DELHI PUBLIC SCHOOL

BANGALORE NORTH

COMPUTER SCIENCE PROJECT

Super Market Management System

by

Dhruv Rajeshkumar Shah

DELHI PUBLIC SCHOOL BANGALORE NORTH

CERTIFICATE

This is to certify that the following students of class XII

1) **STUDENT NAME** **:** Dhruv Rajeshkumar Shah

   BOARD ROLL NO **:** 18608480

2) **STUDENT NAME** **:** Anirudh Satish

   BOARD ROLL NO **:** 18608464

have successfully completed the Project Work titled

SuperMarket Management System

under the guidance of

Mrs Uzma Fathima

during the academic year 2020-2021 in partial fulfillment of practical examination for the subject COMPUTER SCIENCE (083) conducted by AISSCE, CBSE

[INTERNAL EXAMINER] **[EXTERNAL EXAMINER]**

[PRINCIPAL]

ACKNOWLEDGEMENT

We would like to express our special thanks of gratitude to the management and principal of Delhi public school Bangalore north for giving us this opportunity to do this project. We would also like to thank our class teacher and computer teacher Mrs Uzma Fathima Ma’am without whose guidance we couldn’t have finished this project.

Lastly, we would also like to thank our parents who not only supported us but also provided us with the necessary resources for us to complete the project in the given time frame.

ABSTRACT

This is a proposed integrated super market management system that aims to support the needs of immense amount data handling in a Supermarket on a day to day basis. It is user-friendly and simple yet a very powerful tool that can help Supermarket store owners, managers and employees to help make tedious tasks such as product management, employee management, order management, bill management, profit statistics calculation and billing much easier.

This software uses python for front end and processing and mysql and textfiles for backend storage of data.

This is a must have application in all supermarket stores to help them make every tasks more efficient and easier!

TABLE OF CONTENTS

INTRODUCTION

Purpose

The main purpose of this project is to aid the owners and staff of large and small supermarket entities in various data management and storage tasks. A lot of transactions take place everyday and it gets hard to maintain a record of all these transactions, for this purpose we have created a simple and easy to use software program that can run on very basic systems and doesn't require much space.

Existing system

A lot of supermarket management systems do exist already but they are all very complicated and not easy to understand. They often take more time to set up and require high spec systems to run. They are often very expensive applications and require training to use them properly. These are not very suitable for small businesses and ventures.

Proposed system

We have considered the following goals while developing this project:

- Management and processing of large amount of data in an organized manner which would've been a very tedious and laborious task
- Simple and Easy structure which is very easy to understand
- Integration and Relations of data for easy processing
- Quick and easy guided set up process that requires less than 1 minute because of the auto-installation of all required files
- Lightweight program capable of powerful tasks
- Security of data and its access being confined to selected individuals

HARDWARE AND SOFTWARE SPECIFICATIONS

Both development and implementation of this software requires the same hardware and software specifications.

Hardware requirements-

1. Processors: Intel Atom® processor or Intel® Core™ i3 processor

2. Disk space: 1 GB

3. Operating systems: Windows* 7 or later, macOS, and Linux

Software requirements-

1. Python version 3+ with latest pip installer

2. MySql version 4.1 +

PROJECT DESIGN

Project Modules and Flow Design

The main functions of this software are distributed in these python files according to their utility-

Main- The basic backbone module that integrates the features of all the other modules and runs. The main file that has to be run to access the software.

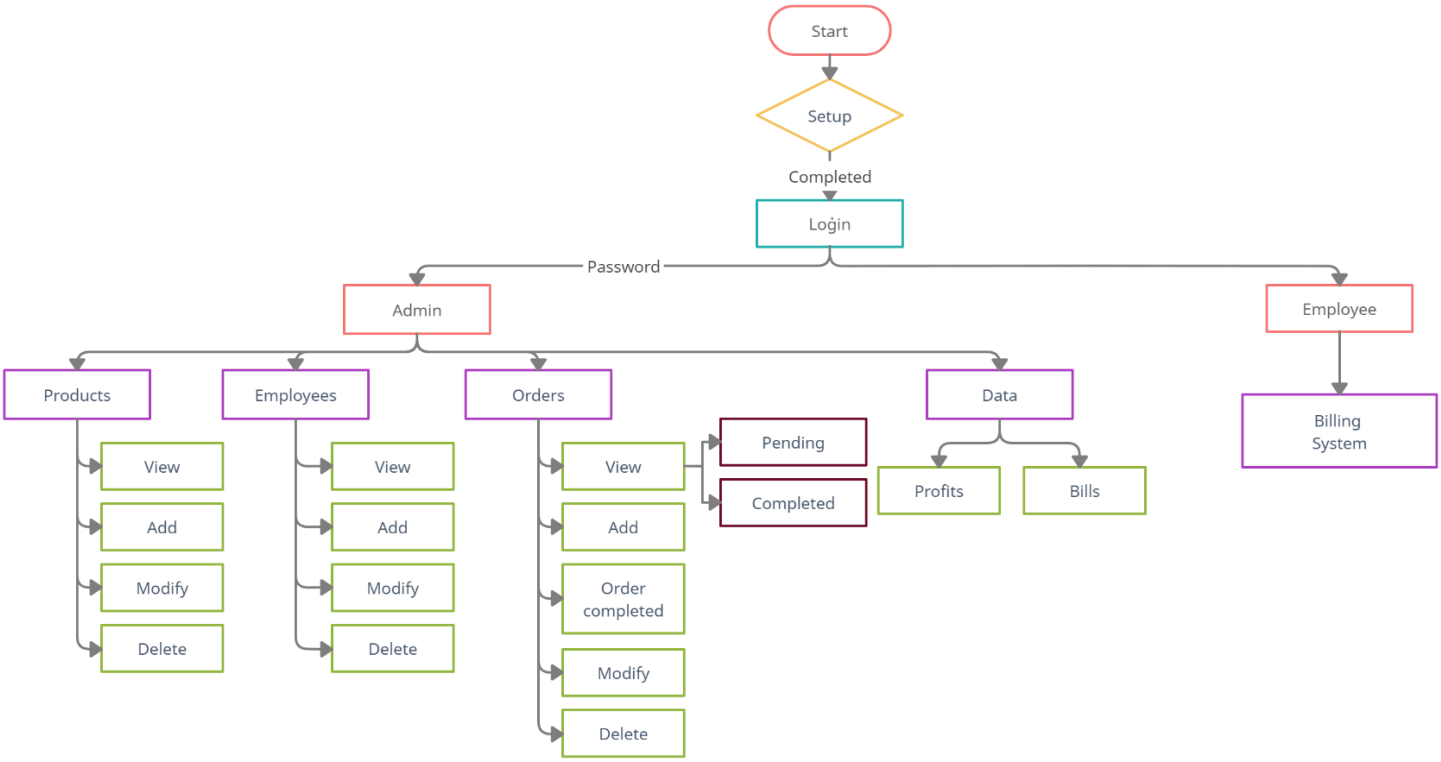
Setup- The module is executed when user uses the program for the first time. It is used to setup the databases and files.

