

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI-590018, KARNATAKA



A MINI PROJECT REPORT
ON

“FUEL MANAGEMENT SYSTEM”

Submitted in partial fulfillment of requirements for the award of 5th semester,

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE & ENGINEERING

Submitted By:
ANTONY ASHWIN ANTO
(USN: 1MJ19CS706)

Under the Guidance of
Mrs. Tessy Rose
Assistant Professor,
Department of Computer Science & Engineering



An Autonomous Institute

(Affiliated to Visvesvaraya Technological University, Belagavi)

Approved By AICTE, New Delhi,

Recognized by UGC under 2(f) & 12(B)

Accredited by NBA and NAAC)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

2021-2022



An Autonomous Institute

(Affiliated to Visvesvaraya Technological University, Belagavi)

Approved By AICTE, New Delhi,

Recognized by UGC under 2(f) & 12(B)

Accredited by NBA and NAAC)

Department of Computer Science and Engineering

Certificate

This is to certify that the mini project entitled "**Fuel Management System**" is a bonafide work carried out by **ANTONY ASHWIN ANTO (1MJ19CS706)**, a bonafide student of MVJ College of Engineering in partial fulfillment for the award of degree of Bachelor of Engineering in Computer Science & Engineering of the Visvesvaraya Technological University, Belagavi during the year 2021-22. It is certified that all the corrections/suggestions indicated for Internal Assessment have been incorporated in the Report. The mini-Project Report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said degree.

Signature of the Guide

Signature of the HOD

Signature of the Examiners

Internal

External

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success.

I wish to place on record my thanks to our Principal **Dr.P.Mahabaleshwarappa**, & Vice-Principal **Dr.M.Brindha**, MVJ College of Engineering for providing facilities..

I wish to place on record my thanks to **Professor. R. Tamilarasi**, Head of the Department, Computer Science, and Engineering, MVJ College of Engineering, Bangalore for providing encouragement and guidance.

I consider it a privilege and honor to express my sincere gratitude to my guide **Mrs. Tessy Rose** CSE Department of Computer Science & Engineering for their valuable guidance throughout the tenure of this mini-project work and whose support and encouragement made this work possible.

I wish to extend my profound sense of gratitude to my parents, friends and all my faculty members for all the sacrifices they made during my project and providing me with moral support and encouragement whenever required.

Thank you.

ABSTRACT

My project, "Fuel Management System" provides a simple interface for fuelling a vehicle, maintaining employee details, and maintenance of Fuel invoices. The manual system that is employed is laborious and inadequate. It only makes the process more difficult. This project aims to develop an arrangement to partially computerize the work performed in Petrol Stations like generating fuel consumption invoices, recording total consumption, consumption date, consumption time, consumption location, and most importantly, number plate and license number of the customer. This project focuses on providing information in a simple and intelligible manner. This project is ideal for those who want to know more about Fuel Management Systems, develop software/websites based on the same concept, or implement this project in the real world.

This project provides facilities like Invoice Generation and Automatic storage of consumption details, reducing paperwork and manual calculations in the current stations. Furthermore, building predictions and statics becomes easier with fuel consumption data. Finally, this system may function as a base for a more sophisticated arrangement in the future which may involve consumption slab-based invoices or automatic refueling using artificial intelligence.

TABLE OF CONTENT

Chapter Number	Chapter Name	Page No
Chapter 1	Introduction	1
Chapter 2	System Requirement	2
	2.1 Hardware Requirements	2
	2.2 Software Requirements	2
Chapter 3	Problem Description	3
Chapter 4	System Design	7
	4.1 ER Diagram	7
	4.2 Schema Diagram	8
	4.3 Normal Forms	9
Chapter 5	Implementation	10
	5.1 Source code	10
	5.2 SQL Trigger	30
Chapter 6	Screenshots	31
Chapter 7	Conclusion and Future Scope	37
References		38

LIST OF FIGURES

Fig Number	Fig Name	Chapter No	Page No
4.1	ER Diagram	4	7
4.2	Schema Diagram	4	8
6.1	Register section	6	34
6.2	Employee section	6	35
6.3	Supplier section	6	36
6.4	Invoice section	6	37
6.5	options	6	38
6.6	section	6	38
6.7	login	6	39

LIST OF TABLES

Table Number	Table Name	Chapter No	Page No
3.1	Employee	3	3
3.2	Gas station	3	4
3.3	invoice	3	5
3.4	state	3	5
3.5	supplier	3	5
3.6	vehicle details	3	6

Chapter 1

INTRODUCTION

The project, “Fuel Management System” provides a simple interface for fuelling a vehicle, maintaining employee details, and maintenance of Fuel invoices. The manual system that is employed is laborious and inadequate. It only makes the process more difficult. This project aims to develop an arrangement to partially computerize the work performed in Petrol Stations like generating fuel consumption invoices, recording total consumption, consumption date, consumption time, consumption location, and most importantly, number plate and license number of the customer.

