



UNIVERSITI TEKNOLOGI MARA

KEDAH BRANCH

**SCHOOL OF INFORMATION SCIENCE COLLEGE OF COMPUTING,
INFORMATICS AND MATHEMATICS**

Diploma in Library Informatics

(IM 144)

PROGRAMMING FOR LIBRARIES

(IML 208)

‘REPORT ABOUT TRAIN TICKET RECEIPT SYSTEM’

PREPARED BY:

ZAHIRATUL SYAZWINA BINTI ZAIDI

2022660998

CLASS:

KCDIM1443F

PREPARED FOR:

SIR AIRUL SHAZWAN BIN NORSHAHIMI

SUBMISSION DATE

4th JANUARY 2024

‘REPORT ABOUT TRAIN TICKET RECEIPT SYSTEM’

PREPARED BY:

ZAHIRATUL SYAZWINA BINTI ZAIDI

2022660998

CLASS:

KCDIM1443F

Diploma in Library Informatics

(IM 144)

SCHOOL OF INFORMATION SCIENCE

COLLEGE OF COMPUTING, INFORMATICS AND MATHEMATICS

UNIVERSITI TEKNOLOGI MARA (UITM)

KEDAH BRANCH

ACKNOWLEDGEMENT

For starters, I would like to express my special thanks and gratitude to my lecturer, Sir Airul Shazwan Bin Norshahimi, who became my guide for this assignment. As my advisor, he provided me with his input on what could be added to the project. This project could not have been done and perfected without his help, and he also gave me clear and concise instructions to make it easier for me to proceed with this report smoothly. He also assisted me with encouragement and guidance to succeed in this assignment.

My deepest appreciation then goes to my parents, who inspire me to persevere in the face of adversity. They have been the ones to constantly check up on me to make sure that I'm doing well. I cannot forget their support, and I hope I will receive the same love and guidance in my future endeavours as well.

Finally, I am thankful to my friends for sharing their thoughts with me in order to help me produce great innovation. This task's outcome could not be achieved otherwise. My sincere appreciation is extended to everyone who provided me with direct or indirect assistance.

TABLE OF CONTENT

ACKNOWLEDGEMENT.....	i
1.0 INTRODUCTION.....	1
2.0 FLOW CHART	2
3.0 DATABASE.....	3
4.0 CODING	3
5.0 GUI.....	6
6.0 CONCLUSION	8

1.0 INTRODUCTION

This assignment includes using Python to create a train ticket receipt system. The system allows users to input details like username, IC number, date, time, coach, seat, from, and to. In order to make it easier for users to know the price of their ticket.

Firstly, users need to fill in their username. Then, they have to enter their IC number with 12 digits. After that, they must enter the date using the yyyy-mm-dd format so there won't be an error. Next, the user needs to enter the time provided in the system. After that, the user has to choose the coach between A-F. Next is the seat which the user can choose. The last one is that the user needs to fill in the from and to of their destination. After completing and filling in everything, the user can click on 'Generate Receipt', and the receipt of the train ticket will appear.

A train ticket receipt system is designed to assist users with figuring out the price of their ticket. The price of the ticket in coach B-F from 8 a.m. to 11 a.m. will be RM 50.00, while the price from 12 p.m. to 10 p.m. will be RM 30.00. However, in coach A which is the VIP section, all of the tickets will be RM 50.00. Additionally, the user may view the details of the ticket they have ordered in addition to the ticket price, which will make it simpler for them to complete the payment.

Last but not least, the information of the user that has been filled out in the system will appear on phpMyAdmin. We can see all the user information there, as we already imported our SQL file there.

