**A PROJECT REPORT**

**ON**

**BUS**

**BOOKING**

**SYSTEM**

[For C.B.S.E Examination 2022-23]

**SUBMITTED BY:**

**NAME: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**ROLLNO.: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**UNDER THE GUIDANCE OF: Ms. Sapna Gupta**

**PGT (COMP.SC)**

**CERTIFICATE**

This is to certify that the Project / Dissertation entitled Bus Booking System is a bona fide work done by **Kevin Jose** of class XII Session 2022-23 has been carried out under my direct supervision and guidance. This report or a similar report on the topic has not been submitted for any other examination and does not form a part of any other course undergone by the candidate.

…………………………………………………….. ………………………………..…………………………………..

**Signature of Student Signature of Teacher/Guide**

**Name: Name:** Ms. Sapna Gupta

**Roll No.: Design.:** PGT (Comp.Sc.)

**ACKNOWLEDGEMENT**

I, undertook this Project work, as the part of my XII-Informatics Practices course. I had tried to apply my best of knowledge and experience, gained during the study and class work experience. However, developing software system is generally a quite complex and time-consuming process. It requires a systematic study, insight vision and professional approach during the design and development. Moreover, the developer always feels the need, the help and good wishes of the people near you, who have considerable experience and idea. I would like to extend my sincere thanks and gratitude to my teacher **Ms. Sapna Gupta**, for giving valuable time and moral support to develop this software. I also feel indebted to my friends for the valuable suggestions during the project work.

|  |  |  |
| --- | --- | --- |
| SNO. | TOPIC | PAGE NO. |
| 1. | Introduction |  |
| 2. | System Analysis |  |
| 3. | Theoretical Background |  |
| 4. | System Design and Development |  |
| 5. | Source Code |  |
| 6. | Output |  |
| 7. | References |  |

**CONTENTS**

**INTRODUCTION**

This software project is developed to automate the functionalities of a travel agency. The purpose of the software project is to develop a System to automate the record keeping of tickets, schedules, admins. A booking system mainly consists of a computerized database, a collection of interrelated tables for a particular subject or purpose, capable to produce different reports relevant to the user or admin.

An application program is tied with the database for easy access and interface to the database. Using Application program or front-end, we can store, retrieve and manage all information in proper way. This software, being simple in design and user-friendly, does not require much of training to users, and can be used as a powerful tool for automating a **BUS BOOKING SYSTEM**.

During coding and design of the software Project, PyCharm IDE and Python IDLE which are powerful front-end tools were used for the writing the source code. As a back-end a powerful, open-source RDBMS, MySQL is used as per requirement of the CBSE curriculum of Computer Science.

**SYSTEM ANALYSIS**

**The Hardware used:**

While developing the system, the used hardware is:

PC with Intel(R) Core (TM) i5-7200U processor having 8.00 GB RAM, 64-bit Operating System and other required devices.

**The Softwares used:**

* Microsoft Windows® 10 Pro as Operating System.
* Python 3.8 as Front-end Development environment.
* MySQL as Back-end Sever with Database for Testing.
* MS-Word 2019 for documentation.

**THEORETICAL**

**BACKGROUND**

**What is Database?**

**Introduction and Concepts:** A database is a collection of information related to a particular subject or purpose, such as tracking customer orders or maintaining a music collection. Using any RDBMS application software like MS SQL Server, MySQL, Oracle, Sybase etc, you can manage all your information from a single database file. Within the file, divide your data into separate storage containers called tables. You may and retrieve the data using queries.

A table is a collection of data about a specific topic, such as products or suppliers. Using a separate table for each topic means you can store that data only once, which makes your database more efficient and reduces data-entry errors. Table organises data into columns (called fields) and rows (called records).

A Primary key is one or more fields whose value or values uniquely identify each record in a table. In a relationship, a primary key is used to refer to specific record in one table from another table. A primary key is called foreign key when it is referred to from another table.

To find and retrieve just the data that meets conditions you specify, including data from multiple tables, create a query. A query can also update or delete multiple records at the same time, and perform built-in or custom calculations on your data.

**Role of RDBMS Application Program:**

A computer database works as a electronic filing system, which has a large number of ways of cross-referencing, and this allows the user many different ways in which to re-organize and retrieve data. A database can handle business inventory, accounting and filing and use the information in its files to prepare summaries, estimates and other reports. The management of data in a database system is done by means of a general-purpose software package called a Database Management System (DBMS). Some commercially available DBMS are MS SQL Server, MS ACCESS, INGRES, ORACLE, and Sybase. A database management system, therefore, is a combination of hardware and software that can be used to set up and monitor a database, and can manage the updating and retrieval of database that has been stored in it. Most of the database management systems have the following capabilities:

* Creating of a table, addition, deletion, modification of records.
* Retrieving data collectively or selectively.
* The data stored can be sorted or indexed at the user's discretion and direction.
* Various reports can be produced from the system. These may be either standardized report or that may be specifically generated according to specific user definition.
* Mathematical functions can be performed and the data stored in the database can be manipulated with these functions to perform the desired calculations.
* To maintain data integrity and database use. The DBMS interprets and processes users' requests to retrieve information from a database. In most cases, a query request will have to penetrate several layers of software in the DBMS and operating system before the physical database can be accessed. The DBMS responds to a query by invoking the appropriate subprograms, each of which performs its special function to interpret the query, or to locate the desired data in the database and present it in the desired order.

**What is My SQL?**

The management of data in a database system is done by means of a general-purpose software package called a Database Management System (DBMS). Some commercially available RDBMS are MS SQL Server, MS ACCESS, INGRES, ORACLE, and Sybase. MySQL, the most popular Open-Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. MySQL is named after co-founder Monty Widenius's daughter, My. The name of the MySQL Dolphin (our logo) is “Sakila”.

* MySQL is a database management system. A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.
* MySQL software is Open Source. Open-Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License),
* The MySQL Database Server is very fast, reliable, and easy to use. If that is what you are looking for, you should give it a try. MySQL Server also has a practical set of features developed in close cooperation with our users. You can find a performance comparison of MySQL Server with other database managers on our benchmark page. MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.
* MySQL Server works in client/server or embedded systems. The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs). The Main Features of MySQL
* Written in C and C++.
* Works on many different platforms.
* Uses multi-layered server design with independent modules.
* Provides transactional and non-transactional storage engines.
* Designed to make it relatively easy to add other storage engines. This is useful if you want to provide an SQL interface for an in-house database.
* Uses a very fast thread-based memory allocation system.
* Executes very fast joins using an optimized nested-loop join.
* Password security by encryption of all password traffic when you connect to a server.
* Support for large databases. We use MySQL Server with databases that contain 50 million records. We also know of users who use MySQL Server with 200,000 tables and about 5,000,000,000 rows.
* The Connector/ODBC (My ODBC) interface provides MySQL support for client programs that use ODBC (Open Database Connectivity) connections.
* The Connector/J interface provides MySQL support for Java client programs that use JDBC connections. Clients can be run on Windows or Unix. Connector/J source is available.

What is Python? Python is an open source, object-oriented high-level programming language developed by Guido Van Rossum in 1991 at the National Research Institute for Mathematics, Netherlands. Features of Python:

* It is an interactive, interpreted language.
* It is a loosely typed object –oriented language.
* It is a free open –source and portable language.
* It supports GUI.
* It can be easily compatible with other languages like C, C++ etc.
* It is used for both scientific and non-scientific programming

**Installing Python:**

It can be installed by using website: <https://www.python.org/downloads/>

**Interacting with Python:**

Python programs can be run in two ways:

* Using Command line window
* Using IDLE
* Using any other Python IDE (For eg. PyCharm etc.)

**SYSTEM DESIGN AND DEVELOPMENT**

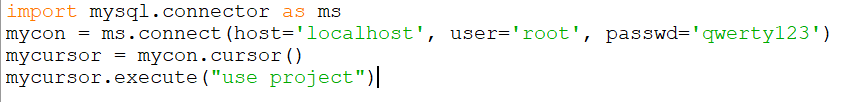
**1.Modules Used:**

* **pickle Module**

Pickle in Python is primarily used in [serializing and deserializing a Python object structure](https://wiki.python.org/moin/UsingPickle). In other words, it’s the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions, or transport data over the network.

* **mysql.connector Module**

MySQL Connector/Python enables Python programs to access MySQL databases, using an API that is compliant with the Python Database API Specification v2.0 (PEP 249). It is written in pure Python and does not have any dependencies except for the Python Standard Library.

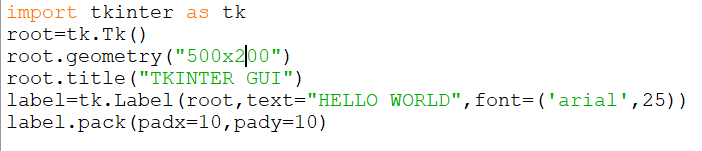


* **os Module**

The OS module in Python provides functions for interacting with the operating system. OS comes under Python’s standard utility modules. This module provides a portable way of using operating system-dependent functionality. The \*os\* and \*os.path\* modules include many functions to interact with the file system.

* **tkniter Module**

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.



* **random Module**

Python Random module is an in-built module of Python which is used to generate random numbers. These are pseudo-random numbers means these are not truly random. This module can be used to perform random actions such as generating random numbers, print random a value for a list or string, etc.

**2.Functions Used:**

* **Built-in functions:**

def(), open(), range(), close(), len(), int(), str()

* **User defined functions:**

Some of the main user defined functions that are used during the course of making this project are mentioned below:

**1.main()** – it starts the main program and displays the option “Admin” and “User”

**2.admin()** – it consists of two functions **check\_passwd()** and **admin\_home\_page()**. Checks if the admin is valid or not.

**3.insert\_schedule\_interface()** – it adds a schedule into the MySQL data base and the binary file as entered by the admin. Also consists of a function **add\_schedule()**.

**4.update\_schedule\_info\_interface()** – updates the schedule as selected by the admin. Has different options of updating. Consists of eleven other functions.

**updating\_options\_schedule()**, **new\_dep()**, **up\_dep()** etc.

**5.display\_per\_bus\_interface()** – displays all the tickets booked for a particular bus code.

**6**.**admin\_log\_out** – logs out of the admin account.

**7.user\_home\_page()** – displays the functions available for the user.

**8**.**book\_tickets\_interface()** – it takes all the required inputs from the user to buy a ticket then display the available buses for a particular departure and destination. A payment function has also been added to this. A ticket is generated with the option of downloading the ticket as a text file at a user set location. Includes the following functions:

**display\_schedule\_information(), payment\_method(), book\_tickets(), ticket\_generator(), download\_ticket().**

**9.search\_ticket\_information()** – displays the ticket of a particular ticket number given by the user. Includes one other function: **display\_ticket\_information().**

**10.update\_ticket\_information()** – updates a chosen information about the user/customer of a ticket already booked. Includes a few other functions like,

**update\_ticket\_information\_interface(), new\_name(),up\_name() etc.**

**11.ticket\_cancellation\_interface()** – cancels the ticket booked of a particular ticket number given by the user. Includes one other function **cancel\_ticket().**

**12.user\_exit()** – closes the user home page and takes the customer back to the main home page.

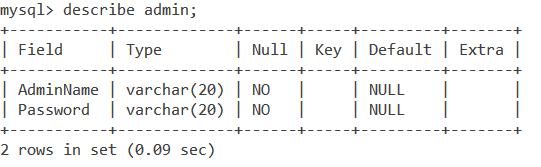
**13**.**delete\_daily\_database\_records()** – this function deletes the records from both the databases who’s date of boarding is passed.

