECT * FROM registration")

        result = mycursor.fetchall()
```

```
        print_record = ''
        if result:
            latest_record = result[-1]
            print_record = f"> {latest_record[0]} \t {latest_record[1]}"
        else:
            print_record = "No records Available"

        view_record = tk.Label(output_frame, text=print_record, fg="white", bg="#836953")
        view_record.grid(row=0, column=3)


        mycursor.close()
        mydb.close()

    def edit_data ():
        boarddate = board_date_entry.get()
        groomingpackage = grooming_package_combobox.get()
        groomingcut = grooming_cut_combobox.get()

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

        mycursor.execute("SELECT * FROM boarding")
        boarding_records = mycursor.fetchone()
        mycursor.fetchall()
        mycursor.execute("SELECT * FROM grooming")
        grooming_records = mycursor.fetchone()
        mycursor.fetchall()

```

```
        edit_one = ''
        edit_two = ''
        if boarding_records is not None:
            sql_boarding = "UPDATE boarding SET DAY_CHECK_IN = %s WHERE DAY_CHECK_IN = %s"
            val_boarding = (boarddate, boarding_records[3])
            mycursor.execute(sql_boarding, val_boarding)
            mydb.commit()
        if grooming_records is not None:
            sql_grooming = "UPDATE grooming SET PACKAGE = %s, HAIRCUT = %s WHERE PACKAGE = %s AND HAIRCUT = %s"
            val_grooming = (groomingpackage, groomingcut,grooming_records[0],grooming_records[1])
            mycursor.execute(sql_grooming, val_grooming)
            mydb.commit()
            edit_one ="Groom ✓ Board ✓"
        else:
            edit_two ="No updated"

        edit_record = tk.Label(output_frame, text=edit_one, fg="white", bg="#836953")
        edit_record.grid(row=0, column=5)
        edit_record = tk.Label(output_frame, text=edit_two, fg="white", bg="#836953")
        edit_record.grid(row=0, column=5)
```

```
200
201     def delete_data():
202         mydb = mysql.connector.connect(
203         host = "localhost",
204         user = "root",
205         password = "",
206         database = "cat_hotel")
207
208         mycursor = mydb.cursor()
209
210         mycursor.execute("SELECT COUNT(*) FROM registration")
211         count = mycursor.fetchone()[0]
212         mycursor.execute("SELECT COUNT(*) FROM boarding")
213         count = mycursor.fetchone()[0]
214         mycursor.execute("SELECT COUNT(*) FROM treatment")
215         count = mycursor.fetchone()[0]
216         mycursor.execute("SELECT COUNT(*) FROM grooming")
217         count = mycursor.fetchone()[0]
218
```

```
219         delete_one = ''
220         delete_two = ''
221         if count > 0:
222             sql  = "DELETE FROM registration ORDER BY NAME DESC LIMIT 1"
223             sql2 = "DELETE FROM boarding ORDER BY SUITE DESC LIMIT 1"
224             sql3 = "DELETE FROM treatment ORDER BY TREATMENT_TYPE DESC LIMIT 1"
225             sql4 = "DELETE FROM grooming ORDER BY PACKAGE DESC LIMIT 1"
226             mycursor.execute(sql)
227             mycursor.execute(sql2)
228             mycursor.execute(sql3)
229             mycursor.execute(sql4)
230             mydb.commit()
231             delete_one = "Last record deleted."
232         else:
233             delete_two = "No record   to delete."
234         delete_record = tk.Label(output_frame, text=delete_two, fg="white", bg="#836953")#
235         delete_record.grid(row=0, column=7)
236         delete_record_two = tk.Label(output_frame, text=delete_one, fg="white", bg="#836953")
237         delete_record_two.grid(row=0, column=7)
238
```

```python
    #register window
    root = tk.Toplevel(window)
    root.geometry("870x560")
    root['background'] = '#836953'
    root.title("Cat Owner Registration")

    label = tk.Label(root, text = "Customer Registration", font = ('Comic Sans MS bold', 18), fg="white", bg="#836953")
    label.pack(padx=20, pady=1)

    out_frame = tk.Frame(root, bg="#836953")
    out_frame.pack(padx=20, pady=1)

    frame = tk.LabelFrame(out_frame, bg="#836953")
    frame.grid(row=0, column=0)

    #owner
    cat_owner_frame = tk.LabelFrame(frame, text = "Personal Detail", fg="white", bg="#836953")
    cat_owner_frame.grid(row= 0, column=0, sticky="News", padx=1, pady=1)

    owner_id = tk.Label(cat_owner_frame, text="Id :", fg="white", bg="#836953")
    owner_id.grid(row=0, column=0)
    owner_id_entry = tk.Entry(cat_owner_frame, width=23)
    owner_id_entry.grid(row=0, column=1)

    owner_name_label = tk.Label(cat_owner_frame, text="Name :", fg="white", bg="#836953")
    owner_name_label.grid(row=1, column=0)
    owner_name_entry = tk.Entry(cat_owner_frame, width=23)
    owner_name_entry.grid(row=1, column=1)

    owner_birth_date = tk.Label(cat_owner_frame, text="    Birth Date :    ", fg="white", bg="#836953")
    owner_birth_date.grid(row=2, column=0)
```

