VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Machhe, Belagavi-590018

A

Mini Project Report

On

"Supermarket Management System"

Submitted in partial fulfillment required for award of the Graduation Degree

Bachelor of Engineering

In

Computer Science and Engineering

5th Semester

17CSL58-DBMS Laboratory with Mini Project

Submitted by

Abdul Rahman A	**1HK17CS006**
Junaid Ahmed Baig	**1HK17CS060**

Under the guidance of

Prof. K. Sangeetha Supriya

Assistant Professor

Department of Computer Science & Engineering

DEC 2019

HKBK COLLEGE *of* ENGINEERING

Nagawara, Bangalore–560 045

Approved by AICTE & Affiliated to VTU

Department of Computer Science and Engineering

Certificate

Certified that the Project entitled **“Supermarket Management System”,** carried out by **Abdul Rahman A (1HK17CS006), Junaid Ahmed Baig (1HK17CS060)** are bonafide students of **HKBK COLLEGE *of* ENGINEERING**, in partial fulfillment for the award of **Bachelor of Engineering** in **Computer Science and Engineering** of the **Visvesvaraya Technological University**, Belagavi, during the year 2019–20. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of **17CSL58–Database Management Laboratory with Mini Project prescribed** for the said Degree.

Prof. K.Sangeetha Supriya	**Dr. Loganathan R**
Guide	Professor & HOD

EXTERNAL VIVA

Name of the Examiners	Signature with Date
1.______________________	______________________
2.______________________	______________________

ACKNOWLEDGEMENT

We would like to express our regards and acknowledgement to all who helped us in completing this project successfully.

First of all we would take this opportunity to express our heartfelt gratitude to **Mr. C M Ibrahim**, Chairman, HKBKGI and **Mr. Faiz Mohammed**, Director, HKBKGI for providing facilities throughout the course.

We express our sincere gratitude to our Principal, HKBKCE, for his support towards the attainment of knowledge .

We consider it as a great privilege to convey our sincere regards to **Dr. Loganathan. R.**, Professor and HOD, Department of CSE, HKBKCE for his constant encouragement throughout the course of the project.

We would specially like to thank our guide, **Prof. Sangeetha**, Assistant Professor, Department of CSE for her/his vigilant supervision and his/her constant encouragement. She/He spent her/his precious time in reviewing the project and provided many insightful comments and constructive criticism.

Finally, we thank Almighty, all the staff members of CSE Department, our family members and friends for their constant support and encouragement in carrying out the Project work.

1HK17CS006 Abdul Rahman A
1HK17CS060 Junaid Ahmed Baig

ABSTRACT

The project “Supermarket Management System” deals with the computerization of sales and transaction of items in a supermarket. The software used, helps the salesperson in managing the various types of records pertaining to his/her customer. The primary activity is to record the displayed items in a system along with their names and prices that the supermarket agrees to sell and this authority is solely given to the admin who can view the periodic activities any time for reference. Any further modifications to be made in the name and price of an item can only be done by the admin. The admin also provides each employee with a unique username and password through which they can log in. Every supermarket provides itself an authorized vendor for the supply of goods at the unavailability of stocks.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER -1
INTRODUCTION

Chapter 1

INTRODUCTION

1.1 Introduction:

Supermarket management system is the system where all the aspects related to the proper management of supermarket is done. These aspects involve managing information about the various products, staff, managers, customers, billing etc. This system provides an efficient way of managing the supermarket information. Also allows the customer to purchase and pay for the items purchased.

This project is based on the sales transaction and billing of items in a supermarket. The first activity is based on adding the items to the system along with the rate which are present in the supermarket and the name of the items which the supermarket will agree to sell. This authority is given only to admin (administrator). Any modifications to be done in the item name and the rate can be done only by admin. He also has the right to delete any item. As the customer buys the products and comes to the billing counter, the user is supposed to enter the item name he purchased and the quantity of the item he had purchased. This is not a huge a task.

This study is to produce software which manages the sales activity done in a supermarket, maintaining the stock details, maintaining the records of the sales done for a particular month/year. The users will consume less time in calculation and the sales activity will be completed within a fraction of seconds whereas manual system will make the user to write it down which is a long procedure and so paper work will be reduced and the user can spend more time on the monitoring the supermarket. The project will be user friendly and easy to use.

The system will display all the items whose name starts with the letter selected by the user. He can select out of those displayed. Finally a separate bill will be generated for each customer. This will be saved in the database. Any periodic records can be viewed at any time. If the stock is not available, the supermarket orders and buys from a prescribed vendor. The amount will be paid by deducting the total amount acquired in the sales activity. Admin provides a unique username and password for each employee through which he can login.

1.2 Problem Statements:

Problems in existing system:

- Time consumption
- Poor Communication
- Physical Counts
- Daily Purchases
- Ordering Supplies

1.3 Proposed Solution:

This research work covers stock control, management and tends to correct anomalies in Supermarket business. It analyses opening of new stocks, stock updates and ability to view existing ones. It provides quick way of operation by capturing the manual process and automating them. This project is helpful to computerize the item transaction, sales activity record keeping which is a very huge task and maintaining the stock.

1.4 Objectives:

1. To study the functions of Supermarket management system.
2. To explore the challenges being faced by the manual system.
3. To make a software fast in processing, with good user interface.
4. To ensure accurate statistics of product item.
5. For Easy record of goods in store and proper identification.

1.5 Outcomes of the project:

There are a number of advantages to billing that include the faster presentation of invoices and reductions in costs in handling paper document.

CHAPTER -2
REQUIREMENT SPECIFICATION

Chapter 2

REQUIREMENT SPECIFICATION

2.1 Initial Investigation:

While many businesses would not be able to cope without some form of subscription software helping them to run their subscription billing, there are those that do undertake this task. Often they use forms, sheets, databases and spreadsheets to keep on top of their subscriber lists.

While some may feel that they can run a highly successful subscription business this way, there are numerous issues they face. Here is a list of the potential challenges associated with a manual subscription billing system.

1. Missed Billing Period
2. Incorrect Invoices
3. Scalability
4. Credit Control

2.2 Information Gathering:

The information for the project has been collected from Google. Some information has been collected by watching certain useful videos on the related topic. Most of the things has been included with the help of our mentors. Some parts of the project is done through analyzing certain other existing projects.

2.3 Feasibility Study:

"Feasibility Study" is a test of the system according to its workability, impact of the organization, ability to meet user needs and effective use of the resources. We can test our system by different type of the feasibilities. There are 5 types of the feasibilities which are discussed here. These are as follows:

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 support good user interface and easy database handling.

Technical needs may include: Front-End Selection: Front-End means a language that is used for user interface designing and coding. Front-End should have following qualities:

- It must have a graphical user interface that assist employees that are not from some IT background.
- Scalability and Extensibility
- Robustness
- According to the organization requirements and culture.
- Must provide excellent reporting features with good printing support.
- Platform independent.
- Easy to deploy and maintain.
- Event driven programming.
- Front-End must support some popular Back-End like MS Access, SQL Server and Oracle. According to the above stated features we selected Visual C#.Net as Front–End for developing our project. Visual C#.Net is used in Microsoft Visual Studio.Net 2015.

Back-End Selection: Back-End means a language that is used for database management.

Back-End should have following qualities:

- Multiple user support.
- Provide inherent feature for security.
- Efficient data retrieval and maintenance.
- Stored procedures.
- Popularity.

- Operating System compatible.
- Easy to install.
- Various drivers must be available
- Efficient data handling.
- Easy to implement with Front-End.

According to the above stated features we selected Microsoft SQL as Back–End for developing our project. We will use Microsoft SQL 2014 specifically because it has more feature features then other later versions and it is easy to make and maintain database. It is also easy to implement Microsoft SQL 2014 with Visual C#.Net in Microsoft Visual Studio.Net 2015.

2. **Economic Feasibility**: In this we consider following costs:

- The cost to conduct a full system investigation.
- The cost of hardware and software for class of application being considered.
- The benefit in the form of the reduced cost.
- Our system has a lot of features at a minimum cost so it is feasible to implement and it will be very much beneficial to the sellers in the reduced cost. It's software and hardware cost is also low then the existing system.

3. **Operational Feasibility**: In this feasibility we consider following points:

- What changes will be brought with the system.
- What new skills will be required? Do the existing staff members have these skills? If not, can they be trained in due course of time?

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.

4. **Schedule Feasibility**: Time evaluation is most important consideration in development of the project. So the project is concerned should be completed with fixed in scheduled time as far as company is concerned. New system is not so much big so it is easy to make in few days.

5. **Behavioral Feasibility**: People are inherently resisted to change and a computer

means “change is the only certainty”. An estimate should be made of how strong a reaction the user staff in going to have towards development of new system. Thus special efforts can be made to educate and train the staff.

2.4 Existing System:

Many Supermarkets use this type of billing system for a decade. It is also improved many times according to requirements of sellers and customers. It does the same work that is calculating the bill, gives it to the customer and maintain proper database. They are accurate in calculation and printing, they also generate records. A new concept is also added in the billing system is that they also maintain relationships with the customers who purchase more products from the store regularly. System also concerns their requirements and gives them more commission. It also shows the overall profit and profit on a particular product and give repots which items are required and which have cross their expiry date.

2.5 Proposed System:

The project is on Supermarket Billing. Supermarket is the place where customers come to purchase their daily using products and pay for that. So there is a need to calculate how many products are sold and to generate the bill for the customer. To make software fast in processing, with good user interface so that user can change it and it should be used for a long time without error and maintenance.

2.6 Advantages Of Proposed System:

The new system should concern the requirements of the customer and the sellers. It has the following qualities:

- Reduction in processing cost.
- Error reduction.
- Automatic posting.
- Improve reporting.
- Automatic production of the documents and Reports.
- Faster response time.
- Ability to meet user requirements.

- Flexibility.
- Reduced dependency.
- Improves resource uses.
- Reduction in use of the paper.
- Reduction in Man Power.

Proposed system has these qualities including the qualities of the existing system

CHAPTER 3
SYSTEM REQUIRMENT ANALYSIS

Chapter 3

SYSTEM REQUIREMENT SPECIFICATION

3.1 Functional Requirements:

SYSTEM ENVIRONMENT: This software is designed and developed to perform effectively. It is simple and provides good graphical user interface (GUI) to both new as well as experienced user of the system.

PRODUCT FUNCTIONS: The admin can insert, delete and update in the software database to store information

USER CHARACTERISTICS: The users are required to have basic computer knowledge and English. So that they can make use of it without any problems.

