**PROJECT REPORT**

**SUPERMARKET BILLING SYSTEM**

***SUBMITTED BY***

**Mrinalini Vaish [RA2211026010365]**

**Devasye Sachdeva [RA2211026010367]**

**Himanshu Bhadani [RA2211026010368]**

***from***

**Branch – Computer Science Engineering with specialization in Artificial Intelligence and Machine Learning**

**Section – W1**

***Under the guidance of***

**Dr. V. V. Ramalingam**

**Associate Professor**

**Department of Computing Technologies**

***In partial satisfaction of the requirements for the degree of***

**BACHELOR OF TECHNOLOGY IN**

**COMPUTER SCIENCE ENGINEERING**

Logo, company name

Description automatically generated

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKALUTHUR – 603203

Logo, company name

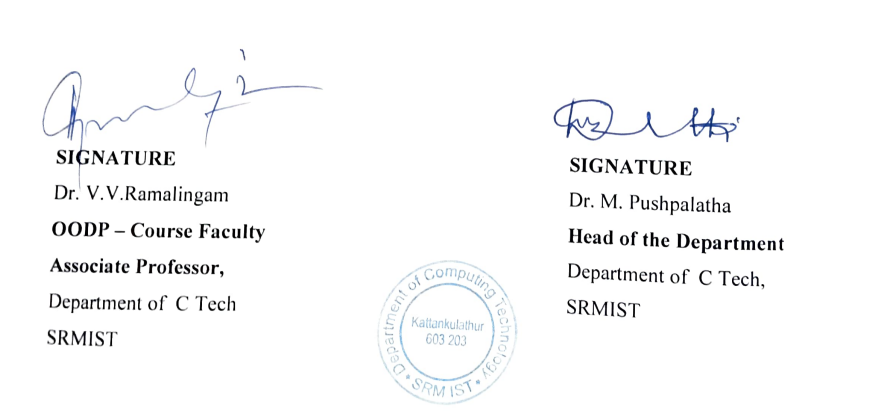
Description automatically generated

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKALUTHUR – 603203

**BONAFIDE CERTIFICATE**

Certified that this Project Report titled “SUPERMARKET BILLING SYSTEM” is the bonafide work done by Devasye Sachdeva [RA2211026010367] who completed the project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form part of any other work.



**ABSTRACT**

The invoicing system for the store includes automation in the supermarket. This software will help vendors manage various types of information about customers. This product will help users work in a good environment. It requires a lot of time and energy that could be used for better work. In addition, as the power of customers increases, managing each customer’s information becomes tedious. In the guide, sellers face many inefficiencies. Data storage is one of the most important issues. Writing the perfect user experience report is difficult. Large files should be stored where important and irrelevant data should be stored. Automate all these problems and try to eliminate them in the most appropriate way. The new system will meet every market seller’s needs so that the seller can manage the system effectively. The purpose of the development of the supermarket is to make the system more reliable, easier, faster, and richer in information.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **DESCRIPTION** | **PAGE NO.** |
|  | INTRODUCTION | 1 – 4 |
|  | PROBLEM STATEMENT | 5 |
|  | AIM OF THE PROJECT | 6 |
|  | OBJECT-ORIENTED PROGRAMMING CONCEPT | 7 – 8 |
|  | FILE CONCEPT | 9 |
|  | SYSTEM SPECIFICATIONS | 10 |
|  | MODULES | 11 |
|  | UML DIAGRAMS | 12 – 19 |
|  | CODING | 20 – 27 |
|  | OUTPUT |  |
|  | CONCLUSION |  |

**INTRODUCTION**

Supermarket billing system is a system to automate the process of ordering and billing a supermarket store. The supermarket is the place where customers come to purchase their daily use products and pay for them. So, there is a need to calculate how many products are sold and to generate the bill for the customer. This system is built for fast data processing and bill generation for supermarket customers. It also allows the customer to purchase and pay for the items purchased. The users will consume less time in calculation and the sales activity will be completed within a fraction of a second whereas a manual system will make the user write it down which is a long procedure, and it also consumes a lot of time. Because of this software, paperwork will be reduced, and the user can spend more time on monitoring the supermarket. The project will be user-friendly and easy to use. This project is helpful to computerize the bill report and generate the details of the item. The billing data is a vast collection of product names, prices, and other product-specific data. A product when billed is searched and its price is added to the bill based on the product quantity. The system also contains discounts on various products so that the product is offered at discounted price while billing. The supermarket billing system is built to help supermarkets calculate and display bills and serve the customer in a faster and more efficient manner. This software project consists of an effective and easy GUI to help the employees with easy bill calculation and provide efficient customer service. With the continuous development and improvement of computer, communication, network, and scale database technology, the commercial supermarket has become a developing technology worldwide.

