



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

Karpool

Data Structures and Algorithms

Slot: B1

19BCE2074 Kulvir Singh

19BCE0032 Sudhanshu Sainath

19BCE0102 Pallavi

19BCE0037 Nishtha Singh

Dr. Sendhil Kumar K.S

Associate Professor Grade 1

June 2020

Aim:

The project which goes by the title of **Karpool** is an application which allots a chauffeur along with a car to the user travel. The application offers three types of rides which targets a huge set of users. The aim of the project is to allot the car and its driver in the fastest way possible and in a continuous manner to reduce waiting time.

Objective:

The main and primary objective of the project is that it allots the driver from its database and matches to the available car in the car database. On completion of the above process, user will get the information about the driver and the car allotted for his or her journey. Once the user gets the correct information delivered to him or her, only then the application would run successfully and its main objective would be complete.

Introduction:

The project built by this team deals with a basic problem faced by people in cities. The time wasted for waiting for a taxi or cab to travel has been an issue for a long time. Karpool is an application which helps the user with this problem and has attempted to solve it.

The application has reduced the waiting time for its user and aims at providing its services as soon as possible. The car allotting system has an improvised stack operated database which helps us in reducing the waiting time. As soon as a user requests, a request is made to the database and the free driver is pushed out of the stack-based database. A similar process is done for the car database and pops out the topmost available car.

Karpool as a project and application would reach out to various locations. It can also have a huge impact in places where there are no application-based cab services. Especially college students would be the major users of the application. In places where there are no inter-city travel programs or plans, the application would be quite useful. Also, in places where the majority is student-based crowd, travelling from a remote to a major-city has been an issue. Hence the application would be a huge hit in such a city as it would hugely cater to the need of the hour of college students when they are travelling back home.

Hence the reach and scope of the application is vast and widespread as explained above.

Data Structure:

Circular Stack

Arrays

The primary data structure used while making the project is Circular Stack.

The secondary data structure used while making the project is Arrays.

Circular Stack:

A circular stack is an advanced data structure that performs the basic operations of a stack from both the ends of the stack, i.e., the front end or the top of the stack and the back end or the bottom of the stack.

Operations of Circular Stack:

1. PushFront():

This operation appends an element into the from the front end of the stack or the top of the stack.

2. PushBack():

This operation appends an element into the back end or after the last element,ie, bottommost element present in the stack.

3. PopFront():

This operation deletes or removes an element from the stack at the top most position,ie, it deletes or removes the first element present in the stack.

4. PopBack():

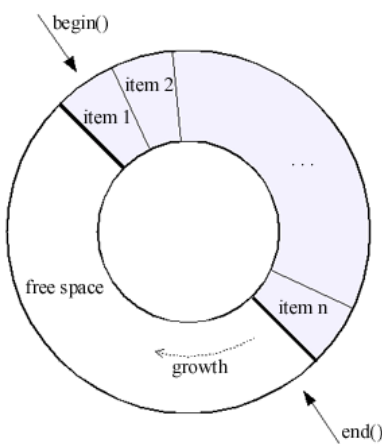
This operation deletes or removes an element from the stack at the bottom most position,ie, it deletes or removes the last element present in the stack.

Working of Circular Stack:

The following working is explained based on the basic concept of circular stack, however while implementing it in our project we have modified it to give least time complexity. The modifications have been explained in the methodology section of the report.

For the working of a circular stack we need a counter which would be assigned to negative one. On a request to add element into the stack we increase the counter by one and add the element in the array created by the programmer. For every insert request we follow the same process and increase the value of the counter by one. On deletion request, the element present in array at the location of the counter is deleted from the array and then the counter is decreased by one. For adding the deleted element back into the stack at the bottom end we need to increase the size of the array by one. On doing so a new memory location is created at the last position of the array and a new element is added back into the stack at the last position. Similarly, for deleting the last element from the array we simply search for the length of the array and remove it out of the array and reduce the size of the array by one.

Circular Stack's pictorial representation:-



Begin() marks the topmost element of the stack

End() marks the bottommost element of the stack

Arrays:

An array is collection of items stored at contiguous memory locations. The idea is to store multiple items of same type together. This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array. Arrays allow random access of elements. This makes accessing elements by position faster. Arrays have better cache locality that can make a pretty big difference in performance. An element is accessed by indexing the array name. This is done by placing the index of the element within square brackets after the name of the array.

Array indexes and respective data stored can be shown as:-

0	1	2	3	4
1000.0	2.0	3.4	7.0	50.0

Detailed Methodology:

The entire project has been coded in python programming language.

We have created a various file such as member's list, employees, cars, travel locations.

All such files are basically the data base files. The main feature of these files is that they have an array. The array is used to store the data. The array is then used as the stack and the circular stack methodology is introduced. After creating the data set for each database, a pop function is applied. In the terms if circular stack we pop the frontmost element from the stack and return it in the main code section under the heading of the file user_interface.

There is an additional function added to the employee data base. After we have performed the pop operation, we wait for the user to give the cue that the ride has been completed and then we append the driver which he had received when we performed the pop operation back to the bottom of the stack.

There is an additional function added to the car data base. After we have performed the pop operation, we wait for the user to give the cue that the ride has been completed and then we append the car name which he had received when we performed the pop operation back to the bottom of the stack.

The user interface file is where the application imports all the databases and uses it information to run the application. The user is asked to log in by inputting his phone number and password. The credentials are then checked if they are present in the database or not. If they are present then the application logs the user in. However if there is not match then the user is asked to create a new account. All the data asked for logging in the account is then stored inside a the database in the required array based circular stack.

Once the user is logged in the application displays the menu of choices of rides that the user can choose. There is also an option where the user can get to know the ride description. Once the user chooses to know each type of ride, the description is listed out and displayed on the console screen and the input for the type of ride reappears.

On inputting the choice of ride, the user gets the required information about the driver and the car allotted to him or her. This action is brought about by tapping into the database files. Once the request for a ride is made by the user, the control goes to the employee and car databases. The top most element of each database is selected and then pop operation is performed. Once we get the name of the car and driver, the control of the program is shifted back to the user interface and the elements of which were thrown out of the stack are returned. Then the returned elements are printed on the console.

The application then waits for the user to confirm the completion of the journey. On receiving the confirmation which is brought about by entering 0, the command is shifted to the car and employee database along with the driver name and car name. On entering the database, the returned names of the car and driver are appended to the respective car and employee database. This is brought about by using the circular stack operation of push back.

After the databases are refilled, the command shifts back to the user interface where the program logs the user out and the session is over with an option to re-login.

So, this is how the entire program forms a continuous chain of allotting cars and drivers on request of the user.

Algorithmic Architecture:

Greedy Algorithm

The entire project is based on using greedy algorithm technique which has helped in reducing the compiling time and time complexity.

A greedy algorithm is any algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage with the intent of finding a global optimum. In many problems, a greedy strategy does not usually produce an optimal solution, but nonetheless a greedy heuristic may yield locally optimal solutions that approximate a globally optimal solution in a reasonable amount of time.