GENERAL CONSTRAINTS: The system should have window XP operating system or higher.

ASSUMPTIONS AND DEPENDENCIES: This project aims to fulfill the needs of recording, viewing, updating, deleting and modifying existing details of planets. The product is user friendly and easy to use.

3.2 Non-Functional Requirements:

USABILITY: A non-functional requirement is a requirement that specifies criteria that can be used to check operation of system, rather than specific behavior.
PERFOMANCE: It is the time taken by system to response. It takes two seconds to load the page. The information is verified within five seconds.

SPEED: The speed of the system is measured by processed transactions, screen refresh time, event response time.

SIZE: We measure the size in Mbytes and number from chips

RELIABILITY: The Reliability function is theoretically defined as the probability of success. The system should be so reliable that it would work for a long time without any problem and we can easily trust the system. There are two terms related to the software reliability.

1. Fault means a defect in the software like a bug in the code which can cause failure in the software.
2. Failure means that when the behavior of the software is not same as the specified one.

ROBUSTNESS: It is the ability of a computer system to cope with errors during execution and cope with erroneous input. It is measured by time to start after failure, percentage of events causing failure, probability of data corruption on failure.

PORTABILITY: The system shall run in any window/Linux environment and a server to access database.

SUPPORTABILITY: The system needs to be cost-effective to maintain. Maintainability requirements may cover diverse levels of documentation, such as system documentation, as well as test documentation.

SECURITY: The system is secured for any research purpose, only the authorized user, admin can alter the item information.

AVAILABILITY: The system is available only under normal working conditions.