FEASIBILITY STUDY

A feasibility study is a process of analyzing and identifying if a problem can be solved or not solved, focusing on helping answer the essential question of “Should we continue the proposed project ideas?” All activities of the study are directed toward helping answer this question. A Feasibility Study is generic in nature and can be applied to any type of project, be it for systems and software development, acquiring, or any other project. A feasibility study is a test of the system according to its workability, impact on the organization, ability to meet user needs, and effective use of the resources. We can test our system by different types of feasibilities. There are basically six parts to any effective Feasibility Study:

1. The Project Scope – Used to define the business problem and/or opportunity to be addressed. The adage, "The problem well stated is half solved," is very apropos. The scope should be definitive and to the point; a rambling narrative serves no purpose and can confuse project participants.
2. The Current Analysis –used to define and understand the current method of implementation, such as a system, a product, etc. From this analysis, it is not uncommon to discover there is nothing wrong with the current system or product other than some misunderstandings regarding it or perhaps it needs some simple modifications as opposed to a major overhaul.
3. Requirements - how requirements are defined depends on the object of the project's attention. For example, how requirements specified for a product are substantially different from requirements for an edifice, a bridge, or an information system.
4. The Approach- represents the recommended solution or course of action to satisfy the requirements.
5. Evaluation - examines the cost-effectiveness of the approach selected. This begins with an analysis of the estimated total cost of the project.
6. Review - all of the preceding elements are then assembled into a Feasibility Study and a formal review is conducted with all parties involved.

There are 5 types of feasibilities –

1. Technical Feasibility

A study of resource availability that may affect the ability to achieve an acceptable system. This evaluation determines whether the technology needed for the proposed system is available or not. This system can be made in any language that supports a good user interface and easy database handling.

Front-End Selection: A language that is used for user interface designing and coding. Front-End should have the following qualities:

1. It must have a graphical user interface that assists employees that are not from some IT background.
2. Scalability and Extensibility
3. Robustness
4. According to the organization’s requirements and culture.
5. Must provide excellent reporting features with a good printing support platform independently.
6. Easy to deploy and maintain.
7. Event-driven programming.
8. Economic Feasibility

The cost to conduct a full system investigation. The cost of hardware and software for the class of application being considered. The benefit is in the form of reduced cost. Our system has a lot of features at a minimum cost so it is feasible to implement, and it will be very beneficial to the sellers in the reduced cost. Its software and hardware costs are also low than the existing system.

1. Operational Feasibility

In the new system, we made some major changes for the staff members so that they have to be trained to use the newly added facilities. These major changes are possible and give a new era in the Supermarket in production and sales management.

1. Schedule Feasibility

Time evaluation is the most important consideration in the development of the project. So, the project is concerned should be completed at a fixed scheduled time as far as the company is concerned. The new system is not so big so it is easy to make in a few days.

SCOPE OF THE WORK

The scope of this project is to investigate and design a software solution that can facilitate both customers and salespersons in performing their daily tasks, improving efficiency, and helping them to be more productive. This project will provide a solution through which salespersons can easily manage, handle and generate all required information in their respective formats when needed. It provides a quick way of operation by capturing the manual process and automating them. It will help them to manage the bill details, financial data, and historical data and also in producing documents of different formats for different customers. This solution will help salespersons in reducing the effort spent on managing many bills. It will also provide them the opportunity to explore the possibility of generating documents and managing financial details. This system will help the salesperson to manage fast billing. It will help to maintain the data of the purchased items. It also gives bills to the customers. It will set the rates of taxes and commissions on products. The project will enable you to see the report regarding products and categories in a fixed period. It can also change the Graphical User Interface of the system.

**PROBLEM STATEMENT**

In today’s fast-paced society, it’s very hard to be competitive without using cutting-edge technology available in the market. After years of business, the data has grown much. It is becoming a challenge for people to manage that data in an effective way. To be more productive in order processing, he needs a solution that can facilitate their current processes with the use of technology and software. With the increased number of orders, it is becoming difficult for salespeople to manage orders in an effective and efficient manner. It is very hard to go through all paperwork and backtracking orders. If there is any complaint or review of any order, it takes a large amount of effort and time to backtrack and fix the problem. This results in a loss of resources, increased time, and low output.

Drawbacks of Manual System (Current System):

1. Time-consuming: Getting the required information from the available data takes a lot of time. Changing, editing, and updating the information contained in several files is a slow and time-consuming process.
2. Poor communication: A manual system requires employees and managers to write down each time an item is removed from the inventory. If one employee forgets to mention that the last coffee product has been removed from the inventory the admin or manager expects the item to still be available for a customer during sale.
3. Need of Effort: In a manual system, an item’s record is maintained in separate files, so it takes much effort to collect data from several stores and if we want to change or delete the data of any transaction then it has to be changed or deleted from all the files and places it stored.
4. Needs Large Space: In manual work done data item has to be stored in several places, similarly student record is maintained in separate registers. It requires more storage space.

