**MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR**

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

**Skill Based Mini Project Report**

**on**

# "Unified Travelling and Transport System"

**Submitted By:**

**Harsh Shrivastava (0901CS211049)**

**Faculty Mentor:**

**Dr. Ranjeet Kumar Singh**

**Prof. Hemlata Arya**

**Dr. Devesh Kumar Lal**

Submitted to:

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR - 474005 (MP) est. 1957

JAN-JUNE 2023

## ABSTRACT

The Unified Travelling and Transport System (UTTS) is a software application that provides a unified platform for booking and managing travelling and transportation services. The system is built using Python programming language and employs a database management system to store and manage information related to travelling and transportation services.

The UTTS offers a seamless booking experience to its users, allowing them to search for and book transportation services from a single platform. The system also provides real-time information on the availability of different transportation services and their schedules.

The database management system used in the UTTS is designed to handle large amounts of data related to different transportation services. It offers efficient data storage and retrieval capabilities, ensuring that the system can handle a large number of users and transactions.

The UTTS also incorporates advanced security features to protect user data and prevent unauthorized access. The system ensures that user information is encrypted and stored securely, and access to sensitive data is restricted only to authorized personnel.

Overall, the UTTS offers a reliable and efficient platform for booking and managing transportation services, making travel planning easy and convenient for users.

**Keyword:** Unified Travelling and Transport System, Python, database management system, transportation services, real-time information, advanced security measures, data storage and retrieval.

# TABLE OF CONTENTS

# Chapter 1: INTRODUCTION

## 1.1 Introduction to UTTS:

The UTTS aims to simplify the process of booking transportation services for users, by providing a unified platform to access information related to different transportation services. This eliminates the need for users to visit multiple websites or applications to find and book transportation services, making the process more convenient and streamlined.

The system also provides real-time information on the availability of different transportation services and their schedules, allowing users to plan their travel more effectively. In addition, the UTTS features advanced security measures to protect user data and prevent unauthorized access.

## 1.2 Objective:

The Unified Travelling and Transport System (UTTS) has several objectives, which include:

1. **Providing a unified platform:** The primary objective of the UTTS is to provide a unified platform for users to access information related to different transportation services. By consolidating information from various sources, the UTTS aims to simplify the process of booking transportation services for users.
2. **Real-time information:** The system also aims to provide users with real-time information on the availability of different transportation services and their schedules. This helps users to plan their travel more effectively and make informed decisions about their transportation options.
3. **Efficient data management:** The UTTS employs a database management system that is designed to handle large amounts of data related to different transportation services. The system aims to ensure efficient data storage and retrieval, making it capable of handling a large number of users and transactions.
4. **Advanced security measures:** The system aims to provide advanced security measures to protect user data and prevent unauthorized access. This helps to ensure that user information is kept safe and secure.
5. **Convenience and efficiency:** The overall objective of the UTTS is to provide a convenient and efficient platform for booking and managing transportation services. By

streamlining the process of booking transportation services, the UTTS aims to save users time and effort and make travel planning more accessible to all.

## 1.3 How it Works:

The Unified Travelling and Transport System (UTTS) works by providing a unified platform for booking and managing transportation services. Here is a brief overview of how the system works:

1. **User registration:** Users can register on the UTTS platform by providing their personal information and creating a login account.
2. **Searching for transportation services:** Users can search for transportation services based on their travel requirements, such as the date, time, and destination.
3. **Availability and schedules:** The system provides real-time information on the availability of different transportation services and their schedules, enabling users to select the most convenient and efficient transportation option.
4. **Booking and payment:** Users can book their chosen transportation services and make payments using the secure payment gateway integrated into the UTTS platform.
5. **Confirmation and ticket generation:** Once the booking and payment are confirmed, users receive a confirmation message and an e-ticket for their transportation service.
6. **Managing bookings:** Users can manage their bookings through the UTTS platform, making changes or canceling their bookings if necessary.
7. **Data management:** The UTTS employs a database management system that stores and manages data related to transportation services, ensuring efficient data storage and retrieval.

# Chapter 2: IMPLEMENTATION

## 2.1 Code:

### 2.1.1 Graphics User Interface:

- Login Page:

```
import tkinter as tk

import tkinter.messagebox
import mysql.connector
import customtkinter
from PIL import Image, ImageTk
import os
from tkinter import PhotoImage
from tkinter import messagebox
from tkinter.font import Font

customtkinter.set_appearance_mode("dark")

class Login(customtkinter.CTk):
    width = 1240  #helps in image width
    height = 1080 #helps in image height
    def __init__(self):
        super().__init__()

        # OPENEING WINDOW SIZE
        self.title("Login")
        self.geometry(f"{1240}x{720}")
        self.bg_image = 
customtkinter.CTkImage(Image.open("Image/Background_gradient.jpg"),size=(self.width, 
self.height))
        self.bg_image_label = customtkinter.CTkLabel(self, image=self.bg_image)
        self.bg_image_label.grid(row=0, column=0)

        # LOGIN FRAME INSIDE WINDOW
        # TEXT : "Welcome!\nUnified Travelling & Transport System"
        self.login_frame = customtkinter.CTkFrame(self, corner_radius=15)
        self.login_frame.grid(row=0, column=0, sticky="ns")
        self.login_label = customtkinter.CTkLabel(self.login_frame, 
text="Welcome!\nUnified Travelling & Transport System",font=customtkinter.CTkFont(size=24, 
weight="bold", slant="roman", family="Helvetica"))
        self.login_label.grid(row=0, column=0, padx=30, pady=(150, 15))

        #TEXT : LOGIN PAGE
        self.login_label_2 = customtkinter.CTkLabel(self.login_frame, text="Login 
Page",font=customtkinter.CTkFont(size=40, weight="bold"))
        self.login_label_2.grid(row=1, column=0, padx=30, pady=(50, 15))

        #TEXT : USERNAME
        self.username_entry = customtkinter.CTkEntry(self.login_frame, width=300, 
placeholder_text="Username")
        self.username_entry.grid(row=2, column=0, padx=30, pady=(15, 15))

        #TEXT : PASSWORD
        self.password_entry = customtkinter.CTkEntry(self.login_frame, width=300, 
show="*", placeholder_text="Password")
        self.password_entry.grid(row=3, column=0, padx=30, pady=(0, 15))

        #TEXT : LOGIN BUTTON TEXT
        self.login_button = customtkinter.CTkButton(self.login_frame, text="Login", 
command=self.login_event, width=200)
        self.login_button.grid(row=4, column=0, padx=30, pady=(15, 15))
    def login_event(self):

        UTTSdb = mysql.connector.connect(
        host='localhost',
        user='root',
        password='harsh',
        database='UTTS')

        entered_username = self.username_entry.get()
        entered_password = self.password_entry.get()

        cur=UTTSdb.cursor()
        s="SELECT * FROM users WHERE first_name = '{}' AND Password =
'{}'".format(entered_username,entered_password)
        cur.execute(s)
        QueryCheckForPassword=cur.fetchone()

        if QueryCheckForPassword:
            self.destroy()
            import StartPageGUI
            StartPageGUI.Main().mainloop()

        else:
            print("error")
            return messagebox.showerror('Error','Incorrect Username or Password')

        print("Login pressed - username:", entered_username, "password:",entered_password)

if __name__ == "__main__":
    app9 = Login()
    app9.mainloop()
```

