



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..... | 1 |
| 2.0 PROBLEM STATEMENT | 2 |
| 3.0 OBJECTIVES | 3 |
| 4.0 FLOWCHART..... | 4-6 |
| 5.0 SNAPSHOT OF CODE..... | 7-16 |
| 6.0 SNAPSHOT OF GUI..... | 17 |
| 7.0 SNAPSHOT OF DATABASE..... | 18-22 |
| 8.0 CONCLUSION | 23 |

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.

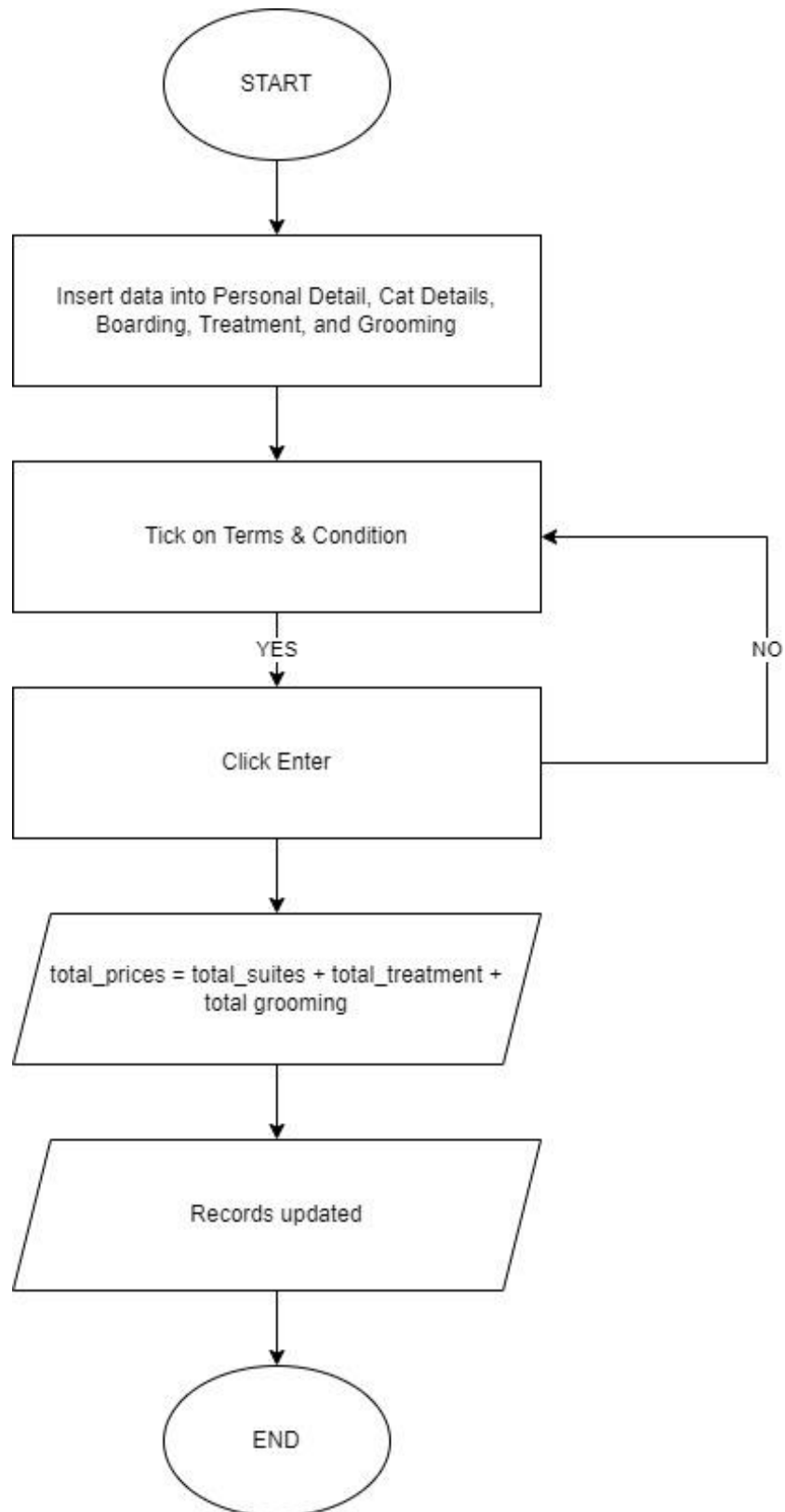
1. The database may be corrupted which probably switches within the cats.
For example, the veterinarian almost gives false medicine to the cat because the data has been switched accidentally.
2. Data breaches and concerns about confidentiality are more likely when sensitive data about pets and their 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

