BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

A Project Report

ON

"Billing System"

Submitted in partial fulfillment of the requirement for the award of Bachelor degree in Computer Science and Engineering

for Session 2016 - 2020

Submitted by

Sahil Khan

Computer Science and Engineering

(BU2016UGCS046)

UNDER THE SUPERVISION OF

Mrs. Parul Gazta

Assistant. Professor,
School of Computer Science & Engineering,
Bahra University

Submitted To

School of Computer Science and Engineering

Bahra University, Shimla (Waknaghat) Distt solan, Himachal Pradesh – India

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CERTIFICATE OF ORIGINALITY

This is to certify that, the work entitled **"Billing System"** submitted by **Sahil Khan** having roll no **BU2016UGCS046** in partial fulfillment of the requirement for the award of degree of **Bachelor in Computer Science and Engineering,** Bahra University, Shimla Hills, Solan (H.P.) has been carried out under the supervision of **Mrs, Parul Gazta, (Assistant Professor)**, School of Computer Science & Engineering. This work has not been submitted partially or fully to any other University or Institute for the award of any other degree.

Date:

Place:

Dr. Priyanka Sharma
Associate Professor & Head
School of Computer Science & Engineering

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CERTIFICATE BY SUPERVISOR

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

DOCUMENT BY THE COMPANY

QA InfoTech
Your Software Testing Partner

Date: 15-Jul-20

RELIEVING LETTER

This is to certify that Sahil Khan (Emp ID: 4987) has worked in our organization from 20-Jan-20 till 9-Jun-20 and has been relieved from services as Software Engineer with effect 9-Jun-20.

We wish Sahil Khan all the best for all future endeavours. For any question or clarifications, please write to hrbgvqa@qainfotech.com .

For QA InfoTech Software Services (P) Limited,

Mayuri Jagannathan
(Vice President, Human Resources)

QA InfoTech Software Service (P) Ltd.
A-8, Sector- 68, Noida, UP, India- 201309
Tel.- 8010180180, Fax: +91-120-6101801
Regd. Office:- 2894/B, Hakim Manzil,
Kumbharwali Pole, Shahpur, Ahmedabad,
Gujarat, 380001, India
Email ID- info@qainfotech.com, www.qainfotech.com
CIN No.: U74999GJ2017PTC098705

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

ACKNOWLEDGMENT

On the very outset of this report, I would like to extend my sincere and heartfelt obligation towards all the people who have helped me in this endeavor. Without their active guidance, help, cooperation and encouragement. I would not have made headway in the project. I acknowledge with deep sense of gratitude and most sincere appreciation, the valuable guidance and unfailing encouragement rendered to me by **Er. Rahul Gupta (Software Engineer at QA Infotech)** for his proficient and enthusiastic guidance, useful encouragement and immense help. I have deep sense of admiration for their inmate goodness and inexhaustible enthusiasm. I wish to extend my sincere gratitude to **Mrs. Priyanka Sharma (HOD of CSE Department) and Mrs. Parul Gazta (Assistant Professor)** for their guidance, encouragement and valuable suggestions which proved extremely useful and helpful in completion of this industrial training. My heartfelt gratitude and indebtedness goes to all teachers and guidance group who with their encouraging, caring words, constructive criticism and segmentation have contributed directly or indirectly in a significant way towards completion of this training. My special thanks goes to my friends whose support and encouragement have been a constant source of assurance, guidance, strength, and inspection to me. I am immensely grateful to my parents, my family. They have always supported me and taught me the things that matter most in life. I am proudly grateful to all of them.

Sahil Khan

Signature

Date

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

ABSTRACT

Nowadays it's very important to manage customer's data easily and effectively. This software will help the salespersons in creating and managing the records pertaining to customers. The product will help the user to work in a highly effective and efficient environment.

The salespersons have been recording the customer information in the past and even in the present through their manual efforts on their ledger. And indeed, it consumes considerable time and energy that could be utilized in better productive activities. Apart from that, with increasing customer Strength, the task of managing information of each customer is indeed a cumbersome task.

There is a lot of reason for the introduction of this project. In the manual ledger System, there are several inefficiencies that the salesperson faces. The information retrieval is one of the foremost problems. It is very difficult to get the previous invoice data of the customers from the ledger. Large records-books have to be maintained where relevant and irrelevant information has to be stored which is a very untidy and clumsy process.

On the other hand, many inherent problems exist in any manual system. Usually, they lack efficiency. Less efficiency has a great impact on the productivity of any human being keeping the data up-to-date.

The project "Billing System" is developed to make the system reliable, easier, fast, and more informative.

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

List of Figures

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

List of Tables

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

Index

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

Contents

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CHAPTER 1

INTRODUCTION

One of the foremost vital side to running any business – big or small, in private control or public corporation - is billing. without billing, cash flow can dry up and business can collapse.

Invoicing is the initial billing stage. this can be wherever your sales order or estimate becomes an actual charge, complete with elements, labor, sales tax, shipping, or any whatever charges apply to your specific business. The invoice is vital to the billing process as a result of it offers your client the firm and final price.

The invoice is important to the billing process because it gives your customer the firm and final price. The statement provides your customer with a reminder of when their bill is due, how much the bill is, and any interest or other fees incurred since the date of purchase or the last billing cycle.

An invoice, bill, or tab is a commercial document issued by a seller to a buyer, relating to a sale transaction and indicating the products, quantities, and agreed prices for products or services the seller had provided the buyer.

Billing is the process of sending an invoice (a bill) to customers for goods or services. Nowadays it is very important to get a product invoice from the seller because it is proof of the product given by the seller. Invoice helps both the buyer as well as the seller.

Today, salesperson needed a type of software which helps them in creating and managing the customer data. This customer data will help the salesperson to know the customer and also helps to calculate the profit and loss of the business. A system is needed to store the data for future uses. Approximate every salesperson needed a system that helps them in price calculations, tax calculations, customer data storage, and invoice generation.

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

1.1 Overview of Project

A billing system is an application that calculates the bills and generates the invoice. This helps the salesperson to add, calculate, and manage the product and tax-related information and able to generate an invoice. Billing System is a desktop-based application for daily needs stores and retail shops to manage their product billing and other billing related tasks. Billing System helps to store the product and customer-related data and can generate bills. Billing System can store and print the invoices from the database and helps to manage the product and invoice data.

Billing System is a desktop-based application that is used in calculates the bills and generate invoices. This application can use to create maintain customer invoice data and also for basic calculations with CGST and SGST tax calculation feature. This application helps the salesperson to manage the customer data. Billing System uses a database to store the generated invoice data and this feature also helps to check the old generated invoice data. By this system, the salesperson will able to create, maintain, and also able to print the invoice using an attached printer.

Billing System is a faster and easy to use software. It is extremely reliable software and also data inserted, updated, deleted easily.There is no need to use paper to store the user and product information. Owner can easily track and search the previous data of the user. Owner can also search the product data by generated invoice numbers.The motivation behind this project is to store the sold product information and also the customer information who bought the product. It also helps to calculate the sold product price with CGST and SGST prices

1.2 Purpose

The billing system can be used to replace the traditional way of storing and managing the data in the ledgers. The basic purpose of this system is to maintain the customer data and provide feature like invoice generation and invoice print option. The process of data management on the ledger is a very time consuming and risky job because the ledger can easily deteriorate. Also, the biggest problem with ledgers is to search for the old record. So, this system comes with features like creating invoice, calculation the price with different tax amounts, saved invoice search option, and invoice print option.

1.3 Scope

Billing System is a desktop-based application that is created to generate and maintain the sold product invoices. The software will display a view of calculations of every sold product and invoice. The system will store and recognize customer invoices. This system can use in stores, industries, malls, and shops for the purpose like the generation of purchase and sales report.

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CHAPTER 2

REQUIREMENT ANALYSIS

2.1 Introduction:

A software requirement is a capability needed by the user to solve a problem or to achieve an objective. In other words, the requirement is a software capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documentation. Ultimately, what we want to achieve is to develop quality software that meets customers' real needs on time and within budget.

Requirements analysis is critical to the success or failure of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

2.2 Feasibility Study:

A feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort, and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its work-ability, which is the impact on the organization, the ability to meet their user needs, and the effective use of resources. Thus when a new application is proposed it normally goes through a feasibility study before it is approved for development.

Three key considerations involved in the feasibility analysis are :

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

2.2.1 Technical Feasibility

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed. Technical issues raised during the investigation are:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within the latest technology.

Through the technology may become obsolete after some time, because never version of the same software supports older versions, the system may still be used. So there are minimal constraints involved with this project. The system has been developed using Java the project is technically feasible for development.

2.2.2 Economic Feasibility

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on the project, which will give the best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during the preliminary investigation:

• The costs conduct a full system investigation.

• The cost of the hardware and software.

• The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend on the proposed system. Also, all the resources are already available, it indicates the system is economically possible for development.

2.2.3 Social feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.3 Existing System

2.3.1 Analysis of Existing System

In the Existing System, there are some issues of data management and maintenance. These systems are slow and time-consuming. If the user wants to search some previous customer data then it will take much time for the search. For small shops and businesses, it is fine but for larger systems, it will consume much time which means loss. The management of data on excel files and ledgers is a bit difficult but data search is much difficult than data creation. So, that's the reason the user needs a new updated, and reliable system.