This project provides facilities like Invoice Generation and Automatic storage of consumption details, reducing paperwork and manual calculations in the current stations. Furthermore, building predictions and statics becomes easier with fuel consumption data. Finally, this system may function as a base for a more sophisticated arrangement in the future which may involve consumption slab-based invoices or automatic refuelling using artificial intelligence.

Development of front-end and back-end is through .NET (C#) with a client-side Microsoft SQL Database to assess SQL queries.

Chapter 2

SYSTEM REQUIREMENTS

2.1 Hardware Requirements

Windows

- Operating System: Windows 7 or later
- Processor: Intel Pentium 4 or later
- Memory: 2 GB minimum and recommended
- Screen resolution: 1280*1024 or larger
- Application Window Size: 1024*680 or larger
- Internet Connection: Not required

2.2 Software Requirements

- Client: Windows 7 or later
- Web Server: None
- .NET Framework: 4.7 or above
- Database: Microsoft SQL
- Language: .NET (C#)

Chapter 3

PROBLEM DESCRIPTION

Development of a database application to enable petrol station attendants to record transaction and other useful data for statistics or automation.

The database uses the following tables for maintaining the data:

1. Employee
2. Gas Station
3. Invoice
4. Price in Each State
5. Supplier
6. Vehicle Details

The description of the tables is as follows:

Table 3.1 Employee

COLUMN NAME	DATATYPE & SIZE	CONSTRAINTS	DESCRIPTION
Employee_id	Varchar(16)	PK, NOT NULL	Unique Id number
First_name	Varchar(16)	-	First Name of employee
Last_name	Varchar(16)	-	Last Name of employee
Gas_station_id	Text	PK, FK, NOT NULL	Gas Station the employee is working in
Salary	Int	-	Employee's salary
Desgination	Varchar(16)	-	If the employee is a pump attendant or a manager
Contact_number	Varchar(16)	-	Employee's contact number
Duty_start_time	Time	-	Employee's duty time
Duty_end_time	Time	-	Employee's duty end time

Table 3.2 Gas Station

COLUMN NAME	DATATYPE & SIZE	CONSTRAINTS	DESCRIPTION
gas_id	Varchar(16)	PK, NOT NULL	Unique id number of gas station
G_state	Varchar(16)	FK	Name of the state in which gas station resides
G_pin	Varchar(16)	-	Pin code of district
Supplier	Varchar(16)	FK	Supplier name
G_pass	Varchar(16)	-	Password for gas station profile
Address	Varchar(16)	-	Gas station address
Petrol_curr	Int	-	Petrol available
Diesel_curr	Int	-	Diesel available
Petrl_max	Int	-	Maximum petrol that be stored
Diesel_max	Int	-	Maximum Diesel that an be stored
Lasr_order_date	Date	-	Date when refill of tank was requested
Order_request	Bit	-	If refill was requested or not

Table 3.3 invoice

COLUMN NAME	DATATYPE & SIZE	CONSTRAINTS	DESCRIPTION
G_id	Varchar(16)	PK, FK	Unique id number of gas station
Fuel_amount	float	-	Total volume of consumption
Number_plate	Varchar(16)	-	Number plate of the vehicle which requested for refuel
Invoice_time	Time	-	Time of invoice generation
Purchase_amount	Float	-	Total cost of consumption
Invoice_date	date	-	Date when invoice was generated
Invoice_id	Int	AUTO INCREMENT, NOT NULL	Unique number to identify the invoice

Table 3.4 state

COLUMN NAME	DATATYPE & SIZE	CONSTRAINTS	DESCRIPTION
State	Varchar(16)	PK, NOT NULL	Name of the state
Petrol_price	Float	-	Petrol price in state
Diesel_price	Float	-	Diesel price in state