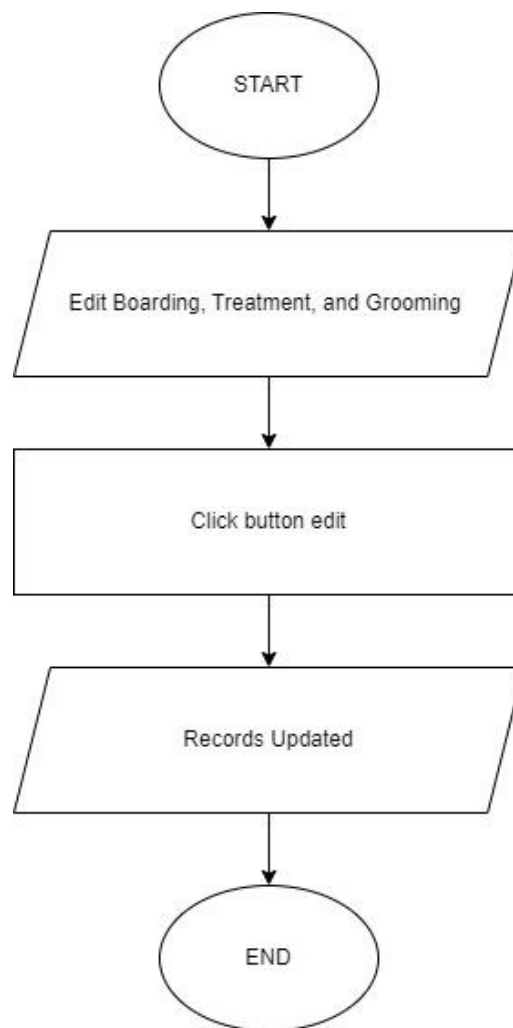
1. To make sure the data in the system will not corrupt and that it might be harmful to the cats.
2. Create a scalable registration system to handle the cat hotel's expansion. It is to make sure the system has the flexibility to easily integrate new features and adjust to changing requirements.
3. To 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