2.0 FLOW CHART

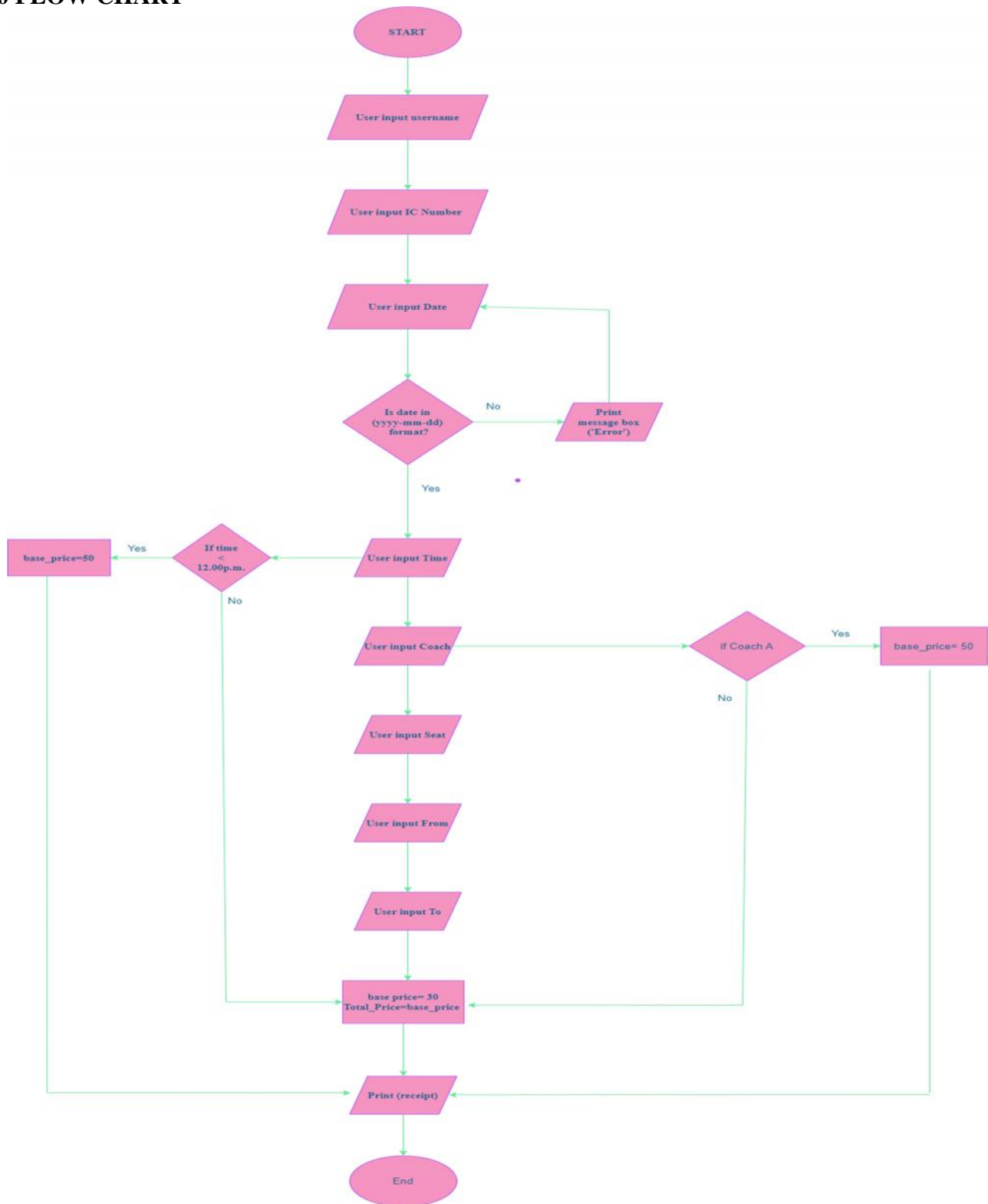
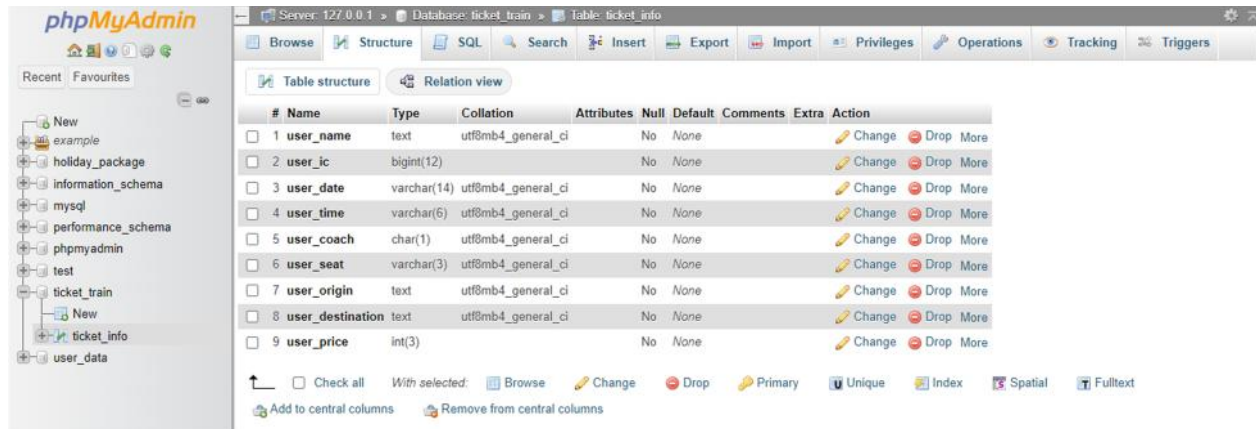


