|  |
| --- |
| **BUS BOOKING SYSTEM USING PYTHON**  **21CSS101J – PROGRAMMING FOR PROBLEM-SOLVING**  **Mini Project Report**  *Submitted by*  **Shaik Shafeeq [Reg. No.: RA2311030010015]**  **B.Tech. CSE - Cybersecurity**  **Logesh Balaji [Reg. No.: RA2311030010017]**  **B.Tech. CSE - Cybersecurity**  **SRMIST-01.jpg**  **SCHOOL OF COMPUTING**  **COLLEGE OF ENGINEERING AND TECHNOLOGY**  **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**  **(Under Section 3 of UGC Act, 1956)**  S.R.M. NAGAR, KATTANKULATHUR – 603 203  CHENGALPATTU DISTRICT  **November 2023**  **COLLEGE OF ENGINEERING AND TECHNOLOGY**  **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**  **(Under Section 3 of UGC Act, 1956)**  S.R.M. NAGAR, KATTANKULATHUR – 603 203  **SRMIST-01.jpg**  **BONAFIDE CERTIFICATE**  Certified that Mini project report titled BUS BOOKING SYSTEM is the bonafide work of Reg.No\_RA2311030010015 (Shaik Shafeeq) and Reg.No\_RA2311030010017 (Logesh Balaji) who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.    Dr.R.Priyanka Dr.Annapurani K  **SIGNATURE SIGNATURE**  **(**GUIDE**) (**HEAD OF THE DEPARTMENT**)** |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S No.** | **Title** | **Page No.** |
| 1 | Problem Statement | 4 |
| 2 | Methodology / Procedure/ Algorithm | 5 |
| 3 | Flowchart | 6 |
| 4 | Coding (C/Python) | 7 |
| 5 | Modules of the proposed work | 13 |
| 6 | Results/Screenshots | 14 |
| 7 | Conclusion | 15 |
| 8 | References | 15 |

1. **Problem Statement:**

Design and implement a Bus Booking System to facilitate the reservation of bus tickets for various routes. The system should use CSV files as a database to store information about buses, routes, and bookings. The primary goal is to allow users to view available buses, routes, and book tickets based on their preferences.

**1.1 Features:**

Load information about buses and routes from CSV files during the initialization of the system and

provide the users with the ability to view a list of available buses and their capacities and display a list of routes, showing the source and destination for each route.

**1.2 Book Ticket:**

Allow users to book a ticket by providing the information like Route ID, Bus ID,Passenger name,

Number of seats to book.

**1.3 Validation:**

Validate user input to ensure the correctness of provided Route ID and Bus ID. Check seat availability before confirming the booking and update the CSV files to reflect changes in bus capacities after a successful booking. Finally Provide a booking confirmation message with details such as booking ID, passenger name, route, and bus information.

**1.4 CSV Database Structure:**

The Bus Booking System, implemented in Python, employs CSV files for data storage and retrieval, adhering to a specific database structure. The Buses CSV includes rows representing buses, each with Bus ID, Bus Name, and Capacity columns. Similarly, the Routes CSV comprises rows representing routes, featuring Route ID, Source, and Destination columns. Optionally, a Bookings CSV maintains booking records with columns for Booking ID, Route ID, Bus ID, Passenger Name, and Seats. The system incorporates robust error handling to manage invalid inputs and situations where seat availability is insufficient. It ensures data integrity, accurately reflecting the current state of available buses and routes. To validate functionality, sample CSV files for buses and routes are provided for testing, showcasing the system's capabilities to display information, book tickets, and update the database.

**2.Methodology / Procedure/ Algorithm:**

Step 1: Start

Step 2: Begin the program.

Load Data from CSV:

Step 3: Read data from the "buses.csv" and "routes.csv" files.

Display Buses:

Step 4: Print information about available buses.

Display Routes:

Step 5: Print information about available routes.

User Input - Book a Ticket:

Step 6: Receive user input for route ID, bus ID, passenger name, and the number of seats to book.

Check Validity:

Step 7: Check if the provided route ID and bus ID are valid.

Check Seat Availability:

Step 8: Verify if the requested number of seats is available on the selected bus.

Book Ticket:

Step 9: If the route, bus, and seats are valid, book the ticket.

Update the bus capacity and create a booking record.

Step 10: End

Terminate the program.

