Experiment :

Date

Page No.

# PROJECT

Students are required to develop a DBMS for the application assigned to them. Following items are required for the project.

a) Problem statement
b) ER model
c) Integrity constraints implemented.
d) Suitable queries to create and manage.

The major headache for any enterprise engaged in trade is keeping its books and mainraining them. For larger companies, they can afford to hire professionals to check their balances but it is not possible for smaller enterprises. They require something easy to use, maintain and most importantly, it should be economical. Therefore, the only thing checking all these boxes is a database management system.

→ PROJECT TITLE : Stock Management System for Small Enterprises

DELTA®

Experiment :

Date

Page No.

→ REQUIREMENTS :

1) The database management system shall be used by a person with low technical knowledge and one who needs it to act efficiently and fast. Hence our queries shall be designed to be as helpful ass possible.

2) One item shall only be bought by one supplier.

3) One customer can have multiple purchases and hence, multiple invoices.

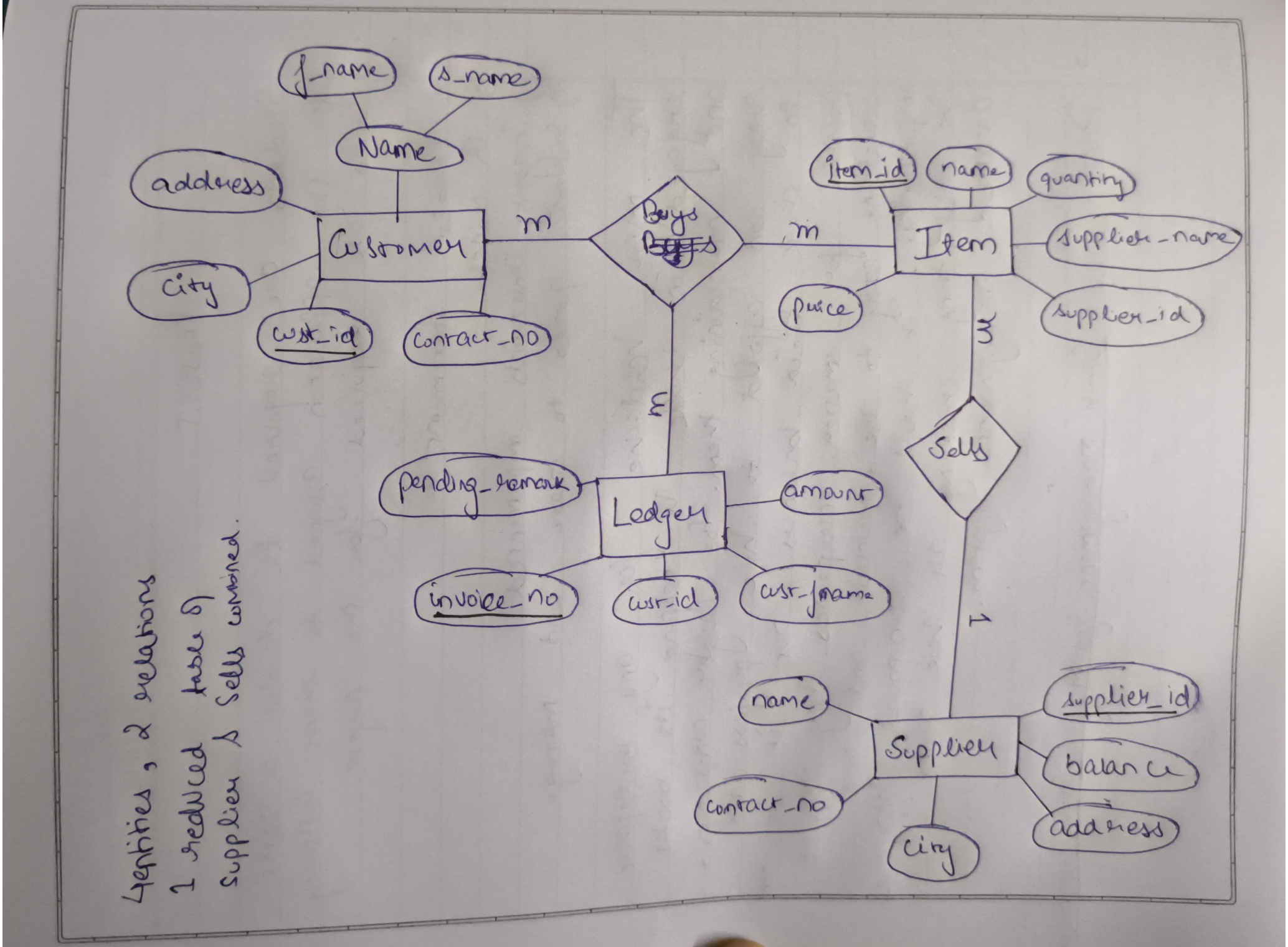
4) One invoice shall only be issued for one customer irrespective of no of purchases.

→ 4 TABLES | SCHEMA :

1) Entity → Customer

Attributes → cust_id (primary key), f_name, l_name, address, city, Contact_no

DELTA®




4 entities, 2 relations
1 reduced based on
Supplier & Sells combined.

Experiment :

Date

Page No.

2) Entity - items

Attributes - item_id (primary key), name, quantity, price, supplier_id, supplier_name.

3) Entity - suppliers

Attributes - supplier_id (primary key), name, contact_no, address, city, balance

4) Entity - ledger

Attributes - invoice_no (primary key), amount, cust_id, cust_name, pending_remarks

DELTA®

# customers

| cust_id | f_name | s_name | address | city | contact_no |
|---|---|---|---|---|---|
| 1000 | Akshit | Bhandari | Pushp Vihar | New Delhi | 8384521234 |
| 1001 | Aditya | Talwar | Thapaar | Ludhiana | 8384456712 |
| 1002 | Sam | Marcy | Khanpur | New Delhi | 7592810283 |
| 1003 | Tripti | Kriti | SNU | Greater Noida | 9482719203 |
| 1004 | Onkar | Jha | Deoli | New Delhi | 6281938264 |
| 1005 | Soubhagya | Kukreja | Lajpat Nagar | New Delhi | 8231801927 |
| 1006 | Siddhant | Yadav | Saket | New Delhi | 8383816844 |
| 1007 | Manav | Mittal | South Ex | New Delhi | 7482981723 |
| 1008 | Ram Kumar | Goyal | Sungroor | Patiala | 8391027321 |

# items

| item_id | name | quantity | price | supplier_id | supplier_name |
|---|---|---|---|---|---|
| 23469 | Combiflam | 35 | 30 | 3000 | DR Distributers Pvt Ltd |
| 23470 | Pantosec DSR | 40 | 100 | 3000 | DR Distributers Pvt Ltd |
| 23471 | iJal | 12 | 18 | 3003 | Anand Pharmaceuticals |
| 23472 | Dairy Milk Silk | 7 | 180 | 3004 | Cadbury India Ltd |
| 23473 | Maybelline Eyeliner | 200 | 4 | 3001 | Lucky Pharma |
| 23474 | Vaseline Jelly | 5 | 12 | 3001 | Lucky Pharma |
| 23475 | Asthakind DS | 80 | 25 | 3000 | DR Distributers Pvt Ltd |
| 23476 | Azax 500 | 124 | 40 | 3003 | Anand Pharmaceuticals |
| 23477 | Vicks | 94 | 20 | 3001 | Lucky Pharma |
| 23478 | Kinder Joy | 55 | 12 | 3002 | Nestle India Pvt Ltd |
| 23479 | Meftal Spas | 60 | 30 | 3000 | DR Distributers Pvt Ltd |

# supplier

| supplier_id | name | contact_no | address | city | balance |
|---|---|---|---|---|---|
| 3000 | DR Distributers Pvt Ltd | 8492018372 | Okhla Industrial Area | New Delhi | -18913 |
| 3001 | Lucky Pharma | 9102872312 | Jamia Milia Area | New Delhi | -912 |
| 3002 | Nestle India Pvt Ltd | 728193021 | Hisar | Hisar | 1213 |
| 3003 | Anand Pharmaceuticals | 9128391238 | Anupam Apartments | New Delhi | *NULL* |
| 3004 | Cadbury India Ltd | 8729102837 | Navi Mumbai | Mumbai | 7812 |

# ledger

| invoice_no | amount | cust_id | cust_fname | pending_remarks |
|---|---|---|---|---|
| 13400 | 341 | 1002 | Sam | *NULL* |
| 13401 | 12 | 1004 | Onkar | *NULL* |
| 13402 | 721 | 1005 | Soubhagya | Online payment declined. Try again. |
| 13403 | 1032 | 1001 | Aditya | *NULL* |
| 13404 | 523 | 1004 | Onkar | not delivered yet |
| 13405 | 4182 | 1007 | Manav | *NULL* |
| 13406 | 132 | 1003 | Tripti | *NULL* |
| 13407 | 13982 | 1006 | Siddhant | 50% Paid in advance |

→ QUERIES :

1) Get details of all payments that are pending from the ledgers entry. All entries in ledgers which dont have NULL in 'pending_remarks' attribute have an issue in payment and hence appear here.

2) Get details of items that are low on stock and need to be replenished. This query is handled by finding items with quantity less than equal to 5.

3) Checking which suppliers has credit pending. Ie, the supplier whose payment we are yet to make.

DELTA®

Experiment :

Date

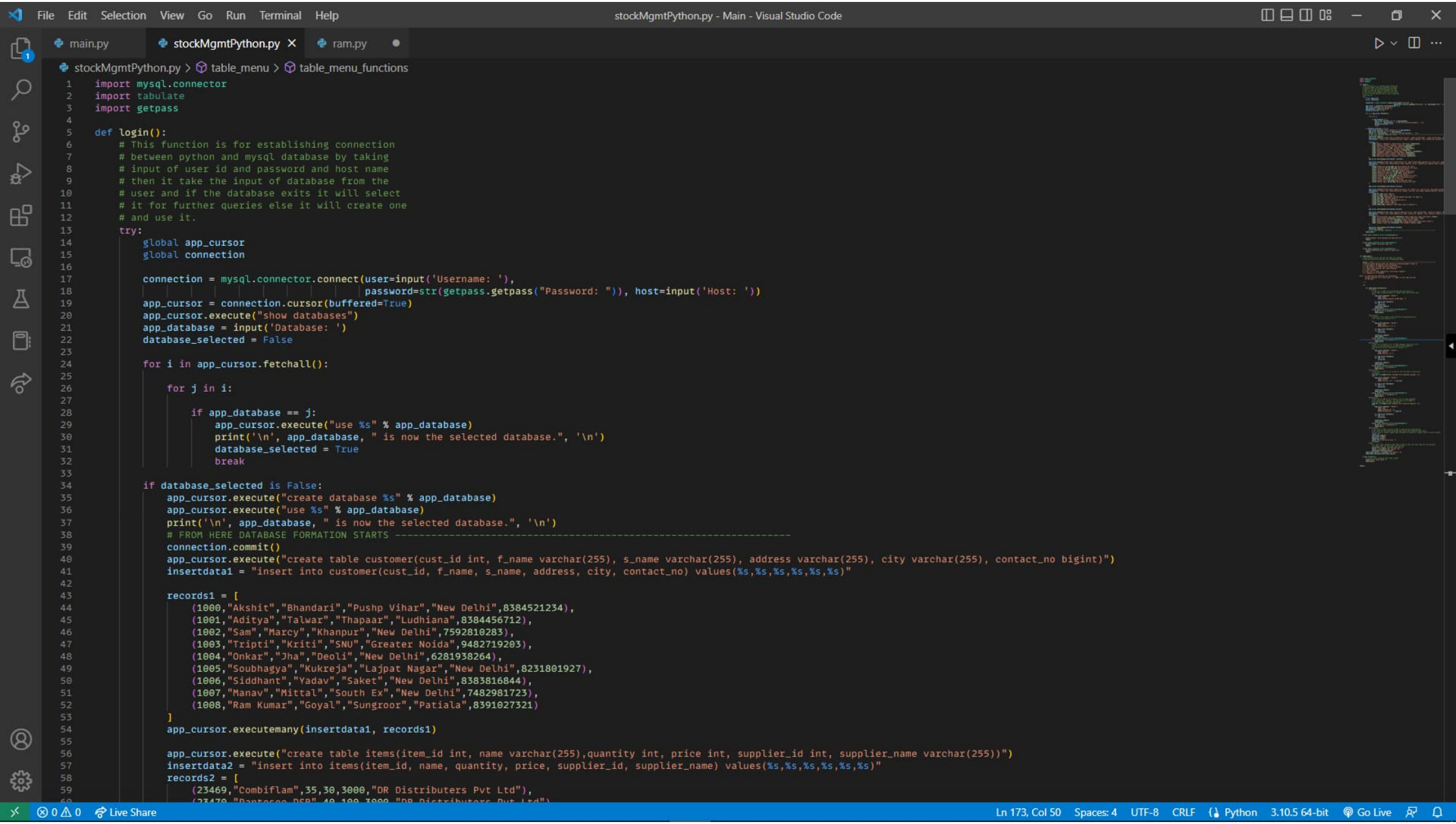
Page No.

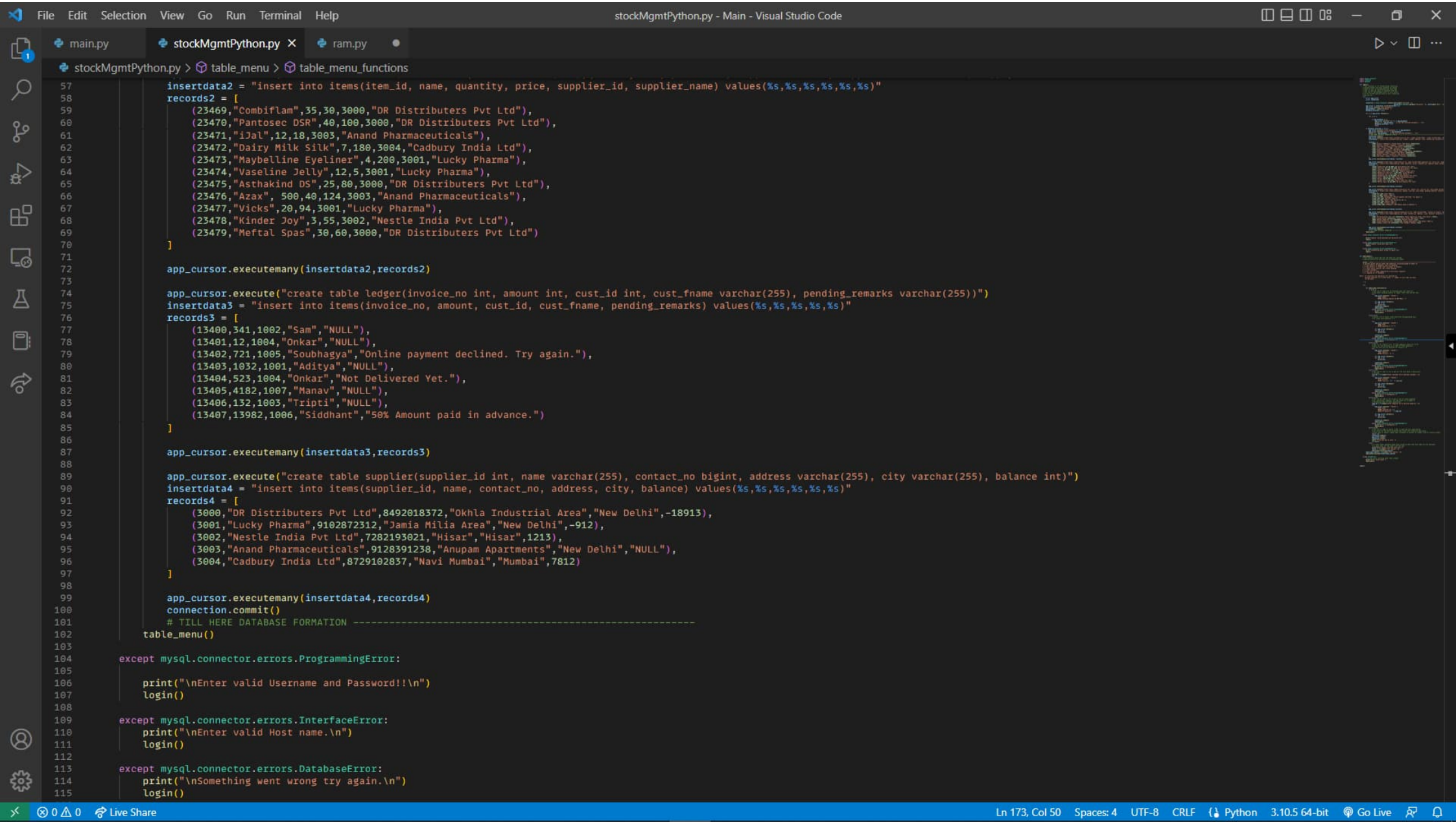
4) Finding all details about a customer. Frequently used to contact customers and process deliveries. They are identified by entered 'cus_id' by user.

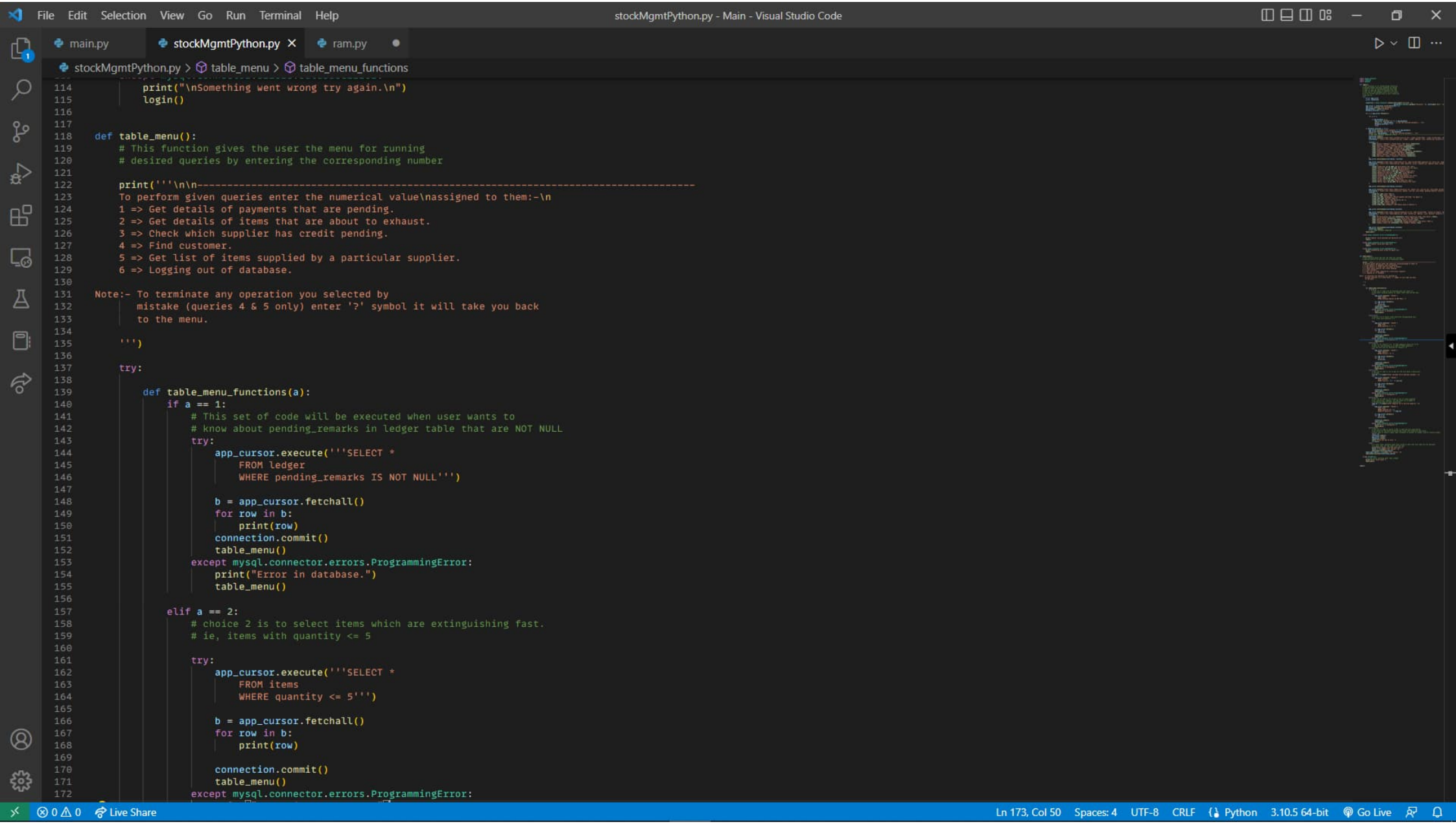
5) Getting list of items supplied by a particular supplier identified by the 'supp_id' entered by user in terminal. The list is ordered in increasing order by quantity of item left in stock. This helps in easier and faster ordering of stock.

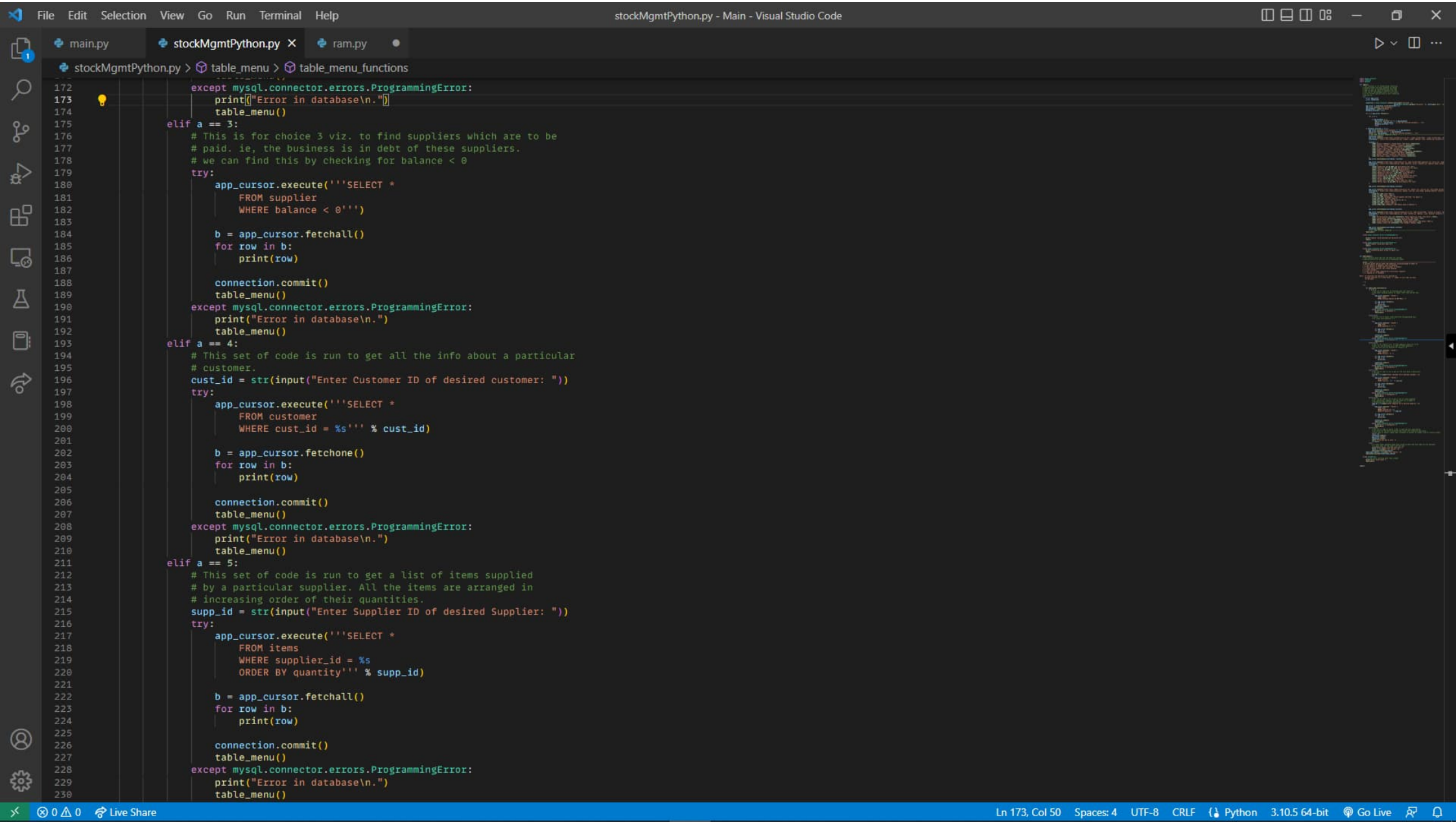
6) Logging out of the database.

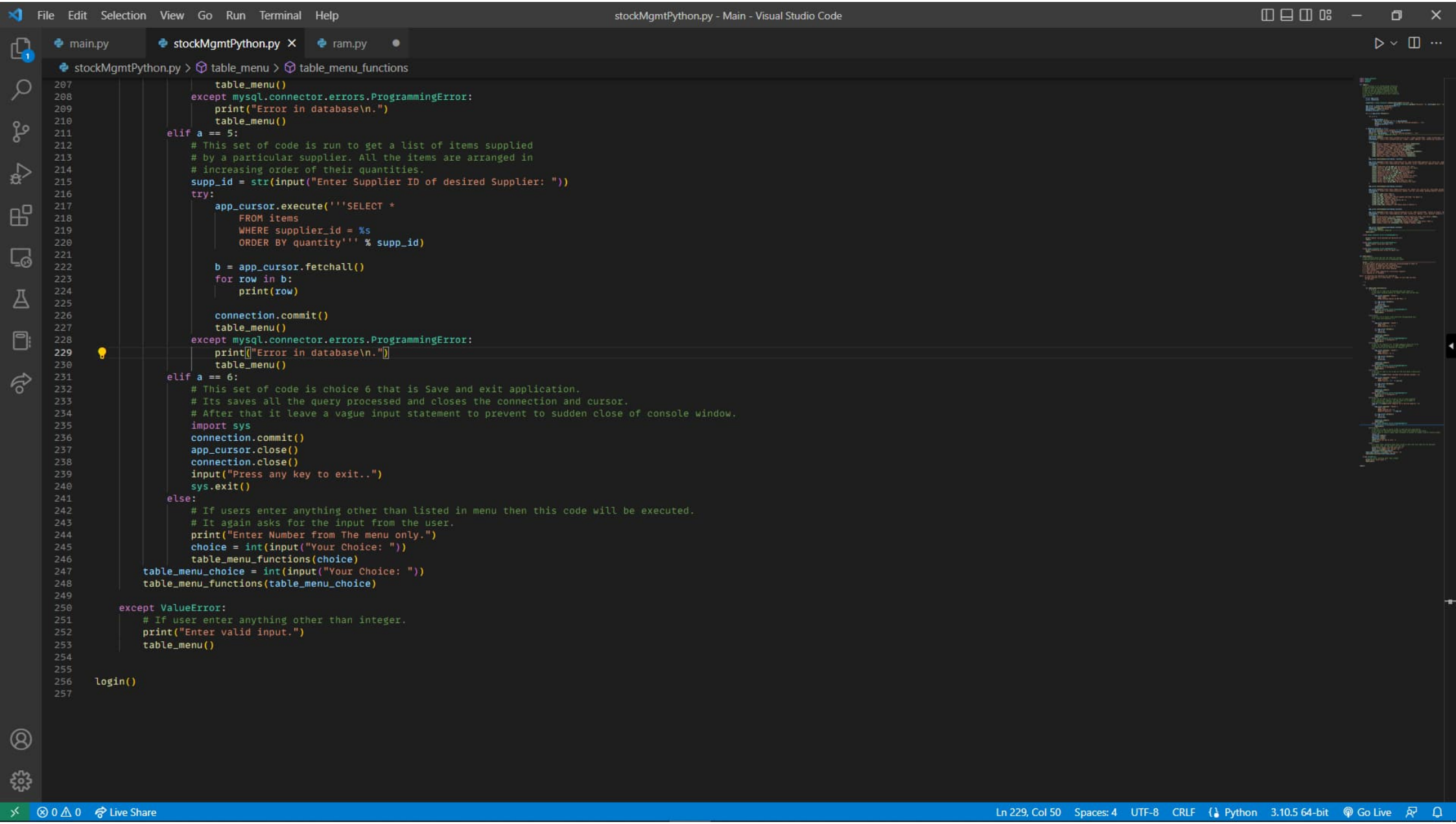
DELTA®

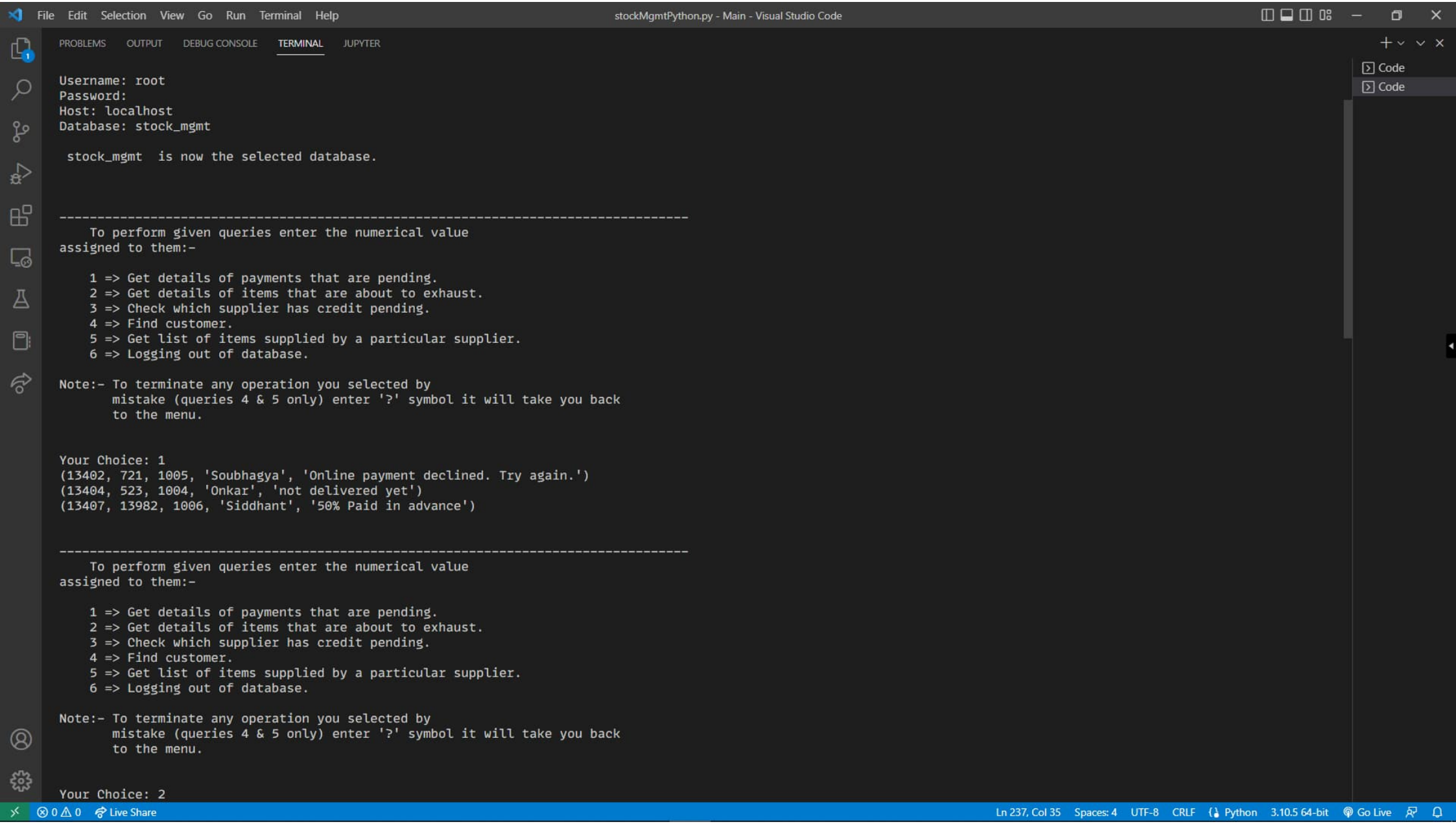

File Edit Selection View Go Run Terminal Help
stockMgmtPython.py - Main - Visual Studio Code
main.py
stockMgmtPython.py
ram.py
stockMgmtPython.py > table_menu > table_menu_functions
import mysql.connector
import tabulate
import getpass
def login():
# This function is for establishing connection
# between python and mysql database by taking
# input of user id and password and host name
# then it take the input of database from the
# user and if the database exits it will select
# it for further queries else it will create one
# and use it.
try:
global app_cursor
global connection
connection = mysql.connector.connect(user=input('Username: '),
password=str(getpass.getpass("Password: ")), host=input('Host: '))
app_cursor = connection.cursor(buffered=True)
app_cursor.execute("show databases")
app_database = input('Database: ')
database_selected = False
for i in app_cursor.fetchall():
for j in i:
if app_database == j:
app_cursor.execute("use %s" % app_database)
print('\n', app_database, " is now the selected database.", '\n')
database_selected = True
break
if database_selected is False:
app_cursor.execute("create database %s" % app_database)
app_cursor.execute("use %s" % app_database)
print('\n', app_database, " is now the selected database.", '\n')
# FROM HERE DATABASE FORMATION STARTS ----------
connection.commit()
app_cursor.execute("create table customer(cust_id int, f_name varchar(255), s_name varchar(255), address varchar(255), city varchar(255), contact_no bigint)")
insertdata1 = "insert into customer(cust_id, f_name, s_name, address, city, contact_no) values(%s,%s,%s,%s,%s,%s)"
records1 = [
(1000,"Akshit","Bhandari","Pushp Vihar","New Delhi",8384521234),
(1001,"Aditya","Talwar","Thapaar","Ludhiana",8384456712),
(1002,"Sam","Marcy","Khanpur","New Delhi",7592810283),
(1003,"Tripti","Kriti","SNU","Greater Noida",9482719203),
(1004,"Onkar","Jha","Deoli","New Delhi",6281938264),
(1005,"Soubhagya","Kukreja","Lajpat Nagar","New Delhi",8231801927),
(1006,"Siddhant","Yadav","Saket","New Delhi",8383816844),
(1007,"Manav","Mittal","South Ex","New Delhi",7482981723),
(1008,"Ram Kumar","Goyal","Sungroor","Patiala",8391027321)
]
app_cursor.executemany(insertdata1, records1)
app_cursor.execute("create table items(item_id int, name varchar(255),quantity int, price int, supplier_id int, supplier_name varchar(255))")
insertdata2 = "insert into items(item_id, name, quantity, price, supplier_id, supplier_name) values(%s,%s,%s,%s,%s,%s)"
records2 = [
(23469,"Combiflam",35,30,3000,"DR Distributers Pvt Ltd"),
Live Share
Ln 173, Col 50
Spaces: 4
UTF-8
CRLF
Python
3.10.5 64-bit
Go Live

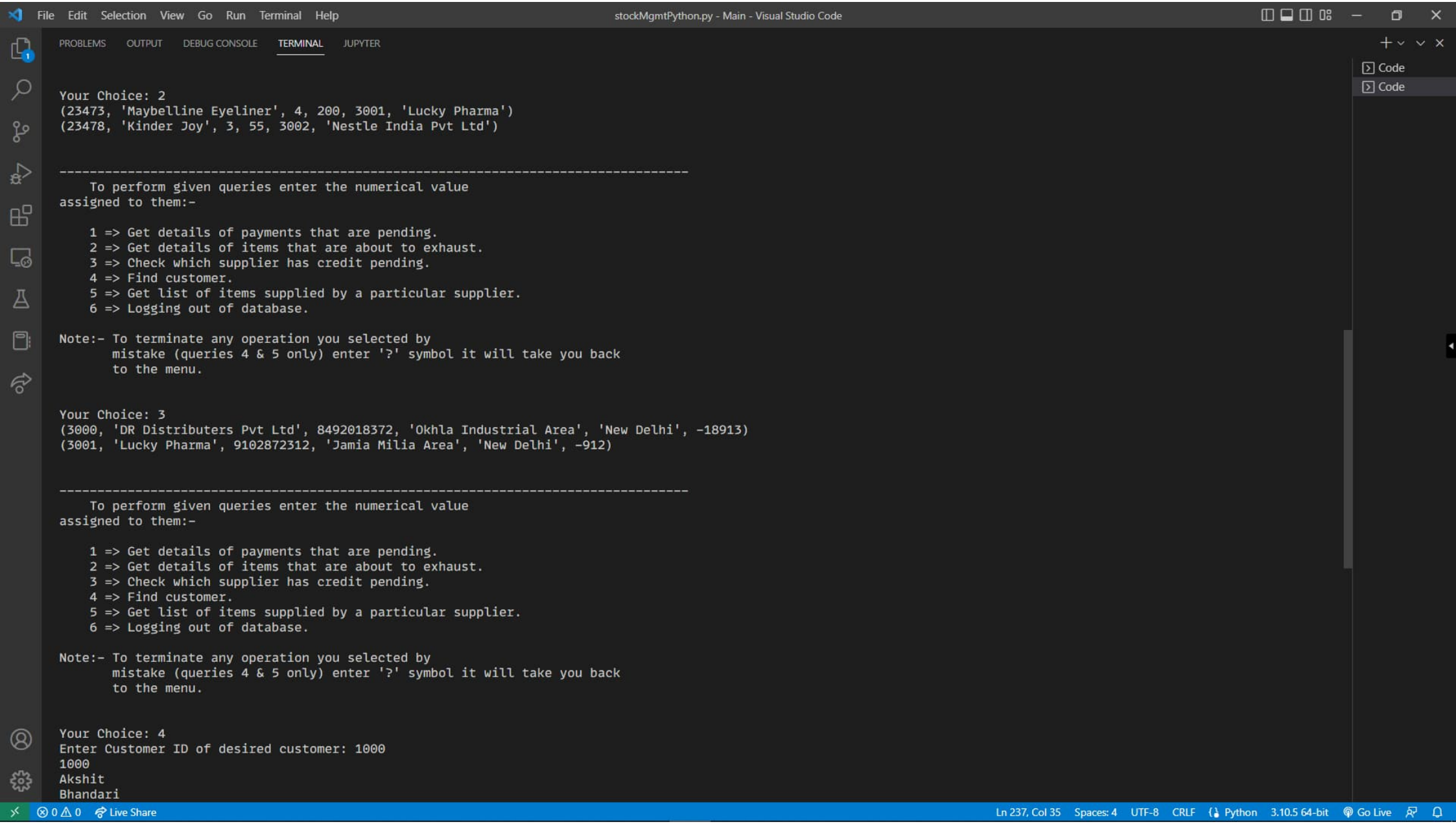

File Edit Selection View Go Run Terminal Help
stockMgmtPython.py - Main - Visual Studio Code
main.py
stockMgmtPython.py
ram.py
stockMgmtPython.py > table_menu > table_menu_functions
insertdata2 = "insert into items(item_id, name, quantity, price, supplier_id, supplier_name) values(%s,%s,%s,%s,%s,%s)"
records2 = [
(23469,"Combiflam",35,30,3000,"DR Distributers Pvt Ltd"),
(23470,"Pantosec DSR",40,100,3000,"DR Distributers Pvt Ltd"),
(23471,"iJal",12,18,3003,"Anand Pharmaceuticals"),
(23472,"Dairy Milk Silk",7,180,3004,"Cadbury India Ltd"),
(23473,"Maybelline Eyeliner",4,200,3001,"Lucky Pharma"),
(23474,"Vaseline Jelly",12,5,3001,"Lucky Pharma"),
(23475,"Asthakind DS",25,80,3000,"DR Distributers Pvt Ltd"),
(23476,"Azax", 500,40,124,3003,"Anand Pharmaceuticals"),
(23477,"Vicks",20,94,3001,"Lucky Pharma"),
(23478,"Kinder Joy",3,55,3002,"Nestle India Pvt Ltd"),
(23479,"Meftal Spas",30,60,3000,"DR Distributers Pvt Ltd")
]
app_cursor.executemany(insertdata2,records2)
app_cursor.execute("create table ledger(invoice_no int, amount int, cust_id int, cust_fname varchar(255), pending_remarks varchar(255))")
insertdata3 = "insert into items(invoice_no, amount, cust_id, cust_fname, pending_remarks) values(%s,%s,%s,%s,%s)"
records3 = [
(13400,341,1002,"Sam","NULL"),
(13401,12,1004,"Onkar","NULL"),
(13402,721,1005,"Soubhagya","Online payment declined. Try again."),
(13403,1032,1001,"Aditya","NULL"),
(13404,523,1004,"Onkar","Not Delivered Yet."),
(13405,4182,1007,"Manav","NULL"),
(13406,132,1003,"Tripti","NULL"),
(13407,13982,1006,"Siddhant","50% Amount paid in advance.")
]
app_cursor.executemany(insertdata3,records3)
app_cursor.execute("create table supplier(supplier_id int, name varchar(255), contact_no bigint, address varchar(255), city varchar(255), balance int)")
insertdata4 = "insert into items(supplier_id, name, contact_no, address, city, balance) values(%s,%s,%s,%s,%s,%s)"
records4 = [
(3000,"DR Distributers Pvt Ltd",8492018372,"Okhla Industrial Area","New Delhi",-18913),
(3001,"Lucky Pharma",9102872312,"Jamia Milia Area","New Delhi",-912),
(3002,"Nestle India Pvt Ltd",7282193021,"Hisar","Hisar",1213),
(3003,"Anand Pharmaceuticals",9128391238,"Anupam Apartments","New Delhi","NULL"),
(3004,"Cadbury India Ltd",8729102837,"Navi Mumbai","Mumbai",7812)
]
app_cursor.executemany(insertdata4,records4)
connection.commit()
# TILL HERE DATABASE FORMATION ----------
table_menu()
except mysql.connector.errors.ProgrammingError:
print("\nEnter valid Username and Password!!\n")
login()
except mysql.connector.errors.InterfaceError:
print("\nEnter valid Host name.\n")
login()
except mysql.connector.errors.DatabaseError:
print("\nSomething went wrong try again.\n")
login()
0 0 Live Share
Ln 173, Col 50 Spaces: 4 UTF-8 CRLF Python 3.10.5 64-bit Go Live

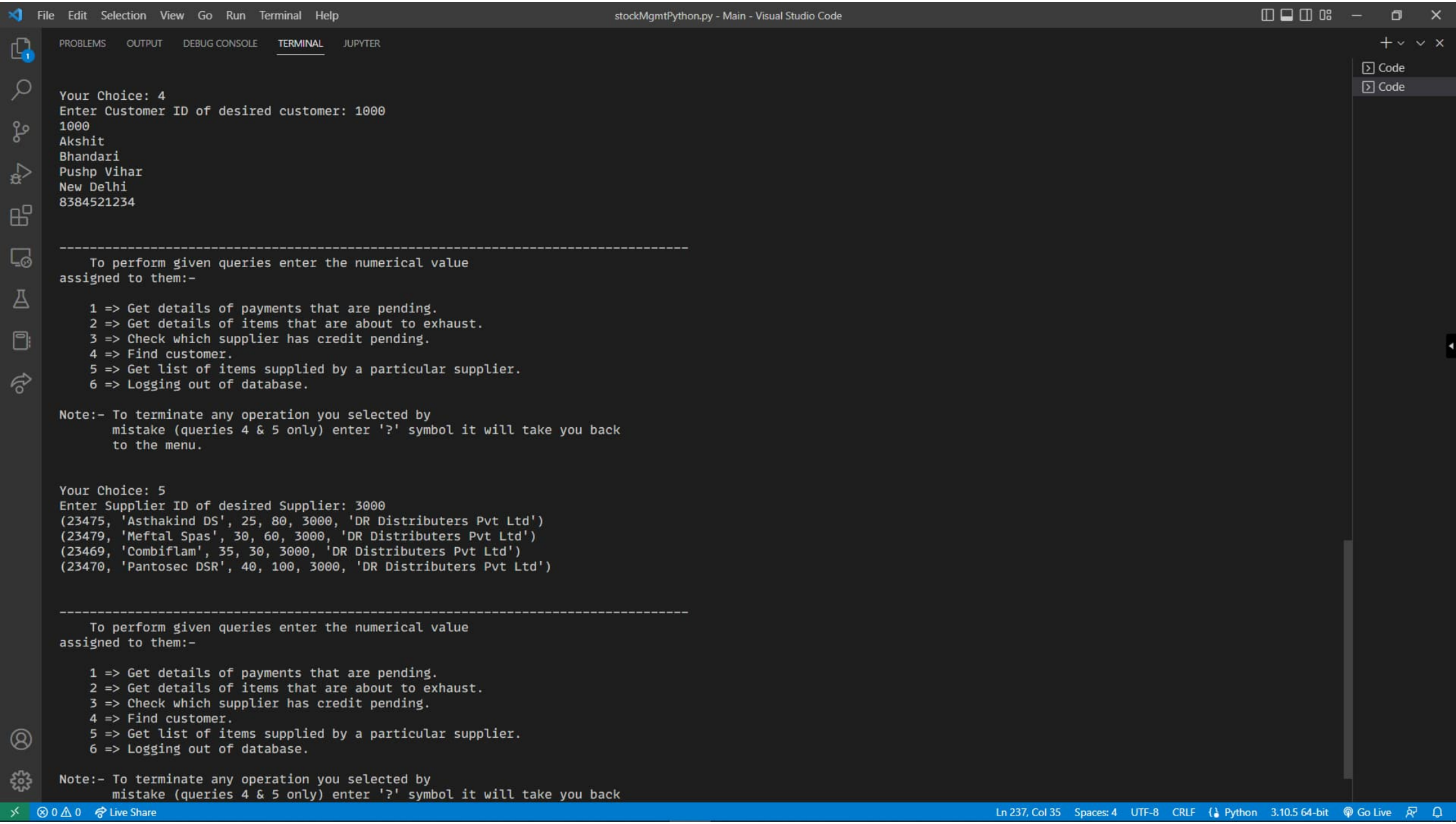

File Edit Selection View Go Run Terminal Help
stockMgmtPython.py - Main - Visual Studio Code
main.py
stockMgmtPython.py
ram.py
stockMgmtPython.py > table_menu > table_menu_functions
print("\nSomething went wrong try again.\n")
login()
def table_menu():
# This function gives the user the menu for running
# desired queries by entering the corresponding number
print('''\n\n----------
To perform given queries enter the numerical value\nassigned to them:-\n
1 => Get details of payments that are pending.
2 => Get details of items that are about to exhaust.
3 => Check which supplier has credit pending.
4 => Find customer.
5 => Get list of items supplied by a particular supplier.
6 => Logging out of database.
Note:- To terminate any operation you selected by
mistake (queries 4 & 5 only) enter '?' symbol it will take you back
to the menu.
''')
try:
def table_menu_functions(a):
if a == 1:
# This set of code will be executed when user wants to
# know about pending_remarks in ledger table that are NOT NULL
try:
app_cursor.execute('''SELECT *
FROM ledger
WHERE pending_remarks IS NOT NULL''')
b = app_cursor.fetchall()
for row in b:
print(row)
connection.commit()
table_menu()
except mysql.connector.errors.ProgrammingError:
print("Error in database.")
table_menu()
elif a == 2:
# choice 2 is to select items which are extinguishing fast.
# ie, items with quantity <= 5
try:
app_cursor.execute('''SELECT *
FROM items
WHERE quantity <= 5''')
b = app_cursor.fetchall()
for row in b:
print(row)
connection.commit()
table_menu()
except mysql.connector.errors.ProgrammingError:
Live Share
Ln 173, Col 50
Spaces: 4
UTF-8
CRLF
Python
3.10.5 64-bit
Go Live


File Edit Selection View Go Run Terminal Help
stockMgmtPython.py - Main - Visual Studio Code
main.py
stockMgmtPython.py
ram.py
stockMgmtPython.py > table_menu > table_menu_functions
except mysql.connector.errors.ProgrammingError:
print("Error in database\n.")
table_menu()
elif a == 3:
# This is for choice 3 viz. to find suppliers which are to be
# paid. ie, the business is in debt of these suppliers.
# we can find this by checking for balance < 0
try:
app_cursor.execute('''SELECT *
FROM supplier
WHERE balance < 0''')
b = app_cursor.fetchall()
for row in b:
print(row)
connection.commit()
table_menu()
except mysql.connector.errors.ProgrammingError:
print("Error in database\n.")
table_menu()
elif a == 4:
# This set of code is run to get all the info about a particular
# customer.
cust_id = str(input("Enter Customer ID of desired customer: "))
try:
app_cursor.execute('''SELECT *
FROM customer
WHERE cust_id = %s''' % cust_id)
b = app_cursor.fetchone()
for row in b:
print(row)
connection.commit()
table_menu()
except mysql.connector.errors.ProgrammingError:
print("Error in database\n.")
table_menu()
elif a == 5:
# This set of code is run to get a list of items supplied
# by a particular supplier. All the items are arranged in
# increasing order of their quantities.
supp_id = str(input("Enter Supplier ID of desired Supplier: "))
try:
app_cursor.execute('''SELECT *
FROM items
WHERE supplier_id = %s
ORDER BY quantity''' % supp_id)
b = app_cursor.fetchall()
for row in b:
print(row)
connection.commit()
table_menu()
except mysql.connector.errors.ProgrammingError:
print("Error in database\n.")
table_menu()
Live Share
Ln 173, Col 50
Spaces: 4
UTF-8
CRLF
Python
3.10.5 64-bit
Go Live


File Edit Selection View Go Run Terminal Help
stockMgmtPython.py - Main - Visual Studio Code
main.py
stockMgmtPython.py
ram.py
stockMgmtPython.py > table_menu > table_menu_functions
table_menu()
except mysql.connector.errors.ProgrammingError:
print("Error in database\n.")
table_menu()
elif a == 5:
# This set of code is run to get a list of items supplied
# by a particular supplier. All the items are arranged in
# increasing order of their quantities.
supp_id = str(input("Enter Supplier ID of desired Supplier: "))
try:
app_cursor.execute('''SELECT *
FROM items
WHERE supplier_id = %s
ORDER BY quantity''' % supp_id)
b = app_cursor.fetchall()
for row in b:
print(row)
connection.commit()
table_menu()
except mysql.connector.errors.ProgrammingError:
print("Error in database\n.")
table_menu()
elif a == 6:
# This set of code is choice 6 that is Save and exit application.
# Its saves all the query processed and closes the connection and cursor.
# After that it leave a vague input statement to prevent to sudden close of console window.
import sys
connection.commit()
app_cursor.close()
connection.close()
input("Press any key to exit..")
sys.exit()
else:
# If users enter anything other than listed in menu then this code will be executed.
# It again asks for the input from the user.
print("Enter Number from The menu only.")
choice = int(input("Your Choice: "))
table_menu_functions(choice)
table_menu_choice = int(input("Your Choice: "))
table_menu_functions(table_menu_choice)
except ValueError:
# If user enter anything other than integer.
print("Enter valid input.")
table_menu()
login()
Live Share
Ln 229, Col 50
Spaces: 4
UTF-8
CRLF
Python
3.10.5 64-bit
Go Live


File Edit Selection View Go Run Terminal Help
stockMgmtPython.py - Main - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Code
Code
Username: root
Password:
Host: localhost
Database: stock_mgmt
stock_mgmt is now the selected database.
To perform given queries enter the numerical value
assigned to them:-
1 => Get details of payments that are pending.
2 => Get details of items that are about to exhaust.
3 => Check which supplier has credit pending.
4 => Find customer.
5 => Get list of items supplied by a particular supplier.
6 => Logging out of database.
Note:- To terminate any operation you selected by
mistake (queries 4 & 5 only) enter '?' symbol it will take you back
to the menu.
Your Choice: 1
(13402, 721, 1005, 'Soubhagya', 'Online payment declined. Try again.')
(13404, 523, 1004, 'Onkar', 'not delivered yet')
(13407, 13982, 1006, 'Siddhant', '50% Paid in advance')
To perform given queries enter the numerical value
assigned to them:-
1 => Get details of payments that are pending.
2 => Get details of items that are about to exhaust.
3 => Check which supplier has credit pending.
4 => Find customer.
5 => Get list of items supplied by a particular supplier.
6 => Logging out of database.
Note:- To terminate any operation you selected by
mistake (queries 4 & 5 only) enter '?' symbol it will take you back
to the menu.
Your Choice: 2
0 0 Live Share
Ln 237, Col 35 Spaces: 4 UTF-8 CRLF Python 3.10.5 64-bit Go Live


File Edit Selection View Go Run Terminal Help
stockMgmtPython.py - Main - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Code
Code
Your Choice: 2
(23473, 'Maybelline Eyeliner', 4, 200, 3001, 'Lucky Pharma')
(23478, 'Kinder Joy', 3, 55, 3002, 'Nestle India Pvt Ltd')
To perform given queries enter the numerical value
assigned to them:-
1 => Get details of payments that are pending.
2 => Get details of items that are about to exhaust.
3 => Check which supplier has credit pending.
4 => Find customer.
5 => Get list of items supplied by a particular supplier.
6 => Logging out of database.
Note:- To terminate any operation you selected by
mistake (queries 4 & 5 only) enter '?' symbol it will take you back
to the menu.
Your Choice: 3
(3000, 'DR Distributers Pvt Ltd', 8492018372, 'Okhla Industrial Area', 'New Delhi', -18913)
(3001, 'Lucky Pharma', 9102872312, 'Jamia Milia Area', 'New Delhi', -912)
To perform given queries enter the numerical value
assigned to them:-
1 => Get details of payments that are pending.
2 => Get details of items that are about to exhaust.
3 => Check which supplier has credit pending.
4 => Find customer.
5 => Get list of items supplied by a particular supplier.
6 => Logging out of database.
Note:- To terminate any operation you selected by
mistake (queries 4 & 5 only) enter '?' symbol it will take you back
to the menu.
Your Choice: 4
Enter Customer ID of desired customer: 1000
1000
Akshit
Bhandari
Live Share
Ln 237, Col 35 Spaces: 4 UTF-8 CRLF Python 3.10.5 64-bit Go Live


File Edit Selection View Go Run Terminal Help
stockMgmtPython.py - Main - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Code
Code
Your Choice: 4
Enter Customer ID of desired customer: 1000
1000
Akshit
Bhandari
Pushp Vihar
New Delhi
8384521234
To perform given queries enter the numerical value
assigned to them:-
1 => Get details of payments that are pending.
2 => Get details of items that are about to exhaust.
3 => Check which supplier has credit pending.
4 => Find customer.
5 => Get list of items supplied by a particular supplier.
6 => Logging out of database.
Note:- To terminate any operation you selected by
mistake (queries 4 & 5 only) enter '?' symbol it will take you back
to the menu.
Your Choice: 5
Enter Supplier ID of desired Supplier: 3000
(23475, 'Asthakind DS', 25, 80, 3000, 'DR Distributers Pvt Ltd')
(23479, 'Meftal Spas', 30, 60, 3000, 'DR Distributers Pvt Ltd')
(23469, 'Combiflam', 35, 30, 3000, 'DR Distributers Pvt Ltd')
(23470, 'Pantosec DSR', 40, 100, 3000, 'DR Distributers Pvt Ltd')
To perform given queries enter the numerical value
assigned to them:-
1 => Get details of payments that are pending.
2 => Get details of items that are about to exhaust.
3 => Check which supplier has credit pending.
4 => Find customer.
5 => Get list of items supplied by a particular supplier.
6 => Logging out of database.
Note:- To terminate any operation you selected by
mistake (queries 4 & 5 only) enter '?' symbol it will take you back
Live Share
Ln 237, Col 35 Spaces: 4 UTF-8 CRLF Python 3.10.5 64-bit Go Live