**AIM OF THE PROJECT**

* To reduce the time for organization.
* To increase the efficiency and accuracy of the system.
* To reduce pressure on the labor and relieve manpower from repetitive jobs.
* To make the retrieval of information faster.
* To make the system more feasible.
* To reduce the large amount of paperwork.
* To reduce the cost factor of the system.
* To provide convenient solution of billing pattern.
* To make the system more flexible.

EXISTING SYSTEM

The existing billing system does the same work that is calculating bills, giving them to customers, and maintaining a database. It is concerned with the requirements and gives them more commission.

1. Graphical User-interface is not so good.
2. Processing speed is slow.
3. Not that flexible.
4. Not able to generate automatic reports.
5. Sometimes the system hangs due to workload.

PROPOSED SYSTEM

It is intended to provide the facility of automating the billing system for the supermarket. This project is designed with the goal to make the existing system more informative, reliable, faster, and easier. It converts manual work to a fully automated system. It helps in eliminating the paperwork. C++ has supported to implement object-oriented concepts.

**OBJECT-ORIENTED PROGRAMMING CONCEPT**

The main aim of OOP is to bind together the data and the functions that operate on them so that no other part of the code can access this data except that function. It allows the decomposition of a problem into several entities called objects.

Basic concepts –

1. Class – It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class.
2. Object – An Object is an identifiable entity with some characteristics and behavior. An Object is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e., an object is created) memory is allocated.
3. Encapsulation – It is defined as binding together the data and the functions that manipulate them. Encapsulation also leads to data abstraction or data hiding. Using encapsulation also hides the data.
4. Abstraction – It means displaying only essential information and hiding the details. Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation.
5. Abstraction using Classes: The class helps us to group data members and member functions using available access specifiers.
6. Abstraction in Header files
7. Polymorphism – The ability of a message to be displayed in more than one form.
8. Operator Overloading: The process of making an operator exhibit different behaviors in different instances is known as operator overloading.
9. Function Overloading: Function overloading is using a single function name to perform different types of tasks. Polymorphism is extensively used in implementing inheritance.
10. Inheritance – The capability of a class to derive properties and characteristics from another class is called inheritance.
11. Sub Class: The class that inherits properties from another class is called Sub class or Derived Class.
12. Super Class: The class whose properties are inherited by a sub-class is called Base Class or Superclass.
13. Reusability: When we want to create a new class and there is already a class that includes some of the code that we want, we can derive our new class from the existing class. By doing this, we are reusing the fields and methods of the existing class.
14. Dynamic Binding – In dynamic binding, the code to be executed in response to the function call is decided at runtime. C++ has [virtual functions](https://www.geeksforgeeks.org/virtual-functions-and-runtime-polymorphism-in-c-set-1-introduction/) to support this. Because dynamic binding is flexible, it avoids the drawbacks of static binding, which connected the function call and definition at build time.
15. Message Passing – A message for an object is a request for the execution of a procedure and therefore will invoke a function in the receiving object that generates the desired results. Message passing involves specifying the name of the object, the name of the function, and the information to be sent.

**FILE CONCEPT**

The I/O system of C++ handles file operations which are very much like the console input and output operations. It uses file streams as an interface between the programs and files. The stream that supplies data to the program is called the input stream and the one that receives data from the program is called the output stream. The input operation involves the creation of an input stream and linking it with the program and input file. Similarly, the output operation involves establishing an output stream with the necessary links with the program and output file.

* filebuf : Its purpose is to set the file buffers to read and write. Contains Open prompt constant used in the open() of file stream classes. Also, contain close() and open() as members.
* fstreambase: Provides operations common to file streams.Serves as a base for fstream, ifstream, and ofstream classes. Contains open() and close() functions.
* ifstream: Provides input operations. Contains open() with default input mode. Inherits the functions get(),getline(),read(),seekg(),tellg() functions from istream.
* ofstream: Provides output operations. Contains open() with default output mode. Inherits put(),seekp(),tellp() and write() functions from ostream.
* fstream: Provides support for simultaneous input and output operations. Contains open with default input mode. Inherits all the functions from istream and ostream classes through iostream.

File Mode Operation

|  |  |
| --- | --- |
| PARAMETERS | MEANING |
| ios::app | Append to end of file |
| ios::ate | Go to the end of the file on opening |
| ios::binary | Binary file |
| ios::in | Open file for reading only |
| ios::nocreate | Open fails if the file does not exist |
| ios::noreplace | Open fails if the file already exists |
| ios::out | Open file writing only |
| ios::trunk | Delete the contents of the file |

**MODULES**

There are five modules in this project.

1. Login Id

This module is made for the login of users. We have three user login id:

1. Administrator
2. Data Entry Operator
3. Bill Calculating Operator
4. Apply Taxes and Commissions

This module is for the administrator who will do the following work:

1. Set the taxes for the products.
2. Set the commission for the products.
3. Check the Report

This module is also for the administrator who can generate or check the report of the product and also see how many products are sold on a particular date or in a specific period of time.