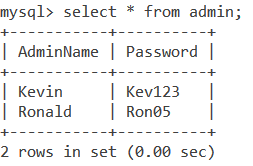
**3.Database Design:**

An important aspect of system design is the design of data storage structure. To begin with a logical model of data structure is developed first. A database is a container object which contains tables, queries, reports and data validation policies enforcement rules or constraints etc. A logical data often represented as records are kept in different tables after reducing anomalies and redundancies. The goodness of data base design lies in the table structure and its relationship. This software project maintains a database named **PROJECT** which contains the following tables.

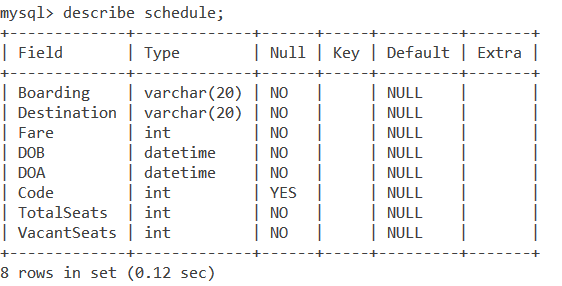
**TABLE DESIGN:** The database of BUS BOOKING SYSTEM contains 3 tables in database Library. The tables are normalized to minimize the redundancies of data and enforcing the validation rules of the organization. Most of the tables are designed to store master records. The tables and their structure are given below.

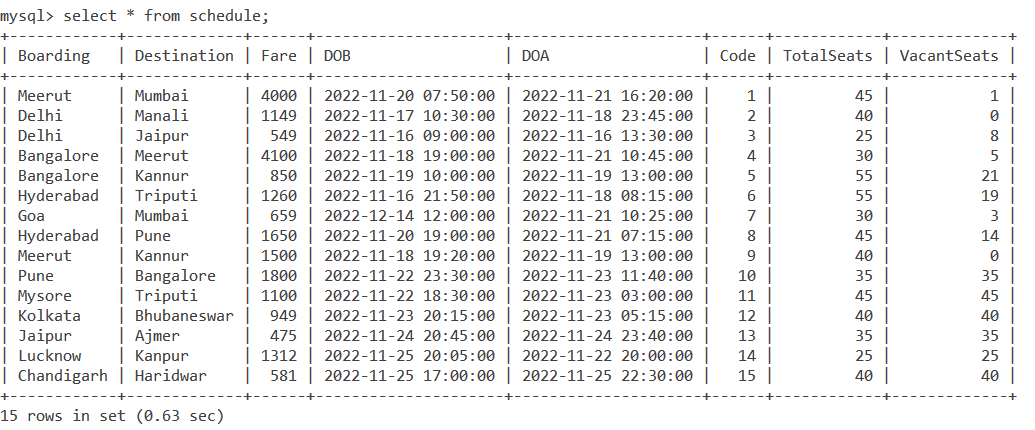
* **Table 1 : Admin**



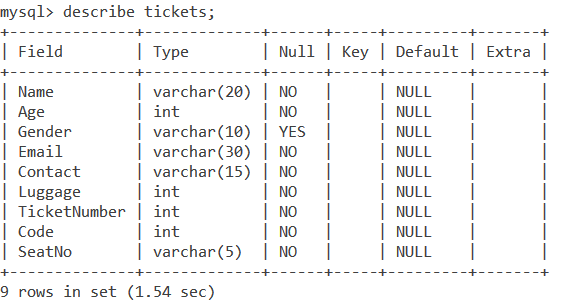


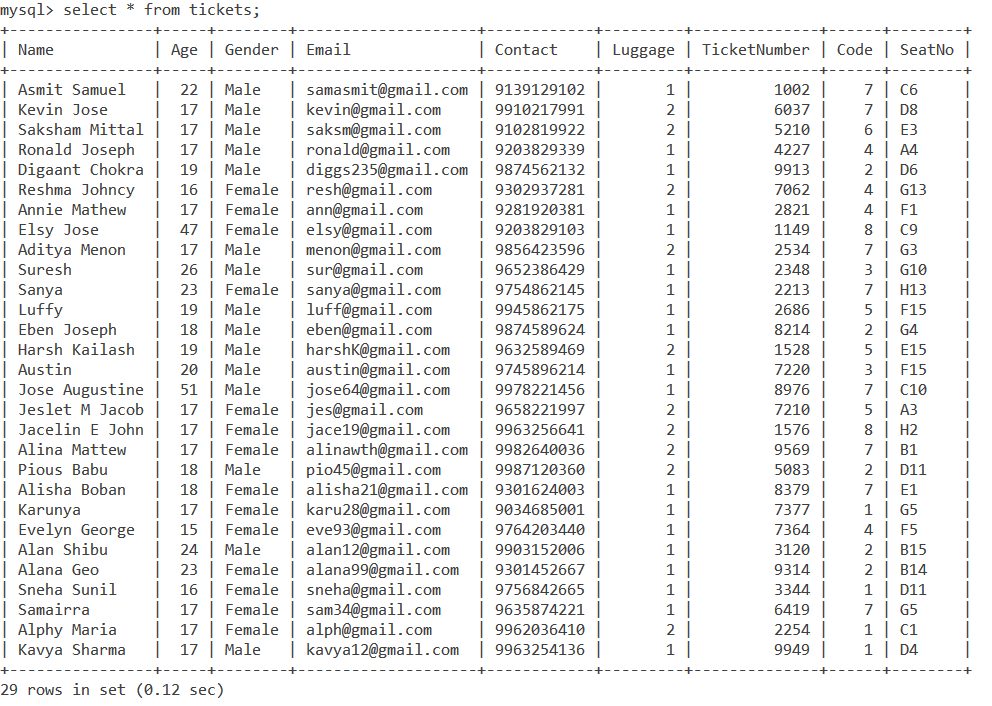
* **Table 2 : Schedule**





* **Table 3 : Tickets**



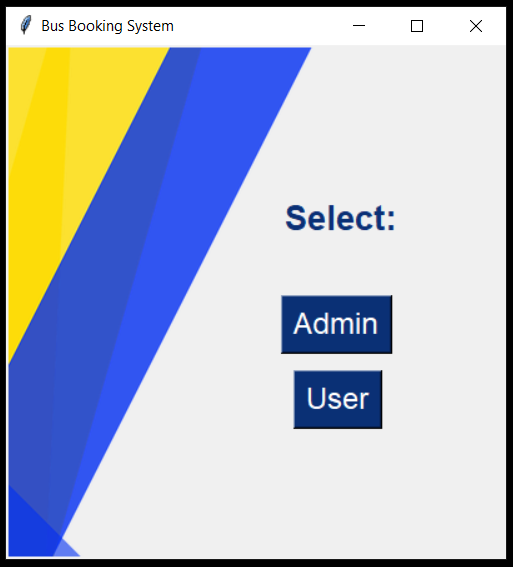


**SOURCE CODE**

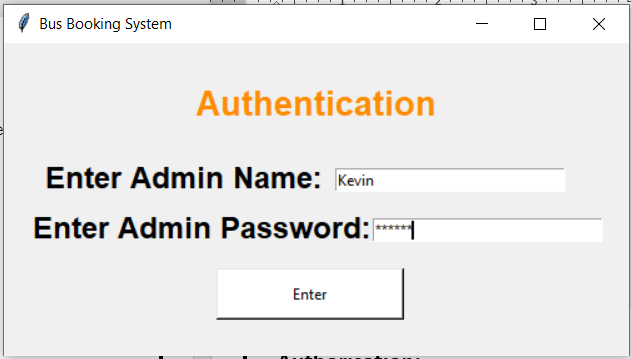
*# IMPORTING REQUIRED MODULES***import** mysql.connector **as** ms  
**import** tkinter **as** tk  
**from** tkinter **import** messagebox  
**from** tkinter **import** filedialog  
**import** random  
**import** pickle  
**import** os  
  
  
*# CONNECTING TO MYSQL*mycon = ms.connect(host=**'localhost'**, user=**'root'**, passwd=**'qwerty123'**)  
mycursor = mycon.cursor()  
mycursor.execute(**"use project"**)  
  
  
*# MAIN HOME PAGE***def** main():  
 **global** root  
 root = tk.Tk()  
 root.geometry(**"400x411"**)  
 root.title(**"Bus Booking System"**)  
 root.minsize(400, 411)  
 root.maxsize(400, 411)  
 img = tk.PhotoImage(name=**'image'**, file=**"bg.png"**)  
 img\_label = tk.Label(root, image=img)  
 img\_label.place(x=0, y=0)  
 label1 = tk.Label(root, text=**"Select:"**, font=(**"Arial Bold"**, 20), fg=**'#0A3075'**)  
 label1.place(x=220, y=120)  
 button1 = tk.Button(root, text=**"Admin"**, font=(**"Arial "**, 18), fg=**'white'**, bg=**'#0A3075'**, command=admin)  
 button1.place(x=220, y=200)  
 button2 = tk.Button(root, text=**"User"**, font=(**"Arial "**, 18), fg=**'white'**, bg=**'#0A3075'**, command=user\_home\_page)  
 button2.place(x=230, y=260)  
 root.mainloop()  
  
  
*# ADMIN AUTHENTICATION***def** admin():  
 root.destroy()  
 **global** root1  
 root1 = tk.Tk()  
 root1.geometry(**"500x250"**)  
 root1.title(**"Bus Booking System"**)  
 au = tk.Label(root1, text=**'Authentication'**, font=**'Arial 20 bold'**, fg=**'dark orange'**)  
 au.pack(padx=40, pady=30)  
 uname = tk.Label(root1, text=**"Enter Admin Name:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 uname.place(x=30, y=90)  
 **global** adname\_ent  
 adname\_ent = tk.Entry(root1, width=30)  
 adname\_ent.place(x=265, y=100)  
 passwd = tk.Label(root1, text=**"Enter Admin Password:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 passwd.place(x=20, y=130)  
 **global** passwd\_ent  
 passwd\_ent = tk.Entry(root1, width=30,show=**"\*"**)  
 passwd\_ent.place(x=295, y=140)  
 button3 = tk.Button(root1, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=check\_passwd)  
 button3.place(x=170, y=180)  
  
  
*# CHECKING PASSWORD***def** check\_passwd():  
 adminname = adname\_ent.get()  
 pswd = passwd\_ent.get()  
 mycursor.execute(**"select \* from admin"**)  
 mydata = mycursor.fetchall()  
 flag = **False  
 for** x **in** mydata:  
 **if** x[0] == adminname **and** x[1] == pswd:  
 flag = **True** messagebox.showinfo(title=**"HI"**, message=**f"Welcome {**adminname**}"**)  
 admin\_home\_page()  
 **if not** flag:  
 messagebox.showerror(title=**"ACCESS DENIED!"**, message=**"Incorrect Username or Password"**)  
  