CHAPTER 4

SYSTEM DESIGN

CHAPTER 4

SYSTEM DESIGN

4.1 System Architecture

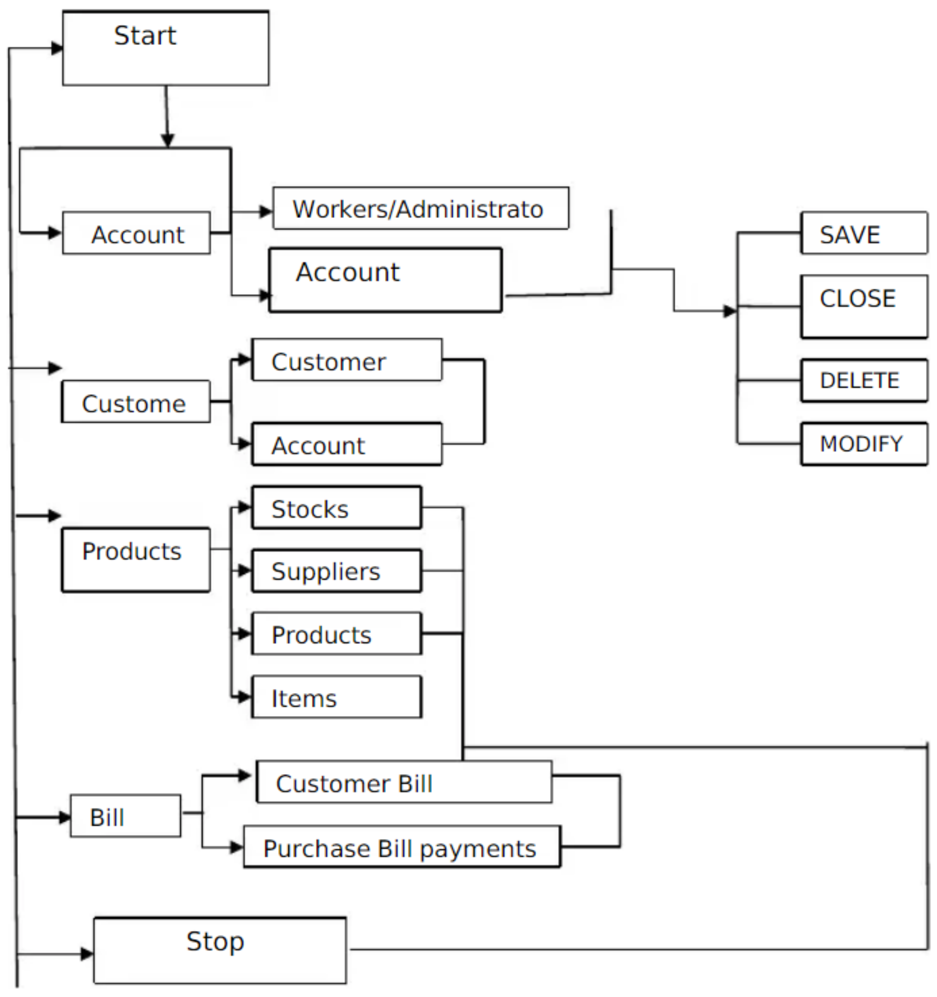


FIG 4.1: SYSTEM ARCHITECTURE

4.2 ER Diagram

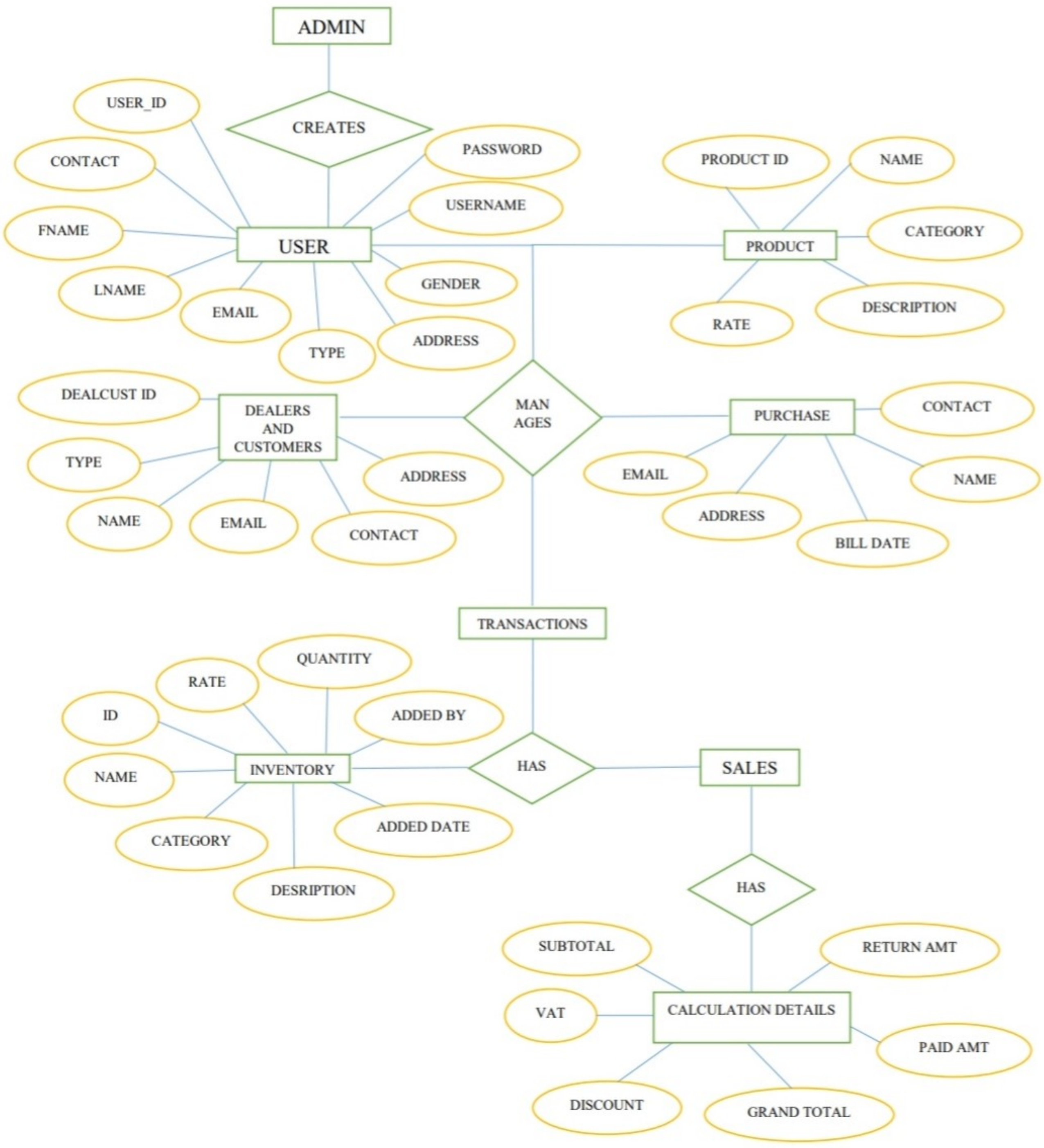


FIG 4.2 ER DIAGRAM

4.3 FLOW CHART

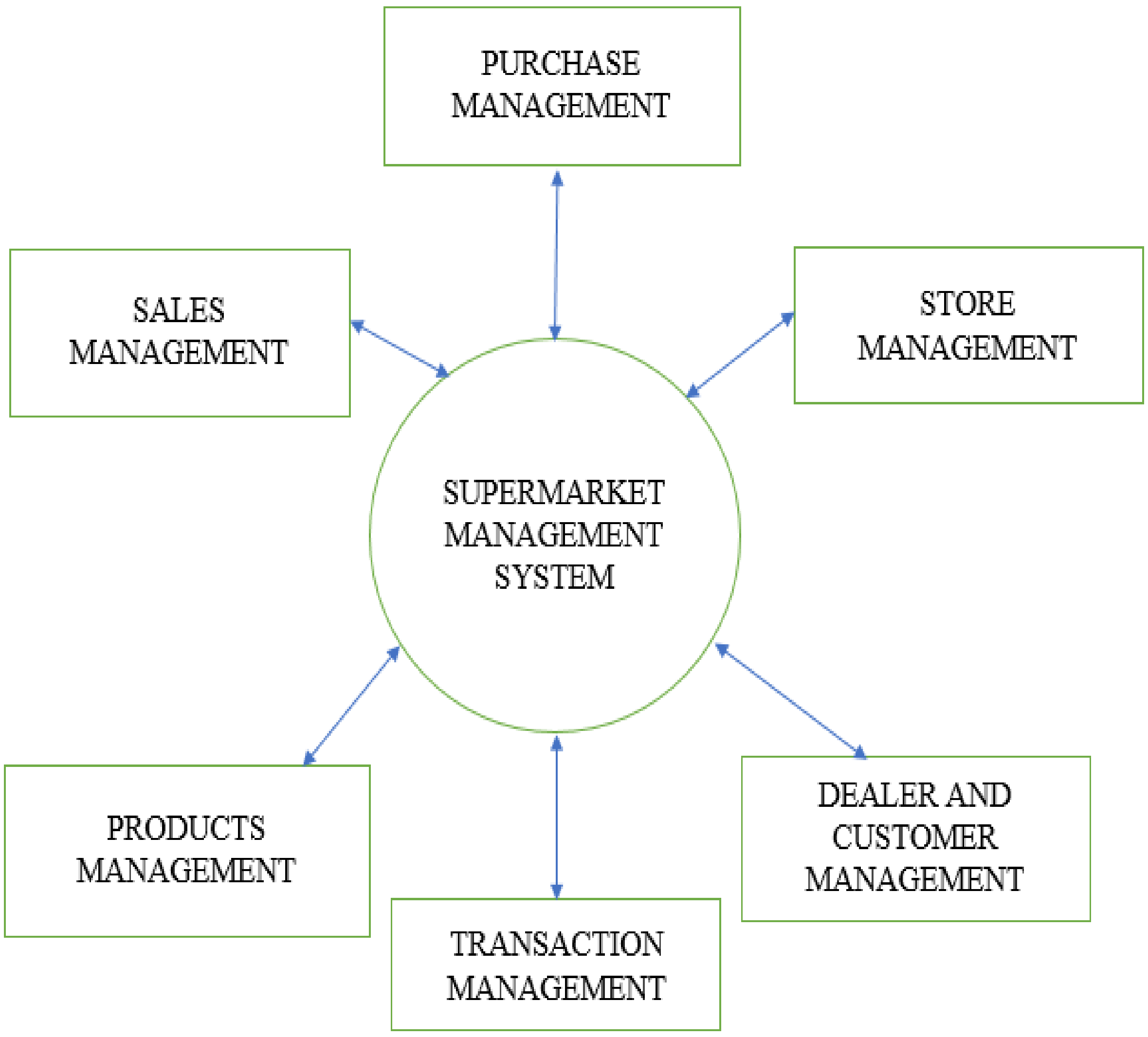
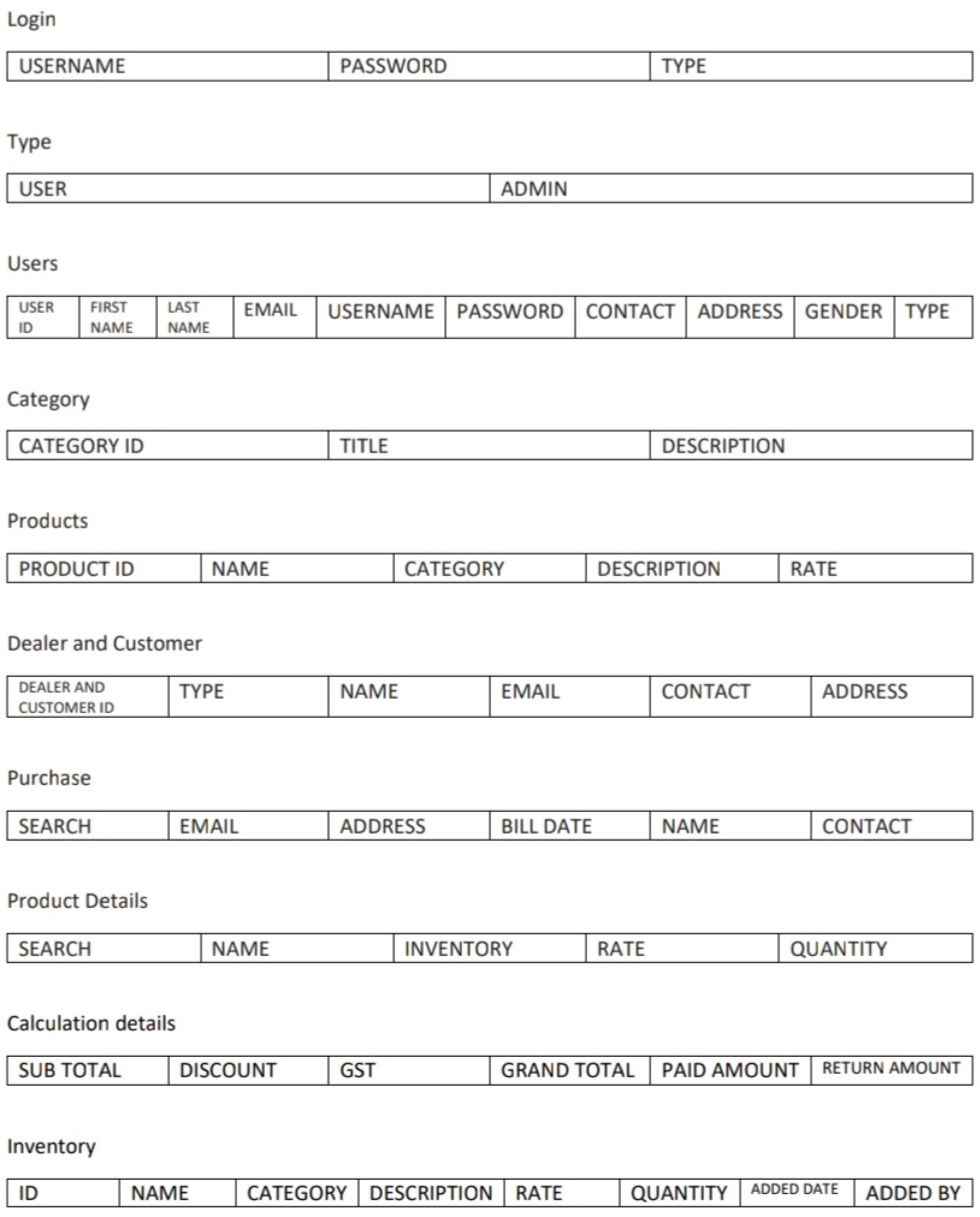


FIG 4.3 FLOW CHART

4.4 SCHEMA DIAGRAM



Login

USERNAME	PASSWORD	TYPE

Type

USER	ADMIN

Users

USER ID	FIRST NAME	LAST NAME	EMAIL	USERNAME	PASSWORD	CONTACT	ADDRESS	GENDER	TYPE

Category

CATEGORY ID	TITLE	DESCRIPTION

Products

PRODUCT ID	NAME	CATEGORY	DESCRIPTION	RATE

Dealer and Customer

DEALER AND CUSTOMER ID	TYPE	NAME	EMAIL	CONTACT	ADDRESS

Purchase

SEARCH	EMAIL	ADDRESS	BILL DATE	NAME	CONTACT

Product Details

SEARCH	NAME	INVENTORY	RATE	QUANTITY

Calculation details

SUB TOTAL	DISCOUNT	GST	GRAND TOTAL	PAID AMOUNT	RETURN AMOUNT

Inventory

ID	NAME	CATEGORY	DESCRIPTION	RATE	QUANTITY	ADDED DATE	ADDED BY

FIG 4.4 SCHEMA DIAGRAM

CHAPTER 5

IMPLEMENTATION

Chapter 5
IMPLEMENTATION

5.1 Tools And Language

Tools used:

1. Microsoft Visual Studio 2015
2. Microsoft SQL Server 2014

Languages used:

1. C# (C Sharp Programming Language)
2. Microsoft Standardized Query Language(MSQL)

5.2 Modules

5.2.1 Module 1-Login

In the login page, both the user and admin can login to their dashboards by giving their correct username, password and type. The admin has permission to access the entire database. He can register the users and admins. If any of the user or admin enters the wrong username or password he or she will not be able to login

Code of Login Page:

```
namespace AnyStore.UI
{
    public partial class frmLogin : Form
    {
        public frmLogin()
        {
            InitializeComponent();
        }
```

```
loginBLL l = new loginBLL();
loginDAL dal = new loginDAL();
public static string loggedIn;
private void pboxClose_Click(object sender, EventArgs e)
{
   //Code to close this form
   this.Close();
}
private void btnLogin_Click(object sender, EventArgs e)
{
   l.username = txtUsername.Text.Trim();
   l.password = txtPassword.Text.Trim();
   l.user_type = cmbUserType.Text.Trim();
     //Checking the login credentials
   bool sucess = dal.loginCheck(l);
   if(sucess==true)
   {
      //Login Successfull
      MessageBox.Show("Login Successful.");
      loggedIn = l.username;
      //Need to open Respective Forms based on User Type
      switch(l.user_type)
      {
         case "Admin":
            {
               //Display Admin Dashboard
               frmAdminDashboard admin = new frmAdminDashboard();
               admin.Show();
               this.Hide();
            }
            break;
            case "User":
                {
                    //Display User Dashboard
                    frmUserDashboard user = new frmUserDashboard();
                    user.Show();
                    this.Hide();
                }
                break;
                default:
                {
                    //Display an error message
                    MessageBox.Show("Invalid User Type.");
                }
                break;
            }
        }
        else
        {
            //login Failed
            MessageBox.Show("Login Failed. Try Again");
        }
    }
  }
}
```

5.2.2 Module 2-Admin Dashboard

In the admin dashboard, the admin can add the details of the users, category, products, dealer and customer and view the details of the inventory and transactions.

```
namespace AnyStore
{
  public partial class frmAdminDashboard : Form
  {
    public frmAdminDashboard()
```

```
{
    InitializeComponent();
}

private void usersToolStripMenuItem_Click(object sender, EventArgs e)
{
    frmUsers user = new frmUsers();
    user.Show();
}
private void frmAdminDashboard_FormClosed(object sender, FormClosedEventArgs e)
{
    frmLogin login = new frmLogin();
    login.Show();
    this.Hide();
}
    private void frmAdminDashboard_Load(object sender, EventArgs e)
{
    lblLoggedInUser.Text = frmLogin.loggedIn;
}
    private void categoryToolStripMenuItem_Click(object sender, EventArgs e)
{
    frmCategories category = new frmCategories();
    category.Show();
}
    private void productsToolStripMenuItem_Click(object sender, EventArgs e)
{
    frmProducts product = new frmProducts();

    product.Show();
}
private void deealerAndCustomerToolStripMenuItem_Click(object sender, EventArgse)
{
    frmDeaCust DeaCust = new frmDeaCust();
            DeaCust.Show();
        }
        private void transactionToolStripMenuItem_Click(object sender, EventArgse)
        {
            frmTransactions transaction = new frmTransactions();
            transaction.Show();
        }
        private void inventoryToolStripMenuItem_Click(object sender, EventArgs e)
        {
            frmInventory inventory = new frmInventory();
            inventory.Show();
        }
    }
}
```

