**Certificate**

****

This is to certify that **Kapil Singh Negi** student of Class XII has successfully completed project on topic, **Railway Management System** under the guidance of Mr. Deepak Pithadia, during the Academic year 2023-2024, for partial fulfilment of Computer Science (083), Practical Examination conducted by AISSCE 2023-2024.

Subject: Computer Science (083)

Board Roll number:

Subject Teacher Examiner

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Date: November 9, 2023

**Index**

|  |  |  |
| --- | --- | --- |
| S. No. | Particulars | Page No. |
| 1. | **Acknowledgement** | **2** |
| 2. | **Introduction** | **3** |
| 3. | **System Requirement Hardware/ Software** | **4** |
| 4. | **Purpose of Creating Project** | **5** |
| 5. | **Coding** | **6** |
| 6. | **Screen Shots** | **44** |
| 7. | **Conclusion** | **63** |
| 8. | **Bibliography / References** | **64** |

**Acknowledgement**

I would like to thank my computer teacher Mr. Deepak Pithadia for guiding me throughout this project work.

A special acknowledgement goes to our principal Ms. Sabina Sawhney who gave me the golden opportunity of this wonderful project, which also helped me in doing a lot of research and I came to know about so many new things.

I wish to thank my parents as well for their support and encouragement without which I could not have completed this project in the limited time frame.

**Introduction**

Railway ManagementSystem provides a way to book tickets conveniently and make the overall process easier for the customer as well as the administrator.

**System Requirements**

**Hardware Specification:**

* X86 64-bit CPU (intel/AMD architecture)
* 8 GB RAM
* 5 GB free disk space

**Software:**

Languages Used: Python Programming Language

Sequence Query Language

Python version (Recommended): 3.8

SQL version (Recommended): 5.5

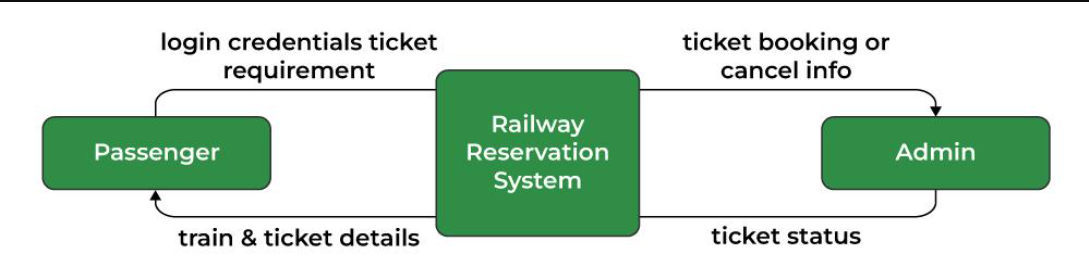
**Purpose of Creating Project**

The objective of the Railway Management System is to make the booking of trains more convenient and user-friendly.

It eases the process of booking tickets for everyone, from a single person travelling or a company using it for transporting goods in bulk.

It has a main administrator which has the ability to oversee all the operations that go about in the railways and make changes accordingly.

**Data-Flow Diagram (DFD):**



**Coding**

import mysql.connector

import datetime

def train\_fig():

return '''

▒▒▒▒▒▒ ░░▒▒░░

█████████████████████████████████████████████████████

██▓▓░░░░▒▒░░ ▒▒░░▒▒░░ ░░▒▒▒▒▒▒ ▒▒░░ ░░ ▒▒▒▒▒▒ ░░▒▒▒▒

██░░░░██ ██

▓▓▒▒░░██▓▓░░ ▒▒░░▓▓ ▒▒░░ ▓▓▓▓ ▒▒░░▓▓ ▒▒░░▓▓ ▒▒░░▓▓ ▓▓░

████████ ▒▒▓▓▓▓ ▓▓▒▒ ▒▒▒▒ ▓▓▒▒▒▒ ▒▒▒▒▓▓ ▓▓▒▒▒▒ ▓▓▒▒▓▓

██ ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒

██▒▒▒▒▒▒▒▒ ░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░

███████████████████████████████████████████████████████████

██▓▓▓ ██▓▓▓▓ ██▓▓▓▓ ██▓▓▓▓

███ ███ ▓▓█ ▓▓█

'''

def rules():

print('\n', '\t'\*3, 'RULES AND REGULATIONS TO FOLLOW WHILE TRAVELLING')

print('''\n1. Please maintain decorum and cleanliness while travelling.

2. In case of a medical emergency, please follow these series of steps:

\ta. Pull the chain immediately, dont wait for the next station

\tb. Dial “919” and inform them of the situation.

\tc. Calm the patient and wait until help arrives.

3. Buying tickets from vendors/agents is a punishable offense with penalty being 6 months

jail/paying a fine of Rs. 5000

4. Pulling the chain unnecessarily is a punishable offense and can lead to penalty of a

fine of Rs. 2000 or 1 month in prison

5. Carrying contraband, alcohols, weapons and other illegal items is illegal and can lead to

differing sentences for prison.

6. For inconveniences/complements while travelling, call at "97XXXXXX08" and submit your

complaint there\n''')

date = str(datetime.datetime.now())[:10]

time = str(datetime.datetime.now())[11:19]

sqldb = mysql.connector.connect(host='localhost', user='root', passwd='kapil2006@')

csr = sqldb.cursor()

try:

csr.execute('use cs\_project')

except mysql.connector.errors.ProgrammingError:

csr.execute('create database cs\_project')

csr.execute('use cs\_project')

while True:

print('''\nwelcome!!

what do you wish to do..?

1. check rules and regulations

2. login

3. sign up

4. about us''')

choice\_1 = 0

try:

choice\_1 = int(input('\nenter choice: '))

except ValueError:

print('invalid choice, you can only choose from 1 to 4')

temp = input('press enter to go back')

print('='\*92, '\n')

continue

if choice\_1 == 1:

print('-' \* 92)

rules()

temp = input('press enter to go back')

elif choice\_1 == 2:

username = input('enter username: ')

password = input('enter password: ')

if username == 'Admin' and password == 'kapil2006@':

while True:

print('-'\*92, '\n\nwhat do you wish to do..?')

print('\n1. alter train sheet\n2. alter user list\n3. server shut down\n4. logout')

try:

choice = int(input('\nenter choice: '))

except ValueError:

print('invalid choice, you can only choose from 1 to 4')

temp = input('press enter to go back')

continue

if choice == 1:

while True:

print('-'\*92, '\n\nwhat do you wish to do..?')

print('\n1. view train\_sheet\n2. append train\_sheet\n3. update status\n4. go back')

try:

choice = int(input('\nenter choice: '))

except ValueError:

print('invalid choice, you can only choose from 1 to 4')

temp\_1 = input('press enter to go back')

continue

if choice == 1:

csr.execute('select \* from train\_sheet')

temp = csr.fetchall()

k = ' '

print('-'\*92, '\n\n')

print('train\_num', ' start\_point', k\*12+'destination', k\*12+'departure', k\*3+'arrival',

k\*5+'train\_type')

print()

for i in temp:

print(i[0], k\*6+i[1], ' '\*(22 - len(i[1])), i[2], ' '\*(22 - len(i[2])),

str(i[3])+' ', str(i[5])+' ', i[8])

temp = input('\npress enter to go back')

elif choice == 2:

while True:

print('-' \* 92, '\n\nplease input values carefully')

print('if you enter an invalid value, you have to fill all values again')

train\_details = []

csr.execute('select city\_name from cities')

cities = csr.fetchall()

print('\n')

try:

temp = int(input('enter train\_number: '))

csr.execute('select train\_num from train\_sheet')

if (temp,) in csr.fetchall():

print('invalid input, train number already exists in database')

temp\_1 = input('press enter to go back')

continue

if len(str(temp)) == 5:

train\_details.append(temp)

else:

print('invalid input, train number can only be 5 digit')

temp\_1 = input('press enter to go back')

continue

except ValueError:

print('invalid train\_num')

temp\_1 = input('press enter to go back')

continue

temp = input('enter start\_point: ').title().strip()

if len(temp) > 20:

print('length of start\_point should not exceed 20 letters')

temp\_1 = input('press enter to go back')

continue

train\_details.append(temp)

temp = input('enter destination: ').title().strip()

if len(temp) > 20:

print('length of destination should not exceed 20 letters')

temp\_1 = input('press enter to go back')

continue

if (train\_details[1],) not in cities and (temp,) not in cities:

print('entered start\_point or destination not in database')

temp\_1 = input('press enter to continue')

continue

train\_details.append(temp)

temp = input('enter departure date: ')

try:

temp\_2 = datetime.date.fromisoformat(temp)

train\_details.append(temp)

except ValueError:

print('the date entered is in wrong format use "YYYY-MM-DD"')

temp\_1 = input('press enter to continue')

continue

temp = input('enter departure time: ')

try:

temp\_2 = datetime.time.fromisoformat(temp)

train\_details.append(temp)

except ValueError:

print('the time entered is in wrong format use "HH:MM:SS"')

temp\_1 = input('press enter to continue')

continue

try:

temp\_1 = int(input('enter avg speed of train(Km/H): '))

except ValueError:

print('invalid input, speed should only be a numerical value')

temp\_2 = input('press enter to go back')

continue

csr.execute(f'select \* from cities where city\_name = "{train\_details[1]}" or city\_name = "{train\_details[2]}"')

temp\_2 = csr.fetchall()

temp\_3 = round(((((temp\_2[0][2]-temp\_2[1][3])\*\*2 + (temp\_2[0][3]-temp\_2[1][3])\*\*2)\*\*0.5)\*4.9)/temp\_1, 2)//24

temp = datetime.datetime.strptime(train\_details[3], '%Y-%m-%d').date() + datetime.timedelta(days=temp\_3)

train\_details.append(str(temp))

coach\_sheet = (str(train\_details[0]) + '\_coach\_sheet\_' +

str(train\_details[3].replace('-', '\_')))

train\_details.append(coach\_sheet)

train\_details.append('on\_time')

temp = input('enter type of train(passenger/freight): ').lower()

if temp == 'passenger' or temp == 'freight':

pass

else:

print('type is invalid')

temp\_1 = input('press enter to go back')

continue

train\_details.append(temp)

if train\_details[8] == 'passenger':

csr.execute(f'create table {coach\_sheet} select \* from False\_Passenger\_Coach\_Sheet')

try:

ac\_coaches = int(input('\nenter number of ac coaches: '))

ac\_seats = int(input('enter number of ac seats per coach: '))

ac\_fee = int(input('enter booking fee of one ac seat: '))

if ac\_seats > 99:

print('invalid input, too many seats')

continue

sleeper\_coaches = int(input('\nenter number of sleeper coaches: '))

sleeper\_seats = int(input('enter number of sleeper seats per coach: '))

sleeper\_fee = int(input('enter booking fee of one sleeper seat: '))

if sleeper\_seats > 99:

print('invalid input, too many seats')

continue

except ValueError:

print('invalid input, you can only enter numerical values')

temp\_1 = input('press enter to go back')

continue

k = 65

temp\_2 = temp\_1 = 0

if ac\_coaches % 9 == 0:

temp\_3 = 0

else:

temp\_3 = 1

for i in range(k, k + ac\_coaches // 9 + temp\_3):

temp = chr(i)

for i in range(1, 10):

csr.execute(f'insert into {coach\_sheet} values("{temp + str(i)}","AC",{ac\_fee},{ac\_seats}, {ac\_seats})')

sqldb.commit()

temp\_1 += 1

if temp\_1 == ac\_coaches:

k = ord(temp) + 1

break

if sleeper\_coaches % 9 == 0:

temp\_3 = 0

else:

temp\_3 = 1

for i in range(k, k + sleeper\_coaches // 9 + temp\_3):

temp = chr(i)

for i in range(1, 10):

csr.execute(f'insert into {coach\_sheet} values("{temp + str(i)}","SLEEPER",{sleeper\_fee},{sleeper\_seats}, {sleeper\_seats})')

sqldb.commit()

temp\_2 += 1

if temp\_2 == sleeper\_coaches:

break

if train\_details[8] == 'freight':

csr.execute(f'create table {coach\_sheet} select \* from false\_freight\_coach\_sheet')

try:

wagon\_nums = int(input('enter number of wagons: '))

if wagon\_nums > 99:

print('invalid input, number of wagons can only be chosen from 1-99')

continue

except ValueError:

print('invalid input, number of wagons can only be chosen from 1-99')

continue

k = 65

temp\_1 = 0

if wagon\_nums % 9 == 0:

temp\_2 = 0

else:

temp\_2 = 1

for i in range(k, k + wagon\_nums // 9 + temp\_2):

temp = chr(i)

for i in range(1, 10):

csr.execute(f'insert into {coach\_sheet} values("{temp + str(i)}",NULL, "VACANT", NULL)')

sqldb.commit()

temp\_1 += 1

if temp\_1 == wagon\_nums:

break

while True:

print('-'\*92)

print("\nhere is the listing of entered data, please recheck")

print('\ntrain\_num: ', train\_details[0])

print('start\_point: ', train\_details[1])

print('destination: ', train\_details[2])

print('departure\_date: ', train\_details[3])

print('departure\_time: ', train\_details[4])

print('arrival\_date: ', train\_details[5])

print('coach\_sheet: ', train\_details[6])

print('status: ', train\_details[7])

print('train\_type: ', train\_details[8])

if train\_details[8] == 'passenger':

print('\nnumber of ac coaches: ', ac\_coaches, ', \t\tnumber of seats per coach: ', ac\_seats)

print('number of sleeper coaches:', sleeper\_coaches, ',\t\tnumber of seats per coach: ', sleeper\_seats)

print('booking fee of one ac\_seat: ', ac\_fee)

print('booking fee for one sleeper seat: ', sleeper\_fee)

elif train\_details[8] == 'freight':

print('number of empty wagons: ', wagon\_nums)

temp = input('\ndo you wish to continue with the following data(y/n): ')

if temp == 'y':

print('-'\*92, '\n\n---TRAIN DETAILS HAVE BEEN ADDED---')

temp\_2 = input('\npress enter to go back')

k = 0

break

elif temp == 'n':

k = 1

try:

csr.execute(f'drop table {coach\_sheet}')

sqldb.commit()

except mysql.connector.errors.ProgrammingError:

pass

temp\_3 = input('do you wish to (1)re-enter data or (2)exit: ')

if temp\_3 == '1':

pass

elif temp\_3 == '2':

k = 2

else:

print('invalid input you can only select from 1 or 2')

temp\_2 = input('press enter to go back')

continue

break

else:

print('invalid choice, you can only select from "y" or "n"')

temp\_2 = input('press enter to go back')

if k == 1:

continue

elif k == 2:

break

csr.execute('insert into train\_sheet values(%s,"%s","%s","%s","%s","%s","%s","%s","%s")' % tuple(train\_details))

sqldb.commit()

break

elif choice == 3:

csr.execute('select train\_num from train\_sheet')

train\_nums = csr.fetchall()

try:

print('-'\*92)

train\_num = int(input('\nenter train number: '))

if (train\_num,) not in train\_nums:

print('the entered train number does not exist')

temp = input('press enter to go back')

continue

except ValueError:

print('the entered train number does not exist')

temp = input('press enter to go back')

continue

csr.execute(f'select \* from train\_sheet where train\_num = "{train\_num}"')

temp = csr.fetchone()

while True:

print('\nhere are the detailed info on train number', train\_num)

print('\nstart point: ', temp[1])

print('destination: ', temp[2])

print('departure date: ', temp[3])

print('arrival date: ', temp[5])

print('current status: ', temp[7])

print('\nwhat do you wish to do..?')