- Home Page:

```
import tkinter

import tkinter.messagebox
import customtkinter
from PIL import Image, ImageTk
import os
from tkinter import PhotoImage


window1 = customtkinter.CTk()
# AVAILABLE MODES->"System", "Dark", "Light"
customtkinter.set_appearance_mode("System")
#AVAILABLE THEMES->"blue", "green", "dark-blue"
customtkinter.set_default_color_theme("green")

class Main(customtkinter.CTk):

    def __init__(self):
        super().__init__()

        #DEFINING WINDOW NAME AND SIZE
        self.title("Home page")
        self.geometry(f"{1440}x{540}")

        # configure grid layout (4x4)
        self.grid_columnconfigure(1, weight=0)
        self.grid_columnconfigure((2, 3), weight=0)
        self.grid_rowconfigure((0, 1, 2), weight=0)

        self.sidebar_frame = customtkinter.CTkFrame(self, width=120, corner_radius=15)
        self.sidebar_frame.grid(row=9, column=0, rowspan=4, sticky="ns")
        self.sidebar_frame.grid_rowconfigure(1, weight=1)

        # PROFILE BUTTON
        self.profile_button = customtkinter.CTkButton(self.sidebar_frame,text="Profile")
        self.profile_button.grid(row=0,column=0,padx=10,pady=15)

        # LOGIN BUTTON
        self.back_to_loginPage_button =
customtkinter.CTkButton(self.sidebar_frame,text="Log Out", command=self.open_Login_window)
        self.back_to_loginPage_button.grid(row=1, column=0, padx=10, pady=15)

        # APPEARANCE TEXT
        self.appearance_mode_label = customtkinter.CTkLabel(self.sidebar_frame,
text="Appearance Mode:", anchor="w")
        self.appearance_mode_label.grid(row=5, column=0, padx=10, pady=(5,0))
        # APPEARANCE BUTTON
        self.appearance_mode_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame,
values=["Light", "Dark", "System"],command=self.change_appearance_mode_event)
        self.appearance_mode_optionemenu.grid(row=6, column=0, padx=20, pady =(10,10))

        # SCALING TEXT
        self.scaling_label = customtkinter.CTkLabel(self.sidebar_frame, text="UI
Scaling:")
        self.scaling_label.grid(row=7, column=0, padx=20, pady=(10,0))
        # SCALING BUTTON
        self.scaling_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame,
values=["70%", "80%", "90%", "100%", "110%", "120%",
"130%"],command=self.change_scaling_event)
        self.scaling_optionemenu.grid(row=8, column=0, padx=20, pady=(5,20))

        self.appearance_mode_optionemenu.set("Dark")
        self.scaling_optionemenu.set("100%")


        self.upper_name_frame = customtkinter.CTkFrame(self, width=140,height=50,
corner_radius=15)
        self.upper_name_frame.grid(row=0, column=2, rowspan=4, sticky="nsew")
        self.upper_name_frame.grid_rowconfigure(6, weight=1)

        self.project_name = customtkinter.CTkLabel(self.upper_name_frame,
text="UTTS\nUnified Traveling and Transportation System",
font=customtkinter.CTkFont(size=38, weight="bold"))
        self.project_name.grid(row=0, column=2, padx=200, pady=6)

        #CREATING TAB VIEW
        self.tabview = customtkinter.CTkTabview(self, width=1240, height= 50,
corner_radius=10)
        self.tabview.grid(row=4, column=2, padx=0, pady=(5, 10))
        self.tabview.add("Travel")
        self.tabview.tab("Travel").grid_columnconfigure(0, weight=0)  # configure grid of
individual tabs
        self.tabview.add("Transport")
        self.tabview.tab("Transport").grid_columnconfigure(0, weight=1)

        #TRAVEL
        self.Bus_button = customtkinter.CTkButton(self.tabview.tab("Travel"),
text="Bus",command=self.open_Bus_window)
        self.Bus_button.grid(row=4, column=0, padx=(0,110), pady=(10, 10),sticky="w")

        self.Car_button = customtkinter.CTkButton(self.tabview.tab("Travel"),
text="Car",command=self.open_Car_window)
        self.Car_button.grid(row=4, column=1, padx=(110,110), pady=(10, 10),sticky="nsew")

        self.Train_button = customtkinter.CTkButton(self.tabview.tab("Travel"),
text="Train",command=self.open_Train_window)
```

```
        self.Train_button.grid(row=4, column=2, padx=(110,110), pady=(10,
10),sticky="nsew")

        self.Airplane_button = customtkinter.CTkButton(self.tabview.tab("Travel"),
text="Airplane",command=self.open_Airplane_window)
        self.Airplane_button.grid(row=4, column=3, padx=(110,0), pady=(10, 10),sticky="e")


        #TRANSPORT
        self.Truck_button = customtkinter.CTkButton(self.tabview.tab("Transport"),
text="Truck",command=self.open_Truck_window)
        self.Truck_button.grid(row=4, column=0, padx=20, pady=(10, 10),sticky="w")

        self.Ship_button = customtkinter.CTkButton(self.tabview.tab("Transport"),
text="Railways",command=self.open_Ship_window)
        self.Ship_button.grid(row=4, column=1, padx=20, pady=(10, 10),sticky="e")


        self.tabview = customtkinter.CTkTabview(self, width=240, height= 80)
        self.tabview.grid(row=10, column=2, padx=0, pady=0,sticky="s")
        self.tabview.add("!Error Encountered!")

        self.string_input_button = customtkinter.CTkButton(self.tabview.tab("!Error
Encountered!"), text="Feedback Button",command=self.open_input_dialog_event)
        self.string_input_button.grid(row=10, column=19, padx=20, pady=(10, 10),
sticky="nsew")
        self.string_input_button.pack(side="top", anchor="center")


    def open_input_dialog_event(self):
        dialog = customtkinter.CTkInputDialog(text="Please enter your valuable Feedback
:", title="Feedback Window")
        print("Feedback :", dialog.get_input())

    def change_appearance_mode_event(self, new_appearance_mode: str):
        customtkinter.set_appearance_mode(new_appearance_mode)

    def change_scaling_event(self, new_scaling: str):
        new_scaling_float = int(new_scaling.replace("%", "")) / 100
        customtkinter.set_widget_scaling(new_scaling_float)

    def open_Login_window(self):
        self.destroy()
        import Login_page
        Login_page.Login().mainloop()

    def open_Bus_window(self):
        self.destroy()
        import Bus_Home_page
        Bus_Home_page.Bus().mainloop()

    def open_Car_window(self):
        self.destroy()
        import Car_Home_page
        Car_Home_page.Car().mainloop()

    def open_Train_window(self):
        self.destroy()
        import Train_Home_page
        Train_Home_page.Train().mainloop()

    def open_Airplane_window(self):
        self.destroy()
        import Airplane_Home_page
        Airplane_Home_page.Airplane().mainloop()
    def open_Truck_window(self):
        self.destroy()
        import Truck_Home_page
        Truck_Home_page.Truck().mainloop()

    def open_Ship_window(self):
        self.destroy()
        import Ship_Home_page
        Ship_Home_page.Ship().mainloop()


if __name__ == "__main__":
    app1 = Main()
    app1.mainloop()
```