Table 3.5 supplier

COLUMN NAME	DATATYPE & SIZE	CONSTRAINTS	DESCRIPTION
Supplier name	Varchar(16)	PK, NOT NULL	Name of the state
Supplier petrol price	Float	-	Petrol price in state
Supplier diesel price	Float	-	Diesel price in state
Supplier email	Varchar(32)	-	Support email of the supplier

Table 3.6 vehicle_details

COLUMN NAME	DATATYPE & SIZE	CONSTRAINTS	DESCRIPTION
License number	Varchar(32)	-	License number of vehicle owner
Number plate	Varchar(16)	PK, NOT NULL	Vehicle's unique registration number
Vehicle class	Varchar(16)	-	Type of vehicle
Petrol or diesel	Varchar(2)	-	Type of fuel required for the vehicle
Total fuel consumed	REAL	-	Total fuel consumed by the vehicle

Chapter 4

SYSTEM DESIGN

Section 4.1 ER Diagram:

An Entity Relationship Diagram is a data modelling technique that graphically illustrates an information systems entity and the relationships between those entities.

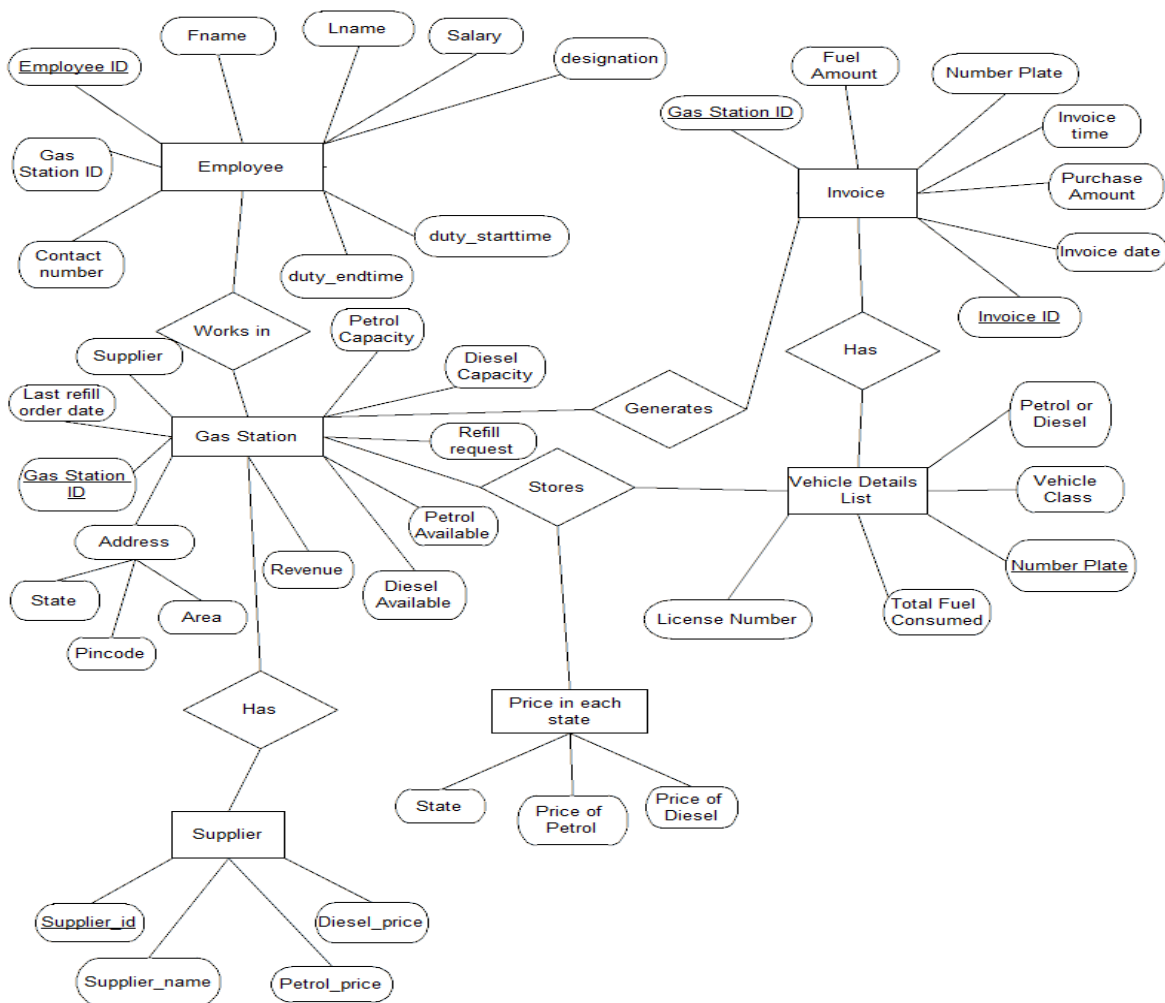


Fig 4.1

ER diagram for Fuel Management System.