Menus- Contains functions for all the menus in the program. Integrates all the utility functions and has all the function call statements for performing various tasks.

Databases- Contains all the main database utility functions to view, add, modify and delete records in the database and text files.

Billing system- Contains billing related functions for billing and storage and display of bills. Also contains function for calculating profit statistics.

Flow Chart


Start
Setup
Completed
Login
Password
Admin
Employee
Products
Employees
Orders
Data
Billing System
View
Add
Modify
Delete
View
Add
Modify
Delete
View
Add
Order completed
Modify
Delete
Pending
Completed
Profits
Bills

Database Design Structure

Database Name: Same as the name of the store, provided by the user during setup.

Purpose: It contains all the necessary data which is required to be stored by a supermarket.

Tables used for database/project

Table 1: **PRODUCTS –** Stores information about various products in the store.

Table 2: **EMPLOYEE –** Stores personal details of all employees working in the store.

Table 3: **ORDERS –** Stores information about pending orders from suppliers.

Table 4: **BILLS –** Stores information about the bills issued.

Table 1: **PRODUCTS**

Field	Type	Null	Key	Default	Extra
Pno	int	NO	PRI	NULL	
Pname	varchar(50)	NO	UNI	NULL	
CP	float	NO		NULL	
SP	float	NO		NULL	
Stock	int	YES		NULL	
Sold	int	YES		NULL	
Profit	float	YES		NULL	

Table 2: **EMPLOYEE**

Field	Type	Null	Key	Default	Extra
E_ID	int	NO	PRI	NULL	
Ename	varchar(20)	NO		NULL	
DOB	date	NO		NULL	
Join_Date	date	NO		NULL	
PhoneNo	bigint	NO	UNI	NULL	

Table 3: **ORDERS**

Field	Type	Null	Key	Default	Extra
InvoiceNo	bigint	NO	PRI	NULL	
Pno	int	NO		NULL	
Price	float	NO		NULL	
Qty	bigint	NO		NULL	
Total	bigint	NO		NULL	
Date_of_order	date	NO		NULL	

Table 4: **BILLS**

Field	Type	Null	Key	Default	Extra
BillNo	int	NO	PRI	NULL	
Date	date	NO		NULL	
Total	float	NO		NULL	

SAMPLE TABLES

Sample products 1: **PRODUCTS**

Pno	Pname	CP	SP	Stock	Sold	Profit
1	Milk	12	18	13	7	6
2	Bread	15	30	4	4	15

Sample products 2: **EMPLOYEE**

E_ID	Ename	DOB	Join_Date	PhoneNo
1	Dhruv R Shah	2003-05-20	2018-09-12	9111006969
2	Anirudh Satish	2003-02-11	2018-09-23	9969666969
3	Darsh Rawat	2004-01-23	2019-08-02	9642069420

Sample products 3: **ORDERS**

InvoiceNo	Pno	Price	Qty	Total	Date_of_order
2	2	15	5	75	2021-03-13
3	2	15	6	90	2021-03-13

Sample products 4: **BILLS**

BillNo	Date	Total
1	2021-03-13	114
2	2021-03-13	132

PYTHON LIBRARIES AND FILES

LIBRARIES

This project uses a number of built-in as well as installed libraries and modules for various functions.

Built in-

OS- system(), mkdir(), path()

Used for clearing screen, creating bills directory, checking if files such as smms, sno, completed orders exists.

Datetime- date.today(), datetime.now()

Used to get the current date and time.

Installed libraries-

Art- text2art()

Used for ascii art text for headings.

Mysql.connector- connect(), cursor(), cursor.execute(), cursor.fetchall(), commit()

Used for establishing database connections and database management functions.

Tabulate- tabulate()

Used for displaying data in a tabulated form.

DIRECTORY/FOLDERS

bills- directory/folder created during setup to store bills

FILES

This project uses text files for various purposes such as displaying certain text, storing basic data and password, serial numbers, records and bills. This project has both premade text files which come with the package program and ones that are created after execution.

Premade text files-

Requirements.txt- Used to display the system requirements before setup to ensure all functions work properly

Requirements - Notepad

File Edit Format View Help

```
SuperMarket Management is an easy to use very powerful and helpful app for managing a supermarket.
This all in one app offers following utilities-
1. Product Management
2. Employee Management
3. Orders Management and log
4. Bills storage and management
5. Profit stats
6. Billing System

To run this app on your system you need to have the following requirements -

Hardware requirements-
1. Processors: Intel Atom® processor or Intel® Core™ i3 processor
2. Disk space: 1 GB
3. Operating systems: Windows* 7 or later, macOS, and Linux

Software requirements-
1. Python version  3+  with latest pip installer
2. MySql version 4.1 +

This app has its own modules but also uses certain inbuilt and existing libraries.
These need to be installed in the system via the pip installer.

REQUIRED LIBRARY FILES TO BE INSTALLED-
1. mysql connector
2. art
3. os
4. datetime
5. tabulate
```

(remains same throughout)

Created after execution-

<u>SMMS.txt</u>- The main text file whose existence determines weather setup is done or not. Stores name of store/database, password for the software and mysql password

SMMS - Notepad

File Edit Format View Help

```
['Freshh', '123', 'dhruv']
```

(is sample, subjected to change)

<u>sno.txt</u>- Used to store product no, employee no, order no and bill no for automated numbering

sno - Notepad

File Edit Format View Help

```
3
4
4
2
```

(is sample, subjected to change)

<u>completed_orders.txt</u>- Used to store details of completed orders

completed_orders - Notepad

File Edit Format View Help

```
["Invoice no","Product No","Price","Quantity","Total","Date of order","Date of Deliviery"]
[1, 1, 12.0, 10, 120, datetime.date(2021, 3, 13), datetime.date(2021, 3, 13)]
```