  
*# ADMIN HOME PAGE***def** admin\_home\_page():  
 **try**:  
 root1.destroy()  
 **except**:  
 **pass  
 global** root2  
 root2 = tk.Tk()  
 root2.geometry(**"410x420"**)  
 root2.title(**"Bus Booking System : Admin"**)  
 root2.protocol(**"WM\_DELETE\_WINDOW"**, on\_closing\_root2\_admin)  
 label = tk.Label(root2, text=**"Select Function:"**, font=(**"Arial Bold"**, 20), fg=**'dark orange'**)  
 label.pack(padx=30, pady=30)  
 button4 = tk.Button(root2, text=**"Add Schedules"**, width=20, height=2, bg=**'white'**,  
 command=insert\_schedule\_interface)  
 button4.place(x=125, y=100)  
 button5 = tk.Button(root2, text=**"Update Schedules"**, width=20, height=2, bg=**'white'**,  
 command=update\_schedule\_info\_interface)  
 button5.place(x=125, y=180)  
 button6 = tk.Button(root2, text=**"Display Per Bus"**, width=20, height=2, bg=**'white'**,  
 command=display\_per\_bus\_interface)  
 button6.place(x=125, y=260)  
 button7 = tk.Button(root2, text=**"Log Out"**, width=20, height=2, bg=**'white'**,  
 command=admin\_log\_out)  
 button7.place(x=125, y=340)  
  
  
*# ADMIN FUNCTION 1 : ADD SCHEDULE***def** insert\_schedule\_interface():  
 root2.destroy()  
 **global** root3  
 root3 = tk.Tk()  
 root3.geometry(**"600x600"**)  
 root3.title(**"Bus Booking System"**)  
  
 label = tk.Label(root3, text=**"Enter the following details:"**, font=(**"Arial Bold"**, 18))  
 label.pack(padx=30, pady=30)  
  
 boar = tk.Label(root3, text=**"Enter Departure:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 boar.place(x=40, y=130)  
 **global** boar\_ent  
 boar\_ent = tk.Entry(root3, width=30)  
 boar\_ent.place(x=325, y=140)  
  
 des = tk.Label(root3, text=**"Enter Destination"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 des.place(x=40, y=170)  
 **global** des\_ent  
 des\_ent = tk.Entry(root3, width=30)  
 des\_ent.place(x=325, y=180)  
  
 fare = tk.Label(root3, text=**"Enter Fare:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 fare.place(x=40, y=210)  
 **global** fare\_ent  
 fare\_ent = tk.Entry(root3, width=30)  
 fare\_ent.place(x=325, y=220)  
  
 dob = tk.Label(root3, text=**"Enter Date of Departure:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 dob.place(x=40, y=250)  
 **global** dob\_ent  
 dob\_ent = tk.Entry(root3, width=30)  
 dob\_ent.place(x=325, y=260)  
  
 doa = tk.Label(root3, text=**"Enter Date of Arrival:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 doa.place(x=40, y=290)  
 **global** doa\_ent  
 doa\_ent = tk.Entry(root3, width=30)  
 doa\_ent.place(x=325, y=300)  
  
 code = tk.Label(root3, text=**"Enter Code:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 code.place(x=40, y=330)  
 **global** code\_ent  
 code\_ent = tk.Entry(root3, width=30)  
 code\_ent.place(x=325, y=340)  
  
 ts = tk.Label(root3, text=**"Enter Total Seats:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 ts.place(x=40, y=370)  
 **global** ts\_ent  
 ts\_ent = tk.Entry(root3, width=30)  
 ts\_ent.place(x=325, y=380)  
  
 button3 = tk.Button(root3, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=add\_schedule)  
 button3.place(x=200, y=500)  
  
  
*# INSERTING INTO DATABASE/BINARY FILE***def** add\_schedule():  
 boar = boar\_ent.get()  
 dest = des\_ent.get()  
 fare = fare\_ent.get()  
 doB = dob\_ent.get()  
 doA = doa\_ent.get()  
 code = code\_ent.get()  
 ts = ts\_ent.get()  
 vs = ts  
 **try**:  
 query = **"insert into Schedule values('{}','{}',{},'{}','{}',{},{},{})"**.format(boar, dest, fare, doB, doA, code,  
 ts, ts)  
 mycursor.execute(query)  
  
 f = open(**"schedules.dat"**, **"ab"**)  
 info = [boar, dest, fare, doB, doA, code, ts, vs]  
 pickle.dump(info, f)  
 f.close()  
  
 messagebox.showinfo(title=**"Bus Booking System"**, message=**"Schedule Added"**)  
 root3.destroy()  
 admin\_home\_page()  
 mycon.commit()  
 **except**:  
 messagebox.showerror(title=**"Error"**,message=**"Invalid Input"**)  
  
  
  
*# ADMIN FUNCTION 2 : UPDATION OF SCHEDULE DATABASE/BINARY FILE***def** update\_schedule\_info\_interface():  
 root2.destroy()  
 **global** root3  
 root3 = tk.Tk()  
 root3.geometry(**"400x250"**)  
 root3.title(**"Bus Booking System"**)  
  
 label = tk.Label(root3, text=**"Updation:"**, font=(**"Arial Bold"**, 20), fg=**'dark orange'**)  
 label.pack(padx=30, pady=30)  
  
 code = tk.Label(root3, text=**"Enter Code:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 code.place(x=25, y=100)  
 **global** code\_ent1  
 code\_ent1 = tk.Entry(root3, width=30)  
 code\_ent1.place(x=175, y=110)  
  
 button = tk.Button(root3, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=updating\_options\_schedule)  
 button.place(x=120, y=160)  
  
  
*# UPDATING OPTIONS***def** updating\_options\_schedule():  
 mycursor.execute(**"select \* from schedule"**)  
 mydata = mycursor.fetchall()  
 code1 = code\_ent1.get()  
 c = 0  
 **try**:  
 **for** b, d, fa, dob, doa, co, ts, vs **in** mydata:  
 **if** co == int(code1):  
 **global** cod  
 cod = co  
 c += 1  
 root3.destroy()  
 **global** root4  
 root4 = tk.Tk()  
 root4.geometry(**"400x430"**)  
 root4.title(**"Bus Booking System"**)  
 label = tk.Label(root4, text=**"Select your choice:"**, font=(**"Arial Bold"**, 18), fg=**'dark orange'**)  
 label.pack(padx=30, pady=30)  
  
 button1 = tk.Button(root4, text=**"Departure"**, width=20, height=2, bg=**'white'**, command=new\_dep)  
 button1.place(x=130, y=100)  
  
 button2 = tk.Button(root4, text=**"Destination"**, width=20, height=2, bg=**'white'**, command=new\_des)  
 button2.place(x=130, y=150)  
  
 button3 = tk.Button(root4, text=**"Fare"**, width=20, height=2, bg=**'white'**, command=new\_fare)  
 button3.place(x=130, y=200)  
  
 button4 = tk.Button(root4, text=**"Date of Boarding"**, width=20, height=2, bg=**'white'**, command=new\_dob)  
 button4.place(x=130, y=250)  
  
 button5 = tk.Button(root4, text=**"Date of Arrival"**, width=20, height=2, bg=**'white'**, command=new\_doa)  
 button5.place(x=130, y=300)  
 **if** c == 0:  
 messagebox.showerror(title=**"Error"**, message=**"No matches found"**)  
 **except**:  
 messagebox.showerror(title=**"Error"**, message=**"Invalid Input"**)  
  
  
*# OPTION 1 : DEPARTURE***def** new\_dep():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"490x175"**)  
 root5.title(**"Bus Booking System:Update Schedule"**)  
 dep = tk.Label(root5, text=**"Enter New Departure:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 dep.place(x=15, y=40)  
 **global** dep\_ent  
 dep\_ent = tk.Entry(root5, width=30)  
 dep\_ent.place(x=270, y=50)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_dep)  
 button.place(x=170, y=100)  
  
  
*# DEPARTURE UPDATION***def** up\_dep():  
 ndep = dep\_ent.get()  
 mycursor.execute(**"update schedule\  
 set Boarding='%s'\  
 where code=%d "** % (ndep, cod))  
 f = open(**"schedules.dat"**, **"rb+"**)  
 **while True**:  
 **try**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[5] == cod:  
 mydata[0] = ndep  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 messagebox.showinfo(title=**""**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 admin\_home\_page()  
 mycon.commit()  
  
  
*# OPTION 2 : DESTINATION***def** new\_des():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"490x175"**)  
 root5.title(**"Bus Booking System"**)  
 des = tk.Label(root5, text=**"Enter New Destination:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 des.place(x=10, y=40)  
 **global** des\_ent  
 des\_ent = tk.Entry(root5, width=30)  
 des\_ent.place(x=280, y=50)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_des)  
 button.place(x=170, y=100)  
  
  
*# DESTINATION UPDATION***def** up\_des():  
 ndes = des\_ent.get()  
 mycursor.execute(**"update schedule\  
 set Destination='%s'\  
 where code=%d "** % (ndes, cod))  
 f = open(**"schedules.dat"**, **"rb+"**)  
 **while True**:  
 **try**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[5] == cod:  
 mydata[1] = ndes  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 messagebox.showinfo(title=**""**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 admin\_home\_page()  
 mycon.commit()  
  
  
*# OPTION 3 : FARE***def** new\_fare():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"490x175"**)  
 root5.title(**"Bus Booking System"**)  
 fa = tk.Label(root5, text=**"Enter New Fare:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 fa.place(x=45, y=40)  
 **global** fa\_ent  
 fa\_ent = tk.Entry(root5, width=30)  
 fa\_ent.place(x=235, y=50)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_fare)  
 button.place(x=170, y=100)  
  
  
*# FARE UPDATION***def** up\_fare():  
 nfa = fa\_ent.get()  
 mycursor.execute(**"update schedule\  
 set Fare=%s\  
 where code=%d "** % (nfa, cod))  
 f = open(**"schedules.dat"**, **"rb+"**)  
 **while True**:  
 **try**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[5] == cod:  
 mydata[2] = nfa  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 messagebox.showinfo(title=**""**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 admin\_home\_page()  
 mycon.commit()  
  
  
*# OPTION 4 : DATE OF BOARDING***def** new\_dob():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"490x195"**)  
 root5.title(**"Bus Booking System"**)  
 dob = tk.Label(root5, text=**"Enter New Date of Boarding:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 dob.place(x=85, y=40)  
 **global** dob\_ent  
 dob\_ent = tk.Entry(root5, width=30)  
 dob\_ent.place(x=155, y=90)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_dob)  
 button.place(x=170, y=135)  
  
  
*# DATE OF BOARDING UPDATION***def** up\_dob():  
 ndob = dob\_ent.get()  
 mycursor.execute(**"update schedule\  
 set DOB='%s'\  
 where code=%d "** % (ndob, cod))  
 f = open(**"schedules.dat"**, **"rb+"**)  
 **while True**:  
 **try**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[5] == cod:  
 mydata[3] = ndob  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 messagebox.showinfo(title=**""**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 admin\_home\_page()  
 mycon.commit()  
  
  
*# OPTION 5 : DATE OF ARRIVAL***def** new\_doa():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"490x195"**)  
 root5.title(**"Bus Booking System"**)  
 doa = tk.Label(root5, text=**"Enter New Date of Arrival:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 doa.place(x=95, y=40)  
 **global** doa\_ent  
 doa\_ent = tk.Entry(root5, width=30)  
 doa\_ent.place(x=155, y=90)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_doa)  
 button.place(x=170, y=135)  
  