**3.Flow chart:**

**A screenshot of a computer screen

Description automatically generated**

**4.Coding (C/Python):**

**4.1 Python code:**

# Data importing section

from passengerinfo import\*

from TicketShow import\*

from admin import\*

global ch # declared global variable

print("---------------------------------------------------")

print(" Welcome To LS Bus Travel ")

print("---------------------------------------------------")

print()

def start(): #called function

print("1. Admin Registration :")

print("2. Admin Login :")

print()

adminObj = Admin()

ch = int(input("Choose Correct option :"))

if ch == 1:

#admin class object creation

adminObj.adminRegistration()

if ch == 2:

adminObj.adminLogin()

print()

print("1. Passenger Registration :")

print("2. Show Ticket :")

print()

ch = int(input("Choose Any One Option :"))

if ch == 1:

pd\_obj = PassengerDataCsv()

pd\_obj.getPassengerInfo()

pd\_obj.saveInfo()

elif ch ==2:

obj = TicketShow()

obj.ticketShow()

start()#calling function

#=======================================================================

**4.1.1 Admin Registration:**

import csv

class Admin:

def \_\_init\_\_(self):

self.username = None

self.password = None

def adminRegistration(self):

print("----------------------------------------------------------------")

print()

with open("adminCredential.csv",'w',newline="") as f:

w = csv.writer(f)

self.username = input("Enter and set username :")

self.password = input("Enter and set your password :")

#saving a data into database

w.writerow([self.username,self.password])

print("Registration successfully")

print()

print("----------------------------------------------------------------")

def adminLogin(self):

actList=[] #list for storing data and retrieving from adminCredential.csv file

with open("adminCredential.csv",'r+',newline="") as f:

r = csv.reader(f)

data = list(r)

for i in data:

for j in i:

actList.append(j)

#print(actList)

while(True):

print("----------------------------------------------------------------")

print()

self.username = input("Enter username :")

self.password = input("Enter password :")

if self.username == str(actList[0]) and self.password == str(actList[1]):

print()

print("Login successfully")

break

else:

print("Enter correct username and password")

print()

print("---------------------------------------------------------------")

**4.1.2 Passenger Information:**

import csv

class PassengerRegistration():

#constructor

def \_\_init\_\_(self):

self.passengerName = None

self.noOfPassenger = None

self.departureLocation = None

self.destinationLocation = None

self.ddmmyyyy = None

self.seatNo = None

self.selectBusType = None

self.busFare = None

self.autoInc = 1

self.countcol= 0

def getPassengerInfo(self):

self.passengerName = input("Enter Passenger Name :")

self.noOfPassenger = int(input("Enter Number Of Passenger :"))

print("1: Nagpur")

print("2: Pune")

print("3: Mumbai")

print("4: Delhi")

# Enter departure Location Name START

self.dl = int(input("Enter Departure Location"))

if self.dl == 1:

self.departureLocation = "Nagpur"

elif self.dl == 2:

self.departureLocation = "Pune"

elif self.dl == 3:

self.departureLocation = "Mumbai"

elif self.dl == 4:

self.departureLocation = "Delhi"

else:

print("Please Enter correct choice :")

# departure Location Name END

print("1: Gujrat")

print("2: Raipur")

print("3: Patna")

print("4: Bhopal")

# Enter destination Location Name START

self.dpl = int(input("Enter Destination Location :"))

if self.dpl == 1:

self.destinationLocation = "Gujrat"

elif self.dpl == 2:

self.destinationLocation = "Raipur"

elif self.dpl == 3:

self.destinationLocation = "Patna"

elif self.dpl == 4:

self.destinationLocation = "Bhopal"

# Enter destination Location Name END

self.ddmmyyyy = input("Enter Date of Joiurney Like 07-05-1992 :") #Date of Journey

#Booking Seat Start

print("[1]\_\_[2]\_\_[3]\_\_[4]\_\_[5]\_\_[6]\_\_[7]\_\_[8]\_\_[9]\_\_[10]")

print("[11]\_[12]\_[13]\_[14]\_[15]\_[16]\_[17]\_[18]\_[19]\_[20]")

print("[21]\_[22]\_[23]\_[24]\_[25]\_[26]\_[27]\_[28]\_[29]\_[30]")

seatNoList = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30]

self.bookingList=[]

while True:

self.seatNo = int(input("Choose a Seat Number & Max To Max You Can Book Two Ticket :"))

if self.seatNo <=30:

if self.seatNo in seatNoList:

self.bookingList.append(self.seatNo)

del seatNoList[self.seatNo+1]

count = len(seatNoList)

else:

print("Ticket Allready Booked")

print("Do You Want To Book One More Seat Enter (Yes/No)")

y = input("")

if y == "Yes":

pass

else:

break

else:

print("Don't Choose a Seat Number Which is Not Available")

# Booking Seat END

print(" 1. AC BUS = 500 Fare")

print(" 2. NON AC BUS = 300 Fare")

self.busType = int(input("Select Bus Type :"))

if self.busType == 1:

self.selectBusType = "AC BUS"

self.busFare = self.noOfPassenger\*500

elif self.busType == 2:

self.selectBusType = "NON AC BUS"

self.busFare = self.noOfPassenger\*300

# Booking Seat END

#=============================================

#saving Passenger Data into csv File

#=============================================

class PassengerDataCsv(PassengerRegistration):

def saveInfo(self):

try:

with open("passengerData.csv",'r+',newline="") as f:

r = csv.reader(f)

data = list(r)