(is sample, subjected to change)

SOURCE CODE

main.py

```
#importing libraries and modules
from os import *
from setup import *

#checks if the file with basic necessary info exists
#opens set up if it doesn't
if path.exists("SMMS.txt")==False or path.getsize("SMMS.txt")==0:
    setup()

#the main menu and program
from menus import *
main()
```

setup.py

```
#importing library files and modules
from os import system,path,mkdir
from art import *

#to box text
def box(msg, indent=1, width=None, title=None):
    lines = msg.split('\n')
    space = " " * indent
    if not width:
        width = max(map(len, lines))
    box = f'╔{"=" * (width + indent * 2)}╗\n'
    if title:
        box += f'╟{space}{title:<{width}}{space}╟\n'
        box += f'╟{space}{"-" *
len(title):<{width}}{space}╟\n'
    box += ''.join([f'╟{space}{line:<{width}}{space}╟\n'
for line in lines])
    box += f'╚{"=" * (width + indent * 2)}╝'
    print(box)
```

```
#main set up function when installed on a new system
def setup():
    tprint("SUPERMARKET \nMANAGEMENT SYSTEM","big")
    print("Welcome to this Super Market Management
System")
    print("Lets get started!")
    print()
    #to display the file which containts the requirements
    w=open("Requirements.txt")
    box(msg=w.read(),title="REQUIREMENTS")
    print()
    r=input("Are all the requirements satisfied?[yes/no] :
")
    #if requirements aren't met, program asks user to
satisfy them first
    #program terminates if conditions not met
    if r in "noNO":
        print()
        print("Please use a system which satisfies the
requirements!")
        quit()

    #if requirements are met the code continues
    #user is asked to provide the necessary details and
password for mysql connectivity
    system('cls')
    print("SETUP")
    print()
    dbname=input("Enter the name of store(seperate words
with '_') : ")
    password=input("set password for admin access : ")
    pcheck=input("re-enter password :")
    print()
    dbpw=input("Enter mysql password : ")

    #details are then stored in a textfile
    if password==pcheck:
```

```
        f=open("SMMS.txt","w")
        f.write(str([dbname,password,dbpw]))
        print("Details successfully updated!")
        f.close()
        load_resources()
        input("""Your app is ready for use!
Press enter to continue...""")


    else:
        print("Your password does not match. Try again")
        input()
        system('cls')
        setup()





#function to load necessary load_resources
#creates database and tables
def load_resources():
    #creates folder for storing bills
    if path.exists("bills")==False:
        mkdir("bills")

    #creates necessary textfiles

    #file to save product no, employee no, order no and
bill no
    if path.exists("sno.txt")==False or
path.getsize("sno.txt")==0:
        f=open("sno.txt","w")
        f.writelines(["0\n","0\n","0\n"+"0"])
        f.close()

    #file to save completed orders
    if path.exists("completed_orders.txt")==False or
path.getsize("completed_orders.txt")==0:
```

```
        f=open("completed_orders.txt","w")
        f.write("[\"Invoice no\",\"Product
No\",\"Price\",\"Quantity\",\"Total\",\"Date of
order\",\"Date of Deliviery\"]\n")
        f.close()

    #creating database and required tables
    f=open("SMMS.txt","r")
    data=eval(f.read())
    f.close()
    dbname=data[0]
    dbpw=data[2]

    import mysql.connector as mc

    mycon=mc.connect(host="localhost", user="root",
passwd=dbpw)
    cursor=mycon.cursor()

    cursor.execute("create database if not exists
{}".format(dbname))
    cursor.execute("use {}".format(dbname))

    cursor.execute("create table if not exists
PRODUCTS(Pno integer primary key,Pname varchar(50) not
null unique,CP float not null,SP float not null,Stock
int,Sold int,Profit float)")

    cursor.execute("create table if not exists
EMPLOYEE(E_ID integer primary key,Ename varchar(20) not
null,DOB date not null,Join_Date date not null,PhoneNo
bigint not null unique)")

    cursor.execute("create table if not exists
ORDERS(InvoiceNo bigint primary key,Pno integer not
null,Price float not null,Qty bigint not null,Total bigint
not null,Date_of_order date not null)")

    cursor.execute("create table if not exists
```

```
BILLS(BillNo integer primary key,Date date not null,Total
float not null)")
```

menus.py

```
#importing libraries and modules
from databases import *
from os import system
from setup import *
from art import *
from billing_system import *


#menu for viewing,adding,modifying and deleting from
tables
def vadm_menu(t):
    system('cls')
    print(t.upper())
    print()
    opt=int(input("""Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : """))

    if opt==1:
        view(t)
    elif opt==2:
        add(t)
    elif opt==3:
        modify(t)
    elif opt==4:
        delete(t)
    elif opt==5:
        admin()
    else:
        print("INVALID!")
```

```
    print()
    admin()


#for product management functions
def products():
    print()
    print("PRODUCTS")
    print()
    t="products"
    vadm_menu(t)

#for employee management functions
def employee():
    print()
    print("EMPLOYEE MANAGEMENT")
    print()
    t="employee"
    vadm_menu(t)

#for order management functions
def orders():
    system('cls')
    print("ORDERS")
    print()
    t="orders"
    opt=int(input("""Enter the option you want to access:
1. View
2. Add
3. Modify
4. Order delivered
5. Delete order
6. Go back : """))

    if opt==1:
        system('cls')
        print()
        opt2=int(input("""Enter the table you would like
to view :
```

```
1. Pending Orders
2. Completed Orders
3. Go back : """))
        if opt2==1:
            view(t)
        elif opt2==2:
            comp_orders(t)
        elif opt2==3:
            orders()
        else:
            print("INVALID INPUT")

    elif opt==2:
        add(t)
    elif opt==3:
        modify(t)
    elif opt==4:
        ord_del(t)
    elif opt==5:
        delete(t)
    elif opt==6:
        admin()
    else:
        print("INVALID!")
    print()
    admin()

#for stored data and profit stats
def data():
    system('cls')
    print("DATA")
    print()
    t="products"
    dat=int(input("""Select the option you want to access:
1. Bills
2. Profits
3. Go back :
 """))
    if dat==1:
```

```
        read_bills()
        admin()
    elif dat==2:
        total_profit()
    elif dat==3:
        admin()
    else:
        print("INVALID!")
    admin()

#for admin menu functions
def admin():
    system('cls')
    print()
    print("ADMIN")
    print()
    o=int(input("""Select the option you want to access:
1. Products
2. Employee Management
3. Orders
4. Data:
5. Exit admin : """))
    if o==1:
        products()
    elif o==2:
        employee()
    elif o==3:
        orders()
    elif o==4:
        data()
    elif o==5:
        main()
    else:
        print("INVALID!")

#for employee functions
def Employee():
    billing()
```

```
#main menu
def main():

    while True:
        system('cls')
        print("")
        head=text2art(dbname,font="big")
        print(head)
        print("Have a wonderful experience!")
        print("---------------------------------")
        print("LOGIN")
        a=int(input("""Select access mode :
        1. Admin
        2. Employee
        3. Exit : """))
        if a==1:
            system("cls")
            print("LOGIN")
            pwd=input("Enter password : ")
            f=open("SMMS.txt","r")
            data=eval(f.read())
            f.close()
            password=data[1]
            if pwd==password:
                admin()
            else:
                print("Incorrect password, access
denied!")
                input("Press enter to continue..")
                main()
        elif a==2:
            Employee()
        elif a==3:
            end()

        else:
            print("INVALID!")
            input("Press enter to continue..")
```

```
#end
def end():
    system('cls')
    box(text2art("""Thank  you  for  using

    SuperMarket Management
    system
    by
    Dhruv and Anirudh""","big"))
    quit()
```