  
*# DATE OF ARRIVAL UPDATION***def** up\_doa():  
 ndoa = doa\_ent.get()  
 mycursor.execute(**"update schedule\  
 set DOA='%s'\  
 where code=%d "** % (ndoa, cod))  
 f = open(**"schedules.dat"**, **"rb+"**)  
 **while True**:  
 **try**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[5] == cod:  
 mydata[4] = ndoa  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 messagebox.showinfo(title=**""**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 admin\_home\_page()  
 mycon.commit()  
  
  
*# ADMIN FUNCTION 3 : DISPLAYING OF TICKETS BOOKED FOR A PARTICULAR BUS  
# TAKES THE BUS CODE AND DISPLAYS ALL THE TICKETS BOOKED FOR THAT BUS***def** display\_per\_bus\_interface():  
 root2.destroy()  
 **global** root3  
 root3 = tk.Tk()  
 root3.geometry(**"420x180"**)  
 root3.title(**"Bus Booking System"**)  
 code = tk.Label(root3, text=**"Enter Code:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 code.place(x=40, y=50)  
 **global** code\_ent2  
 code\_ent2 = tk.Entry(root3, width=30)  
 code\_ent2.place(x=190, y=60)  
 button = tk.Button(root3, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=display\_per\_bus)  
 button.place(x=130, y=100)  
  
  
*# DISPLAYS TICKETS***def** display\_per\_bus():  
 **try**:  
 mycursor.execute(**"select \* from tickets where code=%s"** % int(code\_ent2.get()))  
 mydata = mycursor.fetchall()  
 **if** mydata == []:  
 messagebox.showinfo(title=**"Info"**, message=**"No tickets booked"**)  
 **else**:  
 root3.destroy()  
 **global** root4  
 root4 = tk.Tk()  
 root4.geometry(**"1000x400"**)  
 root4.title(**"Bus Booking System"**)  
 code = tk.Label(root4, text=**"Tickets Booked:"**, font=**'Arial 18 bold'**, fg=**'orange'**)  
 code.pack(padx=30, pady=30)  
  
 *# calls main() on closing window* root4.protocol(**"WM\_DELETE\_WINDOW"**, on\_closing\_root4\_admin)  
  
 labelframe = tk.Frame(root4)  
 labelframe.columnconfigure(0, weight=1)  
 labelframe.columnconfigure(1, weight=1)  
 labelframe.columnconfigure(2, weight=1)  
 labelframe.columnconfigure(3, weight=1)  
 labelframe.columnconfigure(4, weight=1)  
 labelframe.columnconfigure(5, weight=1)  
 labelframe.columnconfigure(6, weight=1)  
 labelframe.columnconfigure(7, weight=1)  
 labelframe.columnconfigure(8, weight=1)  
 labelframe.columnconfigure(9, weight=1)  
  
 x = 0  
 name = tk.Label(labelframe, text=**"Name"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 name.grid(row=x, column=0, sticky=tk.W + tk.E)  
 age = tk.Label(labelframe, text=**"Age"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 age.grid(row=x, column=1, sticky=tk.W + tk.E)  
 gen = tk.Label(labelframe, text=**"Gender"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 gen.grid(row=x, column=2, sticky=tk.W + tk.E)  
 email = tk.Label(labelframe, text=**"Email"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 email.grid(row=x, column=3, sticky=tk.W + tk.E)  
 con = tk.Label(labelframe, text=**"Contact"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 con.grid(row=x, column=4, sticky=tk.W + tk.E)  
 lugg = tk.Label(labelframe, text=**"Luggage"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 lugg.grid(row=x, column=5, sticky=tk.W + tk.E)  
 tkn1 = tk.Label(labelframe, text=**"Ticket Number"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 tkn1.grid(row=x, column=6, sticky=tk.W + tk.E)  
 code1 = tk.Label(labelframe, text=**"Code"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 code1.grid(row=x, column=7, sticky=tk.W + tk.E)  
 sno = tk.Label(labelframe, text=**"Seat Number"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 sno.grid(row=x, column=8, sticky=tk.W + tk.E)  
  
 r = 1  
 **for** n, a, g, e, c, l, tkn, co, sn **in** mydata:  
 name = tk.Label(labelframe, text=n, font=**'Arial 12 '**, fg=**'black'**)  
 name.grid(row=r, column=0, sticky=tk.W + tk.E)  
 age = tk.Label(labelframe, text=a, font=**'Arial 12 '**, fg=**'black'**)  
 age.grid(row=r, column=1, sticky=tk.W + tk.E)  
 gen = tk.Label(labelframe, text=g, font=**'Arial 12 '**, fg=**'black'**)  
 gen.grid(row=r, column=2, sticky=tk.W + tk.E)  
 email = tk.Label(labelframe, text=e, font=**'Arial 12 '**, fg=**'black'**)  
 email.grid(row=r, column=3, sticky=tk.W + tk.E)  
 con = tk.Label(labelframe, text=c, font=**'Arial 12 '**, fg=**'black'**)  
 con.grid(row=r, column=4, sticky=tk.W + tk.E)  
 lugg = tk.Label(labelframe, text=l, font=**'Arial 12 '**, fg=**'black'**)  
 lugg.grid(row=r, column=5, sticky=tk.W + tk.E)  
 tkn = tk.Label(labelframe, text=tkn, font=**'Arial 12 '**, fg=**'black'**)  
 tkn.grid(row=r, column=6, sticky=tk.W + tk.E)  
 code1 = tk.Label(labelframe, text=co, font=**'Arial 12 '**, fg=**'black'**)  
 code1.grid(row=r, column=7, sticky=tk.W + tk.E)  
 sno = tk.Label(labelframe, text=sn, font=**'Arial 12 '**, fg=**'black'**)  
 sno.grid(row=r, column=8, sticky=tk.W + tk.E)  
 r += 1  
 labelframe.pack(fill=**'x'**)  
 **except**:  
 messagebox.showerror(title=**"Error"**,message=**"Invalid Input"**)  
  
  
  
*# ADMIN FUNCTION 4 : LOGGING OUT***def** admin\_log\_out():  
 **if** messagebox.askyesno(title=**"Log Out"**, message=**"Want to log out?"**):  
 root2.destroy()  
 main()  
  
  
*# USER HOME PAGE***def** user\_home\_page():  
 **try**:  
 root.destroy()  
 **except**:  
 **pass  
  
 global** root2  
 root2 = tk.Tk()  
 root2.geometry(**"500x600"**)  
 root2.title(**"Bus Booking System : User"**)  
 label = tk.Label(root2, text=**"Select Function:"**, font=(**"Arial Bold"**, 18), fg=**'dark orange'**)  
 label.pack(padx=30, pady=30)  
 button4 = tk.Button(root2, text=**"Book Tickets"**, width=20, height=2, bg=**'white'**,  
 command=book\_tickets\_interface)  
 button4.place(x=170, y=100)  
 button5 = tk.Button(root2, text=**"Search Ticket"**, width=20, height=2, bg=**'white'**,  
 command=search\_ticket\_information)  
 button5.place(x=170, y=200)  
 button6 = tk.Button(root2, text=**"Update Ticket"**, width=20, height=2, bg=**'white'**,  
 command=update\_ticket\_information)  
 button6.place(x=170, y=300)  
 button7 = tk.Button(root2, text=**"Cancel Booking"**, width=20, height=2, bg=**'white'**,  
 command=ticket\_cancellation\_interface)  
 button7.place(x=170, y=400)  
 button8 = tk.Button(root2, text=**"Exit"**, width=20, height=2, bg=**'white'**,  
 command=user\_exit)  
 button8.place(x=170, y=500)  
  
  
*# USER FUNCTION 1 : BOOK TICKETS***def** book\_tickets\_interface():  
 **global** root3  
 root3 = tk.Tk()  
 root3.geometry(**"600x600"**)  
 root3.title(**"Bus Booking System:Book Tickets"**)  
 root2.destroy()  
 label = tk.Label(root3, text=**"Enter the following details:"**, font=(**"Arial Bold"**, 18), fg=**'Orange'**)  
 label.pack(padx=30, pady=30)  
  
 name1 = tk.Label(root3, text=**"Enter Name:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 name1.place(x=40, y=130)  
 **global** name\_ent1  
 name\_ent1 = tk.Entry(root3, width=30)  
 name\_ent1.place(x=325, y=140)  
  
 age1 = tk.Label(root3, text=**"Enter Age"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 age1.place(x=40, y=170)  
 **global** age\_ent1  
 age\_ent1 = tk.Entry(root3, width=30)  
 age\_ent1.place(x=325, y=180)  
  
 gender1 = tk.Label(root3, text=**"Enter Gender:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 gender1.place(x=40, y=210)  
 **global** gen\_ent1  
 gen\_ent1 = tk.Entry(root3, width=30)  
 gen\_ent1.place(x=325, y=220)  
  
 email1 = tk.Label(root3, text=**"Enter Email:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 email1.place(x=40, y=250)  
 **global** em\_ent1  
 em\_ent1 = tk.Entry(root3, width=30)  
 em\_ent1.place(x=325, y=260)  
  
 mob1 = tk.Label(root3, text=**"Enter Mobile:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 mob1.place(x=40, y=290)  
 **global** mob\_ent1  
 mob\_ent1 = tk.Entry(root3, width=30)  
 mob\_ent1.place(x=325, y=300)  
  
 lugg1 = tk.Label(root3, text=**"Enter Luggage:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 lugg1.place(x=40, y=330)  
 **global** lugg\_ent1  
 lugg\_ent1 = tk.Entry(root3, width=30)  
 lugg\_ent1.place(x=325, y=340)  
  
 fro = tk.Label(root3, text=**"Enter From:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 fro.place(x=40, y=370)  
 **global** fro\_ent  
 fro\_ent = tk.Entry(root3, width=30)  
 fro\_ent.place(x=325, y=380)  
  
 to = tk.Label(root3, text=**"Enter To:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 to.place(x=40, y=410)  
 **global** to\_ent  
 to\_ent = tk.Entry(root3, width=30)  
 to\_ent.place(x=325, y=420)  
  
 button3 = tk.Button(root3, text=**"Next"**, width=20, height=2, bg=**'white'**, command=display\_schedule\_information)  
 button3.place(x=200, y=500)  
  
  
*# DISPLAYING THE AVAILABLE BUSES***def** display\_schedule\_information():  
 **global** name, age, gender, email, mobile, luggage, age1, lug1  
 name = name\_ent1.get()  
 age = str(age\_ent1.get())  
 age1 = int(age)  
 gender = gen\_ent1.get()  
 email = em\_ent1.get()  
 mobile = mob\_ent1.get()  
 luggage = str(lugg\_ent1.get())  
 lug1 = int(luggage)  
 fro\_ent1 = (fro\_ent.get()).strip()  
 to\_ent1 = (to\_ent.get()).strip()  
 root3.destroy()  
 **if** name == **"" or** age == **"" or** gender == **"" or** email == **"" or** mobile == **"" or** luggage == **""**:  
 messagebox.showerror(title=**"Error"**, message=**"All information to be entered"**)  
 **else**:  
 mycursor.execute(**"select \* from schedule\  
 where boarding='%s' and destination='%s' "** % (fro\_ent1, to\_ent1))  
 mydata = mycursor.fetchall()  
  