print('1. set status = "on\_time"\n2. set status = "delayed"\n3. set status = "complete"\n4. go back')

try:

temp\_2 = int(input('\nenter choice: '))

status = ''

except ValueError:

print('invalid choice, you can only select from 1 to 4')

temp\_3 = input('press enter to go back')

print('-'\*92)

continue

if temp\_2 == 1:

csr.execute(f'update train\_sheet set status = "on\_time" where train\_num = {train\_num}')

sqldb.commit()

status = 'on\_time'

print('\n\t\t---STATUS HAS BEEN SET TO ON\_TIME---\n')

break

elif temp\_2 == 2:

csr.execute(f'update train\_sheet set status = "delayed" where train\_num = {train\_num}')

sqldb.commit()

status = 'delayed'

print('\n\t\t---STATUS HAS BEEN SET TO DELAYED---\n')

break

elif temp\_2 == 3:

csr.execute(f'update train\_sheet set status = "complete" where train\_num = {train\_num}')

sqldb.commit()

status = 'complete'

print('\n\t\t---STATUS HAS BEEN SET TO COMPLETE---\n')

break

elif temp\_2 == 4:

break

else:

print('invalid choice, you can only select from 1 to 4')

temp\_3 = input('press enter to go back')

print('-'\*92)

csr.execute('select user\_booking from user\_list where account = "business"')

temp = csr.fetchall()

for i in temp:

csr.execute(f'update {i[0]} set journey\_status = "{status}" where train\_number = {train\_num}')

sqldb.commit()

temp\_3 = input('press enter to go back')

elif choice == 4:

break

else:

print('invalid choice, you can only choose from 1 to 4')

temp\_1 = input('press enter to go back')

elif choice == 2:

print('-'\*92)

while True:

print('\nwhat do you wish to do..?')

print('\n1. view user list\n2. update sus value\n3. go back')

try:

choice\_1 = int(input('\nenter choice: '))

except ValueError:

print('invalid choice, you can only choose from 1 to 3')

choice\_1 = input('press enter to go back')

print('-' \* 92)

continue

if choice\_1 == 1:

csr.execute('select \* from user\_list')

temp = csr.fetchall()

k = ' '

print('-'\*92)

print('\nuser id', k\*6, 'username', k\*16, 'aadhar number', k\*3, 'age', k\*3, 'sus value', k\*4, 'account\n')

for i in temp:

print(i[0], k\*4, i[1], k\*(24-len(i[1])), i[3], k\*4, datetime.datetime.now().year - int(str(i[4])[:4]), k\*5, i[6], k\*12, i[7])

temp\_1 = input('\npress enter to go back')

elif choice\_1 == 2:

csr.execute('select user\_id from user\_list')

user\_ids = csr.fetchall()

try:

print('-' \* 92)

user\_id = int(input('\nenter user\_id: '))

if (user\_id,) not in user\_ids:

print('the entered user\_id does not exist')

temp = input('press enter to go back')

continue

except ValueError:

print('the entered user\_id does not exist')

temp = input('press enter to go back')

continue

csr.execute(f'select \* from user\_list where user\_id = "{user\_id}"')

temp = csr.fetchone()

while True:

print('\nhere are the detailed info on user\_id', user\_id)

print('\nusername: ', temp[1])

print('password: ', temp[2])

print('aadhar number: ', temp[3])

print('date of birth: ', temp[4])

print('sus value: ', temp[6])

print('account type: ', temp[7])

if temp[7] == 'business':

print('company name: ', temp[8])

temp\_1 = input('\ndo you wish to reset the sus value(y/n): ')

if temp\_1 == 'y':

csr.execute(f'update user\_list set sus\_val = 0 where user\_id = "{user\_id}"')

sqldb.commit()

print('\n\t\t---SUS VALUE HAS BEEN SET TO ZERO---\n')

break

elif temp\_1 == 'n':

break

else:

print('invalid input, you can only choose from "y" or "n"')

temp\_3 = input('press enter to go back')

print('-'\*92)

temp\_3 = input('press enter to go back')

elif choice\_1 == 3:

break

else:

print('invalid choice, you can only choose from 1 to 3')

temp = input('press enter to go back')

print('-'\*92)

elif choice == 3:

print('-' \* 92)

temp = input('\nenter "CONFIRM" if you really want to shut down the server: ')

if temp != "CONFIRM":

print('\nas you have written', temp, 'instead of "CONFIRM",')

print('the server shut down has been aborted')

temp = input('\npress enter to go back')

else:

print('\n\n', '\t'\*4, '---SERVER HAS BEEN SHUT DOWN---\n')

print('\*'\*92)

csr.close()

sqldb.disconnect()

exit()

elif choice == 4:

print('-'\*92, '\n\n\t\t\t\t--you have been logged out--')

temp = input('\npress enter to go back')

print('='\*92)

break

else:

print('invalid choice, you can only choose from 1 to 4')

temp = input('press enter to go back')

continue

csr.execute('select \* from user\_list')

user\_list = csr.fetchall()

csr.execute('select username from user\_list')

username\_list = csr.fetchall()

if (username,) in username\_list:

i = user\_list[username\_list.index((username,))]

user\_id = i[0]

if i[6] < 4:

if password == i[2] and i[7] == 'Personal':

csr.execute('update user\_list set sus\_val = 0 where user\_id = %s' % (user\_id,))

sqldb.commit()

while True:

print('-'\*92, '\n\nwelcome, ' + username, '\n\n1. check profile')

csr.execute(f'select user\_booking from user\_list where user\_id = "{user\_id}"')

if csr.fetchone()[0] == None:

print('2. book new seat')

temp = 0

else:

print('2. view ticket details')

temp = 1

print('3. log out')

try:

choice\_2 = int(input('\nenter choice: '))

print('-'\*92)

except ValueError:

print('invalid choice, you can only choose from 1 to 3')

temp = input('press enter to go back')

continue

if choice\_2 == 1:

csr.execute('select \* from user\_list where user\_id = "' + str(user\_id) + '"')

i = csr.fetchone()

print('\nuser id : ', i[0])

print('username : ', i[1])

print('aadhar number:', i[3])

print('age : ', datetime.datetime.now().year - int(str(i[4])[:4]))

temp = input('\npress enter to go back')

elif choice\_2 == 2 and temp == 0:

print("\nhere are the listings of the train stations all over India:\n")

csr.execute('select city\_name from cities')

temp = csr.fetchall()

temp\_1 = temp\_2 = None

for i in range(0, len(temp), 3):

try:

temp\_1 = 25 - len(temp[i][0])

print(temp[i][0], end=' '\*temp\_1)

temp\_2 = 25 - len(temp[i + 1][0])

print(temp[i + 1][0], end=' ' \* temp\_2)

print(temp[i+2][0], end='\n')

except IndexError:

continue

start\_point = input('\n\nenter name of start point (or exit): ').title().strip()

if start\_point == 'Exit':

temp = input('press enter to go back')

continue

destination = input('enter name of destination: ').title().strip()

csr.execute('select city\_name from cities')

temp = csr.fetchall()

if (start\_point,) in temp and (destination,) in temp:

csr.execute('select \* from train\_sheet where start\_point = "%s" and destination = "%s" and departure\_date > "%s" and train\_type ="passenger"' % (start\_point, destination, date))

temp = csr.fetchall()

train\_nums = []

while True:

if not temp == []:

print('-'\*92, '\n\nfollowing trains are available for travel:\n')

for i in temp:

csr.execute('select sum(seats\_available) from %s where coach\_type = "ac"' % (i[6]))

ac\_seats = csr.fetchone()[0]

if ac\_seats == None:

ac\_seats = 0

csr.execute('select sum(seats\_available) from %s where coach\_type = "sleeper"' % (i[6]))

sleeper\_seats = csr.fetchone()[0]

if sleeper\_seats == None:

sleeper\_seats = 0

if ac\_seats + sleeper\_seats != 0:

print('train number -', i[0])

print('\tdate of departure: ', i[3])

print('\ttime of departure: ', i[4])

print('\testimated date of arrival: ', i[5])

print('\tavailable ac seats: ', ac\_seats)

print('\tavailable sleeper seats: ', sleeper\_seats, '\n')

train\_nums.append(i[0])

try:

booking\_train = input('enter train number to proceed (or exit): ').lower().strip()

if booking\_train == 'exit':

temp = input('press enter to go back')

break

booking\_train = int(booking\_train)

except ValueError:

print('the entered train number is invalid')

temp\_1 = input('press enter to go back')

continue

if booking\_train in train\_nums:

csr.execute('select coach\_sheet from train\_sheet where train\_num = %s' %(booking\_train,))

coach\_sheet = csr.fetchone()[0]

while True:

csr.execute('select \* from %s where seats\_available != 0' % coach\_sheet)

print('-'\*92, '\n\nyou can select from the following coaches:\n')

temp\_1 = 1

coach\_nums = []

for i in csr.fetchall():

coach\_nums.append(i[0])

print(str(temp\_1)+'. coach number -', i[0])

print('\tcoach type: ', i[1])

print('\tbooking fee(one seat): ', i[2])

print('\tseats available ', i[3], '\n')

temp\_1 += 1

temp = input('\nenter the coach number to book a seat: ').upper()

if temp not in coach\_nums:

print('entered coach number is invalid')

temp = input('press enter to go back')

continue

csr.execute(f'select coach\_fee from {coach\_sheet} where coach\_num = "{temp}"')

print('-'\*92, '\n\nplease deposit Rs.', csr.fetchone()[0], 'into the counter')

temp\_1 = input('\n\n\npress enter to continue')

print('-'\*92, '\n')

train\_fig()

print('\t'\*7, '---TICKET BOOKING WAS SUCCESSFUL---\n\n')

temp\_1 = input('press enter to go back')

csr.execute(f'update {coach\_sheet} set seats\_available = seats\_available - 1 where coach\_num = "{temp}"')

csr.execute(f'update user\_list set user\_booking = "{coach\_sheet+"\_"+str(temp)}" where user\_id ="{user\_id}"')

sqldb.commit()

break

break

else:

print('the entered train number is invalid')

temp\_1 = input('press enter to go back')

else:

print('\nApologies.. it seems that no trains are up for the journey')

temp = input('press enter to go back')

break

else:

print('\neither start point or destination is chosen incorrectly')

temp = input('press enter to go back')

elif choice\_2 == 2 and temp == 1:

csr.execute(f'select \* from user\_list where user\_id = "{user\_id}"')

booking\_code = csr.fetchone()[5]

csr.execute(f'select \* from {booking\_code[:-3]} where coach\_num = "{booking\_code[-2:]}"')

coach\_details = csr.fetchone()

csr.execute(f'select \* from train\_sheet where train\_num = "{booking\_code[:5]}"')

train\_details = csr.fetchone()

while True:

print('\nTICKET DETAILS -\n')

print('train number: ', booking\_code[:5])

print('date of departure: ', train\_details[3])

print('time of departure: ', train\_details[4])

print('coach number: ', booking\_code[-2:])

print('train status: ', train\_details[7])

print('\nwhat do you wish to do..?')