databases.py

```
#getting the database name and password from tect file
f=open("SMMS.txt","r")
data=eval(f.read())
f.close()
dbname=data[0]
dbpw=data[2]

#importing libraries and modules
from os import path
import datetime
import mysql.connector as mc
from tabulate import tabulate
#establishing connection with mysql
mycon=mc.connect(host="localhost", user="root",
passwd=dbpw)
cursor=mycon.cursor()

cursor.execute("use {}".format(dbname))

#serial number functions
#to update serial no when new record is added
def sno_upt(t,sno):
    sno=str(sno)+"\n"
    f=open("sno.txt","r+")
```

```
    d=f.readlines()
    if t=="products":
        d[0]=sno
    elif t=="employee":
        d[1]=sno
    elif t=="orders":
        d[2]=sno
    elif t=="bills":
        d[3]=sno
    f.seek(0)
    f.writelines(d)
    f.close()

#to fetch serial number
def sno_read(t):
    f=open("sno.txt","r")
    d=f.readlines()
    if t=="products":
        sno=int(d[0])+1
    elif t=="employee":
        sno=int(d[1])+1
    elif t=="orders":
        sno=int(d[2])+1
    elif t=="bills":
        sno=int(d[3])+1
    f.close()
    return sno


#database functions

#to view table
def view(t):
    print()
    cursor.execute("select * from {}".format(t))
    d=cursor.fetchall()
    cursor.execute("desc {}".format(t))
    d2=cursor.fetchall()
    h=[]
```

```
    for i in d2:
        h.append(i[0])
    print()
    print(t)
    print (tabulate(d,h,tablefmt="pretty"))

    print()
    input("Press enter to continue..")
    return d

#to add new records to the table
def add(t):
    cursor.execute("desc {}".format(t))
    d=cursor.fetchall()
    n=int(input("Enter no of records that need to be added
: "))


    for i in range(n):
        sno=sno_read(t)
        l=[sno]
        k=0

        for j in d[1:]:
            if "Total" in j[0]:
                l.append(l[2]*l[3])
                continue
            if "Sold" in j[0]:
                l.append(0)
                continue
            if "Profit" in j[0]:
                l.append(l[3]-l[2])
                continue
            if "Date_of_order" in j[0]:
                l.append(str(datetime.date.today()))
                continue
            if t=="orders" and "Price" in j[0]:
                cursor.execute("Select CP from products
where Pno={}".format(l[1]))
```

```
                price=cursor.fetchone()[0]
                l.append(price)
                continue


            if "varchar" in j[1] or "char" in j[1] or
"date" in j[1]:
                l.append(input("enter the value of
{}:".format(j[0])))
            elif j[1] in "integer" or j[1] in "bigint":
                l.append(int(input("enter the value of
{}:".format(j[0]))))
            else:
                l.append(float(input("enter the value of
{}:".format(j[0]))))
            k+=1
        l=str(tuple(l))
        cursor.execute("insert into {}
values{}".format(t,l))
        print()
        mycon.commit()
        print("records added successfully!")
        sno_upt(t,sno)
        print()
        input("Press enter to continue..")
        print()


#to mark an order delivered
def ord_del(t):
    view(t)
    inv=int(input("Enter Invoice number of the order that
has been delivered :"))
    cursor.execute("select * from orders where
InvoiceNo={}".format(inv))
    od=list(cursor.fetchall()[0])
    cursor.execute("select Pno from Products")
    pno=cursor.fetchall()
    n_qty=od[3]
```

```
    p1=(od[1],)
    if p1 in pno:
        cursor.execute("select stock from products where
Pno={}".format(p1[0]))
        o_qty=cursor.fetchall()[0][0]
        cursor.execute("update products set stock={} where
Pno={}".format((o_qty+n_qty),p1[0]))
        cursor.execute("delete from orders where
Pno={}".format(p1[0]))
        od.append(datetime.date.today())
        co=open("completed_orders.txt","a+")
        co.write(str(od)+"\n")
        co.close()
        print()
        print("successfully updated details!")
        mycon.commit()
    else:
        print("Product does not exist in database, please
add product in products and retry!")
    print()
    input("Press enter to continue..")
    print()

#to display completed orders
def comp_orders(t):
    print()
    f=open("completed_orders.txt")
    d=[]
    h=eval(f.readline())
    for i in f.readlines():
        d.append(eval(i))
    print (tabulate(d,h,tablefmt="pretty"))
    f.close()
    print()
    input("Press enter to continue..")

#to modify records
def modify(t):
    view(t)
```

```
    print()
    cursor.execute("desc {}".format(t))
    d=cursor.fetchall()
    dic={}
    for i in d:
        dic[i[0]]=i[1]

    n=int(input("Enter the {} : ".format(d[0][0])))
    op=""
    a=1
    print()
    cursor.execute("select * from {} where
{}={}".format(t,d[0][0],n))
    r=cursor.fetchall()
    h=[]
    for i in d:
        h.append(i[0])
    print()
    print (tabulate(r,h))
    print()

    for i in d:
        op=op+str(a)+". "+i[0]+"\n"
        a+=1

    f=input("""Enter the field name you would like to
modify {}""".format("\n"+op))
    val=input("Enter new value :")


    if "varchar" in dic[f] or "char" in dic[f] or "date"
in dic[f]:
        cursor.execute("update {} set {}='{}' where
{}".format(t,f,val,d[0][0]+"="+str(n)))

    else:
        cursor.execute("update {} set {}={} where
{}".format(t,f,val,d[0][0]+"="+str(n)))
```

```
    print("successfully updated")
    mycon.commit()
    print()
    input("Press enter to continue..")


#to delete records
def delete(t):
    view(t)
    cursor.execute("desc {}".format(t))
    d=cursor.fetchall()
    n=int(input("Enter the {} : ".format(d[0][0])))
    cursor.execute("delete from {} where
{}".format(t,d[0][0]+"="+str(n)))
    print()
    mycon.commit()
    print("Record deleted successfully")
    print()
    input("Press enter to continue...")
```