For example, a greedy strategy for the travelling salesman problem (which is of a high computational complexity) is the following heuristic: "At each step of the journey, visit the nearest unvisited city." This heuristic does not intend to find a best solution, but it terminates in a reasonable number of steps; finding an optimal solution to such a complex problem typically requires unreasonably many steps. In mathematical optimization, greedy algorithms optimally solve combinatorial problems having the properties of matroids, and give constant-factor approximations to optimization problems with submodular structure.

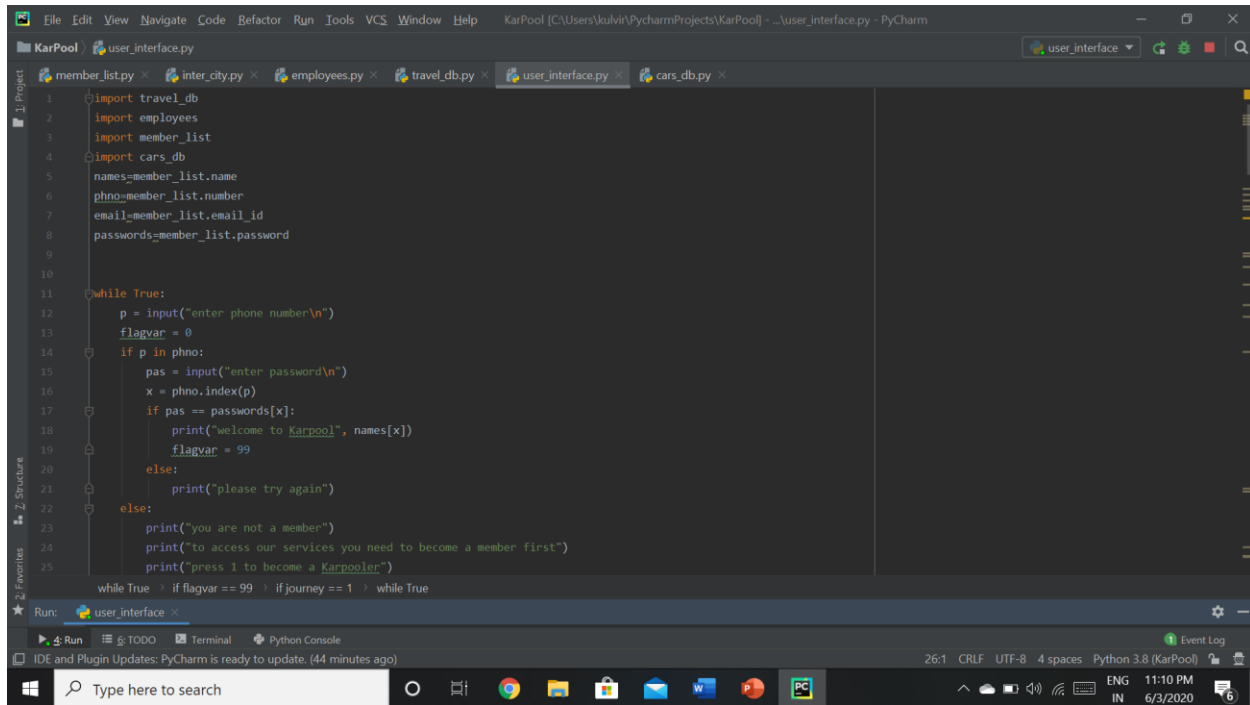
Greedy algorithms mostly (but not always) fail to find the globally optimal solution because they usually do not operate exhaustively on all the data. They can make commitments to certain choices too early which prevent them from finding the best overall solution later. For example, all known greedy coloring algorithms for the graph coloring problem and all other NP-complete problems do not consistently find optimum solutions. Nevertheless, they are useful because they are quick to think up and often give good approximations to the optimum.

If a greedy algorithm can be proven to yield the global optimum for a given problem class, it typically becomes the method of choice because it is faster than other optimization methods like dynamic programming. Examples of such greedy algorithms are Kruskal's algorithm and Prim's algorithm for finding minimum spanning trees, and the algorithm for finding optimum Huffman trees.

Greedy algorithms appear in network routing as well. Using greedy routing, a message is forwarded to the neighboring node which is "closest" to the destination. The notion of a node's location (and hence "closeness") may be determined by its physical location, as in geographic routing used by ad hoc networks. Location may also be an entirely artificial construct as in small world routing and distributed hash table.

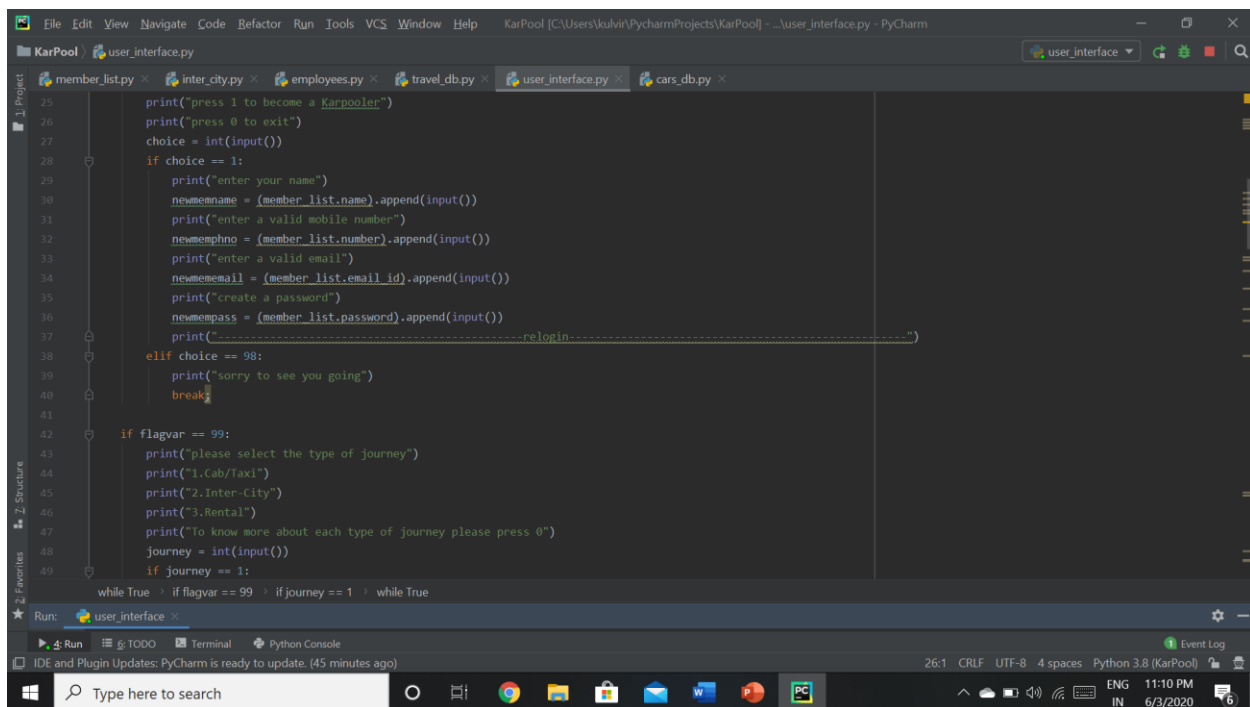
Demonstration:

Code:



