# **INFORMATICS PRACTICES PROJECT**

**2021-22**



**Bal Bhawan School School**

Shamla Hills,Bhopal

# Topic :

# COMMERCE WEBSITE

Database management system using Python-MySQL connector

### Guided by:

**Mr. Abhishek Shrivasta &**

**Mrs. Sudha Nair**

Project made by:

**Mohammad Maasir (**Class XII Science)

Certificate

this is to certify that **MOHAMMAD MAASIR** of class **12th SCIENCE** of **BAL BHAWAN SCHOOL**, Shamla Hills, Bhopal has completed his informatics practices project for the session 2021-22 on the topic **“E-COMMERCE WEBSITE DATABASE MANAGEMENT SYSTEM”** using Python – MySQL RDBMS technologies.

I CERTIFY THAT THE PROJECT IS IN ACCORDANCEWITH THE GUIDELINES ISSUED BY

**CENTRAL BOARD OF SECONDARY EDUCATION**,INDIA.

**Principal**

**Internal**

**External**

# /

# Message

#### It is a sign of creative activity and cooperation, that our students, guided and inspired by the teachers are preparing themselves not only for a bright future but also to become a successful person of character for the nation.

I am delighted to know that Mohammad Maasir of class XII–Science has completed his project on the topic “E-Commerce Website database management system” within the stipulated time period.

**Mrs.Humaira Arif**

(Principal, Bal Bhawan School, Bhoplal)

# Acknowledgement

I owe a great many thanks to a great many people who helped and supported me during the making of this project. My deepest thanks to my teachers, **Mr. Abhishek Shrivastava & Mrs. Sudha Nair**, the guides of the project for guiding and correcting various mistakes with attention and care. They have taken pain to go through the project and make necessary correction as and when needed. I express my thanks to the respected principal **Mrs. Humaira Arif** of Bal Bhavan School, for extending her support. Thanks, and appreciation to all the helpful people for their support. I also extend my heartfelt thanks to my family & well-wishers for their effective encouragement, guidance, and patience throughout the project.

Contents

1.Introduction to Python

2.MySQL Database System

3.Classification of SQL Statements

4.Front End & Back End

5.Minimum Hardware & Software Requirements

6.User working Analysis

7.ER Model Arrow Diagram

8.Manually Highlighted Front End Coding for Python-MySQL connector

9.Front end coding screenshots

9.Entering data to the server using the front end program made.(screenshots)

10.Output of entered data in MySQL.

11.Code for Data visualization(graphs) or Data Analysis using Matplotlib.pyplot

12.Data analysis output (screenshots of graphs)

13.(Microsoft Visual Studio Code +GitHub IDE) highlighted code for front end Python-MySQL connector

**Introduction to Python**

What is Python?

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

* web development (server-side),
* software development,
* mathematics,
* system scripting. What can Python do?
* Python can be used on a server to create web applications.
* Python can be used alongside software to create workflows.
* Python can connect to database systems. It can also read and modify files.
* Python can be used to handle big data and perform complex mathematics.
* Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

* Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.).
* Python has a simple syntax similar to the English language.
* Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
* Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
* Python can be treated in a procedural way, an object-oriented way or a functional way.

Good to know

* The most recent major version of Python is Python 3, which we shall be using in this project. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
* In this project Python will be written in the default python IDLE(Integrated Development and Learning Environment). It is possible to write Python in an Integrated Development Environment, such as Virtual Studio Code, Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

* Python was designed for readability, and has some similarities to the English language with influence from mathematics.
* Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
* Python relies on indentation, using white space, to define scope; such as the scope of loops, functions and classes. Other

programming languages often use curly-brackets for this purpose.

**MySQL**

MySQL is freely available open source Relational Database management System (RDBMS) that uses Structured Query Language (SQL). It is downloadable from site [www.mysql.org](http://www.mysql.org/) In a MySQL database, information is stored in tables. A single MySQL database can contain many tables at once and store thousands of individual records. MySQL provides you with a rich set of features that that support a secure environment for storing, maintaining, and accessing data. MySQL is a fast, reliable, scalable alternative to many of the commercial RDBMS available.

**MySQL Database System**

My SQL database system refers to the combination of a MySQL server instance and a MySQL database. MySQL operates using client/server architecture in which the server runs on the machine containing the database and clients connect to the server over a network. MySQL is a multi-user database system, meaning several users can access the database simultaneously. The sever (MySQL server) listens for client requests coming in over the network and accesses database contents according to those requests and provides that to the clients. Clients are programs that connect to the database server and issue queries in a pre-specified format. MySQL is compatible with the standards-based SQL (Structured Query Language). The client program may contact the server programmatically (meaning a program call the server during execution) or manually.

**Classification of SQL Statements**

SQL provides many different types of commands used for different purposes. SQL commands can be divided into following categories :-

*(please turn over)*

**(i)Data Definition Language (DDL)Commands:**

Commands that allows you to perform tasks related to data definition. Example-

* Creating, altering and dropping
* Granting and revoking privileges and roles
* Maintenance commands
* making changes to database, permanent
* undoing changes to database, permanent
* creating save points
* setting properties for current transactions.

### (ii)Data Manipulation Language (DML) Commands:

Commands that allow you to perform tasks related to data manipulation. Example:

retrieval, insertion, deletion and modification of data stored in database

**(iii)Transaction Control Language (TCL) Commands:** Commands that allow you to manage and control the transaction (a transaction is one complete unit of work involving many steps) for example:

* MySQL was created and is supported by MySQL LAB, a company based in Sweden. This company is now a subsidiary of Sun Microsystems, which holds the copyright to most of the codebase. On April 20TH, 2009 Oracle database, announced a deal to acquire Sun Microsystems.

***Brief History About MySQL***

* The chief inventor of MySQL was Michael Widenius (a.k.a. Monty). MySQL has been named after Monty’s daughter My.
* The logo of MySQL, the dolphin, is named as “Sakila”.

***Front-End and Back-End***

### **FRONT END**

The project uses python as its front end, that is it is the part where user interact. The front end basically includes input facilities through which the user interacts. Apart from the user interaction, the front end also contains database objects that form a layer between the user and BACK END.

### BACK END

MySQL is used as a back end. It handles all database accesses through one or more servers. A Server is a special computer that is responsible for processing requests send to it. It processes all the data and query passed to it and sends result back to the front end where it is displayed to the user

**Minimum Hardware Specification** of the machine to be used :-

1. P-1 Processor
2. 32MB RAM
3. 1.4inch color monitor
4. 2.1GB HDD
5. 233 MHZ CPU Clock Speed

**Minimum Software Specification** of the machine to be used :-

(tools & platforms used)

Python 3.6 and above(I am using python3.10)

MySQL version 8.0 is used in this project.

This project uses GUI provided by Python and MySQL which is a RDBMS.

(additionally I have also used Microsoft Visual Studio Code as code editor.)

### *Hardware And Software Requirements*

*ENTITY RELATIONSHIP MODEL FOR MY E- COMMERCE WEBSITE*

**Description About My E-Commerce Website :**

This is a website with Registered buyers, and sellers use to do their respective tasks. Furthermore, this website also keeps records of Registered Delivery Mans. All the three category of people mentioned here have to make an account on the website, with distinct user-id, and create a password for their account. Furthermore, they have to submit their **mobile number, email address and pin code(city)** while making the account.

Any customer can order products from the website and give ratings & reviews about the products or items. They can also give ratings about the items, sellers and deliverymen.

Apart from ratings and reviews, the users are free to complain about any issue related to their orders.

Dates and time of all important events will be recorded.

A user which is found not active for 2 years should be given warning notification (on Email/SMS) that his/her account would be removed from the database if he/she remains inactive for 6 more months.

The name of this imaginary website is “Eshop”

**Entity Relationship Model :**

The ER model will define the database which stores information about:-

1. Customer (consumer) details
2. Seller details
3. Item details
4. Record of all orders
5. Delivery boy details
6. Ratings of: items, seller and delivery boys,which are done by the customers, will be stored in respective tables.no need to create separate table
7. Complaints done by the costumers on a particular order regarding delivery boy or item. Will be stored in a single table.

*(six tables in total)*

|  |  |
| --- | --- |
| ***Abbreviation*** | ***meaning*** |
| C\_ID | Customer ID  (unique identity) |
| O\_ID | Order ID  (unique identity) |
| S\_ID | Seller ID  (unique identity) |
| DELBOY | Delivery Boy |

***Reference for abbreviations***

# NEW ER DIAGRAM

Simplified Entity Relation diagram for the relational database. The tables in the database will be created according this diagram.

Diagram created by me, using draw.io

This diagram is also available on my GitHub Page : github.com/maasir554

Direct Link : [https://github.com/maasir554/maasir554/blob/main/Untitled%20Diagram.drawio](https://github.com/maasir554/maasir554/blob/main/Untitled Diagram.drawio)

***ER DIAGRAM :***

FRONT END CODING

PYTHON(and SQL connector), Manually Highlighted

Please refer to last pages of this document for viewing code highlighted by GitHub+VS Code .

#importing the connector :-

import mysql.connector as c

#defining the database :-

#Note: the name of database is Eshop. you have to create database Eshop #to run this program.

database = c.connect(host = "localhost", user = 'root', password = "password", database = 'eshop')

#defining cursor(it will be used to excute SQL commands) :-

cursor = database.cursor(buffered=True)

#making the tables in SQL database using python commands

print('You are connected to the server eshop successfully.')

cursor.execute('CREATE TABLE IF NOT EXISTS items(item\_id INT(4) PRIMARY KEY NOT NULL,item\_name VARCHAR(50),category VARCHAR(30),seller\_id INT(4) NOT NULL,price INT(8) NOT NULL,date\_added DATE,rating FLOAT(2,1))' )

cursor.execute('CREATE TABLE IF NOT EXISTS customers(customer\_id INT(4) PRIMARY KEY NOT NULL,customer\_name VARCHAR(30) NOT NULL,customer\_age INT(3),gender ENUM("male","female","others"),email VARCHAR(30),mobile VARCHAR(12),city VARCHAR(20))')

cursor.execute('CREATE TABLE IF NOT EXISTS delboy(delboy\_id INT(4) PRIMARY KEY NOT NULL,name VARCHAR(30) NOT NULL,city VARCHAR(20),mobile VARCHAR(12),salary INT(4),rating float(2,1))')

cursor.execute('CREATE TABLE IF NOT EXISTS sellers(seller\_id INT(4) PRIMARY KEY NOT NULL,name VARCHAR(30) NOT NULL,city VARCHAR(20),mobile VARCHAR(12),email VARCHAR(30),rating float(2,1))')

cursor.execute('CREATE TABLE IF NOT EXISTS orders(order\_id INT(4) PRIMARY KEY NOT NULL,item\_id INT(4) NOT NULL,customer\_id INT(4) NOT NULL,date\_placed DATE,delboy\_ID INT(4) NOT NULL)')

cursor.execute('CREATE TABLE IF NOT EXISTS complains(complain\_id INT(4) PRIMARY KEY NOT NULL,order\_id INT(4) NOT NULL,subject VARCHAR(20) NOT NULL)')

#MAKING THE MENU(USER INTERFACE) FOR FRONT END

def menu():

print("--- menu opened ---")

a = int(input("to add data please type : 1 \nto delete data please type : 2 \nto exit this menu, please type 3 \nResponse : "))

if (a==1):

b= int(input("type number : \n'1' to add customer data \n'2' to add seller data \n'3' to add item data \n'4' to add order data \n'5' to add delboy data \n'6' to add complain. \nResponse : "))

if (b==1):

add\_customer()

elif (b==2):

add\_seller()

elif (b==3):

add\_item()

elif (b==4):

add\_order()

elif (b==5):

add\_delboy()

elif (b==6):

add\_complain()

else:

print('please enter a valid response')

elif (a==2):

b= int(input("type number : \n'1' to delete customer data \n'2' to delete seller data \n'3' to delete item data \n'4' to delete order data \n'5' to delete delboy data \n'6' to delete complain \nResponse : "))

if (b==1):

delete\_customer()

elif (b==2):

delete\_seller()

elif (b==3):

delete\_item()

elif (b==4):

delete\_order()

elif (b==5):

delete\_delboy()

elif (b==6):

delete\_complain()

else:

print('please enter a valid response')

menu()

elif(a==3):

print('Thank You! for using this program, Have a nice day! \n you can now exit or call functions manually. \nlist of the functions(prefix add\_ OR delete\_ before them) : \ncustomer\nseller\nitem\norder\ndelboy\ncomplain\n--- menu closed ---')

else:

print('please enter a valid response')

menu()

#DEFINING FUNCTIONS FOR FEEDING DATA IN TEH TABLES OF THE SERVER

def add\_customer():

print("add\_customer() function started.")

c\_id = input('enter customer ID : ')

name = input('enter customer name : ')

age = input("enter customer age : ")

gender = input("enter gender of customer : ")

email = input('enter customer email ID : ')

mobile = input('enter customer mobile number : ')

city = input('enter customer city : ')

data = (c\_id,name,age,gender,email,mobile,city)

statement = "INSERT INTO customers VALUES(%s,%s,%s,%s,%s,%s,%s)"

cursor.execute(statement,data)

database.commit()

print('data inserted to table customer successfully!')

b = input("do you want ot add more data to THIS table? (y/n) : ")

if (b=='y'):

add\_customer()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. toa call back use menu())")

#DEFINING FUNCTIONS FOR FEEDING DATA IN THE TABLES OF THE SERVER

def add\_customer():

print("add\_customer() function started.")

c\_id = input('enter customer ID : ')

name = input('enter customer name : ')

age = input("enter customer age : ")

gender = input("enter gender of customer : ")

email = input('enter customer email ID : ')

mobile = input('enter customer mobile number : ')

city = input('enter customer city : ')

data = (c\_id,name,age,gender,email,mobile,city)

statement = "INSERT INTO customers VALUES(%s,%s,%s,%s,%s,%s,%s)"

cursor.execute(statement,data)

database.commit()

print('data inserted to table customer successfully!')

b = input("do you want ot add more data to THIS table? (y/n) : ")

if (b=='y'):

add\_customer()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. toa call back use menu())")

def add\_seller():

print("add\_seller() function started.")

s\_id = input('enter seller ID : ')

name = input('enter seller name : ')

city = input('enter seller city : ')

mobile = input('enter seller mobile number : ')

email = input('enter seller email ID : ')

rating = input('enter the rating of seller : ')

data = (s\_id,name,city,mobile,email,rating)

statement = "INSERT INTO sellers VALUES(%s,%s,%s,%s,%s,%s)"

cursor.execute(statement,data)

database.commit()

print('data inserted to table sellers successfully!')

b = input("do you want ot add more data to THIS table? (y/n) : ")

if (b=='y'):

add\_seller()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back to call back use menu())")

def add\_item():

print("add\_item() function started.")

item\_id = input('enter item ID : ')

name = input('enter item name : ')

cat = input('enter category of item : ')

sid = input('enter seller\_id of item seller : ')

price = input('enter price of item : ')

date\_added = input('enter date added : ')

rating = input('enter rating of item : ')

data = (item\_id,name,cat,sid,price,date\_added,rating)

statement = "INSERT INTO items VALUES(%s,%s,%s,%s,%s,%s,%s)"

cursor.execute(statement,data)

database.commit()

print('data inserted to table items successfully!')

b = input("do you want ot add more data to THIS table? (y/n) : ")

if (b=='y'):

add\_item()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back use menu())")

def add\_order():

print("add\_order() function started.")

oid = input('enter order id : ')

iid = input('enter item id : ')

cid = input('enter customer id : ')

dp = input('enter date placed : ')

dbid = input('enter delivery boy id : ')

data = (oid,iid,cid,dp,dbid)

statement = 'insert into orders values(%s,%s,%s,%s,%s)'

cursor.execute(statement,data)

database.commit()

print('data inserted to the table ORDERS successfully!')

b = input("do you want ot add more data to THIS table? (y/n) : ")

if (b=='y'):

add\_order()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back use menu())")

def add\_delboy():

print("add\_delboy() function started.")

did = input('enter delboy ID : ')

name = input('enter delboy name : ')

city = input('enter delboy city : ')

mobile = input('enter delboy mobile number : ')

salary = input('enter delboy salary : ')

rating = input('enter the rating of delboy : ')

data = (did,name,city,mobile,salary,rating)

statement = "INSERT INTO delboy VALUES(%s,%s,%s,%s,%s,%s)"

cursor.execute(statement,data)

database.commit()

print('data inserted to table delboy successfully!')

b = input("do you want ot add more data to THIS table? (y/n) : ")

if (b=='y'):

add\_delboy()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back use menu())")

def add\_complain():

print("add\_complain() function started.")

cid = input("enter complain ID : ")

oid = input("enter order id : ")

sub = input("enter the subject of complain : ")

data = (cid,oid,sub)

statement = "INSERT INTO COMPLAINS VALUES(%s,%s,%s)"

cursor.execute(statement,data)

database.commit()

print("data inserted to the table COMPLAINS successfully!")

b = input("do you want ot add more data to THIS table? (y/n) : ")

if (b=='y'):

add\_complain()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. toa call back use menu())")

#DEFINING FUNCTIONS TO DELETE DATA FROM TABLES OF DATABASE :-

def delete\_customer():

print("delete\_customer() function started.")

cid = input('enter customer\_id : ')

cursor.execute('DELETE FROM customers WHERE customer\_id=%s',(cid,))

database.commit()

print('deleted the customer data of customer\_id : ',cid,'from table customers successfully!')

b = input("do you want ot delete more data to THIS table? (y/n) : ")

if (b=='y'):

delete\_customer()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back use menu())")

def delete\_seller():

print("delete\_seller() function started.")

sid = input('enter seller\_id : ')

cursor.execute('DELETE FROM sellers WHERE seller\_id=%s',(sid,))

database.commit()

print('deleted the seller data of seller\_id : ',sid,' from table sellers successfully!')

b = input("do you want ot delete more data to THIS table? (y/n) : ")

if (b=='y'):

delete\_seller()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back, use menu())")

def delete\_item():

print("delete\_item() function started.")

iid = input('enter item\_id : ')

cursor.execute('DELETE FROM items WHERE item\_id=%s',(iid,))

database.commit()

print('deleted the item data of item\_id : ',iid,' from table items successfully!')

b = input("do you want ot delete more data to THIS table? (y/n) : ")

if (b=='y'):

delete\_item()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back, use menu())")

def delete\_order():

print("delete\_order() function started.")

x = input('enter order\_id : ')

cursor.execute('DELETE FROM orders WHERE order\_id=%s',(x,))

database.commit()

print('deleted the item data of order\_id : ',x,' from table orders successfully!')

b = input("do you want ot delete more data to THIS table? (y/n) : ")

if (b=='y'):

delete\_order()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back, use menu())")

def delete\_delboy():

print("delete\_delboy() function started.")

x = input('enter delboy\_id : ')

cursor.execute('DELETE FROM delboy WHERE delboy\_id=%s',(x,))

database.commit()

print('deleted the data of delboy\_id ',x,' from table delboy successfully!')

b = input("do you want ot delete more data to THIS table? (y/n) : ")

if (b=='y'):

delete\_delboy()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back, use menu())")

def delete\_complain():

print("delete\_complain() function started.")

x = input('enter complain\_id : ')

cursor.execute('DELETE FROM complains WHERE complain\_id=%s',(x,))

database.commit()

print('deleted the data of complain\_id ',x,' from table complains successfully!')

b = input("do you want ot delete more data to THIS table? (y/n) : ")

if (b=='y'):

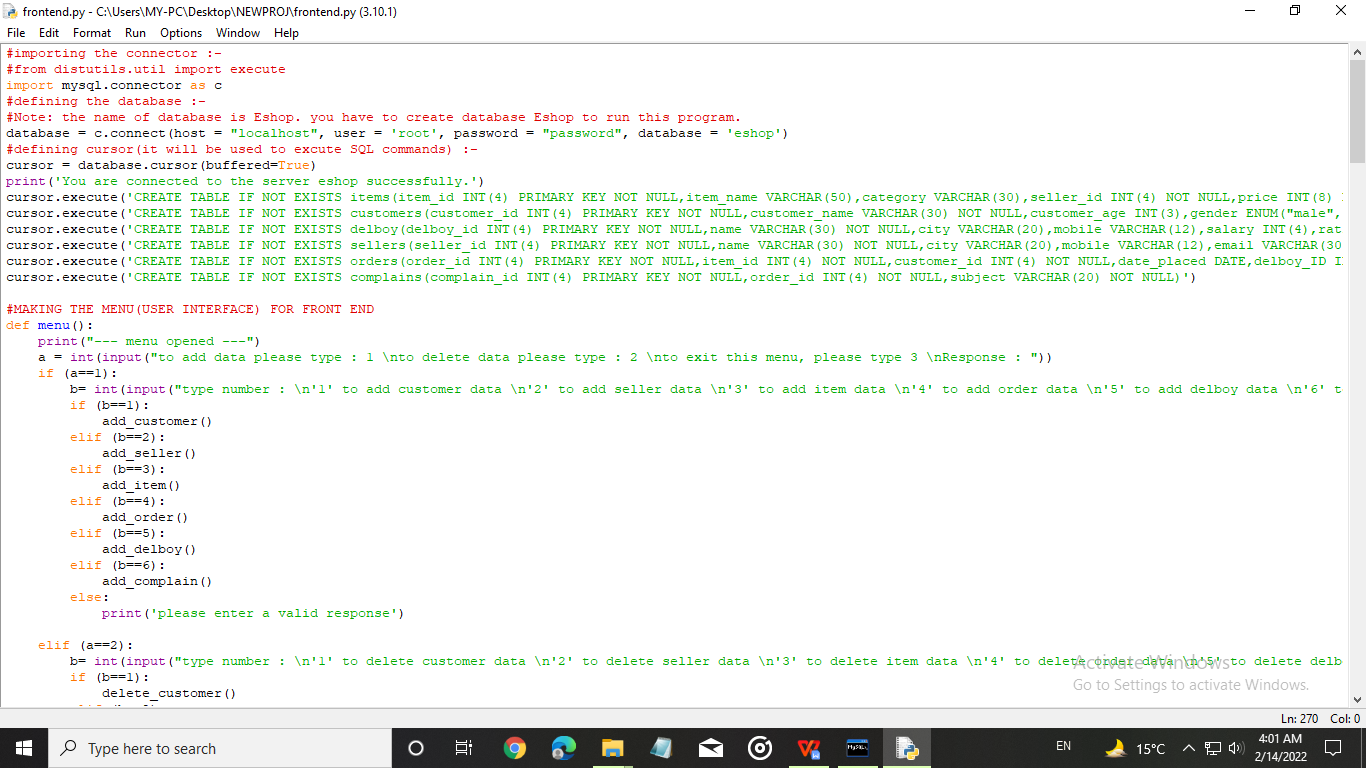
delete\_complain()

elif (b=='n'):

menu()

else:

print("please enter a valid response. (you exited the menu. to call back, use menu())")



ORIGINAL FILE SCREENSHOTS

NOTE

The above boxes of code are NOT the screenshot,These are copied from my file frontend.py and pasted here,To dispay the text clearly.

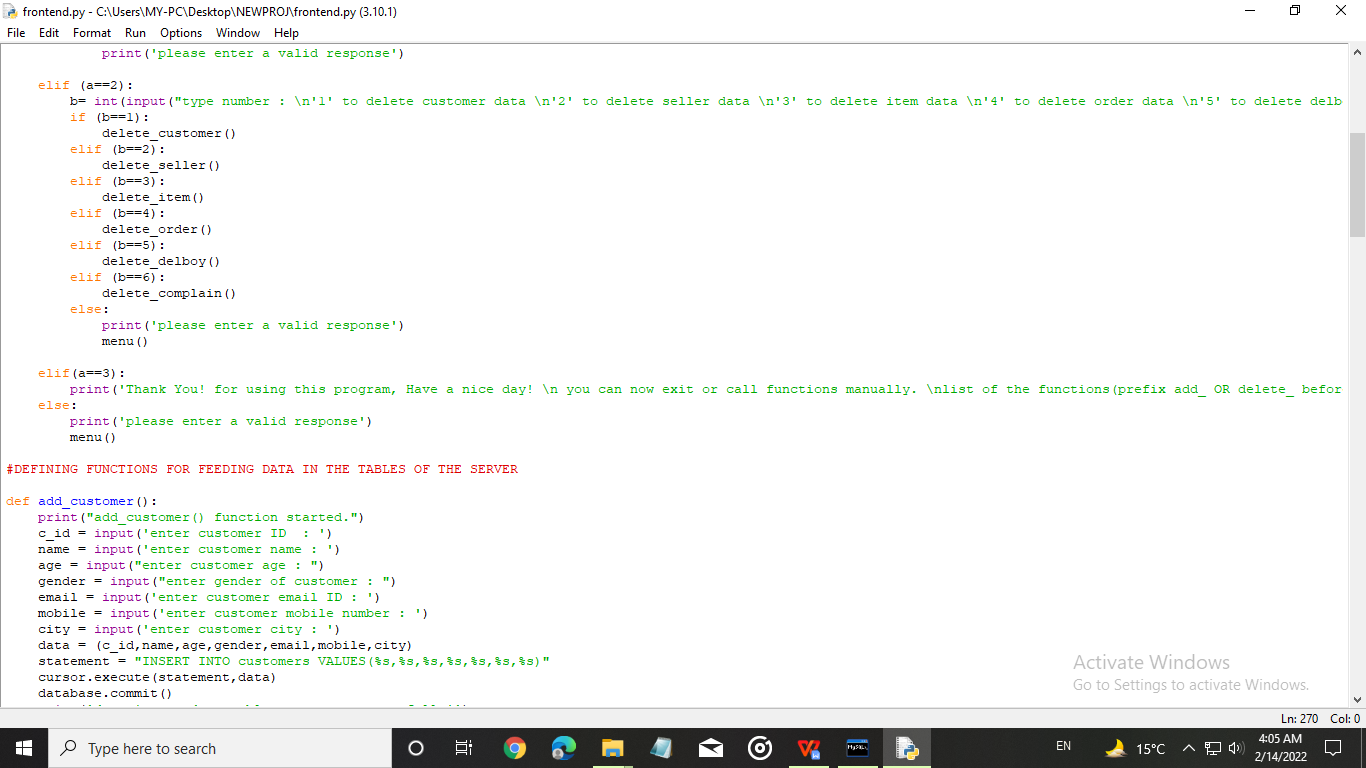
Original file’s screenshots are pasted below for reference.

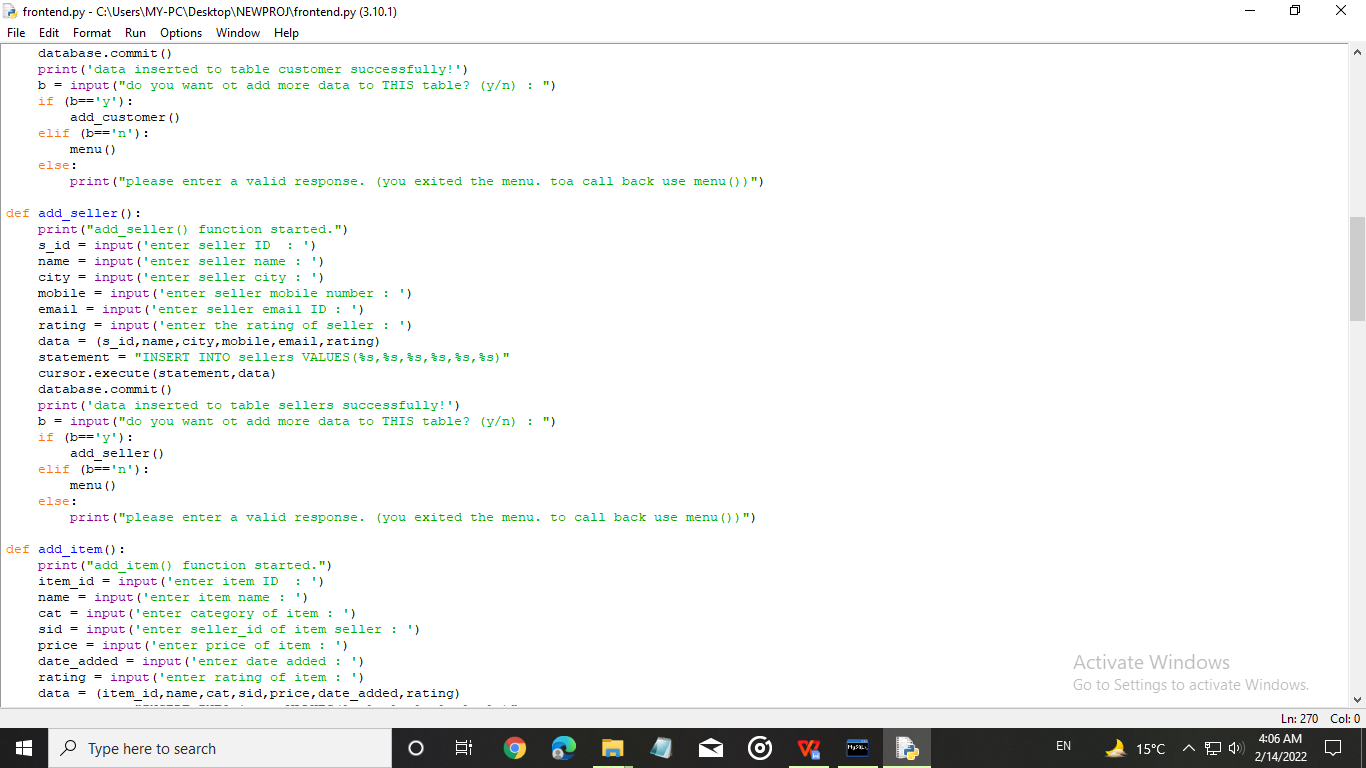
#FINALLY, SERVING THE PROGRAM TO USER : -

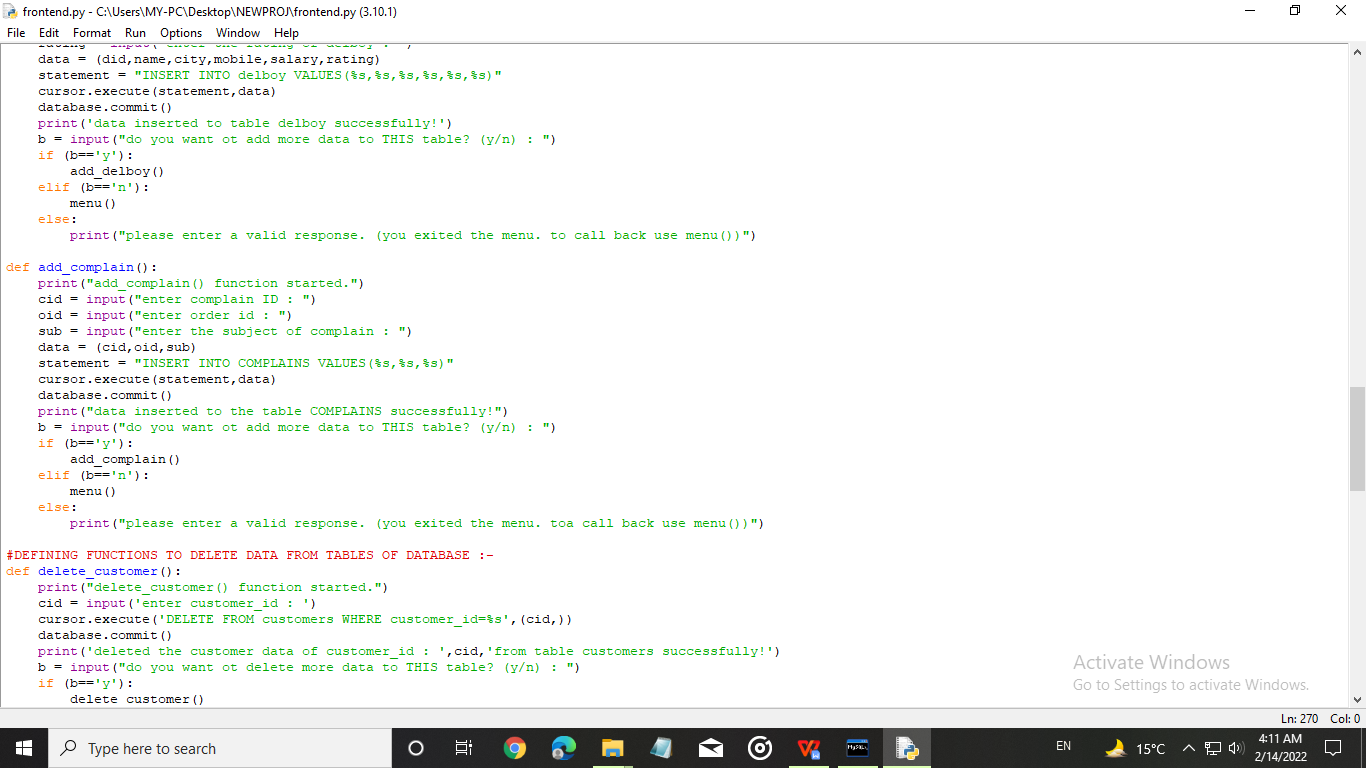
print(“ WELCOME TO THE DATABASE MANAGEMENT USER INTERFACE OF ESHOP")

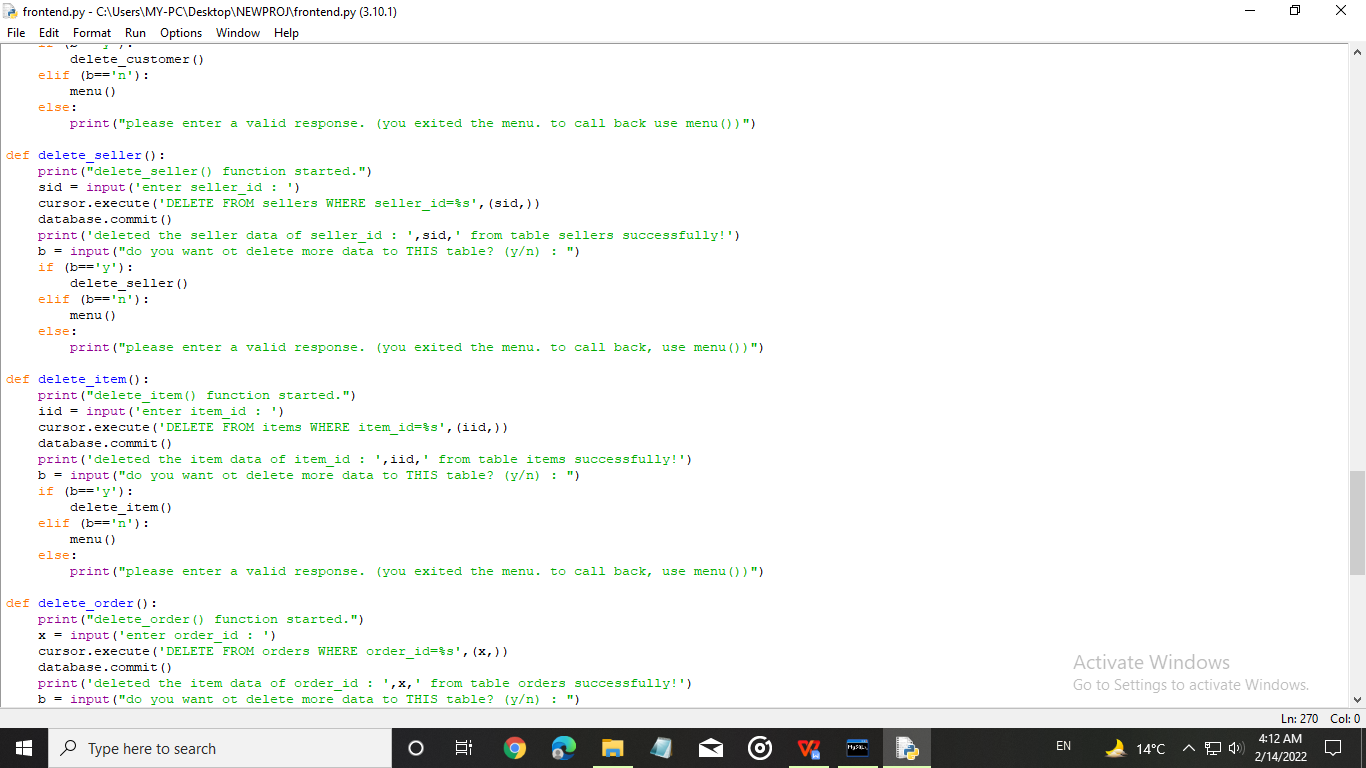
print("(created by Mohammad Maasir @ date 13th Feb,2022, as a school project.)")

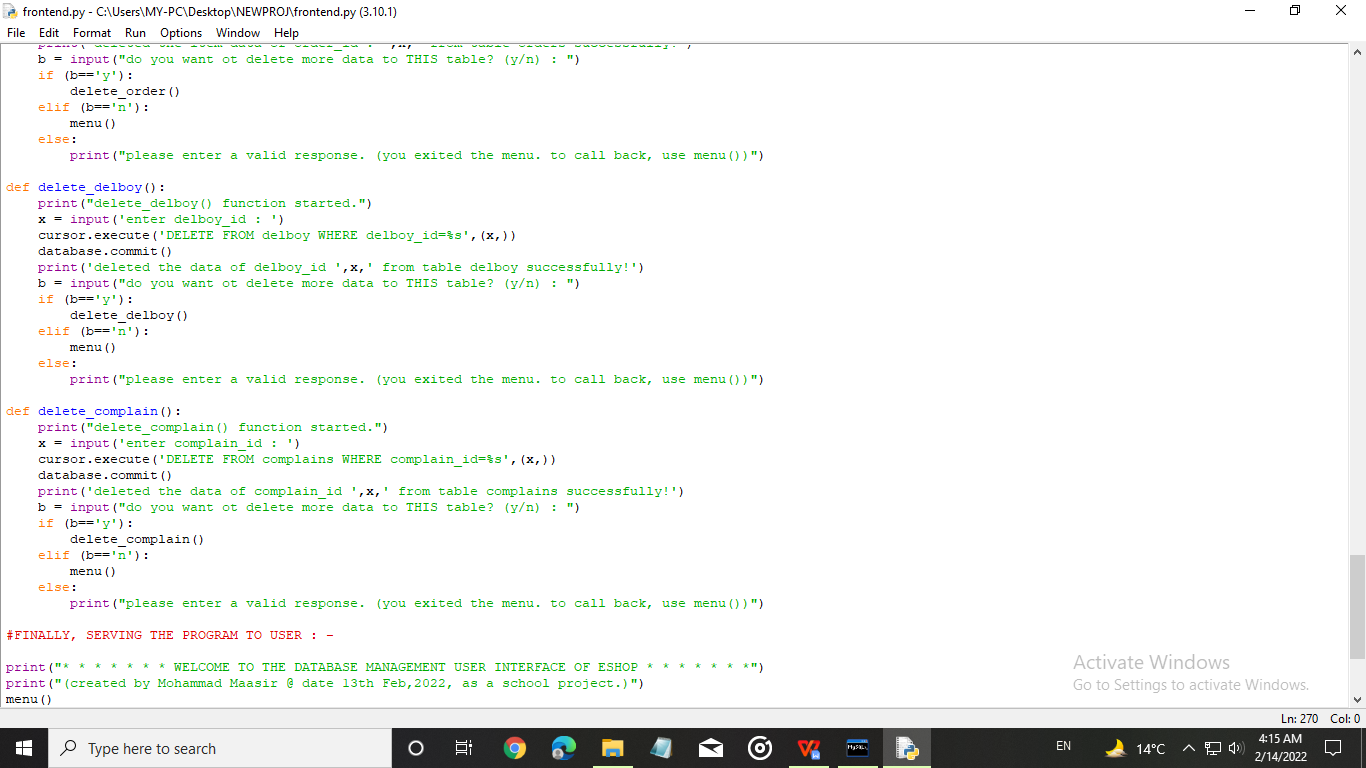
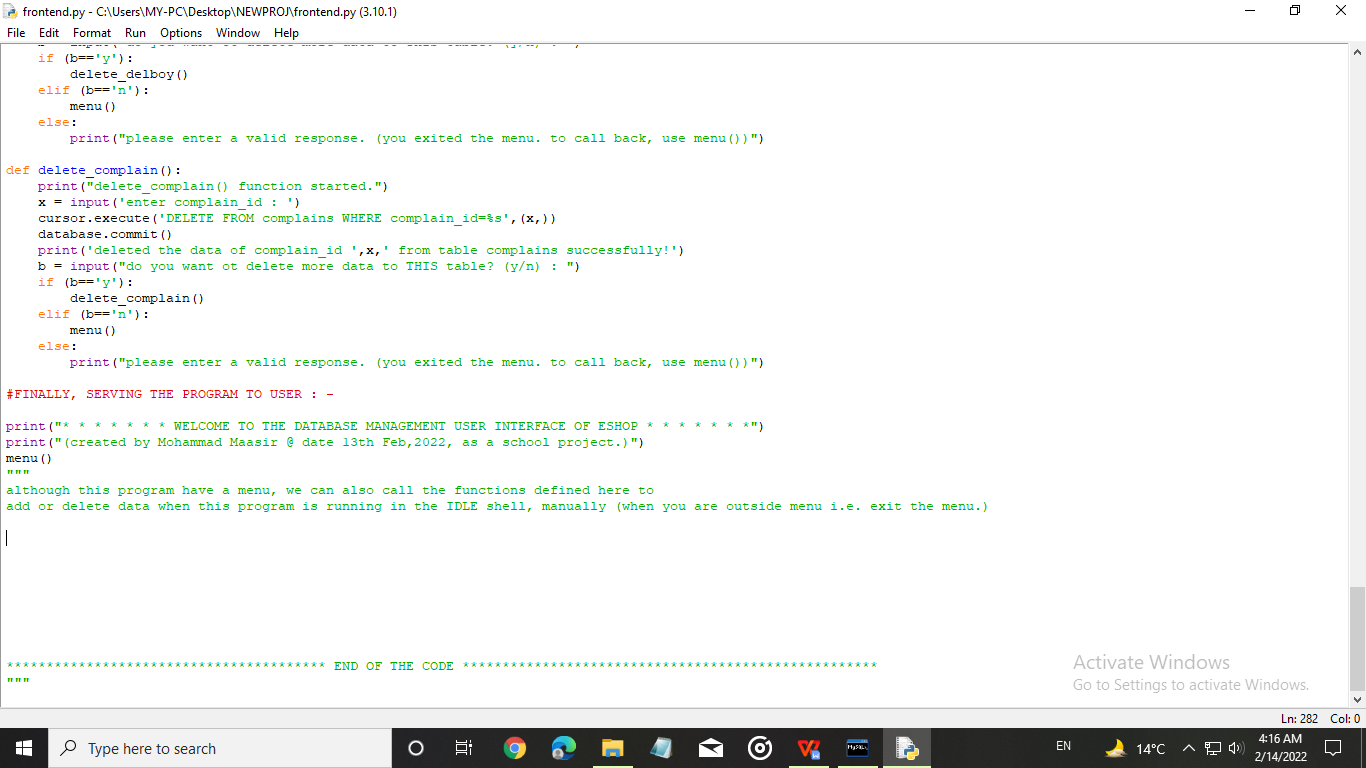
menu()

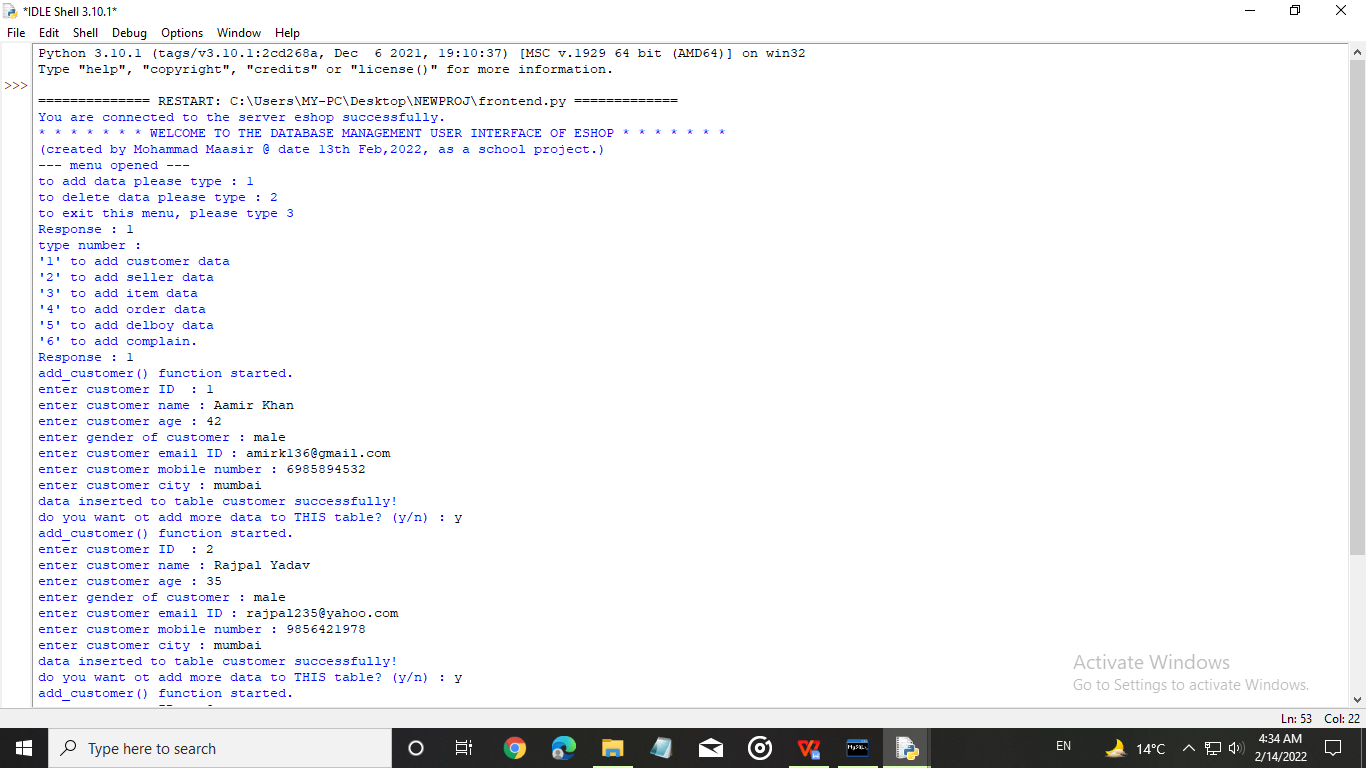
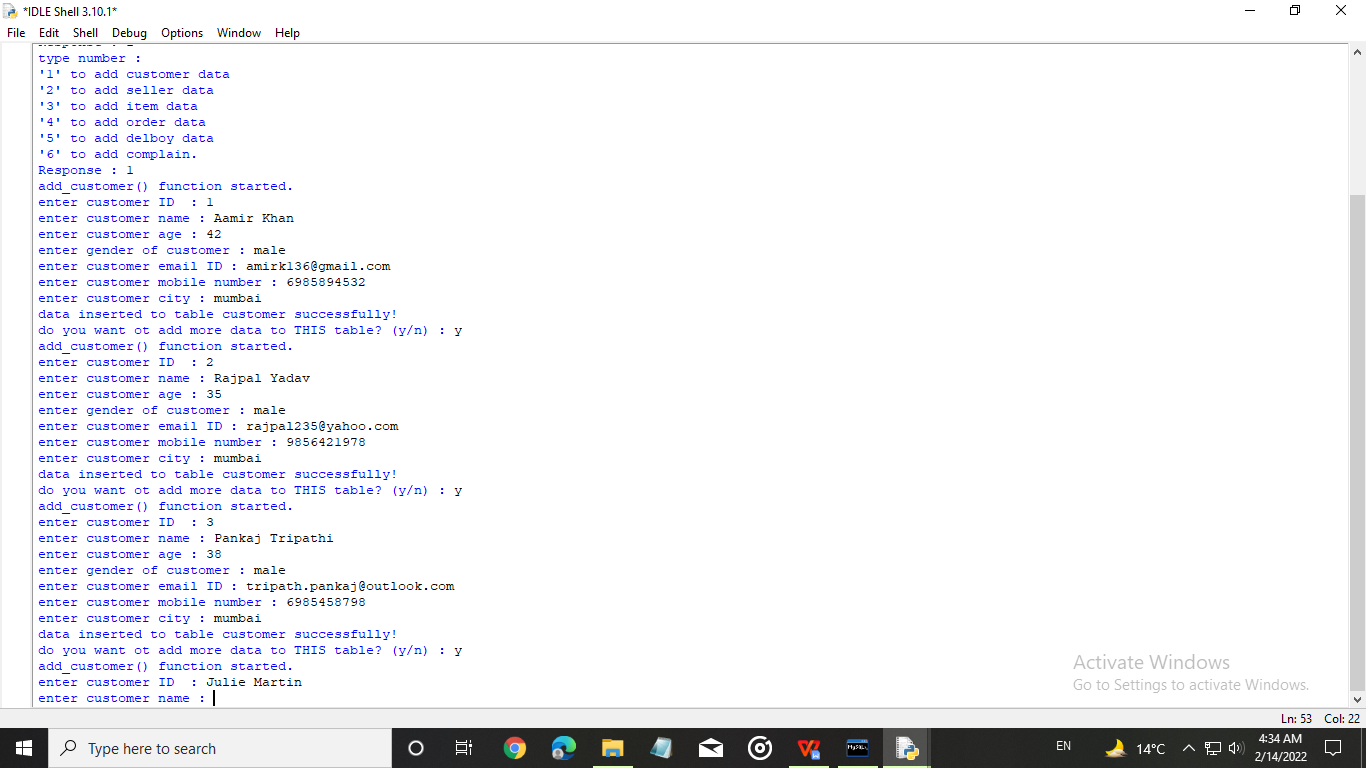
****

****

****

****

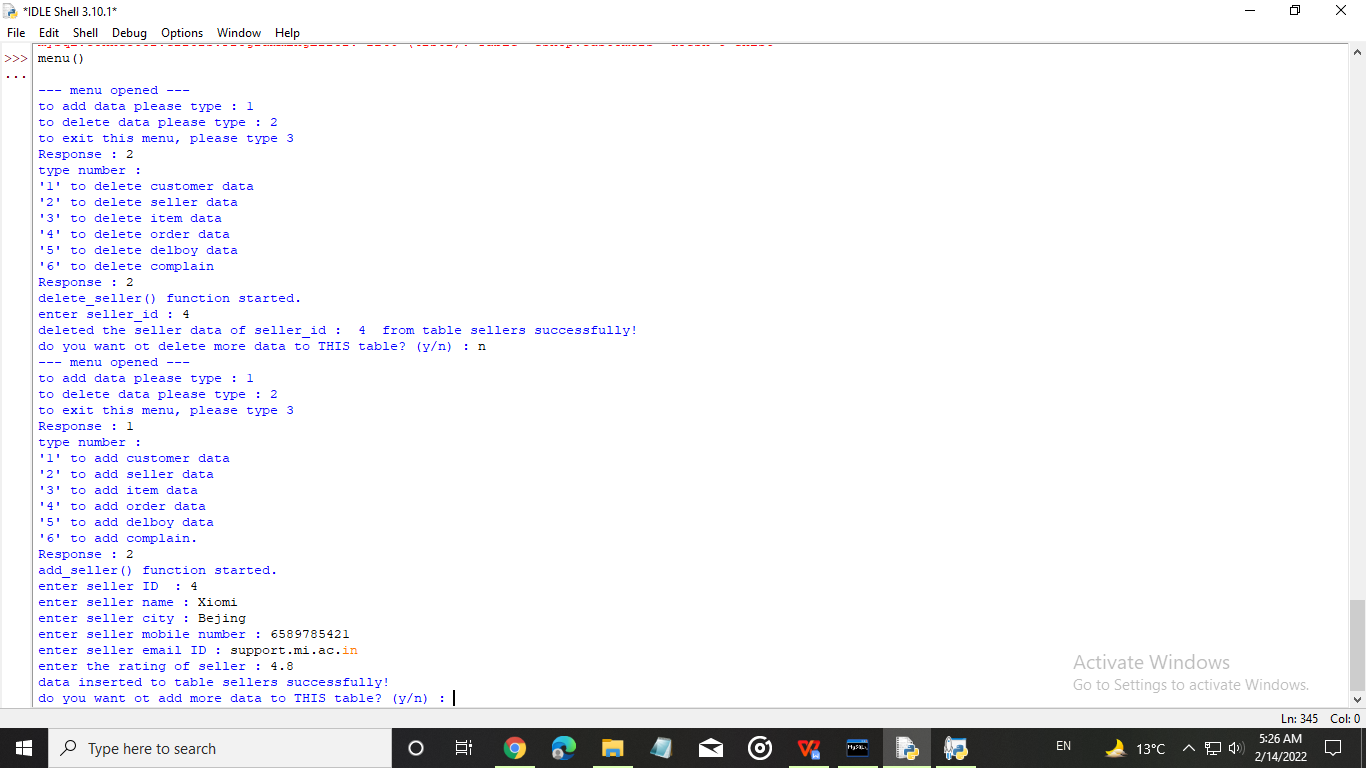
****



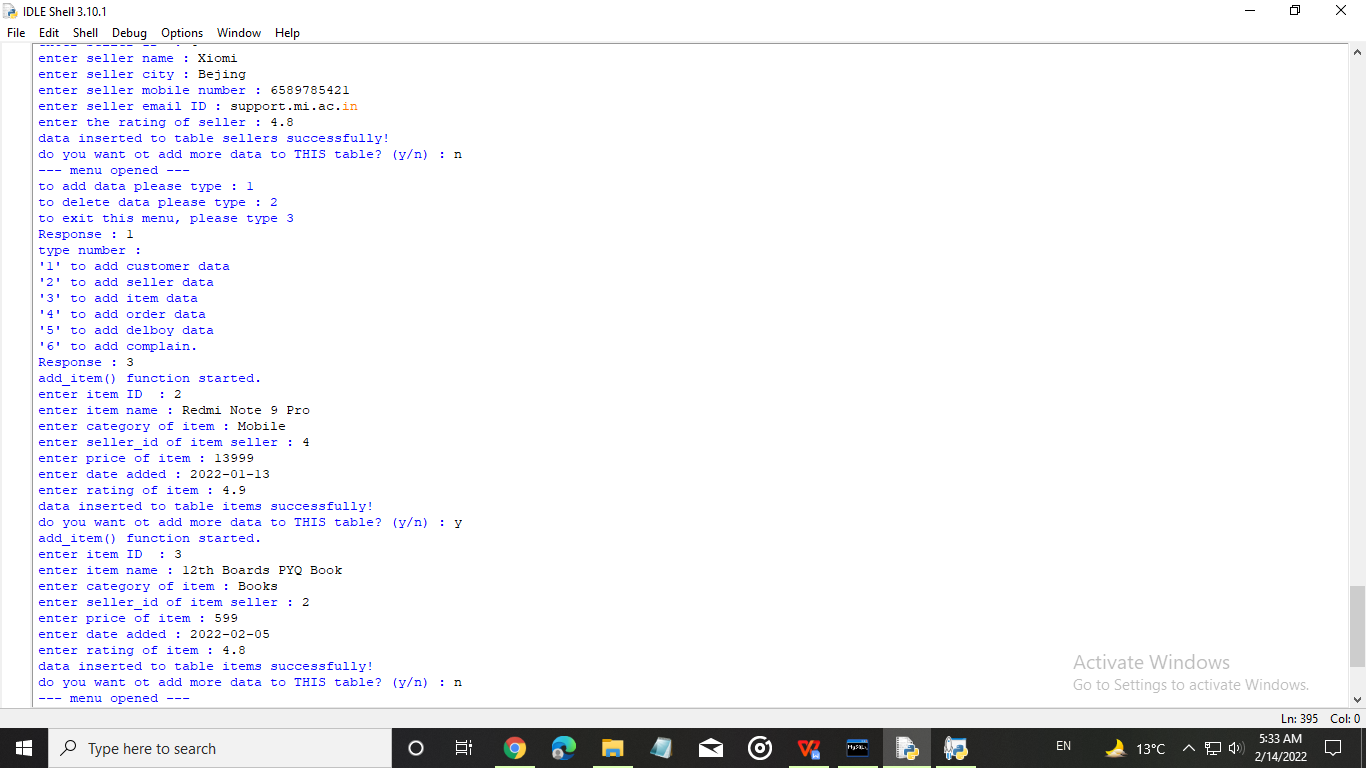
*For customers :-*

ENTERING DATA TO THE SERVER (Using the above code to feed/insert data in the tables of the server)

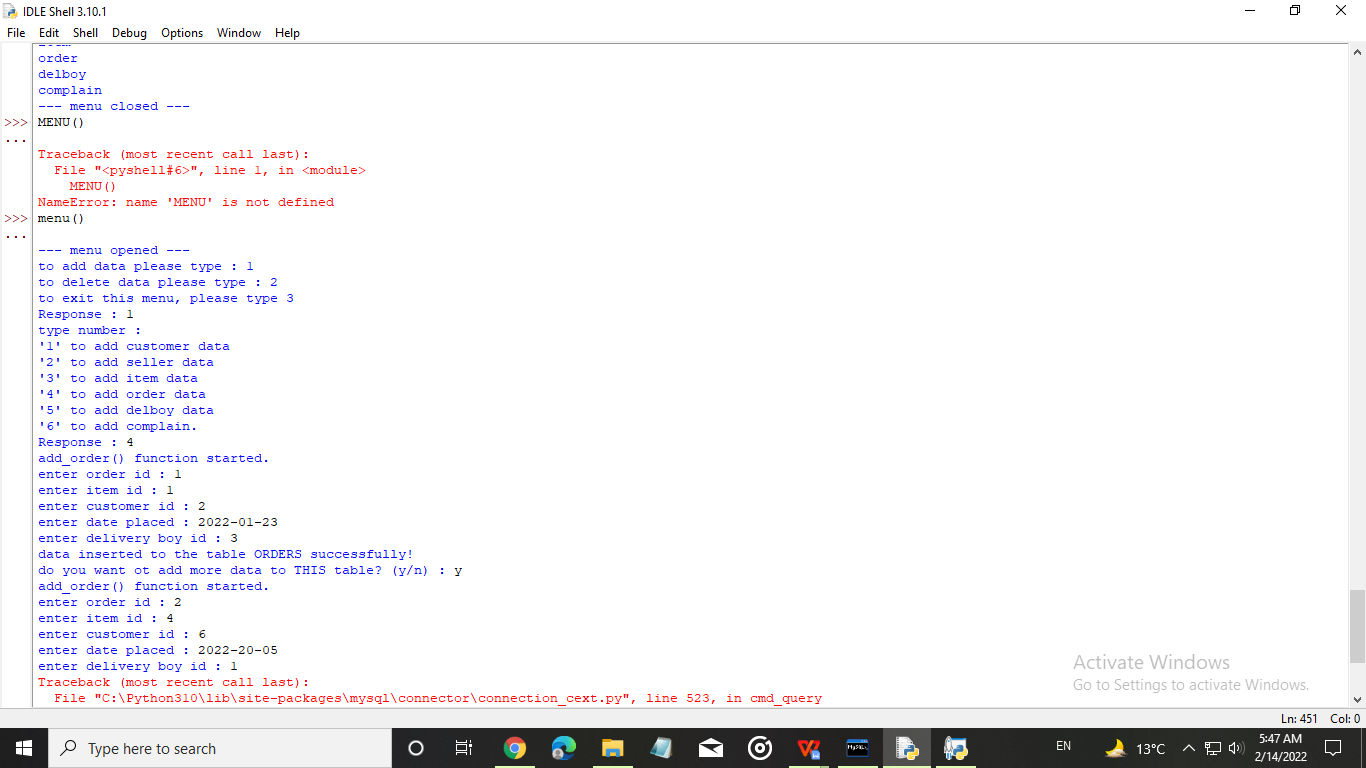
****



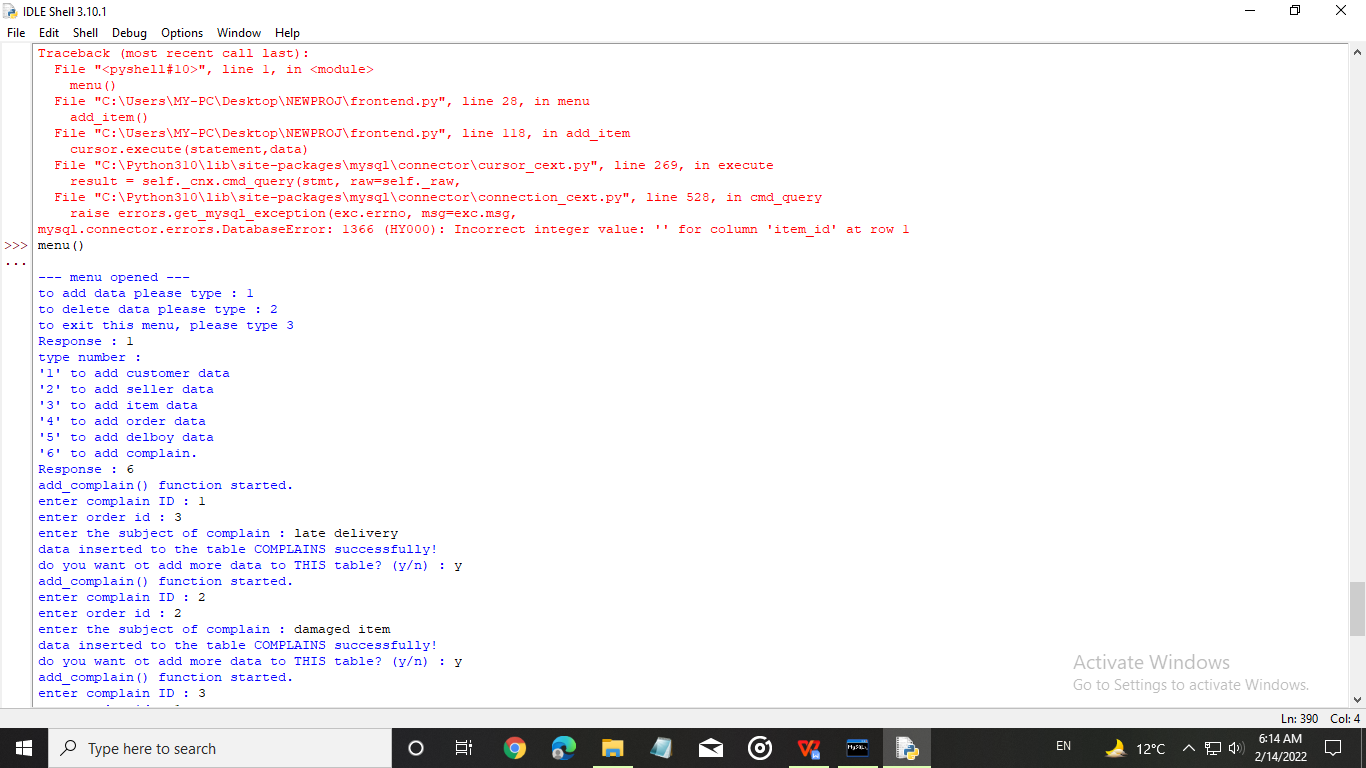
*For sellers :-*



*For items*

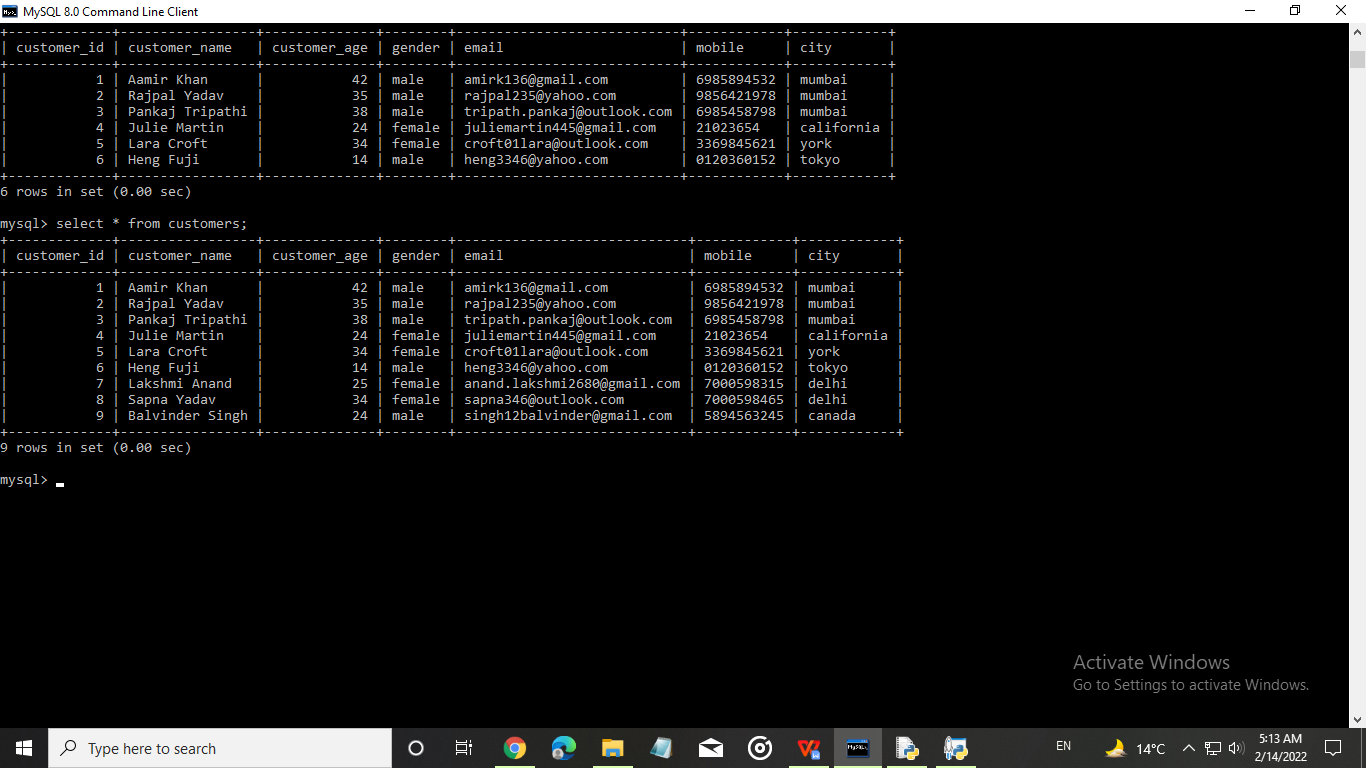
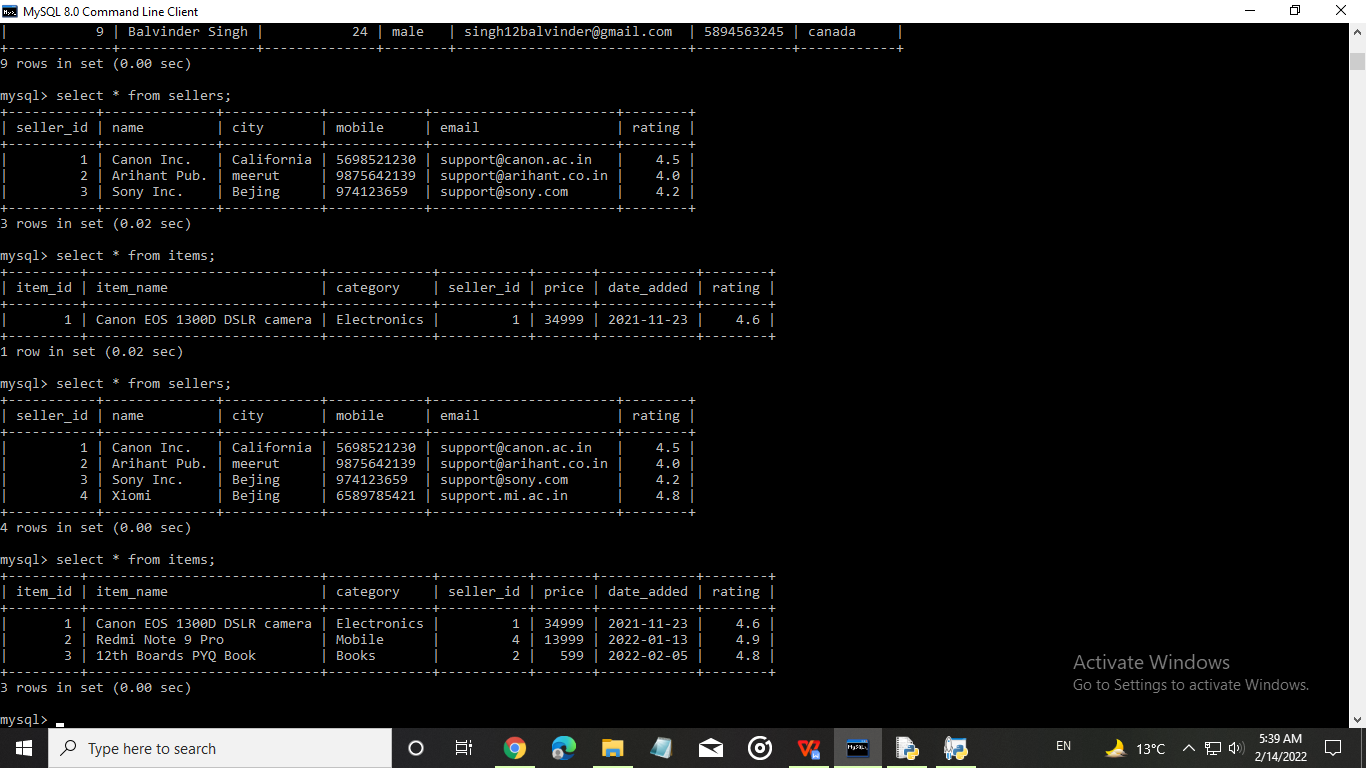
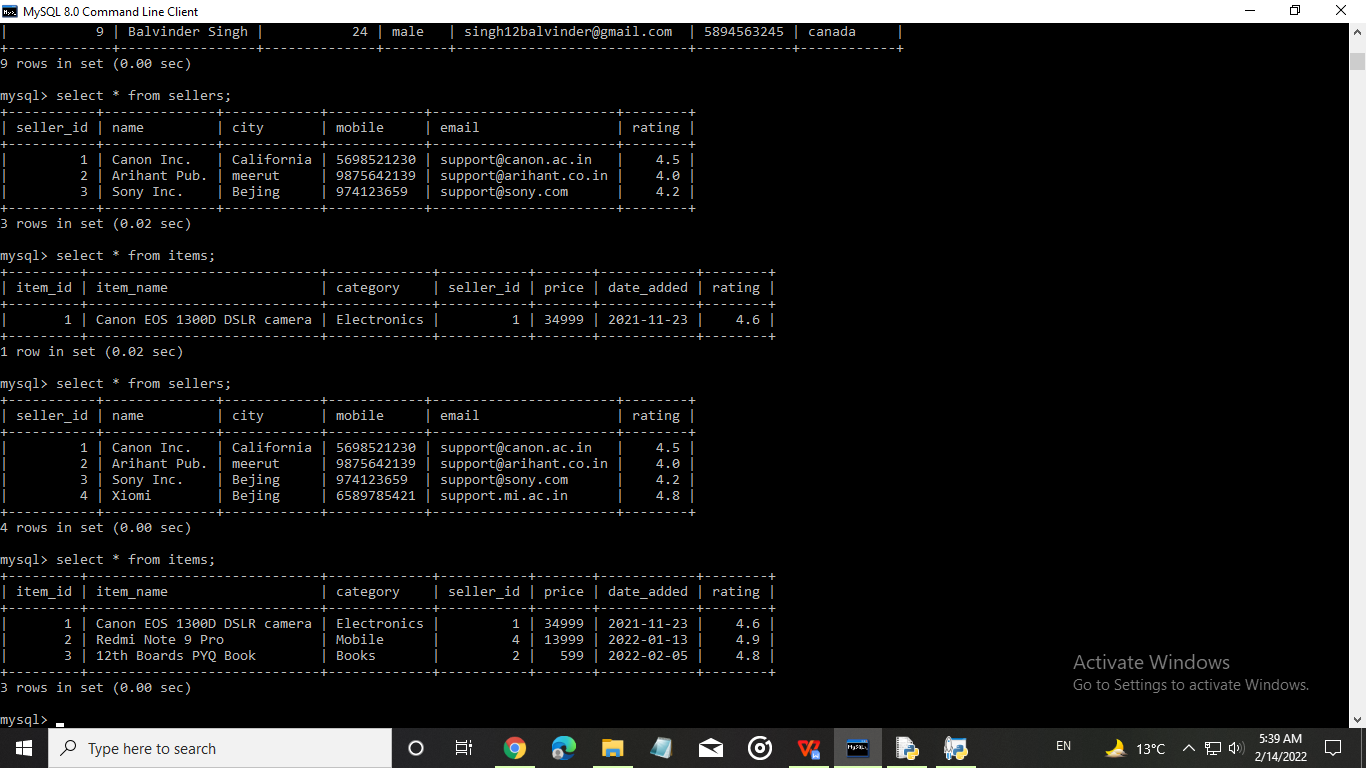


*For Orders placed*

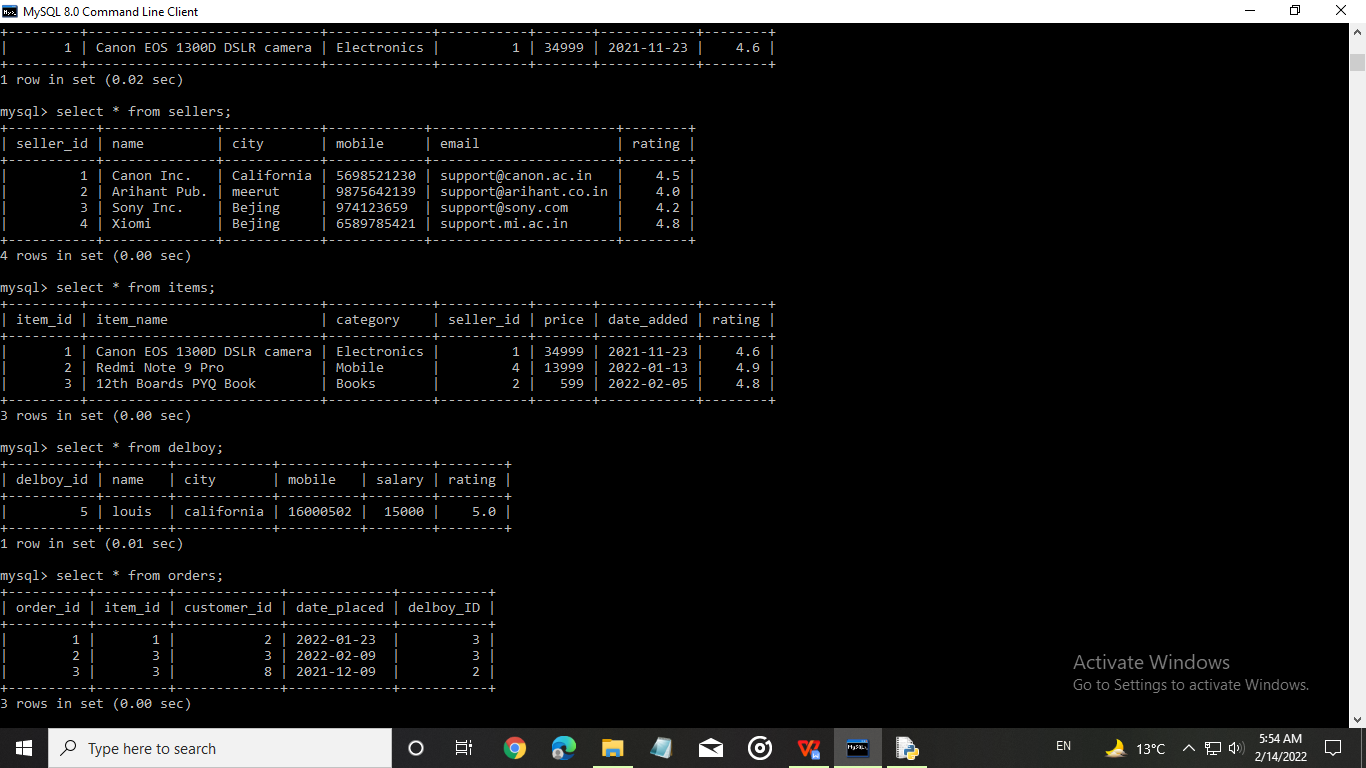


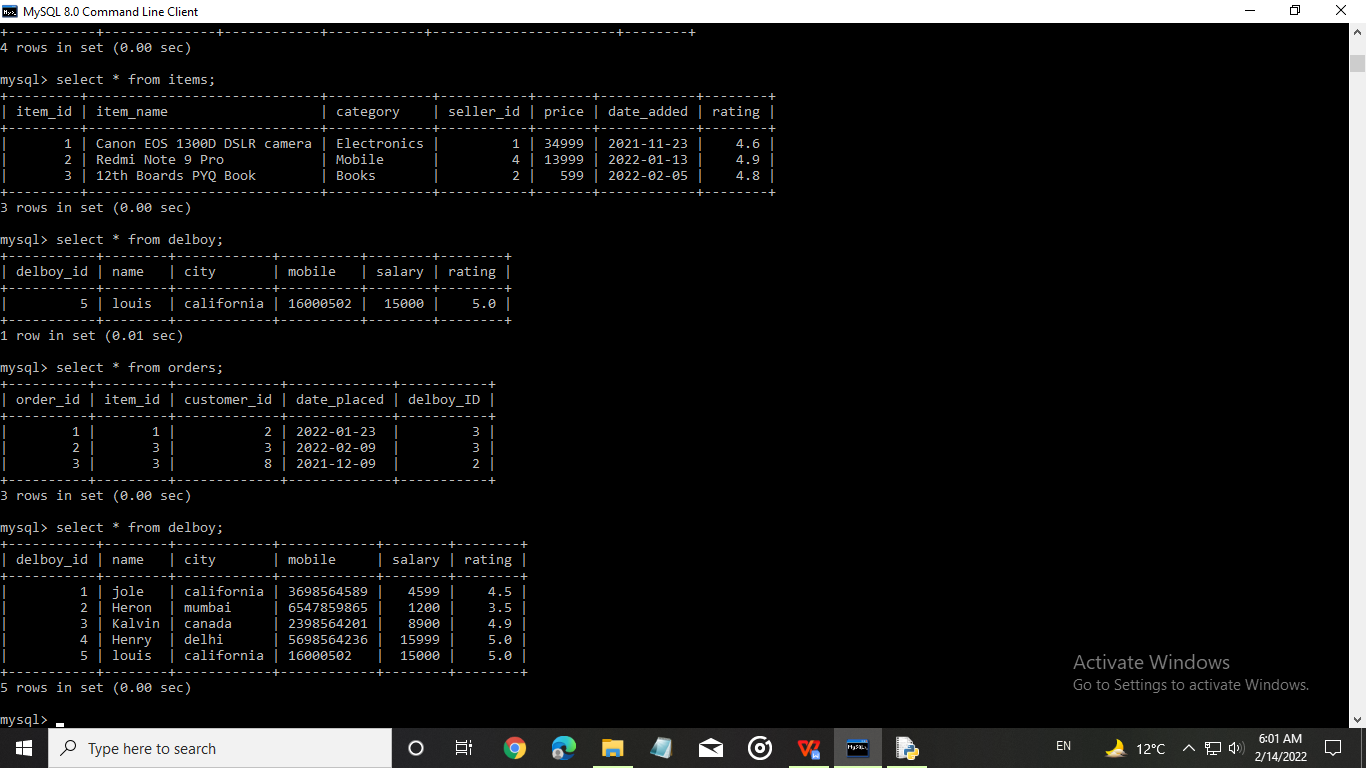
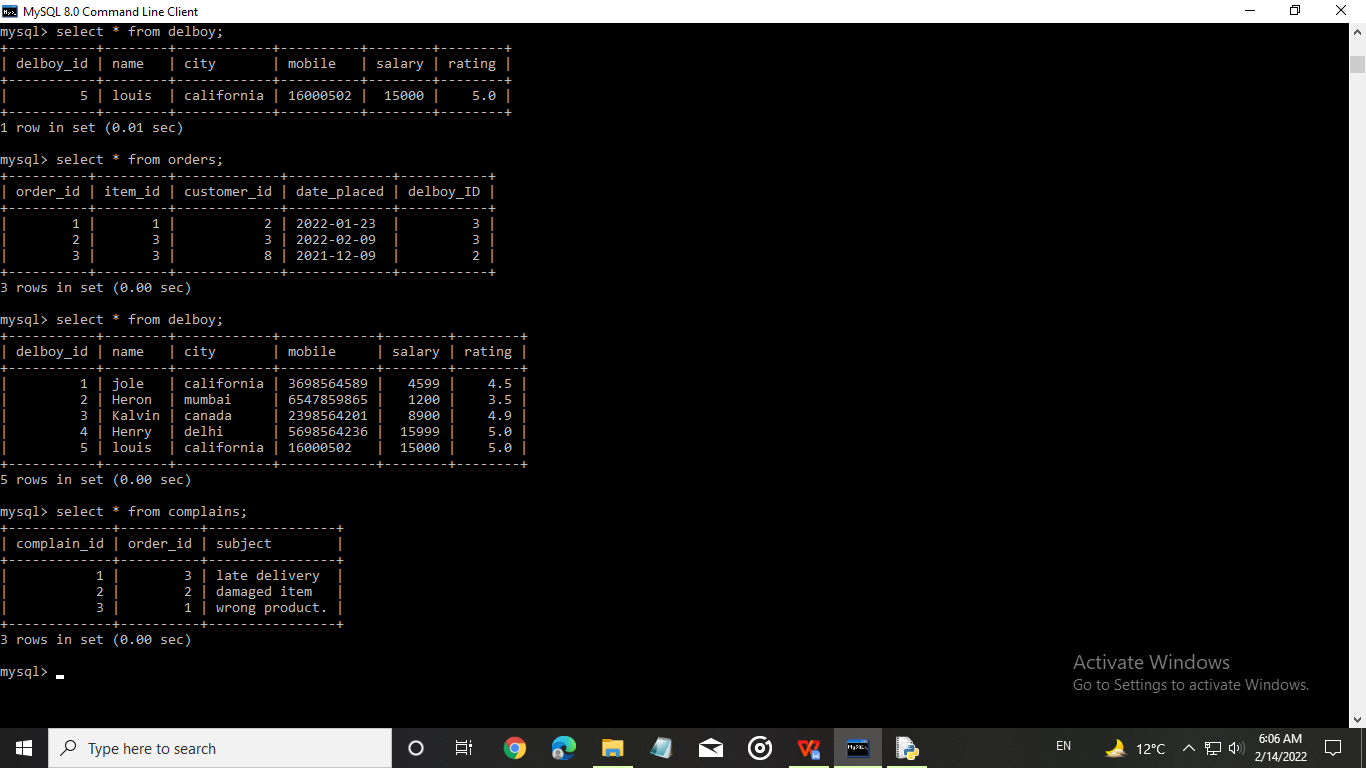
*For complains*

*For delivery boys*

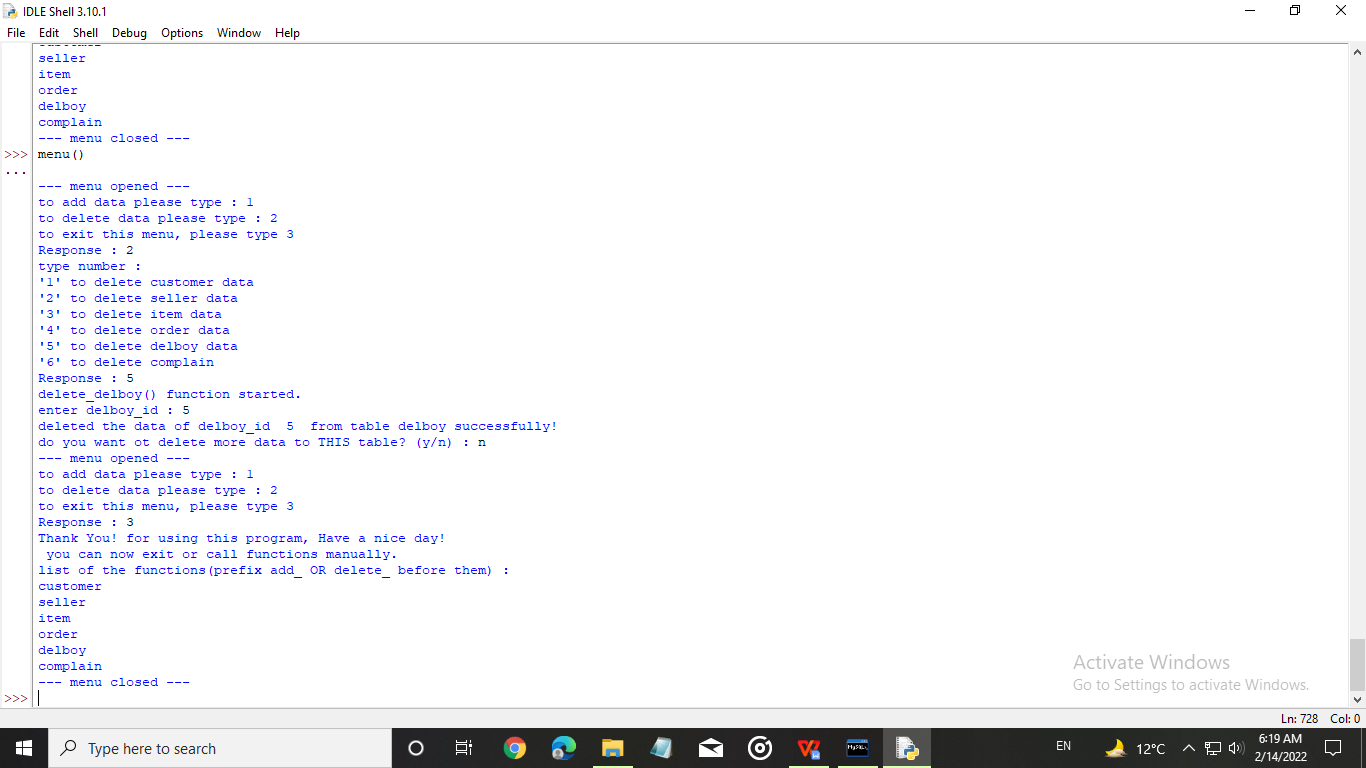
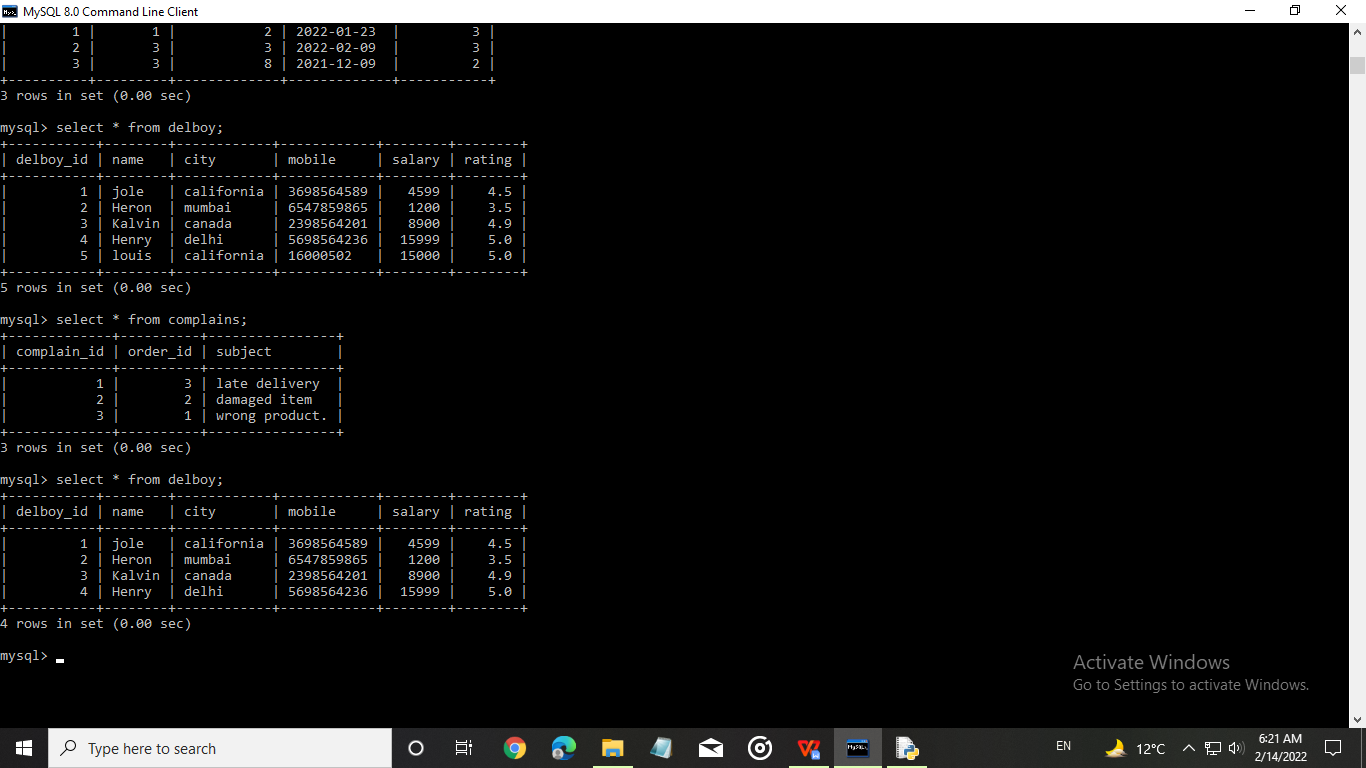
****

OUTPUT FOR THE COMMANDS

****

****

DEMONSTRATION FOR DELETING DATA



*Code for Graphs(using Python’s Matplotlib)*

import matplotlib.pyplot as plt

import numpy as np

#graph 1

x1 = [2000,2005,2010,2015,2020]

y1= [15,18,19,22,35]

plt.plot(x1,y1,marker='o',color = 'violet')

plt.xlabel("Year")

plt.ylabel("number of users (in Thousands)")

plt.show()

#graph 2

arr1 = [48,60,2]

labels1 = ['males','females','others']

colors = ['yellow','hotpink','grey']

plt.pie(arr1,labels = labels1,colors=colors)

plt.legend(title = 'Genderwise Proportion of number of customers as of the year 2021 :')

plt.show()

#graph 3

x2 = ['Grocery','Wardrobe','Electronics','Books','Furniture']

y2 = [24,57,107,68,20]

plt.bar(x2,y2, color="lightgreen")

plt.xlabel("Category of items")

plt.ylabel("Profit made by the website in the year 2020")

plt.show()

#graph 4

x3 = ['Peter England','Canon Inc.','Sony Inc.','Xiomi','Apple Inc.','Samsung']

y3 = [2.7,4.6,3.5,4.5,4.8,2.2]

plt.barh(x3,y3,color="pink")

plt.ylabel("Popular Seller(Brand) Names")

plt.xlabel("Mean Customer Rating out of 5.0")

plt.show()

#graph 5

x4 = ['Grocery','Wardrobe','Electronics','Books','Furniture']

y4 = [5056,3269,25699,1509,31220]

plt.bar(x4,y4,color = 'orange')

plt.xlabel('Category of Item : ')

plt.ylabel('Average Product Retail Price in Rupees : ')

plt.show()

NOTE :

The data I am using in the graphs in the following sections is related to the website management system. But this data is NOT related to any real life survey.

GRAPHS RELATED TO MY PROJECT

#graph 6

arr2 = [18,20,25,23,54,16,12,19,13,23,25,24,26,26,24,23,21,33,32,33,35,41,48,48,41,49,45,48,47,27,55,56,45,44,41,43,42,41,42,42,41,22,21,21,21,21,23,23,22,22,21,21,]

bins = [0,5,10,15,20,25,30,35,40,45,50,55,60,65,70,70]

plt.hist(arr2,bins=bins,color = 'skyblue')

plt.xlabel('age groups')

plt.ylabel('number of customers(in thousands)')

plt.show()

#graph 9 number of sellers and location

country = ['China','Japan','USA','Korea','India','Others']

seller = [15,7,4,5.8,3,5]

colors = ["red",'hotpink','Blue','violet','lightblue','lightgreen']

plt.barh(country,seller,color=colors)

plt.ylabel("country")

plt.xlabel("number of sellers in thousands")

plt.show()

#graph 8 profit of company v/s year

x6 = [2000,2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,2015,2016,2017,2018,2019,2020,2021,2022,]

y6=[0.1,0.8,1,1.5,-2.4,3.1,07.5,4.7,14,5.0,1.0,12,-0.5,1.0,2.7,5.0,15.0,30.5,28,20,31,35,12]

plt.plot(x6,y6,marker = 'o',color = 'green')

plt.xlabel("Financial Year :")

plt.ylabel("Profit made by our website : ")

plt.show()

#graph 7 : customer proportion(location wise)

x5 = ['Delhi','Mumbai','California','Canada','Washington D.C.','U.A.E','Bejing','Tokyo','others']

y5 = [10,12,15,8,12,9,17,14,18]

explode = [0.1,0.05,0.20,0.15,0.09,0.1,0.2,0.1,0.05]

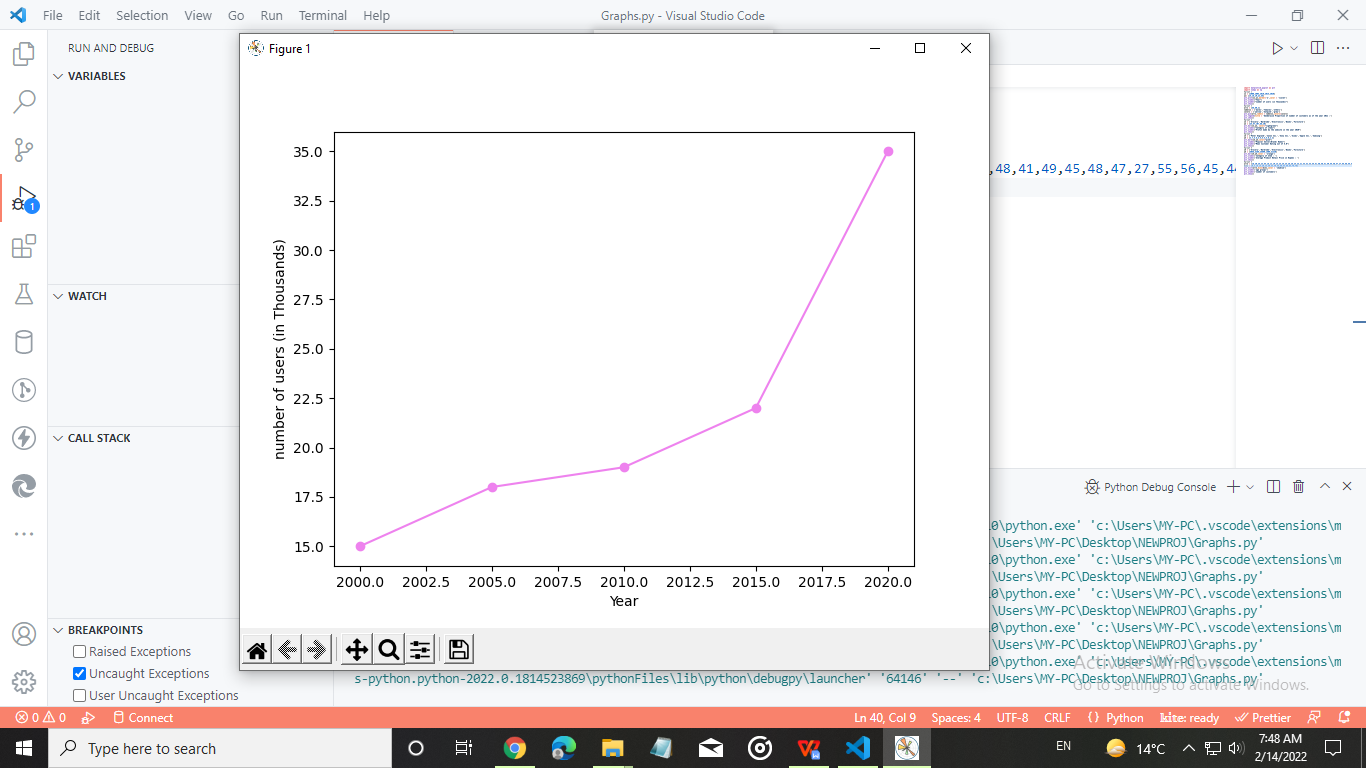
plt.pie(y5,labels=x5,explode=explode,shadow = True,colors = ['pink','lightblue','lightgreen','hotpink','crimson','purple','violet','skyblue','grey'],autopct='%1.1f%%')

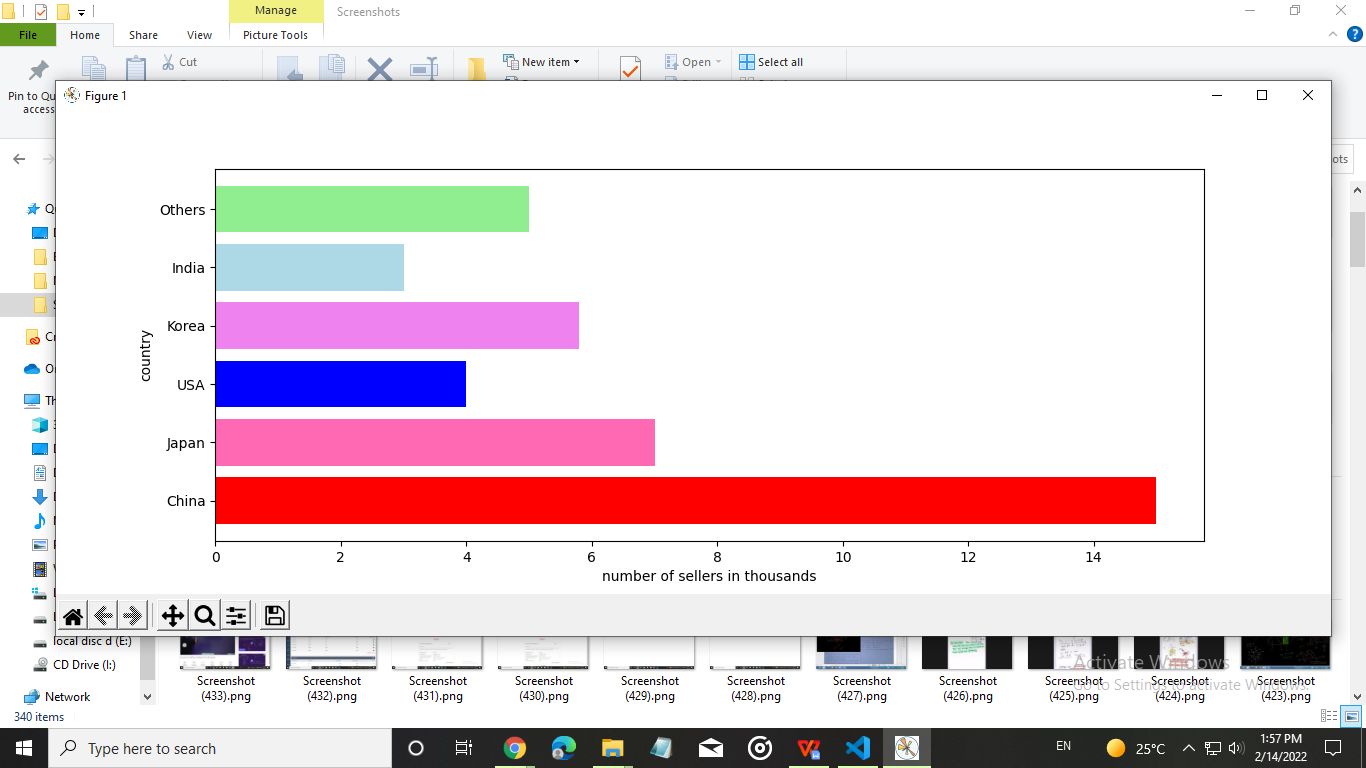
plt.legend(title = 'Proportion of customers location wise',bbox\_to\_anchor=(1,1))

plt.show()

*Output for Graph Code:*

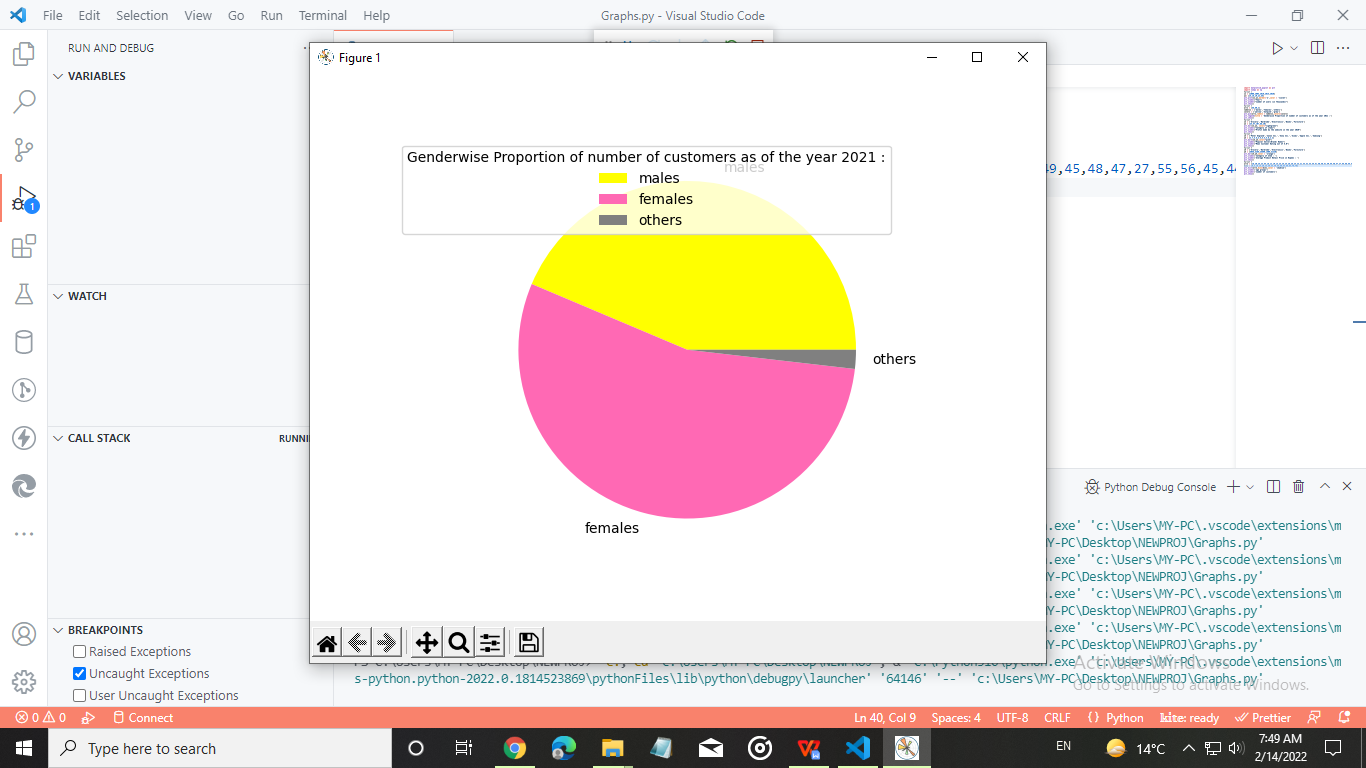
Total number of users v/s year

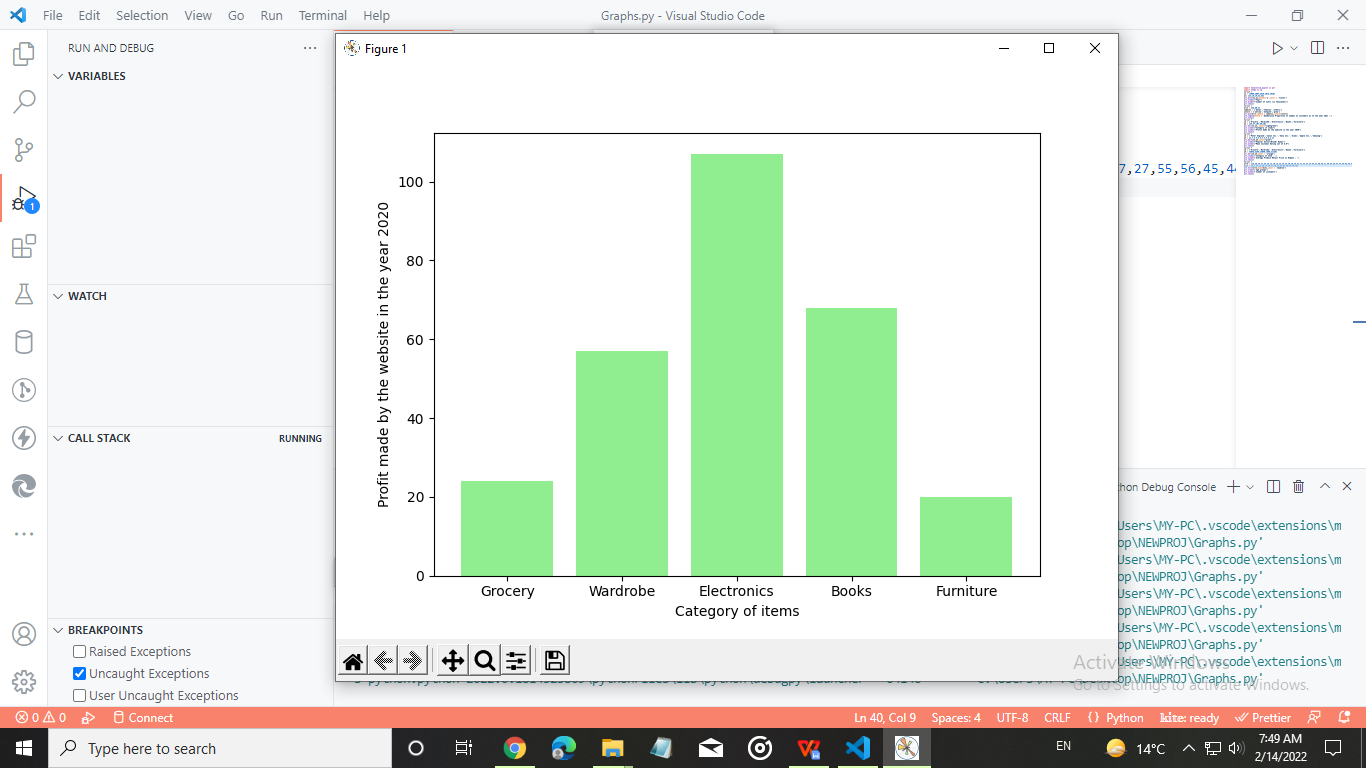




Number of sellers according to countries

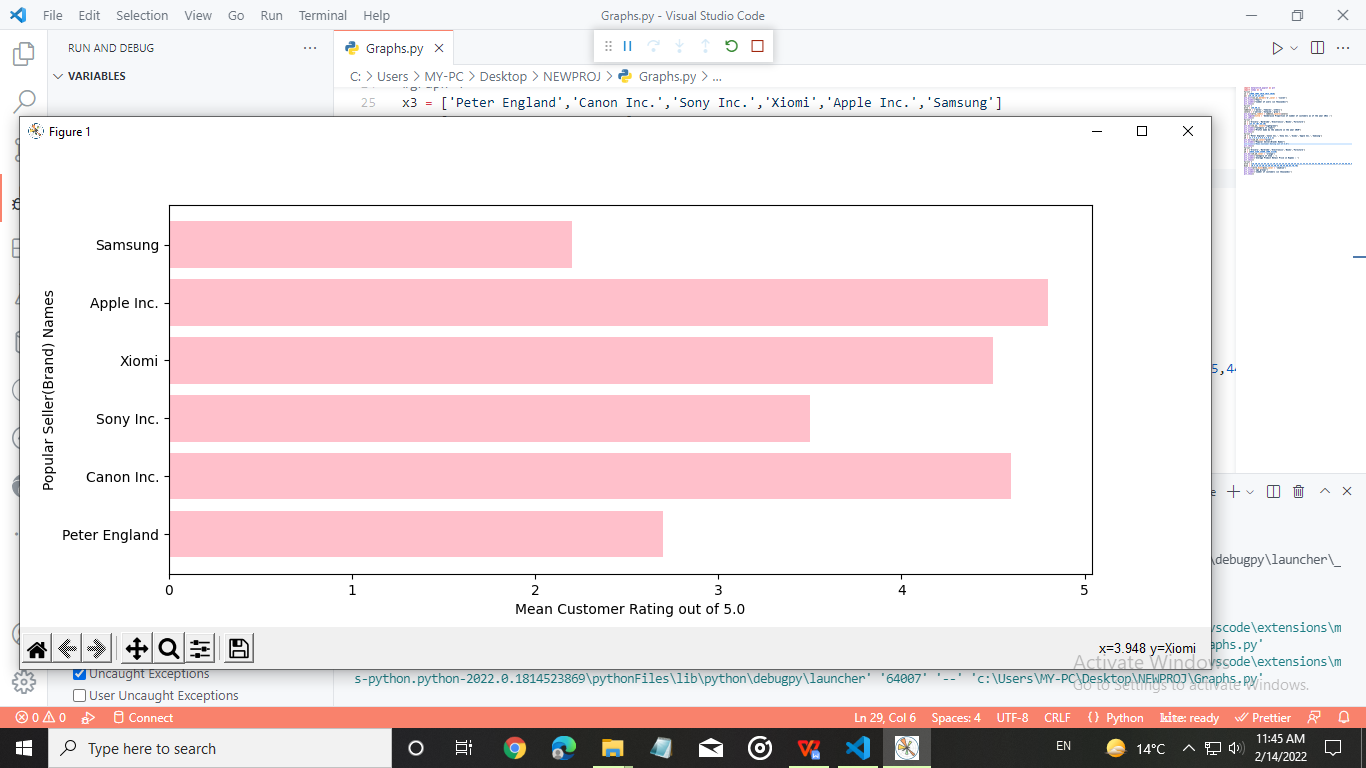
Customer proportion(Gender wise)

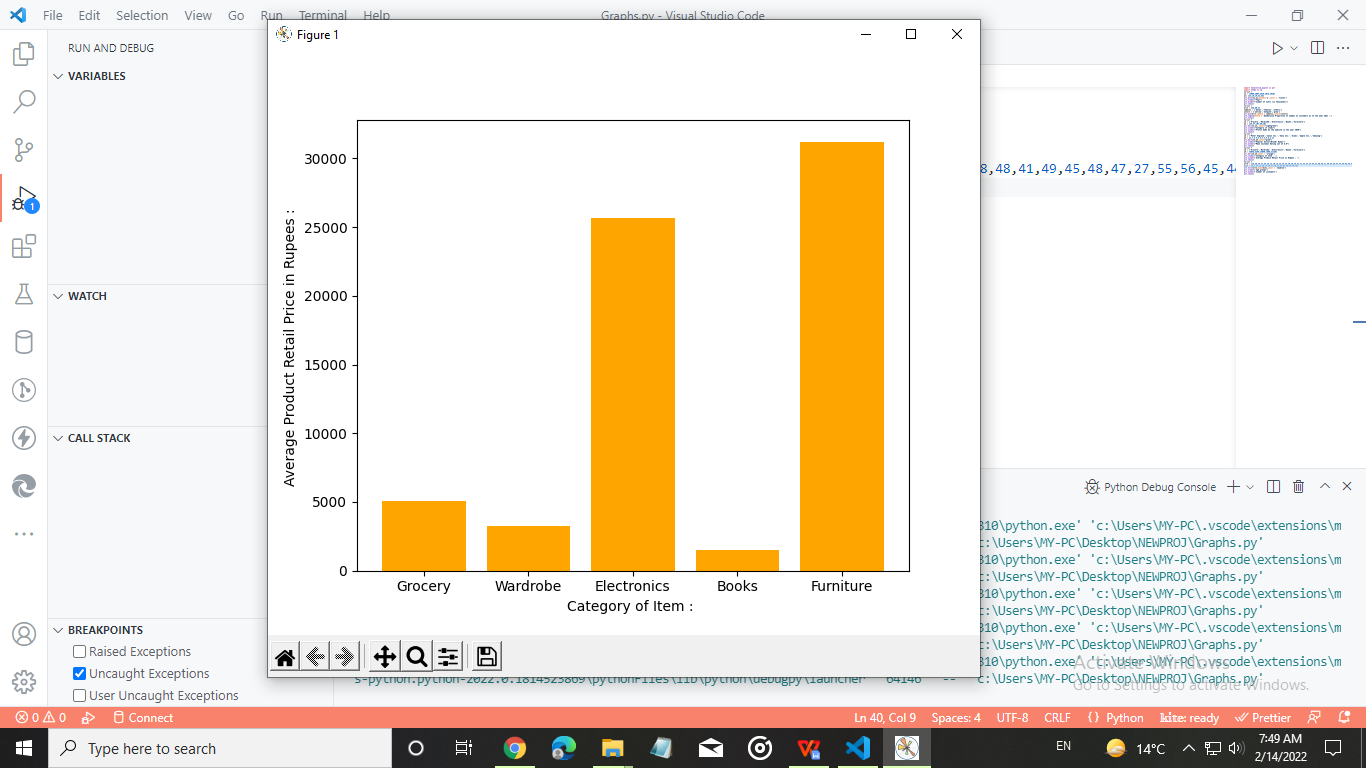
****

****

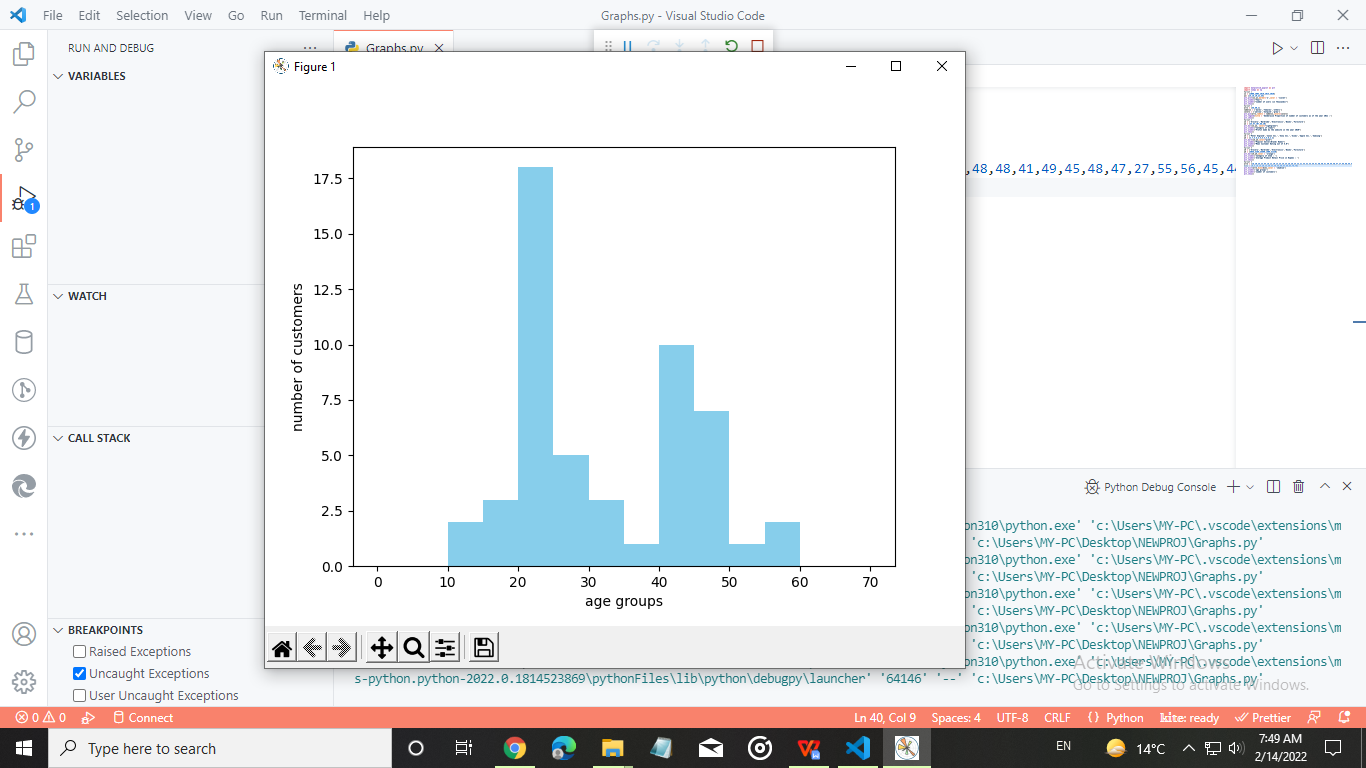
Profit made by website(in lakhs) v/s Category of Product

Some popular sellers v/s Mean customer rating

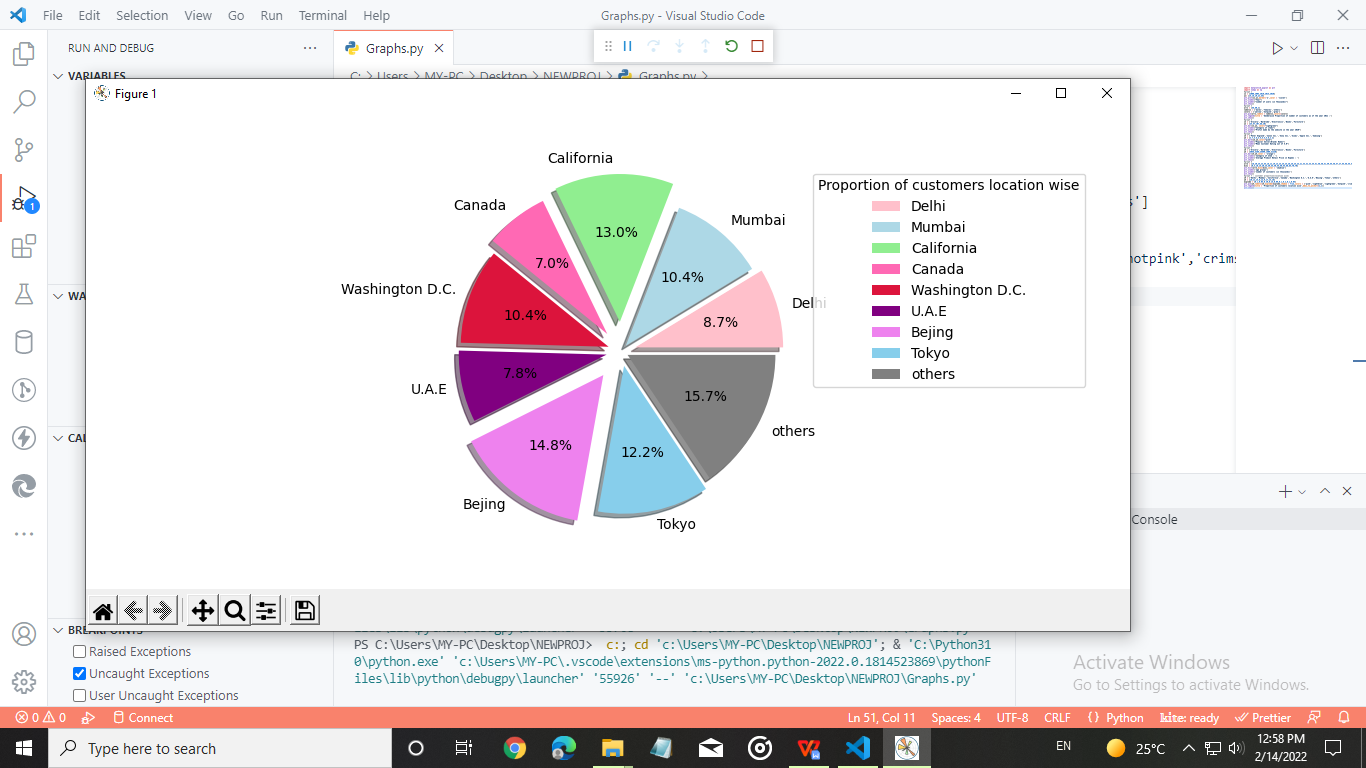
****

****

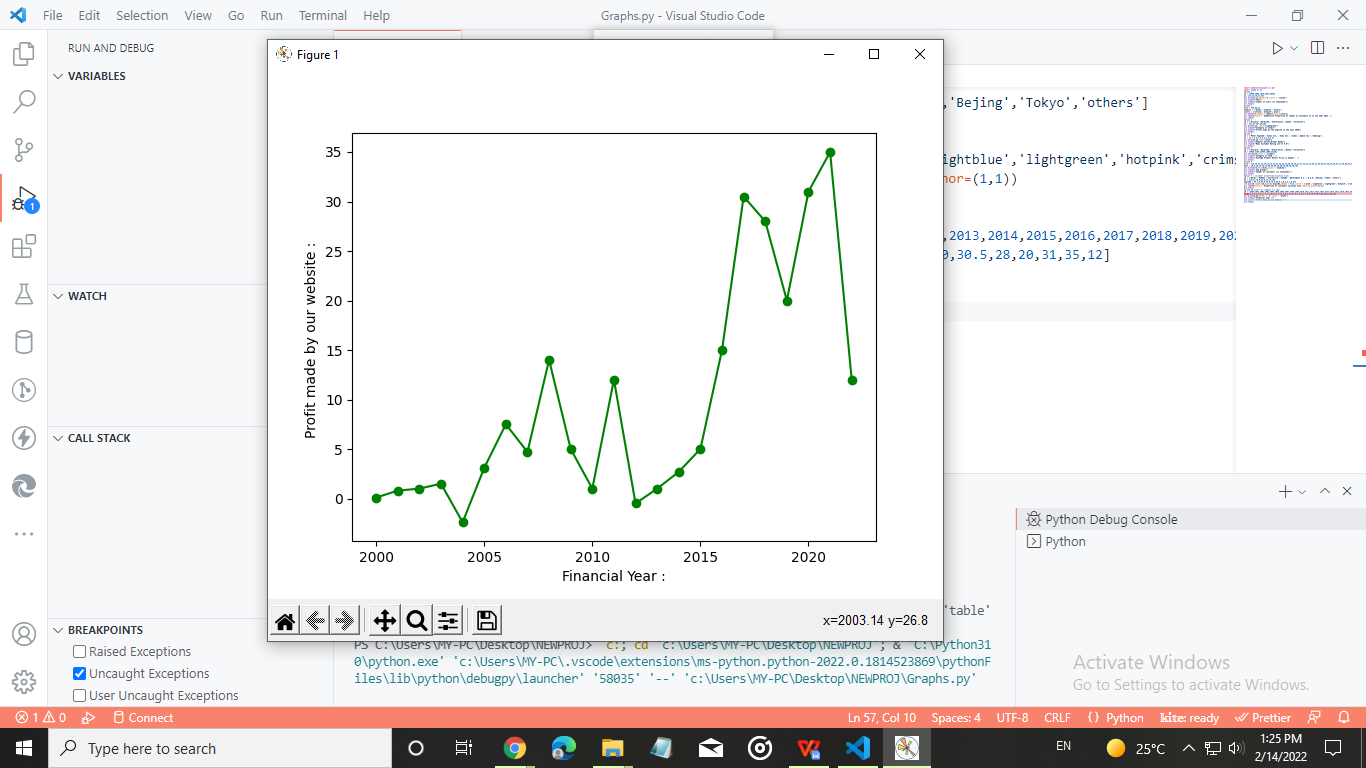
Avg. Product retail price v/s Category of products

****

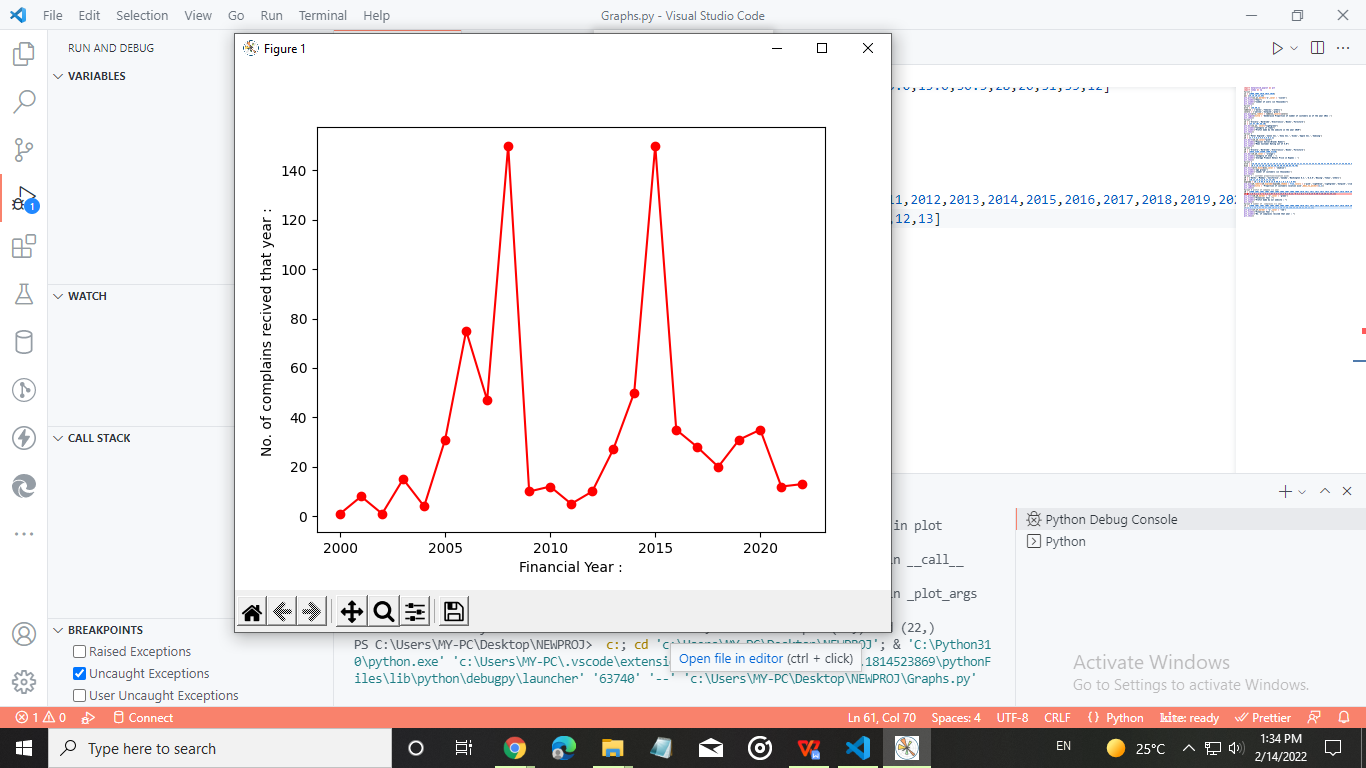
histogram of no. of users v/s age

****

Proportion of customers location wise

****

Profit made(in thousand rupees) v/s year



Number of complains received

HILIGHTED CODE FROM VS CODE

#importing the connector :-

#from distutils.util import execute

import mysql.connector as c

#defining the database :-

#Note: the name of database is Eshop. you have to create database Eshop to run this program.

database = c.connect(host = "localhost", user = 'root', password = "password", database = 'eshop')

#defining cursor(it will be used to excute SQL commands) :-

cursor = database.cursor(buffered=True)

print('You are connected to the server eshop successfully.')

cursor.execute('CREATE TABLE IF NOT EXISTS items(item\_id INT(4) PRIMARY KEY NOT NULL,item\_name VARCHAR(50),category VARCHAR(30),seller\_id INT(4) NOT NULL,price INT(8) NOT NULL,date\_added DATE,rating FLOAT(2,1))' )

cursor.execute('CREATE TABLE IF NOT EXISTS customers(customer\_id INT(4) PRIMARY KEY NOT NULL,customer\_name VARCHAR(30) NOT NULL,customer\_age INT(3),gender ENUM("male","female","others"),email VARCHAR(30),mobile VARCHAR(12),city VARCHAR(20))')

cursor.execute('CREATE TABLE IF NOT EXISTS delboy(delboy\_id INT(4) PRIMARY KEY NOT NULL,name VARCHAR(30) NOT NULL,city VARCHAR(20),mobile VARCHAR(12),salary INT(4),rating float(2,1))')

cursor.execute('CREATE TABLE IF NOT EXISTS sellers(seller\_id INT(4) PRIMARY KEY NOT NULL,name VARCHAR(30) NOT NULL,city VARCHAR(20),mobile VARCHAR(12),email VARCHAR(30),rating float(2,1))')

cursor.execute('CREATE TABLE IF NOT EXISTS orders(order\_id INT(4) PRIMARY KEY NOT NULL,item\_id INT(4) NOT NULL,customer\_id INT(4) NOT NULL,date\_placed DATE,delboy\_ID INT(4) NOT NULL)')

cursor.execute('CREATE TABLE IF NOT EXISTS complains(complain\_id INT(4) PRIMARY KEY NOT NULL,order\_id INT(4) NOT NULL,subject VARCHAR(20) NOT NULL)')

#MAKING THE MENU(USER INTERFACE) FOR FRONT END

def menu():

    print("--- menu opened ---")

    a = int(input("to add data please type : 1 \nto delete data please type : 2 \nto exit this menu, please type 3 \nResponse : "))

    if (a==1):

        b= int(input("type number : \n'1' to add customer data \n'2' to add seller data \n'3' to add item data \n'4' to add order data \n'5' to add delboy data \n'6' to add complain. \nResponse : "))

        if (b==1):

            add\_customer()

        elif (b==2):

            add\_seller()

        elif (b==3):

            add\_item()

        elif (b==4):

            add\_order()

        elif (b==5):

            add\_delboy()

        elif (b==6):

            add\_complain()

        else:

            print('please enter a valid response')

elif (a==2):

        b= int(input("type number : \n'1' to delete customer data \n'2' to delete seller data \n'3' to delete item data \n'4' to delete order data \n'5' to delete delboy data \n'6' to delete complain \nResponse : "))

        if (b==1):

            delete\_customer()

        elif (b==2):

            delete\_seller()

        elif (b==3):

            delete\_item()

        elif (b==4):

            delete\_order()

        elif (b==5):

            delete\_delboy()

        elif (b==6):

            delete\_complain()

        else:

            print('please enter a valid response')

            menu()

    elif(a==3):

        print('Thank You! for using this program, Have a nice day! \n you can now exit or call functions manually. \nlist of the functions(prefix add\_ OR delete\_ before them) : \ncustomer\nseller\nitem\norder\ndelboy\ncomplain\n--- menu closed ---')

    else:

        print('please enter a valid response')

        menu()

#DEFINING FUNCTIONS FOR FEEDING DATA IN THE TABLES OF THE SERVER

def add\_customer():

    print("add\_customer() function started.")

    c\_id = input('enter customer ID  : ')

    name = input('enter customer name : ')

    age = input("enter customer age : ")

    gender = input("enter gender of customer : ")

    email = input('enter customer email ID : ')

    mobile = input('enter customer mobile number : ')

    city = input('enter customer city : ')

    data = (c\_id,name,age,gender,email,mobile,city)

    statement = "INSERT INTO customers VALUES(%s,%s,%s,%s,%s,%s,%s)"

    cursor.execute(statement,data)

    database.commit()

    print('data inserted to table customer successfully!')

    b = input("do you want ot add more data to THIS table? (y/n) : ")

    if (b=='y'):

        add\_customer()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back use menu())")

def add\_seller():

    print("add\_seller() function started.")

    s\_id = input('enter seller ID  : ')

    name = input('enter seller name : ')

    city = input('enter seller city : ')

    mobile = input('enter seller mobile number : ')

    email = input('enter seller email ID : ')

    rating = input('enter the rating of seller : ')

    data = (s\_id,name,city,mobile,email,rating)

    statement = "INSERT INTO sellers VALUES(%s,%s,%s,%s,%s,%s)"

    cursor.execute(statement,data)

    database.commit()

    print('data inserted to table sellers successfully!')

    b = input("do you want ot add more data to THIS table? (y/n) : ")

    if (b=='y'):

        add\_seller()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back use menu())")

def add\_item():

    print("add\_item() function started.")

    item\_id = input('enter item ID  : ')

    name = input('enter item name : ')

    cat = input('enter category of item : ')

    sid = input('enter seller\_id of item seller : ')

    price = input('enter price of item : ')

    date\_added = input('enter date added : ')

    rating = input('enter rating of item : ')

    data = (item\_id,name,cat,sid,price,date\_added,rating)

    statement = "INSERT INTO items VALUES(%s,%s,%s,%s,%s,%s,%s)"

    cursor.execute(statement,data)

    database.commit()

    print('data inserted to table items successfully!')

    b = input("do you want ot add more data to THIS table? (y/n) : ")

    if (b=='y'):

        add\_item()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back use menu())")

def add\_delboy():

    print("add\_delboy() function started.")

    did = input('enter delboy ID  : ')

    name = input('enter delboy name : ')

    city = input('enter delboy city : ')

    mobile = input('enter delboy mobile number : ')

    salary = input('enter delboy salary : ')

    rating = input('enter the rating of delboy : ')

    data = (did,name,city,mobile,salary,rating)

    statement = "INSERT INTO delboy VALUES(%s,%s,%s,%s,%s,%s)"

    cursor.execute(statement,data)

    database.commit()

    print('data inserted to table delboy successfully!')

    b = input("do you want ot add more data to THIS table? (y/n) : ")

    if (b=='y'):

        add\_delboy()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back use menu())")

def add\_complain():

    print("add\_complain() function started.")

    cid = input("enter complain ID : ")

    oid = input("enter order id : ")

    sub = input("enter the subject of complain : ")

    data = (cid,oid,sub)

    statement = "INSERT INTO COMPLAINS VALUES(%s,%s,%s)"

    cursor.execute(statement,data)

    database.commit()

    print("data inserted to the table COMPLAINS successfully!")

    b = input("do you want ot add more data to THIS table? (y/n) : ")

    if (b=='y'):

        add\_complain()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. toa call back use menu())")

#DEFINING FUNCTIONS TO DELETE DATA FROM TABLES OF DATABASE :-

def delete\_customer():

    print("delete\_customer() function started.")

    cid = input('enter customer\_id : ')

    cursor.execute('DELETE FROM customers WHERE customer\_id=%s',(cid,))

    database.commit()

    print('deleted the customer data of customer\_id : ',cid,'from table customers successfully!')

    b = input("do you want ot delete more data to THIS table? (y/n) : ")

    if (b=='y'):

        delete\_customer()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back use menu())")

def delete\_seller():

    print("delete\_seller() function started.")

    sid = input('enter seller\_id : ')

    cursor.execute('DELETE FROM sellers WHERE seller\_id=%s',(sid,))

    database.commit()

    print('deleted the seller data of seller\_id : ',sid,' from table sellers successfully!')

    b = input("do you want ot delete more data to THIS table? (y/n) : ")

    if (b=='y'):

        delete\_seller()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back, use menu())")

def delete\_item():

    print("delete\_item() function started.")

    iid = input('enter item\_id : ')

    cursor.execute('DELETE FROM items WHERE item\_id=%s',(iid,))

    database.commit()

    print('deleted the item data of item\_id : ',iid,' from table items successfully!')

    b = input("do you want ot delete more data to THIS table? (y/n) : ")

    if (b=='y'):

        delete\_item()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back, use menu())")

def delete\_order():

    print("delete\_order() function started.")

    x = input('enter order\_id : ')

    cursor.execute('DELETE FROM orders WHERE order\_id=%s',(x,))

    database.commit()

    print('deleted the item data of order\_id : ',x,' from table orders successfully!')

    b = input("do you want ot delete more data to THIS table? (y/n) : ")

    if (b=='y'):

        delete\_order()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back, use menu())")

def delete\_delboy():

    print("delete\_delboy() function started.")

    x = input('enter delboy\_id : ')

    cursor.execute('DELETE FROM delboy WHERE delboy\_id=%s',(x,))

    database.commit()

    print('deleted the data of delboy\_id ',x,' from table delboy successfully!')

    b = input("do you want ot delete more data to THIS table? (y/n) : ")

    if (b=='y'):

        delete\_delboy()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back, use menu())")

def delete\_complain():

    print("delete\_complain() function started.")

    x = input('enter complain\_id : ')

    cursor.execute('DELETE FROM complains WHERE complain\_id=%s',(x,))

    database.commit()

    print('deleted the data of complain\_id ',x,' from table complains successfully!')

    b = input("do you want to delete more data to THIS table? (y/n) : ")

    if (b=='y'):

        delete\_complain()

    elif (b=='n'):

        menu()

    else:

        print("please enter a valid response. (you exited the menu. to call back, use menu())")

#FINALLY, SERVING THE PROGRAM TO USER : -

print("\* \* \* \* \* \* \* WELCOME TO THE DATABASE MANAGEMENT USER INTERFACE OF ESHOP \* \* \* \* \* \* \*")

print("(created by Mohammad Maasir @ date 13th Feb,2022, as a school project.)")

menu()

"""

although this program have a menu, we can also call the functions defined here to

add or delete data when this program is running in the IDLE shell, manually (when you are outside menu i.e. exit the menu.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* END OF THE CODE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"""

*Git Hub Page Link*

This project is available on my GitHub page. link:

<https://github.com/maasir554/maasir554/blob/main/frontend.py>

For viewing more projects made by me, refer to my GitHub page:

<https://github.com/maasir554/>

1. <https://www.mysqltutorial.org/>
2. <https://dev.mysql.com/doc/refman/5.6/en/>
3. <https://www.geeksforgeeks.org/>
4. <https://www.w3schools.com/sql/>
5. <https://www.w3schools.com/python/>
6. <https://app.diagrams.net/> (or draw.io , used for ER Diagram)
7. <https://www.youtube.com/c/CodeWithHarry>
8. <https://codewithharry.com/>
9. <https://www.youtube.com/c/TechWithTim>
10. <https://matplotlib.org/stable/tutorials/introductory/pyplot.html>
11. <https://code.visualstudio.com/>
12. NCERT textbooks class 11th and 12th : <https://ncert.nic.in/textbook.php>
13. Sumita Arora class 12th

BIBLIOGRAPHY

*(End of this document)*

Thank You

In this project, a SQL database was created, and connected to python successfully.

Various operations such as insertion of values, deletion of values and data analysis was done using Python-MySQL connector, and Matplotlib.

Conclusion