5.2.3 Users form

In users form the details of the users is inserted by admin and he can add, update or delete any user.

```
namespace AnyStore
{
    public partial class frmAdminDashboard : Form
    {
        public frmAdminDashboard()
        {

            InitializeComponent();
        }
        private void usersToolStripMenuItem_Click(object sender, EventArgs e)
        {
            frmUsers user = new frmUsers();
            user.Show();
        }
```

```
private void frmAdminDashboard_FormClosed(object sender, FormClosedEventArgs e)
        {
           frmLogin login = new frmLogin();
           login.Show();
           this.Hide();
        }
        private void frmAdminDashboard_Load(object sender, EventArgs e)
        {
           lblLoggedInUser.Text = frmLogin.loggedIn;
        }
        private void categoryToolStripMenuItem_Click(object sender, EventArgs e)
        {
           frmCategories category = new frmCategories();
           category.Show();
        }
        private void productsToolStripMenuItem_Click(object sender, EventArgs e)
        {
           frmProducts product = new frmProducts();
           product.Show();
        }
       private void deealerAndCustomerToolStripMenuItem_Click(object sender, EventArgse)
        {
           frmDeaCust DeaCust = new frmDeaCust();
           DeaCust.Show();
        }
        private void transactionToolStripMenuItem_Click(object sender, EventArgse)
        {
           frmTransactions transaction = new frmTransactions();
           transaction.Show();
        }
         private void inventoryToolStripMenuItem_Click(object sender, EventArgs e)
        {
            frmInventory inventory = new frmInventory();
            inventory.Show();
        }
    }
}
```

5.2.4 Category and Products form

In category form the category of the products is given by the admin and the corresponding products are added in products form

```
namespace AnyStore.UI
{
    public partial class frmCategories : Form
    {
        public frmCategories()
        {
            InitializeComponent();
        }

        private void pictureBoxClose_Click(object sender, EventArgs e)

        {
            this.Hide();
        }
        categoriesBLL c = new categoriesBLL();
        categoriesDAL dal = new categoriesDAL();
        userDAL udal = new userDAL();
        private void btnADD_Click(object sender, EventArgs e)
        {
            //Get the values from Categroy Form
            c.title = txtTitle.Text;
            c.description = txtDescription.Text;
            c.added_date = DateTime.Now;
```

```
      //Getting ID in Added by field
    string loggedUser = frmLogin.loggedIn;
    userBLL usr = udal.GetIDFromUsername(loggedUser);
    //Passign the id of Logged in User in added by field
    c.added_by = usr.id;
    //Creating Boolean Method To insert data into database
    bool success = dal.Insert(c);
    //If the category is inserted successfully then the value of the success will    be true
    else it will be false
    if(success==true)
    {
       //NewCAtegory Inserted Successfully
       MessageBox.Show("New Category Inserted Successfully.");
       Clear();
       //Refresh Data Grid View
       DataTable dt = dal.Select();

       dgvCategories.DataSource = dt;
    }
    else
    {
       //FAiled to Insert New Category
       MessageBox.Show("Failed to Insert New CAtegory.");
    }
}
public void Clear()
{
    txtCategoryID.Text = "";
    txtTitle.Text = "";
    txtDescription.Text = "";
    txtSearch.Text = "";
}
```

```
private void frmCategories_Load(object sender, EventArgs e)
{
  //Here write the code to display all the categries in DAta Grid View
  DataTable dt = dal.Select();
  dgvCategories.DataSource = dt;
}
  private void dgvCategories_RowHeaderMouseClick(object sender,
  DataGridViewCellMouseEventArgs e)
{
  //Finding the Row Index of the Row Clicked on Data Grid View
  int RowIndex = e.RowIndex;
  txtCategoryID.Text = dgvCategories.Rows[RowIndex].Cells[0].Value.ToString();
  txtTitle.Text = dgvCategories.Rows[RowIndex].Cells[1].Value.ToString();
  txtDescription.Text = dgvCategories.Rows[RowIndex].Cells[2].Value.ToString();
}
  private void btnUpdate_Click(object sender, EventArgs e)
{
  //Get the Values from the CAtegory form
  c.id = int.Parse(txtCategoryID.Text);
  c.title = txtTitle.Text;
  c.description = txtDescription.Text;
  c.added_date = DateTime.Now;
  //Getting ID in Added by field
  string loggedUser = frmLogin.loggedIn;
  userBLL usr = udal.GetIDFromUsername(loggedUser);
  //Passign the id of Logged in User in added by field
  c.added_by = usr.id;
  //Creating Boolean variable to update categories and check
  bool success = dal.Update(c);
  //If the cateory is updated successfully then the value of success will be      true else
  it will be false
  if(success==true)
  {
  //CAtegory updated Successfully
    MessageBox.Show("Category Updated Successfully");
    Clear();
    //Refresh Data Gid View
    DataTable dt = dal.Select();
    dgvCategories.DataSource = dt;
  }
  else
  {
    //FAiled to Update Category
    MessageBox.Show("Failed to Update Category");
  }
}
  private void btnDelete_Click(object sender, EventArgs e)
{
  //Get te ID of the Category Which we want to Delete
  c.id = int.Parse(txtCategoryID.Text);
  //Creating Boolean Variable to Delete The CAtegory
  bool success = dal.Delete(c);
  //If the CAtegory id Deleted Successfully then the vaue of success will be true else it
  will be false
  if(success==true)
  {
    //Category Deleted Successfully
    MessageBox.Show("Category Deleted Successfully");
    Clear();
    //REfreshing DAta Grid View
    DataTable dt = dal.Select();
    dgvCategories.DataSource = dt;
  }
  else
  {
          //FAiled to Delete CAtegory
              MessageBox.Show("Failed to Delete CAtegory");
          }
      }
      private void txtSearch_TextChanged(object sender, EventArgs e)
      {
          //Get the Keywords
          string keywords = txtSearch.Text;

          //Filter the categories based on keywords
          if(keywords!=null)
          {
              //Use Searh Method To Display Categoreis
              DataTable dt = dal.Search(keywords);
              dgvCategories.DataSource = dt;
          }
          else
          {
              //Use Select Method to Display All Categories
              DataTable dt = dal.Select();
              dgvCategories.DataSource = dt;
          }
      }
  }
}
```

5.2.5 Dealer and Customer form

In dealer and customer form, both the user and the admin and user can add, update or delete the details of the dealer or the customers.

```
namespace AnyStore.UI
{
    public partial class frmDeaCust : Form
    {
     public frmDeaCust()
    {
      InitializeComponent();
    }

    private void pictureBoxClose_Click(object sender, EventArgs e)
    {
      //Write the code to close this form
      this.Hide();
    }
    DeaCustBLL dc = new DeaCustBLL();
    DeaCustDAL dcDal = new DeaCustDAL();
    userDAL uDal = new userDAL();
    private void btnAdd_Click(object sender, EventArgs e)
    {
      //Get the Values from Form
      dc.type = cmbDeaCust.Text;
      dc.name = txtName.Text;
      dc.email = txtEmail.Text;
      dc.contact = txtContact.Text;
      dc.address = txtAddress.Text;
      dc.added_date = DateTime.Now;
      //Getting the ID to Logged in user and passign its value in dealer or customer module
      string loggedUsr = frmLogin.loggedIn;
      userBLL usr = uDal.GetIDFromUsername(loggedUsr);
      dc.added_by = usr.id;
      //Creating boolean variable to check whether the dealer or cutomer is added or not
      bool success = dcDal.Insert(dc);
      if(success==true)
      {
        //Dealer or Cutomer inserted successfully
        MessageBox.Show("Dealer or Customer Added Successfully");
        Clear();
```

```
            //Refresh Data Grid View
            DataTable dt = dcDal.Select();
            dgvDeaCust.DataSource = dt;
        }
        else
        {
            //failed to insert dealer or customer
        }
    }
    public void Clear()
    {
        txtDeaCustID.Text = "";
        txtName.Text = "";
        txtEmail.Text = "";
        txtContact.Text = "";
        txtAddress.Text = "";
        txtSearch.Text = "";
    }
    private void frmDeaCust_Load(object sender, EventArgs e)
    {
        //Refresh Data Grid View
        DataTable dt = dcDal.Select();
        dgvDeaCust.DataSource = dt;
    }
        private void dgvDeaCust_RowHeaderMouseClick(object sender,
        DataGridViewCellMouseEventArgs e)
    {
        //int variable to get the identityof row clicked
        int rowIndex = e.RowIndex;

        txtDeaCustID.Text = dgvDeaCust.Rows[rowIndex].Cells[0].Value.ToString();
        cmbDeaCust.Text = dgvDeaCust.Rows[rowIndex].Cells[1].Value.ToString();
        txtName.Text = dgvDeaCust.Rows[rowIndex].Cells[2].Value.ToString();
```

```
                txtEmail.Text = dgvDeaCust.Rows[rowIndex].Cells[3].Value.ToString();
                txtContact.Text = dgvDeaCust.Rows[rowIndex].Cells[4].Value.ToString();
                txtAddress.Text = dgvDeaCust.Rows[rowIndex].Cells[5].Value.ToString();
            }
            private void btnUpdate_Click(object sender, EventArgs e)
            {
                //Get the values from Form
                dc.id = int.Parse(txtDeaCustID.Text);
                dc.type = cmbDeaCust.Text;
                dc.name = txtName.Text;
                dc.email = txtEmail.Text;
                dc.contact = txtContact.Text;
                dc.address = txtAddress.Text;
                dc.added_date = DateTime.Now;
                //Getting the ID to Logged in user and passign its value in dealer or customer
                module
                string loggedUsr = frmLogin.loggedIn;
                userBLL usr = uDal.GetIDFromUsername(loggedUsr);
                dc.added_by = usr.id;
                //create boolean variable to check whether the dealer or customer is updated  or not
                bool success = dcDal.Update(dc);
                if(success==true)
                {
                    //Dealer and Customer update Successfully
                    MessageBox.Show("Dealer or Customer updated Successfully");
                    Clear();
                    //Refresh the Data Grid View
                    DataTable dt = dcDal.Select();
                    dgvDeaCust.DataSource = dt;
                }
                else
                {
                    //Failed to udate Dealer or Customer
                    MessageBox.Show("Failed to Udpate Dealer or Customer");
                }
            }
            private void btnDelete_Click(object sender, EventArgs e)
            {
                //Get the id of the user to be deleted from form
                dc.id = int.Parse(txtDeaCustID.Text);
                //Create boolean variable to check wheteher the dealer or customer is deleted   or not
                bool success = dcDal.Delete(dc);
                if(success==true)
                {
                    //Dealer or Customer Deleted Successfully
                    MessageBox.Show("Dealer or Customer Deleted Successfully");
                    Clear();
                    //Refresh the Data Grid View
                    DataTable dt = dcDal.Select();
                    dgvDeaCust.DataSource = dt;
                }
                else
                {
                    //Dealer or Customer Failed to Delete
                    MessageBox.Show("Failed to Delete Dealer or Customer");
                }
            }
            private void txtSearch_TextChanged(object sender, EventArgs e)
            {
                //Get the keyowrd from text box
                string keyword = txtSearch.Text;
                if(keyword!=null)
                {
                    //Search the Dealer or Customer
                    DataTable dt = dcDal.Search(keyword);
```

```
                dgvDeaCust.DataSource = dt;
            }
            else
            {
                //Show all the Dealer or Customer
                DataTable dt = dcDal.Select();
                dgvDeaCust.DataSource = dt;
            }
        }
    }
}
```