billing_system.py

```
#importing functions and modules
from databases import *
from os import system
from tabulate import tabulate
import datetime as dt
import menus

#to box text
def box(msg, indent=1, width=None, title=None):
    lines = msg.split('\n')
    space = " " * indent
    if not width:
        width = max(map(len, lines))
    box = f'╔{"=" * (width + indent * 2)}╗\n'
    if title:
```

```
        box += f'╟{space}{title:<{width}}{space}╟\n'
        box += f'╟{space}{"-" *
len(title):<{width}}{space}╟\n'
    box += ''.join([f'╟{space}{line:<{width}}{space}╟\n'
for line in lines])
    box += f'╚{"=" * (width + indent * 2)}╝'
    print(box)

#billing system
def billing():
    t="bills"
    system('cls')
    print()
    print("BILLING SYSTEM")
    print()
    view("products")
    print()
    bno=sno_read(t)
    d=[]
    h=["Product No","Product name","Price","Qty","Amount"]
    a=True
    total=0
    while a!=False:
        print()
        pno=int(input("Enter product number : "))
        l=[pno]
        cursor.execute("Select Pname,sp from products
where pno={}".format(pno))
        pdt=cursor.fetchall()[0]
        l.extend(pdt)
        l.append(int(input("Enter quantity : ")))
        l.append(l[2]*l[3])
        total+=l[4]
        d.append(l)
        print()
        a=input("Press enter to continue and any other key
to stop bill.. ")
        if a!="":
            a=False
```

```
    system('cls')
    dttm=dt.datetime.now()
    date=dttm.strftime("%Y-%m-%d")
    time=dttm.strftime("%H:%M")
    b=tabulate(d,h)
    bprv=("BILL NO : "+str(bno)+
    "\n\nDate : "+str(date)+
    "\nTime : "+str(time)+
    "\n\n"+tabulate(d,h)+
    "\n\nTOTAL : "+str(total))
    box(bprv)
    print()
    bill_conf(bno,date,time,b,total,d)
    print()
    cont=input("Press enter to continue billing, any other
key to exit to main menu..")
    if cont=="":
        billing()

#confirming the bill
def bill_conf(bno,date,time,b,total,d):
    conf=input("Please confirm if the bill is correct[y/n]
: ")

    if conf in "YESyes":
        f=open("bills\\Bill{}.txt".format(bno),"w")
        f.writelines(["BILL NO :{}\n\n".format(bno),"Date
: {}\n".format(date),"Time : {}\n\n".format(time),b,"\n\n
Total : {}".format(total)])
        f.close()
        cursor.execute("insert into BILLS
values({},'{}',{})".format(int(bno),date,total))
        for i in d:
            pno=i[0]
            qty=i[3]
            cursor.execute("select stock,sold from
products where pno={}".format(pno))
            data1=cursor.fetchall()[0]
            stock=data1[0]-qty
```

```
            sold=data1[1]+qty
            cursor.execute("update products set stock={},
sold={} where Pno={}".format(stock,sold,pno))
        mycon.commit()
        sno_upt("bills",bno)
        print("Bill generated successfully!")

    elif conf in "NOno":
        print()
        print("Bill cancelled and discarded!")


    else:
        print("Invalid option selected!")
        bill_conf()

#displaying old bills
def read_bills():
    system('cls')
    print("Bills")
    print()
    dt=input("Enter date of the bill[yyyy-mm-dd]: ")
    cursor.execute("Select * from bills where
date='{}'".format(dt))
    d=cursor.fetchall()
    if d==[]:
        print()
        print("No bills found on this date!")
        print()
        input("Press Enter to continue..")

    else :
        cursor.execute("desc bills")
        d2=cursor.fetchall()
        h=[]
        for i in d2:
            h.append(i[0])
        print()
        print(tabulate(d,h))
```

```
        print()
        bno=int(input("Enter bill no : "))
        f=open("bills\\Bill{}.txt".format(bno))
        bill=f.read()
        system('cls')
        box(bill)
        print()
        input("Press enter to continue...")

#displaying total profits
def total_profit():
    system('cls')
    print()
    total=0
    h=["Product No","Product Name","Quantity
Sold","Profit","Total profit"]
    cursor.execute("Select pno,pname,sold,profit from
products")
    initd=cursor.fetchall()
    d=[]
    for i in initd:
        i=list(i)
        t=i[2]*i[3]
        i.append(t)
        total+=t
        d.append(i)
    print(tabulate(d,h,tablefmt="pretty"))
    print()
    print("Total profit on all the products : ",total)
    print()
    input("Press enter to continue..")
```

OUTPUT SNAPSHOTS

Setup screen 1- Requirements confirmation

```
Welcome to this Super Market Management System
Lets get started!

REQUIREMENTS
------------
SuperMarket Management is an easy to use very powerful and helpful app for managing a supermarket.
This all in one app offers following utilities-
1. Product Management
2. Employee Management
3. Orders Management and log
4. Bills storage and management
5. Profit stats
6. Billing System

To run this app on your system you need to have the following requirements -

Hardware requirements-
1. Processors: Intel Atom® processor or Intel® Core™ i3 processor
2. Disk space: 1 GB
3. Operating systems: Windows* 7 or later, macOS, and Linux

Software requirements-
1. Python version  3+  with latest pip installer
2. MySql version 4.1 +

This app has its own modules but also uses certain inbuilt and existing libraries.
These need to be installed in the system via the pip installer.

REQUIRED LIBRARY FILES TO BE INSTALLED-
1. mysql connector
2. art
3. os
4. datetime
5. tabulate

Are all the requirements satisfied?[yes/no] : yes
```

If “yes” in entered, it takes you to setup screen 2 or terminates program and asks user to meet requirements first.