Section 4.2 Schema Diagram:

A database Schema is a skeleton structure that represents the logical view of the entire database.

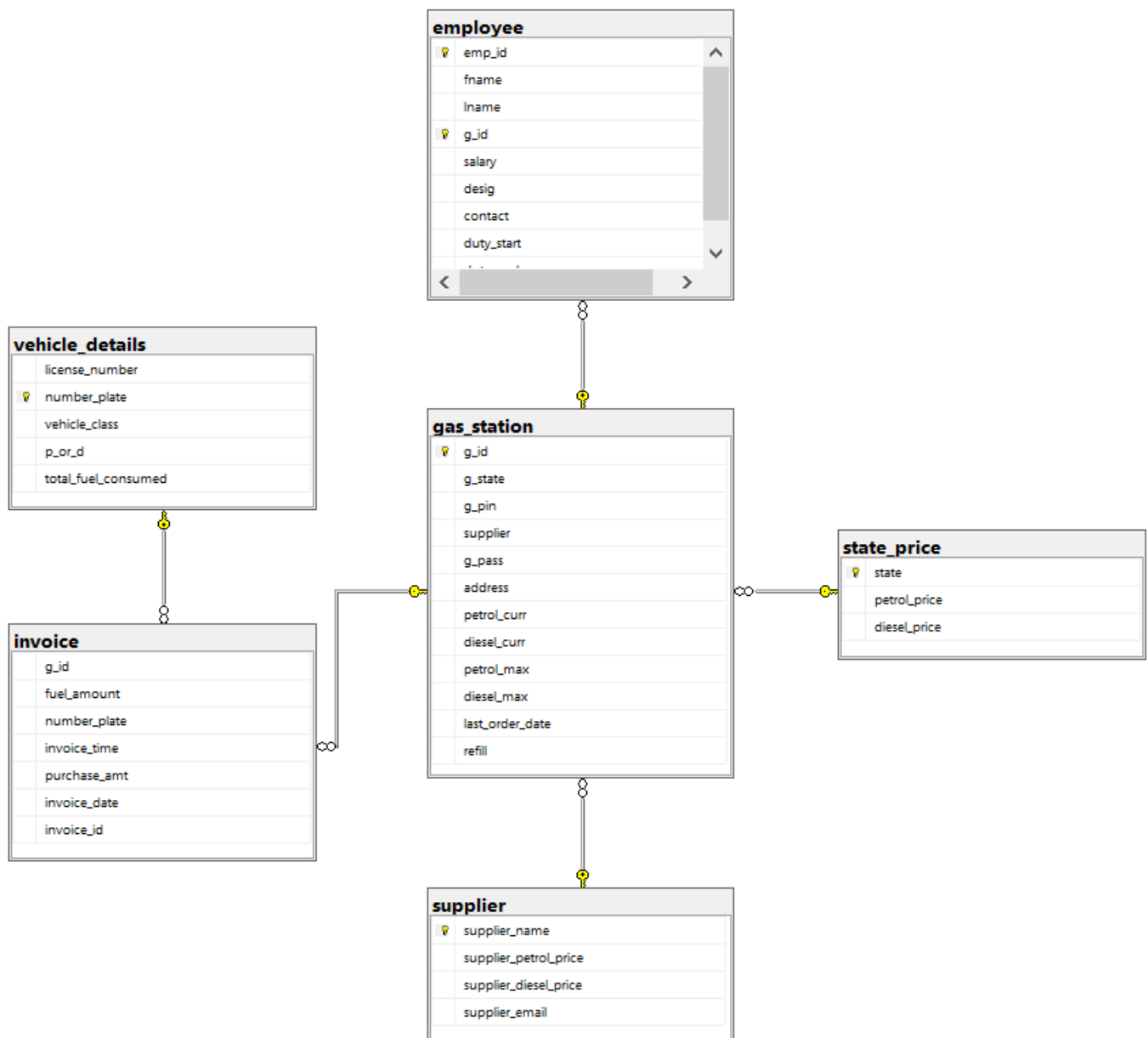


Fig 4.2

Schema diagram for Fuel Management System

4.3 Normal Forms

1. NF

Database is in first normal form, if it satisfies the following conditions

- 1.Contains only atomic values.
- 2.There are no repeating groups.