This screenshot shows the first part of the `user_interface.py` file in the PyCharm IDE. The code imports several modules and sets up initial variables for a user interface. It includes a `while True` loop for user authentication.

```
1 import travel_db
2 import employees
3 import member_list
4 import cars_db
5 names=member_list.name
6 phno=member_list.number
7 email=member_list.email_id
8 passwords=member_list.password
9
10
11 while True:
12     p = input("enter phone number\n")
13     flagvar = 0
14     if p in phno:
15         pas = input("enter password\n")
16         x = phno.index(p)
17         if pas == passwords[x]:
18             print("welcome to Karpool", names[x])
19             flagvar = 99
20         else:
21             print("please try again")
22     else:
23         print("you are not a member")
24         print("to access our services you need to become a member first")
25         print("press 1 to become a Karpooler")
26
27 while True: > if flagvar == 99 > if journey == 1 > while True
```



This screenshot shows the second part of the `user_interface.py` file. It continues the `while True` loop, handling user registration and journey selection.

```
25 print("press 1 to become a Karpooler")
26 print("press 0 to exit")
27 choice = int(input())
28 if choice == 1:
29     print("enter your name")
30     newmemname = (member_list.name).append(input())
31     print("enter a valid mobile number")
32     newmemphno = (member_list.number).append(input())
33     print("enter a valid email")
34     newmememail = (member_list.email_id).append(input())
35     print("create a password")
36     newmempass = (member_list.password).append(input())
37     print("-----relogin-----")
38 elif choice == 98:
39     print("sorry to see you going")
40     break
41
42 if flagvar == 99:
43     print("please select the type of journey")
44     print("1.Cab/Taxi")
45     print("2.Inter-City")
46     print("3.Rental")
47     print("To know more about each type of journey please press 0")
48     journey = int(input())
49     if journey == 1:
50
51 while True: > if flagvar == 99 > if journey == 1 > while True
```

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ..user_interface.py - PyCharm
KarPool user_interface.py
member_list.py x inter_city.py x employees.py x travel_db.py x user_interface.py x cars_db.py x
49 if journey == 1:
50     print("please wait while we check the availability")
51
52     print("your driver for the ride will be ", employees.pop())
53     print("the car allotted to you is ", cars_db.pop())
54     while True:
55         print("press 0 on completion of the journey")
56         anoin = input()
57         if anoin=="0":
58             break
59
60
61
62
63 elif journey == 3:
64     print("please wait while we check the availability")
65     print("your ride will be delivered by : ", employees.pop())
66     print("the car allotted to you is ", cars_db.pop())
67     while True:
68         print("press 0 on completion of the journey")
69         anoin = input()
70         if anoin=="0":
71             break
72
73
74 while True:
75     if flagvar == 99:
76         if journey == 1:
77             while True:
```

Run: user_interface

IDE and Plugin Updates: PyCharm is ready to update. (45 minutes ago)

26:1 CRLF UTF-8 4 spaces Python 3.8 (KarPool)

ENG 11:10 PM 6/3/2020

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ..user_interface.py - PyCharm
KarPool user_interface.py
member_list.py x inter_city.py x employees.py x travel_db.py x user_interface.py x cars_db.py x
73
74 elif journey == 2:
75     print("Please view our travel map")
76     for ite in travel_db.destinations:
77         print(ite)
78     choicetravel1 = int(input())
79     print("you have selected your travel plan as ", travel_db.destinations[choicetravel1 - 1])
80     print("please wait while we check the availability")
81     print("your driver for the ride will be ", employees.pop())
82     print("the car allotted to you is ", cars_db.pop())
83     while True:
84         print("press 0 on completion of the journey")
85         anoin = input()
86         if anoin=="0":
87             break
88
89 elif journey == 0:
90     print("-----description of each type-----")
91     print("1>Cab/Taxi :-")
92     print("This will allot you a car along with a chauffeur.")
93     print("Please tell the chauffeur your destination of choice and you will be dropped at your destination.")
94     print("Rates will be applied based on the meter fitted in the car")
95     print("2>Inter-City :-")
96     print("This will allot you a car along with a chauffeur.")
97     print("This ride will take you across cities.")
98
99 while True:
100     if flagvar == 99:
101         if journey == 1:
102             while True:
```

Run: user_interface

IDE and Plugin Updates: PyCharm is ready to update. (45 minutes ago)

26:1 CRLF UTF-8 4 spaces Python 3.8 (KarPool)

ENG 11:10 PM 6/3/2020

The screenshot shows the PyCharm IDE with the file `user_interface.py` open. The code is as follows:

```
103 print("Rates will be applied according to the meter fitted in the car")
104 print("please select the type of journey")
105 print("1.Cab/Taxi")
106 print("2.Inter-City")
107 print("3.Rental")
108 journey1 = int(input())
109 if journey1 == 1:
110     print("please wait while we check the availability")
111     print("your driver for the ride will be ", employees.pop())
112     print("the car allotted to you is ", cars_db.pop())
113     while True:
114         print("press 0 on completion of the journey")
115         anoin = input()
116         if anoin == "0":
117             break
118
119 elif journey1 == 3:
120     print("please wait while we check the availability")
121     print("your car will be dropped by : ", employees.pop())
122     print("the car allotted to you is ", cars_db.pop())
123     while True:
124         print("press 0 on completion of the journey")
125         anoin = input()
126         if anoin == "0":
127             break
```

The bottom status bar shows: 26:1 CRLF UTF-8 4 spaces Python 3.8 (KarPool) 11:10 PM 6/3/2020.

The screenshot shows the PyCharm IDE with the file `user_interface.py` open. The code is as follows:

```
136 print("your driver for the ride will be ", employees.pop())
137 print("the car allotted to you is ", cars_db.pop())
138 while True:
139     print("press 0 on completion of the journey")
140     anoin = input()
141     if anoin == "0":
142         break
143
144 else:
145     print("invalid choice please re-login")
146
147 else:
148     print("invalid choice please re-login")
149 print("thank you for choosing KarPool\n\n\n\n\n")
150 print("we are logging you out automatically for safety reasons")
151 print("press 1 to re-login")
152 finalcheck = (input())
153 if finalcheck != "1":
154     print("you have pressed the wrong button and hence all unsaved data has been deleted for safety reasons")
155     break
156
157
```

The bottom status bar shows: 26:1 CRLF UTF-8 4 spaces Python 3.8 (KarPool) 11:11 PM 6/3/2020.