print('1. download ticket\n2. cancel booking\n3. go back')

try:

choice\_3 = int(input('\nenter choice: '))

except ValueError:

print('invalid choice, you can only choose from 1 to 3')

continue

if choice\_3 == 1:

print()

path = input(r'enter path address to save ticket file: ')+'\\train\_ticket\_'+booking\_code+'.txt'

try:

f = open(path, 'w', encoding='UTF-8')

f.write(('\t'\*7)+'---TRAIN TICKET---'+train\_fig())

f.write('\ntrain num: '+str(train\_details[0]))

f.write('\nstart point: '+str(train\_details[1]))

f.write('\ndestination: '+str(train\_details[2]))

f.write('\ndeparture date : '+str(train\_details[3]))

f.write('\narrival date: '+str(train\_details[5]))

f.write('\ndeparture time: '+str(train\_details[4]))

f.write('\ntrain status: '+str(train\_details[7]))

f.write('\n\nbooking code: '+booking\_code)

f.write('\nprint date: '+str(datetime.datetime.now())[:10])

f.close()

except FileNotFoundError or PermissionError:

print('\nthere occurred a problem in saving the file,')

print('please recheck the path address and try again')

temp = input('press enter to go back')

print('-'\*92)

continue

print('-'\*92, '\n\n\nticket has been saved as', path)

temp = input('press enter to go back')

break

elif choice\_3 == 2:

print('-'\*92, '\n\nAre you sure you want to cancel the ticket,')

print('you would have to deposit 15% of the booking fee as cancellation fee')

temp = input('enter "CONFIRM" to cancel booking: ')

if temp != 'CONFIRM':

print('\nas you have written', temp, 'instead of "CONFIRM"')

print('the ticket cancellation will not take place')

temp = input('press enter to go back')

break

csr.execute(f'select coach\_fee from {booking\_code[:-3]} where coach\_num = "{booking\_code[-2:]}"')

print('-'\*92, '\n\nplease deposit Rs.', csr.fetchone()[0] \* 0.15, 'into the counter')

temp = input('\n\n\npress enter to continue')

print('-'\*92, '\n\n---YOUR TICKET HAS SUCCESSFULLY BEEN CANCELLED---')

csr.execute(f'update user\_list set user\_booking = NULL where user\_id = {user\_id}')

csr.execute(f'update {booking\_code[:-3]} set seats\_available = seats\_available + 1 where coach\_num = "{booking\_code[-2:]}"')

sqldb.commit()

temp = input('\n\n\npress enter to go back')

break

elif choice\_3 == 3:

break

else:

print('invalid choice, you can only choose from 1 to 3')

temp = input('press enter to go back')

elif choice\_2 == 3:

print('\n\t\t\t\t--you have been logged out--')

temp = input('\npress enter to go back')

break

else:

print('invalid choice, you can only choose from 1 to 3')

temp = input('press enter to go back')

elif password == i[2] and i[7] == 'Business':

csr.execute('update user\_list set sus\_val = 0 where user\_id = "%s"' % (user\_id,))

sqldb.commit()

while True:

print('-'\*92, '\n\nwelcome,', username)

print('\n1. check profile\n2. book coaches\n3. check bookings\n4. log out')

try:

choice\_4 = int(input('\nenter choice: '))

except ValueError:

print('invalid choice, you can only choose from 1 to 4')

temp = input('press enter to go back')

continue

csr.execute('select \* from user\_list where user\_id = "' + str(user\_id) + '"')

user\_details = csr.fetchone()

if choice\_4 == 1:

print('-'\*92, '\n\nuser id : ', user\_details[0])

print('username : ', user\_details[1])

print('aadhar number: ', user\_details[3])

print('age : ', datetime.datetime.now().year - int(str(user\_details[4])[:4]))

print('company affiliated: ', user\_details[8])

temp = input('\npress enter to go back')

elif choice\_4 == 2:

print('-'\*92, "\nhere are the listings of the train stations all over India:\n")

csr.execute('select city\_name from cities')

temp = csr.fetchall()

temp\_1 = temp\_2 = None

for i in range(0, len(temp), 3):

try:

temp\_1 = 25 - len(temp[i][0])

print(temp[i][0], end=' ' \* temp\_1)

temp\_2 = 25 - len(temp[i + 1][0])

print(temp[i + 1][0], end=' ' \* temp\_2)

print(temp[i + 2][0], end='\n')

except IndexError:

continue

start\_point = input('\n\nenter name of start point (or exit): ').title().strip()

if start\_point == 'Exit':

temp = input('press enter to go back')

continue

destination = input('enter name of destination: ').title().strip()

csr.execute('select city\_name from cities')

temp = csr.fetchall()

if (start\_point,) in temp and (destination,) in temp:

csr.execute('select \* from train\_sheet where start\_point = "%s" and destination = "%s" and departure\_date > "%s" and train\_type = "freight"' % (start\_point, destination, date))

temp = csr.fetchall()

train\_nums = []

while True:

False\_train\_nums = []

if not temp == []:

print('-' \* 92, '\n\nfollowing trains are available for travel:\n')

for i in temp:

csr.execute(f'select count(\*) from {i[6]} where status = "VACANT"')

coc\_avail = csr.fetchone()[0]

print('train number -', i[0])

print('\tdate of departure: ', i[3])

print('\ttime of departure: ', i[4])

print('\testimated date of arrival: ', i[5])

if coc\_avail != 0:

print('\tcoaches available: ', coc\_avail, '\n')

train\_nums.append(i[0])

else:

print('\tcoaches available: NONE\n')

False\_train\_nums.append(i[0])

try:

booking\_train = int(input('enter train number to proceed: '))

except ValueError:

print('the entered train number is invalid')

temp\_1 = input('press enter to go back')

continue

if booking\_train in train\_nums:

csr.execute('select coach\_sheet from train\_sheet where train\_num = %s' % (booking\_train,))

coach\_sheet = csr.fetchone()[0]

csr.execute(f'select count(\*) from {coach\_sheet} where Status = "VACANT"')

vacant\_coaches = csr.fetchone()[0]

while True:

csr.execute('select \* from coach\_types')

temp = csr.fetchall()

print('-'\*92, '\n\nhere are the types of coaches you can choose from:\n')

try:

for i in temp:

print(str(i[0])+'.', i[1], ' '\*(25-len(i[1])), 'fee per coach =',

i[2], ' '\*(11 - len(str(i[2]))))

type\_coach = int(input('\nselect the type of coach you wish to book: '))

if type\_coach > i[0]:

print('invalid input, type of coach coach can only be selected from 1-'+str(i[0]))

temp\_1 = input('press enter to go back')

continue

num\_coach = int(input('enter the number of coaches you wish to book: '))

if num\_coach > coc\_avail:

print('invalid input, no.of available coaches are only', coc\_avail)

temp\_1 = input('press enter to go back')

continue

csr.execute(f'select \* from coach\_types where sr\_no = {type\_coach}')

temp = csr.fetchone()

print('-' \* 92, '\n\nplease deposit Rs.', temp[2]\*num\_coach, 'into the counter')

temp\_1 = input('\n\npress enter to continue')

print('-' \* 92, '\n\n---YOUR BOOKING WAS SUCCESSFUL---')

csr.execute(f'select coach\_num from {coach\_sheet} where status = "VACANT"')

coach\_nums = csr.fetchall()

for i in range(num\_coach):

csr.execute(f'update {coach\_sheet} set coach\_type = "{temp[1]}", status = "BOOKED", booked\_for = "{user\_details[8]}" where coach\_num = "{coach\_nums[i][0]}"')

sqldb.commit()

csr.execute(f'select status from train\_sheet where train\_num = {booking\_train}')

temp\_1 = csr.fetchone()[0]

csr.execute(f'insert into {user\_details[5]} values({booking\_train},{num\_coach},"{temp[1]}","{temp\_1}")')

sqldb.commit()

while True:

temp = input('\n\ndo you wish to continue(y/n): ').lower()

if temp == 'y':

break

elif temp == 'n':

temp = 0

break

else:

print('invalid choice, you can only choose from "y" or "n"')

temp = input('press enter to go back')

print('-'\*92)

if not temp:

break

except ValueError or IndexError:

print('invalid input, type of coach coach can only be selected from 1-'+str(i[0]))

temp\_1 = input('press enter to go back')

continue

break

elif booking\_train in False\_train\_nums:

print('it seems that there are no coaches available in the train')

if train\_nums:

temp\_1 = input('press enter to go back')

continue

else:

temp\_1 = input('press enter to go to home page')

break

else:

print('the entered train number is invalid')

temp\_1 = input('press enter to go back')

else:

print('\nApologies.. it seems that no trains are up for the journey')

temp = input('press enter to go back')

break

else:

print('\neither start point or destination is chosen incorrectly')

temp = input('press enter to go back')

elif choice\_4 == 3:

csr.execute(f'select train\_number from {user\_details[5]} group by train\_number ')

train\_nums = csr.fetchall()

temp = 1

if train\_nums:

while True:

print('-' \* 92, '\n\nthe bookings were made in the following trains-')

for i in train\_nums:

csr.execute(f'select \* from train\_sheet where train\_num = {i[0]}')

temp\_1 = csr.fetchone()

print('\n', str(temp)+'.', i[0])

print('\tstart point: ', temp\_1[1])

print('\tdestination: ', temp\_1[2])

print('\tarrival date: ', temp\_1[5])

print('\tjourney status: ', temp\_1[7])

temp+= 1

try:

temp\_1 = int(input('\nenter train\_number to proceed: '))

except ValueError:

print('invalid input, entered train\_number is incorrect')

temp\_3 = input('press enter to go back')

continue

if (temp\_1,) not in train\_nums:

print('invalid input, entered train\_number is incorrect')

temp\_3 = input('press enter to go back')

continue

csr.execute(f'select coach\_sheet from train\_sheet where train\_num = {temp\_1}')

temp\_2 = csr.fetchone()[0]

csr.execute(f'select coach\_type, count(\*) from {temp\_2} where booked\_for = "{user\_details[8]}" group by coach\_type')

temp\_2 = csr.fetchall()

print('-'\*92, '\n')

for i in temp\_2:

print('coach\_type: ', i[0], ' '\*(25-len(i[0])), 'coaches booked: ', i[1])

temp\_3 = input('\npress enter to go back')

break

else:

print('-'\*92, '\n\n\nit seems that there are no bookings yet..')

temp = input('\npress enter to go back')

elif choice\_4 == 4:

print('-'\*92, '\n\n', '\t'\*4, '---you have been logged out---')

temp = input('\npress enter to go back')

break

else:

print('invalid choice, you can only choose from 1 to 4')

temp = input('press enter to go back')

else:

csr.execute('update user\_list set sus\_val = sus\_val + 1 where user\_id = '+str(i[0]))

sqldb.commit()

print('either username or password is incorrect')

temp = input('press enter to go back')

else:

print('\nthere have been numerous suspicious attempts to log in this account')

print('hence, you will not be able to log in to this account for a while')

print('\nplease contact the server manager to resolve this issue')

print('contact info:\n\temail- xyz@gmail.com\n\tph.no.- 915XX XXXX4')

else:

print('either username or password is incorrect')

temp = input('press enter to go back')

elif choice\_1 == 3:

while True:

print('-'\*92)

username = input('\nenter username: ')

csr.execute('select username from user\_list')

username\_list = csr.fetchall()

if not username.isalnum():

print('\nusername can not have any special characters or spaces')

temp = input('press enter to go back')

continue

if (username,) in username\_list:

print('\nusername is already taken, please choose another username')

temp = input('press enter to go back')

continue

password = input('enter password: ')

temp = input('enter aadhaar number: ')

if temp.isdigit() and len(temp) == 12:

aadhar\_num = temp

else:

print('\naadhar number can not have spaces and should contain only 12 digits')

temp = input('press enter to go back')

continue

account = input('enter account type(personal/business): ').title().strip()

csr.execute('select max(user\_id) from user\_list')

user\_id = csr.fetchone()[0] + 1

if account not in ('Personal', 'Business'):

print('\naccount can only be Personal or Business rest are invalid')

temp = input('press enter to go back')

continue

if account == 'Business':

company\_name = input('enter name of (company/brand): ')

csr.execute('select company\_name from user\_list where account = "business"')

temp = csr.fetchall()

if (company\_name,) in temp:

print('there is already an account made for', company\_name)

print('you can only make one account for a particular company/brand')

temp = input('press enter to go back')

continue

if len(company\_name) > 25:

print('name of company is too long please abbreviate it')

temp = input('press enter to go back')

break

csr.execute(f'create table {str(user\_id)+"\_bookings"} select \* from false\_business\_booking\_sheet')

DOB = input('enter DOB: ')

try:

if account == 'Business':

csr.execute(f'insert into user\_list values({user\_id},"{username}","{password}","{aadhar\_num}","{DOB}","{str(user\_id)+"\_bookings"}",{0},"{account}","{company\_name}")')

else:

csr.execute(f'insert into user\_list values({user\_id},"{username}","{password}","{aadhar\_num}","{DOB}",NULL,0,"{account}",NULL)')

print('\n---YOUR ACCOUNT HAS BEEN CREATED---')

except mysql.connector.errors.DataError:

print('\nDOB is to be written YYYY-MM-DD format')

continue

sqldb.commit()

temp = input('press enter to go back')

break

elif choice\_1 == 4:

print('-'\*92)

print('''\n\nwe hope to create software that makes the customer experience easier and more satisfying.