1. Enter the information about products.

This module is for the data entry operator who will do the following work:

1. Enter which products come in the store.
2. Enter the prices and expiry date of the product.
3. Calculate the bill.

This module is for bill-calculating operator who will do the following work:

1. Calculate the bill.
2. Print the bill.

**UML DIAGRAMS**

1. USE CASE DIAGRAM

**Diagram

Description automatically generated**

1. CLASS DIAGRAM

Diagram

Description automatically generated

1. SEQUENCE DIAGRAM

Diagram

Description automatically generated

1. STATE CHART DIAGRAM

**Diagram

Description automatically generated**

1. ACTIVITY DIAGRAM

Diagram

Description automatically generated

Diagram

Description automatically generated

1. PACKAGE DIAGRAM

Diagram

Description automatically generated

1. COMPONENT DIAGRAM

Diagram

Description automatically generated

1. DEPLOYMENT DIAGRAM

Diagram

Description automatically generated

**CODING**

#include <iostream>

#include <fstream>

using namespace std;

class shopping

{

    private:

        int pcode;

        float price;

        float dis;

        string pname;

    public:

        void menu();

        void administrator();

        void buyer();

        void add();

        void edit();

        void rem();

        void list();

        void receipt();

};

void shopping::menu()

{

    m:

    int choice;

    string email;

    string password;

    cout<<"\t\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    cout<<"\t\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    cout<<"\t\t\t\t                 Supermarket main menu            \n";

    cout<<"\t\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    cout<<"\t\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    cout<<"\t\t\t\t|  1) Administrator     |\n";

    cout<<"\t\t\t\t|                       |\n";

    cout<<"\t\t\t\t|  2) Buyer             |\n";

    cout<<"\t\t\t\t|                       |\n";

    cout<<"\t\t\t\t|  3) Exit              |\n";

    cout<<"\n\t\t\t     Please Select!";

    cin>>choice;

    switch (choice)

    {

        case 1:

        cout<<"\t\t\t Please login \n";

cout<<"\t\t\t Enter email   \n";

        cin>>email;

        cout<<"\t\t\t Password      \n";

        cin>>password;

        if(email=="robby@email.com" && password=="robby@123")

        {

            administrator();

        }

        else{

            cout<<"Invalid email/password";

        }

        break;

    case 2:

    {

        buyer();

    }

    case 3:

    {

        exit(0);

    }

    default:{

        cout<<"Please select from the given options";

    }

    goto m;}

}

void shopping:: administrator()

{

    m:

    int choice;

    cout<<"\n\n\n\\t\t\t Administrator menu";

    cout<<"\n\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1)Add the product\_\_\_\_\_\_";

    cout<<"\n\t\t\t                                     ";

    cout<<"\n\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2)Modify the product\_\_\_\_\_\_";

    cout<<"\n\t\t\t                                      ";

    cout<<"\n\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3)Delete the product\_\_\_\_\_\_";

    cout<<"\n\t\t\t                                      ";

    cout<<"\n\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4) Back to main menu\_\_\_\_\_\_";

    cout<<"\n\t\t\t Please enter your choice";

    cin>>choice;

    switch(choice)

    {

        case 1:

            add();

            break;

        case 2:

            edit();

            break;

        case 3:

            rem();

            break;

        case 4:

            menu();

            break;

        default:

        cout<<"Invalid choice!";

    }

    goto m;

}

void shopping:: buyer()

{

    m:

    int choice;

    cout<<"\t\t\t  Buyer \n";

    cout<<"\t\t\t                \n";

    cout<<"\t\t\t1) Buy product\n";

    cout<<"\t\t\t                \n";

    cout<<"\t\t\t2) Go back      \n";

    cout<<"\t\t\t Enter your choice: ";

    cin>>choice;

    switch(choice)

    {

        case 1:

            receipt();

            break;

        case 2:

            menu();

        default:

            cout<<"Invalid choice";

    }

    goto m;

}

void shopping::add()

{

    m:

    fstream data;

    int c;

    int token=0;

    float p;

    float d;

    string n;

    cout<<"\n\n\t\t\t Add new product";

    cout<<"\n\n\t\t\t Product code of the product";

    cin>>pcode;

    cout<<"\n\n\t\t\t Name of the product";

    cin>>pname;

    cout<<"\n\n\t\t\t Price of the product";

cin>>price;

    cout<<"\n\n\t\t\t Discount on product";

    cin>>dis;

    data.open("database.txt",ios::in);

    if(!data)

    {

        data.open("database.txt",ios::app|ios::out);

        data<<"  "<<pcode<<"  "<<pname<<"  "<<price<<"  "<<dis<<"\n";

        data.close();

    }

    else{

        data>>c>>n>>p>>d;

        while(!data.eof())

        {

            if(c==pcode)

            {

                token++;

            }

            data>>c>>n>>p>>d;

        }

        data.close();

    }

    if(token==1)

        goto m;

    else{

        data.open("database.txt",ios::app|ios::out);

        data<<"  "<<pcode<<"  "<<pname<<"  "<<price<<"  "<<dis<<"\n";

        data.close();