```python
grp_pjc_cat_hotel.py > register_page > delete_data
    d_b_entry = ttk.Combobox(cat_owner_frame, values=["1","2","3","4","5","6","7","8","9","10",
                                                      "11","12","13","14","15","16","17","18","19","20",
                                                      "21","22","23","24","25","26","27","28","29","30","31"])
    d_b_entry.grid(row=2, column=1)
    d_b_entry.set("Date")
    d_b_entry["state"] = 'readonly'

    m_b_entry = ttk.Combobox(cat_owner_frame, values=["01","02","03","04","05","06",
                                                      "07","08","09","10","11","12"])
    m_b_entry.grid(row=3, column=1)
    m_b_entry.set("Month")
    m_b_entry["state"] = 'readonly'

    y_b_entry = ttk.Combobox(cat_owner_frame, values=list(range(1960, 2200)))
    y_b_entry.grid(row=4, column=1)
    y_b_entry.set("Year")
    y_b_entry["state"] = 'readonly'

    for widget in cat_owner_frame.winfo_children():
        widget.grid_configure(padx= 10, pady=5)

    cont = tk.LabelFrame(frame, text="-", fg="white", bg="#836953")
    cont.grid(row=0, column=1, sticky="News", padx=1, pady=1)

    owner_address_label = tk.Label(cont, text="       Address :      ", fg="white", bg="#836953")
    owner_address_label.grid(row=5, column=0)
    owner_address_entry = tk.Entry(cont, width=23)
    owner_address_entry.grid(row=5, column=1)

    owner_phone_label = tk.Label(cont, text="Contact No :", fg="white", bg="#836953")
    owner_phone_label.grid(row=6, column=0)
    owner_phone_entry = tk.Entry(cont, width=23)
    owner_phone_entry.grid(row=6, column=1)
```

```
grp_pjc_cat_hotel.py > register_page > delete_data
305     owner_email_address_label = tk.Label(cont, text="Email Address :", fg="white", bg="#836953")
306     owner_email_address_label.grid(row=7, column=0)
307     owner_email_address_entry = tk.Entry(cont, width=23)
308     owner_email_address_entry.grid(row=7, column=1)
309
310     for widget in cont.winfo_children():
311         widget.grid_configure(padx= 10, pady=5)
312
313     #cat
314     cat_frame = tk.LabelFrame(frame, text="Cat Details", fg="white", bg="#836953")
315     cat_frame.grid(row=0, column=2, sticky="News", padx=1, pady=1)
316
317     cat_name_label = tk.Label(cat_frame, text="Name :", fg="white", bg="#836953")
318     cat_name_label.grid(row=0, column=0)
319     cat_name_entry = tk.Entry(cat_frame, width=23)
320     cat_name_entry.grid(row=0, column=1)
321
322     cat_age_label = tk.Label(cat_frame, text="Age :", fg="white", bg="#836953")
323     cat_age_label.grid(row=1, column=0)
324     cat_age_combobox = ttk.Combobox(cat_frame, values=["Kitten(0-1)", "Young Adult(1-6)",
325                                                        "Mature Adult(7-10)", "Senior(>10)"])
326     cat_age_combobox.grid(row=1, column=1)
327     cat_age_combobox.set("")
328     cat_age_combobox["state"] = 'readonly'
329
330     cat_breed_label = tk.Label(cat_frame, text="     Breed :     ", fg="white", bg="#836953")
331     cat_breed_label.grid(row=2, column=0)
332     cat_breed_combobox = ttk.Combobox(cat_frame, values=["Domestic", "American", "Siamese", "Maine Coon", "Ragdoll",
333                                                          "Russian Blue", "Bengal", "Bombay", "Persian"])
334     cat_breed_combobox.grid(row=2, column=1)
335     cat_breed_combobox.set("")
336     cat_breed_combobox["state"] = 'readonly'
337
```

```
338     cat_noted_label = tk.Label(cat_frame, text="Notes :", fg="white", bg="#836953")
339     cat_noted_label.grid(row=3, column=0)
340     cat_noted_entry = tk.Entry(cat_frame, width=23)
341     cat_noted_entry.grid(row=3, column=1)
342
343     for widget in cat_frame.winfo_children():
344         widget.grid_configure(padx=10, pady=5)
345
346     #boarding
347     boarding_frame = tk.LabelFrame(frame, text="Boarding", fg="white", bg="#836953")
348     boarding_frame.grid(row= 1, column=0, sticky="News", padx=1, pady=1)
349
350     suite_type_label = tk.Label(boarding_frame, text="Suite Type :", fg="white", bg="#836953")
351     suite_type_label.grid(row=0, column=0)
352     suite_type_combobox = ttk.Combobox(boarding_frame, values=["Gold", "Platinum", "Diamond"])
353     suite_type_combobox.grid(row=0, column=1)
354     suite_type_combobox.set("")
355     suite_type_combobox["state"] = 'readonly'
356
357     board_type_label = tk.Label(boarding_frame, text="Boarding Type :", fg="white", bg="#836953")
358     board_type_label.grid(row=1, column=0)
359     board_type_combobox = ttk.Combobox(boarding_frame, values=["Self Check-in & out",
360                                                                "Self Check-in & Drop",
361                                                                "Pickup & Self Check-out",
362                                                                "Pickup & Drop"])
363     board_type_combobox.grid(row=1, column=1)
364     board_type_combobox.set("")
365     board_type_combobox["state"] = 'readonly'
```