2.4 Proposed System

2.4.1 Analysis of Proposed System

Billing System is a desktop-based application that is created to generate and maintain the sold product invoices. The software will display a view of calculations of every sold product and invoice. The system will store and recognize customer invoices. This system can use in stores, industries, malls, and shops for the purpose like the generation of purchase and sales report. Billing System is a faster and easy to use software. It is extremely reliable software and also data inserted, updated, deleted easily.There is no need to use paper to store the user and product information. Owner can easily track and search the previous data of the user. Owner can also search the product data by generated invoice numbers.The motivation behind this project is to store the sold product information and also the customer information who bought the product. It also helps to calculate the sold product price with CGST and SGST prices

2.5 SYSTEM REQUIREMENTS

2.5.1 Hardware Requirements

• System : Pentium IV 2.4 GHz.

• Hard Disk : 20 GB.

• RAM : 256 Mb.

2.5.2 Software Requirements

• Operating system : Windows 7/8/10.

• Database: Microsoft Access Database

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CHAPTER 3

SYSTEM DESIGN

3.1 Introduction

Systems design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap and synergy with the disciplines of systems analysis, systems architecture and systems engineering. It describes desired features and operations in detail, including screen layouts, business rules, process diagrams, pseudo-code and other documentation.

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces. System design is the process of designing the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system. System Analysis is the process that decomposes a system into its component pieces for the purpose of defining how well those components interact to accomplish the set requirements. The purpose of the System Design process is to provide sufficient detailed data and information about the system and its system elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture.

3.2 System Architecture

A system architecture is a conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

A system architecture can consist of system components and the sub-systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages.

System architecture conveys the informational content of the elements consisting of a system, the relationships among those elements, and the rules governing those relationships.

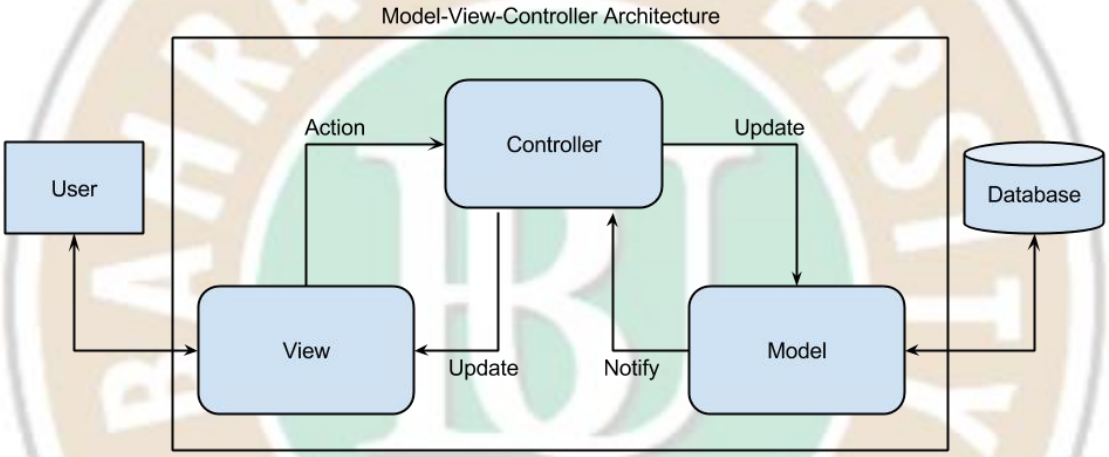


Fig 3.1 System Architecture

3.3 Use Case Diagram

A use case diagram is a dynamic or behavior diagram in Unified Modeling Language. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system.Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities.

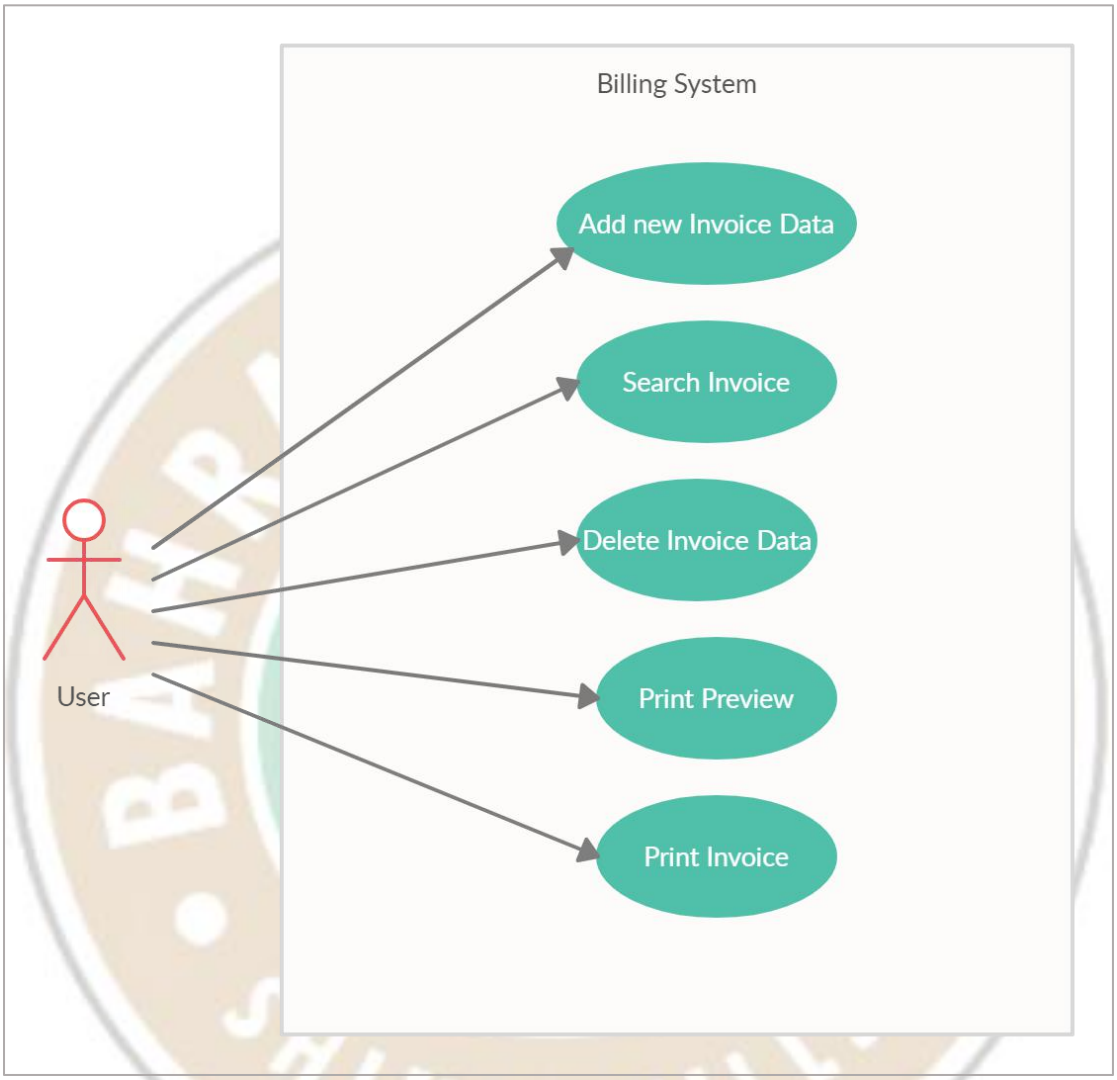


Fig 3.2 Use Case Diagram

3.4 Data Flow Diagram

Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation. DFD graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system.

The visual representation makes it a good communication tool between User and System designer. The structure of DFD allows starting from a broad overview and expands it to a hierarchy of detailed diagrams.