    }

    cout<<"\n\n\t\t\t Record Inserted";

}

void shopping::edit()

{

    fstream data,data1;

    int pkey;

    int token=0;

    int c;

    float p;

    float d;

    string n;

    cout<<"\n\n\n\t\t Modify the record";

    cout<<"\n\n\n\t\t Product code:";

    cin>>pkey;

    data.open("database.txt",ios::in);

    if(!data)

    {

        cout<<"\n\nFIle doesnt exist";

    }

    else{

        data1.open("database1.txt", ios::app|ios::out);

        data>>pcode>>pname>>price>>dis;

        while(!data.eof())

        {

            if(pkey==pcode)

            {

                cout<<"\n\n\t\t Product new code:";

                cin>>c;

                cout<<"\n\n\t\t Name of the product";

                cin>>n;

                cout<<"\n\n\t\t price : ";

                cin>>p;

                cout<<"\n\n\t\t Discount: ";

                cin>>d;

                data1<<"  "<<c<<" "<<n<<" "<<p<<" "<<d<<"\n";

                cout<<"\n\n\n\t\t\t Record edited ";

                token++;

            }

            else {

                data1<<" "<<pcode<<" "<<pname<<" "<<price<<" "<<dis<<"\n";

            }

            data>>pcode>>pname>>price>>dis;

        }

        data.close();

        data1.close();

        remove("database.txt");

        rename("database1.txt","database.txt");

        if(token==0)

        {

            cout<<"\n\n\t\t Record not found sorry!";

        }

    }

}

void shopping::rem()

{

    fstream data,data1;

    int pkey;

    int token=0;

    cout<<"\n\n\n\t Delete Product";

    cout<<"\n\n\n\t Product code";

    cin>>pkey;

    data.open("database.txt",ios::in);

    if(!data)

    {

        cout<<"File doesnt exist";

    }

    else{

data1.open("database1.txt",ios::app|ios::out);

        data>>pcode>>pname>>price>>dis;

        while(!data.eof())

        {

            if(pcode==pkey)

            {

                cout<<"\n\n\t Product deleted succesfully";

                token++;

            }

            else{

                data1<<" "<<pcode<<" "<<pname<<" "<<price<<" "<<dis<<"\n";

            }

            data>>pcode>>pname>>price>>dis;

        }

        data.close();

        data1.close();

        remove("database.txt");

        rename("database1.txt","database.txt");

        if(token==0)

        {

            cout<<"\n\n\n Record not found";

        }

    }

}

void shopping::list()

{

    fstream data;

    data.open("database.txt",ios::in);

    cout<<"\n\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    cout<<"ProNo\t\tName\t\tPrice\n";

    cout<<"\n\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    data>>pcode>>pname>>price>>dis;

    while(!data.eof())

    {

        cout<<pcode<<"\t\t"<<pname<<"\t\t"<<price<<"\n";

        data>>pcode>>pname>>price>>dis;

    }

    data.close();

}

void shopping::receipt()

{

    m:

    fstream data;

    int arrc[100];

    int arrq[100];

    string choice;

    int c=0;

    float amount=0;

    float dis=0;

    float total=0;

    cout<<"\n\n\t\t\t RECEIPT";

data.open("database.txt",ios::in);

    if(!data)

    {

        cout<<"n\n\n Empty database";

    }

    else{

        data.close();

        list();

        cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

        cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

        cout<<"\n            Place the order            \n";

        cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

        cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

        do

        {

            cout<<"\n\n Enter product code:";

            cin>>arrc[c];

            cout<<"\n\nEnter the product quantity:";

            cin>>arrq[c];

            for(int i=0;i<c;i++)

            {

                if(arrc[c]==arrc[i])

                {

                    cout<<"Duplicate product code. Please try again!";

                    goto m;

                    }

                }

                c++;

                cout<<"\n\n Do you want to buy another product? If yes then press y else no";

                cin>>choice;

            }

            while(choice =="y");

            cout<<"\n\n\n\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_RECEIPT\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

            cout<<"\nProduct no\t Product name \t product quantity\t price\t Amount\t Amount with discount\n";

            for(int i=0;i<c;i++)

            {

                data.open("database.txt",ios::in);

                data>>pcode>>pname>>price>>dis;

                while(!data.eof())

                {

                    if(pcode==arrc[i])

                    {

                        amount=price\*arrq[i];

                        dis=amount-(amount\*dis/100);

                        total=total+dis;

                        cout<<"\n"<<pcode<<"\t\t"<<pname<<"\t\t"<<arrq[i]<<"\t\t"<<price<<"\t\t"<<amount<<"\t\t\t"<<dis;

                    }

                    data>>pcode>>pname>>price>>dis;

                }

            }

            data.close();

        }

        cout<<"\n\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_";

     cout<<"n\n Total amount : "<<total;

 }

int main()