```
367     board_day_label = tk.Label(boarding_frame, text="Day :", fg="white", bg="#836953")
368     board_day_label.grid(row=2, column=0)
369     board_day_combobox = ttk.Combobox(boarding_frame, values=["1","2","3","4","5","6","7","8","9","10",
370                                                               "11","12","13","14","15","16","17","18","19","20",
371                                                               "21","22","23","24","25","26","27","28","29","30","31"])
372     board_day_combobox.grid(row=2, column=1)
373     board_day_combobox.set("")
374     board_day_combobox["state"] = 'readonly'
375
376     board_date_label = tk.Label(boarding_frame, text="Date Check In :", fg="white", bg="#836953")
377     board_date_label.grid(row=3, column=0)
378     board_date_entry = tk.Entry(boarding_frame, width=23)
379     board_date_entry.grid(row=3, column=1)
380     board_date_entry.insert(0, "dd/mm/yyyy")
381
382     for widget in boarding_frame.winfo_children():
383         widget.grid_configure(padx=10, pady=5)
384
```

```
385     #treatment
386     treatment_frame = tk.LabelFrame(frame, text="Treatment", fg="white", bg="#836953")
387     treatment_frame.grid(row=1, column=1, sticky="News", padx=1, pady=1)
388
389     treatment_type_label = tk.Label(treatment_frame, text="Treatment Type :", fg="white", bg="#836953")
390     treatment_type_label.grid(row=0, column=0)
391     treatment_type_combobox = ttk.Combobox(treatment_frame, values=["None","vaccine", "Fungus"])
392     treatment_type_combobox.grid(row=0, column=1)
393     treatment_type_combobox.set("")
394     treatment_type_combobox["state"] = 'readonly'
395
396     treatment_session_label = tk.Label(treatment_frame, text="Session :", fg="white", bg="#836953")
397     treatment_session_label.grid(row=1, column=0)
398     treatment_session_combobox = ttk.Combobox(treatment_frame, values=["0","1","2","3","4"])
399     treatment_session_combobox.grid(row=1, column=1)
400     treatment_session_combobox.set("")
401     treatment_session_combobox["state"] = 'readonly'
402
403     medicine_type_Label = tk.Label(treatment_frame, text="Medicine Type :", fg="white", bg="#836953")
404     medicine_type_Label.grid(row=2, column=0)
405     medicine_type_combobox = ttk.Combobox(treatment_frame, values=["None","Rabies FVRCP", "FelV VACCINE","FHV-1",
406                                                                    "ITRACONAZOLE", "TERBINAFINE", "FLUCONAZOLE"])
407     medicine_type_combobox.grid(row=2, column=1)
408     medicine_type_combobox.set("")
409     medicine_type_combobox["state"] = 'readonly'
410
411     for widget in treatment_frame.winfo_children():
412         widget.grid_configure(padx=10, pady=5)
413
```

```
414        #grooming
415        grooming_frame = tk.LabelFrame(frame, text="Grooming", fg="white", bg="#836953")
416        grooming_frame.grid(row=1, column=2, sticky="News", padx=1, pady=1)
417
418        grooming_package_label = tk.Label(grooming_frame, text="Package :", fg="white", bg="#836953")
419        grooming_package_label.grid(row=0, column=0)
420        grooming_package_combobox = ttk.Combobox(grooming_frame, values=["None","Basic", "Full", "Deluxe"])
421        grooming_package_combobox.grid(row=0, column=1)
422        grooming_package_combobox.set("")
423        grooming_package_combobox["state"] = 'readonly'
424
425        grooming_cut_label = tk.Label(grooming_frame, text="Cat Haircut :", fg="white", bg="#836953")
426        grooming_cut_label.grid(row=1, column=0)
427        grooming_cut_combobox = ttk.Combobox(grooming_frame, values=["None","Teddy Bear","lion","Asian Lion","tiger",
428                                                                     "Natural Look","Panther","Belly Shave","Butt Shave"])
429        grooming_cut_combobox.grid(row=1, column=1)
430        grooming_cut_combobox.set("")
431        grooming_cut_combobox["state"] = 'readonly'
432
433        for widget in grooming_frame.winfo_children():
434            widget.grid_configure(padx= 10, pady=5)
435
436        #terms
437        terms_frame = tk.LabelFrame(out_frame, text="Terms & Condition", fg="white", bg="#836953")
438        terms_frame.grid(row=1, column=0, sticky="News", padx=1, pady=1)
439
440        accept_var = tk.StringVar(value="Not Accepted")
441        terms_check = tk.Checkbutton(terms_frame, text="I accept the terms and condition.", fg="#3d251e",
442                                     variable=accept_var, onvalue="Accepted", offvalue="Not Accepted", bg="#836953")
443        terms_check.grid(row=0, column=0)
444
```

```
445        #notes
446        output_frame = tk.LabelFrame(out_frame, text="Notes :", fg="white", bg="#836953")
447        output_frame.grid(row=2, column=0, sticky="News", padx=1, pady=1)
448
449        total_label = tk.Label(output_frame, text="Total :", fg="white", bg="#836953")
450        total_label.grid(row=0, column=0)
451        total_label = tk.Label(output_frame, text=". . . . . .", fg="#836953", bg="#836953")
452        total_label.grid(row=0, column=1)
453        view_view = tk.Label(output_frame, text="View Data :", fg="white", bg="#836953")
454        view_view.grid(row=0, column=2)
455        view_view = tk.Label(output_frame, text=". . . . . .", fg="#836953", bg="#836953")
456        view_view.grid(row=0, column=3)
457        edit_view = tk.Label(output_frame, text="Edit Data :", fg="white", bg="#836953")
458        edit_view.grid(row=0, column=4)
459        edit_view = tk.Label(output_frame, text=". . . . . .", fg="#836953", bg="#836953")
460        edit_view.grid(row=0, column=5)
461        delete_view = tk.Label(output_frame, text="Delete Data :", fg="white", bg="#836953")
462        delete_view.grid(row=0, column=6)
463        delete_view = tk.Label(output_frame, text=". . . . . .", fg="#836953", bg="#836953")
464        delete_view.grid(row=0, column=7)
465
466        for widget in output_frame.winfo_children():
467            widget.grid_configure(padx= 10, pady=5)
468
469        button_frame = tk.LabelFrame(out_frame, text="", fg="white", bg="#836953")
470        button_frame.grid(row=3, column=0, sticky="News", padx=1, pady=1)
471
```