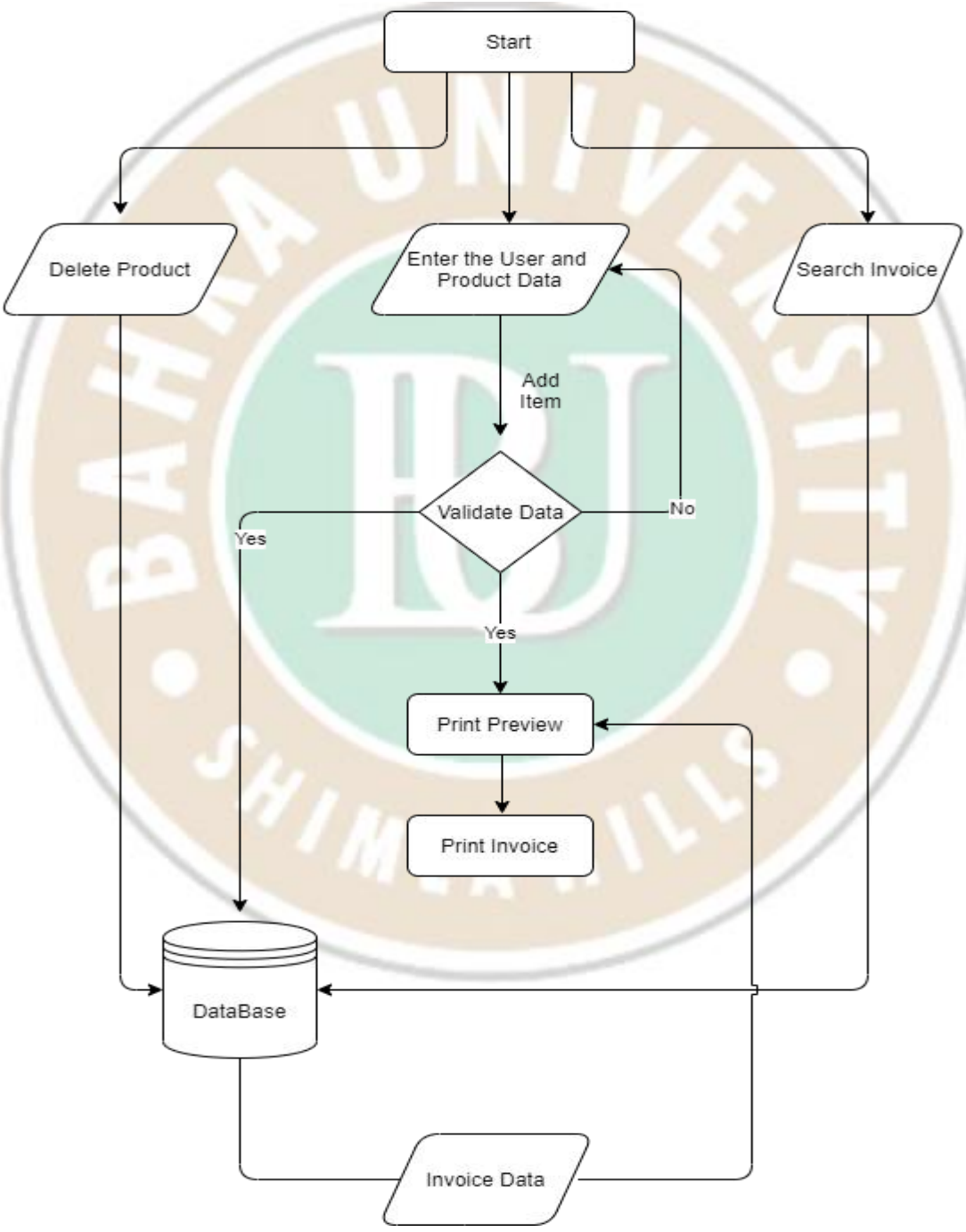


Fig 3.3 Data Flow Diagram

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CHAPTER 4

CODING

4.1 Form.cs

```
using BillingSystem.Model;
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Linq;
using System.Windows.Forms;
using System.Data.OleDb;
using System.Data;
using System.Data.SqlClient;

namespace BillingSystem
{
    public partial class Form1 : Form
    {

        static String conString = @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=D:\Projects\Dot NET\BillingSystem\BillingSystem\Database\productInfo.accdb; Persist Security Info = False; ";

        OleDbConnection connection = new OleDbConnection(conString);


        public Form1()
        {
            InitializeComponent();
        }


        private void Form1_Load(object sender, EventArgs e)
        {
            this.sweetDataTableAdapter.Fill(this.productInfoDataSet.SweetData);

        }
```

```
        //Variables
        int countValue = 0;
        int num = 1;
        private List<Allitems> GridItems = new List<Allitems>();
        double itemProduct;
        double totalPrice;
        double itemFinalPrice;
        double cgstCalValue;
        double valuecgst;
        double valuesgst;
        double sgstCalValue;
        double calculatedCgst;
        double calculatedSgst;
        double totalGST;
        double labourPrice;
        bool QtyTypeCheck;
        bool PriceTypeCheck;
        int index;
        double Invoiceval;
        DataGridViewRow selectedRow;


        /// First Focus
        private void add_Item(object sender, EventArgs e)
        {
            double temp;
            bool isNumber = double.TryParse(invoiceNo.Text.Trim(), out temp);

                if (invoiceNo.Text == string.Empty || !isNumber)
                {
                    MessageBox.Show("Enter Valid Invoice Number", "Error",
MessageBoxButtons.OK);
                }
                else
                {
                    new_Order.Enabled = true;

                    if (itemName.Text != string.Empty && QtyTypeCheck ==
true && PriceTypeCheck == true && itemQantity.Text != string.Empty &&
itemQantity.Text != string.Empty)
                    {
                        num++;
                    }
```

```
                    if (showMessage())
                    {
                        // hsn default Value
                        if (hsnCode.Text == string.Empty && gstinNo.Text == string.Empty)
                        {
                            hsnCode.Text = "20001";
                            gstinNo.Text = "02ADZPN7340J2Z2";
                        }

                        dataGridConnect();
                        gstData();

                        /// DataBase /////
                        try
                        {
                            OleDbCommand command = connection.CreateCommand();
                            connection.Open();
                            Invoiceval = Convert.ToDouble(invoiceNo.Text);
                            command.CommandText = "Insert into SweetData (SrNo, InvoiceNo, ProductName, Rate, Qantity, Amount, InvoiceDate, State1, StateCode1, TransportMode, VehicalNo, DateOfSupply, PlaceOfSupply, NameAndAdd, GstinNo, State2, StateCode2, Labour, HsnCode, CgstRate, SgstRate, CgstAmount, SgstAmount) values('" + num + "', '" + Invoiceval + "', '" + itemName.Text + "', '" + itemPrice.Text + "','" + itemQantity.Text + "','" + itemProduct + "', '" + invoiceDate.Text + "', '" + state.Text + "', '" + stateCode.Text + "', '" + transpostMode.Text + "', '" + vehicalNo.Text + "', '" + dateOfSupply.Text + "','" + placeOfSupply.Text + "','" + nameAndAddress.Text + "','" + gstinNo.Text + "','" + state2.Text + "','" + state2Code.Text + "','" + labourValue.Text + "', '" + hsnCode.Text + "','" + "2.5%" + "','" + "2.5 %" + "','" + itemProduct * 0.025 + "','" + itemProduct * 0.025 + "')";
                            command.Connection = connection;
                            command.ExecuteNonQuery();
                            connection.Close();
                        }
                        catch (Exception ex)
                        {
                            MessageBox.Show("" + ex);
                        }
                    }
```

```
                    itemName.Clear();
                    itemQantity.Clear();
                    itemPrice.Clear();
                    itemName.Focus();
                    add_item.Enabled = false;
                    deleteItem.Enabled = true;
                }

        }

        private bool isInvoiceExisted(string invoiceNumber)
        {
            bool result = false;
            try
            {
                OleDbCommand cmd = connection.CreateCommand();
                connection.Open();
                string query = "Select Count(*) From SweetData Where
InvoiceNo = " + invoiceNumber;
                cmd.CommandText = query;
                int rowCount = (int)cmd.ExecuteScalar();
                connection.Close();
                if (rowCount > 0)
                    result = true;
                else
                    result = false;

            }catch(Exception ex)
            {
                MessageBox.Show(ex.ToString());
            }
            return result;
        }

        private void resetValues_Click(object sender, EventArgs e)
        {
            countValue++;
            num = 0;

            itemName.Clear();
            itemQantity.Clear();
            itemPrice.Clear();
            invoiceNo.Clear();
```

```
            totalPrice = 0;
            itemFinalPrice = 0;
            itemProduct = 0;

            itemName.Focus();

            /// AMOUNT SHOW ITEMS
            allProductTotalAmount.Text = "0";
            cgstAmount.Text = "0";
            sgstAmount.Text = "0";
            grandTotal.Text = "0";

            dataGridView.DataSource = null;
            GridItems.Clear();
        }

        private void valueEntry()
        {
            if (itemQantity.Text != string.Empty && itemPrice.Text != string.Empty)
            {
                double rate = Convert.ToDouble(itemPrice.Text);
                double qty = Convert.ToDouble(itemQantity.Text);
                itemProduct = qty * rate;
            }
            itemFinalPrice = itemFinalPrice + itemProduct;
        }
        ///
        /// CalCulations
        ///

        private void gstData()
        {
            valueEntry();

            //CGST
            valuecgst = 0.025;
            cgstCalValue = itemFinalPrice * valuecgst;

            //SGST
            valuesgst = 0.025;
            sgstCalValue = itemFinalPrice * valuesgst;
```

```
            //CGST AMOUNT.....
            calculatedCgst =    cgstCalValue;
            //SGST AMOUNT.....
            calculatedSgst =    sgstCalValue;
            //Total GST
            totalGST = cgstCalValue + sgstCalValue;

            ////////////////////////////////////        Labour Amount ////////////////
            if (labourValue.Text == string.Empty)
            {
                labourPrice = 0;
            }
            else
            {
                labourPrice = Convert.ToDouble(labourValue.Text);
            }

            totalPrice = itemFinalPrice + totalGST + labourPrice;

            //// Show CGST AND SGST AMOUNT
            cgstAmount.Text = calculatedCgst.ToString();
            sgstAmount.Text = calculatedSgst.ToString();

            //// SHOW ALL PRODUCT SUM without GST
            allProductTotalAmount.Text = itemFinalPrice.ToString();

            // Total Price with GST
            grandTotal.Text = totalPrice.ToString();
        }

        ///Dialog Message on empty textArea
        private bool showMessage()
        {
            if (itemName.Text.Trim() == string.Empty)
            {
                MessageBox.Show("Enter Product Name", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                itemName.Focus();
                return false;
            }

            if (itemQantity.Text.Trim() == string.Empty)
            {
```

```
                MessageBox.Show("Enter Qantity of the Product", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                itemQantity.Focus();
                return false;
            }
            else
            {
                double temp;
                bool isNumber = double.TryParse(itemQantity.Text.Trim(), out
temp);
                QtyTypeCheck = isNumber;
                if (!isNumber)
                {
                    MessageBox.Show("Enter a Numeric Qantity Value",
"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                    itemQantity.Clear();
                    itemQantity.Focus();
                    return false;
                }
            }
            if (itemPrice.Text.Trim() == string.Empty)
            {
                MessageBox.Show("Enter Price of the Product", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                itemPrice.Focus();
                return false;
            }
            else
            {
                double temp;
                bool isNumber = double.TryParse(itemPrice.Text.Trim(), out
temp);
                PriceTypeCheck = isNumber;
                if (!isNumber)
                {
                    MessageBox.Show("Enter a Numeric Price Value", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                    itemPrice.Clear();
                    itemPrice.Focus();
                    return false;
                }
            }
            return true;            }
```

```
            // // // // // // // // // //                Data Grid View                // //
// // / // / // //
            public void dataGridConnect()
            {
                Allitems items = new Allitems()
                {
                    SrNo = num,
                    ProductName = itemName.Text,
                    Qantity = Convert.ToDouble(itemQantity.Text.Trim()),
                    Rate = Convert.ToDouble(itemPrice.Text.Trim()),
                    TotalAmount = Convert.ToDouble(itemQantity.Text.Trim()) *
Convert.ToDouble(itemPrice.Text.Trim()),
                    HsnCode = hsnCode.Text,
                    CgstRate = "2.5%",
                    SgstRate = "2.5%",
                    CgstAmount = (((Convert.ToDouble(itemQantity.Text.Trim()) *
Convert.ToDouble(itemPrice.Text.Trim())) * 0.025)),
                    SgstAmount = (((Convert.ToDouble(itemQantity.Text.Trim()) *
Convert.ToDouble(itemPrice.Text.Trim())) * 0.025))

                };
                GridItems.Add(items);
                dataGridView.DataSource = null;
                dataGridView.DataSource = GridItems;
            }

            private void itemPrice_TextChanged(object sender, EventArgs e)
            {
                add_item.Enabled = true;
            }
            //// //// //// //// ////                EnterPress Events..... //// //// ////
//// ////
            private void onEnter(object sender, KeyEventArgs e)
            {
                if (e.KeyCode == Keys.Enter)
                {
                    itemQantity.Focus();
                }
                else if (e.KeyCode == Keys.Escape)
                {
                    new_Order.Focus();
                }
            }
```

```
private void StateCodeToTransportMode(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Enter)
    {
        transpostMode.Focus();
    }
    else if (e.KeyCode == Keys.Up)
    {
        state.Focus();
    }
}

private void PlaceOfSupplyToName(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Enter)
    {
        nameAndAddress.Focus();
    }
    else if (e.KeyCode == Keys.Up)
    {
        dateOfSupply.Focus();
    }
}

private void state2CodeToNewOrder(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Enter)
    {
        itemName.Focus();
    }
    else if (e.KeyCode == Keys.Up)