2. NF

A relation schema R is in second normal form, if every non primary attribute A is fully functionally dependant on Primary key of R

- 1.It is in 1NF
- 2.All non-key attributes are fully functionally dependant on primary key.

3. NF

1. It is in 2NF
2. There are no transitive functional dependencies

Though the above table does not satisfy 2NF, it can be normalised to 2NF and possibly, to 3NF.

Chapter 5

IMPLEMENTATION

CODE

program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace FMS
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new login_prompt());
        }
    }
}
```

loginform.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
namespace FMS
{
    public partial class login_prompt : Form
    {
        public string blank_error = "This field cannot be blank";
        public login_prompt()
        {
            InitializeComponent();
        }
        public string conn_string = @"Data Source=LAPTOP-LAPTOPNAME\SQLEXPRESS;Initial
Catalog=dbms_miniproject;Persist Security Info=True;User ID=root;Password=password123";

        SqlCommand cmd;
        private void g_login(object sender, EventArgs e)
        {
            String g_username = username_text.Text;
            String g_password = password_text.Text;
        }
    }
}
```



```

String g_state;
String g_pin;
String g_address;
double g_petrol_max;
double g_diesel_max;
double g_petrol_curr;
double g_diesel_curr;

if (g_username != "" || g_password != "")
{
    using (SqlConnection conn = new SqlConnection(conn_string))
    {
        try
        {
            String query = "Select * from gas_station where g_id = '" + g_username + "'
AND g_pass = '" + g_password + "'";
            SqlDataAdapter sda = new SqlDataAdapter(query, conn);
            DataTable dtable = new DataTable();
            sda.Fill(dtable);
            DataRow[] g_rows = dtable.Select();
            g_state = g_rows[0]["g_state"].ToString();
            g_pin = g_rows[0]["g_pin"].ToString();
            g_address = g_rows[0]["address"].ToString();
            g_petrol_max = double.Parse(g_rows[0]["petrol_max"].ToString());
            g_diesel_max = double.Parse(g_rows[0]["diesel_max"].ToString());
            g_petrol_curr = double.Parse(g_rows[0]["petrol_curr"].ToString());
            g_diesel_curr = double.Parse(g_rows[0]["diesel_curr"].ToString());

            if (dtable.Rows.Count > 0)
            {
                //MessageBox.Show();
                Menuform form2 = new Menuform(g_username, g_address, g_state, g_pin,
g_petrol_max, g_diesel_max, g_petrol_curr, g_diesel_curr);
                form2.Show();
                this.Hide();
            }
            else
            {
                MessageBox.Show("Invalid Username or Password");
            }
        }
        catch
        {
            MessageBox.Show("Invalid Username or Password");
        }
        finally
        {
            conn.Close();
        }
    }
}
else
{
    MessageBox.Show("Username or Password field is empty");
}
}

private void login_prompt_Load(object sender, EventArgs e)
{
    Size = new Size(400, 200);
    this.Location = new Point((Screen.PrimaryScreen.WorkingArea.Width - this.Width) / 2,
        (Screen.PrimaryScreen.WorkingArea.Height - this.Height) / 2);
    new_state.SelectedIndex = 0;
    new_supplier.SelectedIndex = 0;
    groupBox1.Hide();
}

private void login_exit_event(object sender, EventArgs e)