5.2.6 Transaction Form

In transaction form the admin can view the transactions that are done by the users.

```
namespace AnyStore.UI
{
    public partial class frmTransactions : Form
    {
        public frmTransactions()
        {
            InitializeComponent();
        }

        transactionDAL tdal = new transactionDAL();
        private void pictureBoxClose_Click(object sender, EventArgs e)
        {
            this.Hide();
        }
        private void frmTransactions_Load(object sender, EventArgs e)
        {
            //Dispay all the transactions
            DataTable dt = tdal.DisplayAllTransactions();
```

```
              dgvTransactions.DataSource = dt;
          }
            private void cmbTransactionType_SelectedIndexChanged(object sender, EventArgse)
          {
            //Get the Value from Combobox
            string type = cmbTransactionType.Text;
            DataTable dt = tdal.DisplayTransactionByType(type);
            dgvTransactions.DataSource = dt;
          }
          private void btnAll_Click(object sender, EventArgs e)
          {
            //Dispay all the transactions
            DataTable dt = tdal.DisplayAllTransactions();
            dgvTransactions.DataSource = dt;
          }
        }
      }
```

5.2.7 User dashboard

In user dashboard, the user can purchase, make sales, view inventory and see the details of the dealer and customer.

```
namespace AnyStore
{
    public partial class frmUserDashboard : Form
    {
        public frmUserDashboard()
        {
            InitializeComponent();
        }
        //Set a public static method to specify whether the form is purchase or sales
        public static string transactionType;
       private void frmUserDashboard_FormClosed(object sender, FormClosedEventArgs e)
        {
```

```
        frmLogin login = new frmLogin();
        login.Show();
        this.Hide();
    }
    private void frmUserDashboard_Load(object sender, EventArgs e)
    {
        lblLoggedInUser.Text = frmLogin.loggedIn;
    }
     private void dealerAndCustomerToolStripMenuItem_Click(object sender, EventArgs e)
    {
        frmDeaCust DeaCust = new frmDeaCust();
        DeaCust.Show();
    }
    private void purchaseToolStripMenuItem_Click(object sender, EventArgs e)
    {
        //set value on transactionType static method
        transactionType = "Purchase";
        frmPurchaseAndSales purchase = new frmPurchaseAndSales();
        purchase.Show();
    }
    private void salesFormsToolStripMenuItem_Click(object sender, EventArgs e)
    {
        //Set the value to transacionType method to sales
        transactionType = "Sales";
        frmPurchaseAndSales sales = new frmPurchaseAndSales();
        sales.Show();
    }
    private void inventoryToolStripMenuItem_Click(object sender, EventArgs e)
    {
        frmInventory inventory = new frmInventory();
        inventory.Show();
    }
  }
```

```
}
```

5.2.8 Inventory Form

In inventory form, both the admin and the user can view the inventory.

```
namespace AnyStore.UI
{
  public partial class frmInventory : Form
  {
    public frmInventory()
    {
      InitializeComponent();
    }
    categoriesDAL cdal = new categoriesDAL();
    productsDAL pdal = new productsDAL();
    private void pictureBoxClose_Click(object sender, EventArgs e)
    {
      //Addd Functionality to Close this form
      this.Hide();
    }
    private void frmInventory_Load(object sender, EventArgs e)
    {
      //Display the CAtegories in Combobox
      DataTable cDt = cdal.Select();
      cmbCategories.DataSource = cDt;
      //Give the Value member and display member for Combobox
      cmbCategories.DisplayMember = "title";
      cmbCategories.ValueMember = "title";
      //Display all the products in Datagrid view when the form is loaded
      DataTable pdt = pdal.Select();
      dgvProducts.DataSource = pdt;
    }
    private void cmbCategories_SelectedIndexChanged(object sender, EventArgs e)
    {
            //Display all the Products Based on Selected CAtegory
            string category = cmbCategories.Text;
            DataTable dt = pdal.DisplayProductsByCategory(category);
            dgvProducts.DataSource = dt;
        }
        private void btnAll_Click(object sender, EventArgs e)
        {
            //Display all the productswhen this button is clicked
            DataTable dt = pdal.Select();
            dgvProducts.DataSource = dt;
        }
    }
}
```

CHAPTER 6

TESTING

Chapter 6

TESTING

6.1 Test case: Login

Sl No. Test Case	1
Name of Test Case	Login
Feature being tested	Login Page
Description	What should happen when we enter the correct username, password and type and click the sign in button?
Input	Registered username, password and type
Expected Output	Show the login successful window and open admin or user dashboard
Actual Output	Shows the login successful window and opens admin or user dashboard
Remark	Pass

Table 6.1: Test Case-1

6.2 Test case: Admin Dashboard

Sl No. Test Case	2
Name of Test Case	Admin Dashboard
Feature being tested	Opening of other pages
Description	What should happen when clicked on a particular button?
Input	Click on button (Insert new data)
Expected Output	Open the corresponding page (Insert new data)
Actual Output	Open corresponding page (Insert new data)
Remark	Pass

Table 6.2: Test Case-2

6.3 Test Case: Add Users or Admin

Sl No. Test Case	3
Name of Test Case	Add User details or Admin details
Feature being tested	Inserting the user or admin details
Description	What should happen when we add the details of the user or the admin and click on add, update or delete button?
Input	Add the details and click on the ADD, UPDATE OR DELETE.
Expected Output	All the details getting registered, updated or deleted
Actual Output	Details getting registered, updated or deleted
Remark	Pass

Table 6.3: Table Case-3

6.4 Test Case: Inserting the categories of the product

Sl No. Test Case	4
Name of Test Case	Inserting the details of the categories
Feature being tested	Inserting the categories details
Description	What should happen when we add the details of the categories and click on ADD, UPDATE or DELETE?
Input	Add the details and click on the ADD,UPDATE OR DELETE
Expected Output	All details getting registered
Actual Output	Details getting registered
Remark	Pass

Table 6.4: Table Case-4

6.5 Test Case: Inserting the details of the product

Sl No. Test Case	5
Name of Test Case	Inserting the details of the product
Feature being tested	Inserting the product details
Description	What should happen when we add the details of the product and click on ADD, UPDATE or DELETE?
Input	Add the details and click on the ADD,UPDATE OR DELETE
Expected Output	All details getting registered
Actual Output	Details getting registered
Remark	Pass

Table 6.5: Table Case-5

6.6 Test Case: Inserting the details of the dealers or customers

Sl No. Test Case	6
Name of Test Case	Inserting the details of the dealers or customers
Feature being tested	Inserting the dealers or customer details
Description	What should happen when we add the details of the dealer or customer and click on ADD, UPDATE or DELETE?
Input	Add the details and click on the ADD,UPDATE OR DELETE
Expected Output	All details getting registered
Actual Output	Details getting registered
Remark	Pass

Table 6.6: Table Case-6

6.7 Test Case: Inventory

Sl No. Test Case	7
Name of Test Case	Inventory
Feature being tested	Checking the Inventory toolbox
Description	What should happen when we click on the Inventory toolbox?
Input	Click on the Inventory toolbox
Expected Output	All the details of the products should be shown
Actual Output	Details of the products in the inventory can be seen
Remark	Pass

Table 6.7: Table Case-7

6.8 Test Case: Transaction

Sl No. Test Case	8
Name of Test Case	Transaction
Feature being tested	Transaction done by user
Description	What should happen when the user does the transaction
Input	Click on the transaction toolbox
Expected Output	All the details of the transactions done by the user to be shown
Actual Output	Details of the transaction in the transaction form can be seen
Remark	Pass

Table 6.8: Table Case-8

6.9 Test Case: Purchase

Sl No. Test Case	9
Name of Test Case	Purchase
Feature being tested	Inserting the details of the purchase of the product
Description	What should happen when we add the details of dealer or customer and product details and click on ADD and SAVE?
Input	Inserting the details of the dealer or customer and the details of the product and then clicking on ADD and then SAVE
Expected Output	All the product details should get added to added product box and the transactions should be done when SAVE button is clicked
Actual Output	Details getting registered and transactions are done successfully with the transaction successful window
Remark	Pass

Table 6.9: Table Case-9

6.10 Test Case: Inventory

Sl No. Test Case	10
Name of Test Case	Inventory
Feature being tested	Checking the Inventory toolbox
Description	What should happen when we click on the Inventory toolbox?
Input	Click on the Inventory toolbox
Expected Output	All the details of the products should be shown
Actual Output	Details of the products in the inventory can be seen
Remark	Pass

Table 6.10: Table Case-10

6.11 Test Case: Sales

Sl No. Test Case	11
Name of Test Case	Sales
Feature being tested	Inserting the details of the sales of the product
Description	What should happen when we add the details of dealer or customer and product details and click on ADD and SAVE?
Input	Inserting the details of the dealer or customer and the details of the product and then clicking on ADD and then SAVE
Expected Output	All the product details should get added to added product box and the transactions should be done when SAVE button is clicked
Actual Output	Details getting registered and transactions are done successfully with the transaction successful window
Remark	Pass

Table 6.11: Table Case-11

CHAPTER 7

SNAPSHOT

Chapter 7

SNAPSHOTS

7.1 Login Form

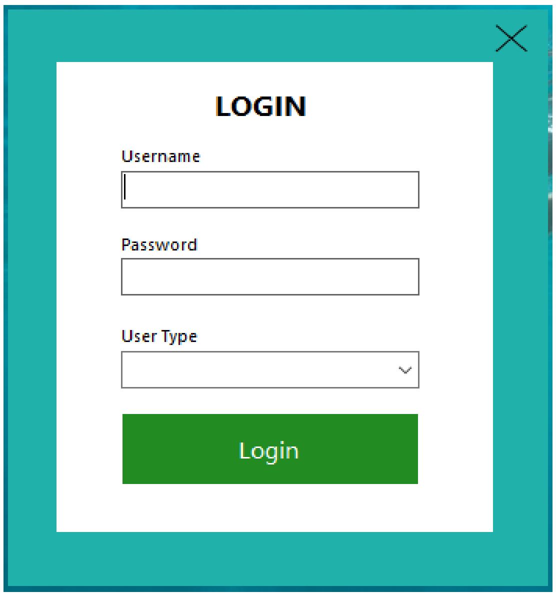


Figure 7.1 Login Form.