    {
        state2.Focus();
    }
}

private void invoiceNoToDate(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Enter)
    {
        invoiceDate.Focus();
    }
}
```

```
private void DateToState(object sender, KeyEventArgs e)
{
	if (e.KeyCode == Keys.Enter)
	{
		state.Focus();
	}
	else if (e.KeyCode == Keys.Up)
	{
		invoiceNo.Focus();
	}
}

private void StateToCode(object sender, KeyEventArgs e)
{
	if (e.KeyCode == Keys.Enter)
	{
		stateCode.Focus();
	}
	else if (e.KeyCode == Keys.Up)
	{
		invoiceDate.Focus();
	}
}

private void QtyToRate(object sender, KeyEventArgs e)
{
	if (e.KeyCode == Keys.Up)
	{
		itemName.Focus();
	}
	else if (e.KeyCode == Keys.Enter)
	{
		itemPrice.Focus();
	}
	else if (e.KeyCode == Keys.Escape)
	{
		new_Order.Focus();
	}

}

private void itemPrice_KeyDown(object sender, KeyEventArgs e)
{
```

```
    if (e.KeyCode == Keys.Up)
     {
          itemQantity.Focus();
     }
     else if (e.KeyCode == Keys.Enter)
     {
          add_item.Focus();
     }

     else if (e.KeyCode == Keys.Escape)
     {
          new_Order.Focus();
     }
}

private void transpostMode_KeyDown(object sender, KeyEventArgs e)
{
     if (e.KeyCode == Keys.Enter)
     {
          vehicalNo.Focus();

     }
     else if (e.KeyCode == Keys.Up)
     {
          stateCode.Focus();
     }
}

private void vehicalNo_KeyDown(object sender, KeyEventArgs e)
{
     if (e.KeyCode == Keys.Enter)
     {
          dateOfSupply.Focus();
     }
     else if (e.KeyCode == Keys.Up)
     {
          transpostMode.Focus();
     }
}

private void dateOfSupply_KeyDown(object sender, KeyEventArgs e)
{
     if (e.KeyCode == Keys.Enter)
```

```
        {
            placeOfSupply.Focus();

        }
        else if (e.KeyCode == Keys.Up)
        {
            vehicalNo.Focus();
        }
}

private void nameAndAddress_KeyDown(object sender, KeyEventArgs e)
{
        if (e.KeyCode == Keys.Enter)
        {
            gstinNo.Focus();

        }
        else if (e.KeyCode == Keys.Up)
        {
            placeOfSupply.Focus();
        }
}

private void state2_KeyDown(object sender, KeyEventArgs e)
{
        if (e.KeyCode == Keys.Enter)
        {
            state2Code.Focus();
        }
        else if (e.KeyCode == Keys.Up)
        {
            gstinNo.Focus();
        }
}

private void gstinNo_KeyDown(object sender, KeyEventArgs e)
{
        if (e.KeyCode == Keys.Enter)
        {
            state2.Focus();
        }
        else if (e.KeyCode == Keys.Up)
        {
```

```
                nameAndAddress.Focus();
            }
        }
        //
        //
        //..... PrintPreview And PrintDocument.....

        private void printPreView(object sender, EventArgs e)
        {
            PrintPreviewDialog BSprintPreviewDialog = new PrintPreviewDialog();
            BSprintPreviewDialog.Document = BSprintDocument;

            if(BSprintPreviewDialog.IsDisposed == true)
            {
                BSprintPreviewDialog = new PrintPreviewDialog();
            }
            BSprintPreviewDialog.Show();
        }

        private void BSprintDocument_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)
        {
            Pen BlackPen = new Pen(Color.Black, 1);



            e.Graphics.DrawString("TAX INVOICE", new Font("Arial", 9, FontStyle.Bold), Brushes.Black, new Point(410, 2));
            e.Graphics.DrawString("Billing System", new Font("Arial", 32, FontStyle.Regular), Brushes.Black, new Point(300, 8));


            //e.Graphics.DrawString(addressOfCorp.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(255, 54));
            e.Graphics.DrawString(GstIN.Text, new Font("Arial", 7, FontStyle.Bold), Brushes.Black, new Point(700, 40));
            e.Graphics.DrawString("Phone : 0177-2841694", new Font("Arial", 7, FontStyle.Bold), Brushes.Black, new Point(700, 20));

            //  e.Graphics.DrawString("Total Amount :" + totalAmount.Text, new Font("Arial", 12, FontStyle.Regular), Brushes.Black, new Point(10, 10));
```

```
                // Reverse Charge
                e.Graphics.DrawString("Reverse Charge", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(10, 76));
                e.Graphics.DrawString("Invoce No.: ", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(10, 100));
                e.Graphics.DrawString(invoiceNo.Text, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(80, 100));
                e.Graphics.DrawString("Invoce Date: ", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(210, 100));
                e.Graphics.DrawString(invoiceDate.Text, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(290, 100));
                e.Graphics.DrawString("State : ", new Font("Arial", 9, FontStyle.Bold),
Brushes.Black, new Point(10, 124));
                e.Graphics.DrawString(state.Text, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(80, 124));
                e.Graphics.DrawString("State Code: ", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(210, 124));
                e.Graphics.DrawString(stateCode.Text, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(290, 124));

                //Transpotation Info
                e.Graphics.DrawString("Transport Mode ", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(430, 76));
                e.Graphics.DrawString(transpostMode.Text, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(540, 76));
                e.Graphics.DrawString("Vehical NO. : ", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(430, 100));
                e.Graphics.DrawString(vehicalNo.Text, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(520, 100));
                e.Graphics.DrawString("Date of Supply :", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(430, 124));
                e.Graphics.DrawString(dateOfSupply.Text, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(530, 124));
                e.Graphics.DrawString("Place Of Supply :", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(600, 124));
                e.Graphics.DrawString(placeOfSupply.Text, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(710, 124));
                //Details Of Receiver
                e.Graphics.DrawString("Details of Receiver Billed to/Consignee
Shipped to", new Font("Arial", 9, FontStyle.Bold), Brushes.Black, new Point(10,
150));
```

```
            e.Graphics.DrawString("Name & Address :", new Font("Arial", 9, FontStyle.Bold), Brushes.Black, new Point(10, 174));
            e.Graphics.DrawString(nameAndAddress.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(130, 174));
            e.Graphics.DrawString("GSTINO :", new Font("Arial", 9, FontStyle.Bold), Brushes.Black, new Point(180, 220));
            e.Graphics.DrawString(gstinNo.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(250, 220));
            e.Graphics.DrawString("State :", new Font("Arial", 9, FontStyle.Bold), Brushes.Black, new Point(380, 220));
            e.Graphics.DrawString(state2.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(430, 220));
            e.Graphics.DrawString("State Code :", new Font("Arial", 9, FontStyle.Bold), Brushes.Black, new Point(600, 220));
            e.Graphics.DrawString(state2.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(690, 220));

            //e.Graphics.DrawString(placeOfSupply.Text, new Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(700, 124));

            // // // // // // // // // // // //                    Lines.....
// // // // //
            Point pointOne = new Point(0, 68);
            Point pointSecond = new Point(850, 68);
            Point p3 = new Point(0, 140);
            Point p4 = new Point(850, 140);
            Point p5 = new Point(425, 62);
            Point p6 = new Point(425, 140);
            Point p7 = new Point(0, 400);
            Point p8 = new Point(855, 400);
            Point p9 = new Point(0, 800);
            Point p10 = new Point(855, 800);
            // // Table Lines
            Point p11 = new Point(218, 240);
            Point p12 = new Point(218, 800);
            Point p13 = new Point(288, 240);
            Point p14 = new Point(288, 800);
            Point p15 = new Point(348, 240);
            Point p16 = new Point(348, 800);
            //Mid Line
            Point p17 = new Point(418, 240);
            Point p18 = new Point(418, 950);
            Point p19 = new Point(518, 240);
```

```
Point p20 = new Point(518, 800);
 Point p21 = new Point(578, 260);
 Point p22 = new Point(578, 800);
 Point p23 = new Point(668, 240);
 Point p24 = new Point(668, 800);
 Point p25 = new Point(758, 260);
 Point p26 = new Point(758, 800);
 //// Below Table lines
 Point p27 = new Point(418, 830);
 Point p28 = new Point(855, 830);
 Point p29 = new Point(418, 860);
 Point p30 = new Point(855, 860);
 Point p31 = new Point(418, 890);
 Point p32 = new Point(855, 890);
 Point p33 = new Point(418, 920);
 Point p34 = new Point(855, 920);
 Point p35 = new Point(0, 950);
 Point p36 = new Point(855, 950);

 // Gst coloum line
 Point p37 = new Point(518, 260);
 Point p38 = new Point(850, 260);
 Rectangle r1 = new Rectangle(0, 240, 850, 40);
 e.Graphics.DrawRectangle(BlackPen, r1);
 e.Graphics.DrawLine(BlackPen, pointOne, pointSecond );
 e.Graphics.DrawLine(BlackPen, p3, p4);
 e.Graphics.DrawLine(BlackPen, p5, p6);
 e.Graphics.DrawLine(BlackPen, p9, p10);
 e.Graphics.DrawLine(BlackPen, p11, p12);
 e.Graphics.DrawLine(BlackPen, p13, p14);
 e.Graphics.DrawLine(BlackPen, p15, p16);
 e.Graphics.DrawLine(BlackPen, p17, p18);
 e.Graphics.DrawLine(BlackPen, p19, p20);
 e.Graphics.DrawLine(BlackPen, p21, p22);
 e.Graphics.DrawLine(BlackPen, p23, p24);
 e.Graphics.DrawLine(BlackPen, p25, p26);
 e.Graphics.DrawLine(BlackPen, p27, p28);
 e.Graphics.DrawLine(BlackPen, p29, p30);
 e.Graphics.DrawLine(BlackPen, p31, p32);
 e.Graphics.DrawLine(BlackPen, p33, p34);
 e.Graphics.DrawLine(BlackPen, p35, p36);
 e.Graphics.DrawLine(BlackPen, p37, p38);
```

```
                // // // // //    List Headings ///
                e.Graphics.DrawString("Discription of product", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(10, 242));
                e.Graphics.DrawString("HSN \nCode", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(220, 242));
                e.Graphics.DrawString("Qty.", new Font("Arial", 9, FontStyle.Bold),
Brushes.Black, new Point(290, 242));
                e.Graphics.DrawString("Rate \n(Rs)", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(350, 242));
                e.Graphics.DrawString("Amount(Rs)", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(420, 242));
                e.Graphics.DrawString("\tCGST\nRate", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(520, 248));
                e.Graphics.DrawString("\nAmount", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(580, 248));
                e.Graphics.DrawString("\tSGST\nRate", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(670, 248));
                e.Graphics.DrawString("\nAmount", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(760, 248));
                // // // // // List Data // // // // //
                int Ypoint = 290;
                foreach (var i in GridItems)
                {   Graphics.DrawString(i.ProductName, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(10, Ypoint));
                    e.Graphics.DrawString(i.HsnCode, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(220, Ypoint));
                    e.Graphics.DrawString(i.Qantity.ToString(), new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(290, Ypoint));
                    e.Graphics.DrawString(i.Rate.ToString(), new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(350, Ypoint));
                    e.Graphics.DrawString(i.TotalAmount.ToString(), new
Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(420, Ypoint));
