A Mini Project Report

on

**FIND MY RESTAURANT**

**Course:** Object Oriented Programming Lab

Sem: III Sec: CSE-B

By

NK.Pranesh

1602-19-733-088

B.Varun

1602-19-733-116



**Department of Computer Science & Engineering**

**Vasavi College of Engineering (Autonomous)**

**Ibrahimbagh, Hyderabad-31**

**2020**

**Acknowledgment**

We respect and thank M. Sashi Kumar sir for constant support and motivation which helped us immensely in completing our project successfully.

|  |  |  |
| --- | --- | --- |
| 1 | Abstract | 4 |
| 2 | Introduction | 5 |
| 3 | Design | 6 |
| 4 | Code/Implementation | 7 |
| 5 | Outputs | 17 |
| 6 | Conclusion | 19 |
| 7 | References | 19 |

**TABLE OF CONTENTS:**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Content** | **Page.No** |

**1.ABSTRACT**

**OBJECTIVE:**

People often get confused which restaurant to choose for good dining experience. They would want to know which restaurants are near them. They would not know the route to the restaurant he chooses. This application is the solution for all these problems. The user can find all retaurants, check their reviews. He can also find the best route to the restaurant from his hotel. The reviews and ratings for each restaurant helps the user to identify his best suited restaurant. The user can find all the restaurants near him and choose the best one. The admin can also add new restaurants and new locations in the map. This option is only available to the admin. So, this application allows users to find their restaurant easily and also get the best route from their location to the restautant chosen.

**WORKING :**

The code will ask the user to login and enter his present location. When the user types his location, he will be displayed with two options, i.e, whether he wants to search a particular restaurant or he wants to check the restaurants near him. When the user chooses the option to search a particular restaurant, he is asked to enter the restaurant name. The details of the restaurant are displayed along with its ratings and reviews.The shortest path ,i.e,from user location to the restaurant chosen, is also shown. He can also add his own rating and review. When the user chooses the option of finding restaurants near him, he will be displayed the restaurants near his entered location. Then he can check any of those restaurants.

Concepts used: Maps(Shortest path algorithm), File Handling.

Language used : Java.

**2.Introduction**

Find my restaurant is an application through which the users can check various restaurants and their reviews along the shortest path from his location to the restaurant. People often get confused which restaurant to choose for good dining experience. They would want to know which restaurants are near them. They would not know the route to the restaurant he chooses. This application is the solution for all these problems. The user can find all retaurants, check their reviews. He can also find the best route to the restaurant from his hotel. The reviews and ratings for each restaurant helps the user to identify his best suited restaurant. The user can find all the restaurants near him and choose the best one. The admin can also add new restaurants and new locations in the map. This option is only available to the admin. So, this application allows users to find their restaurant easily and also get the best route from their location to the restautant chosen.

The code will ask the user to login and enter his present location. When the user types his location, he will be displayed with two options, i.e, whether he wants to search a particular restaurant or he wants to check the restaurants near him. When the user chooses the option to search a particular restaurant, he is asked to enter the restaurant name. The details of the restaurant are displayed along with its ratings and reviews.The shortest path ,i.e,from user location to the restaurant chosen, is also shown. He can also add his own rating and review. When the user chooses the option of finding restaurants near him, he will be displayed the restaurants near his entered location. Then he can check any of those restaurants.

We have implemented the map of the Hyderabad city with the help of hash maps. The hash map contains the key-value pairs where key values are the names of the places and the value is the adjacency list which contains all the adjacent locations along with their edge lengths. The admin can also add new locations to the map. The map is serialized everytime a new location is added and deserialized at the start of the program. An algorithm is written to find the shortest path between two locations which is used to find the shortest distance between the user location and the restaurant chosen.