 **if** mydata == []:  
 messagebox.showinfo(title=**"Error"**, message=**"No buses found"**)  
 **else**:  
 **global** root6  
 root6 = tk.Tk()  
 root6.geometry(**"1000x350"**)  
 root6.title(**"Bus Booking System"**)  
 title = tk.Label(root6, text=**"AVAILABLE BUSES:"**, font=**'Arial 18 bold'**, fg=**'Orange'**)  
 title.pack(padx=30, pady=30)  
  
 **global** root7  
 root7 = tk.Tk()  
 root7.geometry(**"400x220"**)  
 root7.title(**"Bus Booking System"**)  
  
 co = tk.Label(root7, text=**"Enter code:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 co.place(x=40, y=40)  
 **global** code\_ent3  
 code\_ent3 = tk.Entry(root7, width=30)  
 code\_ent3.place(x=190, y=45)  
  
 labelframe = tk.Frame(root6)  
 labelframe.columnconfigure(0, weight=1)  
 labelframe.columnconfigure(1, weight=1)  
 labelframe.columnconfigure(2, weight=1)  
 labelframe.columnconfigure(3, weight=1)  
 labelframe.columnconfigure(4, weight=1)  
 labelframe.columnconfigure(5, weight=1)  
 labelframe.columnconfigure(6, weight=1)  
 labelframe.columnconfigure(7, weight=1)  
  
 x = 0  
 boar1 = tk.Label(labelframe, text=**"DEPARTURE"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 boar1.grid(row=x, column=0, sticky=tk.W + tk.E)  
 des1 = tk.Label(labelframe, text=**"DESTINATION"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 des1.grid(row=x, column=1, sticky=tk.W + tk.E)  
 fa1 = tk.Label(labelframe, text=**"FARE"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 fa1.grid(row=x, column=2, sticky=tk.W + tk.E)  
 dob1 = tk.Label(labelframe, text=**"DATE OF BOARDING"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 dob1.grid(row=x, column=3, sticky=tk.W + tk.E)  
 doa1 = tk.Label(labelframe, text=**"DATE OF ARRIVAL"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 doa1.grid(row=x, column=4, sticky=tk.W + tk.E)  
 co1 = tk.Label(labelframe, text=**"CODE"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 co1.grid(row=x, column=5, sticky=tk.W + tk.E)  
 ts1 = tk.Label(labelframe, text=**"TOTAL SEATS"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 ts1.grid(row=x, column=6, sticky=tk.W + tk.E)  
 vs1 = tk.Label(labelframe, text=**"VACANT SEATS"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 vs1.grid(row=x, column=7, sticky=tk.W + tk.E)  
 r = 1  
 **for** b, d, f, dob, doa, co, ts, vs **in** mydata:  
 boar = tk.Label(labelframe, text=b, font=**'Arial 12 '**, fg=**'black'**)  
 boar.grid(row=r, column=0, sticky=tk.W + tk.E)  
 des = tk.Label(labelframe, text=d, font=**'Arial 12 '**, fg=**'black'**)  
 des.grid(row=r, column=1, sticky=tk.W + tk.E)  
 fa = tk.Label(labelframe, text=f, font=**'Arial 12 '**, fg=**'black'**)  
 fa.grid(row=r, column=2, sticky=tk.W + tk.E)  
 dob = tk.Label(labelframe, text=dob, font=**'Arial 12 '**, fg=**'black'**)  
 dob.grid(row=r, column=3, sticky=tk.W + tk.E)  
 doa = tk.Label(labelframe, text=doa, font=**'Arial 12 '**, fg=**'black'**)  
 doa.grid(row=r, column=4, sticky=tk.W + tk.E)  
 co = tk.Label(labelframe, text=co, font=**'Arial 12 '**, fg=**'black'**)  
 co.grid(row=r, column=5, sticky=tk.W + tk.E)  
 ts = tk.Label(labelframe, text=ts, font=**'Arial 12 '**, fg=**'black'**)  
 ts.grid(row=r, column=6, sticky=tk.W + tk.E)  
 vs = tk.Label(labelframe, text=vs, font=**'Arial 12 '**, fg=**'black'**)  
 vs.grid(row=r, column=7, sticky=tk.W + tk.E)  
 r += 1  
 labelframe.pack(fill=**'x'**)  
  
 button3 = tk.Button(root7, text=**"Next"**, width=20, height=2, bg=**'white'**, command=payment\_procedure)  
 button3.place(x=120, y=110)  
  
  
*# PAYMENT***def** payment\_procedure():  
 mycursor.execute(**"select \* from schedule \  
 where code=%s"** % int(code\_ent3.get()))  
 mydata = mycursor.fetchall()  
 **for** dep, des, fa, dob, doa, code, ts, vs **in** mydata:  
 **if** vs == 0:  
 messagebox.showinfo(title=**"Bus Info"**, message=**"All seats are booked"**)  
 **else**:  
 **global** root8, tfa, code3  
 root8 = tk.Tk()  
 root8.geometry(**"600x300"**)  
 root8.title(**"Payment"**)  
 code3 = int(code\_ent3.get())  
 root6.destroy() *# display\_schedule\_info* root7.destroy()  
 mycursor.execute(**"select \* from schedule \  
 where code=%s"** % code3)  
 mydata = mycursor.fetchall()  
 **for** dep, des, fa, dob, doa, code, ts, vs **in** mydata:  
 **if** age1 >= 18:  
 tfa = fa + (lug1 \* 200)  
 **else**:  
 tfa = (fa \* 0.95) + (lug1 \* 150)  
 fa = tk.Label(root8, text=**"Total Fare:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 fa.place(x=40, y=40)  
 fa\_ = tk.Label(root8, text=str(tfa), font=**'Arial 18 bold'**, fg=**'black'**)  
 fa\_.place(x=300, y=40)  
  
 name2 = tk.Label(root8, text=**"Enter Name:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 name2.place(x=40, y=80)  
 name\_ent2 = tk.Entry(root8, width=30)  
 name\_ent2.place(x=300, y=90)  
  
 crno = tk.Label(root8, text=**"Enter Credit Card No.:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 crno.place(x=40, y=120)  
 crno\_ent = tk.Entry(root8, width=30)  
 crno\_ent.place(x=300, y=125)  
  
 sc = tk.Label(root8, text=**"Enter Security Code:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 sc.place(x=40, y=160)  
 sc\_ent = tk.Entry(root8, width=30)  
 sc\_ent.place(x=300, y=165)  
  
 button3 = tk.Button(root8, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=book\_tickets)  
 button3.place(x=200, y=200)  
  
  
*# ADDING THE RECORD TO THE DATABASE/BINARY FILE***def** book\_tickets():  
 root8.destroy()  
 s = [**"A"**, **"B"**, **"C"**, **"D"**, **"E"**, **"F"**, **"G"**, **"H"**]  
 c = random.randint(1, 15)  
 i = random.randint(0, 7)  
 seat = s[i] + str(c)  
 **global** ticketno  
 ticketno = random.randint(1000, 9999)  
 query = **"insert into tickets values('{}',{},'{}','{}','{}',{},{},{},'{}')"**.format(name, age, gender, email,  
 mobile, luggage, ticketno,  
 code3, seat)  
 mycursor.execute(query)  
 mycursor.execute(**"select \* from schedule where code=%s"** % code3)  
 mydata = mycursor.fetchall()  
 **for** b, d, f, dob, doa, co, ts, vs **in** mydata:  
 **while** vs > 0:  
 vs -= 1  
 mycursor.execute(**"update schedule set vacantseats=%s where code=%d"** % (vs, code3))  
 **break** mycon.commit()  
  
 f = open(**"ticket1.dat"**, **"ab"**)  
 info = [name, age, gender, email, mobile, luggage, ticketno, code3, seat]  
 pickle.dump(info, f)  
 f.close()  
  
 *# vacant seats decrement by 1 after booking* f = open(**"schedules.dat"**, **"rb+"**)  
 **while True**:  
 **try**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[5] == code3:  
 mydata[7] = int(mydata[7] - 1)  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
  
 ticket\_generator()  
 messagebox.showinfo(title=**"Bus Booking System"**, message=**"Booking Successful"**)  
  
  
*# FINAL DISPLAY OF TICKET***def** ticket\_generator():  
 mycursor.execute(**"select Name,Age,Gender,Email,Contact,\  
 Luggage,TicketNumber,SeatNo,Boarding,Destination,Fare,DOB,DOA from tickets,schedule \  
 where TicketNumber=%s and tickets.code=schedule.code"** % ticketno)  
 mydata = mycursor.fetchall()  
 **if** mydata == []:  
 messagebox.showerror(title=**"Info"**, message=**"Ticket Not Found"**)  
 **else**:  
 **global** root4  
 root4 = tk.Tk()  
 root4.geometry(**"375x520"**)  
 root4.title(**"Bus Booking System:Search Ticket"**)  
 heading = tk.Label(root4, text=**"Ticket Booked:"**, font=**'Arial 18 bold'**, fg=**'orange'**)  
 heading.place(x=100, y=30)  
  
 *# calls user() when window is closed* root4.protocol(**"WM\_DELETE\_WINDOW"**, on\_closing\_root4\_user)  
  
 labelframe = tk.Frame(root4)  
  
 y = 1  
 name = tk.Label(labelframe, text=**"Name"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**) *# dark cyan* name.grid(row=0, column=y, sticky=tk.W)  
 age = tk.Label(labelframe, text=**"Age"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 age.grid(row=1, column=y, sticky=tk.W)  
 gen = tk.Label(labelframe, text=**"Gender"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 gen.grid(row=2, column=y, sticky=tk.W)  
 email = tk.Label(labelframe, text=**"Email"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 email.grid(row=3, column=y, sticky=tk.W)  
 con = tk.Label(labelframe, text=**"Contact"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 con.grid(row=4, column=y, sticky=tk.W)  
 lugg = tk.Label(labelframe, text=**"Luggage"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 lugg.grid(row=5, column=y, sticky=tk.W)  
 tkn1 = tk.Label(labelframe, text=**"Ticket Number"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 tkn1.grid(row=6, column=y, sticky=tk.W)  
 sno = tk.Label(labelframe, text=**"Seat Number"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 sno.grid(row=7, column=y, sticky=tk.W)  
 depp = tk.Label(labelframe, text=**"From"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 depp.grid(row=8, column=y, sticky=tk.W)  
 dess = tk.Label(labelframe, text=**"To"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 dess.grid(row=9, column=y, sticky=tk.W)  
 faa = tk.Label(labelframe, text=**"Total Fare"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 faa.grid(row=10, column=y, sticky=tk.W)  
 dobb = tk.Label(labelframe, text=**"Date of Boarding"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 dobb.grid(row=11, column=y, sticky=tk.W)  
 doaa = tk.Label(labelframe, text=**"Date of Arrival"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 doaa.grid(row=12, column=y, sticky=tk.W)  
  