```

```
{
    System.Windows.Forms.Application.ExitThread();
}
private void create_section_controls(object sender, EventArgs e)
{
    Size = new Size(464, 553);
    this.Location = new Point((Screen.PrimaryScreen.WorkingArea.Width - this.Width) / 2,
        (Screen.PrimaryScreen.WorkingArea.Height - this.Height) / 2);
    groupBox1.Show();
}
public void clear_text()
{
    new_g_id_err.Text = "";
    new_pin_err.Text = "";
    new_address_err.Text = "";
    new_petrol_max_err.Text = "";
    new_diesel_max_err.Text = "";
    new_pass_err.Text = "";
    admin_err.Text = "";
}

private bool allright()
{
    clear_text();
    bool check = true;
    if(new_g_id.Text == "")
    {
        new_g_id_err.Text = blank_error;
        check = false;
    }
    if (new_pin.Text.Length != 6)
    {
        new_pin_err.Text = "Invalid pin code";
        check = false;
    }
    else
    {
        try
        {
            int.Parse(new_pin.Text);
        }
        catch (Exception)
        {
            new_pin_err.Text = "Invalid pin code";
        }
    }
    if (new_address.Text == "")
    {
        new_address_err.Text = blank_error;
        check = false;
    }
    if (new_petrol_max.Text == "" && new_diesel_max.Text == "")
    {
        new_petrol_max_err.Text = "Both capacity fields cannot be blank";
        check = false;
    }
    if(new_petrol_max.Text != "")
    {
        try
        {
            double.Parse(new_petrol_max.Text);
        }
        catch(Exception)
        {
            new_petrol_max_err.Text = "Not a valid input";
        }
    }
}
```

```
        check = false;
    }
}
if (new_diesel_max.Text != "")
{
    try
    {
        double.Parse(new_diesel_max.Text);
    }
    catch (Exception)
    {
        new_diesel_max_err.Text = "Not a valid input";
        check = false;
    }
}
if (new_pass.Text == "" && new_confirm_pass.Text == "")
{
    new_pass_err.Text = blank_error;
    check = false;
}
else
{
    if (new_pass.Text != new_confirm_pass.Text)
    {
        new_pass_err.Text = "Passwords not matching";
        check = false;
    }
}
using (SqlConnection conn = new SqlConnection(conn_string))
{
    try
    {
        String query = "Select g_pass from gas_station where g_id = '"+admin+"'";
        SqlDataAdapter sda = new SqlDataAdapter(query, conn);
        DataTable dtable = new DataTable();
        sda.Fill(dtable);
        DataRow[] g_rows = dtable.Select();
        string admin_pass = g_rows[0]["g_pass"].ToString();
        if (new_admin_pass.Text != admin_pass)
        {
            admin_err.Text = "Incorrect admin password";
            check = false;
        }
    }
    catch (Exception)
    {
        admin_err.Text = "Incorrect admin password";
        check = false;
    }
}
return check;
}
private void create_new_user(object sender, EventArgs e)
{
    double new_petrol_double;
    double new_diesel_double;
    clear_text();
    bool check = allright();
    if (!allright())
    {
        return;
    }
    if (new_petrol_max.Text != "")
        new_petrol_double = double.Parse(new_petrol_max.Text);
    else
```

```

        new_petrol_double = 0;
        if (new_diesel_max.Text != "")
            new_diesel_double = double.Parse(new_diesel_max.Text);
        else
            new_diesel_double = 0;
        using (SqlConnection conn = new SqlConnection(conn_string))
        {
            try
            {
                conn.Open();
                String query = "insert into gas_station values('" + new_g_id.Text + "','" +
new_state.Text + "','" + new_pin.Text + "','" + new_supplier.Text + "','" + new_pass.Text + "','" +
new_address.Text + "','0,0,'" + new_petrol_double + "','" + new_diesel_double + "','NULL,0)";
                cmd = new SqlCommand(query, conn);
                cmd.ExecuteNonQuery();
                conn.Close();
                MessageBox.Show("User created");
                clear_text();
            }
            catch (Exception)
            {
                new_g_id_err.Text = "Username already taken";
            }
        }
    }
}
}
}

```

menuform.cs

```

using System;
namespace FMS
{
    public partial class Menuform : Form
    {
        public Menuform(string g_id, string g_address, string g_state, string g_pin, double
g_petrol_max, double g_diesel_max, double g_petrol_curr, double g_diesel_curr)
        {
            InitializeComponent();
            this.G_id = g_id;
        }
        public string G_id { get; set; }
        public string G_address { get; set; }
        public string G_pin { get; set; }
        public string G_state { get; set; }
        public double G_petrol_max { get; set; }
        public double G_diesel_max { get; set; }
        public double G_petrol_curr { get; set; }
        public double G_diesel_curr { get; set; }
        private void Menuform_Load(object sender, EventArgs e)
        {
            g_id.Text = G_id.ToUpper();
            address_label.Text = G_address;
            state_and_pin_label.Text = G_state + " - " + G_pin;
            home_page1.home_g_id = G_id;
            employee_page1.g_id = G_id;
            supplier_page1.g_id = G_id;
            options_page1.g_curr_petrol = G_petrol_curr;
            options_page1.g_curr_diesel = G_diesel_curr;
            invoice_page1.g_id = G_id;
            options_page1.g_id = G_id;
            timer1.Start();
        }
        private void timer1_Tick(object sender, EventArgs e)
        {
            time_label.Text = DateTime.Now.ToString("h:mm:ss tt");
            day_label.Text = DateTime.Today.DayOfWeek.ToString();
        }
    }
}