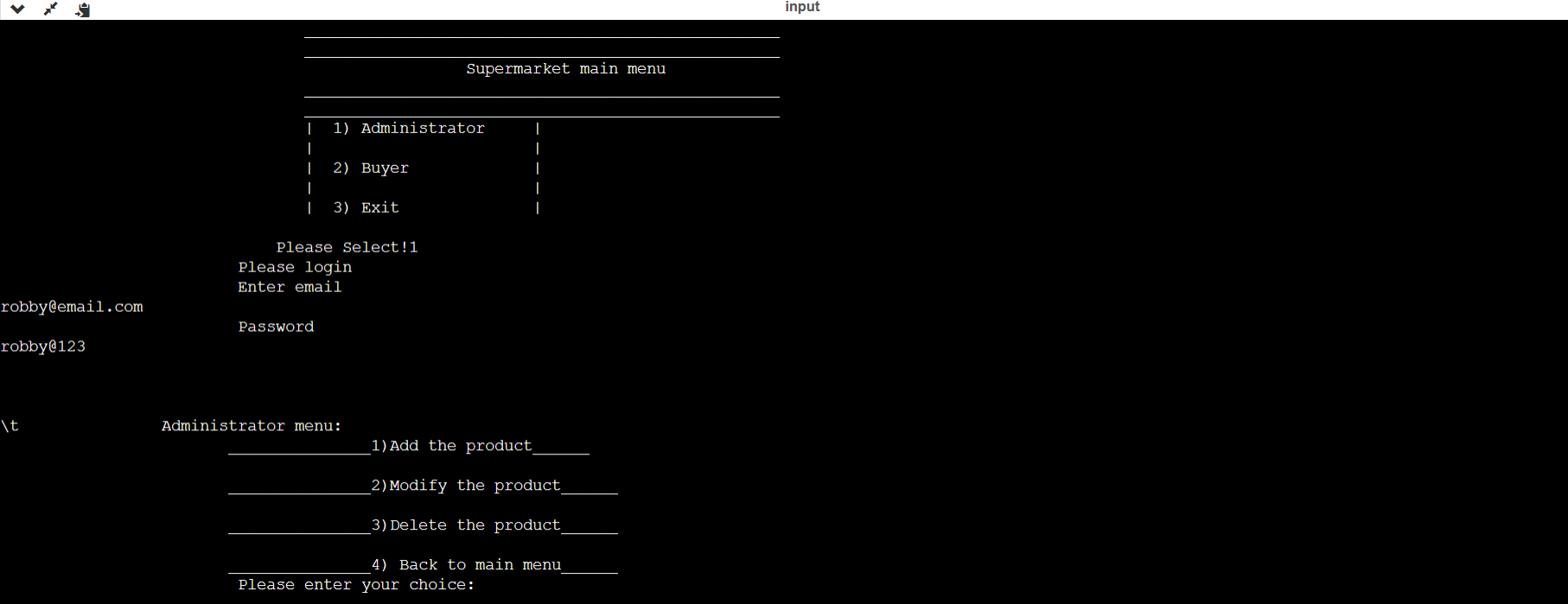
{

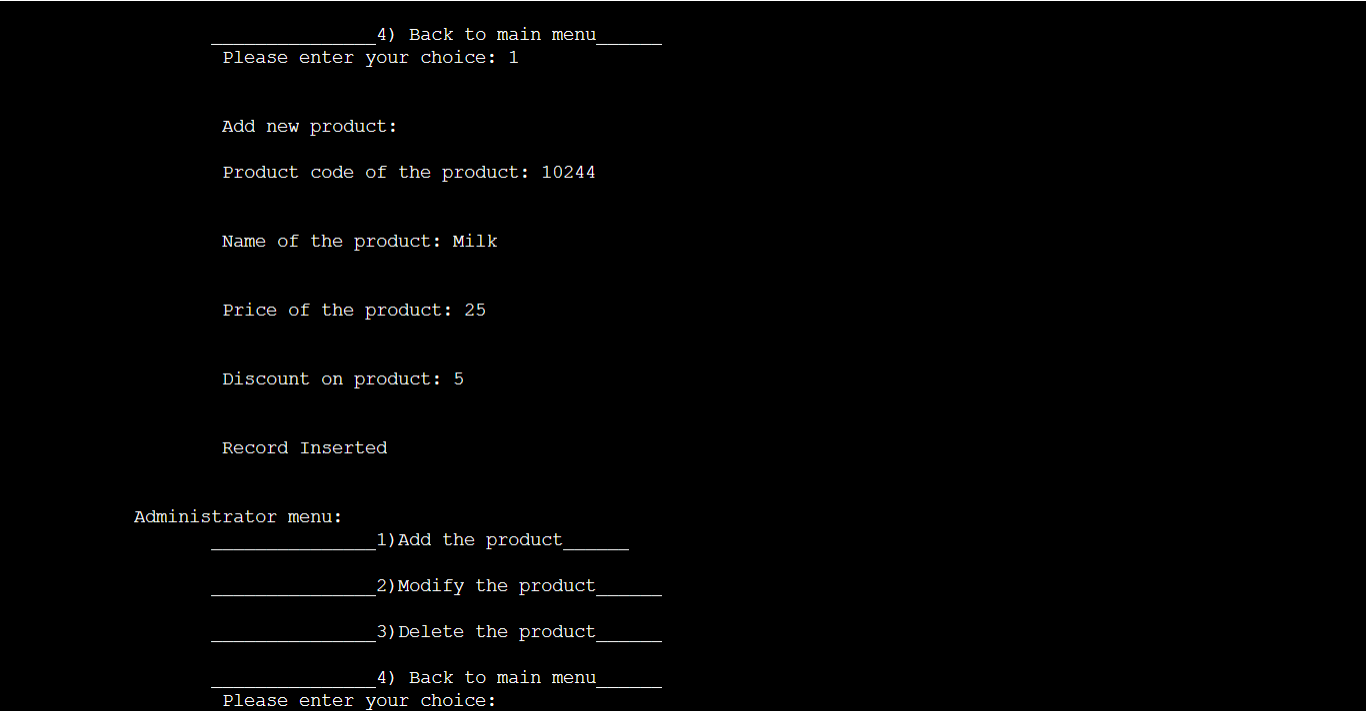
    shopping s;