 r = 2  
 **for** n, a, g, e, c, l, tkn, sn, dep, des, fa, dob, doa **in** mydata:  
 **global** ticketno1  
 ticketno1 = tkn  
 **if** a >= 18:  
 tfa = fa + (l \* 200)  
 **else**:  
 tfa = (fa \* 0.95) + (l \* 150)  
 name = tk.Label(labelframe, text=n, font=**'Arial 12 bold'**, fg=**'black'**)  
 name.grid(row=0, column=r, sticky=tk.W)  
 age = tk.Label(labelframe, text=a, font=**'Arial 12 bold'**, fg=**'black'**)  
 age.grid(row=1, column=r, sticky=tk.W)  
 gen = tk.Label(labelframe, text=g, font=**'Arial 12 bold'**, fg=**'black'**)  
 gen.grid(row=2, column=r, sticky=tk.W)  
 email = tk.Label(labelframe, text=e, font=**'Arial 12 bold'**, fg=**'black'**)  
 email.grid(row=3, column=r, sticky=tk.W)  
 con = tk.Label(labelframe, text=c, font=**'Arial 12 bold'**, fg=**'black'**)  
 con.grid(row=4, column=r, sticky=tk.W)  
 lugg = tk.Label(labelframe, text=l, font=**'Arial 12 bold'**, fg=**'black'**)  
 lugg.grid(row=5, column=r, sticky=tk.W)  
 tkn1 = tk.Label(labelframe, text=tkn, font=**'Arial 12 bold'**, fg=**'black'**)  
 tkn1.grid(row=6, column=r, sticky=tk.W)  
 sno = tk.Label(labelframe, text=sn, font=**'Arial 12 bold'**, fg=**'black'**)  
 sno.grid(row=7, column=r, sticky=tk.W)  
 depp = tk.Label(labelframe, text=dep, font=**'Arial 12 bold'**, fg=**'black'**)  
 depp.grid(row=8, column=r, sticky=tk.W)  
 dess = tk.Label(labelframe, text=des, font=**'Arial 12 bold'**, fg=**'black'**)  
 dess.grid(row=9, column=r, sticky=tk.W)  
 faa = tk.Label(labelframe, text=tfa, font=**'Arial 12 bold'**, fg=**'black'**)  
 faa.grid(row=10, column=r, sticky=tk.W)  
 dobb = tk.Label(labelframe, text=dob, font=**'Arial 12 bold'**, fg=**'black'**)  
 dobb.grid(row=11, column=r, sticky=tk.W)  
 doaa = tk.Label(labelframe, text=doa, font=**'Arial 12 bold'**, fg=**'black'**)  
 doaa.grid(row=12, column=r, sticky=tk.W)  
 labelframe.place(x=40, y=80)  
  
 button = tk.Button(root4, text=**"Download"**, width=20, height=2, bg=**'white'**, command=download\_ticket)  
 button.place(x=100, y=430)  
  
  
*# DOWNLOAD TICKET IN TEXT FILE FORMAT IN A USER SELECTED PATH***def** download\_ticket():  
 mycursor.execute(**"select \* from tickets,schedule \  
 where TicketNumber=%s and tickets.code=schedule.code"** % ticketno)  
 mydata = mycursor.fetchall()  
 **for** n, a, g, e, c, l, tkn, co, sn, dep, des, fa, dob, doa, code, ts, vs **in** mydata:  
 **if** a >= 18:  
 tfa = fa + (l \* 200)  
 **else**:  
 tfa = (fa \* 0.95) + (l \* 150)  
 fname = filedialog.asksaveasfilename(defaultextension=**".txt"**, filetypes=[(**"Text File"**, **".txt"**)])  
 f = open(fname, **"w"**)  
 f.write(**"NAME:"**)  
 f.write(n + **"\n"**)  
 f.write(**"AGE:"**)  
 f.write(str(a) + **"\n"**)  
 f.write(**"GENDER:"**)  
 f.write(g + **"\n"**)  
 f.write(**"EMAIL:"**)  
 f.write(e + **"\n"**)  
 f.write(**"CONTACT:"**)  
 f.write(c + **"\n"**)  
 f.write(**"LUGGAGE:"**)  
 f.write(str(l) + **"\n"**)  
 f.write(**"TICKET NUMBER:"**)  
 f.write(str(tkn) + **"\n"**)  
 f.write(**"CODE:"**)  
 f.write(str(co) + **"\n"**)  
 f.write(**"SEAT NUMBER:"**)  
 f.write(sn + **"\n"**)  
 f.write(**"FROM:"**)  
 f.write(dep + **"\n"**)  
 f.write(**"TO:"**)  
 f.write(des + **"\n"**)  
 f.write(**"TOTAL FARE:"**)  
 f.write(str(tfa) + **"\n"**)  
 f.write(**"DATE OF BOARDING:"**)  
 f.write(str(dob) + **"\n"**)  
 f.write(**"DATE OF ARRIVAL:"**)  
 f.write(str(doa) + **"\n"**)  
 f.close()  
 messagebox.showinfo(title=**"Download Ticket"**, message=**"File Downloaded"**)  
  
  
*# USER FUNCTION 2 : SEARCH TICKET***def** search\_ticket\_information():  
 root2.destroy()  
 **global** root3  
 root3 = tk.Tk()  
 root3.geometry(**"500x180"**)  
 root3.title(**"Bus Booking System:Search Ticket"**)  
 tkn2 = tk.Label(root3, text=**"Enter Ticket Number:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 tkn2.place(x=25, y=50)  
 **global** tkn\_ent2  
 tkn\_ent2 = tk.Entry(root3, width=30)  
 tkn\_ent2.place(x=285, y=60)  
 button = tk.Button(root3, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=display\_ticket\_information)  
 button.place(x=170, y=100)  
  
  
*# DISPLAYING THE SEARCHED TICKET***def** display\_ticket\_information():  
 **try**:  
 mycursor.execute(**"select Name,Age,Gender,Email,Contact,\  
 Luggage,TicketNumber,SeatNo,Boarding,Destination,Fare,DOB,DOA from tickets,schedule \  
 where TicketNumber=%s and tickets.code=schedule.code"** % int(tkn\_ent2.get()))  
 mydata = mycursor.fetchall()  
 **if** mydata == []:  
 messagebox.showerror(title=**"Info"**, message=**"Ticket Not Found"**)  
 **else**:  
 root3.destroy()  
 **global** root4  
 root4 = tk.Tk()  
 root4.geometry(**"375x520"**)  
 root4.title(**"Bus Booking System:Search Ticket"**)  
 heading = tk.Label(root4, text=**"Ticket Booked:"**, font=**'Arial 18 bold'**, fg=**'orange'**)  
 heading.place(x=100, y=30)  
  
 *# calls user() when window is closed* root4.protocol(**"WM\_DELETE\_WINDOW"**, on\_closing\_root4\_user)  
  
 labelframe = tk.Frame(root4)  
  
 y = 1  
 name = tk.Label(labelframe, text=**"Name"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**) *# dark cyan* name.grid(row=0, column=y, sticky=tk.W)  
 age = tk.Label(labelframe, text=**"Age"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 age.grid(row=1, column=y, sticky=tk.W)  
 gen = tk.Label(labelframe, text=**"Gender"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 gen.grid(row=2, column=y, sticky=tk.W)  
 email = tk.Label(labelframe, text=**"Email"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 email.grid(row=3, column=y, sticky=tk.W)  
 con = tk.Label(labelframe, text=**"Contact"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 con.grid(row=4, column=y, sticky=tk.W)  
 lugg = tk.Label(labelframe, text=**"Luggage"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 lugg.grid(row=5, column=y, sticky=tk.W)  
 tkn1 = tk.Label(labelframe, text=**"Ticket Number"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 tkn1.grid(row=6, column=y, sticky=tk.W)  
 sno = tk.Label(labelframe, text=**"Seat Number"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 sno.grid(row=7, column=y, sticky=tk.W)  
 depp = tk.Label(labelframe, text=**"From"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 depp.grid(row=8, column=y, sticky=tk.W)  
 dess = tk.Label(labelframe, text=**"To"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 dess.grid(row=9, column=y, sticky=tk.W)  
 faa = tk.Label(labelframe, text=**"Total Fare"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 faa.grid(row=10, column=y, sticky=tk.W)  
 dobb = tk.Label(labelframe, text=**"Date of Boarding"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 dobb.grid(row=11, column=y, sticky=tk.W)  
 doaa = tk.Label(labelframe, text=**"Date of Arrival"**, font=**'Arial 12 bold'**, fg=**'#20D2B5'**)  
 doaa.grid(row=12, column=y, sticky=tk.W)  
  
 r = 2  
 **for** n, a, g, e, c, l, tkn, sn, dep, des, fa, dob, doa **in** mydata:  
 **global** ticketno  
 ticketno = tkn  
 **if** a >= 18:  
 tfa = fa + (l \* 200)  
 **else**:  
 tfa = (fa \* 0.95) + (l \* 150)  
 name = tk.Label(labelframe, text=n, font=**'Arial 12 bold'**, fg=**'black'**)  
 name.grid(row=0, column=r, sticky=tk.W)  
 age = tk.Label(labelframe, text=a, font=**'Arial 12 bold'**, fg=**'black'**)  
 age.grid(row=1, column=r, sticky=tk.W)  
 gen = tk.Label(labelframe, text=g, font=**'Arial 12 bold'**, fg=**'black'**)  
 gen.grid(row=2, column=r, sticky=tk.W)  
 email = tk.Label(labelframe, text=e, font=**'Arial 12 bold'**, fg=**'black'**)  
 email.grid(row=3, column=r, sticky=tk.W)  
 con = tk.Label(labelframe, text=c, font=**'Arial 12 bold'**, fg=**'black'**)  
 con.grid(row=4, column=r, sticky=tk.W)  
 lugg = tk.Label(labelframe, text=l, font=**'Arial 12 bold'**, fg=**'black'**)  
 lugg.grid(row=5, column=r, sticky=tk.W)  
 tkn1 = tk.Label(labelframe, text=tkn, font=**'Arial 12 bold'**, fg=**'black'**)  
 tkn1.grid(row=6, column=r, sticky=tk.W)  
 sno = tk.Label(labelframe, text=sn, font=**'Arial 12 bold'**, fg=**'black'**)  
 sno.grid(row=7, column=r, sticky=tk.W)  
 depp = tk.Label(labelframe, text=dep, font=**'Arial 12 bold'**, fg=**'black'**)  
 depp.grid(row=8, column=r, sticky=tk.W)  
 dess = tk.Label(labelframe, text=des, font=**'Arial 12 bold'**, fg=**'black'**)  
 dess.grid(row=9, column=r, sticky=tk.W)  
 faa = tk.Label(labelframe, text=tfa, font=**'Arial 12 bold'**, fg=**'black'**)  
 faa.grid(row=10, column=r, sticky=tk.W)  
 dobb = tk.Label(labelframe, text=dob, font=**'Arial 12 bold'**, fg=**'black'**)  
 dobb.grid(row=11, column=r, sticky=tk.W)  
 doaa = tk.Label(labelframe, text=doa, font=**'Arial 12 bold'**, fg=**'black'**)  
 doaa.grid(row=12, column=r, sticky=tk.W)  
 labelframe.place(x=40, y=80)  
  
 button = tk.Button(root4, text=**"Download"**, width=20, height=2, bg=**'white'**, command=download\_ticket)  
 button.place(x=100, y=430)  
 **except**:  
 messagebox.showerror(title=**"Error"**,message=**"Invalid Input"**)  
  