Figure 1 Flow chart

3.0 DATABASE

This is the structure of ticket info.



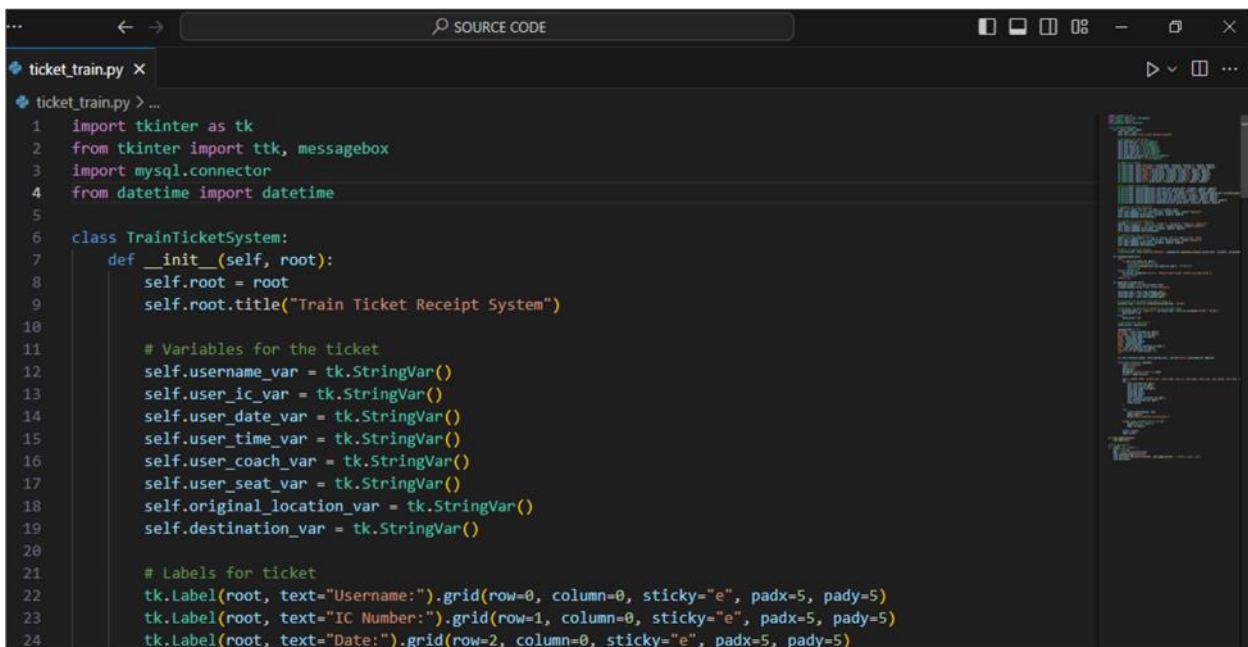
The screenshot shows the phpMyAdmin interface with the 'ticket_info' table selected. The table structure is displayed in a table format with columns for #, Name, Type, Collation, Attributes, Null, Default, Comments, Extra, and Action.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	user_name	text	utf8mb4_general_ci		No	None			Change Drop More
2	user_ic	bigint(12)			No	None			Change Drop More
3	user_date	varchar(14)	utf8mb4_general_ci		No	None			Change Drop More
4	user_time	varchar(6)	utf8mb4_general_ci		No	None			Change Drop More
5	user_coach	char(1)	utf8mb4_general_ci		No	None			Change Drop More
6	user_seat	varchar(3)	utf8mb4_general_ci		No	None			Change Drop More
7	user_origin	text	utf8mb4_general_ci		No	None			Change Drop More
8	user_destination	text	utf8mb4_general_ci		No	None			Change Drop More
9	user_price	int(3)			No	None			Change Drop More