7.2 Admin Dashboard

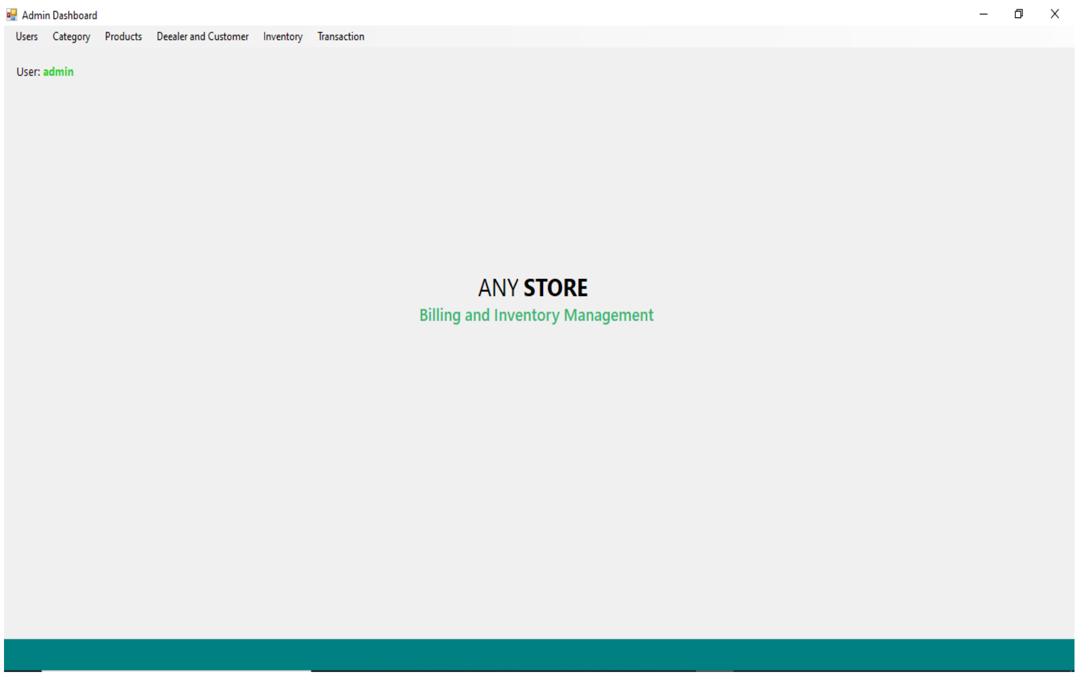
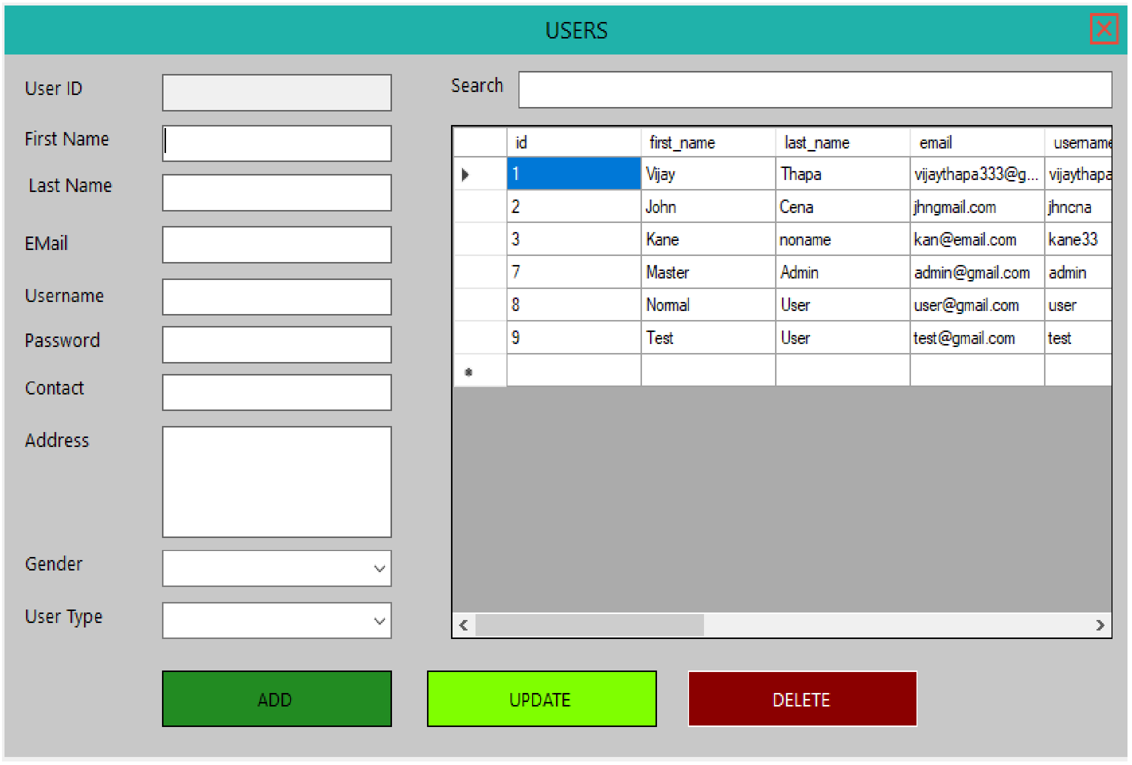


Figure 7.2 Admin Dashboard.

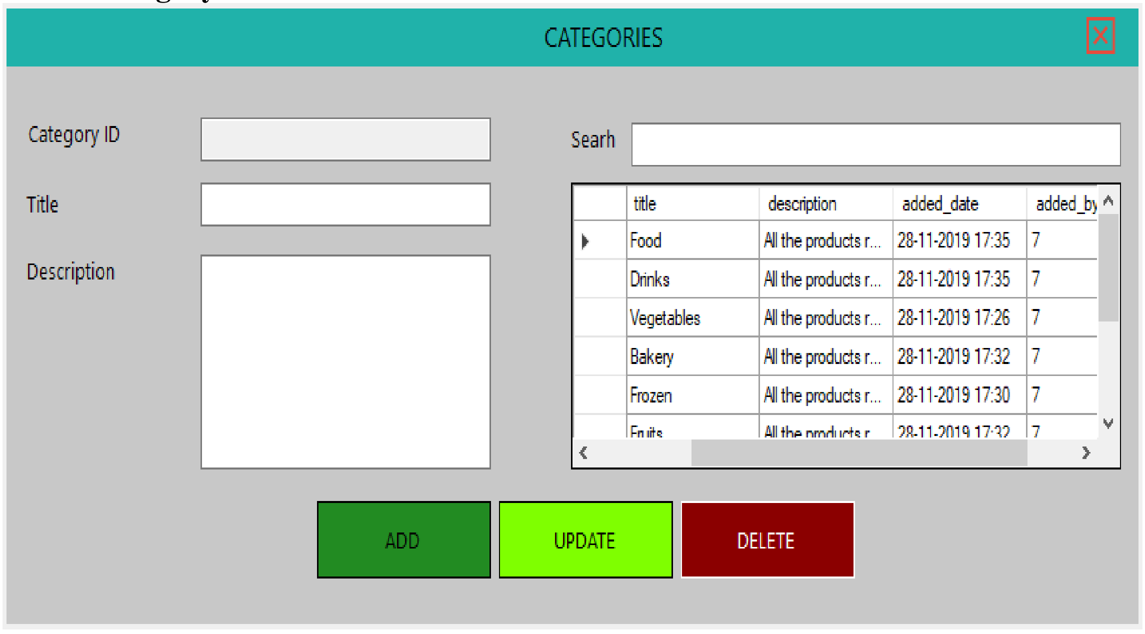
7.2.1 Users Data



id	first_name	last_name	email	username
1	Vijay	Thapa	vijaythapa333@g...	vijaythapa
2	John	Cena	jhngmail.com	jhncna
3	Kane	noname	kan@email.com	kane33
7	Master	Admin	admin@gmail.com	admin
8	Normal	User	user@gmail.com	user
9	Test	User	test@gmail.com	test

Figure 7.2.1 Admin can add the details of the users and view them.