File Edit View Navigate Code Refactor Run Tools VCS Window Help KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ...cars_db.py - PyCharm

KarPool cars_db.py

```
1 cars=[  
2     "Maruti Suzuki Vitara brezza", "Hyundai venue",  
3     "Hyundai grand i10", "Hyundai creta", "tata nexon", "Hyundai elite i20",  
4  
5     "Maruti ertiga", "Toyota innova", "Mahindra xuv 500", "Maruti eeco", "Maruti tuv300", "Chevrolet enjoy", "tata sumo"  
6     "Toyota etios", "Suzuki dzire",  
7     "Hyundai xcent", "tata altroz", "Hyundai Aura",  
8     "Maruti Suzuki ciaz", "honda city", "tata tiger", "tata indigo",  
9     "Chevrolet beat", "Suzuki ritz",  
10    "Maruti wagonR", "Hyundai i20",  
11  
12    "honda amaze", "tata tiago", "tata indica", "tata nano", "Suzuki alto",  
13    "Maruti Suzuki swift", "Maruti Suzuki santro", "Chevrolet spark", "hyundai i10", "Renault kwid"  
14  
15    "Kia seltos", "Hyundai venue", "Mahindra scorpio", "Toyota fortuner", "MG Hector", "Tata nexon"  
16    "Renault triber", "Tata harrier", "Toyota velfire", "honda city",  
17  
18    "ford ecosport", "Maruti s-presso",  
19    "volkswagen polo", "Maruti celerio", "Maruti ignis", "Maruti XL6",  
20    "Maruti s-cross", "Mahindra e2o plus", "Toyota glanza", "ford figo"  
21    "Honda HRV"]  
22  
23 def pop():  
24     x=cars.pop(0)  
25     cars.append(x)  
26     return x
```

Run: user_interface

IDE and Plugin Updates: PyCharm is ready to update. (50 minutes ago)

1:7 CRLF UTF-8 4 spaces Python 3.8 (KarPool)

Type here to search

File Edit View Navigate Code Refactor Run Tools VCS Window Help KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ...travel_db.py - PyCharm

KarPool travel_db.py

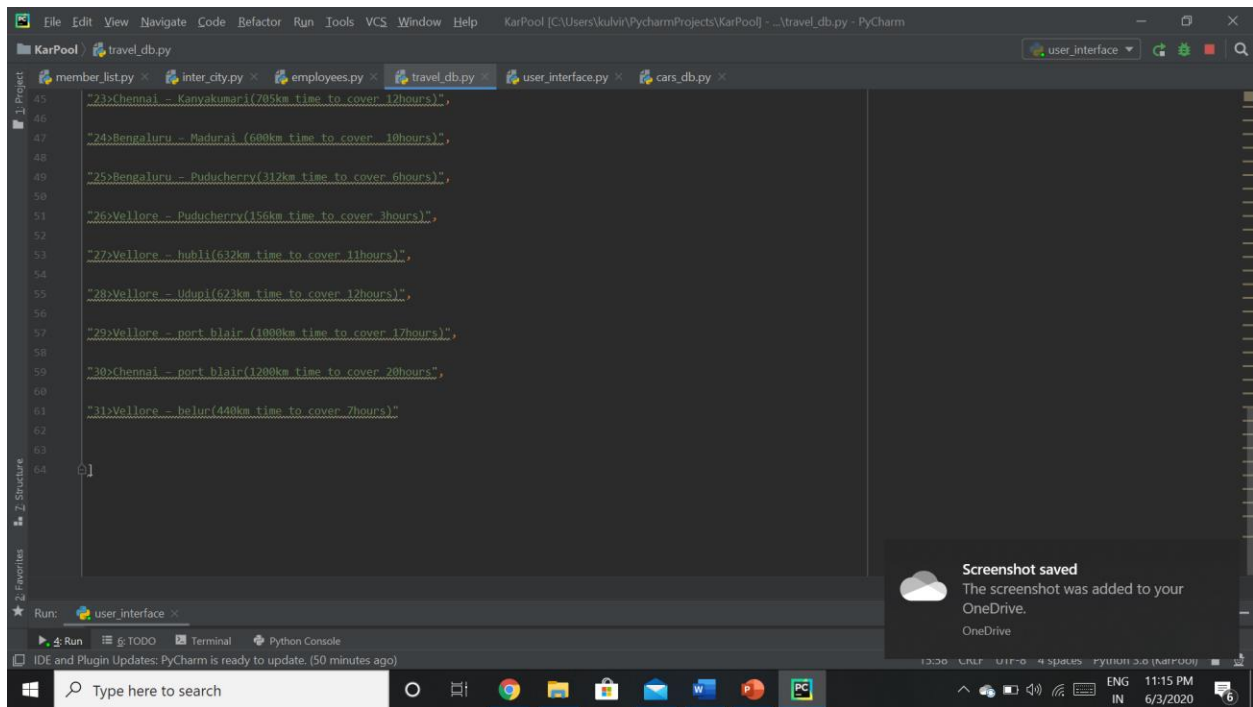
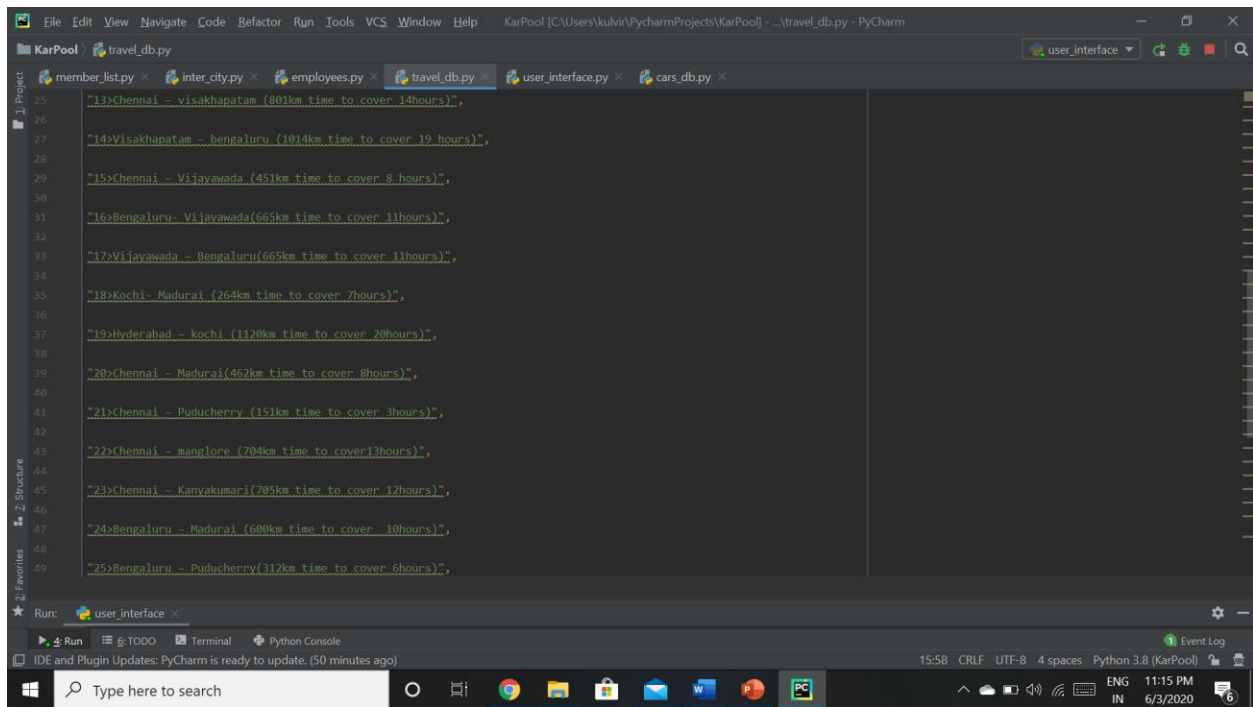
```
1 destinations=[  
2     "1>Vellore - Chennai(137km time to cover 3 hours)",  
3  
4     "2>Vellore - bengaluru(212km time to cover 3hours.45 mins )",  
5  
6     "3>Vellore - mysore city.(346km time to cover 6 hours)",  
7  
8     "4>Vellore - Tirupati.(107km time to cover 2 hours)",  
9  
10    "5>Bengaluru - gnty.(265km time to cover 6 hours)",  
11  
12    "6>Chennai to Munnar(586km time to cover 11 hours)",  
13  
14    "7>Bengaluru - kochi.(558km time to cover 10 hours)",  
15    "8>Bengaluru - Coimbatore.(364km time to cover 7 hours)",  
16  
17    "9>Bengaluru - Trivandrum.(685km time to cover 13 hours)",  
18  
19    "10>Chennai - Trivandrum.(726km time to cover 14 hours)",  
20  
21    "11>Kattadi - Hyderabad.(611km time to cover 10hours)",  
22  
23    "12>Chennai - hyderabad.(626km time to cover 11hours)",  
24  
25    "13>Chennai - visakhapatnam.(801km time to cover 14hours)",
```