Figure 2 database

4.0 CODING

This is all of the coding code for the ticket train from Visual Studio Code.



```
ticket_train.py X
ticket_train.py > ...
1  import tkinter as tk
2  from tkinter import ttk, messagebox
3  import mysql.connector
4  from datetime import datetime
5
6  class TrainTicketSystem:
7      def __init__(self, root):
8          self.root = root
9          self.root.title("Train Ticket Receipt System")
10
11         # Variables for the ticket
12         self.username_var = tk.StringVar()
13         self.user_ic_var = tk.StringVar()
14         self.user_date_var = tk.StringVar()
15         self.user_time_var = tk.StringVar()
16         self.user_coach_var = tk.StringVar()
17         self.user_seat_var = tk.StringVar()
18         self.original_location_var = tk.StringVar()
19         self.destination_var = tk.StringVar()
20
21         # Labels for ticket
22         tk.Label(root, text="Username:").grid(row=0, column=0, sticky="e", padx=5, pady=5)
23         tk.Label(root, text="IC Number:").grid(row=1, column=0, sticky="e", padx=5, pady=5)
24         tk.Label(root, text="Date:").grid(row=2, column=0, sticky="e", padx=5, pady=5)
```

Figure 3 Coding 1

```

... SOURCE CODE
ticket_train.py X
ticket_train.py > ...
23 tk.Label(root, text="IC Number:").grid(row=1, column=0, sticky="e", padx=5, pady=5)
24 tk.Label(root, text="Date:").grid(row=2, column=0, sticky="e", padx=5, pady=5)
25 tk.Label(root, text="Time:").grid(row=3, column=0, sticky="e", padx=5, pady=5)
26 tk.Label(root, text="Coach:").grid(row=4, column=0, sticky="e", padx=5, pady=5)
27 tk.Label(root, text="Seat:").grid(row=5, column=0, sticky="e", padx=5, pady=5)
28 tk.Label(root, text="From:").grid(row=6, column=0, sticky="e", padx=5, pady=5)
29 tk.Label(root, text="To:").grid(row=7, column=0, sticky="e", padx=5, pady=5)
30
31 # Entry widgets for ticket
32 tk.Entry(root, textvariable=self.user_ic_var).grid(row=0, column=1, padx=5, pady=5)
33 tk.Entry(root, textvariable=self.user_date_var, validate="focusout", validatecommand=self.validate_date).grid(row=1, column=1, padx=5, pady=5)
34 tk.Entry(root, textvariable=self.user_time_var).grid(row=3, column=1, padx=5, pady=5)
35 tk.Entry(root, textvariable=self.user_seat_var).grid(row=5, column=1, padx=5, pady=5)
36 tk.Entry(root, textvariable=self.original_location_var).grid(row=6, column=1, padx=5, pady=5)
37 tk.Entry(root, textvariable=self.destination_var).grid(row=7, column=1, padx=5, pady=5)
38
39 # Combobox for time selection
40 times = [f"{hour:02d}:00" for hour in range(8, 23)]
41 self.time_combobox = ttk.Combobox(root, values=times, state="readonly")
42 self.time_combobox.grid(row=3, column=1, padx=5, pady=5)
43 self.time_combobox.set(times[0])
44
45