We prioritize customer satisfaction over anything else.''')

temp = input('\n\npress enter to go back')

else:

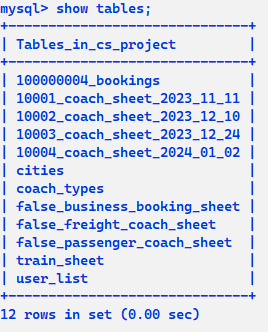
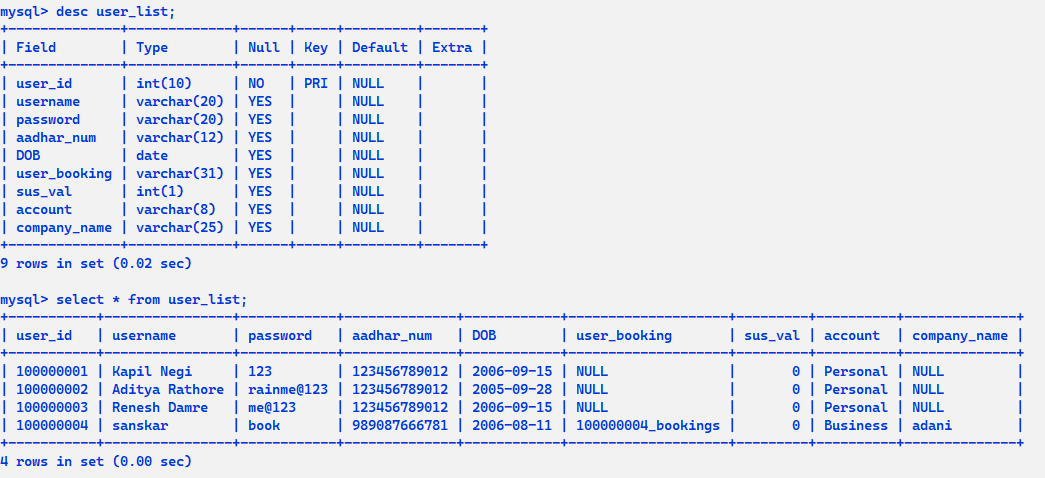
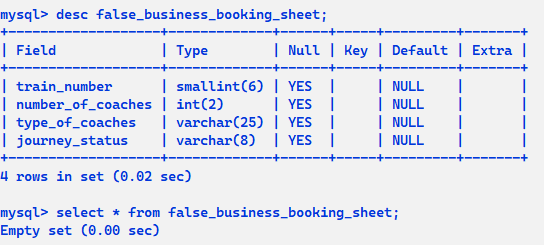
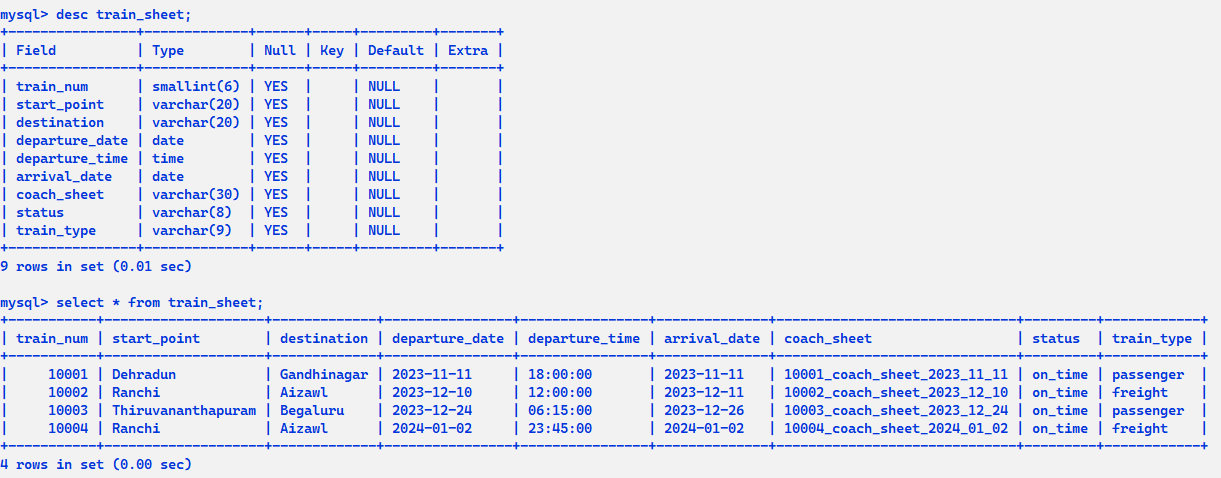
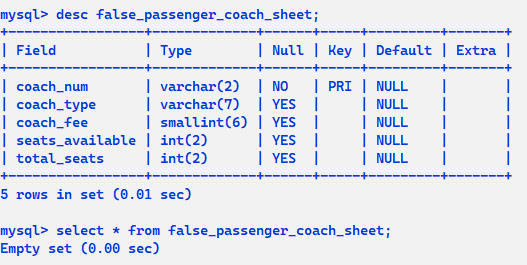
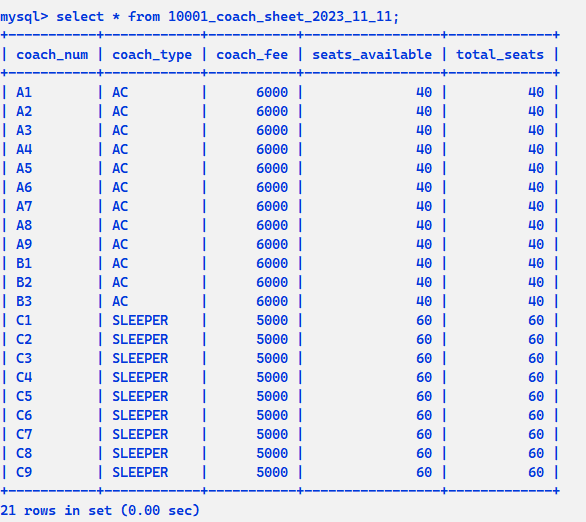
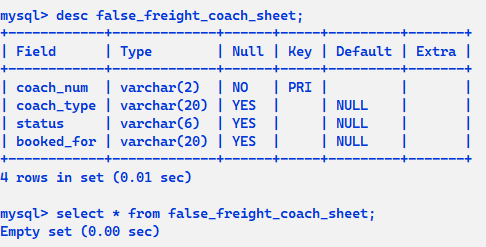
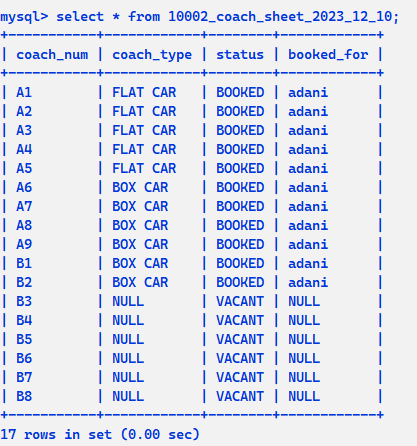
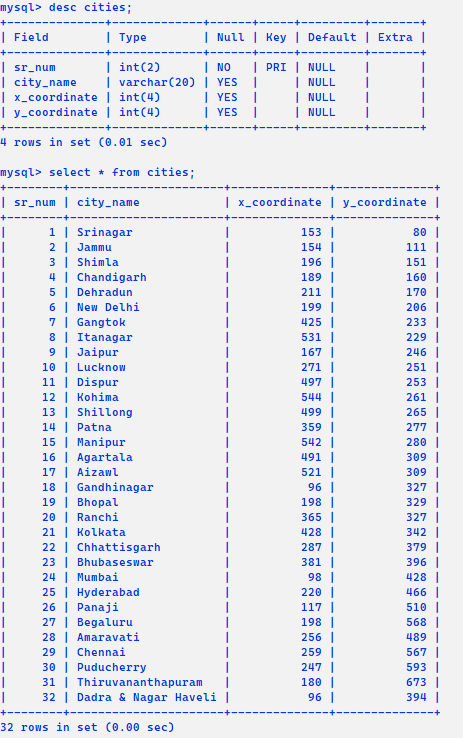
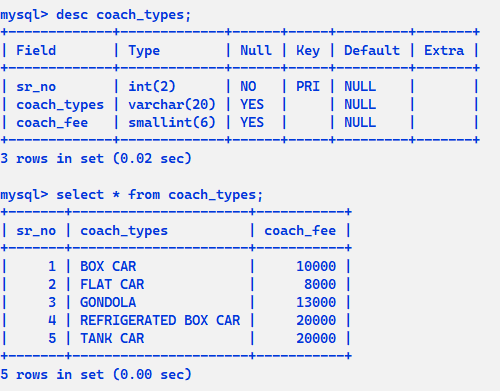
print('invalid choice, you can only choose from 1 to 4')

temp = input('press enter to go back')

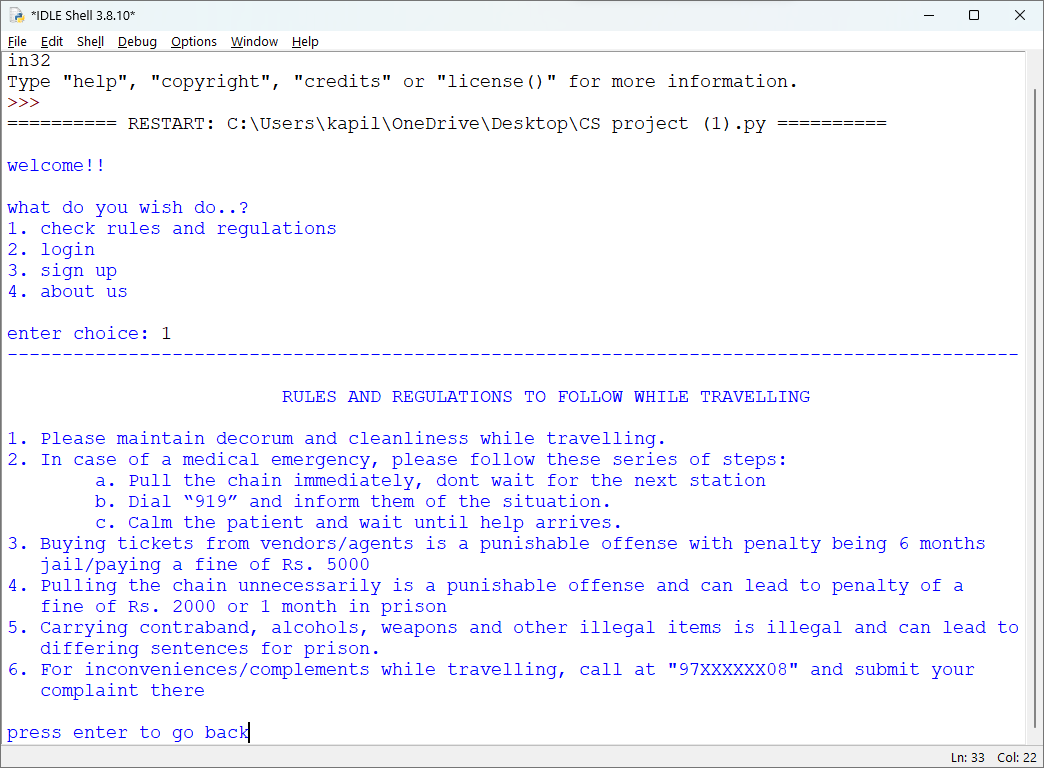
print('=' \* 92, '\n')