                    e.Graphics.DrawString(i.CgstRate, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(520, Ypoint));
                    e.Graphics.DrawString(i.CgstAmount.ToString(), new
Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(580, Ypoint));
                    e.Graphics.DrawString(i.SgstRate, new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(670, Ypoint));
                    e.Graphics.DrawString(i.SgstAmount.ToString(), new
Font("Arial", 8, FontStyle.Regular), Brushes.Black, new Point(760, Ypoint));

                    Ypoint += 25;
                }
```

```
            e.Graphics.DrawString("Amount :", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(418, 810));
            e.Graphics.DrawString(allProductTotalAmount.Text, new Font("Arial",
9, FontStyle.Regular), Brushes.Black, new Point(600, 810));
            e.Graphics.DrawString("Labour/Cartage :", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(418, 840));
            e.Graphics.DrawString(labourValue.Text, new Font("Arial", 9,
FontStyle.Regular), Brushes.Black, new Point(600, 840));
            e.Graphics.DrawString("CGST Amount :", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(418, 870));
            e.Graphics.DrawString(cgstAmount.Text, new Font("Arial", 9,
FontStyle.Regular), Brushes.Black, new Point(600, 870));
            e.Graphics.DrawString("SGST Amount :", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(418, 900));
            e.Graphics.DrawString(sgstAmount.Text, new Font("Arial", 9,
FontStyle.Regular), Brushes.Black, new Point(600, 900));
            e.Graphics.DrawString("Grand Total :", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(418, 930));
            e.Graphics.DrawString(grandTotal.Text, new Font("Arial", 9,
FontStyle.Regular), Brushes.Black, new Point(600, 930));


            // /// // / / Footer

            e.Graphics.DrawString("Note :", new Font("Arial", 9, FontStyle.Bold),
Brushes.Black, new Point(10, 950));
            //e.Graphics.DrawString("All Subjects To Shimla Jurisdiction
Only\nGoods once sold will not be taken back.\nChanna, Paneer & Khoya Products to
be consumed same day.", new Font("Arial", 8, FontStyle.Regular), Brushes.Black,
new Point(40, 970));

            //e.Graphics.DrawString("For R.D.T. Enterprises", new Font("Arial", 9,
FontStyle.Bold), Brushes.Black, new Point(700, 950));
            e.Graphics.DrawString("Authority Signatory", new Font("Arial", 8,
FontStyle.Regular), Brushes.Black, new Point(700, 1050));
        }

        private void PrintButton_Click(object sender, EventArgs e)
        {
            BSprintDocument.Print();
        }
```

```
        private void delete_Click(object sender, EventArgs e)
        {
            try
            {
                int rowIndex = dataGridView.CurrentCell.RowIndex;
                GridItems.RemoveAt(rowIndex);
            }catch(Exception ex)
            {
                MessageBox.Show(""+ex);
            }
        }
        private void dataGridView_MouseDown(object sender, MouseEventArgs e)
        {
            try
            {
                if (e.Button == MouseButtons.Left)
                {
                    if (dataGridView.DataSource == null)
                    {
                        MessageBox.Show("List is Empty.", "Error");
                    }
                    else
                    {
                        var hti = dataGridView.HitTest(e.X, e.Y);
                        dataGridView.Rows[hti.RowIndex].Selected = true;
                    }
                }
            }
            catch(Exception )
            {
                MessageBox.Show("Select an Item.",
"Error",MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        private void dataGridView_CellClick(object sender,
DataGridViewCellEventArgs e)
        {
            index = e.RowIndex;
            selectedRow = dataGridView.Rows[index];
        }

        private void deleteItem_Click(object sender, EventArgs e)
        {
```

```
            if(dataGridView.DataSource != null)
            {
                int selectedvalue = dataGridView.CurrentCell.RowIndex + 1;
                string productName = dataGridView.SelectedCells[1].Value.ToString();
                MessageBox.Show("Invoice Number: "+invoiceNo.Text+"Sr No "+ selectedvalue+ "PName: " + productName);
                DialogResult result = new DialogResult();
                result = MessageBox.Show("Are You Sure To Delete Product At SrNo " + selectedvalue, "Delete Product", MessageBoxButtons.OKCancel);

                if (result == DialogResult.OK)
                {
                    try
                    {
                        int rowIndex = dataGridView.CurrentCell.RowIndex;
                        GridItems.RemoveAt(rowIndex);
                        dataGridView.DataSource = null;
                        dataGridView.DataSource = GridItems;
                        gridItemsCalculation();
                        num--;
                        deleteFromDataBase(selectedvalue, productName);
                        updateSrNumber(GridItems.Count(), selectedvalue);
                    }
                    catch (Exception)
                    {
                        MessageBox.Show("No item in the List.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                }

            }else
            {
                MessageBox.Show("List is Empty.","Error",MessageBoxButtons.OK,MessageBoxIcon.Error);
            }
        }

        private void gridItemsCalculation()
        {
            /// calculation
            double totalAfter = GridItems.Sum(X => X.TotalAmount);
            itemFinalPrice = totalAfter;
```

```
            double cgstAfter = totalAfter * 0.025;
            cgstCalValue = cgstAfter;
            double sgstAfter = totalAfter * 0.025;
            sgstCalValue = sgstAfter;
            double totalGstAfter = cgstCalValue + sgstCalValue;
            totalGST = totalGstAfter;
            double grandTotalAfter = totalAfter + totalGstAfter;
            totalPrice = grandTotalAfter;
            allProductTotalAmount.Text = totalAfter.ToString();
            cgstAmount.Text = cgstAfter.ToString();
            sgstAmount.Text = sgstAfter.ToString();
            grandTotal.Text = grandTotalAfter.ToString();
        }

        private void updateSrNumber(int totalItems, int selectedvalue)
        {
            for (int i = selectedvalue + 1 ; i <= totalItems + 1; i++)
            {
                try
                {
                    OleDbCommand cmd = connection.CreateCommand();
                    connection.Open();
                    string query = "Update SweetData Set SrNo ="+(i-1)+"
Where InvoiceNo =" + invoiceNo.Text + " And SrNo=" + i;
                    cmd.CommandText = query;
                    cmd.Connection = connection;
                    cmd.ExecuteNonQuery();
                    connection.Close();
                    MessageBox.Show("Current val: " + i);
                }
                catch (Exception e)
                {
                    MessageBox.Show(e.ToString());
                }
            }
        }

        private void deleteFromDataBase(int srNo, String productName)
        {
            try
            {
                OleDbCommand command = connection.CreateCommand();
                connection.Open();
```

```
                string query = "Delete From SweetData Where InvoiceNo =" + invoiceNo.Text + " And SrNo =" + srNo.ToString() + " And ProductName ='" + productName + "'";
                    command.CommandText = query;
                    command.Connection = connection;
                    command.ExecuteNonQuery();

                    MessageBox.Show("Data Deleted successfully.", "",MessageBoxButtons.OK);
                    connection.Close();
                }catch(Exception e)
                {
                    MessageBox.Show("" + e);
                }
            }

            private void dataGridView_KeyDown(object sender, KeyEventArgs e)
            {
                if(e.KeyCode == Keys.D)
                {
                    deleteItem.Focus();
                }
            }

            private void ShowDataBtn_Click(object sender, EventArgs e)
            {
                try
                {
                    String SearchInvoiceText = SearchInvoice.Text;
                    if (SearchInvoiceText != "")
                    {
                        connection.Open();
                        OleDbCommand com = new OleDbCommand();
                        string query = "Select SrNo, ProductName, Rate, Qantity, Amount, HsnCode, CgstRate, SgstRate, CgstAmount, SgstAmount, InvoiceDate, State1, StateCode1, TransportMode, VehicalNo, DateOfSupply, PlaceOfSupply, NameAndAdd, GstinNo, State2, StateCode2, Labour, HsnCode from SweetData Where InvoiceNo = " + SearchInvoiceText.ToString();
                        com.CommandText = query;
                        com.Connection = connection;
                        com.ExecuteNonQuery();
                        OleDbDataAdapter da = new OleDbDataAdapter(com);
                        DataTable dt = new DataTable();
```

```
da.Fill(dt);
GridItems.Clear();
foreach (DataRow row in dt.Rows)
{
        int srnum = (int) row[0];
        string pName = row[1].ToString();
        double qantity = (double) row[2];
        double rate = (double) row[3];
        double amount = (double) row[4];
        string hsncode = row[5].ToString();
        //string cgstR = row[6].ToString();
        //string sgstR = row[7].ToString();
        double cgstA = (double) row[8];
        double sgstA = (double) row[9];
        //Put values in text fields
        invoiceNo.Text = SearchInvoiceText;
        invoiceDate.Text = row[10].ToString();
        state.Text = row[11].ToString();
        stateCode.Text = row[12].ToString();
        transpostMode.Text = row[13].ToString();
        vehicalNo.Text = row[14].ToString();
        dateOfSupply.Text = row[15].ToString();
        placeOfSupply.Text = row[16].ToString();
        nameAndAddress.Text = row[17].ToString();
        gstinNo.Text = row[18].ToString();
        state2.Text = row[19].ToString();
        state2Code.Text = row[20].ToString();
        labourValue.Text = row[21].ToString();
        hsnCode.Text = row[22].ToString();
        Allitems items = new Allitems()
        {
            SrNo = srnum,
            ProductName = pName,
            Qantity = qantity,
            Rate = rate,
            TotalAmount = amount,
            HsnCode = hsncode,
            CgstRate = "2.5%",
            SgstRate = "2.5%",
            CgstAmount = cgstA,
            SgstAmount = sgstA
        };
```

```
                    GridItems.Add(items);
                    dataGridView.DataSource = null;
                    dataGridView.DataSource = GridItems;
                }
                gridItemsCalculation();
                deleteItem.Enabled = false;
                connection.Close();
                new_Order.Enabled = true;
                new_Order.Focus();
                add_item.Enabled = false;
            }
            else
            {
                MessageBox.Show("Enter any existing invoice number.");
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(""+ex);
        }
    }
    private void invoiceCheck(object sender, EventArgs e)
    {
        if (isInvoiceExisted(invoiceNo.Text))
        {
            MessageBox.Show("Invoice Number Already Existed.",
"Warning");
            itemName.Enabled = false;
            itemQantity.Enabled = false;
            itemPrice.Enabled = false;
            add_item.Enabled = false;
            invoiceNo.Clear();
            invoiceNo.Focus();
        }
        else
        {
            itemName.Enabled = true;
            itemQantity.Enabled = true;
            itemPrice.Enabled = true;
            add_item.Enabled = true;
        }
    }
}}
```