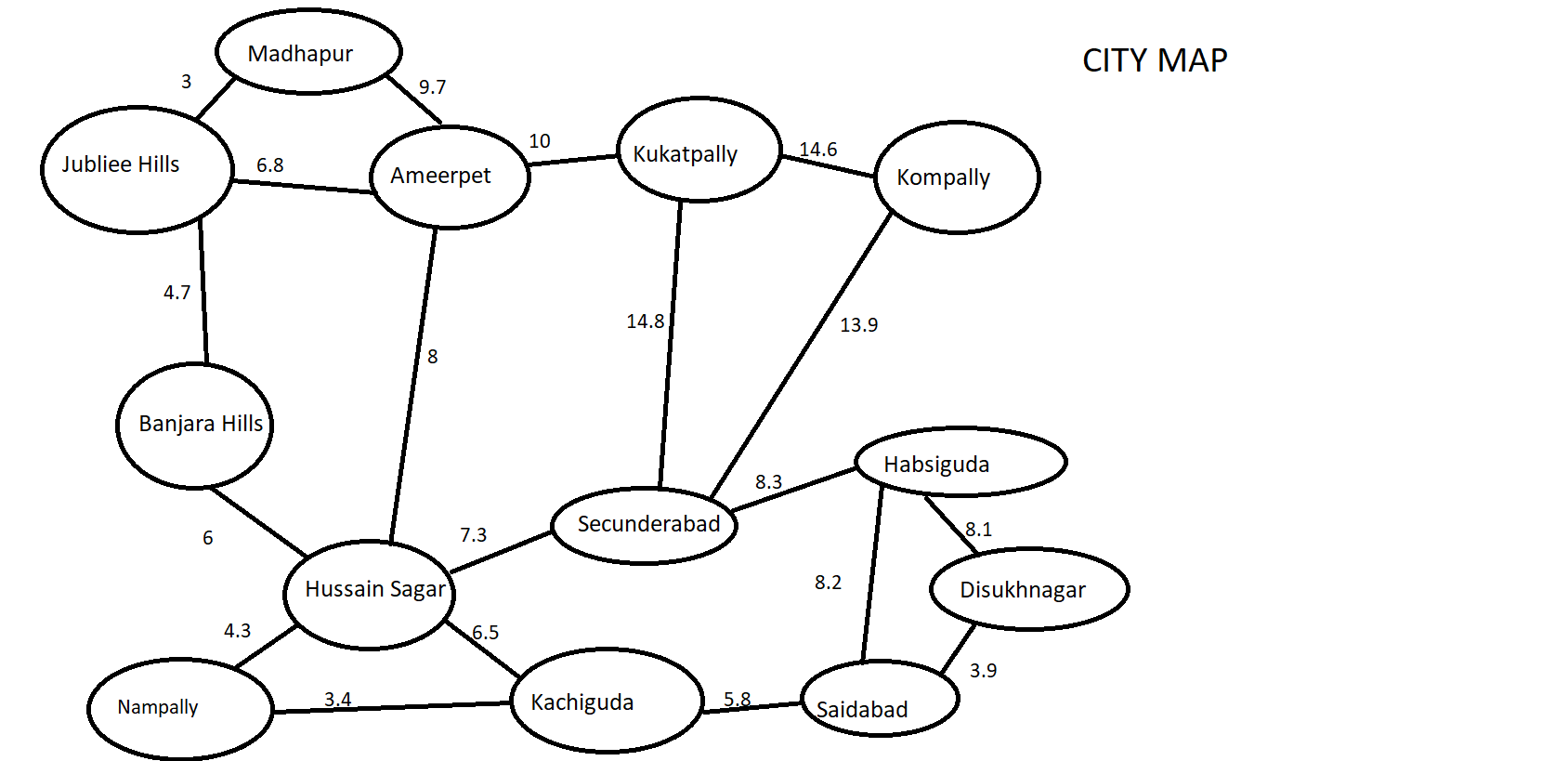
**3. Design**

**4.1 Strategy:**

The code will ask the user to login and enter his present location. When the user types his location, he will be displayed with two options, i.e, whether he wants to search a particular restaurant or he wants to check the restaurants near him. When the user chooses the option to search a particular restaurant, he is asked to enter the restaurant name. The details of the restaurant are displayed along with its ratings and reviews.The shortest path ,i.e,from user location to the restaurant chosen, is also shown. He can also add his own rating and review. When the user chooses the option of finding restaurants near him, he will be displayed the restaurants near his entered location. Then he can check any of those restaurants.

We have implemented the map of the Hyderabad city with the help of hash maps. The hash map contains the key-value pairs where key values are the names of the places and the value is the adjacency list which contains all the adjacent locations along with their edge lengths. The admin can also add new locations to the map. The map is serialized everytime a new location is added and deserialized at the start of the program. An algorithm is written to find the shortest path between two locations which is used to find the shortest distance between the user location and the restaurant chosen.