Run: user_interface

IDE and Plugin Updates: PyCharm is ready to update. (50 minutes ago)

15:58 CRLF UTF-8 4 spaces Python 3.8 (KarPool)

Type here to search



KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ...employees.py - PyCharm

member_list.py x inter_city.py x employees.py x travel_db.py x user_interface.py x cars_db.py x

```
1 emp=[ "Roger","Stuart","Robin","Arron","Rohan","Satyam","Ayush","Harsh","Iushar",
2       "Manav","Mohit","Kumar","Sanga","Broad","Finch","Simon","Ethan","Max","Tom","Ben","Nelson","Roger",
3       "Suman","Robert","Marco","Reus","Leo","Ronaldo","Neymar","Frank","Lampard","Rohan","Anitej","Shrey",
4       "Shreyas","Iyer","Ram","Shayam","Suppandi"]
5
6 def pop():
7     x=emp.pop(0)
8     emp.append(x)
9     return x
```

Run: user_interface x

IDE and Plugin Updates: PyCharm is ready to update. (50 minutes ago)

2:7 CRLF UTF-8 4 spaces Python 3.8 (KarPool)

11:15 PM 6/3/2020

KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ...member_list.py - PyCharm

member_list.py x inter_city.py x employees.py x travel_db.py x user_interface.py x cars_db.py x

```
1 names=["Anuradha",
2       "Rohit",
3       "Sharan",
4       "Rahul",
5       "Darshan",
6       "Kirti",
7       "Varsha",
8       "Divya",
9       "Shreya",
10      "Ayush",
11      "Kulvic",
12      "Pallavi",
13      "Anil",
14      "Pallavi",
15      "Suhani",
16      "Jay",
17      "Chirag",
18      "Amir",
19      "Rahul",
20      "Kirti",
21      "Mahi",
22      "Madhu",
23      "Priya",
24      "Yansh",
25      "Yanshika",
```

Run: user_interface x

IDE and Plugin Updates: PyCharm is ready to update. (50 minutes ago)

73:20 CRLF UTF-8 4 spaces Python 3.8 (KarPool)

11:16 PM 6/3/2020

File Edit View Navigate Code Refactor Run Tools VCS Window Help KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ...member_list.py - PyCharm

KarPool member_list.py user_interface

```
25 "Vanshika",
26 "Kartik",
27 "Harshita",
28 "Sai",
29 "Saurabh",
30 "Varun",
31 "Mohit",
32 "Ujjwal"
33 ]
34 number=["9321232945",
35 "7312323356",
36 "7868763321",
37 "9711165349",
38 "9050673453",
39 "8764536783",
40 "8875645373",
41 "9567482637",
42 "7601625774",
43 "8987547383","7541097627",
44 "9711199277"
45 ,
46 "7584974839",
47 "8494937483",
48 "9564738475",
49 "9837464364",
```

Run: user_interface

IDE and Plugin Updates: PyCharm is ready to update. (50 minutes ago)

73:20 CRLF UTF-8 4 spaces Python 3.8 (KarPool)

Type here to search

File Edit View Navigate Code Refactor Run Tools VCS Window Help KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ...member_list.py - PyCharm

KarPool member_list.py user_interface

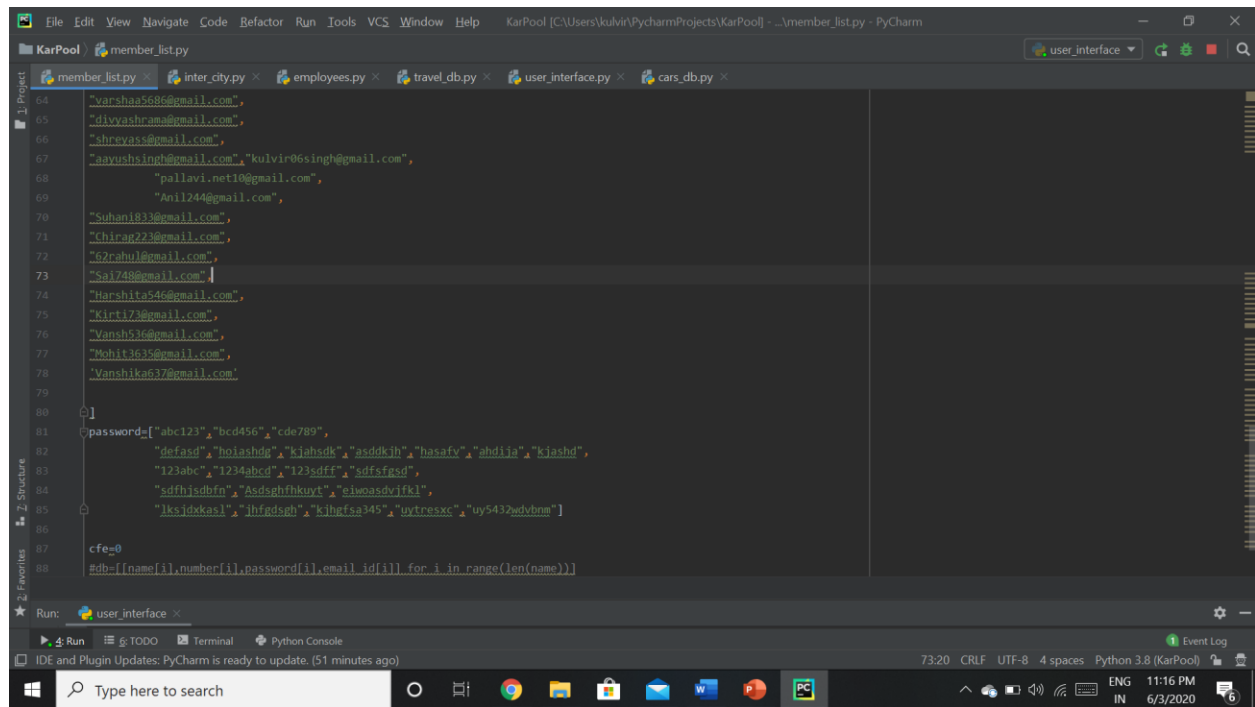
```
43 "8987547383","7541097627",
44 "9711199277"
45 ,
46 "7584974839",
47 "8494937483",
48 "9564738475",
49 "9837464364",
50 "7494746384",
51 "8374658394",
52 "8476398465",
53 "8936475674",
54 "9847563849",
55 "8475638495"
56 ]
57 email_id=["arunadha29@gmail.com",
58 "rohit453@gmail.com",
59 "asharann78@gmail.com",
60 "rahulkumar54@gmail.com",
61 "darshan5634@gmail.com",
62 "kirti116@gmail.com",
63 "varsha5686@gmail.com",
64 "divyashrama@gmail.com",
65 "shreyas@gmail.com",
66 "aayushsingh@gmail.com","kulvir06singh@gmail.com",
```