Setup screen 2- Login info and password

```
SETUP

Enter the name of store(seperate words with '_') : Freshh
set password for admin access : 123
re-enter password :123

Enter mysql password : dhruv
Details successfully updated!
Your app is ready for use!
Press enter to continue...
```

If password and re-entered password match it takes to main menu, else prints error

Main menu screen

```
 ______                    _       _
|  ____|                  | |     | |
| |__    _ __   ___  ___ | |__   | |__
|  __|  | '__| / _ \/ __|| '_ \  | '_ \
| |     | |   |  __/\__ \| | | || | | |
|_|     |_|    \___||___/|_| |_||_| |_|



Have a wonderful experience!
-----------------------------------
LOGIN
Select access mode :
        1. Admin
        2. Employee
        3. Exit : 1
```

Admin access- incorrect password

```
LOGIN
Enter password : xyz
Incorrect password, access denied!
Press enter to continue..
```

If pass word is correct it takes you to admin menu

Admin menu

```
ADMIN

Select the option you want to access:
1. Products
2. Employee Management
3. Orders
4. Data:
5. Exit admin :
```

Products menu

```
PRODUCTS

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back :
```

Adding products

```
PRODUCTS

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 2
Enter no of records that need to be added : 3
enter the value of Pname:Milk
enter the value of CP:12
enter the value of SP:18
enter the value of Stock:10

records added successfully!

Press enter to continue..

enter the value of Pname:BreaD
enter the value of CP:15
enter the value of SP:30
enter the value of Stock:8

records added successfully!

Press enter to continue..

enter the value of Pname:Curd
enter the value of CP:15
enter the value of SP:20
enter the value of Stock:12

records added successfully!

Press enter to continue..
```

Viewing products after adding

```
PRODUCTS

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 1


products
+-----+-------+------+------+-------+------+--------+
| Pno | Pname |  CP  |  SP  | Stock | Sold | Profit |
+-----+-------+------+------+-------+------+--------+
|  1  | Milk  | 12.0 | 18.0 |  10   |  0   |  6.0   |
|  2  | BreaD | 15.0 | 30.0 |   8   |  0   |  15.0  |
|  3  | Curd  | 15.0 | 20.0 |  12   |  0   |  5.0   |
+-----+-------+------+------+-------+------+--------+

Press enter to continue..
```

Notice how 3 records are now added in the products table.

Modifying products

```
PRODUCTS

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 3


products
+-----+-------+------+------+-------+------+--------+
| Pno | Pname |  CP  |  SP  | Stock | Sold | Profit |
+-----+-------+------+------+-------+------+--------+
|  1  | Milk  | 12.0 | 18.0 |  10   |  0   |  6.0   |
|  2  | BreaD | 15.0 | 30.0 |   8   |  0   |  15.0  |
|  3  | Curd  | 15.0 | 20.0 |  12   |  0   |  5.0   |
+-----+-------+------+------+-------+------+--------+

Press enter to continue..

Enter the Pno : 2


  Pno  Pname      CP    SP    Stock    Sold    Profit
-----  -------  ----  ----  -------  ------  --------
    2  BreaD      15    30        8       0        15

Enter the field name you would like to modify
1. Pno
2. Pname
3. CP
4. SP
5. Stock
6. Sold
7. Profit
Pname
Enter new value :Bread
successfully updated

Press enter to continue..
```

Viewing products after modification

```
PRODUCTS

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 1


products
+-----+-------+------+------+-------+------+--------+
| Pno | Pname |  CP  |  SP  | Stock | Sold | Profit |
+-----+-------+------+------+-------+------+--------+
|  1  | Milk  | 12.0 | 18.0 |  10   |  0   |  6.0   |
|  2  | Bread | 15.0 | 30.0 |   8   |  0   |  15.0  |
|  3  | Curd  | 15.0 | 20.0 |  12   |  0   |  5.0   |
+-----+-------+------+------+-------+------+--------+

Press enter to continue..
```

Notice how the Pname of “BreaD” has been modified to “Bread”

Deleting products

```
PRODUCTS

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 4


products
+-----+-------+------+------+-------+------+--------+
| Pno | Pname |  CP  |  SP  | Stock | Sold | Profit |
+-----+-------+------+------+-------+------+--------+
|  1  | Milk  | 12.0 | 18.0 |  10   |  0   |  6.0   |
|  2  | Bread | 15.0 | 30.0 |   8   |  0   |  15.0  |
|  3  | Curd  | 15.0 | 20.0 |  12   |  0   |  5.0   |
+-----+-------+------+------+-------+------+--------+

Press enter to continue..
Enter the Pno : 3

Record deleted successfully

Press enter to continue...
```

Viewing products after deletion

```
PRODUCTS

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 1


products
+-----+-------+------+------+-------+------+--------+
| Pno | Pname |  CP  |  SP  | Stock | Sold | Profit |
+-----+-------+------+------+-------+------+--------+
|  1  | Milk  | 12.0 | 18.0 |  10   |  0   |  6.0   |
|  2  | Bread | 15.0 | 30.0 |   8   |  0   |  15.0  |
+-----+-------+------+------+-------+------+--------+

Press enter to continue..
```

Notice how record with Pno as 3 has been deleted

Employee management menu

```
EMPLOYEE

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : _
```

Adding employees

```
EMPLOYEE

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 2
Enter no of records that need to be added : 4
enter the value of Ename:Dhruv Shah
enter the value of DOB:2003-05-20
enter the value of Join_Date:2018-09-12
enter the value of PhoneNo:9111006969

records added successfully!

Press enter to continue..

enter the value of Ename:Anirudh Satish
enter the value of DOB:2003-02-11
enter the value of Join_Date:2018-09-23
enter the value of PhoneNo:9969666969

records added successfully!

Press enter to continue..

enter the value of Ename:Darsh Rawat
enter the value of DOB:2004-01-23
enter the value of Join_Date:2019-08-02
enter the value of PhoneNo:9642069420

records added successfully!

Press enter to continue..

enter the value of Ename:Harsh Raja
enter the value of DOB:2004-06-12
enter the value of Join_Date:2019-12-12
enter the value of PhoneNo:9910069691

records added successfully!

Press enter to continue..
```