- Bus page:

```
import tkinter as tk

import tkinter.messagebox
import customtkinter
from PIL import Image, ImageTk
import os
from tkinter import PhotoImage
from tkinter import messagebox
from tkinter import ttk
from tkinter import *
from tkinter.ttk import *
import mysql.connector

UTTSdb = mysql.connector.connect(
    host='localhost',
    user='root',
    password='Rajput@MySQL',
    database='UTTS')
cur=UTTSdb.cursor()

window2 = customtkinter.CTk()
customtkinter.set_appearance_mode("System")
customtkinter.set_default_color_theme("blue")

class Bus(customtkinter.CTk):
    def __init__(self):
        super().__init__()
        self.title("Bus Home Page")
        self.geometry(f"{1700}x{580}")
        # self.part1()

   #Appearance and Scaling
        self.sidebar_frame = customtkinter.CTkFrame(self, width=120, corner_radius=0)
        self.sidebar_frame.grid(row=35, column=0, rowspan=4, sticky="ew")
        self.sidebar_frame.grid_rowconfigure(4, weight=1)


        self.appearance_mode_label = customtkinter.CTkLabel(self.sidebar_frame,
text="Appearance Mode:", anchor="w")
        self.appearance_mode_label.grid(row=5, column=0, padx=20, pady=(10,0))
        self.appearance_mode_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame,
values=["Light", "Dark", "System"],
                                                                    command=self.change_appe
arance_mode_event)
        self.appearance_mode_optionemenu.grid(row=6, column=0, padx=20, pady =(10,10),
sticky ="ew")
```

```
        self.scaling_label = customtkinter.CTkLabel(self.sidebar_frame, text="UI 
Scaling:", anchor="w")
        self.scaling_label.grid(row=7, column=0, padx=20, pady=(10,0))
        self.scaling_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame, 
values=["70%", "80%", "90%", "100%", "110%", "120%", "130%"],
                                                               command=self.change_scaling
_event)
        self.scaling_optionemenu.grid(row=8, column=0, padx=20, pady=(10,20), sticky
="ew")

        self.appearance_mode_optionemenu.set("Dark")
        self.scaling_optionemenu.set("110%")

# name of transport
        self.sidebar_frame0 = customtkinter.CTkFrame(self, width=100,height=30, 
corner_radius=0)
        self.sidebar_frame0.grid(row=0, column=15, rowspan=10)
        self.sidebar_frame0.grid_rowconfigure(8, weight=1)
        self.logo_label = customtkinter.CTkLabel(self.sidebar_frame0, text="Bus Booking 
Services", font=customtkinter.CTkFont(size=50, weight="bold"))
        self.logo_label.grid(row=0, column=15, padx=600, pady=50)

# from button
        self.sidebar_frame1=customtkinter.CTkFrame(self,width=200,height=100)
        self.sidebar_frame1.grid(row=15,column=15,rowspan=25, padx=20, pady=10)
        self.to_label = customtkinter.CTkLabel(self.sidebar_frame1, 
text="FROM",font=customtkinter.CTkFont(size=20),anchor="w")
        self.to_label.grid(row=15, column=15, padx=100, pady=10)
        self.from_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame1,values=["-
Select-","GWL", "BHP","MUM", "DLH"])
        self.from_optionemenu.grid(row=16, column=15, padx=100, pady=10)

# to button
        self.to_label = customtkinter.CTkLabel(self.sidebar_frame1, 
text="TO",font=customtkinter.CTkFont(size=20), anchor="w")
        self.to_label.grid(row=15, column=19, padx=20, pady=(10,0))
        self.to_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame1, values=["-
Select-", "DLH","MUM","BHP","GWL"])
        self.to_optionemenu.grid(row=16, column=19, padx=100, pady=10)

# to select no. of adults travelling
        self.adult_label = customtkinter.CTkLabel(self.sidebar_frame1, text="Adults 
",font=customtkinter.CTkFont(size=20) ,anchor="w")
        self.adult_label.grid(row=17, column=15, padx=100, pady=10)
        self.adult_optionemenu= tk.Spinbox(self.sidebar_frame1,from_=0, to=5, increment
=1)
        self.adult_optionemenu.grid(row=18, column=15,padx=100, pady=10)

# to select no. of children travelling
        self.children_label = customtkinter.CTkLabel(self.sidebar_frame1, text="Childrens
", font=customtkinter.CTkFont(size=20),anchor="w")
        self.children_label.grid(row=17, column=19, padx=20, pady=(10,0))
        self.children_optionemenu= tk.Spinbox(self.sidebar_frame1,from_=0, to=5, increment
=1)
        self.children_optionemenu.grid(row=18, column=19, padx=100, pady=10)

# continue button
        self.continue_button = 
customtkinter.CTkButton(self,text="Continue",command=self.end_e)
        self.continue_button.grid(row=100, column=15,rowspan=100, padx=20, pady=(10,10))

# back button
        self.string_input_button = customtkinter.CTkButton(self,text="Back Button", 
command=self.open_Main_window)
        self.string_input_button.grid(row=1, column=0, padx=20, pady=(10, 10))
    def open_Main_window(self):
        self.destroy()
        import StartPageGUI
        StartPageGUI.Main().mainloop()

    def end_e(self):#function to end app-GUI
        global rawa
        global rawc

        global f1
        f1= self.from_optionemenu.get() #from value

        global f2
        f2= self.to_optionemenu.get() #to value

        a = self.adult_optionemenu.get()
        rawa=a
        b= self.children_optionemenu.get()
        rawc=b
        if f1=='-Select-' and f2=='-Select-':
            return messagebox.showerror('Error','Please select Departure & Arrival 
locations')
        elif f2=='-Select-':
            return messagebox.showerror("Error", "Select Arrival location")
        elif f1==f2:
            return messagebox.showerror("Error", "Invalid location entry!")
        elif f1=='-Select-':
            return messagebox.showerror('Error','Please select Departure location')
        elif rawa=='0' and rawc=='0':
            return messagebox.showerror("Error", "choose no. of passengers")
        else:

            Query="SELECT BusID,Name,Duration,type,capacity,fare FROM bus WHERE 
FromLocation='{}' AND ToLocation='{}'".format(f1,f2)
            cur.execute(Query)
            availableBUS=cur.fetchall()

            bus1_ID=availableBUS[0][0]
            bus1_Name=availableBUS[0][1]
            bus1_dur=availableBUS[0][2]
            bus1_type=availableBUS[0][3]
            bus1_cap=availableBUS[0][4]
            bus1_fare=availableBUS[0][5]
            travel_vehicle = "Buses"
            os.environ['TRAVEL_VEHICLE'] = str(travel_vehicle)
            os.environ['F1'] = str(f1)
            os.environ['F2'] = str(f2)
            os.environ['BUS1_ID'] = str(bus1_ID)
            os.environ['BUS1_NAME'] = str(bus1_Name)
            os.environ['BUS1_DUR'] = str(bus1_dur)
            os.environ['BUS1_TYPE'] = str(bus1_type)
            os.environ['BUS1_CAP'] = str(bus1_cap)
            os.environ['BUS1_FARE'] = str(bus1_fare)


            bus2_ID=availableBUS[1][0]
            bus2_Name=availableBUS[1][1]
            bus2_dur=availableBUS[1][2]
            bus2_type=availableBUS[1][3]
            bus2_cap=availableBUS[1][4]
            bus2_fare=availableBUS[1][5]
            os.environ['BUS2_ID'] = str(bus2_ID)
            os.environ['BUS2_NAME'] = str(bus2_Name)
            os.environ['BUS2_DUR'] = str(bus2_dur)
            os.environ['BUS2_TYPE'] = str(bus2_type)
            os.environ['BUS2_CAP'] = str(bus2_cap)
            os.environ['BUS2_FARE'] = str(bus2_fare)
            self.open_Info_window()

    def open_Info_window(self):
        self.destroy()
        import Bus_Route_Info
        Bus_Route_Info.Route().mainloop()

    def change_appearance_mode_event(self, new_appearance_mode: str):
        customtkinter.set_appearance_mode(new_appearance_mode)

    def change_scaling_event(self, new_scaling: str):
        new_scaling_float = int(new_scaling.replace("%", "")) / 100
        customtkinter.set_widget_scaling(new_scaling_float)

if __name__ == "__main__":
    app2 = Bus()
    app2.mainloop()
```