Run: user_interface

IDE and Plugin Updates: PyCharm is ready to update. (51 minutes ago)

73:20 CRLF UTF-8 4 spaces Python 3.8 (KarPool)

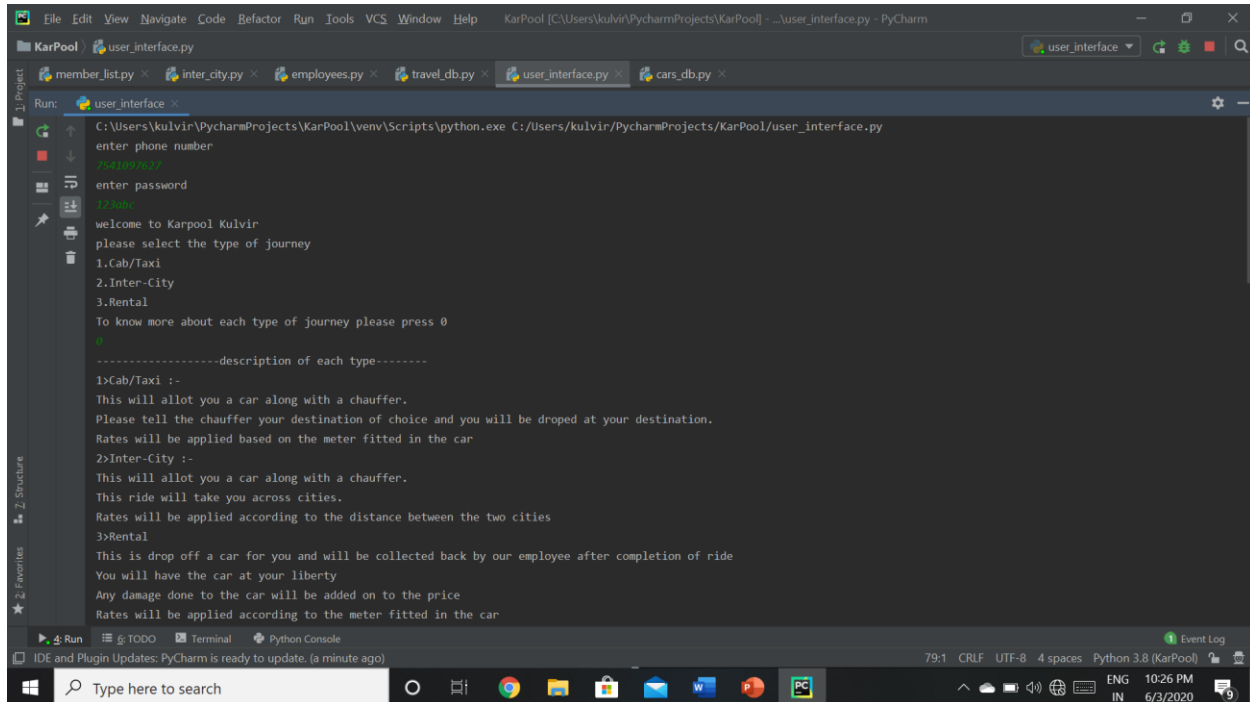
Type here to search



```
64 "varsha5686@gmail.com",
65 "divyasharma@gmail.com",
66 "shreyass@gmail.com",
67 "aayushsingh@gmail.com", "kulvir06singh@gmail.com",
68 "pallavi.net10@gmail.com",
69 "Anil244@gmail.com",
70 "Suhani833@gmail.com",
71 "Chirag223@gmail.com",
72 "62rahul@gmail.com",
73 "Sai748@gmail.com",
74 "Harshita546@gmail.com",
75 "Kirti173@gmail.com",
76 "Vansh536@gmail.com",
77 "Mohit3635@gmail.com",
78 "Vanshika637@gmail.com"
79
80
81 password=["abc123","bcd456","cde789",
82 "defasd","holaashdg","kiahskd","asddkjh","hasafv","ahdila","kiashd",
83 "123abc","1234abcd","123sdf","sdfsfgsd",
84 "sdfhjsdbfn","Asdsgfhfkuyt","eiwoasdvjkl",
85 "lksldxkasi","jhfgdsg","kjhgfsa345","uytresxc","uy5432wdvbnm"]
86
87 cfe=0
88 #db=[(name[i],number[i],password[i],email_id[i]) for i in range(len(name))]
```

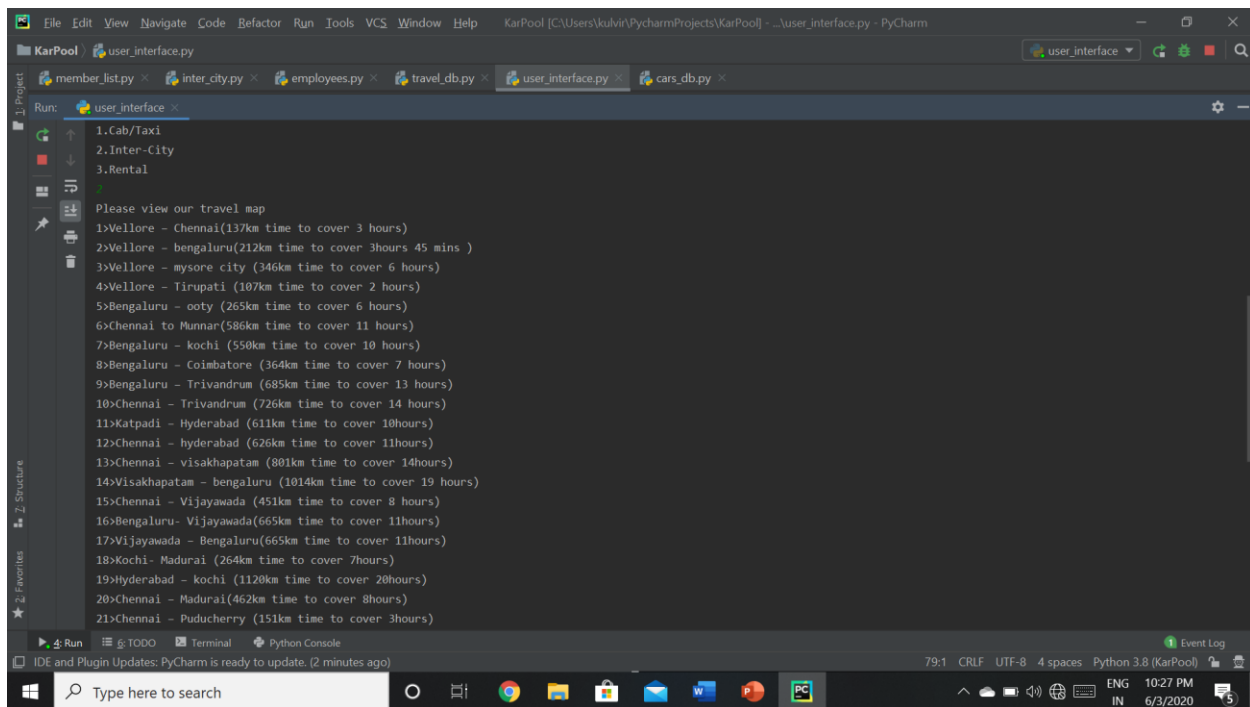
The entire code comprises of the user interface file supported by the various databases.