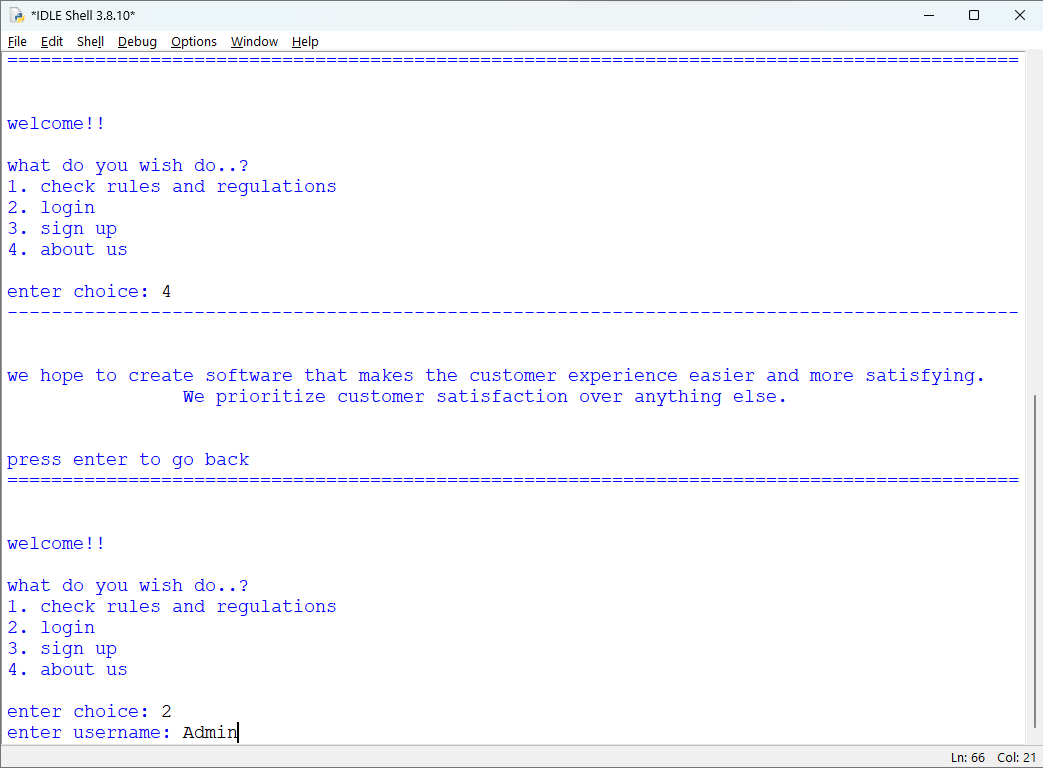
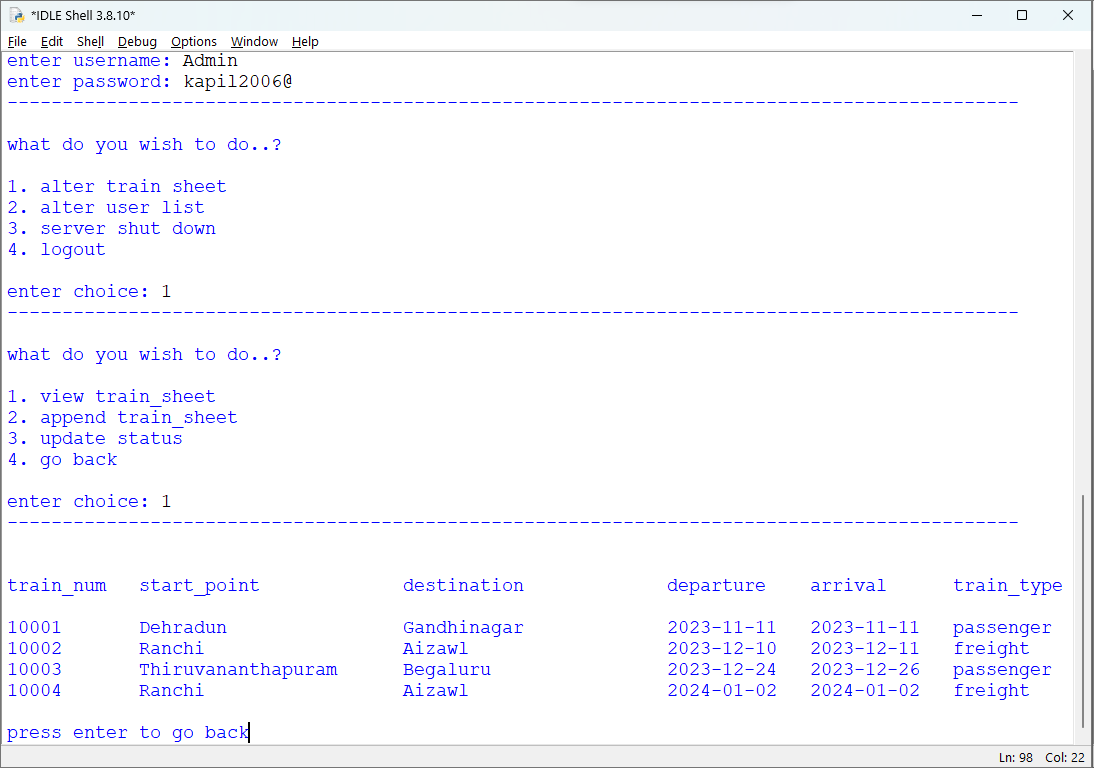
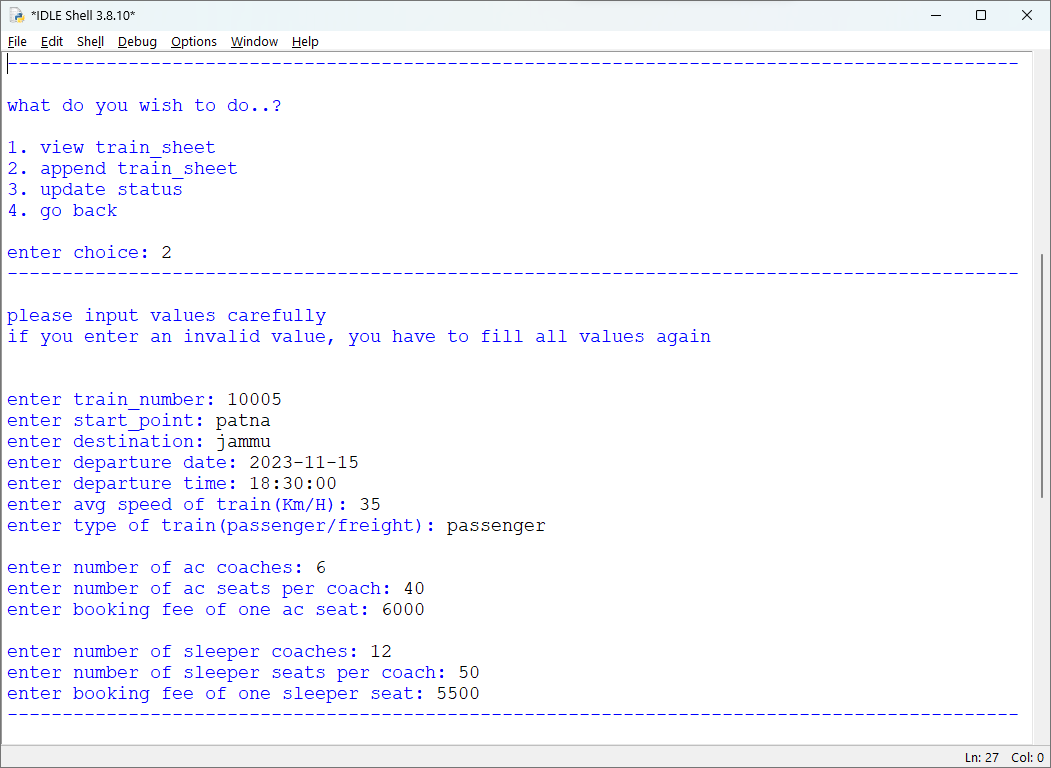
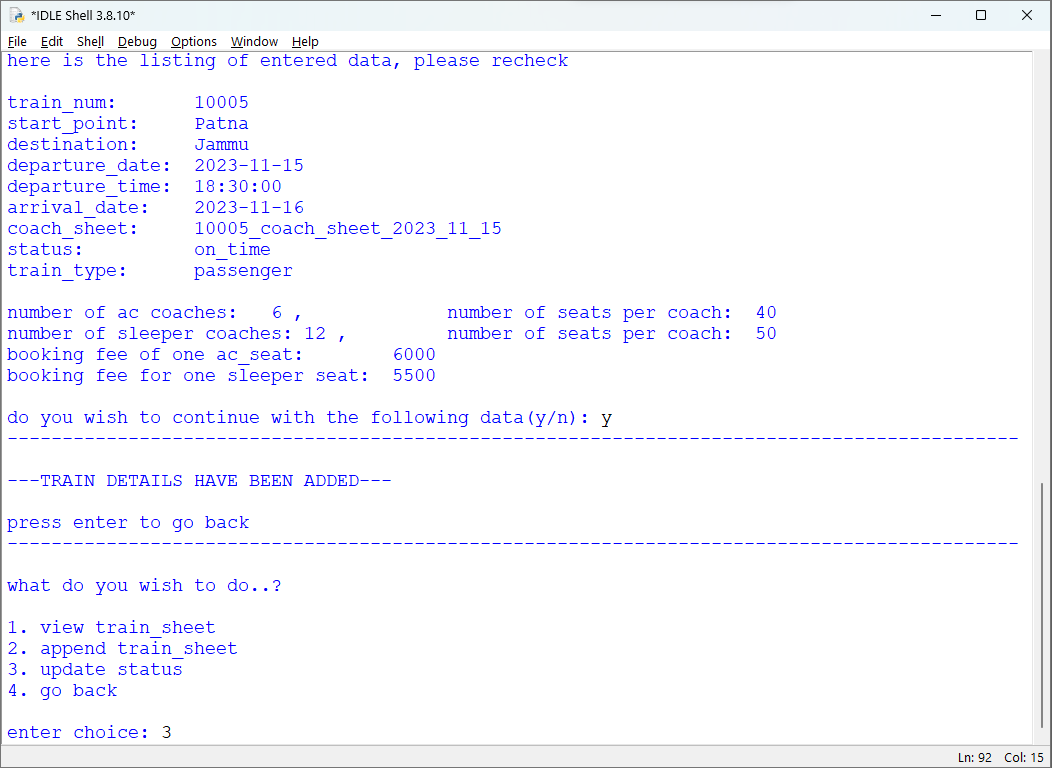
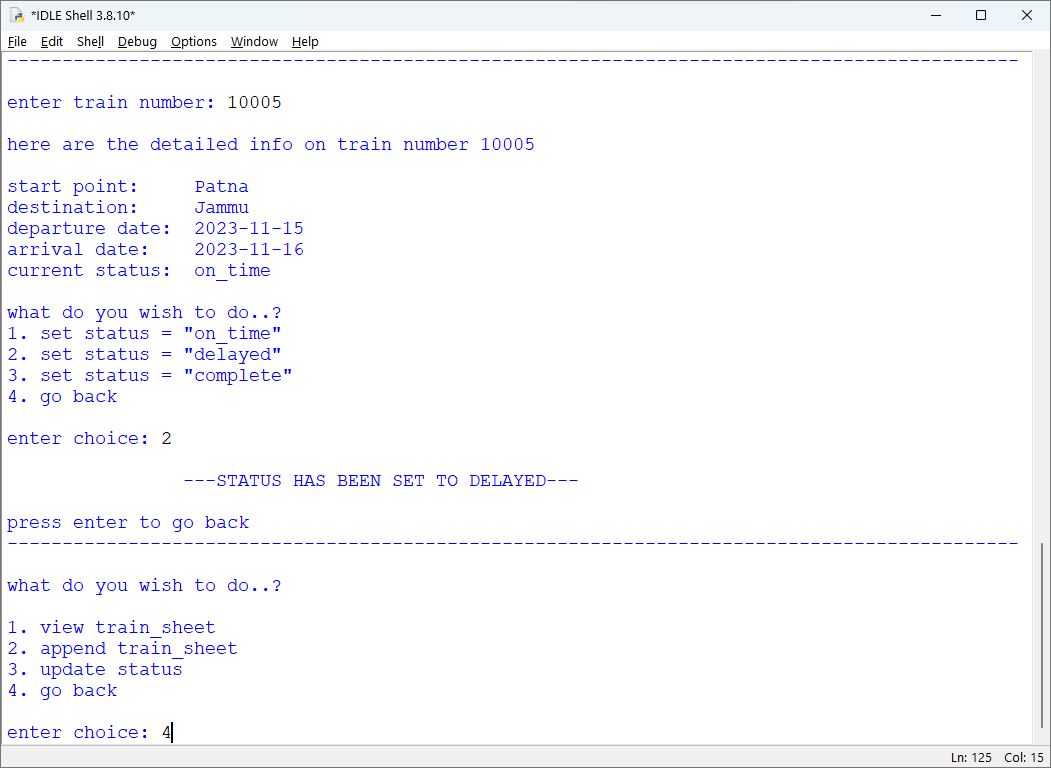
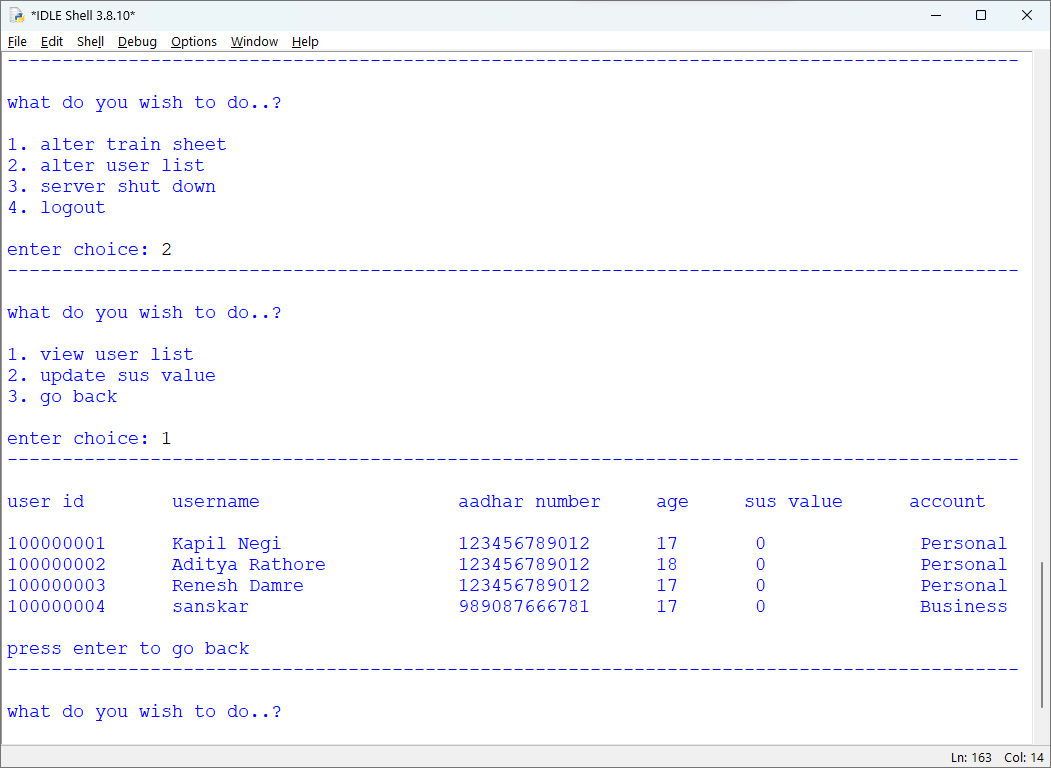
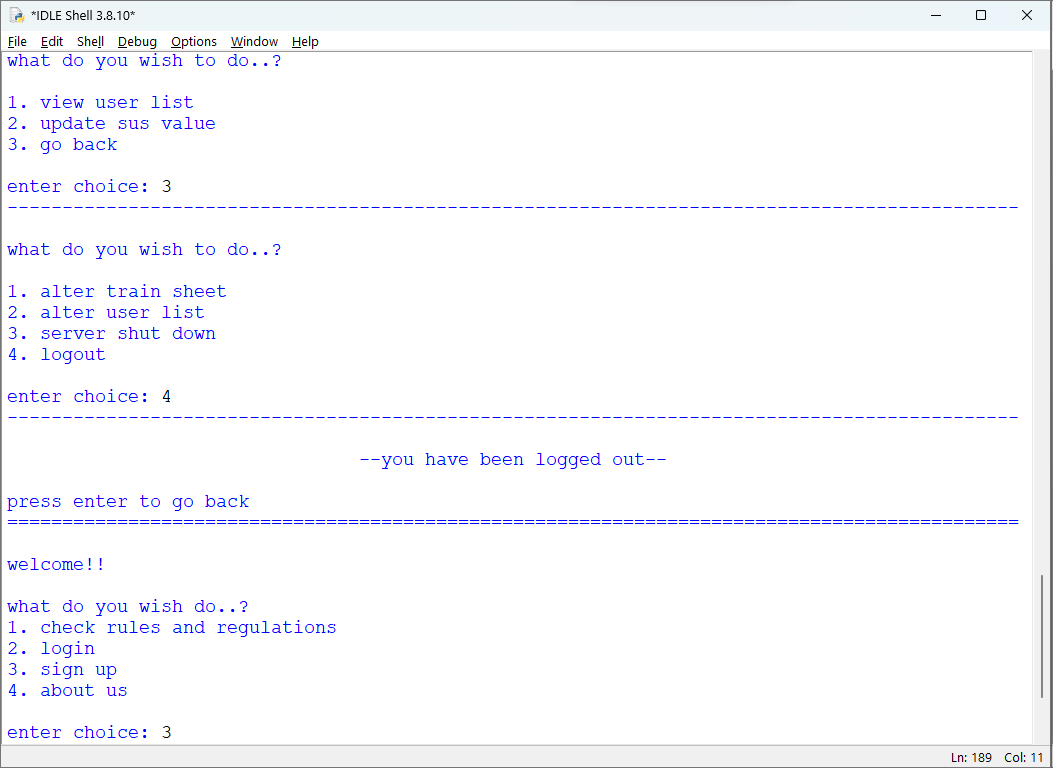