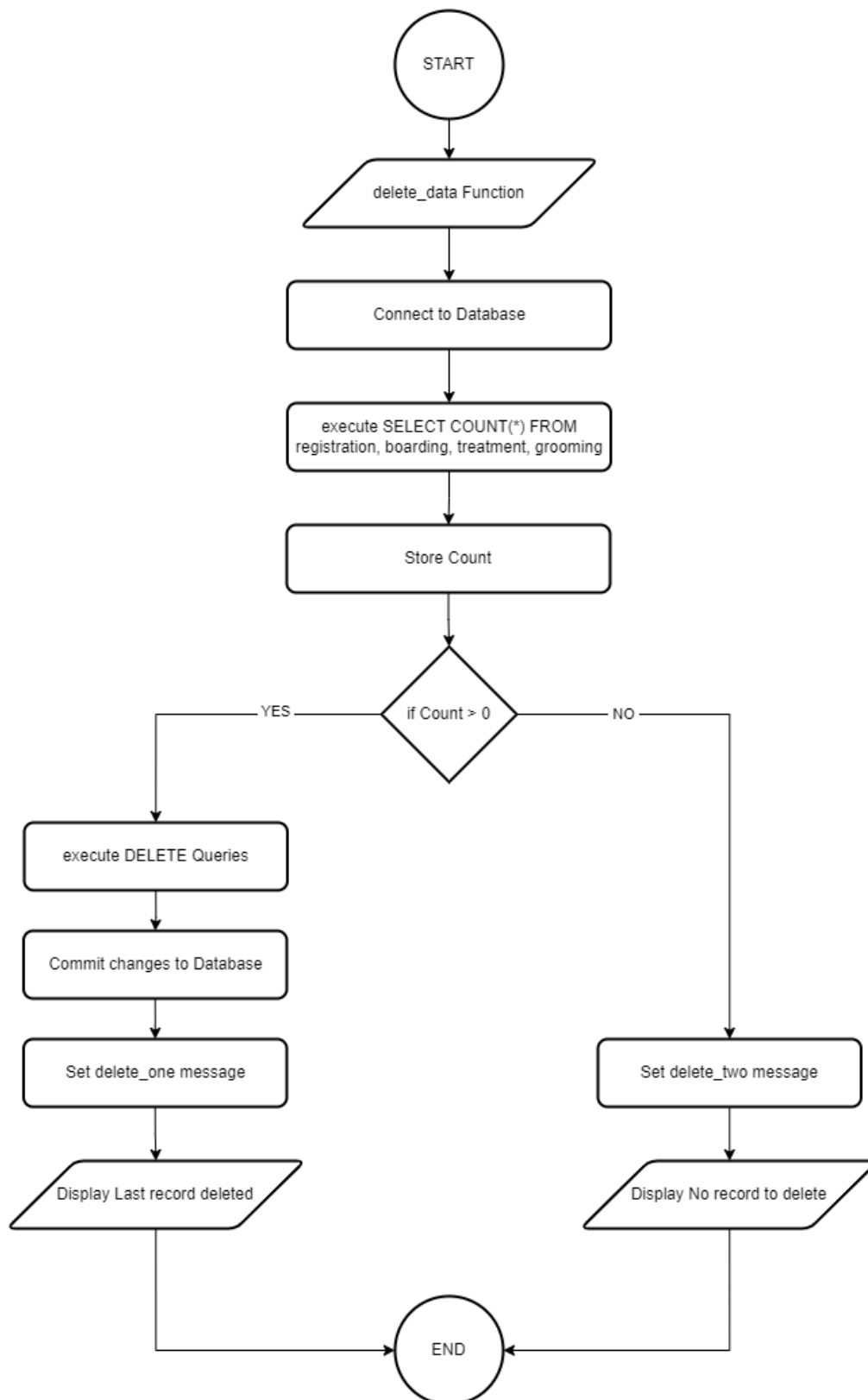
4.1 FLOWCHART OF INSERT DATA



4.2 FLOWCHART OF EDIT DATA



4.3 FLOWCHART OF DELETE FUNCTION



5.0 SNAPSHOT OF CODE

```
grp_pjc_cat_hotel.py X cat_hotel.sql
grp_pjc_cat_hotel.py > ...
1 import tkinter as tk
2 from tkinter import ttk
3 from tkinter import *
4 from tkinter import messagebox
5 import mysql.connector
6
7 mydb = mysql.connector.connect(
8     host = "localhost",
9     user = "root",
10    password = "",
11    database = "cat_hotel")
12
13 mycursor = mydb.cursor()
14
15 #mainwindow
16 window = tk.Tk()
17 window.geometry("870x560")
18 window['background'] = '#836953'
19 window.title("Cat Owner Registration")
20
```

```
15 #mainwindow
16 window = tk.Tk()
17 window.geometry("870x560")
18 window['background'] = '#836953'
19 window.title("Cat Owner Registration")
20
21 labelmenu = tk.Label(window, text="\nWelcome MATE !!", font = ('Comic Sans MS bold', 25), fg="white", bg="#836953")
22 labelmenu.pack(padx=20, pady=1)
23
24 labelmenu = tk.Label(window, text="* Make sure to insert the data carefully, \n so that there will not have any mistake *", font = ('Comic Sans MS bold', 18), fg="white", bg="#836953")
25 labelmenu.pack(padx=20, pady=1)
26
27 labelmenu = tk.Label(window, text="* Only boarding date, grooming package, and haricut can be edited *", font = ('Comic Sans MS bold', 18), fg="white", bg="#836953")
28 labelmenu.pack(padx=20, pady=1)
29
30 labelmenu = tk.Label(window, text="* Please treat our customer carefully and helpfull \n so that we get a good review *", font = ('Comic Sans MS bold', 18), fg="white", bg="#836953")
31 labelmenu.pack(padx=20, pady=1)
32
33 labelmenu = tk.Label(window, text="\n ~~Happy Working~~", font = ('Comic Sans MS bold', 18), fg="white", bg="#836953")
34 labelmenu.pack(padx=20, pady=1)
35
36 labelmenu = tk.Label(window, text=" VVVVV", font = ('Comic Sans MS bold', 18), fg="white", bg="#836953")
37 labelmenu.place(x=380, y=470)
38
```

```

38
39 def register_page():
40
41     def collect_data():
42
43         accepted = accept_var.get()
44
45         suite_type = suite_type_combobox.get()
46         board_type = board_type_combobox.get()
47         board_day = int(board_day_combobox.get())
48         medicine_type = medicine_type_combobox.get()
49         treatment_session = int(treatment_session_combobox.get())
50         grooming_package = grooming_package_combobox.get()
51         grooming_cut = grooming_cut_combobox.get()
52
53         prices = {"Rabies FVRCP" : 90, "Self Check-in & out" : 0, "lion" : 35,
54                 "FelV VACCINE" : 85, "Self Check-in & Drop" : 15, "Asian Lion" : 35,
55                 "FHV-1" : 96, "Pickup & Self Check-out": 15, "Natural Look": 20,
56                 "ITRACONAZOLE" : 55, "Pickup & Drop" : 30, "Panther" : 35,
57                 "TERBINAFINE" : 61, "Basic" : 80, "Belly Shave" : 20,
58                 "FLUCONAZOLE" : 58, "Full" : 150, "Butt Shave" : 20,
59                 "Gold" : 50, "Deluxe" : 200, "None" : 0,
60                 "Platinum" : 100, "Teddy Bear" : 40,
61                 "Diamond" : 150, "tiger" : 35,}
62
63         total_suite = (prices[suite_type] * board_day + prices[board_type])
64         total_treatment = (prices[medicine_type] * treatment_session)
65         total_grooming = (prices[grooming_package] + prices[grooming_cut])
66         total_prices = total_suite + total_treatment + total_grooming
67
68
69         output_total = tk.Label(output_frame, text="", fg="white", bg="#836953")
70         output_total.grid(row=0, column=1)
71         output_total.config(text=f"RM{total_prices}")