Demo Run:



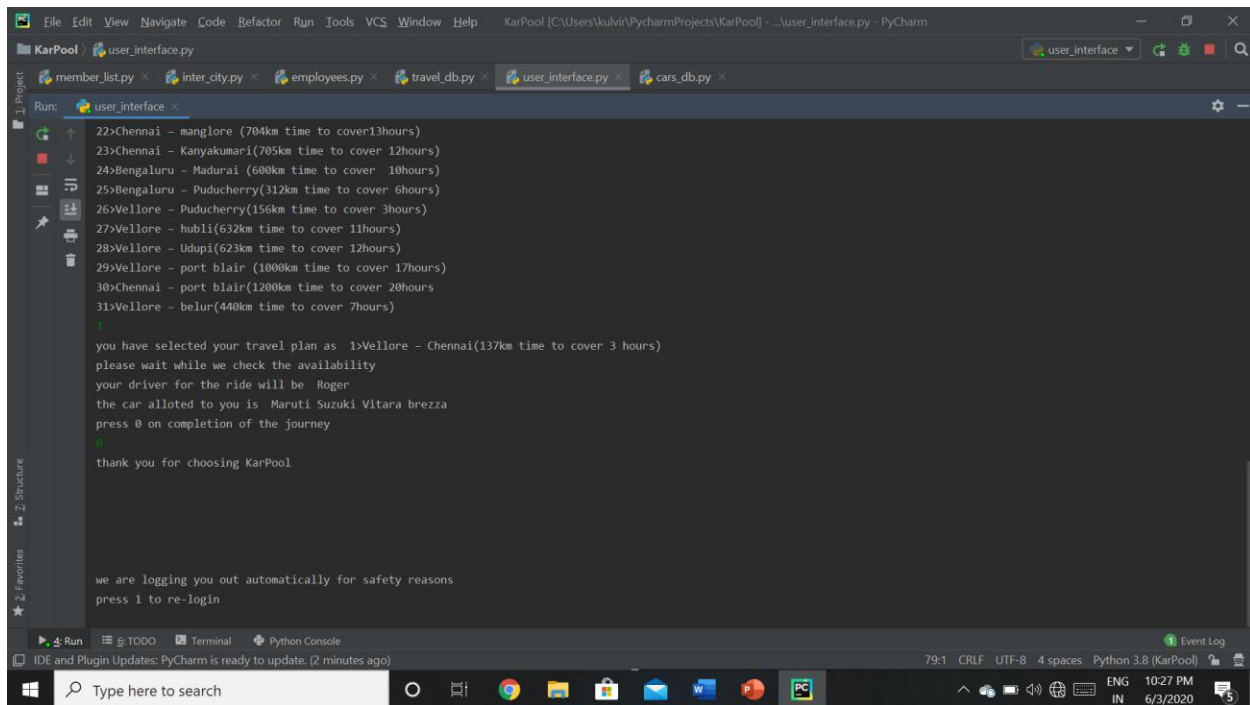
```
File Edit View Navigate Code Refactor Run Tools VCS Window Help KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ...user_interface.py - PyCharm
KarPool user_interface.py
member_list.py inter_city.py employees.py travel_db.py user_interface.py cars_db.py
Run: user_interface
C:\Users\kulvir\PycharmProjects\KarPool\venv\Scripts\python.exe C:\Users\kulvir\PycharmProjects\KarPool\user_interface.py
enter phone number
123456789
enter password
123456
welcome to Karpool Kulvir
please select the type of journey
1.Cab/Taxi
2.Inter-City
3.Rental
To know more about each type of journey please press 0
0
-----description of each type-----
1>Cab/Taxi :-
This will allot you a car along with a chauffer.
Please tell the chauffer your destination of choice and you will be dropped at your destination.
Rates will be applied based on the meter fitted in the car
2>Inter-City :-
This will allot you a car along with a chauffer.
This ride will take you across cities.
Rates will be applied according to the distance between the two cities
3>Rental
This is drop off a car for you and will be collected back by our employee after completion of ride
You will have the car at your liberty
Any damage done to the car will be added on to the price
Rates will be applied according to the meter fitted in the car
Run TODO Terminal Python Console
IDE and Plugin Updates: PyCharm is ready to update. (a minute ago) 79:1 CRLF UTF-8 4 spaces Python 3.8 (KarPool) ENG IN 10:26 PM 6/3/2020
```

The **above** screen-shot shows the user logging into his account and asking the application to explain about the types of rides.