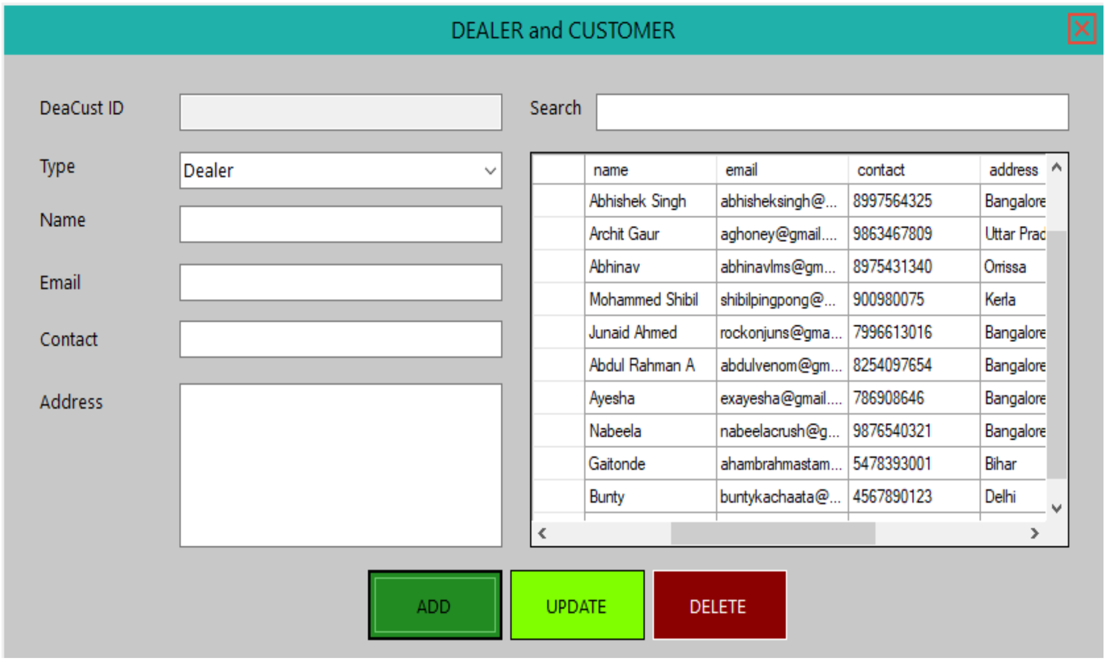
7.2.2 Category



title	description	added_date	added_by
Food	All the products r...	28-11-2019 17:35	7
Drinks	All the products r...	28-11-2019 17:35	7
Vegetables	All the products r...	28-11-2019 17:26	7
Bakery	All the products r...	28-11-2019 17:32	7
Frozen	All the products r...	28-11-2019 17:30	7
Fruits	All the products r...	28-11-2019 17:32	7

Figure 7.2.2 Categories of the products.

7.2.3 Dealer and Customers


DEALER and CUSTOMER

DeaCust ID

Type: Dealer

Name

Email

Contact

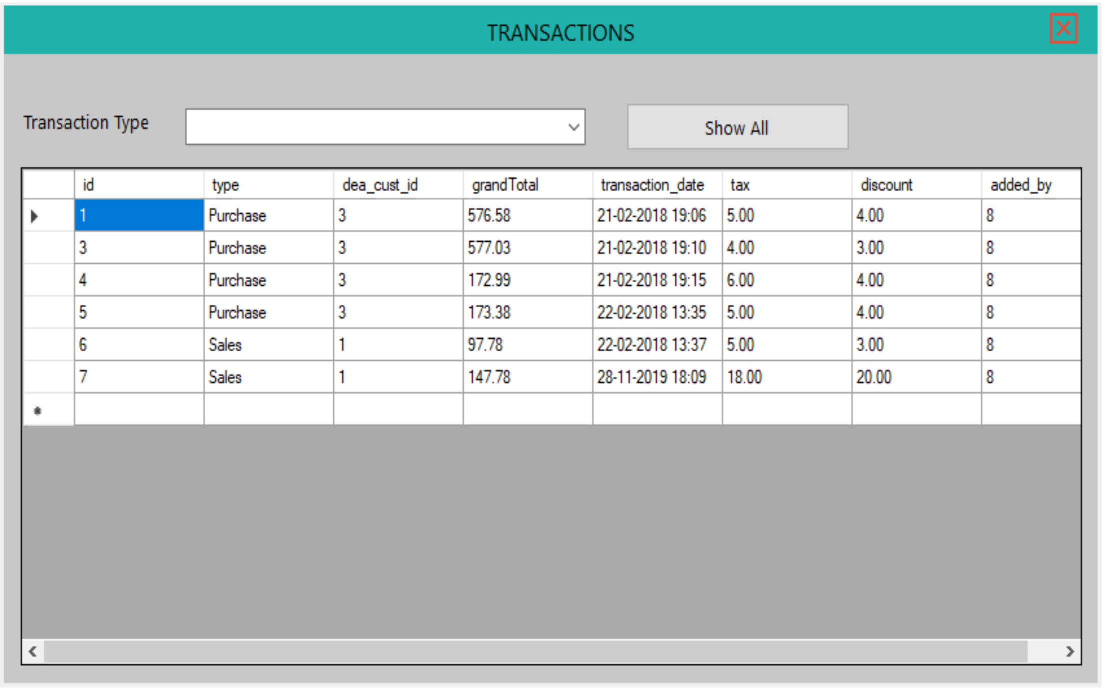
Address

Search

	name	email	contact	address
	Abhishek Singh	abhisheksingh@...	8997564325	Bangalore
	Archit Gaur	aghoney@gmail....	9863467809	Uttar Prac
	Abhinav	abhinavlms@gm...	8975431340	Orrissa
	Mohammed Shibil	shibilpingpong@...	900980075	Kerla
	Junaid Ahmed	rockonjuns@gma...	7996613016	Bangalore
	Abdul Rahman A	abdulvenom@gm...	8254097654	Bangalore
	Ayesha	exayesha@gmail....	786908646	Bangalore
	Nabeela	nabeelacrush@g...	9876540321	Bangalore
	Gaitonde	ahambrahmastam...	5478393001	Bihar
	Bunty	buntykachaata@...	4567890123	Delhi

Figure 7.2.3 Admin can add, update, delete and view the details of the dealers and customers.

7.2.4 Transactions


TRANSACTIONS

Transaction Type

	id	type	dea_cust_id	grandTotal	transaction_date	tax	discount	added_by
▸	1	Purchase	3	576.58	21-02-2018 19:06	5.00	4.00	8
	3	Purchase	3	577.03	21-02-2018 19:10	4.00	3.00	8
	4	Purchase	3	172.99	21-02-2018 19:15	6.00	4.00	8
	5	Purchase	3	173.38	22-02-2018 13:35	5.00	4.00	8
	6	Sales	1	97.78	22-02-2018 13:37	5.00	3.00	8
	7	Sales	1	147.78	28-11-2019 18:09	18.00	20.00	8
*								

Figure 7.2.4 Admin can view the transactions that are done by Users.

7.3 User Dashboard

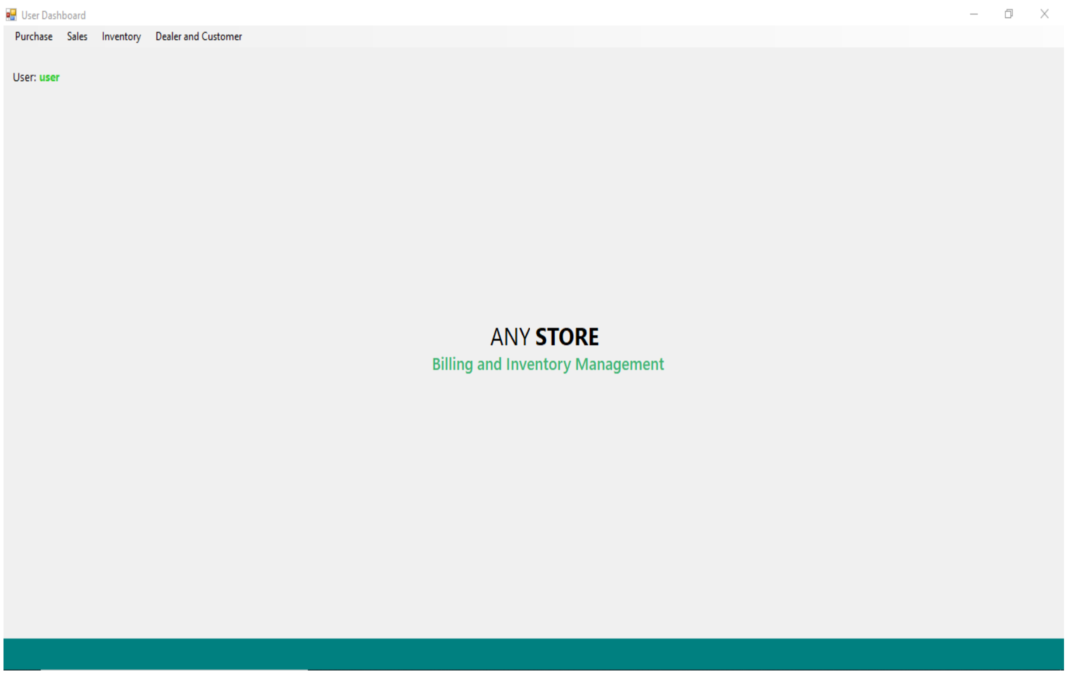
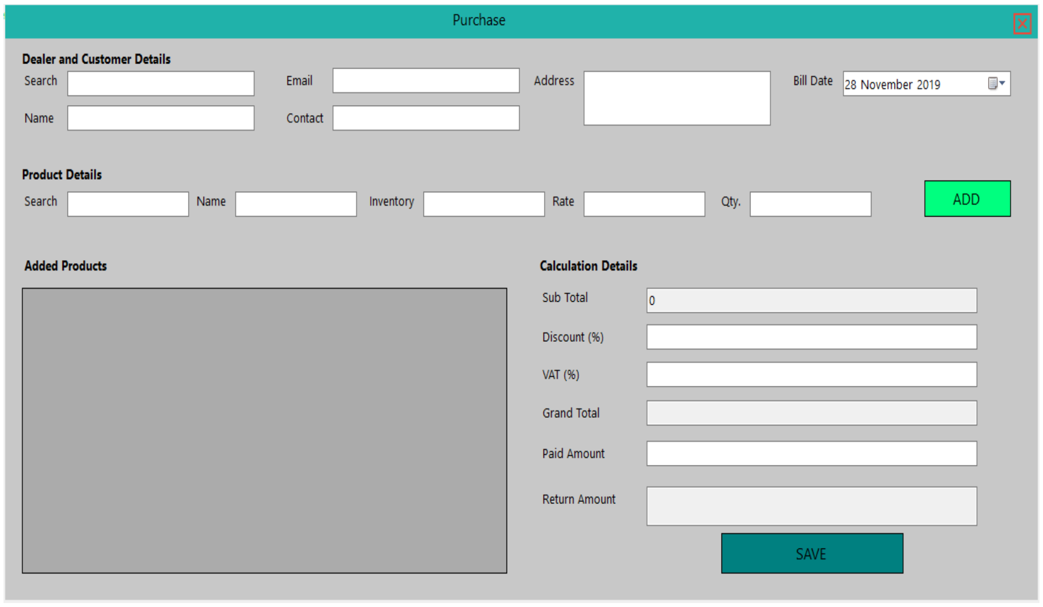


Figure 7.3 User dashboard.