```

```

72
73     if accepted == "Accepted":
74         id = owner_id_entry.get()
75         name = owner_name_entry.get()
76         dbirth = d_b_entry.get()
77         mbirth = m_b_entry.get()
78         ybirth = y_b_entry.get()
79         address = owner_address_entry.get()
80         phone = owner_phone_entry.get()
81         email = owner_email_address_entry.get()
82

```

```

83
84         if name and address and phone and email:
85             cat = cat_name_entry.get()
86             cat_age = cat_age_combobox.get()
87             cat_breed = cat_breed_combobox.get()
88             cat_noted = cat_noted_entry.get()
89             board_date = board_date_entry.get()
90             treatment_type = treatment_type_combobox.get()
91
92             print("Name: ", name)
93             print("Birth Date:", dbirth, "/", mbirth, "/", ybirth)
94             print("Address:", address)
95             print("Contact No:", phone)
96             print("Email:", email)
97             print(".....")
98             print("Cat Name:", cat)
99             print("Age:", cat_age)
100            print("Breed:", cat_breed)
101            print("Notes:", cat_noted)
102            print("Suite Type:", suite_type)
103            print("Board Type:", board_type)
104            print("Board Day:", board_day,)
105            print("Board Date:", board_date)
106            print("Treatment Type:", treatment_type)
107            print("Treatment Session:", treatment_session)
108            print("Medicine Type:", medicine_type)
109            print("Grooming Package:", grooming_package)
110            print("Grooming Cut:", grooming_cut)
111            print("-----")
112        else:
113            tk.messagebox.showwarning(title= "Error", message= "There are items that require your attention!")
114    else:
115        tk.messagebox.showwarning(title= "Error", message= "You have not accepted the terms!")
116

```

```

117
118    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,%s)"
119    val = (id, name, address, phone, email, cat, cat_age, cat_breed, cat_noted)
120    sql2 = "INSERT INTO boarding (SUITE,BOARDING_TYPE,DAY,DAY_CHECK_IN,TOTAL) VALUES (%s,%s,%s,%s,%s)"
121    val2 = (suite_type, board_type, board_day, board_date, total_suite)
122    sql3 = "INSERT INTO treatment (TREATMENT_TYPE, SESSION,MEDICINE_TYPE,TOTAL) VALUES (%s,%s,%s,%s)"
123    val3 = (treatment_type, treatment_session, medicine_type, total_treatment)
124    sql4 = "INSERT INTO grooming (PACKAGE,HAIRCUT,TOTAL) VALUES (%s,%s,%s)"
125    val4 = (grooming_package, grooming_cut, total_grooming)
126
127    mycursor.execute(sql, val)
128    mycursor.execute(sql2, val2)
129    mycursor.execute(sql3, val3)
130    mycursor.execute(sql4, val4)
131    mydb.commit()
132
133
134    def view_data():
135        mydb = mysql.connector.connect(
136            host = "localhost",
137            user = "root",
138            password = "",
139            database = "cat_hotel")
140
141        mycursor = mydb.cursor()
142        mycursor.execute("SELECT * FROM registration")
143
144        result = mycursor.fetchall()
145