```
471         #button
472         enter_button = tk.Button(button_frame, text="Enter", bg="white", command= collect_data)
473         enter_button.grid(row=0, column=0, ipadx=59)
474 
475         view_button = tk.Button(button_frame, text="View", bg="white", command=view_data)
476         view_button.grid(row=0, column=1, ipadx=93)
477 
478         update_button = tk.Button(button_frame, text="Edit", bg="white", command=edit_data)
479         update_button.grid(row=0, column=2, ipadx=93)
480 
481         delete_button = tk.Button(button_frame, text="Delete Last Record", bg="white", command=delete_data)
482         delete_button.grid(row=0, column=3, ipadx=60)
483 
484         for widget in button_frame.winfo_children():
485             widget.grid_configure( pady=5)
486 
487         mainmenu_button = tk.Button(out_frame, text="Close", bg="white", command=root.destroy)
488         mainmenu_button.grid(row=4, column=0, pady=10, ipadx=393)
489 
490     register_button = tk.Button(window, text="Register New Customer", bg="white", command=register_page)
491     register_button.place(x=15, y=390, width=845, height=30)
492 
493     register_button = tk.Button(window, text="Close", bg="white", command=window.destroy)
494     register_button.place(x=15, y=430, width=845, height=30)
495 
496     window.mainloop()
```

6.0 SNAPSHOT OF PROJECT (GUI)

7.0 SNAPSHOT OF DATABASE (XAMPP)

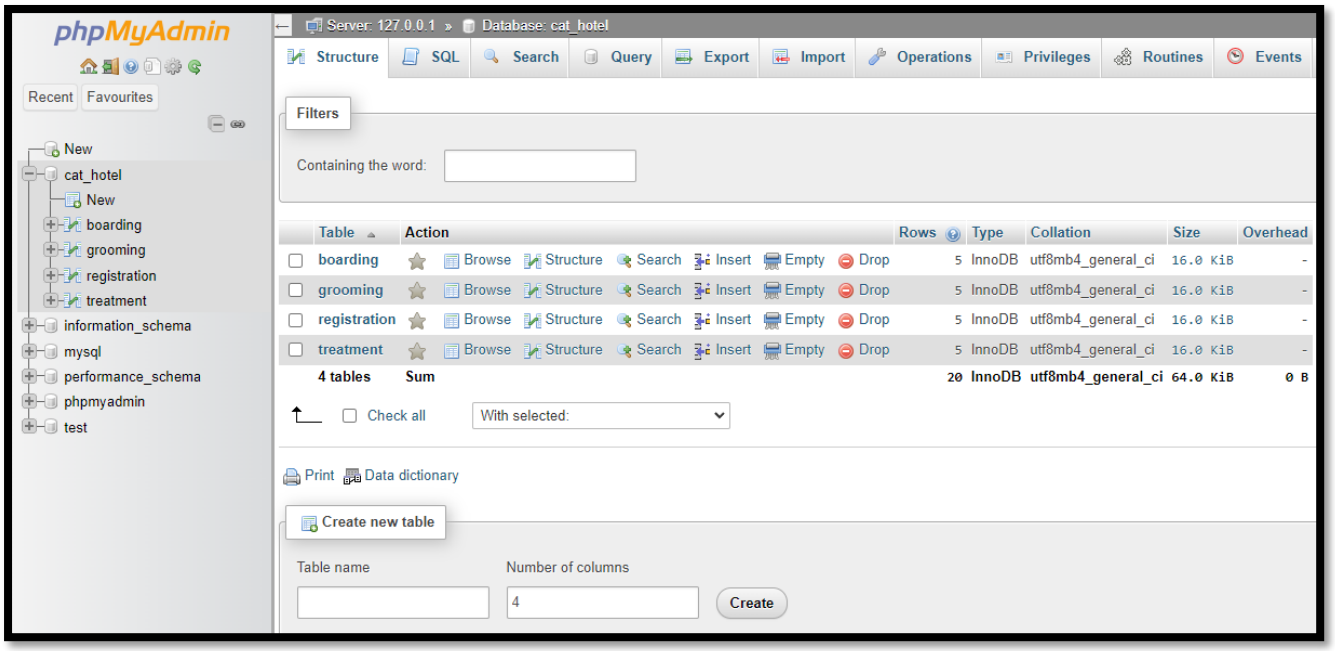
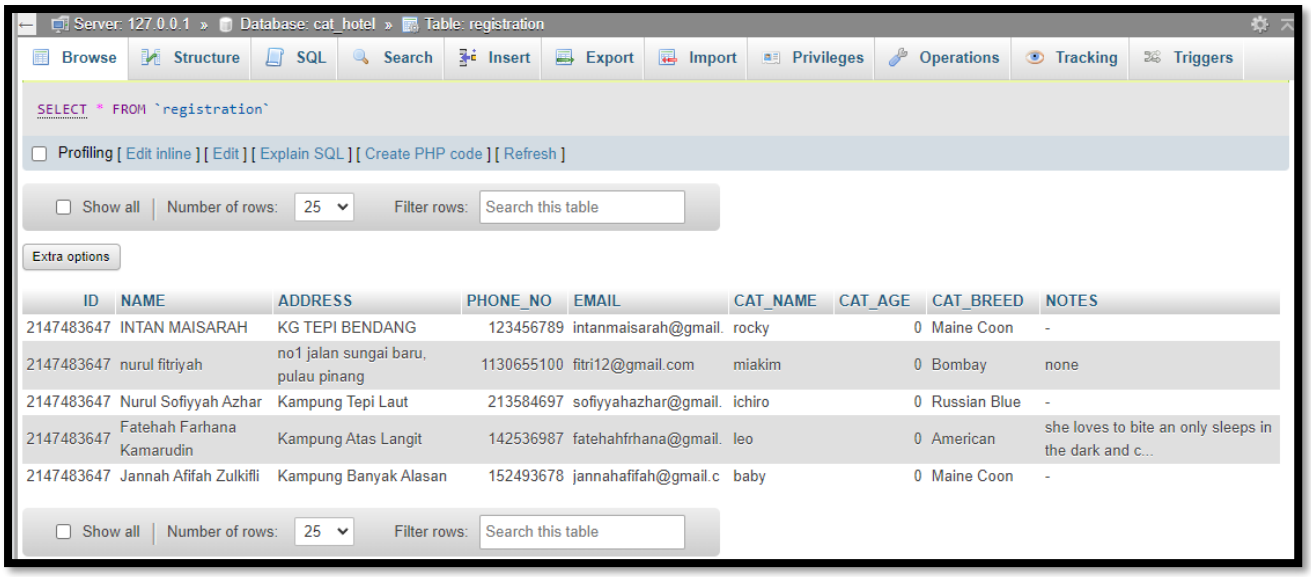


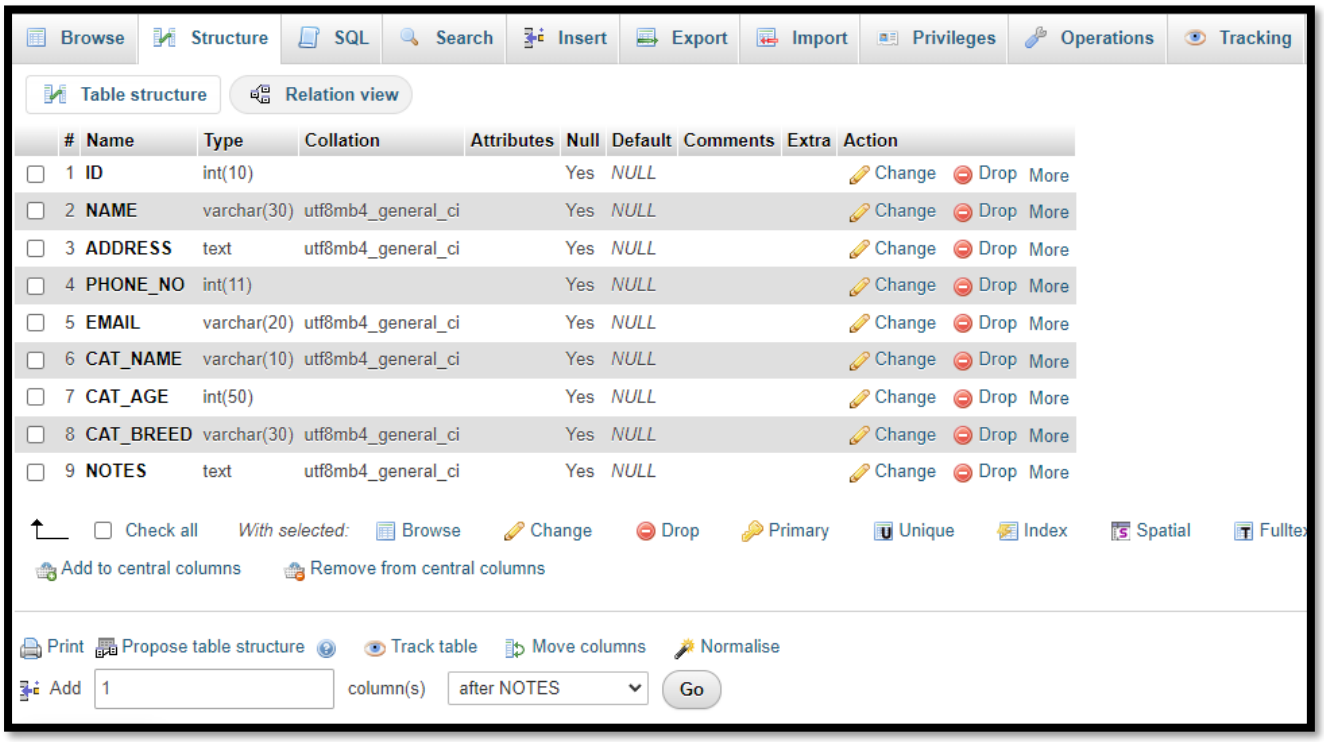
Figure 7.0: DATABASE CAT HOTEL REGISTRATION

7.1 REGISTRATION DATABASE



ID	NAME	ADDRESS	PHONE_NO	EMAIL	CAT_NAME	CAT_AGE	CAT_BREED	NOTES