```

Figure 4 Coding 2

```

... SOURCE CODE
ticket_train.py X
ticket_train.py > TrainTicketSystem > __init__
45
46 # Combobox for coach selection
47 coaches = ["Coach A", "Coach B", "Coach C", "Coach D", "Coach E", "Coach F"]
48 self.coach_combobox = ttk.Combobox(root, values=coaches, state="readonly")
49 self.coach_combobox.grid(row=4, column=1, padx=5, pady=5)
50 self.coach_combobox.set(coaches[0])
51
52 # Combobox for seat selection
53 seats = [f"{row}{seat}" for row in range(1, 17) for seat in ["A", "B"]]
54 self.seat_combobox = ttk.Combobox(root, values=seats, state="readonly")
55 self.seat_combobox.grid(row=5, column=1, padx=5, pady=5)
56 self.seat_combobox.set(seats[0])
57
58 # Button to generate receipt
59 tk.Button(root, text="Generate Receipt", command=self.generate_receipt).grid(row=8, column=0, columnspan=2)
60
61 def validate_date(self):
62     try:
63         if self.user_date_var.get():
64             # Validate the date format
65             datetime.strptime(self.user_date_var.get(), "%Y-%m-%d")
66             return True
67     except ValueError:
68         messagebox.showerror("Error", "Invalid date format. Please use yyyy-mm-dd.")
69         return False

```

Figure 5 Coding 3


```

...  ← →  SOURCE CODE
ticket_train.py X
ticket_train.py > TrainTicketSystem > _init_
69         return False
70     return True
71
72     def generate_receipt(self):
73         receipt_window = tk.Toplevel(self.root)
74         receipt_window.title("Train Ticket Receipt")
75
76         selected_time = self.time_combobox.get()
77         selected_coach = self.coach_combobox.get()
78         selected_seat = self.seat_combobox.get()
79
80         # Calculate the purchase_time before using it
81         purchase_time = datetime.strptime(selected_time, "%H:%M")
82
83         # Determine the base price based on the selected coach
84         if selected_coach == "Coach A" or purchase_time < datetime.strptime("12:00", "%H:%M"):
85             base_price = 50
86         else:
87             base_price = 30
88
89         # Calculate the total price
90         total_price = base_price
91
92         receipt_text = f"""
93         Username: {self.username_var.get()}
94         IC Number: {self.user_ic_var.get()}

```

Figure 6 Coding 4

```

...  ← →  SOURCE CODE
ticket_train.py X
ticket_train.py > TrainTicketSystem > _init_
93         Username: {self.username_var.get()}
94         IC Number: {self.user_ic_var.get()}
95         Date: {self.user_date_var.get()}
96         Time: {selected_time}
97         Coach: {selected_coach}
98         Seat: {selected_seat}
99         From: {self.original_location_var.get()}
100        To: {self.destination_var.get()}
101        Total Price: RM {total_price:.2f}
102        """
103
104        tk.Label(receipt_window, text=receipt_text, justify="left").pack(padx=10, pady=10)
105
106        with mysql.connector.connect(
107            host="localhost",
108            user="root",
109            password="",
110            database="ticket_train") as mydb:
111            cursor = mydb.cursor()
112
113            sql = "INSERT INTO `ticket_info` (user_name, user_ic, user_date, user_time, user_coach, user_seat, user_price)"
114            val = (
115                self.username_var.get(),
116                self.user_ic_var.get(),
117                self.user_date_var.get(),

```

Figure 7 Coding 5

```

117         self.user_date_var.get(),
118         selected_time,
119         selected_coach,
120         selected_seat,
121         self.original_location_var.get(),
122         self.destination_var.get(),
123         total_price,
124     )
125
126     try:
127         cursor.execute(sql, val)
128         mydb.commit()
129         print("Data inserted successfully!")
130
131     except mysql.connector.Error as err:
132         print(f"Error: {err}")
133         mydb.rollback()
134
135     cursor.close()
136     mydb.close()
137
138     def quit_application():
139         root.destroy()
140