```

```

146     print_record = ''
147     if result:
148         latest_record = result[-1]
149         print_record = f"> {latest_record[0]} \t {latest_record[1]}"
150     else:
151         print_record = "No records Available"
152
153     view_record = tk.Label(output_frame, text=print_record, fg="white", bg="#836953")
154     view_record.grid(row=0, column=3)
155
156
157     mycursor.close()
158     mydb.close()
159
160 def edit_data ():
161     boarddate = board_date_entry.get()
162     groomingpackage = grooming_package_combobox.get()
163     groomingcut = grooming_cut_combobox.get()
164
165     mydb = mysql.connector.connect(
166         host = "localhost",
167         user = "root",
168         password="",
169         database="cat_hotel"
170     )
171     mycursor = mydb.cursor()
172
173     mycursor.execute("SELECT * FROM boarding")
174     boarding_records = mycursor.fetchone()
175     mycursor.fetchall()
176     mycursor.execute("SELECT * FROM grooming")
177     grooming_records = mycursor.fetchone()
178     mycursor.fetchall()
179

```

```

179
180 edit_one = ''
181 edit_two = ''
182 if boarding_records is not None:
183     sql_boarding = "UPDATE boarding SET DAY_CHECK_IN = %s WHERE DAY_CHECK_IN = %s"
184     val_boarding = (boarddate, boarding_records[3])
185     mycursor.execute(sql_boarding, val_boarding)
186     mydb.commit()
187 if grooming_records is not None:
188     sql_grooming = "UPDATE grooming SET PACKAGE = %s, HAIRCUT = %s WHERE PACKAGE = %s AND HAIRCUT = %s"
189     val_grooming = (groomingpackage, groomingcut, grooming_records[0], grooming_records[1])
190     mycursor.execute(sql_grooming, val_grooming)
191     mydb.commit()
192     edit_one = "Groom ✓ Board ✓"
193 else:
194     edit_two = "No updated"
195
196 edit_record = tk.Label(output_frame, text=edit_one, fg="white", bg="#836953")
197 edit_record.grid(row=0, column=5)
198 edit_record = tk.Label(output_frame, text=edit_two, fg="white", bg="#836953")
199 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

```

```

239 #register window
240 root = tk.Toplevel(window)
241 root.geometry("870x560")
242 root['background'] = '#836953'
243 root.title("Cat Owner Registration")
244
245 label = tk.Label(root, text = "Customer Registration", font = ('Comic Sans MS bold', 18), fg="white", bg="#836953")
246 label.pack(padx=20, pady=1)
247
248 out_frame = tk.Frame(root, bg="#836953")
249 out_frame.pack(padx=20, pady=1)
250
251 frame = tk.LabelFrame(out_frame, bg="#836953")
252 frame.grid(row=0, column=0)
253
254 #owner
255 cat_owner_frame = tk.LabelFrame(frame, text = "Personal Detail", fg="white", bg="#836953")
256 cat_owner_frame.grid(row=0, column=0, sticky="News", padx=1, pady=1)
257
258 owner_id = tk.Label(cat_owner_frame, text="Id :", fg="white", bg="#836953")
259 owner_id.grid(row=0, column=0)
260 owner_id_entry = tk.Entry(cat_owner_frame, width=23)
261 owner_id_entry.grid(row=0, column=1)
262
263 owner_name_label = tk.Label(cat_owner_frame, text="Name :", fg="white", bg="#836953")
264 owner_name_label.grid(row=1, column=0)
265 owner_name_entry = tk.Entry(cat_owner_frame, width=23)
266 owner_name_entry.grid(row=1, column=1)
267
268 owner_birth_date = tk.Label(cat_owner_frame, text="      Birth Date :      ", fg="white", bg="#836953")
269 owner_birth_date.grid(row=2, column=0)

```

```

grp_pjc_cat_hotel.py > register_page > delete_data
271 d_b_entry = ttk.Combobox(cat_owner_frame, values=["1","2","3","4","5","6","7","8","9","10",
272                                                  "11","12","13","14","15","16","17","18","19","20",
273                                                  "21","22","23","24","25","26","27","28","29","30","31"])
274 d_b_entry.grid(row=2, column=1)
275 d_b_entry.set("Date")
276 d_b_entry["state"] = 'readonly'
277
278 m_b_entry = ttk.Combobox(cat_owner_frame, values=["01","02","03","04","05","06",
279                                                  "07","08","09","10","11","12"])
280 m_b_entry.grid(row=3, column=1)
281 m_b_entry.set("Month")
282 m_b_entry["state"] = 'readonly'
283
284 y_b_entry = ttk.Combobox(cat_owner_frame, values=list(range(1960, 2200)))
285 y_b_entry.grid(row=4, column=1)
286 y_b_entry.set("Year")
287 y_b_entry["state"] = 'readonly'
288
289 for widget in cat_owner_frame.winfo_children():
290     widget.grid_configure(padx= 10, pady=5)
291
292 cont = tk.LabelFrame(frame, text="-", fg="white", bg="#836953")
293 cont.grid(row=0, column=1, sticky="News", padx=1, pady=1)
294
295 owner_address_label = tk.Label(cont, text="      Address :      ", fg="white", bg="#836953")
296 owner_address_label.grid(row=5, column=0)
297 owner_address_entry = tk.Entry(cont, width=23)
298 owner_address_entry.grid(row=5, column=1)
299
300 owner_phone_label = tk.Label(cont, text="Contact No :", fg="white", bg="#836953")
301 owner_phone_label.grid(row=6, column=0)
302 owner_phone_entry = tk.Entry(cont, width=23)
303 owner_phone_entry.grid(row=6, column=1)
304