- Car Page:

```
import tkinter as tk

import tkinter.messagebox
import customtkinter
from PIL import Image, ImageTk
import os
from tkinter import PhotoImage
from tkinter import messagebox
import mysql.connector

UTTSdb = mysql.connector.connect(
    host='localhost',
    user='root',
    password='Rajput@MySQL',
    database='UTTS')
cur=UTTSdb.cursor()

window3 = customtkinter.CTk()
customtkinter.set_appearance_mode("System")
customtkinter.set_default_color_theme("blue")

class Car(customtkinter.CTk):
    def __init__(self):
        super().__init__()
        self.title("Car Home Page")
        self.geometry(f"{1700}x{580}")

        #Appearance and Scaling
        self.sidebar_frame = customtkinter.CTkFrame(self, width=120, corner_radius=0)
        self.sidebar_frame.grid(row=35, column=0, rowspan=4, sticky="ew")
        self.sidebar_frame.grid_rowconfigure(4, weight=1)

        self.appearance_mode_label = customtkinter.CTkLabel(self.sidebar_frame,
text="Appearance Mode:", anchor="w")
        self.appearance_mode_label.grid(row=5, column=0, padx=20, pady=(10,0))
        self.appearance_mode_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame,
values=["Light", "Dark", "System"],
                                                                       command=self.change_appe
arance_mode_event)
        self.appearance_mode_optionemenu.grid(row=6, column=0, padx=20, pady =(10,10),
sticky ="ew")

        self.scaling_label = customtkinter.CTkLabel(self.sidebar_frame, text="UI
Scaling:", anchor="w")
        self.scaling_label.grid(row=7, column=0, padx=20, pady=(10,0))
        self.scaling_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame,
values=["70%", "80%", "90%", "100%", "110%", "120%", "130%"],
                                                               command=self.change_scaling
_event)
        self.scaling_optionemenu.grid(row=8, column=0, padx=20, pady=(10,20), sticky
="ew")

        self.appearance_mode_optionemenu.set("Dark")
        self.scaling_optionemenu.set("110%")

# name of transport
        self.sidebar_frame0 = customtkinter.CTkFrame(self, width=100,height=30,
corner_radius=0)
        self.sidebar_frame0.grid(row=0, column=15, rowspan=10)
        self.sidebar_frame0.grid_rowconfigure(8, weight=1)
        self.logo_label = customtkinter.CTkLabel(self.sidebar_frame0, text="Car Booking
Services", font=customtkinter.CTkFont(size=50, weight="bold"))
        self.logo_label.grid(row=0, column=15, padx=600, pady=50)

# # from button
        self.sidebar_frame1=customtkinter.CTkFrame(self,width=200,height=100)
        self.sidebar_frame1.grid(row=15,column=15,rowspan=25, padx=20, pady=10)
        self.to_label = customtkinter.CTkLabel(self.sidebar_frame1,
text="FROM",font=customtkinter.CTkFont(size=20),anchor="w")
        self.to_label.grid(row=15, column=15, padx=100, pady=10)
        self.from_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame1,values=["-
Select-","Muscat", "Mumbai", "Delhi",'Bangalore'])
        self.from_optionemenu.grid(row=16, column=15, padx=100, pady=10)

# # to button
        self.to_label = customtkinter.CTkLabel(self.sidebar_frame1,
text="TO",font=customtkinter.CTkFont(size=20), anchor="w")
        self.to_label.grid(row=15, column=19, padx=20, pady=(10,0))
        self.to_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame1, values=["-
Select-","Muscat", "Mumbai", "Delhi",'Bangalore'])
        self.to_optionemenu.grid(row=16, column=19, padx=100, pady=10)

# to select no. of adults travelling
        self.adult_label = customtkinter.CTkLabel(self.sidebar_frame1, text="Adults
",font=customtkinter.CTkFont(size=20) ,anchor="w")
        self.adult_label.grid(row=17, column=15, padx=100, pady=10)
        self.adult_optionemenu= tk.Spinbox(self.sidebar_frame1,from_=0, to=5, increment
=1)
        self.adult_optionemenu.grid(row=18, column=15,padx=100, pady=10)

# to select no. of children travelling
        self.children_label = customtkinter.CTkLabel(self.sidebar_frame1, text="Childrens
", font=customtkinter.CTkFont(size=20),anchor="w")
        self.children_label.grid(row=17, column=19, padx=20, pady=(10,0))
        self.children_optionemenu= tk.Spinbox(self.sidebar_frame1,from_=0, to=5, increment
=1)
        self.children_optionemenu.grid(row=18, column=19, padx=100, pady=10)

# continue button
        self.continue_button =
customtkinter.CTkButton(self,text="Continue",command=self.end_e)
        self.continue_button.grid(row=100, column=15,rowspan=100, padx=20, pady=(10,10))

# back button
        self.string_input_button = customtkinter.CTkButton(self,text="Back Button",
command=self.open_Main_window)
        self.string_input_button.grid(row=1, column=0, padx=20, pady=(10, 10))

    def open_Main_window(self):
        self.destroy()
        import StartPageGUI
```

```
            StartPageGUI.Main().mainloop()

    def end_e(self):#function to end app-GUI
        global rawa
        global rawc
        f1= self.from_optionemenu.get()  #from value
        f2= self.to_optionemenu.get() #to value
        a = self.adult_optionemenu.get()
        rawa=a
        b= self.children_optionemenu.get()
        rawc=b
        if f1=='-Select-' and f2=='-Select-':
            return messagebox.showerror('Error','Please select Departure & Arrival 
locations')
        elif f2=='-Select-':
            return messagebox.showerror("Error", "Select Arrival location")
        elif f1==f2:
            return messagebox.showerror("Error", "Invalid location entry!")
        elif f1=='-Select-':
            return messagebox.showerror('Error','Please select Departure location')
        elif rawa=='0' and rawc=='0':
            return messagebox.showerror("Error", "choose no. of passengers")
        else:
            Query="SELECT CarID,Name,Duration,type,capacity,fare FROM car WHERE 
FromLocation='{}' AND ToLocation='{}'".format(f1,f2)
            cur.execute(Query)
            availableCAR=cur.fetchall()

            Car1_PNR=availableCAR[0][0]
            Car1_Name=availableCAR[0][1]
            Car1_dur=availableCAR[0][2]
            Car1_type=availableCAR[0][3]
            Car1_cap=availableCAR[0][4]
            Car1_fare=availableCAR[0][5]

            Car2_ID=availableCAR[1][0]
            Car2_Name=availableCAR[1][1]
            Car2_dur=availableCAR[1][2]
            Car2_type=availableCAR[1][3]
            Car2_cap=availableCAR[1][4]
            Car2_fare=availableCAR[1][5]
            self.open_Info_window()

    def open_Info_window(self):
        self.destroy()
        import Route_Info
        Route_Info.Route().mainloop()

    def change_appearance_mode_event(self, new_appearance_mode: str):
        customtkinter.set_appearance_mode(new_appearance_mode)

    def change_scaling_event(self, new_scaling: str):
        new_scaling_float = int(new_scaling.replace("%", "")) / 100
        customtkinter.set_widget_scaling(new_scaling_float)

if __name__ == "__main__":
    app3 = Car()
    app3.mainloop()
```