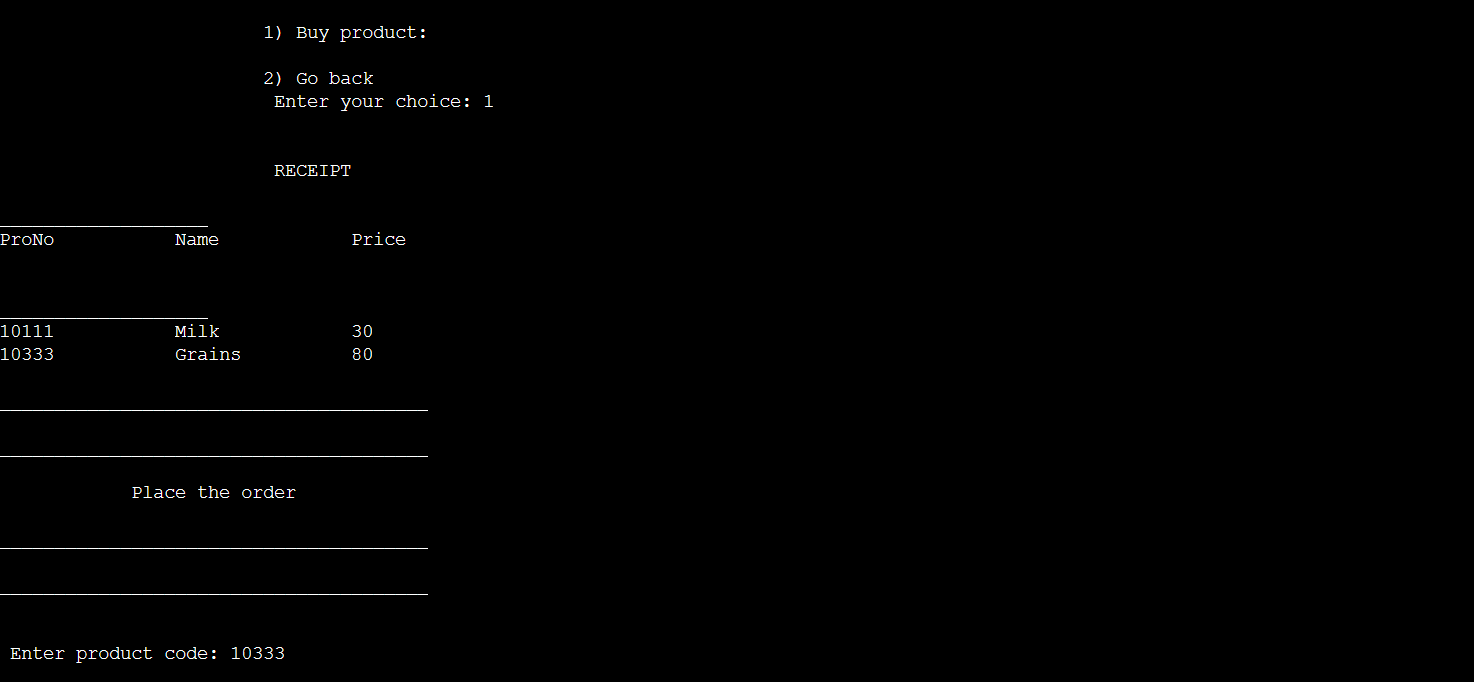
**s.menu();**

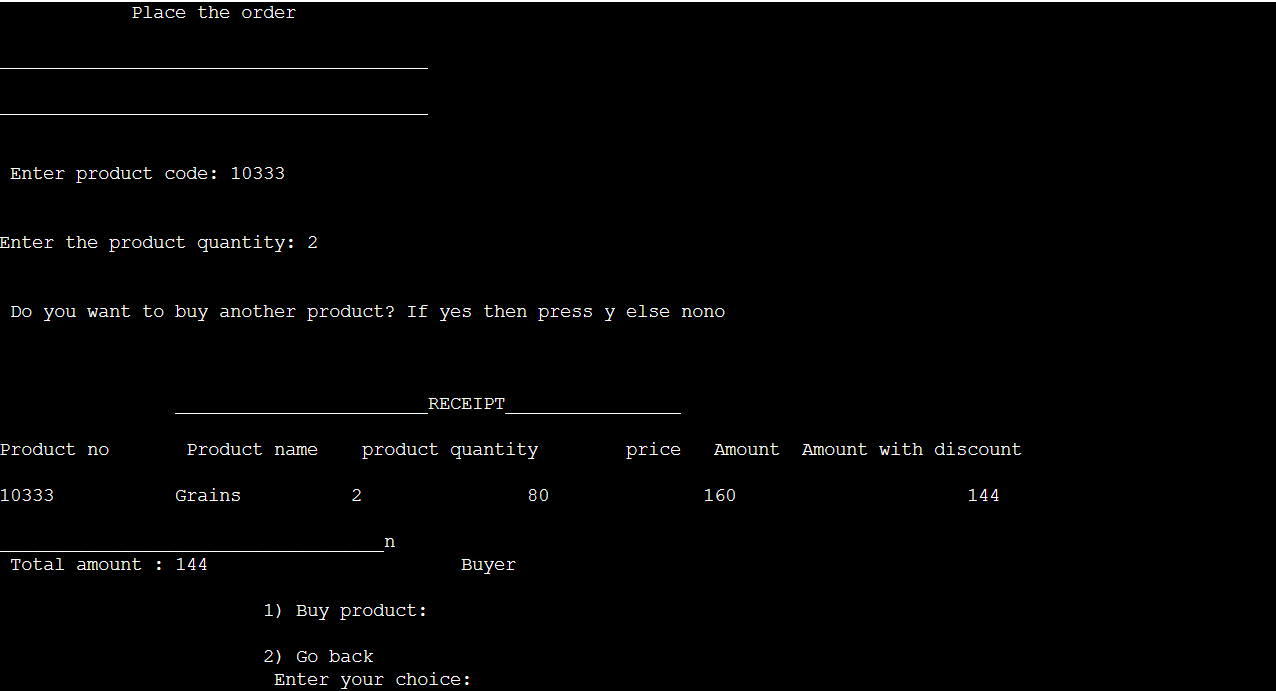
}

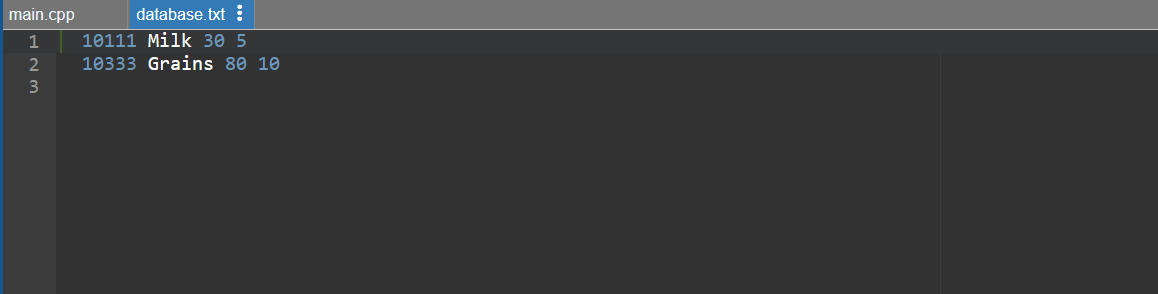
**OUTPUT**











**CONCLUSION**

This software has been designed to reduce the time taken to handle the sales activity. It is designed to replace an existing manual record system for reducing the time taken for calculations and for storing the data.

This system has been developed with the OOPs concept. The system is strong enough to handle daily operations where the database is cleared over certain time. This system will reduce the manual work, calculations, and will also provide periodic reports any time.

**References**

1. [ABSTRACT The Project "Supermarket" Deals With The | PDF | Software Development | System (scribd.com)](https://www.scribd.com/doc/27828662/ABSTRACT-the-Project-supermarket-Deals-With-The)

2. [Online Supermarket System | PDF | Object Relational Mapping | Software Development Process (scribd.com)](https://www.scribd.com/doc/37762263/Online-Supermarket-System)

3. [Super Market Management System UML Diagram | FreeProjectz](https://www.freeprojectz.com/uml-diagram/super-market-management-system-uml-diagram)

4. [GitHub - mahakbansal2019/SuperMarket-Management-System](https://github.com/mahakbansal2019/SuperMarket-Management-System)