```

Figure 8 Coding 6

```

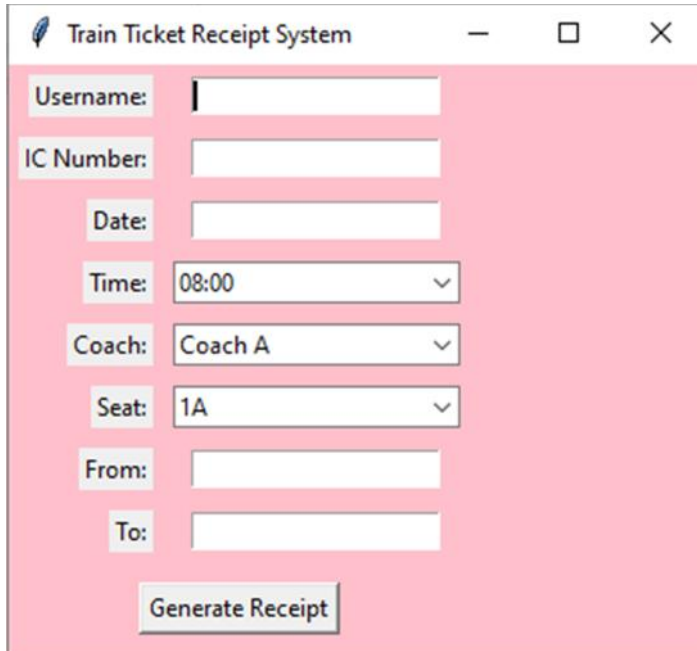
133         mydb.rollback()
134
135     cursor.close()
136     mydb.close()
137
138     def quit_application():
139         root.destroy()
140
141     # Main application
142     if __name__ == "__main__":
143         root = tk.Tk()
144         app = TrainTicketSystem(root)
145         root.configure(bg="#ffc0cb")
146         root.protocol("WM_DELETE_WINDOW", quit_application) # Handle window close
147         root.mainloop()
148

```

Figure 9 Coding 7

5.0 GUI

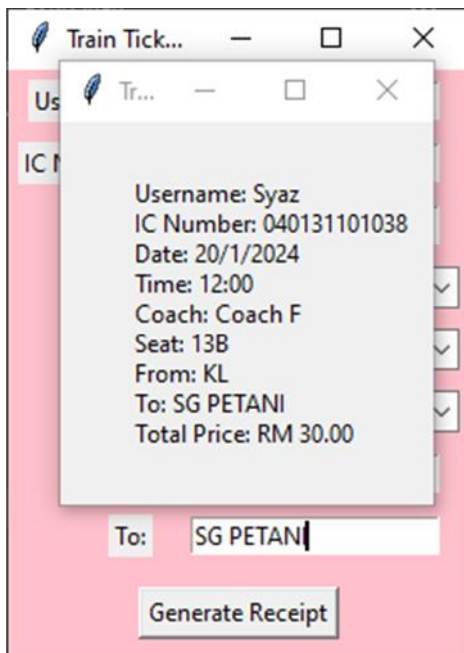
This is the interface of the GUI that will pop out. The user has to insert the date using yyyy-mm-dd format. After the user is done filling out all of it, they have to click on Generate Receipt to get their receipt.



The screenshot shows a window titled "Train Ticket Receipt System" with a pink background. It contains several input fields and dropdown menus arranged vertically: "Username:" with a text box, "IC Number:" with a text box, "Date:" with a text box, "Time:" with a dropdown menu showing "08:00", "Coach:" with a dropdown menu showing "Coach A", "Seat:" with a dropdown menu showing "1A", "From:" with a text box, and "To:" with a text box. At the bottom is a button labeled "Generate Receipt".

Figure 10 GUI interface before generate receipt

This is an example that will be shown. It will show the user the total price.



The screenshot shows the same "Train Ticket Receipt System" window, but with a modal dialog box overlaid in the center. The dialog box has a white background and a title bar that says "Train Tick...". It displays the following information: "Username: Syaz", "IC Number: 040131101038", "Date: 20/1/2024", "Time: 12:00", "Coach: Coach F", "Seat: 13B", "From: KL", "To: SG PETANI", and "Total Price: RM 30.00". Below the dialog box, the "To:" text box now contains "SG PETANI" and the "Generate Receipt" button is visible.

Figure 11 GUI interface after generate receipt

6.0 CONCLUSION

Overall, I've learned so much through the process of designing and developing a train ticket receipt system, thanks to this assignment. It will be easier for me the next time I want to make it since I've been doing it.

I've discovered that it is a bit challenging to do the coding, because I have to type every word precisely so that it doesn't say "Error". It is suggested against doing this when you are tired in order to avoid making any easy mistakes.

Moreover, I need to gain the opinions of others in order to gather more inspiration from them for the database part. Then, I may learn more from their viewpoint, which will help me come up with great ideas.

In conclusion, this assignment has taught me a lot to make a simple computer interface that are useful. Students like myself can improve their understanding of the material they study by doing such things.