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CHAPTER 5

TESTING

Testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves the execution of a software component or system component to evaluate one or more properties of interest. Software testing also helps to identify errors, gaps, or missing requirements contrary to the actual requirements. Testing is important because software bugs could be expensive or even dangerous. Software bugs can potentially cause monetary and human loss.

5.1 Testing Strategies

5.1.1 Unit Testing

This software testing approach is followed by the programmer to test the unit of the program. It helps developers to know whether the individual unit of the code is working properly or not.

5.1.2 Integration testing

It focuses on the construction and design of the software. You need to see that the integrated units are working without errors or not.

5.1.3 System testing

In this method, your software is compiled as a whole and then tested as a whole. This testing strategy checks the functionality, security, portability, amongst others.

5.2 Test Case

A Test Case is a set of actions executed to verify a particular feature or functionality of your software application. A Test Case contains test steps, test data, precondition, post-condition developed for specific test scenario to verify any requirement. The test case includes specific variables or conditions, using which a testing engineer can compare expected and actual results to determine whether a software product is functioning as per the requirements of the customer.

Test Scenarios	Test Steps	Test Data	Test Result
Verify that the user can enter the input in the given text fields.	**1.** Open **Billing System** 2. Enter Data in input fields 3. Click on the add product button.	1. Enter an unique invoice number. 2. Fill the input fields.	**Pass**
Verify that user can search the invoice by invoice number.	**1.** Open **Billing System.** 2. Enter invoice number in previous invoice data input field. 3. Click on the **show data** button.	Enter an existing invoice number.	**Pass**
Verify that the data is visible in the data grid view after adding a product in the invoice.	**1.** Open **Billing System** 2. Enter Data in input fields 3. Click on the **add product** button.	1. Enter an unique invoice number. 2. Fill the input fields.	**Pass**
Verify that the user can delete the product from the data grid view after adding a product in the invoice.	1. Open **Billing System** 2. Enter Data in input fields 3. Click on the **add product** button. 4. Click on the product and click on the **delete selected item** button.	1. Enter an unique invoice number. 2. Fill the input fields.	**Pass**
Verify that the invoice data is correctly visible on the print preview mode.	1. Open **Billing System** 2. Enter Data in input fields 3. Click on the **add product** button. 4. Click on the **print preview** button.	1. Enter an unique invoice number. 2. Fill the input fields.	**Pass**

Table 5.1 Test Cases

5.3 Test Report

Billing System **Test Report** is generated only for the **"Table 5.1 Test cases".**

Project	Total Test	Passed	Skipped	Failed
Billing System	5	5	0	0

Table 5.2 Test Report

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CHAPTER 6

IMPLEMENTATION

Billing System is a desktop-based application that is created to generate and maintain the sold product invoices. The software will display a view of calculations of every sold product and invoice. The system will store and recognize customer invoices. This system can use in stores, industries, malls, and shops for the purpose like the generation of purchase and sales report.