2147483647	INTAN MAISARAH	KG TEPI BENDANG	123456789	intanmaisarah@gmail.	rocky	0	Maine Coon	-
2147483647	nurul fitriyah	no1 jalan sungai baru, pulau pinang	1130655100	fitri12@gmail.com	miakim	0	Bombay	none
2147483647	Nurul Sofiyyah Azhar	Kampung Tepi Laut	213584697	sofiyyahazhar@gmail.	ichiro	0	Russian Blue	-
2147483647	Fatehah Farhana Kamarudin	Kampung Atas Langit	142536987	fatehahfrhana@gmail.	leo	0	American	she loves to bite an only sleeps in the dark and c...
2147483647	Jannah Afifah Zulkifli	Kampung Banyak Alasan	152493678	jannahafifah@gmail.c	baby	0	Maine Coon	-

Figure 7.1: BROWSE



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(10)			Yes	NULL			Change Drop More
2	NAME	varchar(30)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	ADDRESS	text	utf8mb4_general_ci		Yes	NULL			Change Drop More
4	PHONE_NO	int(11)			Yes	NULL			Change Drop More
5	EMAIL	varchar(20)	utf8mb4_general_ci		Yes	NULL			Change Drop More
6	CAT_NAME	varchar(10)	utf8mb4_general_ci		Yes	NULL			Change Drop More
7	CAT_AGE	int(50)			Yes	NULL			Change Drop More
8	CAT_BREED	varchar(30)	utf8mb4_general_ci		Yes	NULL			Change Drop More
9	NOTES	text	utf8mb4_general_ci		Yes	NULL			Change Drop More

Figure 7.2: STRUCTURE

7.2 BOARDING DATABASE

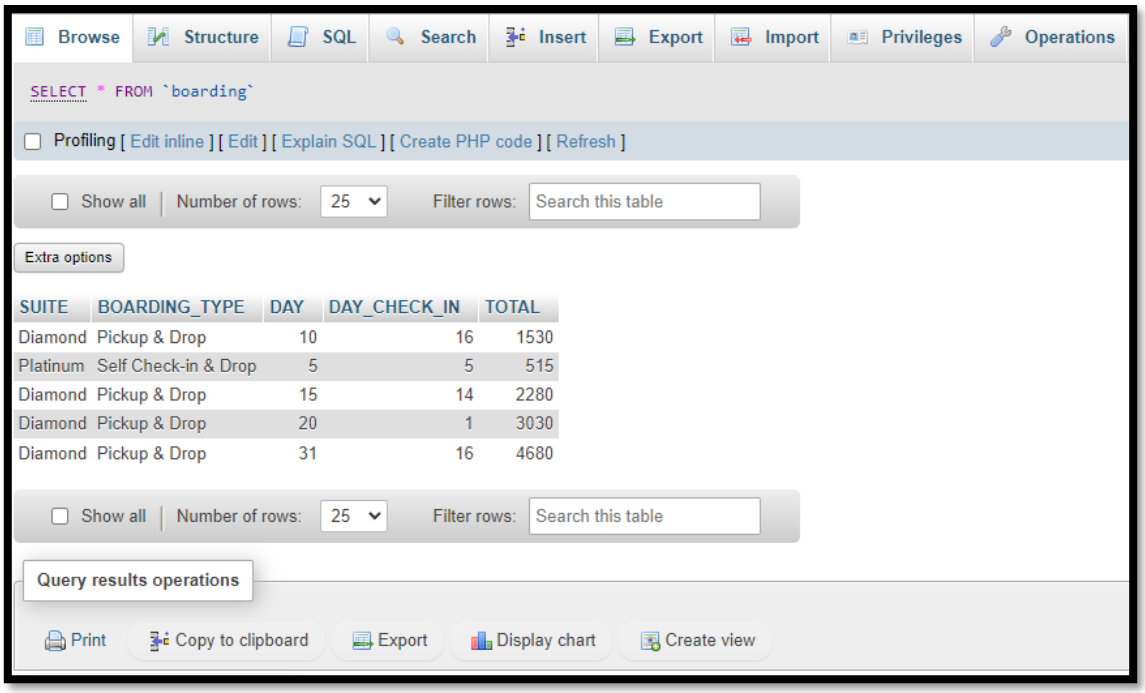


Figure 7.4: BROWSE

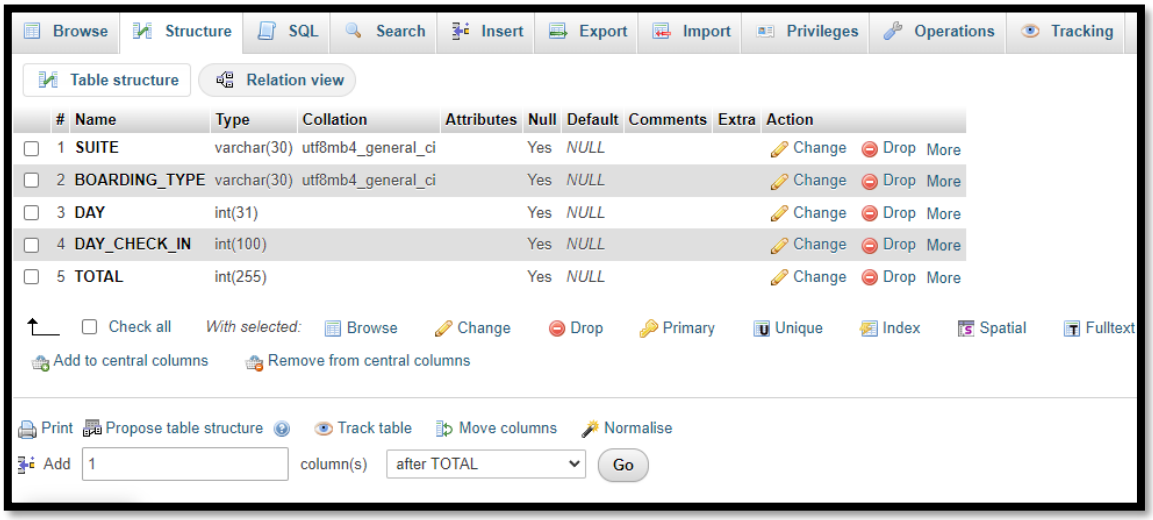


Figure 7.5: STRUCTURE

7.3 TREATMENT DATABASE

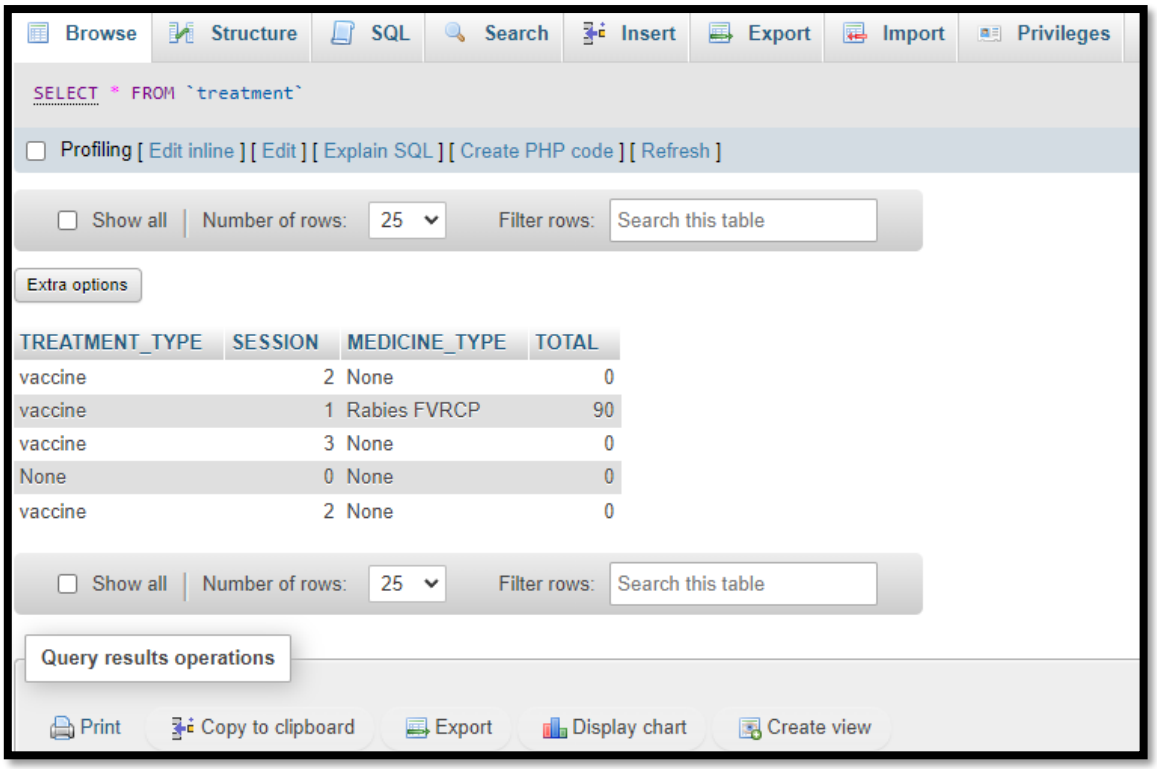


Figure 7.6: BROWSE

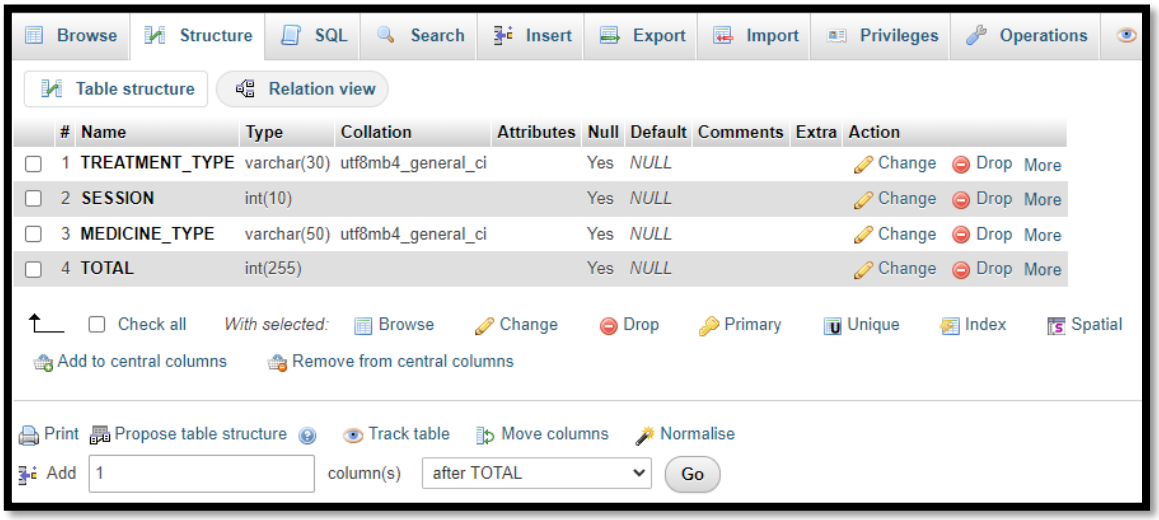


Figure 7.7: STRUCTURE

7.4 GROOMING DATABASE

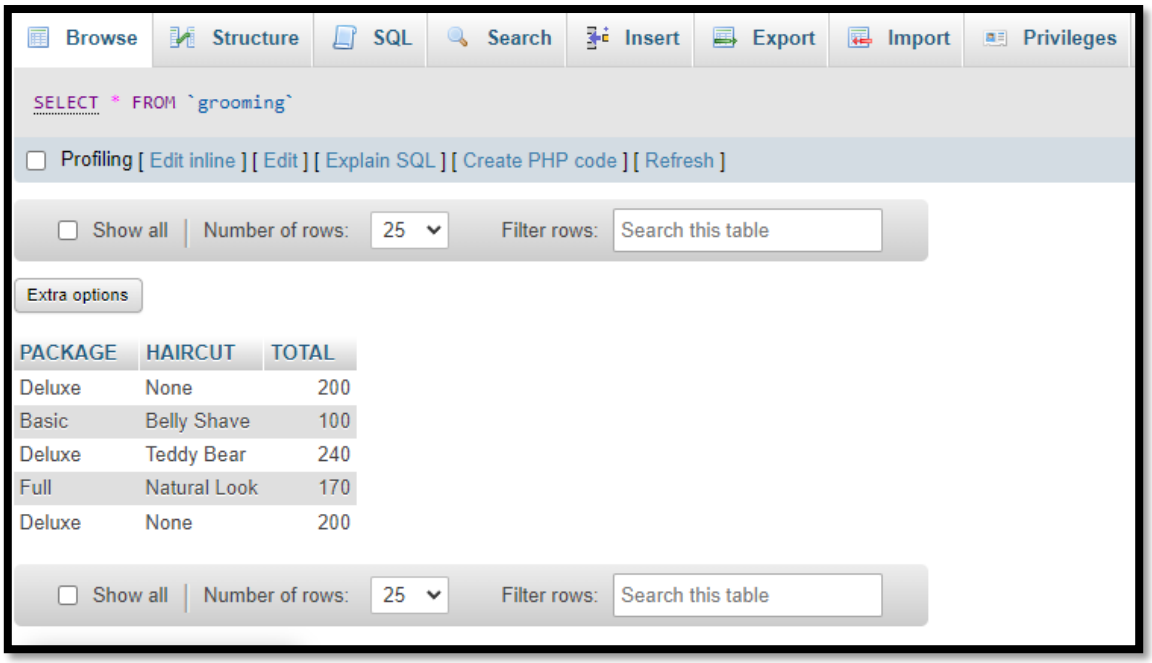


Figure 7.8: BROWSE

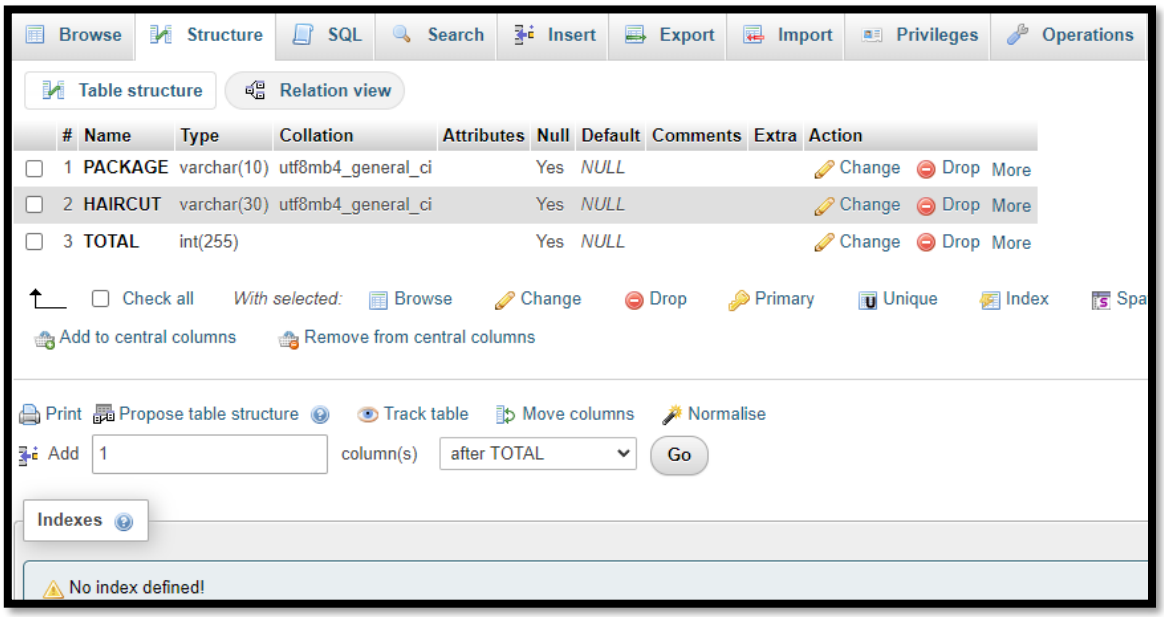


Figure 7.9: STRUCTURE

8.0 CONCLUSION

In conclusion, in this project, we conclude that the problem statement above can be solved with the objectives that have been built. This gives the result of a more effective and user-friendly system to use the registration system that has been designed by us. The database created has difficulty being damaged and the data stored will be maintained because it is confidential information. The system is also very flexible and easy to use. To sum up, both a user-friendly GUI and a database are essential components for a successful software application. They should complement each other, providing a seamless and intuitive experience for users interacting with the system. Balancing aesthetics, functionality, and ease of use in both the interface and underlying database contributes to a positive user experience.