  
*# USER FUNCTION 3 : UPDATION OF TICKET INFORMATION  
# INPUTS THE TICKET NUMBER***def** update\_ticket\_information():  
 root2.destroy()  
 **global** root3  
 root3 = tk.Tk()  
 root3.geometry(**"450x300"**)  
 root3.title(**"Bus Booking System"**)  
  
 label = tk.Label(root3, text=**"Updation:"**, font=(**"Arial Bold"**, 20), fg=**'dark orange'**)  
 label.pack(padx=30, pady=30)  
  
 tkn = tk.Label(root3, text=**"Enter Ticket Number:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 tkn.place(x=100, y=110)  
 **global** tkn\_ent2  
 tkn\_ent2 = tk.Entry(root3, width=30)  
 tkn\_ent2.place(x=130, y=155)  
  
 button = tk.Button(root3, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=update\_ticket\_information\_interface)  
 button.place(x=146, y=200)  
  
  
*# DISPLAYS OPTIONS TO UPDATE***def** update\_ticket\_information\_interface():  
 **global** tkn  
 tkn = int(tkn\_ent2.get())  
 mycursor.execute(**"select \* from tickets\  
 where TicketNumber=%s"** % tkn)  
 mydata = mycursor.fetchall()  
 **if** mydata == []:  
 messagebox.showerror(title=**"Error"**, message=**"Ticket Not Found"**)  
 **else**:  
 root3.destroy()  
 **global** root4  
 root4 = tk.Tk()  
 root4.geometry(**"470x450"**)  
 root4.title(**"Bus Booking System:Update Ticket"**)  
  
 he = tk.Label(root4, text=**"Select Your Choice:"**, font=**'Arial 18 bold'**, fg=**'dark orange'**)  
 he.pack(padx=30, pady=30)  
  
 button1 = tk.Button(root4, text=**"Name"**, width=20, height=2, bg=**'white'**, command=new\_name)  
 button1.place(x=150, y=100)  
  
 button2 = tk.Button(root4, text=**"Age"**, width=20, height=2, bg=**'white'**, command=new\_age)  
 button2.place(x=150, y=150)  
  
 button3 = tk.Button(root4, text=**"Gender"**, width=20, height=2, bg=**'white'**, command=new\_gen)  
 button3.place(x=150, y=200)  
  
 button4 = tk.Button(root4, text=**"Email"**, width=20, height=2, bg=**'white'**, command=new\_email)  
 button4.place(x=150, y=250)  
  
 button5 = tk.Button(root4, text=**"Contact"**, width=20, height=2, bg=**'white'**, command=new\_con)  
 button5.place(x=150, y=300)  
  
 button6 = tk.Button(root4, text=**"Luggage"**, width=20, height=2, bg=**'white'**, command=new\_lug)  
 button6.place(x=150, y=350)  
  
  
*# OPTION 1 : NAME***def** new\_name():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"500x250"**)  
 root5.title(**"Bus Booking System:Update Ticket"**)  
 nnam = tk.Label(root5, text=**"Enter New Name:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 nnam.place(x=40, y=70)  
 **global** nname\_ent  
 nname\_ent = tk.Entry(root5, width=30)  
 nname\_ent.place(x=250, y=80)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_name)  
 button.place(x=170, y=150)  
  
  
*# NAME UPDATION***def** up\_name():  
 nname = nname\_ent.get()  
 mycursor.execute(**"update tickets\  
 set Name='%s'\  
 where TicketNumber=%d "** % (nname, tkn))  
 f = open(**"ticket1.dat"**, **"rb+"**)  
 **try**:  
 **while True**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[6] == tkn:  
 mydata[0] = nname  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 messagebox.showinfo(title=**"Update Ticket"**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 user\_home\_page()  
 mycon.commit()  
  
  
*# OPTION 2 : AGE***def** new\_age():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"500x250"**)  
 root5.title(**"Bus Booking System:Update Ticket"**)  
 nage1 = tk.Label(root5, text=**"Enter New Age:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 nage1.place(x=50, y=70)  
 **global** nage\_ent  
 nage\_ent = tk.Entry(root5, width=30)  
 nage\_ent.place(x=240, y=80)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_age)  
 button.place(x=170, y=150)  
  
  
*# AGE UPDATION***def** up\_age():  
 nage = int(nage\_ent.get())  
 mycursor.execute(**"update tickets\  
 set Age=%s\  
 where TicketNumber=%d "** % (nage, tkn))  
 f = open(**"ticket1.dat"**, **"rb+"**)  
 **try**:  
 **while True**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[6] == tkn:  
 mydata[1] = nage  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
  
 messagebox.showinfo(title=**"Update Ticket"**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 user\_home\_page()  
 mycon.commit()  
  
  
*# OPTION 3 : GENDER***def** new\_gen():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"500x250"**)  
 root5.title(**"Bus Booking System:Update Ticket"**)  
 ngen1 = tk.Label(root5, text=**"Enter New Gender:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 ngen1.place(x=25, y=70)  
 **global** ngen\_ent  
 ngen\_ent = tk.Entry(root5, width=30)  
 ngen\_ent.place(x=250, y=80)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_gen)  
 button.place(x=170, y=150)  
  
  
*# GENDER UPDATION***def** up\_gen():  
 ngen = ngen\_ent.get()  
 mycursor.execute(**"update tickets\  
 set Gender='%s'\  
 where TicketNumber=%d "** % (ngen, tkn))  
 f = open(**"ticket1.dat"**, **"rb+"**)  
 **try**:  
 **while True**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[6] == tkn:  
 mydata[2] = ngen  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 messagebox.showinfo(title=**"Update Ticket"**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 user\_home\_page()  
 mycon.commit()  
  
  
*# OPTION 4 : EMAIL***def** new\_email():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"500x250"**)  
 root5.title(**"Bus Booking System:Update Ticket"**)  
 nemail1 = tk.Label(root5, text=**"Enter New Email:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 nemail1.place(x=40, y=70)  
 **global** nemail\_ent  
 nemail\_ent = tk.Entry(root5, width=30)  
 nemail\_ent.place(x=250, y=80)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_email)  
 button.place(x=170, y=150)  
  
  
*# EMAIL UPDATION***def** up\_email():  
 nemail = nemail\_ent.get()  
 mycursor.execute(**"update tickets\  
 set Email='%s'\  
 where TicketNumber=%d "** % (nemail, tkn))  
 f = open(**"ticket1.dat"**, **"rb+"**)  
 **try**:  
 **while True**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[6] == tkn:  
 mydata[3] = nemail  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
  
 messagebox.showinfo(title=**"Update Ticket"**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 user\_home\_page()  
 mycon.commit()  
  
  
*# OPTION 5 : CONTACT NUMBER***def** new\_con():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"500x250"**)  
 root5.title(**"Bus Booking System:Update Ticket"**)  
 ncon1 = tk.Label(root5, text=**"Enter New Contact:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 ncon1.place(x=25, y=70)  
 **global** ncon\_ent  
 ncon\_ent = tk.Entry(root5, width=30)  
 ncon\_ent.place(x=260, y=80)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_con)  
 button.place(x=170, y=150)  
  
  
*# CONTACT NUMBER UPDATION***def** up\_con():  
 ncon = ncon\_ent.get()  
 mycursor.execute(**"update tickets\  
 set Contact='%s'\  
 where TicketNumber=%d "** % (ncon, tkn))  
 f = open(**"ticket1.dat"**, **"rb+"**)  
 **try**:  
 **while True**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[6] == tkn:  
 mydata[4] = ncon  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 messagebox.showinfo(title=**"Update Ticket"**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 user\_home\_page()  
 mycon.commit()  
  
  
*# OPTION 6 : LUGGAGE***def** new\_lug():  
 root4.destroy()  
 **global** root5  
 root5 = tk.Tk()  
 root5.geometry(**"500x250"**)  
 root5.title(**"Bus Booking System:Update Ticket"**)  
 nlug = tk.Label(root5, text=**"Enter New Luggage:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 nlug.place(x=25, y=70)  
 **global** nlugg\_ent  
 nlugg\_ent = tk.Entry(root5, width=30)  
 nlugg\_ent.place(x=270, y=80)  
 button = tk.Button(root5, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=up\_lug)  
 button.place(x=170, y=150)  
  
  
*# LUGGAGE UPDATION***def** up\_lug():  
 nlugg = int(nlugg\_ent.get())  
 mycursor.execute(**"update tickets\  
 set Luggage=%s\  
 where TicketNumber=%d "** % (nlugg, tkn))  
 f = open(**"ticket1.dat"**, **"rb+"**)  
 **try**:  
 **while True**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[6] == tkn:  
 mydata[5] = nlugg  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 messagebox.showinfo(title=**"Update Ticket"**, message=**"UPDATION SUCCESSFUL"**)  
 root5.destroy()  
 user\_home\_page()  
 mycon.commit()  
  
  
*# USER FUNCTION 4 : CANCEL TICKET***def** ticket\_cancellation\_interface():  
 root2.destroy()  
 **global** root3  
 root3 = tk.Tk()  
 root3.geometry(**"500x180"**)  
 root3.title(**"Bus Booking System:Cancel Ticket"**)  
  
 tkn = tk.Label(root3, text=**"Enter Ticket Number:"**, font=**'Arial 18 bold'**, fg=**'black'**)  
 tkn.place(x=20, y=50)  
 **global** tkn\_ent  
 tkn\_ent = tk.Entry(root3, width=30)  
 tkn\_ent.place(x=270, y=60)  
 button = tk.Button(root3, text=**"Enter"**, width=20, height=2, bg=**'white'**, command=cancel\_ticket)  
 button.place(x=170, y=110)  
  
  
*# DELETING THE RECORD FROM THE DATABASE/BINARY FILE***def** cancel\_ticket():  
 tkn = int(tkn\_ent.get())  
 mycursor.execute(**"select \* from tickets"**)  
 mydata = mycursor.fetchall()  
 c = 0  
 **for** n, a, g, e, c, lug1, tn, co, sn **in** mydata:  
 **if** tn == tkn:  
 c = 1  
 mycursor.execute(**"delete from tickets\  
 where TicketNumber=%s"** % tkn)  
  
 f = open(**'ticket1.dat'**, **'rb'**)  
 n = open(**'ticket2.dat'**, **'wb'**)  
 **try**:  
 **while True**:  
 s = pickle.load(f)  
 **if** s[6] != tkn:  
 pickle.dump(s, n)  
 **except** EOFError:  
 f.close()  
 n.close()  
 os.remove(**'ticket1.dat'**)  
 os.rename(**'ticket2.dat'**, **'ticket1.dat'**)  
  
 messagebox.showinfo(title=**"Cancel Ticket"**, message=**"Cancellation Successful"**)  
 mycursor.execute(**"select \* from schedule where code=%s"** % co)  
 mydata = mycursor.fetchall()  
 **for** b, d, f, dob, doa, co, ts, vs **in** mydata:  
 vs += 1  
 mycursor.execute(**"update schedule set vacantseats=%s where code=%d"** % (vs, co))  
  
 f = open(**"schedules.dat"**, **"rb+"**)  
 **while True**:  
 **try**:  
 pos = f.tell()  
 mydata = pickle.load(f)  
 **if** mydata[5] == co:  
 mydata[7] = int(mydata[7] + 1)  
 f.seek(pos)  
 pickle.dump(mydata, f)  
 **break  
 except** EOFError:  
 f.close()  
 root3.destroy()  
 user\_home\_page()  
 mycon.commit()  
 **break  
 else**:  
 messagebox.showerror(title=**"Error"**, message=**"Ticket Not Found"**)  
  