Users who need to generate, store the invoice data, and also want to print the data can use this application. This application will help the user and makes the job easy and productive.

Billing System is developed by using technologies like Dot Net, C#, and MS Access Database. This application is developed using IDE Visual Studio 2019.

6.1 DOT NET

The .Net framework is a software development platform developed by Microsoft. The framework was meant to create applications, which would run on the Windows Platform. The first version of the .Net framework was released in the year 2002. The version was called .Net framework 1.0. The .Net framework has come a long way since then, and the current version is 4.7.1. The .Net framework can be used to create both - Form-based and Web-based applications. Web services can also be developed using the .Net framework. The framework also supports various programming languages such as Visual Basic and C#.

6.2 C#

C# is pronounced "C-Sharp". It is an object-oriented programming language created by Microsoft that runs on the .NET Framework. C# has roots from the C family, and the language is close to other popular languages like C++ and Java.

The first version was released in year 2002. The latest version, **C# 8**, was released in September 2019.C# is used for Mobile applications, Desktop applications, Web applications, Web services, Web sites, Games, VR, Database applications. It is one of the most popular programming language in the world. It is easy to learn and simple to use. It has a huge community support. C# is an object oriented language which gives a clear structure to programs and allows code to be reused, lowering development costs.

6.3 Microsoft Access

Microsoft Access is a database management system (DBMS) from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software-development tools. It is a member of the Microsoft Office suite of applications, included in the Professional and higher editions or sold separately. Microsoft Access stores data in its own format based on the Access Jet Database Engine. It can also import or link directly to data stored in other applications and databases. Software developers, data architects and power users can use Microsoft Access to develop application software. Like other Microsoft Office applications, Access is supported by Visual Basic for Applications (VBA), an object-based programming language that can reference a variety of objects including the legacy DAO (Data Access Objects), ActiveX Data Objects, and many other ActiveX components.

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

6.4 Snapshots of project

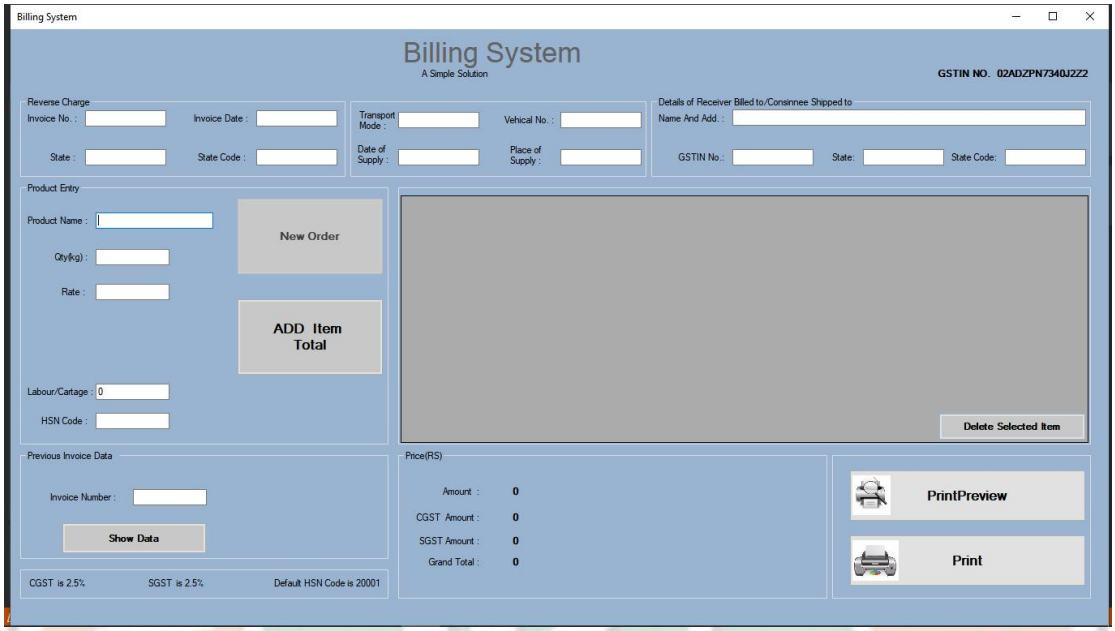
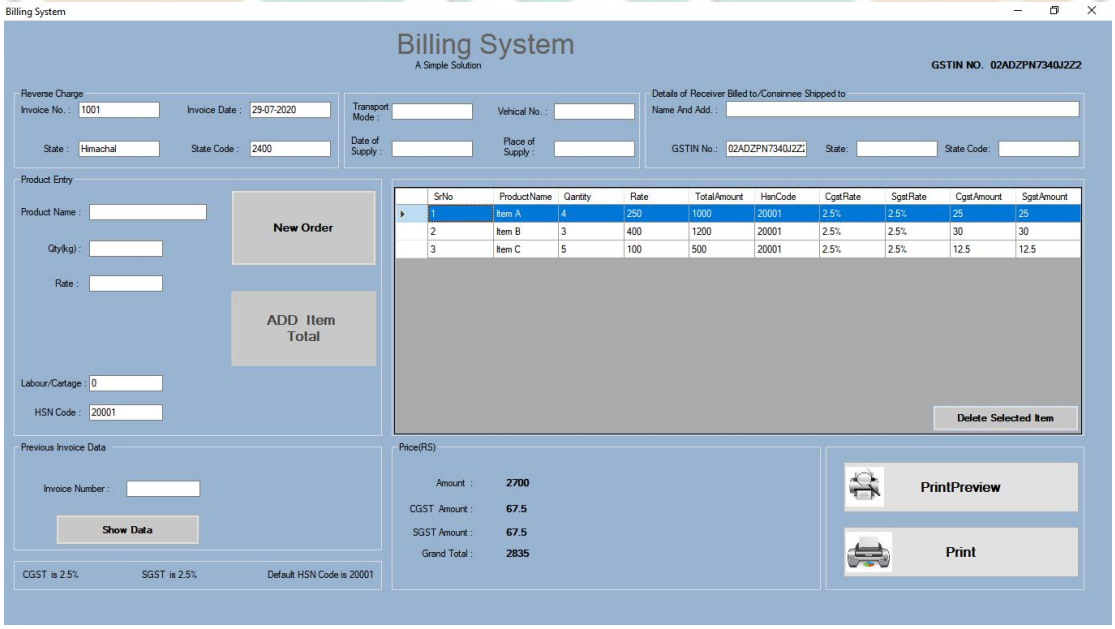