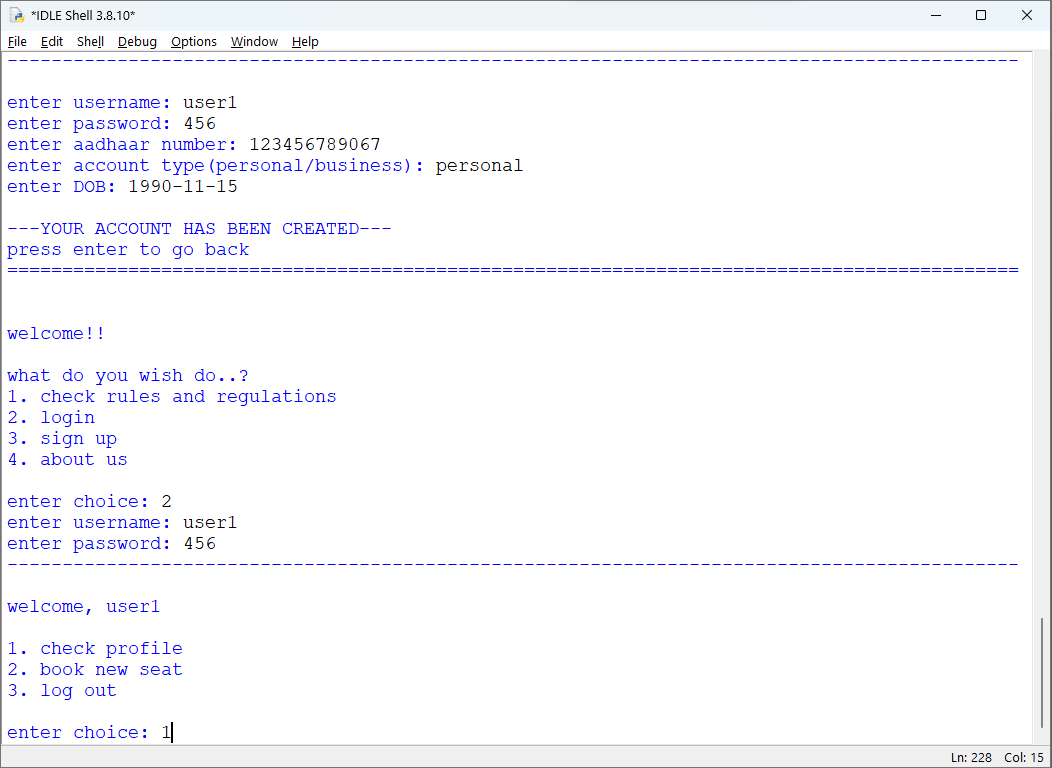
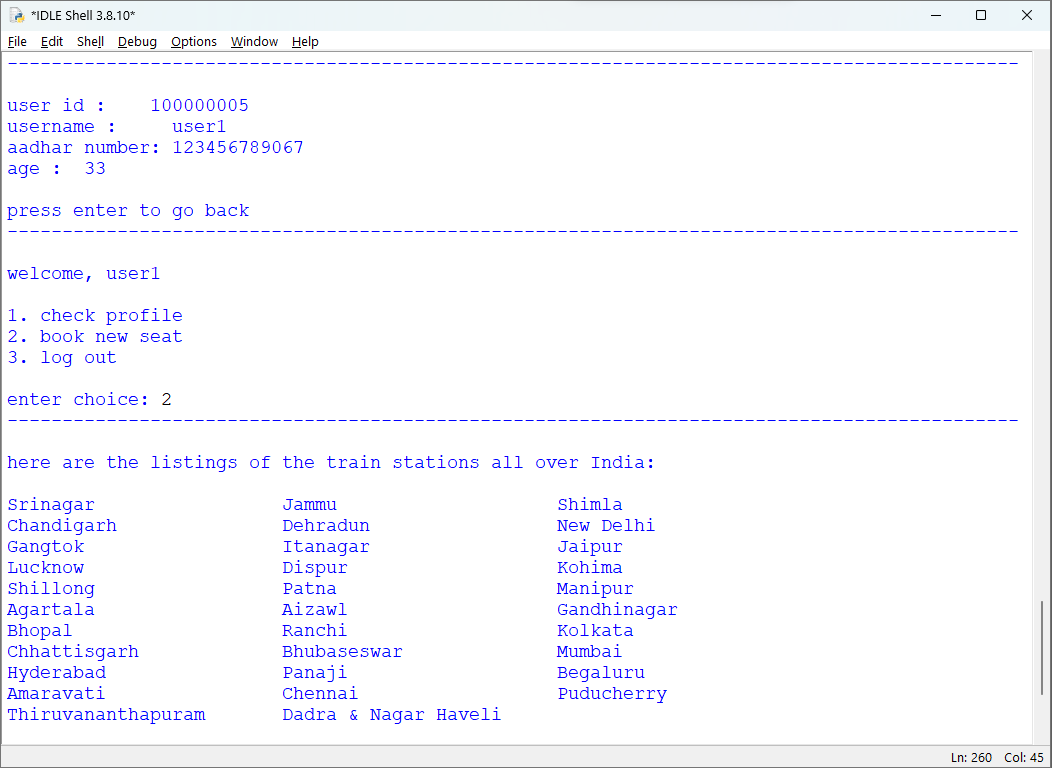
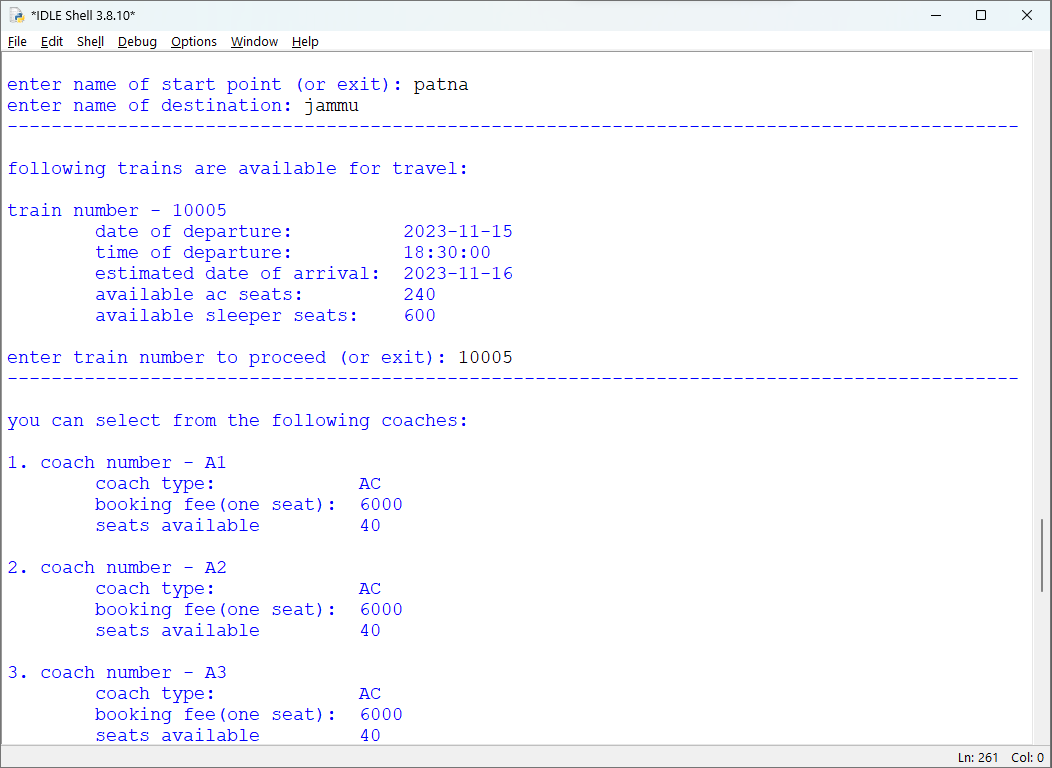
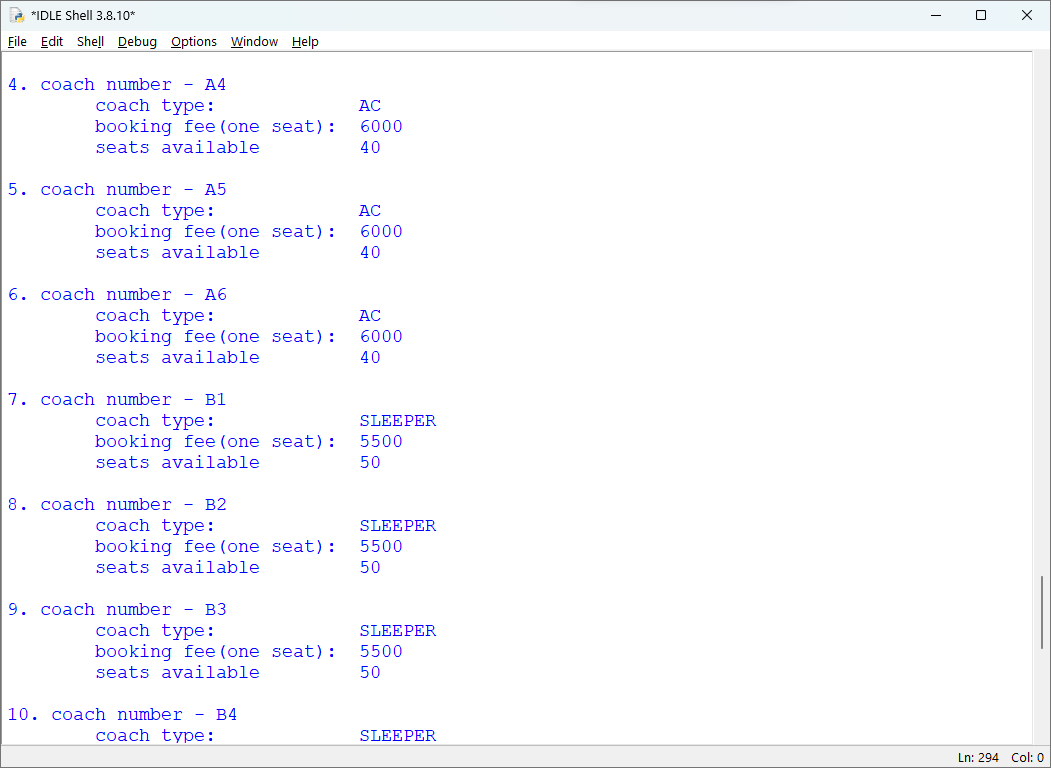
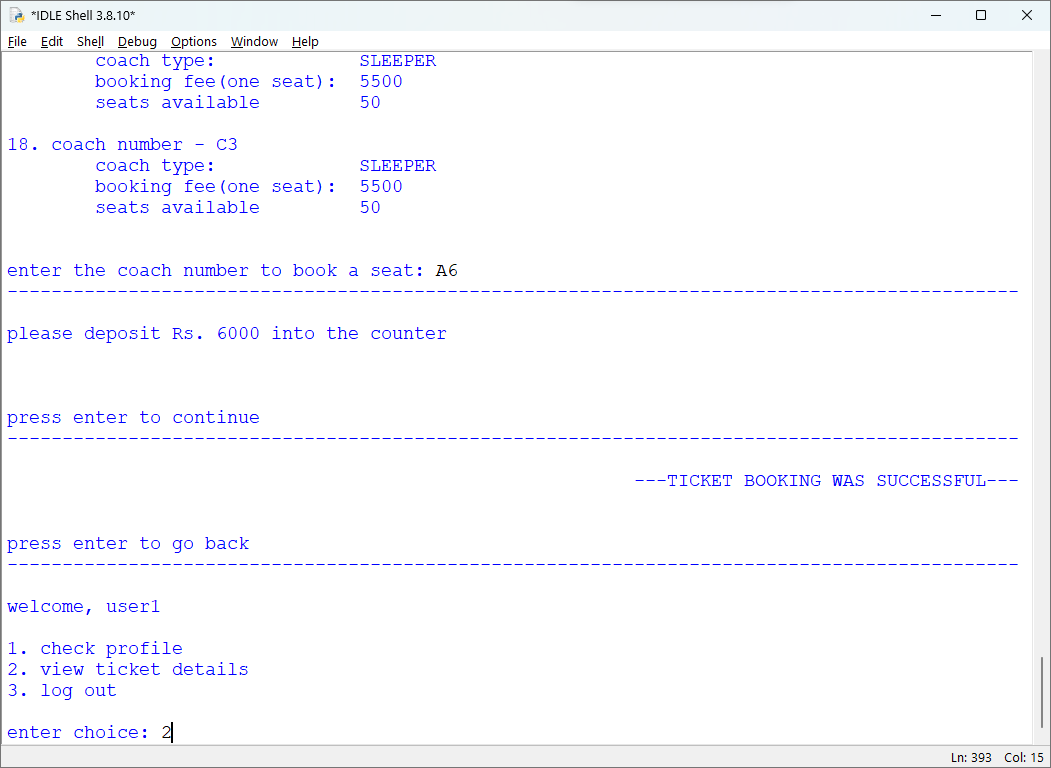
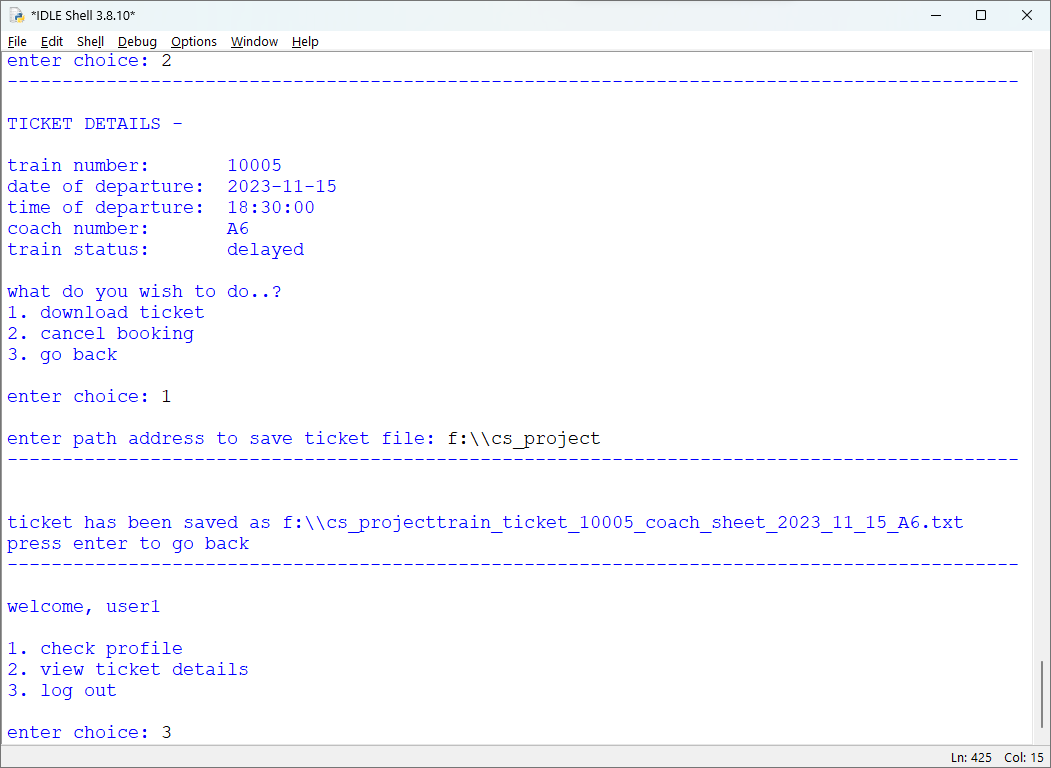
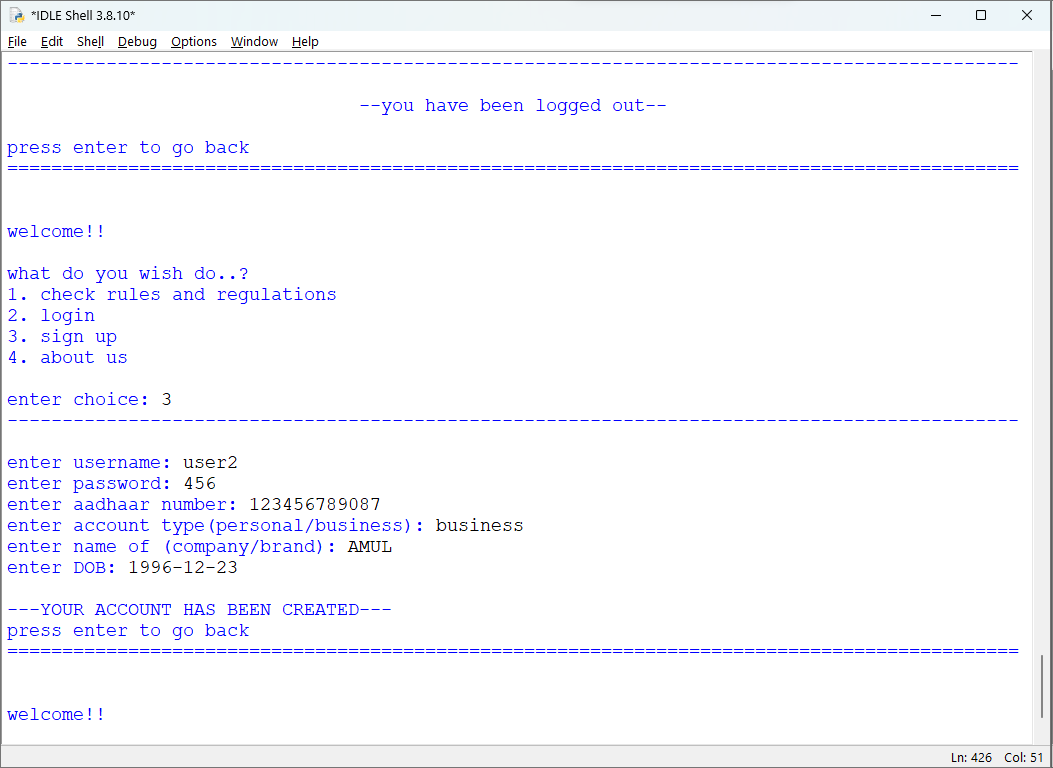