```

```

        date_label1.Text = DateTime.Now.ToString("dd/MM/yyyy");
    }
    private void logout_event(object sender, EventArgs e)
    {
        this.Close();
        login_prompt form1 = new login_prompt();
        form1.Show();
    }
    private void exit_event(object sender, EventArgs e)
    {
        System.Windows.Forms.Application.ExitThread();
    }
    private void register_page1_Load(object sender, EventArgs e)
    {
    }
    private void home_event(object sender, EventArgs e)
    {
        home_page1.BringToFront();
    }
    private void register_event(object sender, EventArgs e)
    {
        register_page1.BringToFront();
    }
    private void employee__event(object sender, EventArgs e)
    {
        employee_page1.BringToFront();
    }
    private void home_page1_Load(object sender, EventArgs e)
    {
    }
    private void employee_page1_Load(object sender, EventArgs e)
    {
    }
    private void supplier_event(object sender, EventArgs e)
    {
        supplier_page1.BringToFront();
    }
    private void invoice_history_event(object sender, EventArgs e)
    {
        invoice_page1.BringToFront();
    }
    private void options_event(object sender, EventArgs e)
    {
        options_page1.BringToFront();
    }
}
}

```

home_page.cs

```

using System;
using System.Data.SqlClient;
namespace FMS
{
    public partial class home_page : UserControl
    {
        static double petrol_price;
        static double diesel_price;
        public double home_g_pmax { get; set; }
        public double home_g_dmax { get; set; }
        public double home_g_pcurr { get; set; }
        public double home_g_dcurr { get; set; }
        public string home_g_id { get; set; }
        // check if number plate in database or not

        private bool check_in_vehicle_list(string number_plate)
        {

```

```
String query = "Select * from vehicle_details where number_plate = '" + number_plate +
"";

SqlDataAdapter sda = new SqlDataAdapter(query, conn);
DataTable dtbl = new DataTable();
sda.Fill(dtbl);

if (dtbl.Rows.Count > 0)
{
    return true;
}
else
{
    return false;
}
}
// check function ends here
// get cost in partiular state
public void get_current_prices()
{
    String query = "select petrol_price, diesel_price from state_price, gas_station where
state = g_state and g_id = '"+home_g_id+"'";
    SqlDataAdapter sda = new SqlDataAdapter(query, conn);
    DataTable dtbl = new DataTable();
    sda.Fill(dtbl);
    DataRow[] g_rows = dtbl.Select();
    petrol_price = double.Parse(g_rows[0]["petrol_price"].ToString());
    diesel_price = double.Parse(g_rows[0]["diesel_price"].ToString());
}
// function ends here

// find vehicle type - petrol or diesel
private string vehicle_type(string number_plate)
{
    string v_type;
    String query = "select p_or_d from vehicle_details where number_plate =
 '"+number_plate+"'";
    SqlDataAdapter sda = new SqlDataAdapter(query, conn);
    DataTable dtbl = new DataTable();
    sda.Fill(dtbl);
    DataRow[] type_rows = dtbl.Select();
    v_type = type_rows[0]["p_or_d"].ToString();
    return v_type;
}
// function ends
//cost -> litres
private double compute_volume(double cost, string v_type)
{
    if (v_type == "P")
        return cost / petrol_price;
    else
        return cost / diesel_price;
}
// function ends here
//litres -> cost
private double compute_cost(double litres, string v_type)
{
    if (v_type == "P")
        return litres * petrol_price;
    else
        return litres * diesel_price;
}
//function ends here

//function to check if fuel available;
private bool transfer_from_bunk_to_vehicle(double litres, string v_type)
```

```
{
    String query = "select petrol_curr, diesel_curr from gas_station where g_id =
    '"+home_g_id+"'";
    SqlDataAdapter sda = new SqlDataAdapter(query, conn);
    DataTable dttable = new DataTable();
    sda.Fill(dttable);
    DataRow[] avail = dttable.Select();
    double petrol_curr = double.Parse(avail[0]["petrol_curr"].ToString());
    double diesel_curr = double.Parse(avail[0]["diesel_curr"].ToString());
    if (litres > petrol_curr && v_type == "P")
        return false;
    else if (litres > diesel_curr && v_type == "D")
        return false;
    else
        return true;
}
//
public home_page()
{
    InitializeComponent();

}
SqlConnection conn = new SqlConnection(@"Data Source=LAPTOP-LAPTOPNAME\SQLEXPRESS;Initial
Catalog=dbms_miniproject;Persist Security Info=True;User ID=root;Password=password123");
SqlCommand cmd;
private void UserControl1_Load(object sender, EventArgs e)
{
    c_or_v_input.SelectedIndex = 1;
}
private void textBox1_TextChanged(object sender, EventArgs e)
{
}
private void refill_event(object sender, EventArgs e)
{
    get_current_prices();
    double cost, volume;
    string v_type;
    string curr_time = DateTime.Now.ToString("HH:mm:ss");
    string curr_date = DateTime.Now.ToString("yyyy-MM-dd");
    string register_number = register_number_input.Text;
    double value;
    try
    {
        value = double.Parse(amount_input.Text);
    }
    catch (FormatException)
    {
        MessageBox.Show("Invalid input");
        return;
    }

    string c_or_v = c_or_v_input.Text;

    if (check_in_vehicle_list(register_number))
    {
        v_type = vehicle_type(register_number);
        if (c_or_v == "L")
        {
            volume = value;
            cost = compute_cost(value, v_type);
        }
        else
        {