Viewing employees after addition

```
EMPLOYEE

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 1


employee
+------+----------------+------------+------------+------------+
| E_ID |     Ename      |    DOB     | Join_Date  |  PhoneNo   |
+------+----------------+------------+------------+------------+
|  1   |   Dhruv Shah   | 2003-05-20 | 2018-09-12 | 9111006969 |
|  2   | Anirudh Satish | 2003-02-11 | 2018-09-23 | 9969666969 |
|  3   |  Darsh Rawat   | 2004-01-23 | 2019-08-02 | 9642069420 |
|  4   |   Harsh Raja   | 2004-06-12 | 2019-12-12 | 9910069691 |
+------+----------------+------------+------------+------------+

Press enter to continue..
```

4 employee records are added

Modifying employees

```
EMPLOYEE

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 3


employee
+------+----------------+------------+------------+------------+
| E_ID |     Ename      |    DOB     | Join_Date  |  PhoneNo   |
+------+----------------+------------+------------+------------+
|  1   |   Dhruv Shah   | 2003-05-20 | 2018-09-12 | 9111006969 |
|  2   | Anirudh Satish | 2003-02-11 | 2018-09-23 | 9969666969 |
|  3   |  Darsh Rawat   | 2004-01-23 | 2019-08-02 | 9642069420 |
|  4   |   Harsh Raja   | 2004-06-12 | 2019-12-12 | 9910069691 |
+------+----------------+------------+------------+------------+

Press enter to continue..

Enter the E_ID : 1


  E_ID  Ename       DOB         Join_Date       PhoneNo
------  ----------  ----------  -----------  ----------
     1  Dhruv Shah  2003-05-20  2018-09-12   9111006969

Enter the field name you would like to modify
1. E_ID
2. Ename
3. DOB
4. Join_Date
5. PhoneNo
Ename
Enter new value :Dhruv R Shah
successfully updated

Press enter to continue..
```

Viewing employees after modification

```
EMPLOYEE

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 1


employee
+------+----------------+------------+------------+------------+
| E_ID |     Ename      |    DOB     | Join_Date  |  PhoneNo   |
+------+----------------+------------+------------+------------+
|  1   |  Dhruv R Shah  | 2003-05-20 | 2018-09-12 | 9111006969 |
|  2   | Anirudh Satish | 2003-02-11 | 2018-09-23 | 9969666969 |
|  3   |  Darsh Rawat   | 2004-01-23 | 2019-08-02 | 9642069420 |
|  4   |   Harsh Raja   | 2004-06-12 | 2019-12-12 | 9910069691 |
+------+----------------+------------+------------+------------+

Press enter to continue..
```

Notice how Ename of record with E_ID 1 has been modified from “Dhruv Shah” to “Dhruv R Shah”

Deleting employees

```
EMPLOYEE

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 4


employee
+------+----------------+------------+------------+------------+
| E_ID |     Ename      |    DOB     | Join_Date  |  PhoneNo   |
+------+----------------+------------+------------+------------+
|  1   |  Dhruv R Shah  | 2003-05-20 | 2018-09-12 | 9111006969 |
|  2   | Anirudh Satish | 2003-02-11 | 2018-09-23 | 9969666969 |
|  3   |  Darsh Rawat   | 2004-01-23 | 2019-08-02 | 9642069420 |
|  4   |   Harsh Raja   | 2004-06-12 | 2019-12-12 | 9910069691 |
+------+----------------+------------+------------+------------+

Press enter to continue..
Enter the E_ID : 4

Record deleted successfully

Press enter to continue...
```

Viewing employees after deletion

```
EMPLOYEE

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 1


employee
+------+----------------+------------+------------+------------+
| E_ID |     Ename      |    DOB     | Join_Date  |  PhoneNo   |
+------+----------------+------------+------------+------------+
|  1   |  Dhruv R Shah  | 2003-05-20 | 2018-09-12 | 9111006969 |
|  2   | Anirudh Satish | 2003-02-11 | 2018-09-23 | 9969666969 |
|  3   |  Darsh Rawat   | 2004-01-23 | 2019-08-02 | 9642069420 |
+------+----------------+------------+------------+------------+

Press enter to continue..
```

Notice record with E_ID 1 deleted

Orders menu

```
ORDERS

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Order delivered
5. Delete order
6. Go back : _
```

Adding orders

```
Enter the option you want to access:
1. View
2. Add
3. Modify
4. Order delivered
5. Delete order
6. Go back : 2
Enter no of records that need to be added : 4
enter the value of Pno:1
enter the value of Qty:10

records added successfully!

Press enter to continue..

enter the value of Pno:2
enter the value of Qty:5

records added successfully!

Press enter to continue..

enter the value of Pno:2
enter the value of Qty:6

records added successfully!

Press enter to continue..

enter the value of Pno:1
enter the value of Qty:5

records added successfully!

Press enter to continue..
```

View orders menu

```
Enter the table you would like to view :
1. Pending Orders
2. Completed Orders
3. Go back : _
```

Viewing pending orders after addition

```
Enter the table you would like to view :
1. Pending Orders
2. Completed Orders
3. Go back : 1


orders
+-----------+-----+-------+-----+-------+---------------+
| InvoiceNo | Pno | Price | Qty | Total | Date_of_order |
+-----------+-----+-------+-----+-------+---------------+
|     1     |  1  | 12.0  | 10  |  120  |  2021-03-13   |
|     2     |  2  | 15.0  |  5  |  75   |  2021-03-13   |
|     3     |  2  | 15.0  |  6  |  90   |  2021-03-13   |
|     4     |  1  | 12.0  |  5  |  60   |  2021-03-13   |
+-----------+-----+-------+-----+-------+---------------+

Press enter to continue..
```

Notice 4 pending orders added

Modifying and deleting orders – It is the same as it is for adding and deleting products and employees.