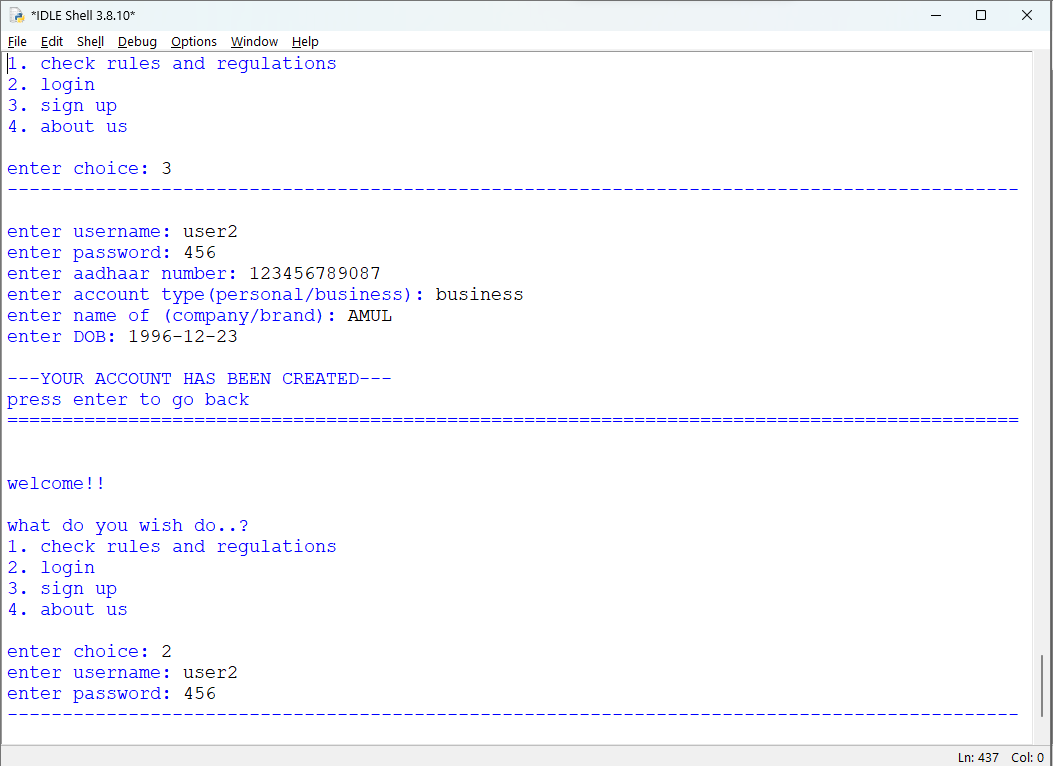
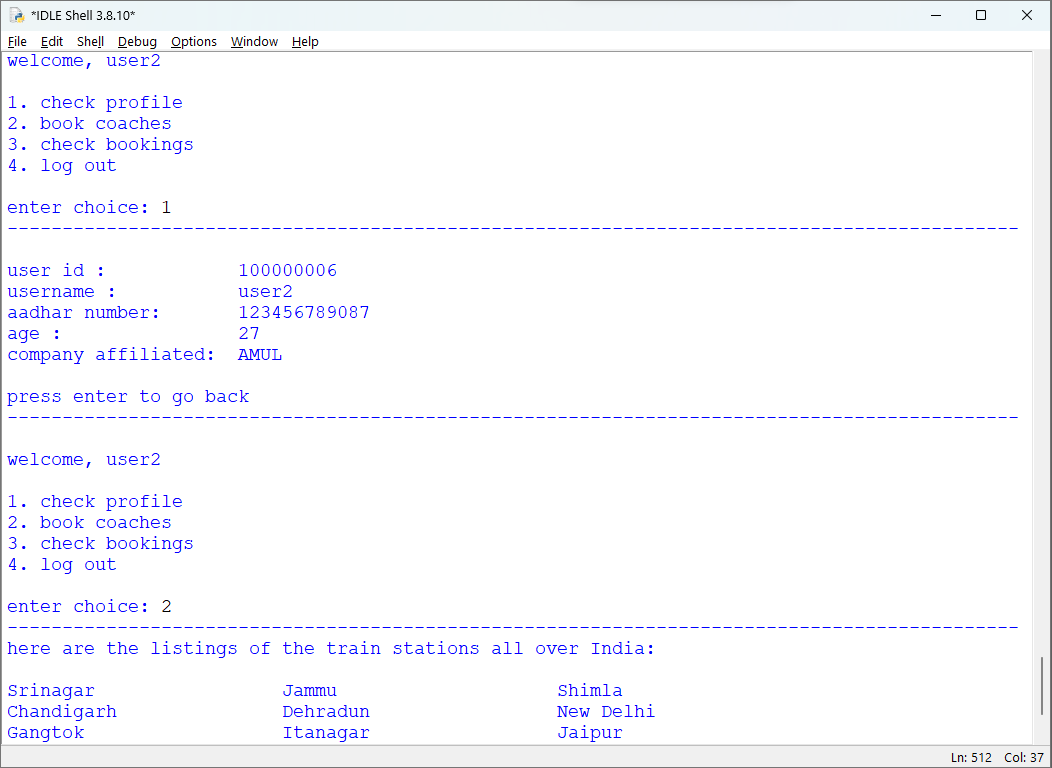
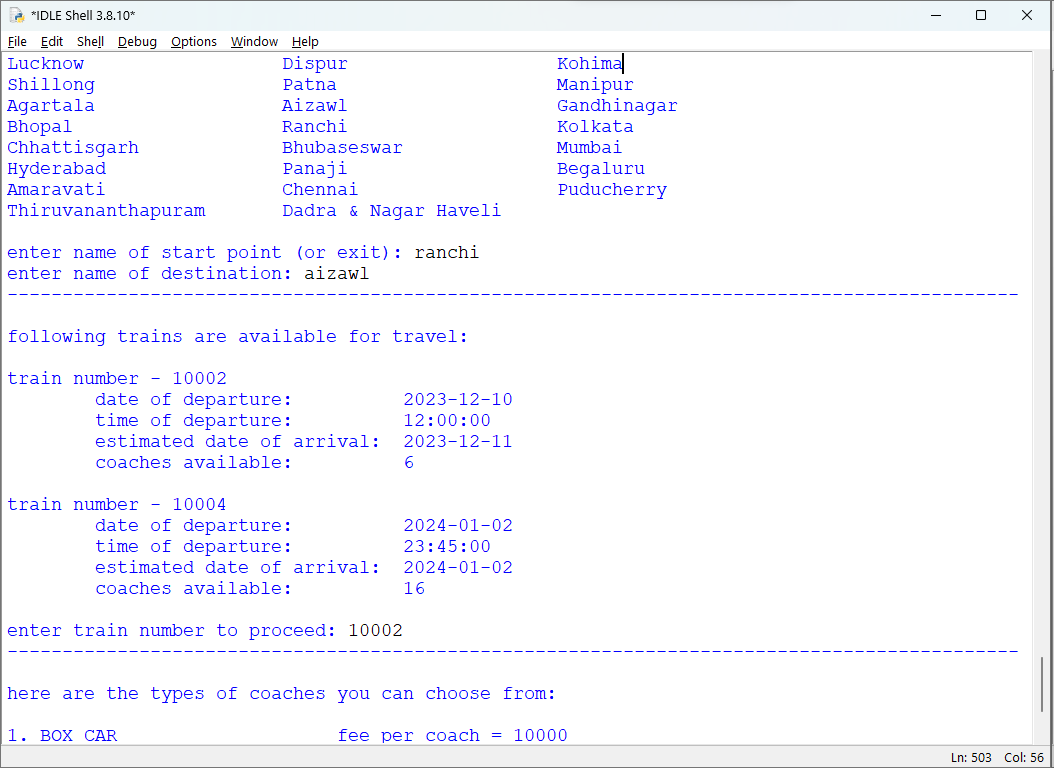
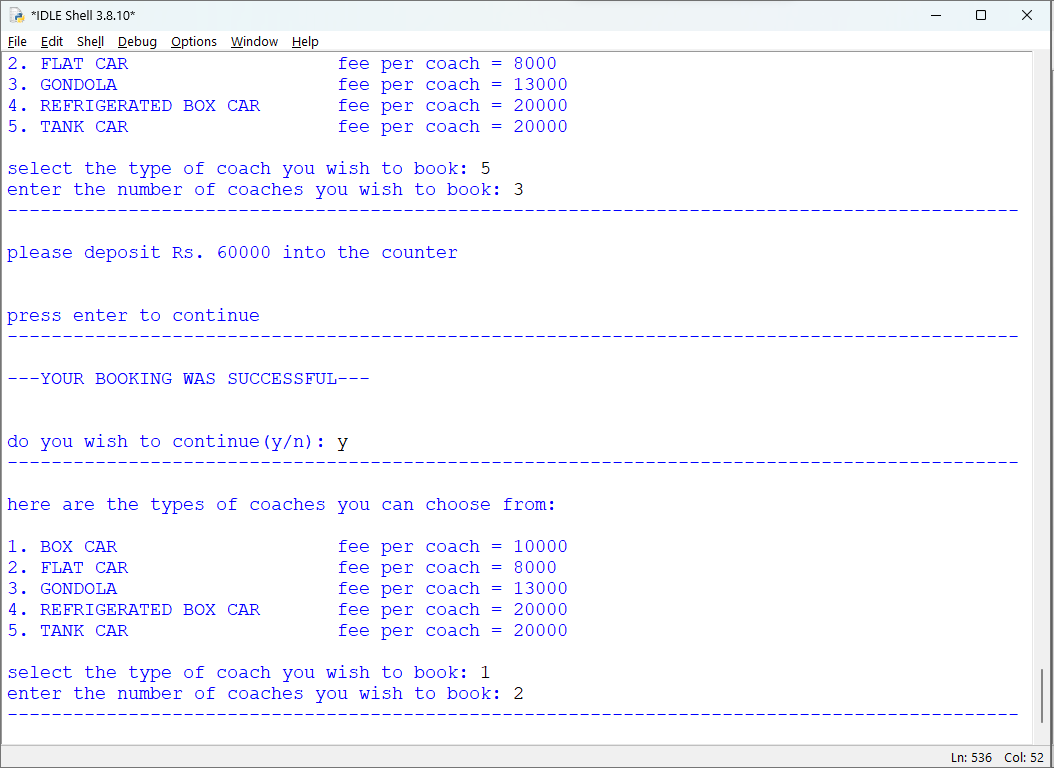
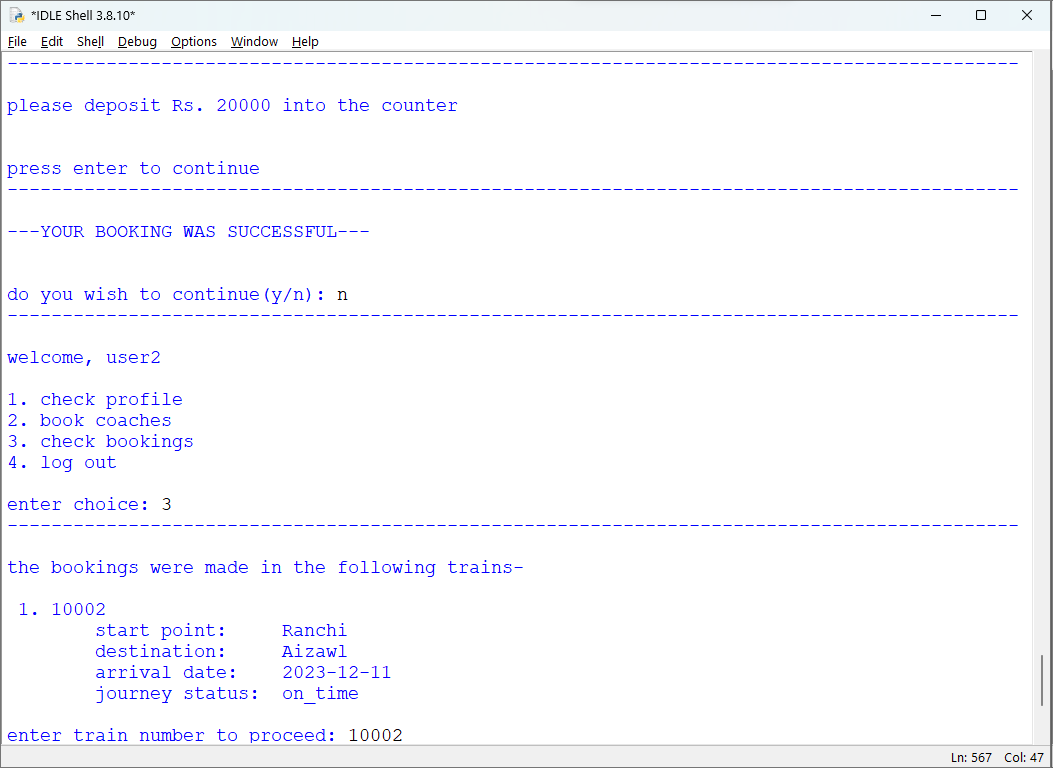
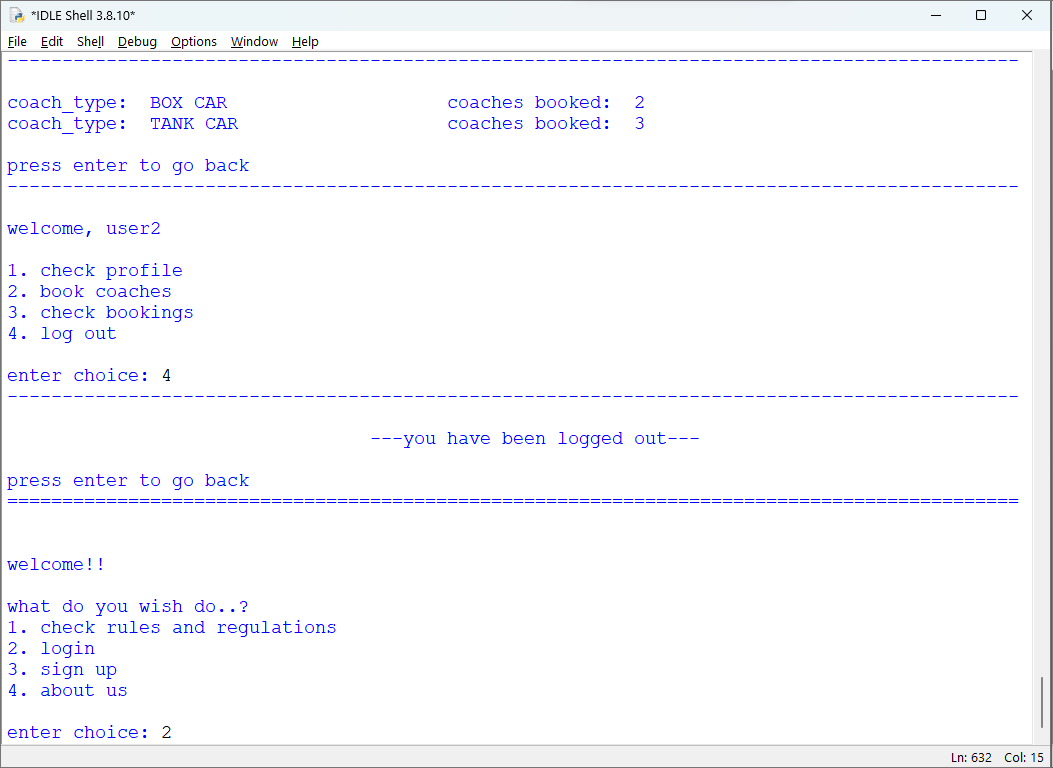
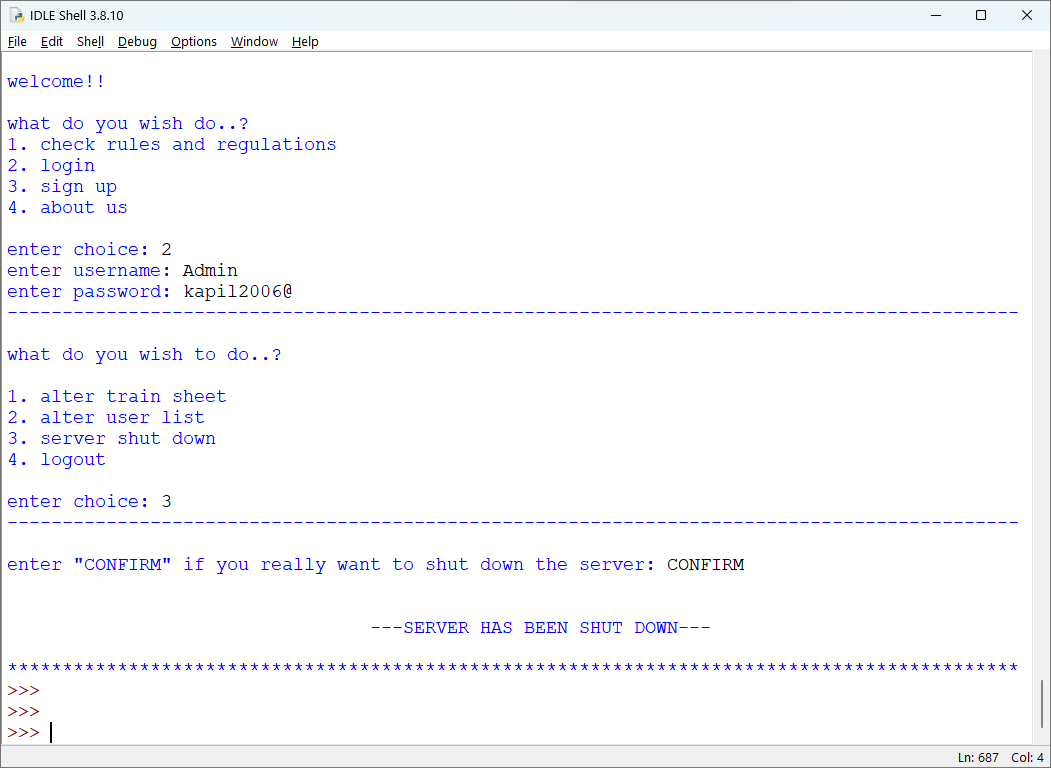
**Screen Shots**