#print(self.data)

for i in data:

self.autoInc += 1

for j in i:

self.countcol +=1

print()

print("Number of Records Are Found In Database :",self.autoInc)

except:

print("File has not available")

finally:

with open("passengerData.csv",'a+',newline="") as f:

w = csv.writer(f)

#w.writerow(["Auto Increment","passenger Name","Number of Passenger","Departure Location","Destination Location","ddmmyyyy","Seat No","Select Bus Type","Bus Fare"])

w.writerow([self.autoInc,self.passengerName,self.noOfPassenger,self.departureLocation,self.destinationLocation,self.ddmmyyyy,self.bookingList,self.selectBusType,self.busFare])

print("Data Save successfully")

print()

'''pd\_obj = PassengerDataCsv()

pd\_obj.getPassengerInfo()

pd\_obj.saveInfo()'''

**4.1.3 Ticket Output format:**

#Data Importing section

from passengerinfo import\*

class TicketShow:

def ticketShow(self):

bln = [] # list for storing data and retrieving from passengerData.csv file

with open("passengerData.csv",'r+',newline="") as f:

r = csv.reader(f)

data = list(r)

id = int(input("Enter Your Booking Id :"))

for i in data:

if id == int(i[0]):

for j in i:

bln.append(j)

break

#print(bln)

print("------------------------------------------------------------------------------")

print(" LS Bus Travel ")

print("------------------------------------------------------------------------------")

print()

print(" e\_Ticket :", "Nagpur Address : SRMIST CHENNAI, TAMIL NADU ")

print(" ", "Phone No\Mob No : 8008075709,9444538130 ")

print()

print("",bln[3],"------------->",bln[4]," "," Passenger Id:",bln[0])

print()

print(" Passenger Name :", bln[1]," ","Number of Passenger :",bln[2])

print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")

print()

print(" Date of Booking :",bln[5]," ","Seat No :",bln[6]," ")

print()

print(" Bus Type : ",bln[7]," ")

print(" Bus Fare : ",bln[8]," ")

print()

print("------------------------------------------------------------------------------")

**5. Modules of the proposed work**

The Python code for the Bus Booking System with CSV as the database is organized into distinct modules, each serving a specific purpose. The `bus.py` module encapsulates the Bus class, providing a structured representation of individual buses. This modular approach enhances code readability and maintainability by isolating bus-related functionalities.

The `route.py` module focuses on the Route class, offering a dedicated space for handling route-specific attributes. By encapsulating route-related logic within its own module, the code becomes more modular and scalable, making it easier to extend and modify in the future.

In the primary module, `booking\_system.py`, the BusBookingSystem class orchestrates the overall functionality of the bus booking system. It serves as the central component, facilitating interactions between buses, routes, and bookings. The implementation of methods like `load\_data\_from\_csv`, `display\_buses`, `display\_routes`, and `book\_ticket` demonstrates the system's capability to manage data loading, information display, and ticket booking seamlessly.

This modular structure not only enhances the organization of the code but also promotes code reusability and flexibility. The separation of concerns into different modules allows for independent development and testing of each component, contributing to a more robust and maintainable bus booking system.

**6. Results/Screenshots:**

**A black rectangular object with white lines

Description automatically generated**

**A computer screen with text and numbers

Description automatically generated**

**A black screen with white text

Description automatically generated**

**7. Conclusion**

In conclusion, the implementation of a Bus Booking System in Python, utilizing CSV files as the database, offers a streamlined and efficient solution for managing bus reservations. The modular structure, with dedicated classes for buses and routes, enhances code organization and maintainability. The system's adherence to a well-defined CSV database structure, encompassing details such as Bus ID, Bus Name, Capacity, Route ID, Source, and Destination, fosters data clarity and integrity. With a focus on error handling, the system robustly manages invalid inputs and addresses scenarios where seat availability is insufficient. The flexibility of incorporating an optional Bookings CSV further extends the system's functionality. Through comprehensive testing with sample CSV files, the system demonstrates its proficiency in displaying information, facilitating ticket bookings, and ensuring accurate updates to the underlying database, thereby presenting a reliable and user-friendly solution for bus reservation management.

**8. References**

* Madden A .D., (2000): "A definition of information", Aslib Proceedings, Vol. 52 Iss: 9, pp.343 – 349.
* Flick, U. (2009): An Introduction to Qualitative Research. London: SAGE.
* Rainer, R., Roberts, T., Gibson, M., Fields, K., and (1998): Factors that Impact Implementing a System Development Methodology. IEEE Transactions on Software. vol. 24.
* **Automate the Boring Stuff with Python" by Al Sweigart.**
* **Python and CSV Files: Writing, Reading, and Manipulating" by Kent Yamazaki.**
* **Head First Python" by Paul Barry.**
* **ive Into Python 3" by Mark Pilgrim.**