Marking orders delivered

```
Enter the option you want to access:
1. View
2. Add
3. Modify
4. Order delivered
5. Delete order
6. Go back : 4


orders
+-----------+-----+-------+-----+-------+---------------+
| InvoiceNo | Pno | Price | Qty | Total | Date_of_order |
+-----------+-----+-------+-----+-------+---------------+
|     1     |  1  | 12.0  | 10  |  120  |  2021-03-13   |
|     2     |  2  | 15.0  |  5  |  75   |  2021-03-13   |
|     3     |  2  | 15.0  |  6  |  90   |  2021-03-13   |
|     4     |  1  | 12.0  |  5  |  60   |  2021-03-13   |
+-----------+-----+-------+-----+-------+---------------+

Press enter to continue..
Enter Invoice number of the order that has been delivered :1

successfully updated details!

Press enter to continue..
```

Viewing pending orders after marking delivered

```
Enter the table you would like to view :
1. Pending Orders
2. Completed Orders
3. Go back : 1


orders
+-----------+-----+-------+-----+-------+---------------+
| InvoiceNo | Pno | Price | Qty | Total | Date_of_order |
+-----------+-----+-------+-----+-------+---------------+
|     2     |  2  | 15.0  |  5  |  75   |  2021-03-13   |
|     3     |  2  | 15.0  |  6  |  90   |  2021-03-13   |
+-----------+-----+-------+-----+-------+---------------+

Press enter to continue..
```

Notice the record marked delivered been deleted from pending orders table

Viewing completed orders after marking delivered

```
Enter the table you would like to view :
1. Pending Orders
2. Completed Orders
3. Go back : 2

+------------+------------+-------+----------+-------+---------------+-------------------+
| Invoice no | Product No | Price | Quantity | Total | Date of order | Date of Deliviery |
+------------+------------+-------+----------+-------+---------------+-------------------+
|     1      |     1      | 12.0  |    10    |  120  |  2021-03-13   |    2021-03-13     |
+------------+------------+-------+----------+-------+---------------+-------------------+

Press enter to continue..
```

Notice the recorded has been added to completed orders table

Viewing products after marking order delivered

```
PRODUCTS

Enter the option you want to access:
1. View
2. Add
3. Modify
4. Delete
5. Go back : 1


products
+-----+-------+------+------+-------+------+--------+
| Pno | Pname |  CP  |  SP  | Stock | Sold | Profit |
+-----+-------+------+------+-------+------+--------+
|  1  | Milk  | 12.0 | 18.0 |  20   |  0   |  6.0   |
|  2  | Bread | 15.0 | 30.0 |   8   |  0   |  15.0  |
+-----+-------+------+------+-------+------+--------+

Press enter to continue..
```

The stock of the product has been updated in products table as well after marking delivered

Data menu

```
DATA

Select the option you want to access:
1. Bills
2. Profits
3. Go back :
 1
```

Viewing bills of specific date

Entering date and bill no

```
Bills

Enter date of the bill[yyyy-mm-dd]: 2021-03-13

+--------+------------+-------+
| BillNo |    Date    | Total |
+--------+------------+-------+
|   1    | 2021-03-13 | 114.0 |
|   2    | 2021-03-13 | 132.0 |
+--------+------------+-------+

Enter bill no : 2
```

Bill is displayed

```
BILL NO :2

Date : 2021-03-13
Time : 23:49

  Product No  Product name      Price    Qty    Amount
 ----------- -------------- ------- ----- --------
           1  Milk                18      4        72
           2  Bread               30      2        60

 Total : 132.0

Press enter to continue...
```

Profit statistics

```
+------------+--------------+---------------+--------+--------------+
| Product No | Product Name | Quantity Sold | Profit | Total profit |
+------------+--------------+---------------+--------+--------------+
|     1      |     Milk     |       7       |  6.0   |     42.0     |
|     2      |    Bread     |       4       |  15.0  |     60.0     |
+------------+--------------+---------------+--------+--------------+

Total profit on all the products :  102.0

Press enter to continue..
```

Billing system

```
BILLING SYSTEM



products
+-----+-------+------+------+-------+------+--------+
| Pno | Pname |  CP  |  SP  | Stock | Sold | Profit |
+-----+-------+------+------+-------+------+--------+
|  1  | Milk  | 12.0 | 18.0 |  20   |  0   |  6.0   |
|  2  | Bread | 15.0 | 30.0 |   8   |  0   |  15.0  |
+-----+-------+------+------+-------+------+--------+

Press enter to continue..


Enter product number : 1
Enter quantity : 3

Press enter to continue and any other key to stop bill..

Enter product number : 2
Enter quantity : 2

Press enter to continue and any other key to stop bill.. 2
```

Bill confirmation

```
BILL NO : 1

Date : 2021-03-13
Time : 23:47

  Product No  Product name      Price    Qty    Amount
------------  --------------  -------  -----  --------
           1  Milk                 18      3        54
           2  Bread                30      2        60

TOTAL : 114.0

Please confirm if the bill is correct[y/n] : yes
Bill generated successfully!

Press enter to continue billing, any other key to exit to main menu..
```

Bill confirmed

```
BILL NO : 1

Date : 2021-03-13
Time : 23:47

  Product No  Product name      Price    Qty    Amount
------------  --------------  -------  -----  --------
           1  Milk                 18      3        54
           2  Bread                30      2        60

TOTAL : 114.0

Please confirm if the bill is correct[y/n] : yes
Bill generated successfully!

Press enter to continue billing, any other key to exit to main menu..
```

Bill Rejected

```
BILL NO : 2

Date : 2021-03-13
Time : 23:48

  Product No  Product name      Price    Qty    Amount
------------  --------------  -------  -----  --------
           1  Milk                 18      6       108
           2  Bread                30      3        90

TOTAL : 198.0

Please confirm if the bill is correct[y/n] : no

Bill cancelled and discarded!

Press enter to continue billing, any other key to exit to main menu..
```

End screen

```
Process returned 0 (0x0)        execution time : 357.107 s
Press any key to continue . . .
```

CONCLUSION

Supermarkets have large inflow of data on an everyday basis and it becomes very hard and laborious task to handle and manage this data. Although there are many supermarket management softwares already in the market, these are often very complicated and hard to use apps. These are often expensive and not viable for small business. In this situation our supermarket management software is a very compact inexpensive, easy to use and navigate tool that works on almost every system with the most basic specs. This makes it suitable for both large and small businesses. It performs all the tasks any other such software would.

Although a very useful and simple tool it does have a few shortcomings such as minimal security, no internet data backups and moreover due to its simple structure and usage of python and mysql for data storage and processing it can easily be hacked or reprogrammed to manipulate data.

We are working on these flaws to make sure we can come up with the best software for serving the supermarket owners' needs. This is still a very useful and powerful tool to cater your supermarket management needs.

BIBLIOGRAPHY

- Sumit Arora's Computer science with python for class XI and class XII
- https://docs.python.org/3/library/datetime.html
- https://docs.python.org/3/library/os.html
- https://pypi.org/project/tabulate/
- https://pypi.org/project/art/