**MySQL Screen Shots**  
(before running the program)

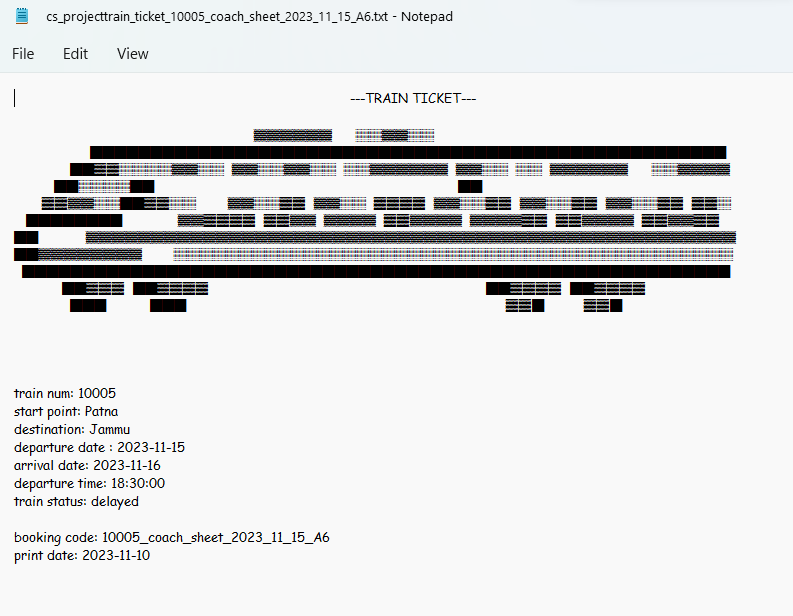
1. Tables in database  
   
2. Table for user details  
   
3. Empty table for user\_booking (business)  
   
4. Table for train details  
   
5. Empty table for coach\_sheet (passenger)  
   
6. Filled table for coach\_sheet (10001- passenger)  
   
7. Empty table for coach\_sheet (freight)  
   
8. Filled table for coach\_sheet (10002- freight)  
   
9. Table for different train stations  
   
10. Table for different type of coaches  
    

**Python Program Screen Shots**



 ****                   

**Saved text file screen shot**

****

**Conclusion**

In conclusion, the implementation of the Railway Management System represents a significant leap forward in enhancing the efficiency, and overall performance of railway operations. By seamlessly integrating advanced technology and predictive analytics, this system not only optimizes train scheduling and routing but also mitigates potential risks and improves the overall user experience.

The comprehensive nature of the Railway Management System ensures that railway operators can make informed decisions, respond promptly and maintain a high level of service reliability.

**Bibliography / References**

1. Arora, Sumita., *Computer science with Python: Textbook for class XII.* New Delhi: Dhanpat Rai & Co. (P), Ltd., 2023
2. Arora, Preeti, *Computer Science with python: Textbook for class-XII.* New Delhi: Sultan Chand & Sons (P) Ltd., 2023
3. Uppal, Shweta, *Computer Science* ; National Council of Education and Research, Page no.:1-32 and 111-172 New Delhi: Publication Department, 2023.
4. “Railway ticket reservation” accessed 7/11/2023 accessed by

https://www.geeksforgeeks.org/online-railway-ticket-reservation-system/

\*\*\*\*\*