- Train Page:

```
import tkinter as tk

import tkinter.messagebox
import customtkinter
from PIL import Image, ImageTk
```

```
import os
from tkinter import PhotoImage
from tkinter import messagebox
import mysql.connector

UTTSdb = mysql.connector.connect(
    host='localhost',
    user='root',
    password='Rajput@MySQL',
    database='UTTS')
cur=UTTSdb.cursor()

window4 = customtkinter.CTk()
customtkinter.set_appearance_mode("System")
customtkinter.set_default_color_theme("blue")

class Train(customtkinter.CTk):

    def __init__(self):
        super().__init__()
        self.title("Train Home Page")
        self.geometry(f"{1700}x{580}")

        #Appearance and Scaling
        self.sidebar_frame = customtkinter.CTkFrame(self, width=120, corner_radius=0)
        self.sidebar_frame.grid(row=35, column=0, rowspan=4, sticky="ew")
        self.sidebar_frame.grid_rowconfigure(4, weight=1)

        self.appearance_mode_label = customtkinter.CTkLabel(self.sidebar_frame,
text="Appearance Mode:", anchor="w")
        self.appearance_mode_label.grid(row=5, column=0, padx=20, pady=(10,0))
        self.appearance_mode_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame,
values=["Light", "Dark", "System"],
                                                                   command=self.change_appe
arance_mode_event)
        self.appearance_mode_optionemenu.grid(row=6, column=0, padx=20, pady =(10,10),
sticky ="ew")

        self.scaling_label = customtkinter.CTkLabel(self.sidebar_frame, text="UI
Scaling:", anchor="w")
        self.scaling_label.grid(row=7, column=0, padx=20, pady=(10,0))
        self.scaling_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame,
values=["70%", "80%", "90%", "100%", "110%", "120%", "130%"],
                                                                command=self.change_scaling
_event)
        self.scaling_optionemenu.grid(row=8, column=0, padx=20, pady=(10,20), sticky
="ew")

        self.appearance_mode_optionemenu.set("Dark")
        self.scaling_optionemenu.set("110%")

# name of transport
        self.sidebar_frame0 = customtkinter.CTkFrame(self, width=100,height=30,
corner_radius=0)
        self.sidebar_frame0.grid(row=0, column=15, rowspan=10)
        self.sidebar_frame0.grid_rowconfigure(8, weight=1)
        self.logo_label = customtkinter.CTkLabel(self.sidebar_frame0, text="Train Booking
Services", font=customtkinter.CTkFont(size=50, weight="bold"))
        self.logo_label.grid(row=0, column=15, padx=600, pady=50)

# # from button
        self.sidebar_frame1=customtkinter.CTkFrame(self,width=200,height=100)
        self.sidebar_frame1.grid(row=15,column=15,rowspan=50, padx=20, pady=10)
        self.to_label = customtkinter.CTkLabel(self.sidebar_frame1,
text="FROM",font=customtkinter.CTkFont(size=20),anchor="w")
        self.to_label.grid(row=15, column=15, padx=100, pady=10)
        self.from_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame1,values=["-
Select-","GWL", "BHP","MUM", "DLH"])
        self.from_optionemenu.grid(row=16, column=15, padx=100, pady=10)

# # to button
        self.to_label = customtkinter.CTkLabel(self.sidebar_frame1,
text="TO",font=customtkinter.CTkFont(size=20), anchor="w")
        self.to_label.grid(row=15, column=19, padx=20, pady=(10,0))
        self.to_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame1, values=["-
Select-","GWL", "BHP","MUM", "DLH"])
        self.to_optionemenu.grid(row=16, column=19, padx=100, pady=10)

# to select no. of adults travelling
        self.adult_label = customtkinter.CTkLabel(self.sidebar_frame1, text="Adults
",font=customtkinter.CTkFont(size=20) ,anchor="w")
        self.adult_label.grid(row=17, column=15, padx=100, pady=10)
        self.adult_optionemenu= tk.Spinbox(self.sidebar_frame1,from_=0, to=5, increment
=1)
        self.adult_optionemenu.grid(row=18, column=15,padx=100, pady=10)

# to select no. of children travelling
        self.children_label = customtkinter.CTkLabel(self.sidebar_frame1, text="Childrens
", font=customtkinter.CTkFont(size=20),anchor="w")
        self.children_label.grid(row=17, column=19, padx=20, pady=(10,0))
        self.children_optionemenu= tk.Spinbox(self.sidebar_frame1,from_=0, to=5, increment
=1)
        self.children_optionemenu.grid(row=18, column=19, padx=100, pady=10)

# continue button
        self.continue_button =
customtkinter.CTkButton(self,text="Continue",command=self.end_e)
        self.continue_button.grid(row=100, column=15,rowspan=100, padx=20, pady=(10,10))

# back button
        self.string_input_button = customtkinter.CTkButton(self,text="Back Button",
command=self.open_Main_window)
        self.string_input_button.grid(row=1, column=0, padx=20, pady=(10, 10))

    def open_Main_window(self):
        self.destroy()
        import StartPageGUI
        StartPageGUI.Main().mainloop()

    def end_e(self):#function to end app-GUI
        global rawa
        global rawc
        global f1
        f1 = self.from_optionemenu.get()  #from value
        global f2
        f2= self.to_optionemenu.get() #to value
        a = self.adult_optionemenu.get()
        rawa=a
        b= self.children_optionemenu.get()
        rawc=b
        if f1=='-Select-' and f2=='-Select-':
            return messagebox.showerror('Error','Please select Departure & Arrival
locations')
        elif f2=='-Select-':
            return messagebox.showerror("Error", "Select Arrival location")
        elif f1==f2:
            return messagebox.showerror("Error", "Invalid location entry!")
        elif f1=='-Select-':
            return messagebox.showerror('Error','Please select Departure location')
        elif rawa=='0' and rawc=='0':
            return messagebox.showerror("Error", "choose no. of passengers")
        else:
            Train1_PNR = 0
            Train2_PNR = 0
            Query="SELECT PNR,Name,Duration,type,capacity,fare FROM train WHERE 
FromLocation='{}' AND ToLocation='{}'".format(f1,f2)
            cur.execute(Query)
            availableTRAIN=cur.fetchall()

            travel_vehicle = "Railway"
            os.environ['TRAVEL_VEHICLE'] = str(travel_vehicle)
            os.environ['F1'] = str(f1)
            os.environ['F2'] = str(f2)

            Number_of_train = len(availableTRAIN)
            print(Number_of_train)
            os.environ['NUMBER_OF_TRAIN'] = str(Number_of_train)
            if Number_of_train == 2:
               Train1_PNR=availableTRAIN[0][0]
               Train1_Name=availableTRAIN[0][1]
               Train1_dur=availableTRAIN[0][2]
               Train1_type=availableTRAIN[0][3]
               Train1_cap=availableTRAIN[0][4]
               Train1_fare=availableTRAIN[0][5]

               os.environ['TRAIN1_PNR'] =  str(Train1_PNR)
               os.environ['TRAIN1_NAME']  = str(Train1_Name)
               os.environ['TRAIN1_DUR']  = str(Train1_dur)
               os.environ['TRAIN1_TYPE']  = str(Train1_type)
               os.environ['TRAIN1_CAP']  = str(Train1_cap)
               os.environ['TRAIN1_FARE']  = str(Train1_fare)

               Train2_PNR=availableTRAIN[1][0]
               Train2_Name=availableTRAIN[1][1]
               Train2_dur=availableTRAIN[1][2]
               Train2_type=availableTRAIN[1][3]
               Train2_cap=availableTRAIN[1][4]
               Train2_fare=availableTRAIN[1][5]

               os.environ['TRAIN2_PNR'] =  str(Train2_PNR)
               os.environ['TRAIN2_PNR'] =  str(Train2_PNR)
               os.environ['TRAIN2_NAME']  = str(Train2_Name)
               os.environ['TRAIN2_DUR']  = str(Train2_dur)
               os.environ['TRAIN2_TYPE']  = str(Train2_type)
               os.environ['TRAIN2_CAP']  = str(Train2_cap)
               os.environ['TRAIN2_FARE'] =  str(Train2_fare)
            elif Number_of_train == 1:
               Train1_PNR=availableTRAIN[0][0]
               Train1_Name=availableTRAIN[0][1]
               Train1_dur=availableTRAIN[0][2]
               Train1_type=availableTRAIN[0][3]
               Train1_cap=availableTRAIN[0][4]
               Train1_fare=availableTRAIN[0][5]

               os.environ['TRAIN1_PNR'] =  str(Train1_PNR)
               os.environ['TRAIN1_NAME']  = str(Train1_Name)
               os.environ['TRAIN1_DUR']  = str(Train1_dur)
               os.environ['TRAIN1_TYPE']  = str(Train1_type)
               os.environ['TRAIN1_CAP']  = str(Train1_cap)
               os.environ['TRAIN1_FARE']  = str(Train1_fare)
            else:
                return messagebox.showerror("Error", "No trian for this Route ")
            self.open_Info_window()

    def open_Info_window(self):
        self.destroy()
        import Train_Route_Info
        Train_Route_Info.Route().mainloop()
    def change_appearance_mode_event(self, new_appearance_mode: str):
        customtkinter.set_appearance_mode(new_appearance_mode)

    def change_scaling_event(self, new_scaling: str):
        new_scaling_float = int(new_scaling.replace("%", "")) / 100
        customtkinter.set_widget_scaling(new_scaling_float)

if __name__ == "__main__":
    app4 = Train()
    app4.mainloop()
```