**4.2 Our Map:**

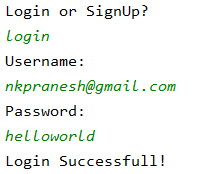
****

**4. Implementation**

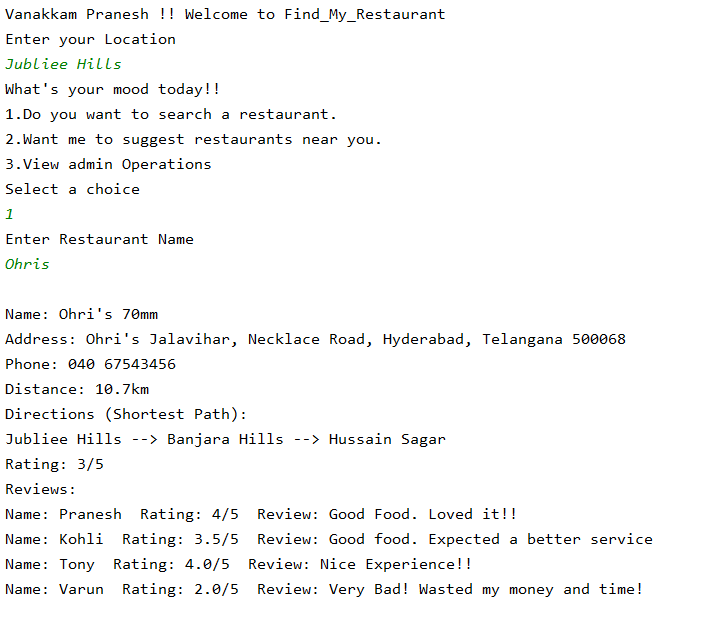
**package** com.company;  
  
**import** java.io.\*;  
**import** java.util.HashMap;  
**import** java.util.Map;  
**import** java.util.Scanner;  
**import** java.util.Stack;  
  
**import static** java.lang.Math.*round*;  
  
**class** sortHotelByDistances  
{  
 String[] **names**;  
 **double**[] **distances**;  
}  
**class** returnMinPath  
{  
 String[] **path**;  
 **double distance**;  
}  
**class** AdjLocation **implements** Serializable  
{  
 **static final long *serialVersionUID***=234;  
 String **name**;  
 **double distance**;  
 AdjLocation(String n,**double** d)  
 {  
 **name**=n;  
 **distance**=d;  
 }  
}  
**class** CityMap **implements** Serializable  
{  
 **static final long *serialVersionUID***=123;  
 Map<String,AdjLocation[]> **cityMap**;  
 CityMap()  
 {  
 **cityMap**=**new** HashMap<String, AdjLocation[]>();  
 }  
  
}  
**class** Functions  
{  
 **public** sortHotelByDistances sortHotelByDistances(String[] names,**double**[] distances,**int** n)  
 {  
 **int** i,j;Double temp1;String temp2;  
 **for**(i=0;i<n-1;i++)  
 {  
 **for**(j=0;j<n-i-1;j++)  
 {  
 **if**(distances[j]>distances[j+1])  
 {  
 temp1=distances[j];  
 distances[j]=distances[j+1];  
 distances[j+1]=temp1;  
  
 temp2=names[j];  
 names[j]=names[j+1];  
 names[j+1]=temp2;  
 }  
 }  
 }  
 sortHotelByDistances returnValue=**new** sortHotelByDistances();  
 returnValue.**names**=names;  
 returnValue.**distances**=distances;  
 **return** returnValue;  
 }  
 **public void** displayRestaurantsNearUser(CityMap cityMap,String userLocation,String userName) **throws** IOException  
 {  
 String[] names=**new** String[10];**int** count=0;  
 **double**[] distances=**new double**[10];  
 File f=**new** File(**"C:\\Users\\K\\Desktop\\OOP MiniProject\\Hotels"**);  
 File res=**null**;  
 System.***out***.println(**"Restaurants near you!!"**);  
 **for**(File a:f.listFiles()) {  
 FileReader fr=**new** FileReader(a);  
 BufferedReader br=**new** BufferedReader(fr);  
 String hotelName=br.readLine();  
 String s=br.readLine();  
 **if**(!s.split(**","**)[0].equals(userLocation))  
 **continue**;  
 **else** {  
 names[count]=hotelName;  
 distances[count]=Double.*parseDouble*(s.split(**","**)[1]);  
 count++;  
 }  
 }  
 sortHotelByDistances displayHotels=sortHotelByDistances(names,distances,count);  
 **for**(**int** m=0;m<count;m++)  
 System.***out***.println(displayHotels.**names**[m]+**" "**+displayHotels.**distances**[m]+**"km"**);  
 System.***out***.println(**"Want to check any of these restaurants(Y/N)?"**);  
 BufferedReader br=**new** BufferedReader(**new** InputStreamReader(System.***in***));  
 String choice=br.readLine();  
 **if**(choice.toLowerCase().equals(**"y"**))  
 searchRestaurant(cityMap,userLocation,userName);  
 }  
 **public void** searchRestaurant(CityMap cityMap,String from,String userNamee)**throws** IOException  
 {  
 BufferedReader br=**new** BufferedReader(**new** InputStreamReader(System.***in***));  
 System.***out***.println(**"Enter Restaurant Name"**);  
 String name=br.readLine();  
 File f=**new** File(**"C:\\Users\\K\\Desktop\\OOP MiniProject\\Hotels"**);  
 File res=**null**;  
 **for**(File a:f.listFiles())  
 {  
 **if**(a.getName().equals(name.toLowerCase()+**".txt"**))  
 res=a;  
 }  
 **if**(res==**null**)  
 {  
 System.***out***.println(**"Restaurant not found!!"**);  
 **return**;  
 }  
 System.***out***.println();  
 FileReader fr=**new** FileReader(res);  
 BufferedReader b=**new** BufferedReader(fr);  
 String s;  
 s=b.readLine();  
 System.***out***.println(**"Name: "**+s);  
 s=b.readLine();  
 String[] loc=s.split(**","**);  
 s=b.readLine();  
 System.***out***.println(**"Address: "**+s);  
 s=b.readLine();  
 System.***out***.println(**"Phone: "**+s);  
 s=b.readLine();  
 returnMinPath path=shortestPath(cityMap,from,loc[0]);  
 **double** d=Math.*round*(path.**distance**\*100)/100d;  
 d=d+(Math.*round*(Double.*parseDouble*(loc[1])\*100)/100);  
 System.***out***.println(**"Distance: "**+d+**"km"**);  
 System.***out***.println(**"Directions (Shortest Path): "**);  
 **for** (**int** j=0;j<path.**path**.**length**;j++)  
 {  
 System.***out***.print(path.**path**[j]);  
 **if**(j!=path.**path**.**length**-1)  
 System.***out***.print(**" --> "**);  
 **else** System.***out***.println();  
 }  
 **if**(s!=**null**) {  
 **double** rating = 0;  
 **int** count = 0;  
 *//System.out.println("Reviews:");* StringBuilder sb = **new** StringBuilder();  
 **do** {  
 String n = s.split(**" "**)[0];  
 String r = s.split(**" "**)[1];  
 rating += Math.*round*(Double.*parseDouble*(r)\*100)/100d;  
 count++;  
 StringBuilder re =**new** StringBuilder();  
 **for**(**int** k=2;k<s.split(**" "**).**length**;k++)  
 re.append(**" "**+s.split(**" "**)[k]);  
 sb.append(**"Name: "** + n + **" Rating: "** + r + **"/5 Review:"** + re.toString() + **"\n"**);  
 s = b.readLine();  
 } **while** (s != **null**);  
 System.***out***.println(**"Rating: "** + Math.*round*(rating\*100 / count)/100+**"/5"**);  
 System.***out***.println(**"Reviews:"**);  
 System.***out***.println(sb.toString());  
 }  
 System.***out***.println(**"Want to rate this restaurant?(Y/N)"**);  
 String userRate=br.readLine();  
 String userR = **null**;  
 String userRe=**null**;  
 **if**(userRate.toLowerCase().equals(**"y"**))  
 {  
 System.***out***.println(**"How much do you rate this restaurant on a scale of 5?"**);  
 userR=br.readLine();  
 System.***out***.println(**"Please give your review.."**);  
 userRe=br.readLine();  
 FileWriter fw=**new** FileWriter(res,**true**);  
 BufferedWriter bw=**new** BufferedWriter(fw);  
 **if**(userNamee==**""**)  
 userNamee=**"Smith"**;  
 bw.write(**"\n"**+userNamee+**" "**+userR+**" "**+userRe);  
 bw.flush();  
 System.***out***.println(**"Review added succesfully!"**);  
 bw.close();  
 fw.close();  
 }  
 b.close();  
 fr.close();  
 }  
 **public void** addRestaurant() **throws** IOException  
 {  
 BufferedReader br=**new** BufferedReader(**new** InputStreamReader(System.***in***));  
 System.***out***.println(**"Enter Restaurant Name:"**);  
 String name=br.readLine();  
 System.***out***.println(**"Enter Restaurant Area:"**);  
 String area=br.readLine();  
 System.***out***.println(**"Enter distance within area:"**);  
 String dis=br.readLine();  
 System.***out***.println(**"Enter Address:"**);  
 String add=br.readLine();  
 System.***out***.println(**"Enter Phone"**);  
 String phone=br.readLine();  
 FileWriter fw=**new** FileWriter(**"C:\\Users\\K\\Desktop\\OOP MiniProject\\Hotels\\"**+name.toLowerCase()+**".txt"**);  
 BufferedWriter bw=**new** BufferedWriter(fw);  
 bw.write(name+**"\n"**);  
 bw.write(area+**","**+ dis+**"\n"**);  
 bw.write(add+**"\n"**);  
 bw.write(phone+**"\n"**);  
 bw.flush();  
 bw.close();  
 fw.close();  
 System.***out***.println(**"Restaurant added succesfully!!"**);  
 }  
 **public boolean** searchAreaName(String key,String[] path,**int** pathcount)  
 {  
 *//System.out.println(key);* **for**(**int** i=0;i<pathcount;i++)  
 {  
 **if**(key.equals(path[i]))  
 {  
 **return true**;  
 }  
 }  
 **return false**;  
 }  
 **public** returnMinPath shortestPath(CityMap cityMap,String from,String to)  
 {  
 **if**(from.equals(to))  
 {  
 returnMinPath a=**new** returnMinPath();  
 String[] path={from};  
 a.**path**=path;  
 a.**distance**=0;  
 **return** a;  
 }  
 **int** keys=cityMap.**cityMap**.keySet().size();  
 String[] names = **new** String[keys];  
 **int** index,i;  
 cityMap.**cityMap**.keySet().toArray(names);  
 **for**(i=0;i<keys;i++)  
 {  
 **if**(names[i].equals(from))  
 {  
 index=i;  
 **break**;  
 }  
 }  
 Stack<Integer> stack=**new** Stack<Integer>();  
 Stack<String> topStackName=**new** Stack<String>();  
 String[] path=**new** String[keys];  
 String[] minPath=**new** String[keys+1];  
 **int** pathCount=0;  
 **double** d=0;  
 **double** min=999999;  
 stack.push(cityMap.**cityMap**.get(names[i]).**length**);  
 topStackName.push(names[i]);  
 path[pathCount++]=names[i];  
 **while** (stack.size()!=0)  
 {  
 **if**(stack.size()==1 && stack.peek()==0)  
 **break**;  
 **if**(!searchAreaName(cityMap.**cityMap**.get(topStackName.peek())[cityMap.**cityMap**.get(topStackName.peek()).**length**-stack.peek()].**name**,path,pathCount))  
 {  
 path[pathCount++]=cityMap.**cityMap**.get(topStackName.peek())[cityMap.**cityMap**.get(topStackName.peek()).**length**-stack.peek()].**name**;  
 d+=cityMap.**cityMap**.get(topStackName.peek())[cityMap.**cityMap**.get(topStackName.peek()).**length**-stack.peek()].**distance**;  
 stack.push(stack.pop()-1);  
 **if**(to.equals(cityMap.**cityMap**.get(topStackName.peek())[cityMap.**cityMap**.get(topStackName.peek()).**length**-stack.peek()-1].**name**))  
 {  
 *//System.out.println("im there");* **if**(d<min)  
 {  
 min=d;  
 minPath=**new** String[pathCount];  
 **for**(**int** k=0;k<pathCount;k++)  
 minPath[k]=path[k];  
 }  
 d=d-cityMap.**cityMap**.get(topStackName.peek())[cityMap.**cityMap**.get(topStackName.peek()).**length**-stack.peek()-1].**distance**;  
 pathCount--;  
 }  
 **else** {  
 String temp=topStackName.peek();  
 topStackName.push(cityMap.**cityMap**.get(topStackName.peek())[cityMap.**cityMap**.get(topStackName.peek()).**length** - stack.peek() - 1].**name**);  
 stack.push(cityMap.**cityMap**.get(cityMap.**cityMap**.get(temp)[cityMap.**cityMap**.get(temp).**length** - stack.peek() - 1].**name**).**length**);  
 }  
 }  
 **else** stack.push(stack.pop()-1);  
 **while**(stack.peek()==0)  
 {  
 pathCount--;  
 topStackName.pop();  
 stack.pop();  
 **if**(stack.size()!=0)  
 d=d-cityMap.**cityMap**.get(topStackName.peek())[cityMap.**cityMap**.get(topStackName.peek()).**length**-stack.peek()-1].**distance**;  
 **else  
 break**;  
 }  
 }  
 returnMinPath returnValue=**new** returnMinPath();  
 returnValue.**path**=minPath;  
 returnValue.**distance**=min;  
 **return** returnValue;  
 }  
 **public** CityMap serialize() **throws** IOException, ClassNotFoundException {  
 FileInputStream fis=**new** FileInputStream(**"C:\\Users\\K\\Desktop\\OOP MiniProject\\Map.txt"**);  
 **if**(fis.available()==0)  
 **return new** CityMap();  
 ObjectInputStream ois=**new** ObjectInputStream(fis);  
 CityMap c=(CityMap)ois.readObject();  
 fis.close();  
 ois.close();  
 **return** c;  
 }  
 **public void** deserialize(CityMap cityMap) **throws** IOException  
 {  
 FileOutputStream fos=**new** FileOutputStream(**"C:\\Users\\K\\Desktop\\OOP MiniProject\\Map.txt"**);  
 ObjectOutputStream oos=**new** ObjectOutputStream(fos);  
 oos.writeObject(cityMap);  
 oos.close();  
 fos.close();  
 }  
 **public** CityMap addArea(CityMap cityMap) **throws** IOException  
 {  
 Scanner sc=**new** Scanner(System.***in***);  
 BufferedReader br=**new** BufferedReader(**new** InputStreamReader(System.***in***));  
 System.***out***.println(**"Enter Area name:"**);  
 String area=br.readLine();  
 System.***out***.println(**"Enter number of adjacent locations"**);  
 **int** n=Integer.*parseInt*(br.readLine());  
 AdjLocation[] adj=**new** AdjLocation[n];  
 **for**(**int** i=0;i<n;i++)  
 {  
 System.***out***.println(**"Enter location "**+(i+1)+**" name:"**);  
 String name=br.readLine();  
 AdjLocation[] temp=**new** AdjLocation[cityMap.**cityMap**.get(name).**length**+1];  
 **for** (**int** k=0;k<cityMap.**cityMap**.get(name).**length**;k++)  
 temp[k]=cityMap.**cityMap**.get(name)[k];  
 System.***out***.println(**"Enter distance"**);  
 **double** distance=Double.*parseDouble*(br.readLine());  
 temp[cityMap.**cityMap**.get(name).**length**]=**new** AdjLocation(area,distance);  
 cityMap.**cityMap**.put(name,temp);  
 adj[i]=**new** AdjLocation(name,distance);  
 }  
 cityMap.**cityMap**.put(area,adj);  
 deserialize(cityMap);  
 System.***out***.println(**"New area added succesfully!!"**);  
 **return** cityMap;  
 }  
 **public** CityMap askUser(CityMap cityMap) **throws** IOException  
 {  
 BufferedReader br=**new** BufferedReader(**new** InputStreamReader(System.***in***));  
 System.***out***.println(**"1. Add new Area to the map"**);  
 System.***out***.println(**"2. Add new restaurant"**);  
 System.***out***.println(**"3. Show Map"**);  
 System.***out***.println(**"Select a choice..."**);  
 **int** ch=Integer.*parseInt*(br.readLine());  
 **switch** (ch)  
 {  
 **case** 1: {  
 cityMap=addArea(cityMap);  
 *//cityMap=askUser(cityMap);* **break**;  
 }  
 **case** 2: {  
 addRestaurant();  
 *//cityMap=askUser(cityMap);* **break**;  
 }  
 **case** 3:{  
 **for**(String a:cityMap.**cityMap**.keySet())  
 {  
 System.***out***.print(a+**"-> "**);  
 **for** (AdjLocation b:cityMap.**cityMap**.get(a))  
 {  
 System.***out***.print(b.**name**+**"-"**+b.**distance**+**" "**);  
 }  
 System.***out***.println();  
 }  
 }  
 **default**: {  
 System.***out***.println(**"Invalid Choice!"**);  
 }  
 }  
 **return** cityMap;  
 }  
}  
**public class** Main  
{  
  
 **public static void** main(String[] args) **throws** IOException, ClassNotFoundException {  
 File accounts=**new** File(**"C:\\Users\\K\\Desktop\\OOP MiniProject\\Accounts.txt"**);  
 String name=**""**;  
 System.***out***.println(**"Login or SignUp?"**);  
 BufferedReader br =**new** BufferedReader(**new** InputStreamReader(System.***in***));  
 String input1=br.readLine();  
 **if**(input1.equalsIgnoreCase(**"SignUp"**))  
 {  
 FileWriter signup=**new** FileWriter(accounts,**true**);  
 FileReader login=**new** FileReader(accounts);  
 BufferedWriter bf=**new** BufferedWriter(signup);  
 BufferedReader bff=**new** BufferedReader(login);  
 StringBuilder signin=**new** StringBuilder();  
 String username;  
 System.***out***.println(**"Username:"**);  
 username = br.readLine();  
 String s;  
 **while** ((s = bff.readLine()) != **null**)  
 {  
 **if** (s.split(**" "**)[0].equals(username))  
 {  
 System.***out***.println(**"Username already exists!"**);  
 System.***out***.println(**"Try Again!\n"**);  
 *main*(**null**);  
 **break**;  
 }  
 }  
 signin.append(username);  
 signin.append(**" "**);  
 System.***out***.println(**"Password:"**);  
 signin.append(br.readLine());  
 signin.append(**" "**);  
 System.***out***.println(**"Your Name:"**);  
 name=br.readLine();  
 signin.append(name);  
 signin.append(**"\n"**);  
 bf.write(signin.toString());  
 System.***out***.println(**"Sigup Succesfull!"**);  
 bf.flush();  
 signup.close();  
 login.close();  
 bf.close();  
 bff.close();  
 }  
 **else if**(input1.equalsIgnoreCase(**"login"**))  
 {  
 FileReader login=**new** FileReader(accounts);  
 BufferedReader bff=**new** BufferedReader(login);  
 System.***out***.println(**"Username:"**);  
 String username=br.readLine();  
 System.***out***.println(**"Password:"**);  
 String password=br.readLine();  
 **int** flag=0,flag1=0;  
 String s;  
 **do** {  
 s=bff.readLine();  
 **if**(s==**null**)  
 **break**;  
 **if**(s.split(**" "**)[0].equals(username))  
 {  
 **if**(s.split(**" "**)[1].equals(password))  
 {  
 System.***out***.println(**"Login Successfull!"**);  
 name=s.split(**" "**)[2];  
 flag1=1;  
 **break**;  
 }  
 }  
 }**while** (s!=**null**);  
 **if**(flag1==0)  
 {  
 System.***out***.println(**"Invalid Username or Password! Try Again!"**);  
 *main*(**null**);  
 }  
 }  
 **else**{  
 System.***out***.println(**"Invalid Option!! Try Again"**);  
 *main*(**null**);  
 }  
 Functions fn=**new** Functions();  
 CityMap cityMap=fn.serialize();  
*// if(name.equals("Pranesh")||name.equals("Varun"))  
// cityMap=fn.askUser(cityMap);* System.***out***.println(**"Vanakkam "**+name+**" !! Welcome to Find\_My\_Restaurant"**);  
 **int** flag=1;  
 String userLocation=**null**;  
 System.***out***.println(**"Enter your Location"**);  
 userLocation = br.readLine();  
 String[] names=**new** String[cityMap.**cityMap**.keySet().size()];  
 cityMap.**cityMap**.keySet().toArray(names);  
 **for** (String a:names)  
 {  
 **if**(a.equals(userLocation))  
 {  
 flag=0;  
 **break**;  
 }  
 }  
 **if**(flag==1)  
 {  
 System.***out***.println(**"Sorry that location is not in our service!!"**);  
 System.*exit*(0);  
 }  
 String cont=**null**;  
 **do**{  
 System.***out***.println(**"What's your mood today!!"**);  
 System.***out***.println(**"1.Do you want to search a restaurant."**);  
 System.***out***.println(**"2.Want me to suggest restaurants near you."**);  
 **if**(name.equals(**"Pranesh"**)||name.equals(**"Varun"**))  
 System.***out***.println(**"3.View admin Operations"**);  
 System.***out***.println(**"Select a choice"**);  
 **int** ch=Integer.*parseInt*(br.readLine());  
 **switch** (ch) {  
 **case** 1: {  
 fn.searchRestaurant(cityMap, userLocation, name);  
 **break**;  
 }  
 **case** 2: {  
 fn.displayRestaurantsNearUser(cityMap,userLocation,name);  
 **break**;  
 }  
 **case** 3: {  
 cityMap = fn.askUser(cityMap);  
 **break**;  
 }  
 }  
 System.***out***.println(**"Do you want to continue?(Y/N)"**);  
 cont=br.readLine();  
 }**while** (cont.toLowerCase().equals(**"y"**));  
 System.*exit*(0);  
 }  
}

**5.OUTPUTS**

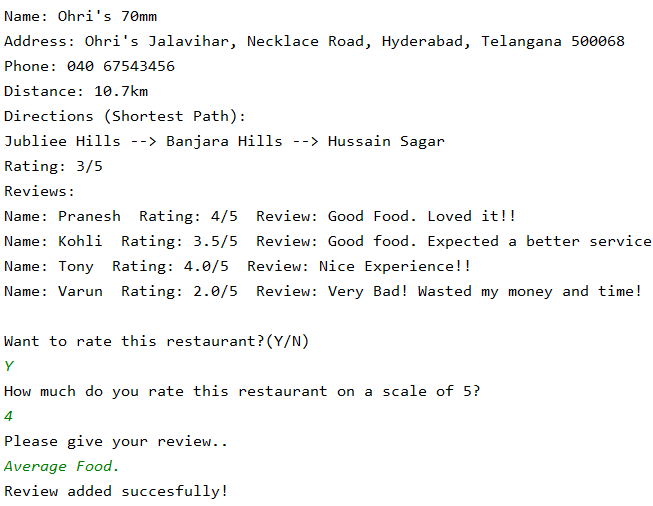
Output 1: User Login.



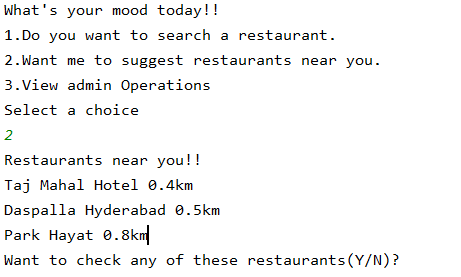
Output 2: User checks a restaurant.



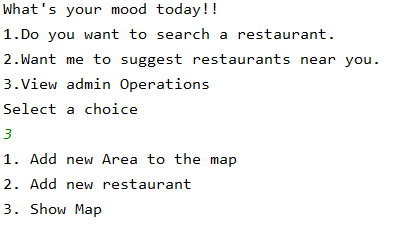
Output 3: User adds his review.



Output 4: User checks restaurants near him.



Output 5: Admin Operations.



**6.Conclusion**

To conclude, people can use this application to check restaurants details and also their reviews and ratings. The user can also get the shortest path from his location to the restaurant he has chosen.

**7.References** :

[1]The Complete Reference Java 7th Edition: Herbert Schildt

[2] <https://www.geeksforgeeks.org/serialization-in-java/>

[3] https://courses.cs.washington.edu/courses/cse373/11wi/lectures/02-25/programs/Graph.java