7.3.1 Purchase

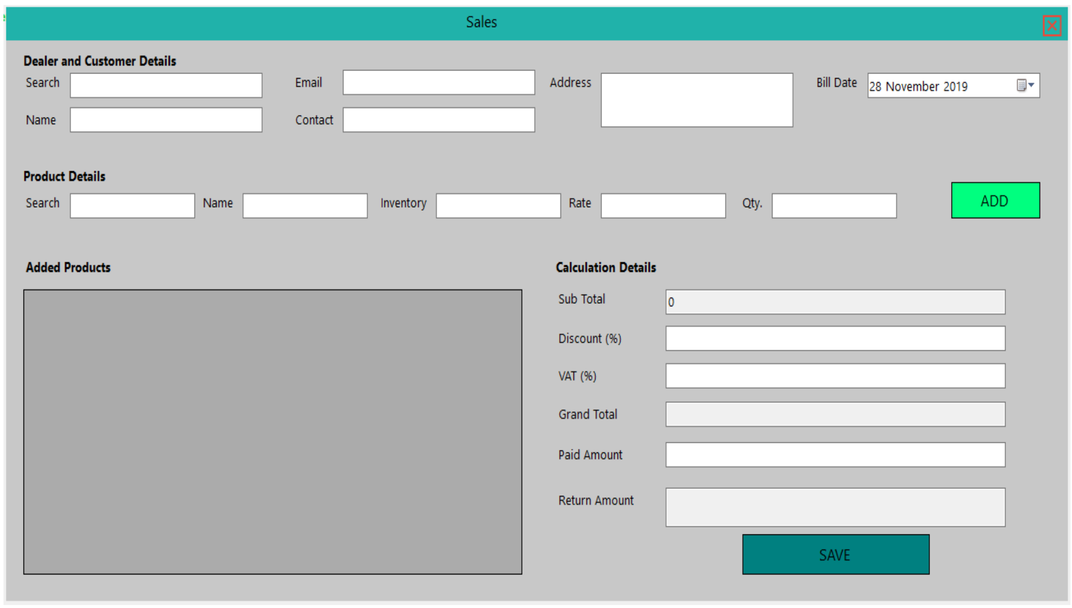


Bill Date: 28 November 2019

Sub Total: 0

Figure 7.3.1 Purchase Form.

7.3.2 Sales


Sales

Dealer and Customer Details

Search

Email

Address

Bill Date 28 November 2019

Name

Contact

Product Details

Search

Name

Inventory

Rate

Qty.

ADD

Added Products

Calculation Details

Sub Total 0

Discount (%)

VAT (%)

Grand Total

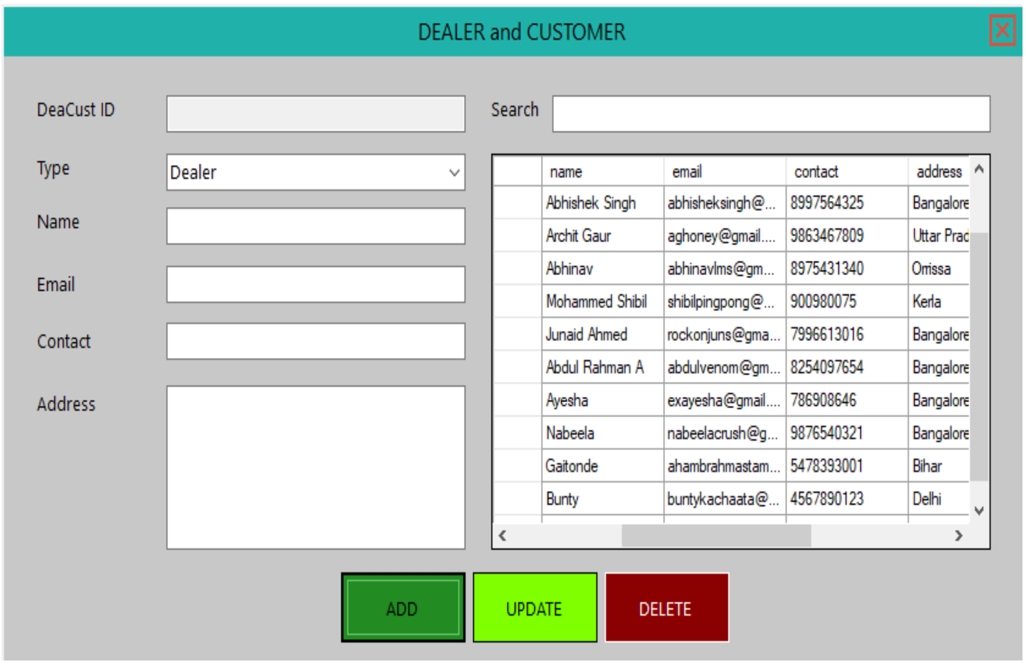
Paid Amount

Return Amount

SAVE

Figure 7.3.2 Sales form.

7.3.3 Dealer and Customer


DEALER and CUSTOMER

DeaCust ID

Search

Type Dealer

Name

Email

Contact

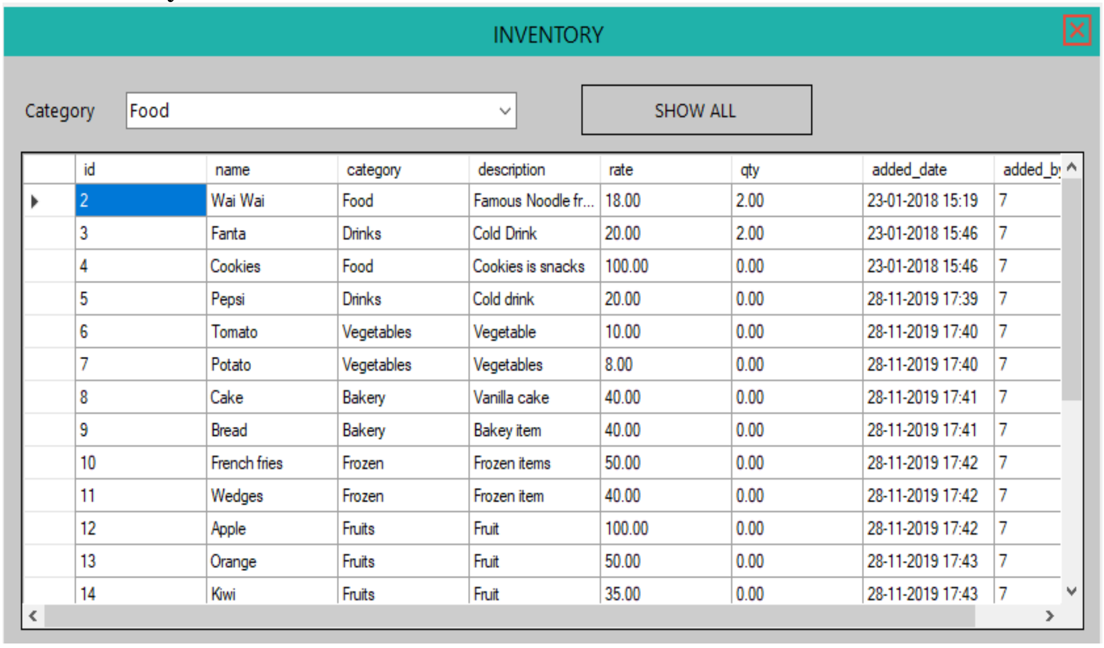
Address

name	email	contact	address
Abhishek Singh	abhisheksingh@...	8997564325	Bangalore
Archit Gaur	aghoney@gmail....	9863467809	Uttar Prad
Abhinav	abhinavlms@gm...	8975431340	Orrissa
Mohammed Shibil	shibilpingpong@...	900980075	Kerla
Junaid Ahmed	rockonjuns@gma...	7996613016	Bangalore
Abdul Rahman A	abdulvenom@gm...	8254097654	Bangalore
Ayesha	exayesha@gmail....	786908646	Bangalore
Nabeela	nabeelacrush@g...	9876540321	Bangalore
Gaitonde	ahambrahmastam...	5478393001	Bihar
Bunty	buntykachaata@...	4567890123	Delhi

ADD UPDATE DELETE

Figure 7.3.3 User can also add, update, delete and view the details of the dealers and customers.

7.4 Inventory



Category: Food

	id	name	category	description	rate	qty	added_date	added_by
▶	2	Wai Wai	Food	Famous Noodle fr...	18.00	2.00	23-01-2018 15:19	7
	3	Fanta	Drinks	Cold Drink	20.00	2.00	23-01-2018 15:46	7
	4	Cookies	Food	Cookies is snacks	100.00	0.00	23-01-2018 15:46	7
	5	Pepsi	Drinks	Cold drink	20.00	0.00	28-11-2019 17:39	7
	6	Tomato	Vegetables	Vegetable	10.00	0.00	28-11-2019 17:40	7
	7	Potato	Vegetables	Vegetables	8.00	0.00	28-11-2019 17:40	7
	8	Cake	Bakery	Vanilla cake	40.00	0.00	28-11-2019 17:41	7
	9	Bread	Bakery	Bakey item	40.00	0.00	28-11-2019 17:41	7
	10	French fries	Frozen	Frozen items	50.00	0.00	28-11-2019 17:42	7
	11	Wedges	Frozen	Frozen item	40.00	0.00	28-11-2019 17:42	7
	12	Apple	Fruits	Fruit	100.00	0.00	28-11-2019 17:42	7
	13	Orange	Fruits	Fruit	50.00	0.00	28-11-2019 17:43	7
	14	Kiwi	Fruits	Fruit	35.00	0.00	28-11-2019 17:43	7

Figure 7.4 Admin and the user can view the inventory.

CHAPTER 8

CONCLUSION & FUTURE WORK

Chapter 8

CONCLUSION AND FUTURE SCOPE

Conclusion

Our project is only a humble venture to satisfy the needs to manage the project. Several user friendly codes have also been adopted. This project shall satisfy all the requirements needed. The objective of the software planning is to provide a framework that enables the manager to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses.

Future Work

1. We can add more advance software for "Supermarket Management System" including more facilities.
2. We can make it accessible worldwide.
3. Implementing the backup mechanism for taking backup of codebase and database on regular basis on different servers.
4. Create master and slave database structure to reduce the overload of database queries

REFERENCES

- http://www.google.com
- http://www.microsoft.com
- http://www.stackoverflow.com
- Library Resources
- Google for problem solving
- J2EE : The complete reference by Herbert Schildt