- Flight Page:

```
import tkinter as tk

import tkinter.messagebox
import customtkinter
from PIL import Image, ImageTk
import os
from tkinter import PhotoImage
from tkinter import messagebox
import mysql.connector

UTTSdb = mysql.connector.connect(
    host='localhost',
    user='root',
    password='Rajput@MySQL',
    database='UTTS')
cur=UTTSdb.cursor()

window5 = customtkinter.CTk()
customtkinter.set_appearance_mode("System")
customtkinter.set_default_color_theme("blue")

class Airplane(customtkinter.CTk):
    def __init__(self):
        super().__init__()
        self.title("Airplane Home Page")
        self.geometry(f"{1700}x{580}")

    #Appearance and Scaling
        self.sidebar_frame = customtkinter.CTkFrame(self, width=120, corner_radius=0)
        self.sidebar_frame.grid(row=35, column=0, rowspan=4, sticky="ew")
        self.sidebar_frame.grid_rowconfigure(4, weight=1)


        self.appearance_mode_label = customtkinter.CTkLabel(self.sidebar_frame,
text="Appearance Mode:", anchor="w")
        self.appearance_mode_label.grid(row=5, column=0, padx=20, pady=(10,0))
        self.appearance_mode_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame,
values=["Light", "Dark", "System"],
                                                                   command=self.change_appe
arance_mode_event)
        self.appearance_mode_optionemenu.grid(row=6, column=0, padx=20, pady =(10,10),
sticky ="ew")


        self.scaling_label = customtkinter.CTkLabel(self.sidebar_frame, text="UI
Scaling:", anchor="w")
        self.scaling_label.grid(row=7, column=0, padx=20, pady=(10,0))
        self.scaling_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame,
values=["70%", "80%", "90%", "100%", "110%", "120%", "130%"],
                                                             command=self.change_scaling
_event)
```

```
        self.scaling_optionemenu.grid(row=8, column=0, padx=20, pady=(10,20), sticky
="ew")

        self.appearance_mode_optionemenu.set("Dark")
        self.scaling_optionemenu.set("110%")

# name of transport
        self.sidebar_frame0 = customtkinter.CTkFrame(self, width=100,height=30,
corner_radius=0)
        self.sidebar_frame0.grid(row=0, column=15, rowspan=10)
        self.sidebar_frame0.grid_rowconfigure(8, weight=1)
        self.logo_label = customtkinter.CTkLabel(self.sidebar_frame0, text="airplane
Booking Services", font=customtkinter.CTkFont(size=50, weight="bold"))
        self.logo_label.grid(row=0, column=15, padx=600, pady=50)

# # from button
        self.sidebar_frame1=customtkinter.CTkFrame(self,width=200,height=100)
        self.sidebar_frame1.grid(row=15,column=15,rowspan=50, padx=20, pady=10)
        self.to_label = customtkinter.CTkLabel(self.sidebar_frame1,
text="FROM",font=customtkinter.CTkFont(size=20),anchor="w")
        self.to_label.grid(row=15, column=15, padx=100, pady=10)
        self.from_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame1,values=["-
Select-","Muscat", "Mumbai", "Delhi",'Bangalore'])
        self.from_optionemenu.grid(row=16, column=15, padx=100, pady=10)

# # to button
        self.to_label = customtkinter.CTkLabel(self.sidebar_frame1,
text="TO",font=customtkinter.CTkFont(size=20), anchor="w")
        self.to_label.grid(row=15, column=19, padx=20, pady=(10,0))
        self.to_optionemenu = customtkinter.CTkOptionMenu(self.sidebar_frame1, values=["-
Select-","Muscat", "Mumbai", "Delhi",'Bangalore'])
        self.to_optionemenu.grid(row=16, column=19, padx=100, pady=10)

# to select no. of adults travelling
        self.adult_label = customtkinter.CTkLabel(self.sidebar_frame1, text="Adults
",font=customtkinter.CTkFont(size=20) ,anchor="w")
        self.adult_label.grid(row=17, column=15, padx=100, pady=10)
        self.adult_optionemenu= tk.Spinbox(self.sidebar_frame1,from_=0, to=5, increment
=1)
        self.adult_optionemenu.grid(row=18, column=15,padx=100, pady=10)

# to select no. of children travelling
        self.children_label = customtkinter.CTkLabel(self.sidebar_frame1, text="Childrens
", font=customtkinter.CTkFont(size=20),anchor="w")
        self.children_label.grid(row=17, column=19, padx=20, pady=(10,0))
        self.children_optionemenu= tk.Spinbox(self.sidebar_frame1,from_=0, to=5, increment
=1)
        self.children_optionemenu.grid(row=18, column=19, padx=100, pady=10)

# continue button
        self.continue_button =
customtkinter.CTkButton(self,text="Continue",command=self.end_e)
        self.continue_button.grid(row=100, column=15,rowspan=100, padx=20, pady=(10,10))

# back button
        self.string_input_button = customtkinter.CTkButton(self,text="Back Button",
command=self.open_Main_window)
        self.string_input_button.grid(row=1, column=0, padx=20, pady=(10, 10))
    def open_Main_window(self):
        self.destroy()
        import StartPageGUI
        StartPageGUI.Main().mainloop()

    def end_e(self):#function to end app-GUI
        global rawa
        global rawc
        global f1
        f1 = self.from_optionemenu.get()  #from value
        global f2
        f2 = self.to_optionemenu.get() #to value
        a = self.adult_optionemenu.get()
        rawa=a
        b= self.children_optionemenu.get()
        rawc=b
        if f1=='-Select-' and f2=='-Select-':
            return messagebox.showerror('Error','Please select Departure & Arrival 
locations')
        elif f2=='-Select-':
            return messagebox.showerror("Error", "Select Arrival location")
        elif f1==f2:
            return messagebox.showerror("Error", "Invalid location entry!")
        elif f1=='-Select-':
            return messagebox.showerror('Error','Please select Departure location')
        elif rawa=='0' and rawc=='0':
            return messagebox.showerror("Error", "choose no. of passengers")
        else:
            Query="SELECT flightNo,Name,Duration,type,capacity,fare FROM flight WHERE 
FromLocation='{}' AND ToLocation='{}'".format(f1,f2)
            cur.execute(Query)
            availableFLIGHT=cur.fetchall()

            Flight1_PNR=availableFLIGHT[0][0]
            Flight1_Name=availableFLIGHT[0][1]
            Flight1_dur=availableFLIGHT[0][2]
            Flight1_type=availableFLIGHT[0][3]
            Flight1_cap=availableFLIGHT[0][4]
            Flight1_fare=availableFLIGHT[0][5]

            Flight2_ID=availableFLIGHT[1][0]
            Flight2_Name=availableFLIGHT[1][1]
            Flight2_dur=availableFLIGHT[1][2]
            Flight2_type=availableFLIGHT[1][3]
            Flight2_cap=availableFLIGHT[1][4]
            Flight2_fare=availableFLIGHT[1][5]
            self.open_Info_window()

    def open_Info_window(self):
        self.destroy()
        import Route_Info
        Route_Info.Route().mainloop()

    def change_appearance_mode_event(self, new_appearance_mode: str):
        customtkinter.set_appearance_mode(new_appearance_mode)

    def change_scaling_event(self, new_scaling: str):
        new_scaling_float = int(new_scaling.replace("%", "")) / 100
        customtkinter.set_widget_scaling(new_scaling_float)

if __name__ == "__main__":
    app5 = Airplane()
    app5.mainloop()
```

- Truck Page:

```
import tkinter

import tkinter.messagebox
import customtkinter
from PIL import Image, ImageTk
import os
from tkinter import PhotoImage
import mysql.connector
```

```
UTTSdb = mysql.connector.connect(
    host='localhost',
    user='root',
    password='Rajput@MySQL',
    database='UTTS')
cur=UTTSdb.cursor()

window6 = customtkinter.CTk()
customtkinter.set_appearance_mode("System")
customtkinter.set_default_color_theme("blue")

class Truck(customtkinter.CTk):
    def __init__(self):
        super().__init__()
        self.title("Truck Home Page")
        self.geometry(f"{1100}x{580}")


        self.string_input_button = customtkinter.CTkButton(self,text="Back Button",
command=self.open_Main_window)
        self.string_input_button.grid(row=10, column=6, padx=20, pady=(10, 10))

    def open_Main_window(self):
            self.destroy()
            import StartPageGUI
            StartPageGUI.Main().mainloop()

if __name__ == "__main__":
    app6 = Truck()
    app6.mainloop()
```

### 2.1.2 Data Query:

- Main code:

```
import mysql.connector

# import GraphicUser_InterFace.Login_page
# import GraphicUser_InterFace.StartPageGUI

UTTSdb = mysql.connector.connect(
    host='localhost',
    user='root',
    password='harsh',
    database='UTTS')

#login page starter
# Login_page = GraphicUser_InterFace.Login_page.Login()
# Login_page.mainloop()

#saving username and password
file = open('LocalDATA//password.txt', 'r')
password = file.read()
file.close()
file = open('LocalDATA//username.txt', 'r')
username = file.read()
file.close()

cur=UTTSdb.cursor()
s="SELECT * FROM users WHERE first_name = '{}' AND Password =
'{}'".format(username,password)
cur.execute(s)
QueryCheckForPassword=cur.fetchall()
print(QueryCheckForPassword)
#If to check wether password right or wrong
#if(QueryCheckForPassword==""):
```

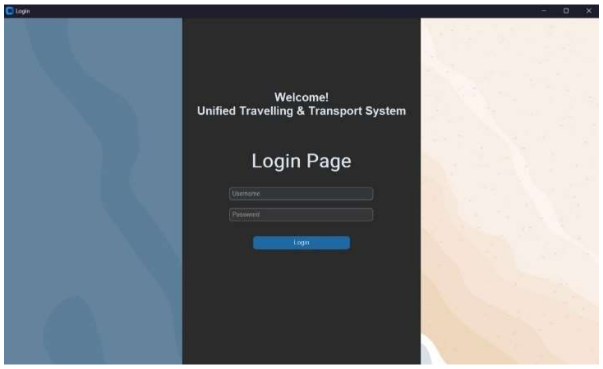
```
s="SELECT * FROM bus"
cur.execute(s)
availableBUS=cur.fetchall()
print(availableBUS)
# Login_page.destroy()
# Home_page = GraphicUser_InterFace.StartPageGUI.Main()
# Home_page.mainloop()

#print(UTTSdb.connection_id)
#print to check connection establishment
```

- Sql Queries:

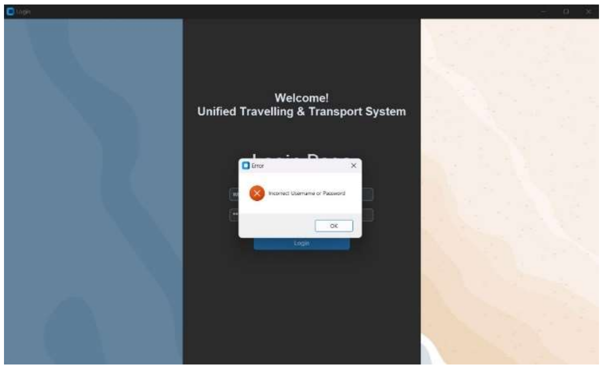
```
DatabaseCreation='CREATE DATABASE UTTS'

UserTableCreation="CREATE TABLE users(UserID INT,first_name varchar(25),last_name
varchar(25),birth_date DATE,phone varchar(15),address varchar(50),city varchar(20),state
varchar(25),points INT)"

DataInsertion="INSERT INTO users(first_name,last_name,birth_date,phone,address,city,state)
VALUES(%s,%s,%s,%s,%s,%s,%s)"
#cur.execute(s,VALUES)
#to execute one tuple

s="INSERT INTO users(first_name,last_name,birth_date,phone,address,city,state)
VALUES(%s,%s,%s,%s,%s,%s,%s)"

Values=[("harsh","shrivastava","2003-06-05",8109288418,"Hathi Khana Road
Morar","Gwalior","MP")]
#cur.executemany(s,Values)
#to insert many tuples we use this *list is passed

ManyValues=[("harsh","shrivastava","2003-06-05",8109288418,"Hathi Khana Road
Morar","Gwalior","MP")
    ("gauri","thakre","2003-11-30",9479675959,"Hostel no. 04, MITS","Gwalior","MP"),
    ("akhil","jain","2003-08-05",7456025891,"Dal Bazar","Gwalior","MP"),
    ("abhishek","rajput","2003-07-05",8109288418,"Hazira","Gwalior","MP"),
    ("shahrukh","khan","1992-01-09",8827344852,"Gandhi Chowk Bazar","Chatarpur","MP"),
    ("shraddha","kapoor","1985-08-30",6212418873,"Juhu","Mumbai","MH"),
    ("sunny","deol","1983-12-25",6748319921,"Jalianvala Bagh","Amritsar","Punjab"),
    ("hema","malini","1948-10-16",9828157533,"Kavi Bharti Nagar","Tiruchirapalli","Tamil
Nadu"),
    ("jethalal","gada","2004-12-02",7852773891,"Phool Bagh","Gwalior","MP"),
    ("kartik","aryan","1995-02-14",7143143143,"Thatipur","Gwalior","MP")]
#cur.executemany(s,Values)
#UTTSdb.commit()

#Query to update
updatedata="UPDATE users SET price=price+10 WHERE price>200"
#cur.execute(updatedata)
#uttsdb.commit()

DeleteData="DELETE FROM users WHERE title='condition'"
#cur.execute(DeleteData)
#uutsdb.commit()
```

## 2.2 Output:

**1.** When we run the Main code, it will open Python terminal of Login Page.

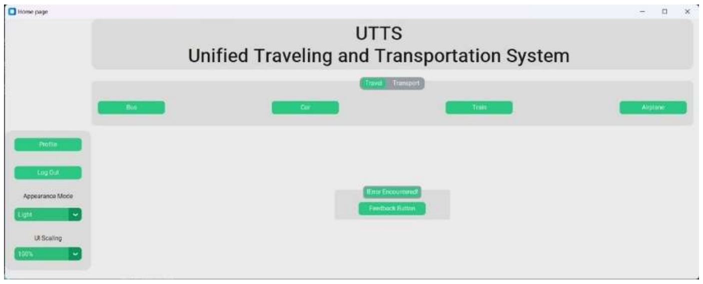


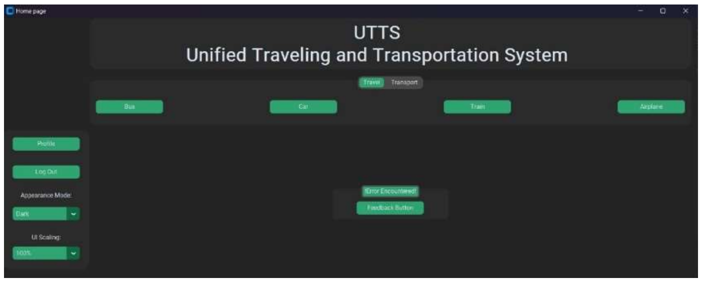
**2.** When we will enter wrong login Credentials, it will show error.

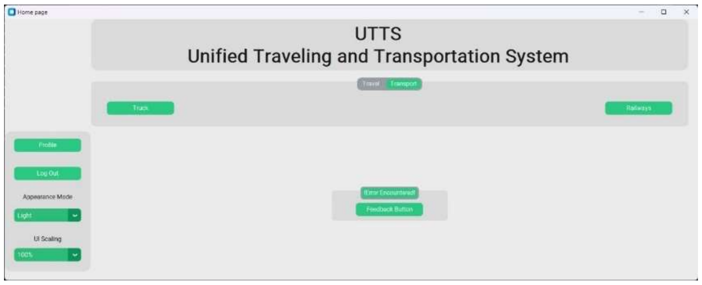


Incorrect Username or Password

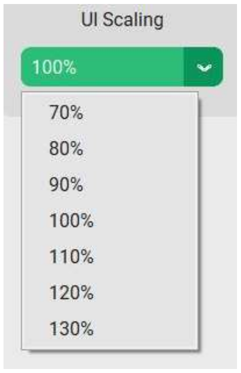
**3.** Now, we have entered correct credentials and login to main page which have Travel and Transport options and we have two mode, Light mode & Light mode. So we can switch to dark mode.

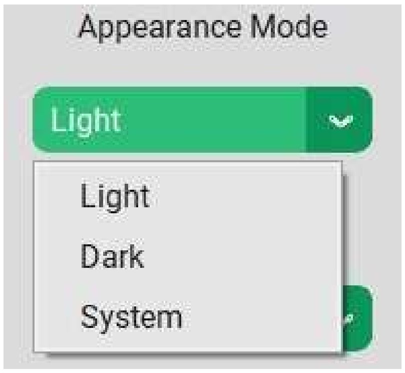


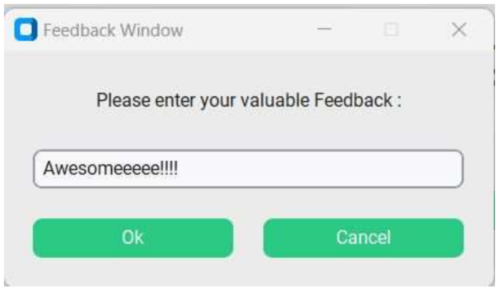




**4.** Home page have various features like Scaling, Appearance mode , Feedback button, etc.



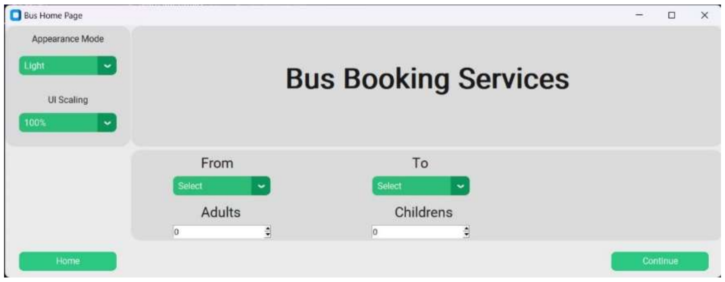


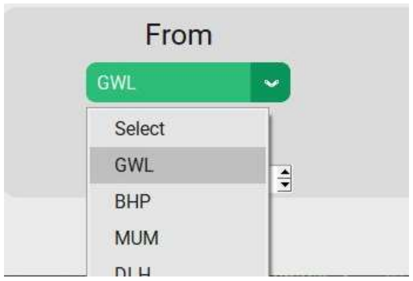


Please enter your valuable Feedback :

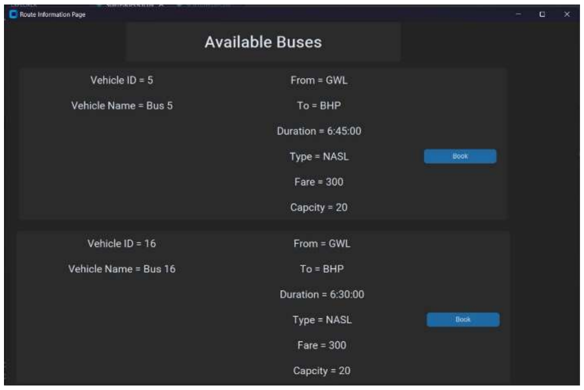
Awesomeeeee!!!!