  
*# USER FUNCTION 5 : EXIT***def** user\_exit():  
 **if** messagebox.askyesno(title=**"Quit?"**, message=**"Back to Home Page?"**):  
 root2.destroy()  
 main()  
  
  
*# DELETES THE RECORDS FROM SCHEDULE AND TICKETS DATABASE WHO'S DATE OF BOARDING HAS PASSED THE CURRENT DATE***def** delete\_daily\_database\_records():  
 mycursor.execute(**"select \*,curdate() from schedule"**)  
 mydata = mycursor.fetchall()  
 **for** dep, des, fa, dob, doa, code, ts, vs, curdate **in** mydata:  
 mycursor.execute(**"select code from schedule\  
 where DOB<'%s' "** % curdate)  
 mydata2 = mycursor.fetchall()  
 mycursor.execute(**"delete from schedule\  
 where DOB<'%s' "** % curdate)  
  
 **for** code **in** mydata2:  
 mycursor.execute(**"delete from tickets\  
 where code=%d "** % code)  
 f = open(**'schedules.dat'**, **'rb'**)  
 n = open(**'schedules2.dat'**, **'wb'**)  
 **try**:  
 **while True**:  
 s = pickle.load(f)  
 **if** s[5] != code:  
 pickle.dump(s, n)  
 **except** EOFError:  
 f.close()  
 n.close()  
 os.remove(**'schedules.dat'**)  
 os.rename(**'schedules2.dat'**, **'schedules.dat'**)  
  
 f = open(**'ticket1.dat'**, **'rb'**)  
 n = open(**'ticket2.dat'**, **'wb'**)  
 **try**:  
 **while True**:  
 s = pickle.load(f)  
 **if** s[5] != code:  
 pickle.dump(s, n)  
 **except** EOFError:  
 f.close()  
 n.close()  
 os.remove(**'ticket1.dat'**)  
 os.rename(**'ticket2.dat'**, **'ticket1.dat'**)  
 mycon.commit()  
  
*# GOES BACK TO THE ADMIN HOME PAGE***def** on\_closing\_root4\_admin():  
 **if** messagebox.askyesno(title=**"Quit?"**, message=**"Back to Home Page?"**):  
 root4.destroy()  
 admin\_home\_page()  
  
*# GOES BACK TO THE USER HOME PAGE***def** on\_closing\_root4\_user():  
 **if** messagebox.askyesno(title=**"Quit?"**, message=**"Back to Home Page?"**):  
 root4.destroy()  
 user\_home\_page()  
  
  
*# GOES BACK TO THE MAIN HOME PAGE***def** on\_closing\_root2\_admin():  
 **if** messagebox.askyesno(title=**"Quit?"**, message=**"Back to Home Page?"**):  
 root2.destroy()  
main()  
*delete\_daily\_database\_records()*

**OUTPUT**

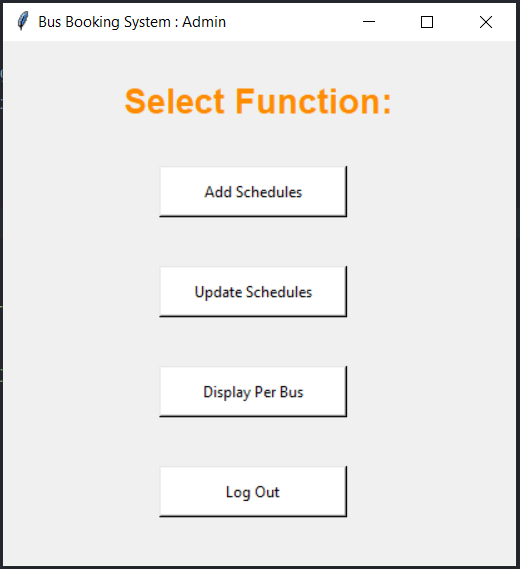
**Primary Window/Home Page:**



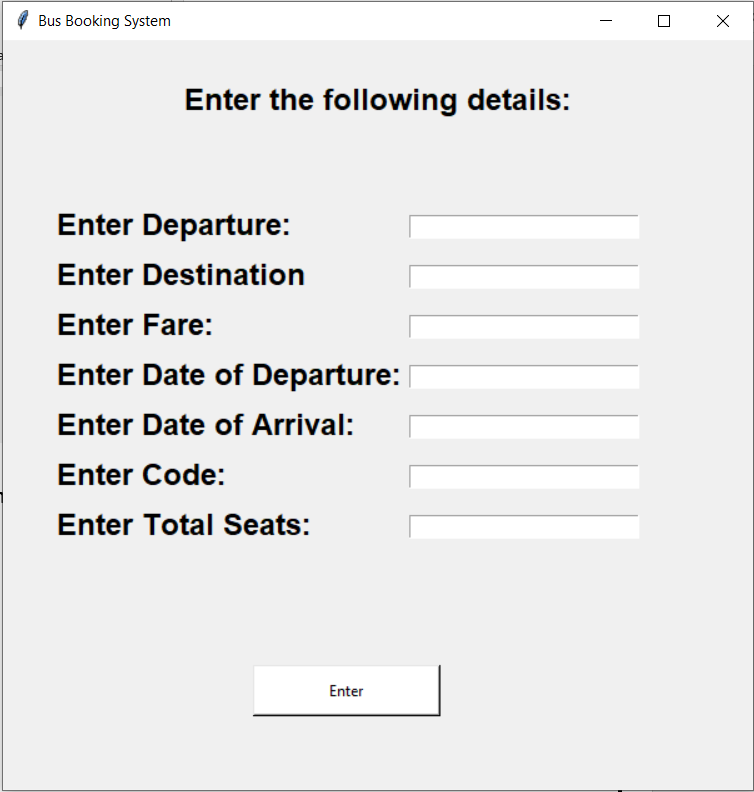
**Authentication:**



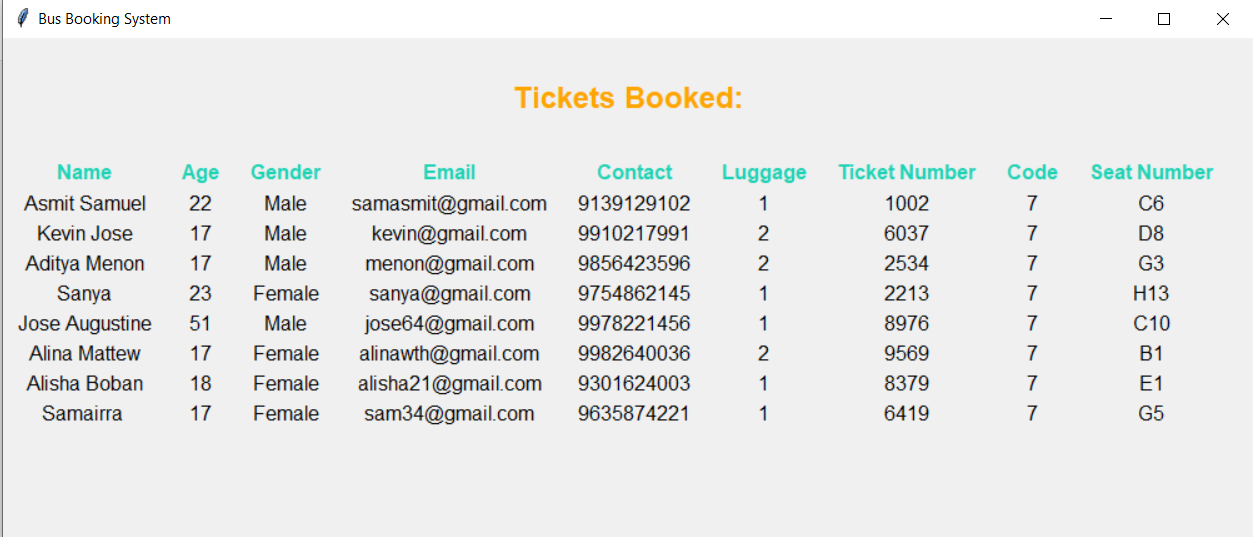
**Admin Home Page:**



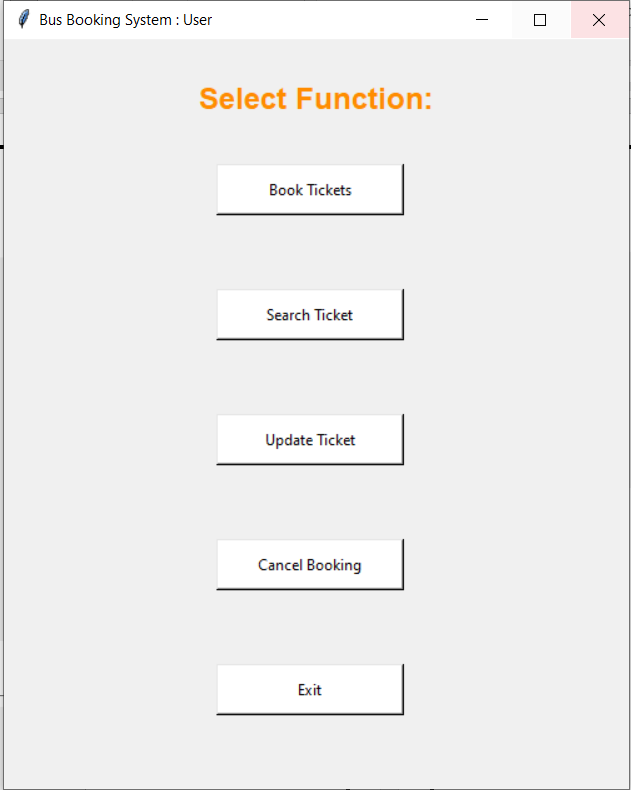
**Add Schedule:**



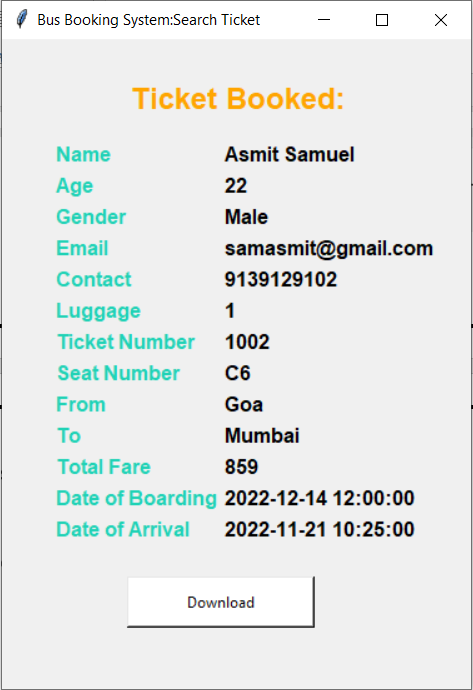
**Display Per Bus:**



**User Home Page:**



**Ticket Generator:**



**REFERENCES**

The following sources were used in the completion of this project:

* Class 11 and 12 Sumita Arora’s Computer Science with Python
* <https://www.javatpoint.com>
* <https://www.tutorialspoint.com>
* <https://www.geeksforgeeks.org>
* <https://www.youtube.com/watch?v=ibf5cx221hk>
* <https://stackoverflow.com/>