Fig 6.1 Billing System UI



6.2 Adding invoice data

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

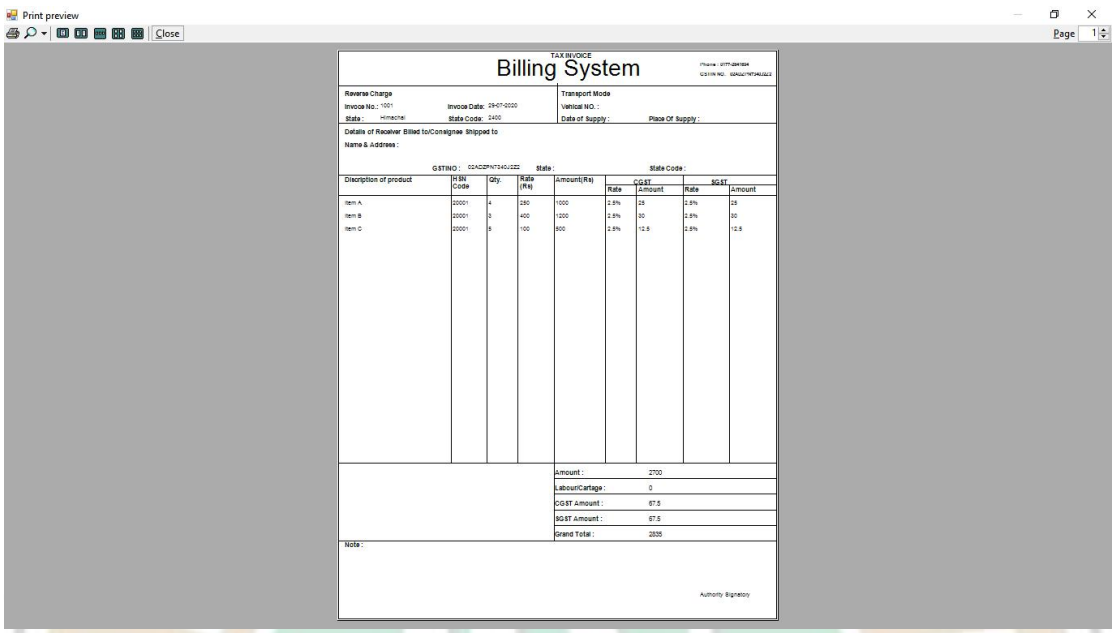


Fig 6.3 Print Layout

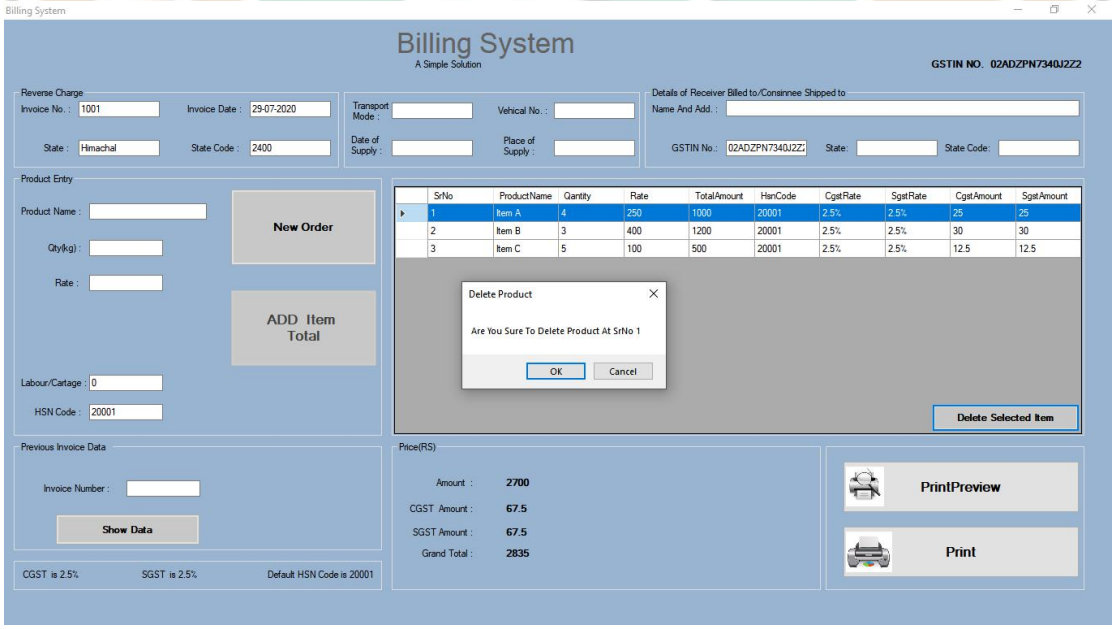


Fig 6.4 Delete item

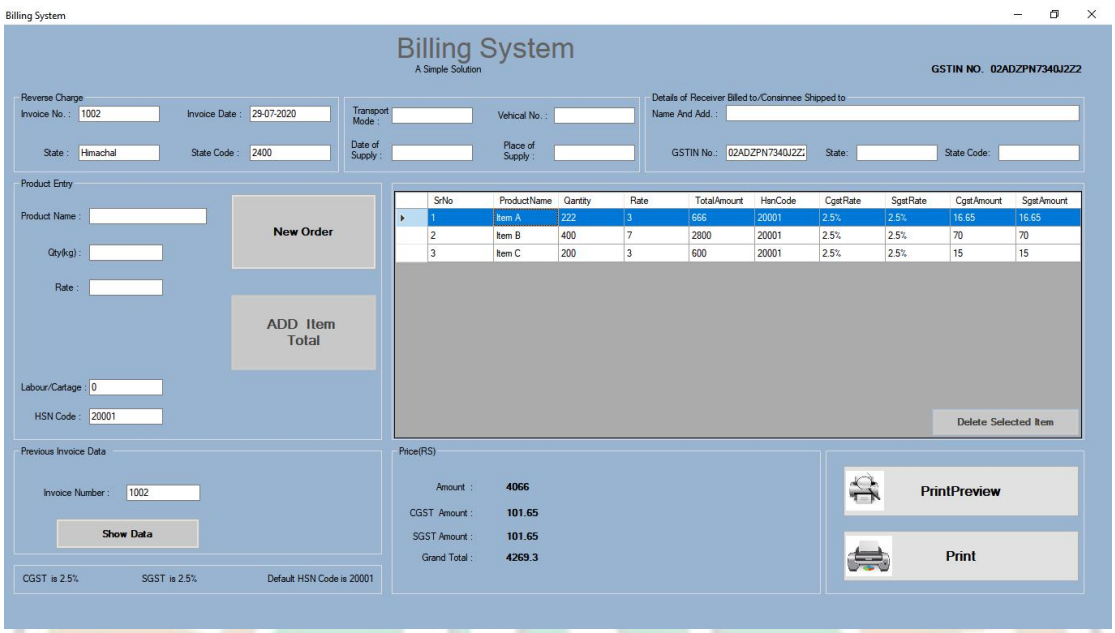


Fig 6.5 Search Invoice

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CHAPTER 7

RESULTS AND CONCLUSION

The Billing System is designed to generate bills when a customer orders an item. This application provides facility for adding customer details, adding item details and it automatically calculates the amount and generates a bill. The system has adequate scope for modification in future if it is necessary. This system can help multiple shop owners, small/big business owners by making their jobs easy by generating quick invoices and by saving customer data. This will make the job more quick and easy. It can be operated very easily and provides an user friendly functions and interface.

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

CHAPTER 8

FUTURE SCOPE

Future scope of the project:

Billing System is user friendly, easy to use application and having the basic functionality of generating and saving invoices. This system will be updated in the future with some advance functionalities :

- System UI can be updated to advance level.
- Customer invoice messaging feature can be added.
- The online payment system can be added.
- A monthly selling graph view feature can be added.
- Admin and Customer account creation feature can be added.

These features can be added to this project to make it an advanced level system. These features will help to improve the functionality of this project.

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

BIBLIOGRAPHY

- Google for problem solving
- https://dotnet.microsoft.com/learn/dotnet/what-is-dotnet
- https://docs.microsoft.com/en-us/dotnet/csharp/
- https://www.tutorialspoint.com/ms_access/ms_access_overview.htm
- https://visualstudio.microsoft.com/vs/

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

APPENDICES

APPENDIX 1

Dot Net Framework

The .Net framework is a software development platform developed by Microsoft. The framework was meant to create applications, which would run on the Windows Platform. The first version of the .Net framework was released in the year 2002. The version was called .Net framework 1.0. The .Net framework has come a long way since then, and the current version is 4.7.1.The .Net framework can be used to create both - Form-based and Web-based applications. Web services can also be developed using the .Net framework. The framework also supports various programming languages such as Visual Basic and C#. So developers can choose and select the language to develop the required application.

.NET is a developer platform made up of tools, programming languages, and libraries for building many different types of applications. There are various implementations of .NET. Each implementation allows .NET code to execute in different places—Linux, macOS, Windows, iOS, Android, and many more. .NET Framework is the original implementation of .NET. It supports running websites, services, desktop apps, and more on Windows. .NET Core is a cross-platform implementation for running websites, services, and console apps on Windows, Linux, and macOS. .NET Core is open source on GitHub. Xamarin/Mono is a .NET implementation for running apps on all the major mobile operating systems, including iOS and Android..NET Standard is a formal specification of the APIs that are common across .NET implementations. This allows the same code and libraries to run on different implementations.

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

APPENDIX 2

C#(C-Sharp)

C# (pronounced "See Sharp") is a modern, object-oriented, and type-safe programming language. C# has its roots in the C family of languages and will be immediately familiar to C, C++, Java, and JavaScript programmers. C# is an object-oriented language, but C# further includes support for component-oriented programming. Contemporary software design increasingly relies on software components in the form of self-contained and self-describing packages of functionality. Key to such components is that they present a programming model with properties, methods, and events. They have attributes that provide declarative information about the component. They incorporate their own documentation. C# provides language constructs to support directly these concepts, making C# a natural language in which to create and use software components. Several C# features aid in the construction of robust and durable applications. Garbage collection automatically reclaims memory occupied by unreachable unused objects. Exception handling provides a structured and extensible approach to error detection and recovery. The type-safe design of the language makes it impossible to read from uninitialized variables, to index arrays beyond their bounds, or to perform unchecked type casts. C# has a unified type system. All C# types, including primitive types such as int and double, inherit from a single root object type. Thus, all types share a set of common operations, and values of any type can be stored, transported, and operated upon in a consistent manner. Furthermore, C# supports both user-defined reference types and value types, allowing dynamic allocation of objects as well as in-line storage of lightweight structures.

BAHRA UNIVERSITY
LEARN • INNOVATE • EXCEL

LIST OF SYMBOLS AND ACRONYMS

ACRONYMS

Acronyms	Full Forms
DBMS	Database Management System
MVC	Modal View Controller
OOP	Object Oriented Programming
DAO	Data Access Objects
GUI	Graphical User Interface
CGST	Central Goods and Service Tax
SGST	State Goods and Service Tax
UI	User Interface
UC	Use Case

LIST OF SYMBOLS AND ACRONYMS

SYMBOLS

Symbols	Meaning
&	Ampersand
<	Less than
>	Greater than
“ ”	double quotation mark
‘ ’	Single quotation mark
==	Equals To
!=	Not Equals To
%	Modulus
*	Multiplication
{ }	Opening and closing braces
;	Semicolon