**5.** Then we will have different booking systems like Bus, Car, etc. which have respective drop down menus.



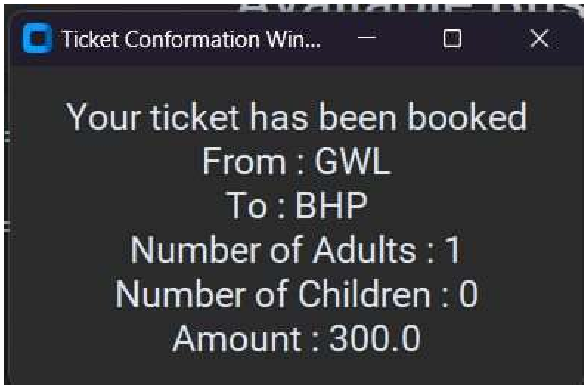


**6.** Then We have Available buses and we can book bus according our choice.



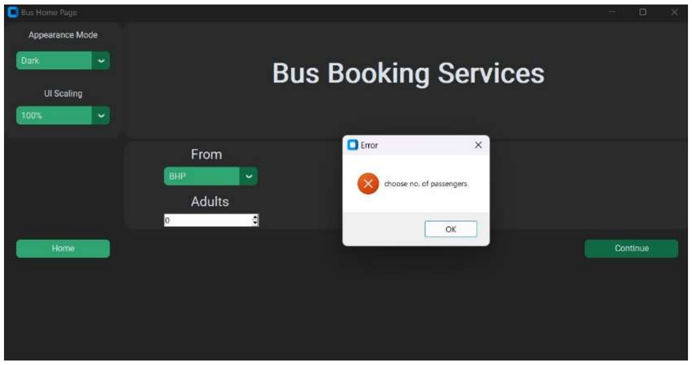
Vehicle ID = 5
Vehicle Name = Bus 5
From = GWL
To = BHP
Duration = 6:45:00
Type = NASL
Fare = 300
Capcity = 20

Vehicle ID = 16
Vehicle Name = Bus 16
From = GWL
To = BHP
Duration = 6:30:00
Type = NASL
Fare = 300
Capcity = 20



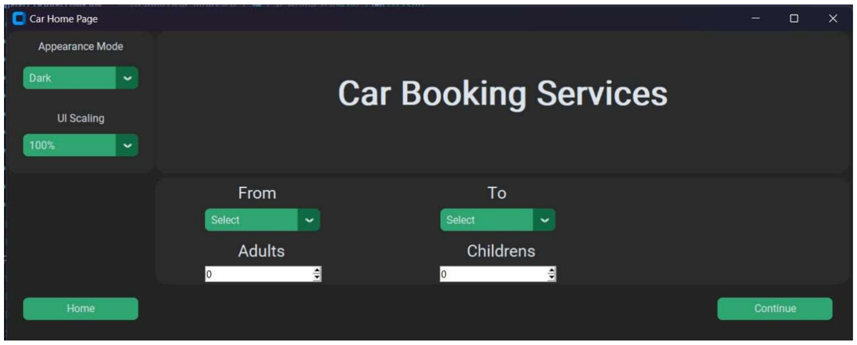
Your ticket has been booked
From : GWL
To : BHP
Number of Adults : 1
Number of Children : 0
Amount : 300.0

**7.** When the entries are incorrect, It will show errors.

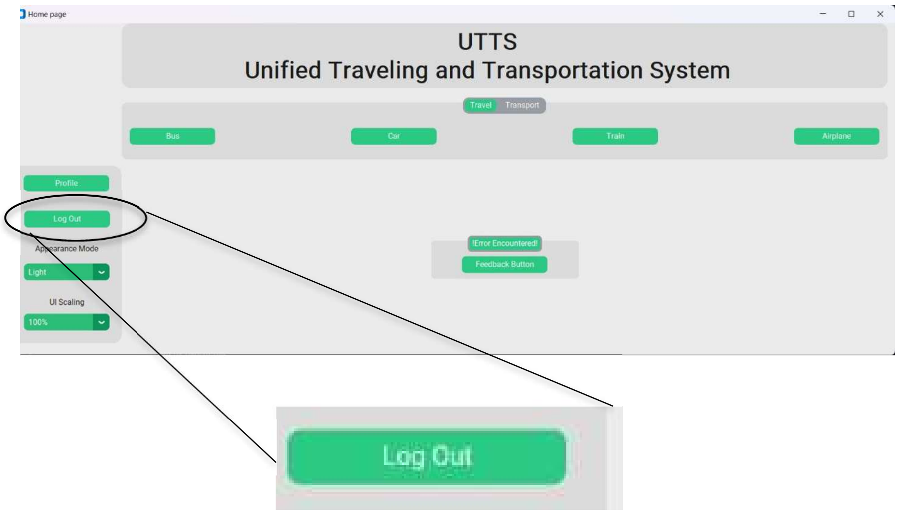


choose no. of passengers

**8.** Similarly, we have other booking systems like Car booking.



**9.** After the Booking will get completed, we can Logout using Login button.



# Chapter 3: APPLICATIONS

## 3.1 Applications of Unified Travelling and Transport System:

The Unified Travelling and Transport System (UTTS) has several applications in various transportation industries, including:

1. **Public transportation:** The UTTS can be used by public transportation providers, such as buses, trains, and metro systems, to provide real-time information on their schedules and availability. This can help to improve the efficiency of public transportation systems and enhance the overall experience for passengers.

2. **Air travel:** The UTTS can be used by airlines to provide real-time information on flight schedules, availability, and ticket prices. This can help to improve the efficiency of airline operations and enhance the overall experience for passengers.

3. **Ride-sharing services:** The UTTS can be used by ride-sharing services, such as Uber and Lyft, to provide a unified platform for booking and managing their services. This can help to streamline the booking process for users and improve the efficiency of ride-sharing services.

4. **Travel agencies:** The UTTS can be used by travel agencies to provide a unified platform for booking and managing transportation services for their clients. This can help to simplify the travel planning process for travel agencies and provide a more efficient service to their clients.

5. **Logistics and delivery services:** The UTTS can be used by logistics and delivery services to provide real-time information on the availability of their services and the status of deliveries. This can help to improve the efficiency of logistics and delivery services and enhance the overall experience for customers.

Overall, the UTTS has a wide range of applications in various transportation industries, providing a reliable and efficient platform for booking and managing transportation services.

# Chapter 4: CONCLUSION

## 4.1 Future Scope:

The Unified Travelling and Transport System (UTTS) has immense potential for future development and expansion. Here are some potential future scope areas for the UTTS:

1. **Integration with emerging technologies:** With the advent of emerging technologies such as Artificial Intelligence (AI), Internet of Things (IoT), and Blockchain, there is an opportunity for the UTTS to integrate these technologies to improve its capabilities. For example, AI could be used to provide personalized recommendations for transportation services based on users' preferences, while Blockchain could be used to improve the security and transparency of the payment process.
2. **Expansion to new markets:** The UTTS could expand to new markets beyond transportation services, such as tourism, hospitality, and entertainment. This could provide a comprehensive platform for users to plan and book their entire travel experience, including transportation, accommodation, and activities.
3. **Collaboration with transportation providers:** The UTTS could collaborate with transportation providers to improve their services and provide a better user experience. For example, transportation providers could provide real-time data on the availability and status of their services to the UTTS, which could then be used to improve the accuracy and efficiency of the booking process.
4. **Integration with smart city initiatives:** As cities become increasingly connected and smart, the UTTS could integrate with smart city initiatives to improve transportation planning and management. For example, the UTTS could use real-time data from smart city sensors to optimize transportation routes and schedules, reduce congestion, and improve the overall efficiency of transportation systems.

Overall, the future scope for the UTTS is vast and promising, with opportunities for innovation, expansion, and collaboration with transportation providers and smart city initiatives.

## 4.2 Conclusion:

In conclusion, the Unified Travelling and Transport System (UTTS) is a promising platform for booking and managing transportation services, providing a unified and efficient experience for users.

The system enables users to search for and book transportation services based on their travel requirements, providing real-time information on availability, schedules, and pricing. The UTTS incorporates advanced security measures to protect user data and employs a database management system for efficient data storage and retrieval.

The UTTS has numerous applications in various transportation industries, including public transportation, air travel, ride-sharing services, travel agencies, logistics and delivery services, and more. The UTTS also has significant potential for future development and expansion, including integration with emerging technologies, expansion to new markets, collaboration with transportation providers, and integration with smart city initiatives.

Overall, the UTTS is a powerful tool for streamlining transportation planning and management, improving the efficiency and convenience of transportation services for users.

## References

1. Python documentation: https://www.python.org/doc/
2. Database Systems: Design, Implementation, and Management by Carlos Coronel, Steven Morris, and Peter Rob.
3. "Developing a Transportation Information System with Python and MongoDB" by Kunpeng Zhang and Han Liu, International Journal of Web Information Systems, Vol. 15, No. 1, January 2019.
4. Python Crash Course: A Hands-On, Project-Based Introduction to Programming, By Eric Matthes
5. Learn Python the Hard Way: 3rd Edition
6. T.R. Padmanabhan, Programming with Python, Springer, 1st Ed., 2016.