```

```

304 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

```

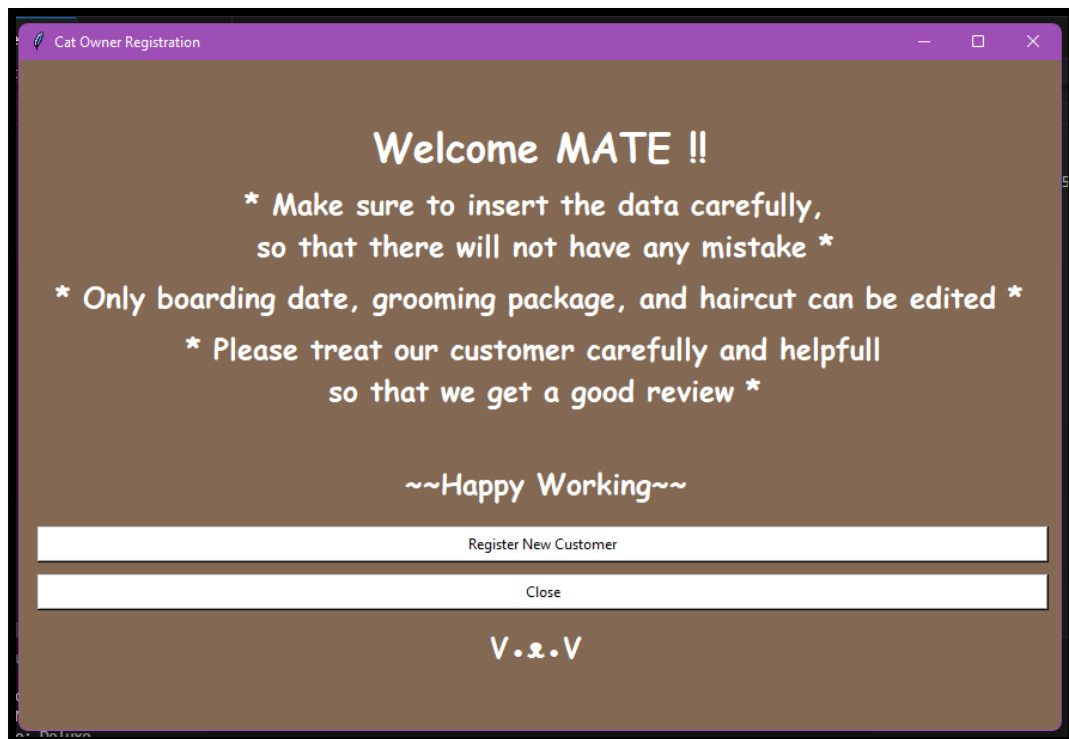


```

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)



The screenshot shows a window titled "Cat Owner Registration" with a brown background. The title "Customer Registration" is centered at the top. The form is divided into several sections:

- Personal Detail:** Includes fields for Id, Name, Birth Date (with Date, Month, and Year dropdowns), Address, Contact No, and Email Address.
- Cat Details:** Includes fields for Name, Age (dropdown), Breed (dropdown), and Notes.
- Boarding:** Includes fields for Suite Type (dropdown), Boarding Type (dropdown), Day (dropdown), and Date Check In (dd/mm/yyyy).
- Treatment:** Includes fields for Treatment Type (dropdown), Session (dropdown), and Medicine Type (dropdown).
- Grooming:** Includes fields for Package (dropdown) and Cat Haircut (dropdown).

Below the form sections, there is a "Terms & Condition" section with a checkbox and the text "I accept the terms and condition." Below that is a "Notes" section with a text area. At the bottom, there are four buttons: "Enter", "View", "Edit", and "Delete Last Record". A "Close" button is located at the very bottom.

7.0 SNAPSHOT OF DATABASE (XAMPP)

The screenshot displays the phpMyAdmin web interface. On the left, a sidebar shows the database structure with 'cat_hotel' selected. The main area shows the 'Structure' tab for the 'cat_hotel' database. A table with 4 columns (Table, Action, Rows, Type, Collation, Size, Overhead) lists the tables: boarding, grooming, registration, and treatment. Each table has 5 rows and is using the InnoDB engine with utf8mb4_general_ci collation. The total size is 64.0 KiB. Below the table list, there are options to 'Check all' and 'With selected:'. At the bottom, there is a 'Create new table' button and a form to create a new table with a name and number of columns (4).

| Table | Action | Rows | Type | Collation | Size | Overhead |
|---------------------------------------|------------|-----------|---------------|---------------------------|-----------------|------------|
| <input type="checkbox"/> boarding | | 5 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| <input type="checkbox"/> grooming | | 5 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| <input type="checkbox"/> registration | | 5 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| <input type="checkbox"/> treatment | | 5 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| 4 tables | Sum | 20 | InnoDB | utf8mb4_general_ci | 64.0 KiB | 0 B |

Figure 7.0: DATABASE CAT HOTEL REGISTRATION

7.1 REGISTRATION DATABASE

Server: 127.0.0.1 » Database: cat_hotel » Table: registration

SELECT * FROM `registration`

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

☐ Show all | Number of rows: 25 | Filter rows: Search this table

Extra options

| 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 Sofiyah Azhar | Kampung Tepi Laut | 213584697 | sofiyahazhar@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 | - |

☐ Show all | Number of rows: 25 | Filter rows: Search this table

Figure 7.1: BROWSE

Table structure | Relation view

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

☐ Check all | With selected: Browse Change Drop Primary Unique Index Spatial Fulltext

Add to central columns Remove from central columns

Print Propose table structure Track table Move columns Normalise

Add 1 column(s) after NOTES Go

Figure 7.2: STRUCTURE

7.2 BOARDING DATABASE