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help KarPool [C:\Users\kulvir\PycharmProjects\KarPool] - ...user_interface.py - PyCharm
KarPool user_interface.py
member_list.py inter_city.py employees.py travel_db.py user_interface.py cars_db.py
Run: user_interface
1.Cab/Taxi
2.Inter-City
3.Rental
Please view our travel map
1>Vellore - Chennai(137km time to cover 3 hours)
2>Vellore - bengaluru(212km time to cover 3hours 45 mins )
3>Vellore - mysore city (346km time to cover 6 hours)
4>Vellore - Tirupati (107km time to cover 2 hours)
5>Bengaluru - ooty (265km time to cover 6 hours)
6>Chennai to Munnar(586km time to cover 11 hours)
7>Bengaluru - kochi (550km time to cover 10 hours)
8>Bengaluru - Coimbatore (364km time to cover 7 hours)
9>Bengaluru - Trivandrum (685km time to cover 13 hours)
10>Chennai - Trivandrum (726km time to cover 14 hours)
11>Katpadi - Hyderabad (611km time to cover 10hours)
12>Chennai - hyderabad (626km time to cover 11hours)
13>Chennai - visakhapatam (801km time to cover 14hours)
14>Visakhapatam - bengaluru (1014km time to cover 19 hours)
15>Chennai - Vijayawada (451km time to cover 8 hours)
16>Bengaluru- Vijayawada(665km time to cover 11hours)
17>Vijayawada - Bengaluru(665km time to cover 11hours)
18>Kochi- Madurai (264km time to cover 7hours)
19>Hyderabad - kochi (1120km time to cover 20hours)
20>Chennai - Madurai(462km time to cover 8hours)
21>Chennai - Puducherry (151km time to cover 3hours)
Run TODO Terminal Python Console
IDE and Plugin Updates: PyCharm is ready to update. (2 minutes ago) 79:1 CRLF UTF-8 4 spaces Python 3.8 (KarPool) ENG IN 10:27 PM 6/3/2020
```

The **above** screen-shot shows the user choosing the type of ride he wants and the following travel-plans are displayed.

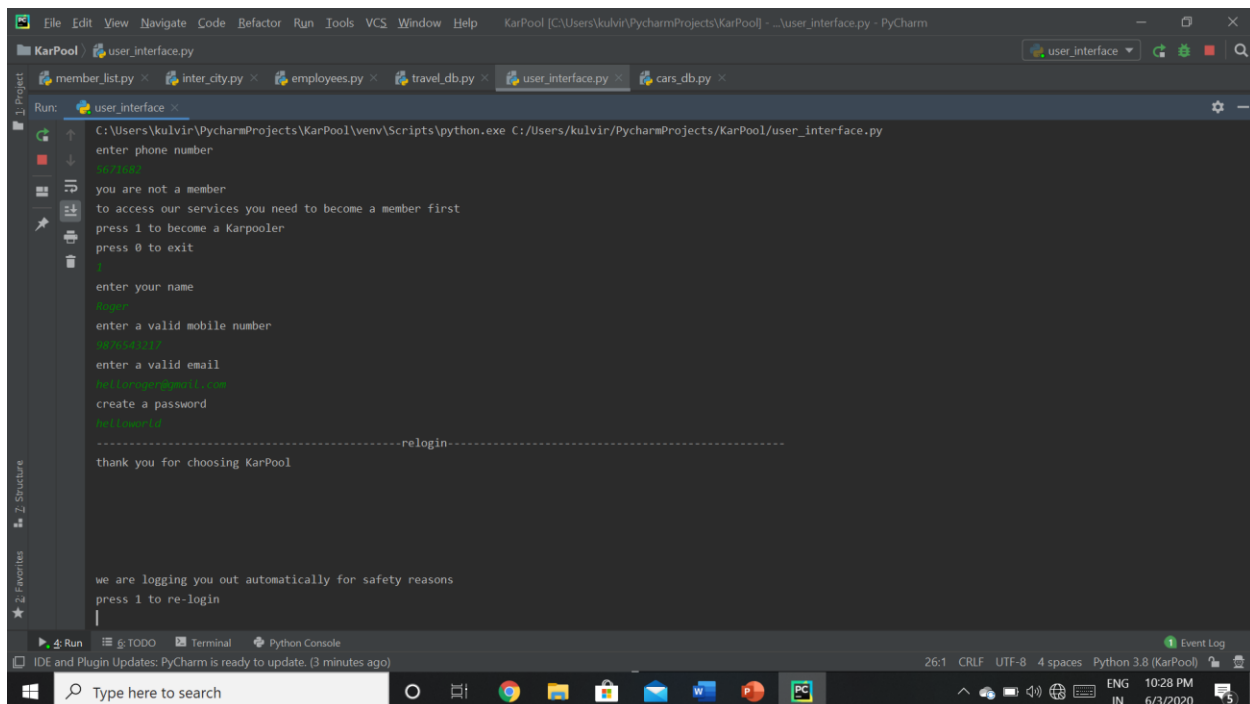


The screenshot shows the PyCharm IDE with a project named 'KarPool'. The 'Run' console displays the output of the 'user_interface.py' script. The script lists 11 travel options with their respective distances and travel times. A user has selected '1>Vellore - Chennai(137km time to cover 3 hours)'. The program then displays the driver's name 'Roger' and the car model 'Maruti Suzuki Vitara brezza'. The user is prompted to press '0' on completion of the journey, which they do. The program then displays 'thank you for choosing KarPool' and 'we are logging you out automatically for safety reasons press 1 to re-login'.

```
Run: user_interface.py
22>Chennai - manglore (704km time to cover 13hours)
23>Chennai - Kanyakumari(705km time to cover 12hours)
24>Bengaluru - Madurai (600km time to cover 10hours)
25>Bengaluru - Puducherry(312km time to cover 6hours)
26>Vellore - Puducherry(156km time to cover 3hours)
27>Vellore - hubli(632km time to cover 11hours)
28>Vellore - Udupi(623km time to cover 12hours)
29>Vellore - port blair (1000km time to cover 17hours)
30>Chennai - port blair(1200km time to cover 20hours)
31>Vellore - belur(440km time to cover 7hours)
1
you have selected your travel plan as 1>Vellore - Chennai(137km time to cover 3 hours)
please wait while we check the availability
your driver for the ride will be Roger
the car allotted to you is Maruti Suzuki Vitara brezza
press 0 on completion of the journey
0
thank you for choosing KarPool

we are logging you out automatically for safety reasons
press 1 to re-login
```

The **above** screen-shot shows that the user has chosen a particular travel plan. The program then displays the driver allotted and the car and waits for the completion of the journey. Once the journey is completed the user inputs 0 and is logged out.



The screenshot shows the PyCharm IDE with the same 'KarPool' project. The 'Run' console displays the output of the 'user_interface.py' script for a new user registration. The user is prompted to enter their phone number, name, valid mobile number, valid email, and create a password. After successful registration, the program displays 'thank you for choosing KarPool' and 'we are logging you out automatically for safety reasons press 1 to re-login'.

```
Run: user_interface.py
C:\Users\kulvir\PycharmProjects\KarPool\venv\Scripts\python.exe C:/Users/kulvir/PycharmProjects/KarPool/user_interface.py
enter phone number
9876543210
you are not a member
to access our services you need to become a member first
press 1 to become a Karpooler
press 0 to exit
1
enter your name
kulvir
enter a valid mobile number
9876543210
enter a valid email
kulvir@gmail.com
create a password
12345678
-----relogin-----
thank you for choosing KarPool

we are logging you out automatically for safety reasons
press 1 to re-login
1
```

The **above** screen-shot demonstrates the signing up of a new user.

Conclusion:

The entire application process is in a fully working phase. The user is able to log in or sign up as per the requirement. The user is successfully added to the database and verified. On logging in the user is able to book a cab. The driver and car information is sent and the application awaits the confirmation of completion. On receiving confirmation of completion from the user the application ends the user's session and is asked to re-login. Therefore, to conclude the entire application is running without any errors. Karpool can have a huge impact in places where there are no application-based cab services. Especially college students would be the major users of the application. In places where there are no inter-city travel programs or plans, the application would be quite useful. Also, in places where the majority is student-based crowd, travelling from a remote to a major-city has been an issue. Hence the application would be a huge hit in such a city as it would hugely cater to the need of the hour of college students when they are travelling back home.

References :

Websites:

<https://docs.python.org/3/tutorial/index.html>

https://www.w3schools.com/python/python_datatypes.asp

https://www.w3schools.com/python/python_lists.asp

<https://docs.python.org/3/reference/import.html>

https://en.wikipedia.org/wiki/Greedy_algorithm#:~:text=A%20greedy%20algorithm%20is%20a%20finding%20a%20global%20optimum.

<https://www.hackerearth.com/practice/algorithms/greedy/basics-of-greedy-algorithms/tutorial/>

<https://www.c-sharpcorner.com/UploadFile/6f0898/the-circular-stack-an-advance-in-data-structure/>

<https://www.geeksforgeeks.org/introduction-to-arrays/>

Books:

Introduction to Algorithms, Third Edition

By Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein