

Index

|  |  |  |
| --- | --- | --- |
| *Sr.No.* | *Name* | *Page No.* |
| 1 | Index | 1-1 |
| 2 | Certificate | 2-2 |
| 3 | Acknowledgement | 3-3 |
| 4 | Introduction | 4-4 |
| 5 | System Requirements | 5-5 |
| 6 | Database Design | 6-6 |
| 7 | Modules/Function in the project | 7-7 |
| 8 | Code | 8-19 |
| 9 | Screenshots | 20-21 |
| 10 | Conclusion | 22-22 |
| 11 | Bibliography | 23-23 |

Acknowledgement

We would like to express our deepest gratitude to our teacher Malvika ma’am for her guidance and support as well as our principal Surendra Sachdeva who gave us this wonderful opportunity to work on this amazing project, which helped in increasing our knowledge about various subject.

We would also like to thank the people who indirectly helped us in the completion of this project within this limited amount of time and providing us support whenever needed.

Introduction

Our goal of this project is to create a menu driven program with which even people with little to no knowledge about managing an inventory and all the essential tasks with it are able to use our code to swiftly and efficiently to complete their desired tasks.

This project ‘ASTER’ is very useful in real life situation and is mainly intended for the various activities that are essential for an inventory management system (like a bakery). It stores the information about product’s ID, Name, Quantity, Cost price and Sell price in a MySQL table. It also stores the total profit earned which can be viewed later. In this SQL-Python project we can see the inventory items, purchase items, replenish the inventory and also see the total profit.

For our project we used bakery as an example to show people that how easy it is to create and operate an effective inventory management system which has all the basic functionalities an inventory management system must have.

System Requirements

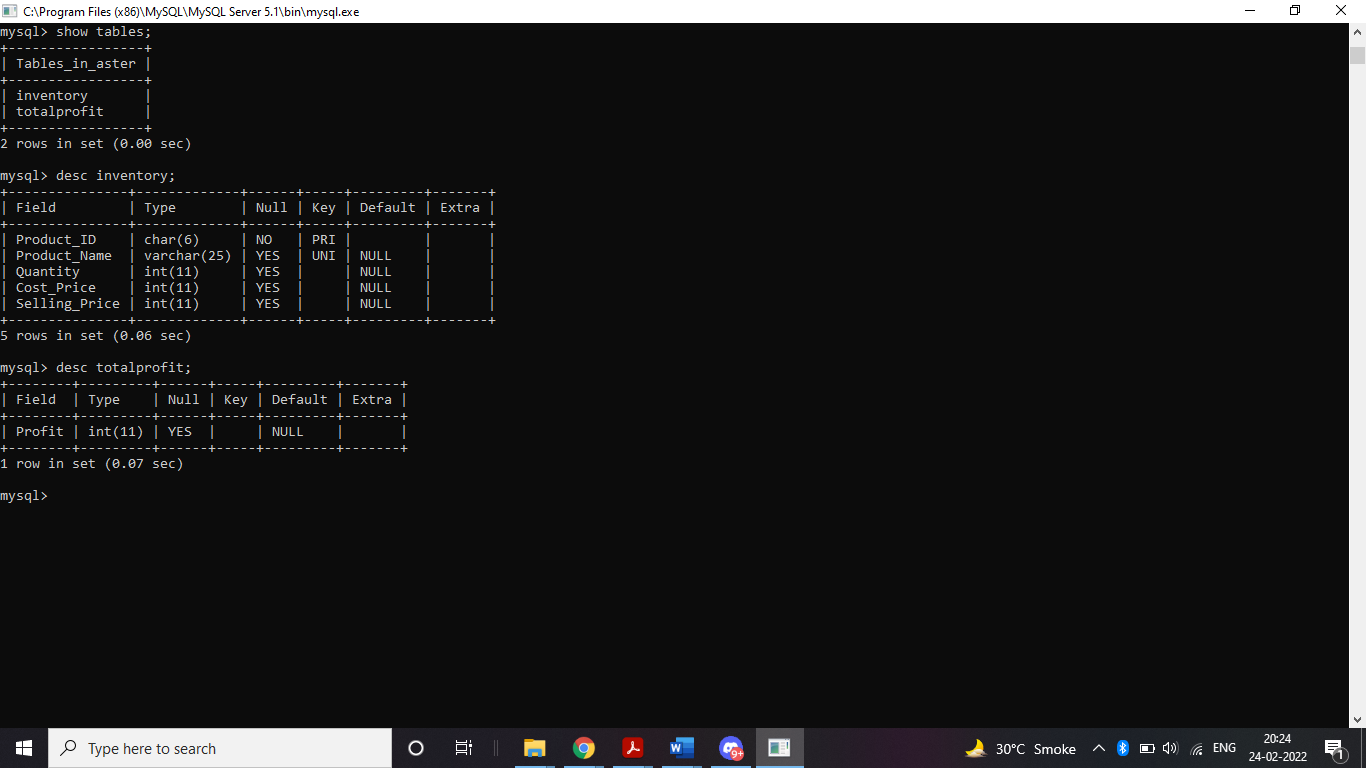
Latest version of python software recommended (3.9.1), it can be used on macOS and Linux but cannot be used on Windows version 7 or earlier (later versions are fine). Disk space should at least be 1Gb but 2-3 Gb is recommended. Processor should be Intel Atom® processor or Intel® Core™ i3 processor. For a smooth code run RAM should be a 4Gb one.

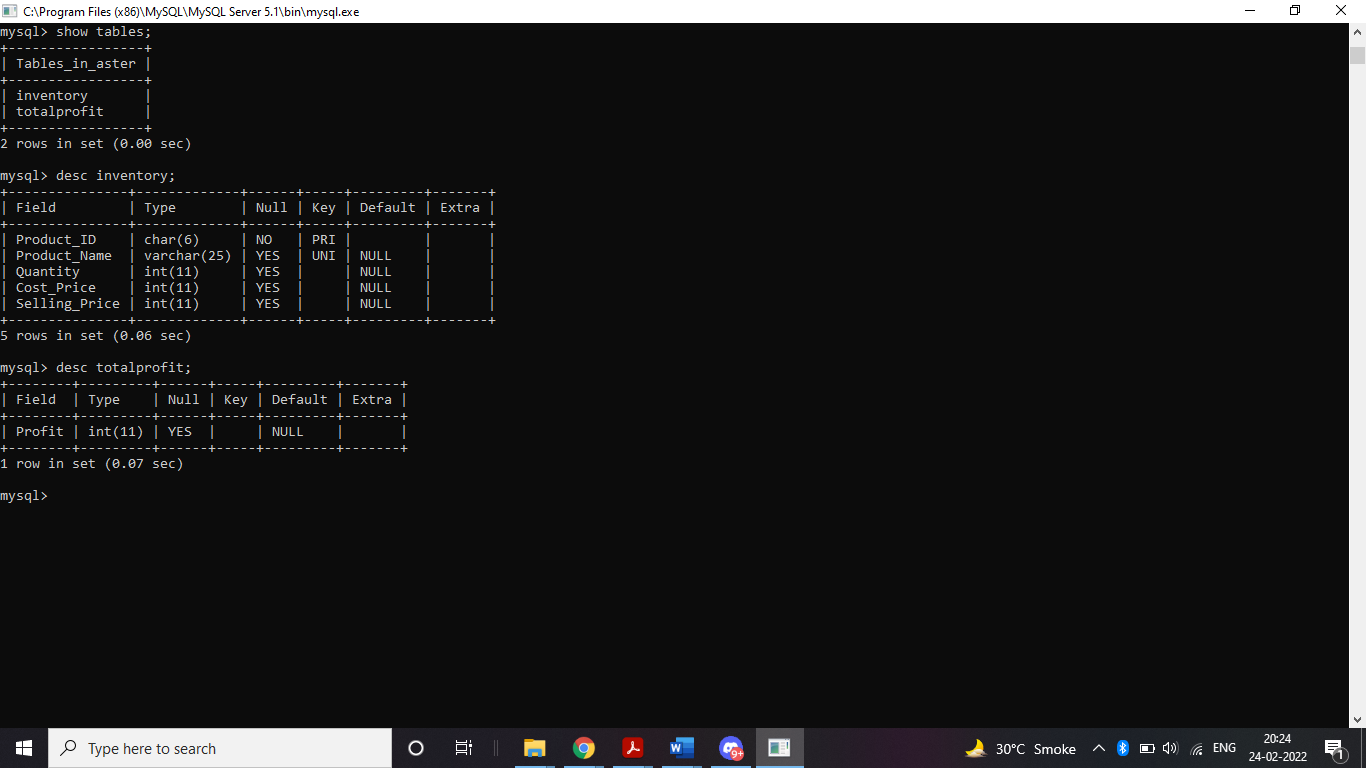
Also, a python simulator like Spyder, PyCharm, Visual Studio Code, etc is necessary to run our code. MySQL must be installed and present with a database named ‘ASTER’, within the database there should be 2 tables named ‘inventory’ and ‘totalProfit’ with the required data entered in those tables. MySQL-Python connector must be installed.

Operators used are: –

* Arithmetic operators – such as (+, -, \*) to add, subtract or multiply variables such as cost price, selling price, quantity to get the desired output
* Comparison operators – such as (==, !=) to compare variables to each other like current quantity, selling quantity to get the desired output
* Assignment operators – such as (=, +=, \*=) to define, add or multiply variables like total price, final price to itself and others in a more systematic format
* Logical operators – such as (and, or) to and more conditions to an if statement to get the desired output

Database Design





Modules/Functions in Project

Modules: -

* MySQL Connector - It links existing Python Program with SQL database

Functions: -

* Help – Shows the list of commands which are executable
* See - It projects the entire inventory in python console window in tabular format
* Purchase - User can purchase products by inputting the product code given in the inventory
* Replenish - This Replenishes products which are either out of stock or lower then set desired number
* Main - The main function is the starting point of the program and breaks the program into logical parts. It also shows bakery profit

Code

def Help(): #menu to help user perform tasks

print()

print("S - See stock")

print("P - Purchase")

print("R - Replenish inventory ")

print("B - Show bakery profit")

print("Q - Quit")

print()

def See(): #view the inventory in tabular format

lines()

print("ASTER inventory - ")

print()

import mysql.connector as mysql

mydb = mysql.connect(host="localhost", user="root", \

password="adwaiy2912", database="ASTER")

mycursor = mydb.cursor()

mycursor.execute("select \* from inventory;")

print ("{:<9} {:<24} {:<7} {:<7} {:<7}".format('ID','Name', 'Qty', 'CP', 'SP'))

for record in mycursor:

ID, Name, Qty, CP, SP = record[0], record[1], record[2], record[3], record[4]

print ("{:<9} {:<24} {:<7} {:<7} {:<7}".format(ID, Name, Qty, CP, SP))

#display the content of the inventory in evenly seperated rows and columns

lines()

def Purchase(): #make a purchase from the inventory

bill={}

totalPrice = 0

import mysql.connector as mysql

mydb = mysql.connect(host="localhost", user="root", \

password="adwaiy2912", database="ASTER")

mycursor = mydb.cursor()

mycursor.execute("select \* from inventory;")

inventory = mycursor.fetchall()

Exit1 = False

while (Exit1 == False):

itemID = input("Enter item ID: ")

x,y,z = itemID[0:1], itemID[1:3], itemID[3:6]

itemID = x.upper() + y.lower() + z #confims that itemid is in correct format

flag = False

for record in range(len(inventory)):

if (itemID in inventory[record][0]):

Exit2 = False

while (Exit2 == False):

item = inventory[record]

print()

print(item[1], "is available for ₹", item[4], "each")

itemQty = int(input("Enter item Quantity: "))

if (itemQty <= item[2] and itemQty > 0):

Exit2 = True

ID = (itemQty,item[0])

mycursor.execute("update inventory set quantity = quantity - %s \

where product\_id = %s",ID)

mydb.commit()

itemTotal = itemQty \* item[4]

totalPrice += itemTotal

if (itemID not in bill):

bill[itemID] = {}

bill[itemID]["Name"] = item[1]

bill[itemID]["Qty"] = itemQty

bill[itemID]["IP"] = item[4]

bill[itemID]["SubT"] = itemQty \* item[4]

elif bill[itemID]["Qty"] + itemQty > item[2]:

print()

print("Total item quantity exceeds inventory limit. \

Thus transaction declined")

ID = (itemQty,item[0])

mycursor.execute("update inventory set quantity = quantity + %s \

where product\_id = %s",ID)

mydb.commit()

else:

bill[itemID]["Qty"] += itemQty

bill[itemID]["SubT"] += itemQty \* item[4]

print()

print ("{:<23} {:<7} {:<7} {:<7}".format('Name', 'Qty', 'IP', 'SubTotal'))

for key, val in bill.items():

Name, Qty, IP, SubT = val["Name"], val["Qty"], val["IP"], val["SubT"]

print ("{:<23} {:<7} {:<7} {:<7}".format(Name, Qty, IP, SubT))

print()

print("Total (not including taxes) = ₹", totalPrice)

#prints bill in tabular format

Exit3 = False

while (Exit3 == False):

print()

print("Press A to Add more items")

print("Press C to Checkout")

print()

choice = input("Enter your choice: ")

choice = choice.upper()

if (choice == "A"):

Exit3 = True

flag = True

break

elif (choice == "C"):

Exit1 = True

Exit3 = True

GST = totalPrice \* 5/100

finalPrice = totalPrice + 2 \* GST

print()

print("CGST = ₹", GST)

print("SGST = ₹", GST)

print()

print("Final total (including tax) = ₹", finalPrice)

print()

print("Thank you for shopping with us")

lines()

return totalPrice

else:

print("Invalid input. Try again")

elif (itemQty != 0 and item[2] != 0):

print()

print("Sorry. We only have", item[2], item[1], "available")

print()

elif (itemQty <= 0):

print()

print("Error. Invalid input")

print()

else:

Exit2 = True

print()

print("Sorry. We are out of stock")

print()

else:

pass

else:

if flag == True:

pass

else:

print("Invalid item ID. Try again")

def Replenish(): #fill items to its default value

amount = 0

import mysql.connector as mysql

mydb = mysql.connect(host="localhost", user="root", \

password="adwaiy2912", database="ASTER")

mycursor = mydb.cursor()

mycursor.execute("select \* from inventory;")

inventory = mycursor.fetchall()

for item in inventory:

currQty = item[2]

if (item[0][0:3] == "Bis" and currQty != 50):

soldQty = 50 - currQty

amount += item[3] \* soldQty

elif (item[0][0:3] == "Bre" and currQty != 30):

soldQty = 30 - currQty

amount += item[3] \* soldQty

elif (item[0][0:3] == "Bro" and currQty != 25):

soldQty = 25 - currQty

amount += item[3] \* soldQty

elif (item[0][0:3] == "Cak" and currQty != 15):

soldQty = 15 - currQty

amount += item[3] \* soldQty

elif (item[0][0:3] == "Coo" and currQty != 60):

soldQty = 60 - currQty

amount += item[3] \* soldQty

elif (item[0][0:3] == "Don" and currQty != 40):

soldQty = 40 - currQty

amount += item[3] \* soldQty

elif (item[0][0:3] == "Rol" and currQty != 20):

soldQty = 20 - currQty

amount += item[3] \* soldQty

if (amount == 0):

print()

print("Inventory is full and hence cannot be replenished")

lines()

return 0

Exit = False

while (Exit == False):

print()

print("Press Y to replenish inventory by paying ₹", amount)

print("Press N to cancel the transaction")

print()

choice = input("Enter your choice: ")

choice = choice.upper()

if (choice == "Y"):

Exit = True

for item in inventory:

ID = (item[0],)

if (item[0][0:3] == "Bis"):

mycursor.execute("update inventory set quantity = 50 where \

product\_id = %s",ID)

mydb.commit()

elif (item[0][0:3] == "Bre"):

mycursor.execute("update inventory set quantity = 30 where \

product\_id = %s",ID)

mydb.commit()

elif (item[0][0:3] == "Bro"):

mycursor.execute("update inventory set quantity = 25 where \

product\_id = %s",ID)

mydb.commit()

elif (item[0][0:3] == "Cak"):

mycursor.execute("update inventory set quantity = 15 where \

product\_id = %s",ID)

mydb.commit()

elif (item[0][0:3] == "Coo"):

mycursor.execute("update inventory set quantity = 60 where \

product\_id = %s",ID)

mydb.commit()

elif (item[0][0:3] == "Don"):

mycursor.execute("update inventory set quantity = 40 where \

product\_id = %s",ID)

mydb.commit()

elif (item[0][0:3] == "Rol"):

mycursor.execute("update inventory set quantity = 20 where \

product\_id = %s",ID)

mydb.commit()

print()

print("Transaction successful. Inventory replenished")

lines()

return amount

elif (choice == "N"):

Exit = True

print()

print("Transaction successfully cancelled")

lines()

return 0

else:

print("Invalid input. Try again")

def lines(): #for better and clear division of commands

print()

print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")

print()

def Main(): #the main menu or parent function

print()

print("Welcome to ASTER BAKERY")

comm = ""

Esc = False

while(Esc == False):

Help()

comm = input("What would you like to do? ")

comm = comm.upper()

import mysql.connector as mysql

mydb = mysql.connect(host="localhost", user="root", \

password="adwaiy2912", database="ASTER")

mycursor = mydb.cursor()

mycursor.execute("select \* from totalprofit;")

for tupProfit in mycursor:

for intProfit in tupProfit:

profit = intProfit #converts bakery profit from nested tuple to integer

if (comm == "S"):

See()

elif (comm == "P"):

purchaseProfit = Purchase()

purchaseProfit = (purchaseProfit,)

mycursor.execute("update totalprofit set profit = profit + %s", purchaseProfit)

mydb.commit()

elif (comm == "R"):

if (profit == 0):

print()

print("No profit earned and hence cannot replenish inventory")

lines()

else:

replenishCost = Replenish()

replenishCost = (replenishCost,)

mycursor.execute("update totalprofit set profit = profit - %s", replenishCost)

mydb.commit()

elif (comm == "B"):

print()

print()

print("Current Bakery profit = ₹", profit)

lines()

elif (comm == "Q"):

Esc = True

print()

print()

print("Thank you for your time")

print()

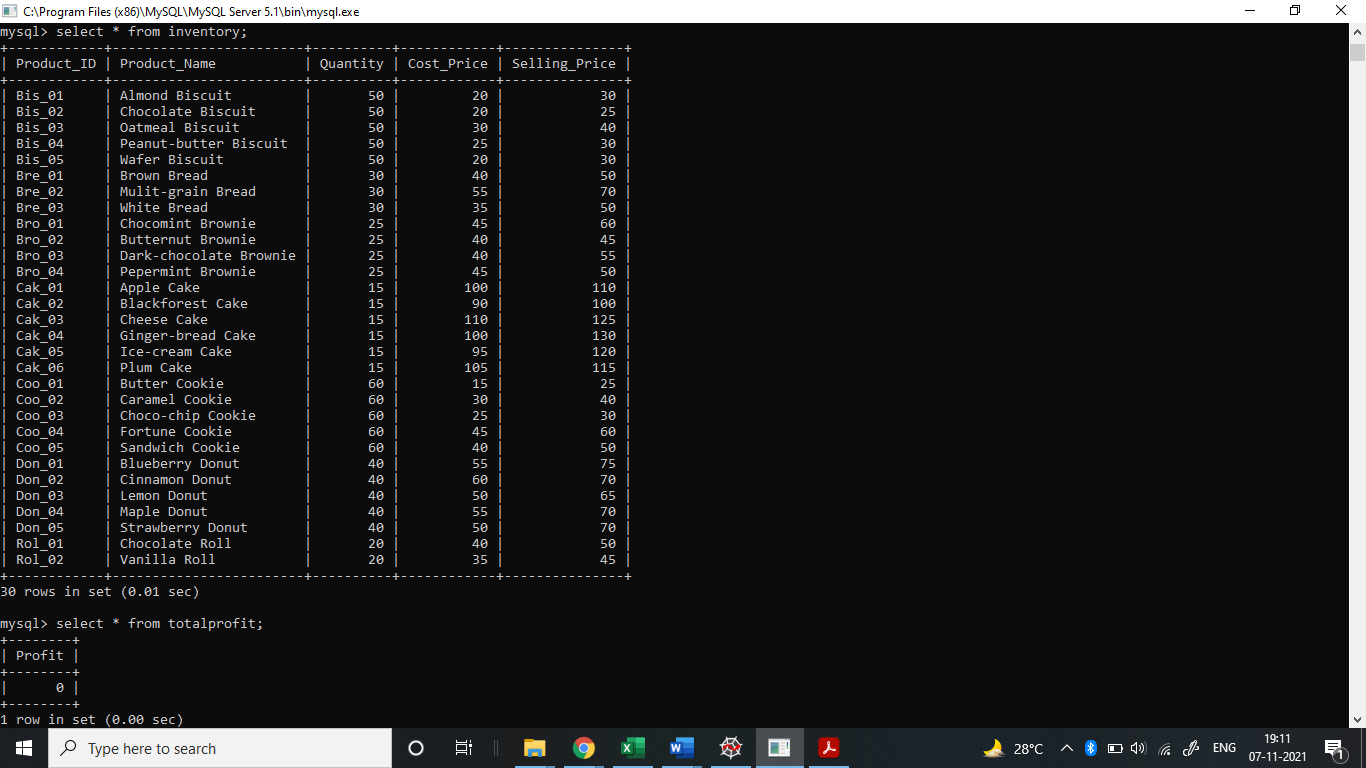
else:

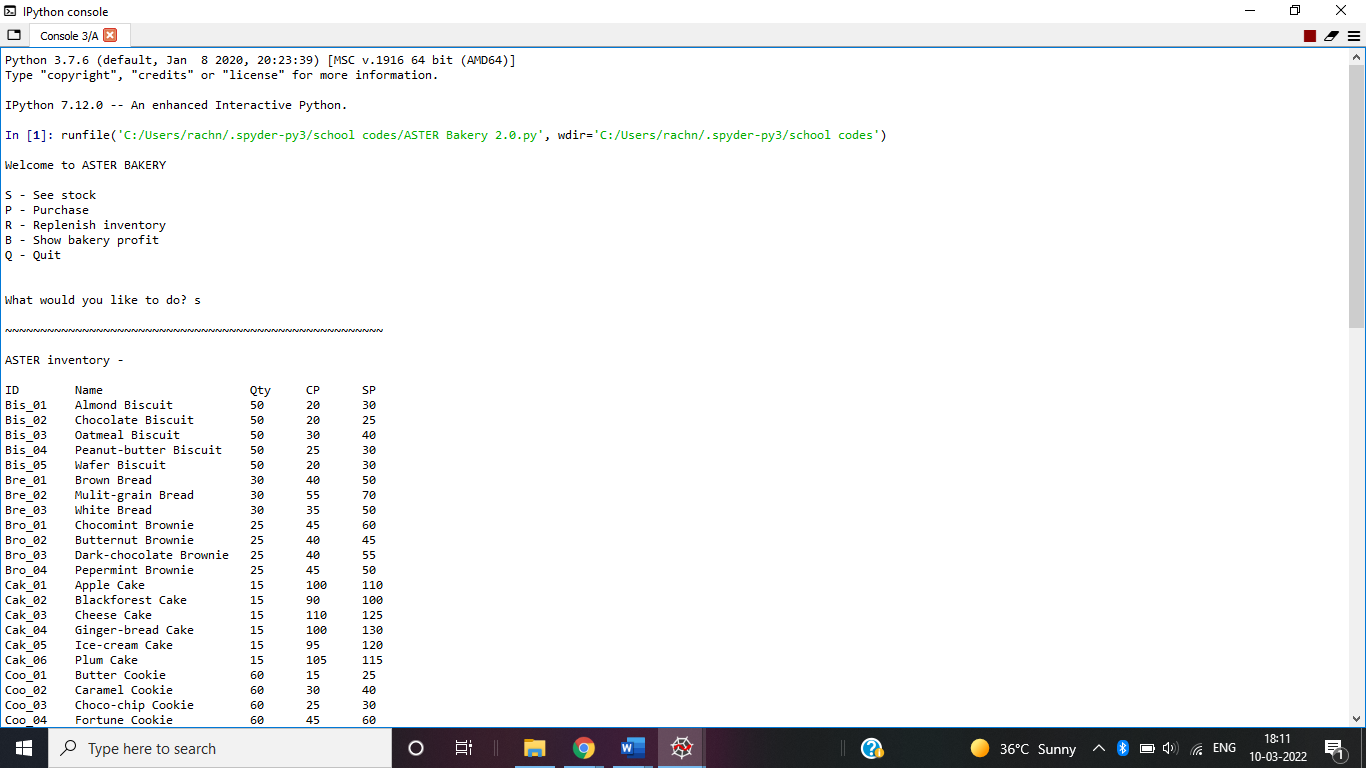
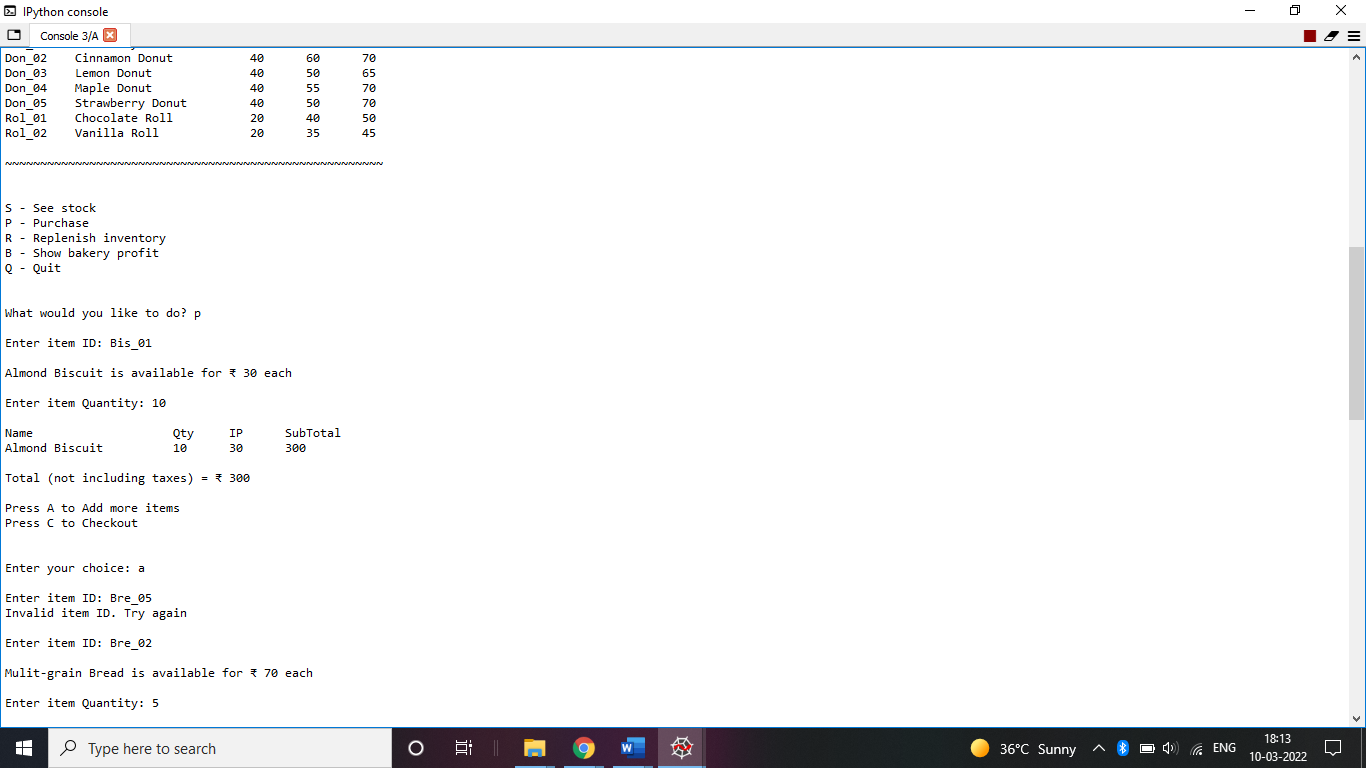
print()

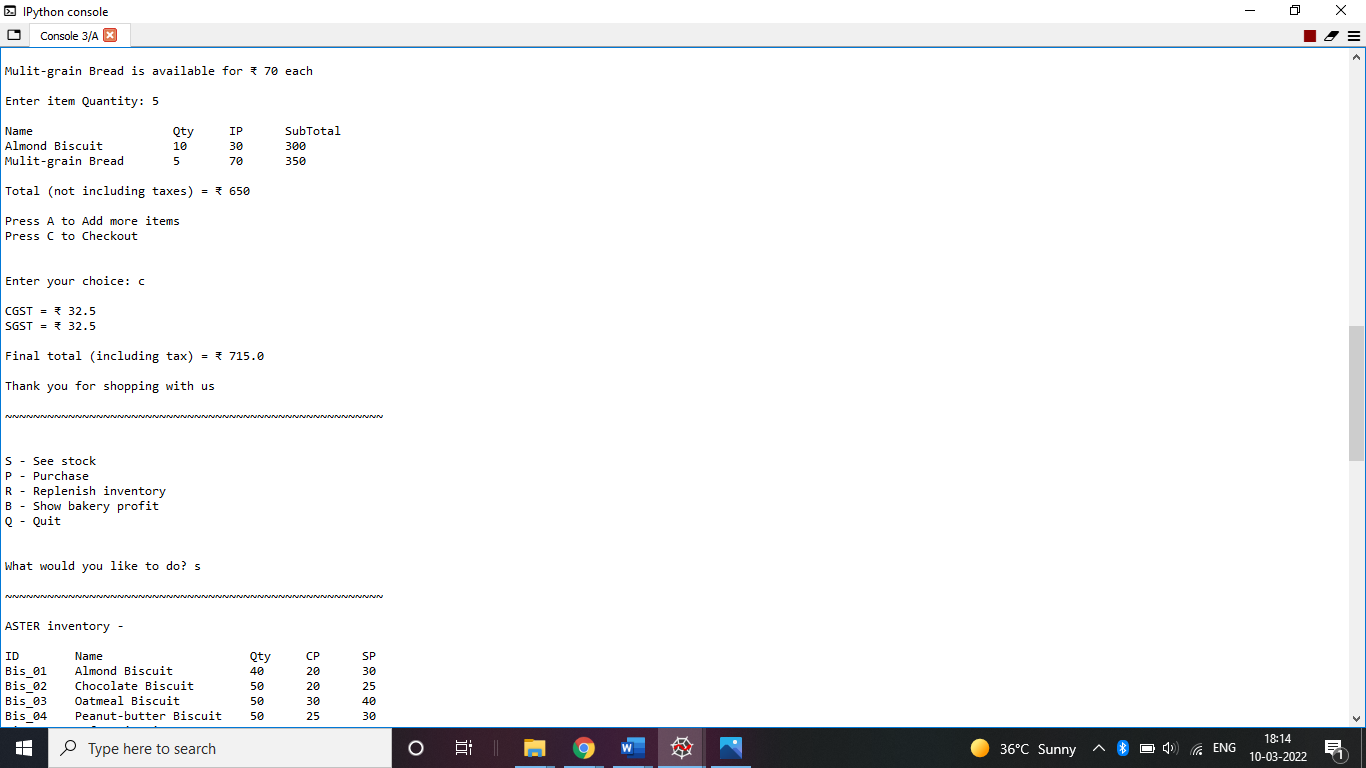
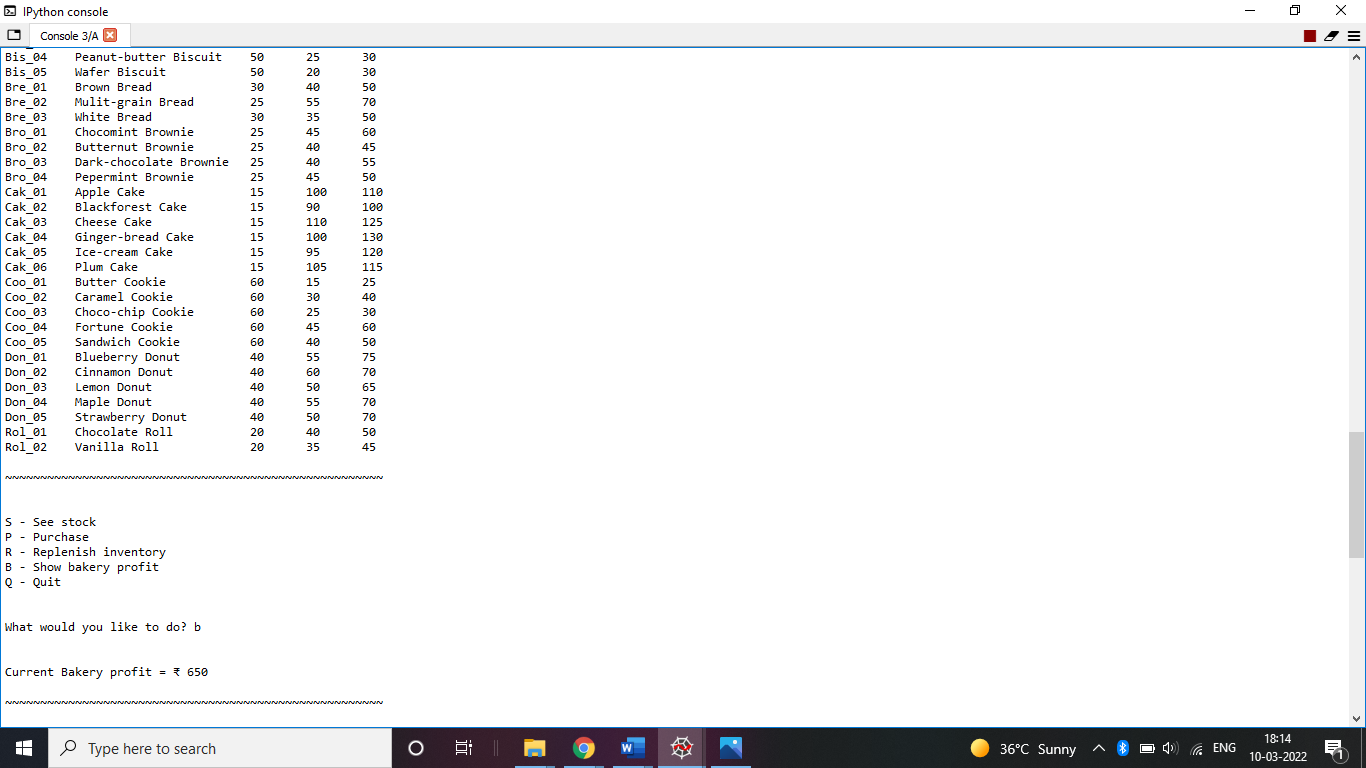
print("Error! Invalid Input. Try Again")

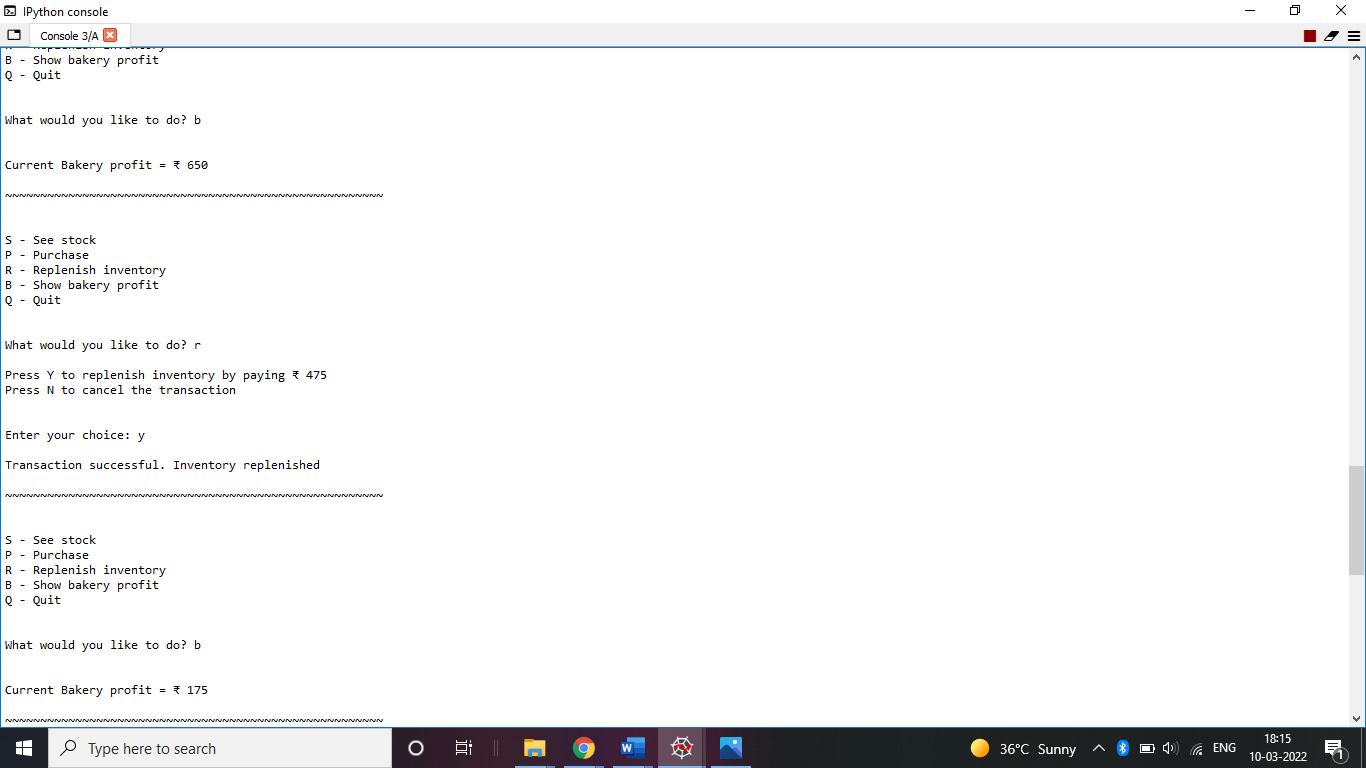
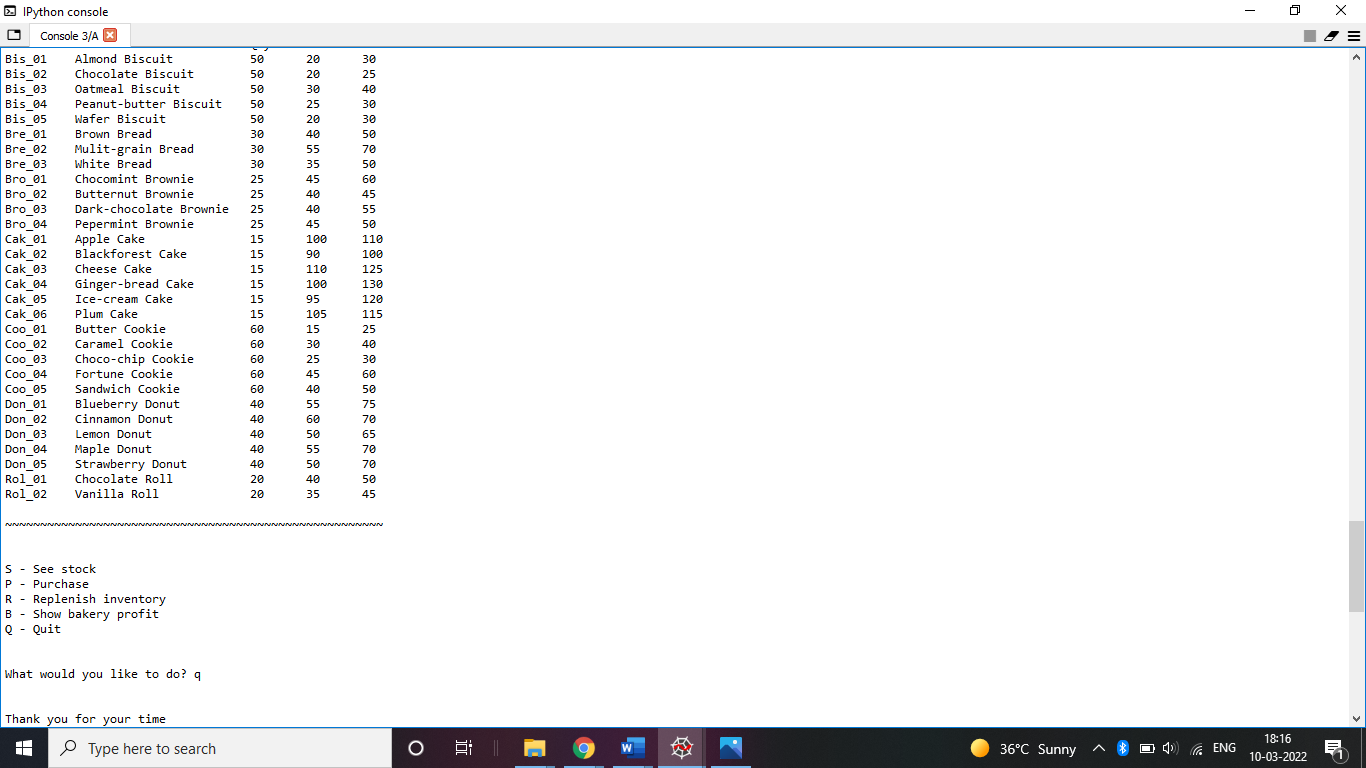
Main()

Screen Shots



Conclusion

This program has all the crucial functionalities that a inventory management system must have. The whole system is a menu driven and hence is user friendly. This program has been thoroughly tested and found to be error free.

While completing this project we faced certain challenges which were resolved with a simple google search or with the help of our teacher Malvika ma’am. This helped us broadening our view and reflecting towards our own mistakes. This project made us work as a team and help each other in times of need.

Bibliography

* <https://www.educba.com/python-print-table/>
* <https://stackabuse.com/python-nested-functions/>
* <https://projectnotes.org/it-projects/inventory-management-system-in-python-with-source-code/>