The BROWSE tab displays the results of a SQL query: `SELECT * FROM `boarding``. The interface includes a toolbar with options like Browse, Structure, SQL, Search, Insert, Export, Import, Privileges, and Operations. Below the query, there are controls for Profiling, Show all, Number of rows (set to 25), and a Filter rows search box. The query results are shown in a table with the following data:

| SUITE | BOARDING_TYPE | DAY | DAY_CHECK_IN | TOTAL |
|----------|----------------------|-----|--------------|-------|
| Diamond | Pickup & Drop | 10 | 16 | 1530 |
| Platinum | Self Check-in & Drop | 5 | 5 | 515 |
| Diamond | Pickup & Drop | 15 | 14 | 2280 |
| Diamond | Pickup & Drop | 20 | 1 | 3030 |
| Diamond | Pickup & Drop | 31 | 16 | 4680 |

At the bottom, there are buttons for Print, Copy to clipboard, Export, Display chart, and Create view.

Figure 7.4: BROWSE

The STRUCTURE tab displays the table structure of the 'boarding' table. The interface includes a toolbar with options like Browse, Structure, SQL, Search, Insert, Export, Import, Privileges, Operations, and Tracking. Below the toolbar, there are tabs for Table structure and Relation view. The table structure is shown in a table with the following columns:

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---------------|-------------|--------------------|------------|------|---------|----------|-------|------------------|
| 1 | SUITE | varchar(30) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |
| 2 | BOARDING_TYPE | varchar(30) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |
| 3 | DAY | int(31) | | | Yes | NULL | | | Change Drop More |
| 4 | DAY_CHECK_IN | int(100) | | | Yes | NULL | | | Change Drop More |
| 5 | TOTAL | int(255) | | | Yes | NULL | | | Change Drop More |

Below the table, there are controls for Check all, With selected, Browse, Change, Drop, Primary, Unique, Index, Spatial, and Fulltext. There are also buttons for Add to central columns and Remove from central columns. At the bottom, there are buttons for Print, Propose table structure, Track table, Move columns, and Normalise. There is also a section for adding a new column with a text input for the column name, a dropdown for the position (after TOTAL), and a Go button.

Figure 7.5: STRUCTURE

7.3 TREATMENT DATABASE

The screenshot shows the 'BROWSE' tab of a database management tool. At the top, there is a toolbar with icons for Browse, Structure, SQL, Search, Insert, Export, Import, and Privileges. Below the toolbar, a SQL query is displayed: `SELECT * FROM `treatment``. Under the query, there are links for Profiling, Edit inline, Edit, Explain SQL, Create PHP code, and Refresh. A control bar allows showing all rows, setting the number of rows to 25, and filtering rows with a search box. An 'Extra options' button is also present. The main area displays a table with the following data:

| TREATMENT_TYPE | SESSION | MEDICINE_TYPE | TOTAL |
|----------------|---------|---------------|-------|
| vaccine | 2 | None | 0 |
| vaccine | 1 | Rabies FVRCP | 90 |
| vaccine | 3 | None | 0 |
| None | 0 | None | 0 |
| vaccine | 2 | None | 0 |

Below the table, there is another control bar with 'Show all', 'Number of rows: 25', and a 'Filter rows' search box. At the bottom, a 'Query results operations' bar includes icons for Print, Copy to clipboard, Export, Display chart, and Create view.

Figure 7.6: BROWSE

The screenshot shows the 'STRUCTURE' tab of the same database management tool. The toolbar includes an additional 'Operations' icon. Below the toolbar, there are tabs for 'Table structure' and 'Relation view'. The 'Table structure' tab is active, displaying a table with the following columns:

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|----------------|-------------|--------------------|------------|------|---------|----------|-------|------------------|
| 1 | TREATMENT_TYPE | varchar(30) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |
| 2 | SESSION | int(10) | | | Yes | NULL | | | Change Drop More |
| 3 | MEDICINE_TYPE | varchar(50) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |
| 4 | TOTAL | int(255) | | | Yes | NULL | | | Change Drop More |

Below the table, there is a control bar with 'Check all', 'With selected:', 'Browse', 'Change', 'Drop', 'Primary', 'Unique', 'Index', and 'Spatial' options. There are also 'Add to central columns' and 'Remove from central columns' buttons. At the bottom, there is a 'Print' button, a 'Propose table structure' button, a 'Track table' button, a 'Move columns' button, and a 'Normalise' button. A form at the bottom allows adding a new column: 'Add 1 column(s) after TOTAL' with a 'Go' button.

Figure 7.7: STRUCTURE

7.4 GROOMING DATABASE

The screenshot shows the 'BROWSE' tab of a database management interface. At the top, there is a toolbar with icons for Browse, Structure, SQL, Search, Insert, Export, Import, and Privileges. Below the toolbar, a SQL query is displayed: `SELECT * FROM `grooming``. Underneath the query, there are links for Profiling, Edit inline, Edit, Explain SQL, Create PHP code, and Refresh. A control bar includes a 'Show all' checkbox, a 'Number of rows' dropdown set to 25, and a 'Filter rows' search box with the placeholder text 'Search this table'. An 'Extra options' button is located below the control bar. The main area displays a table with three columns: PACKAGE, HAIRCUT, and TOTAL. The table contains five rows of data. At the bottom, there is another control bar identical to the one above.

| PACKAGE | HAIRCUT | TOTAL |
|---------|--------------|-------|
| Deluxe | None | 200 |
| Basic | Belly Shave | 100 |
| Deluxe | Teddy Bear | 240 |
| Full | Natural Look | 170 |
| Deluxe | None | 200 |

Figure 7.8: BROWSE

The screenshot shows the 'STRUCTURE' tab of the same database management interface. The toolbar now includes an 'Operations' icon. Below the toolbar, there are tabs for 'Table structure' and 'Relation view'. The 'Table structure' tab is active, displaying a table with columns: #, Name, Type, Collation, Attributes, Null, Default, Comments, Extra, and Action. The table lists three columns: PACKAGE (varchar(10), utf8mb4_general_ci, Yes, NULL), HAIRCUT (varchar(30), utf8mb4_general_ci, Yes, NULL), and TOTAL (int(255), Yes, NULL). Each row has 'Change', 'Drop', and 'More' action links. Below the table, there are checkboxes for 'Check all' and 'With selected:', followed by icons for Browse, Change, Drop, Primary, Unique, Index, and Spatial. There are also buttons for 'Add to central columns' and 'Remove from central columns'. A section for 'Indexes' is visible at the bottom, with a message 'No index defined!'. Above the 'Indexes' section, there are buttons for Print, Propose table structure, Track table, Move columns, and Normalise. At the bottom, there is an 'Add' button, a text input field containing '1', a 'column(s)' label, a dropdown menu set to 'after TOTAL', and a 'Go' button.

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---------|-------------|--------------------|------------|------|---------|----------|-------|------------------|
| 1 | PACKAGE | varchar(10) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |
| 2 | HAIRCUT | varchar(30) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |
| 3 | TOTAL | int(255) | | | Yes | NULL | | | Change Drop More |

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.