```

```

        cost = value;
        volume = compute_volume(value, v_type);
    }
    if (transfer_from_bunk_to_vehicle(volume, v_type) == false)
    {
        MessageBox.Show("Insufficient fuel in storage");
        return;
    }
    try
    {
        conn.Open();
        String query = "insert into invoice values('" + home_g_id + "','" + volume +
        "','" + register_number + "','" + curr_time + "','" + cost + "','" + curr_date + "')";
        cmd = new SqlCommand(query, conn);
        cmd.ExecuteNonQuery();
        MessageBox.Show("Invoice Generated");
        conn.Close();
    }
    catch (Exception)
    {
        MessageBox.Show("No match found, Add vehicle from Register tab");
    }
}

else
{
    MessageBox.Show("No match found, Add vehicle from Register tab");
}
button2.PerformClick();
}
private void button2_Click(object sender, EventArgs e)
{
    String query = "select petrol_curr, diesel_curr, petrol_price, diesel_price from
state_price, gas_station where g_id = '" + home_g_id + "' and state_price.state = g_state";
    SqlDataAdapter sda = new SqlDataAdapter(query, conn);
    DataTable dtable = new DataTable();
    sda.Fill(dtable);
    DataRow[] g_rows = dtable.Select();
    prate.Text = g_rows[0]["petrol_price"].ToString();
    drate.Text = g_rows[0]["diesel_price"].ToString();
    pavail.Text = double.Parse(g_rows[0]["petrol_curr"].ToString()).ToString("F2");
    davail.Text = double.Parse(g_rows[0]["diesel_curr"].ToString()).ToString("F2");
}
}
}

```

registerpage.cs

```

using System;
using System.Data.SqlClient;

namespace FMS
{
    public partial class register_page : UserControl
    {
        public register_page()
        {
            InitializeComponent();
        }
        public string conn_string = @"Data Source=LAPTOP-LAPTOPNAME\SQLEXPRESS;Initial
Catalog=dbms_miniproject;Persist Security Info=True;User ID=root;Password=password123";
        SqlCommand cmd;

        private void register_page_Load(object sender, EventArgs e)
        {
            v_class.SelectedIndex = 1;
            f_type.SelectedIndex = 1;
        }
    }
}

```



```

    }
    private void register_vehicle(object sender, EventArgs e)
    {
        string l_num = l_number.Text;
        string r_num = r_number.Text;
        string vclass = v_class.Text;
        string ftype = f_type.Text;
        if (!all_fine(l_num, r_num))
            return;
        using (SqlConnection conn = new SqlConnection(conn_string))
        {
            //never gonna give you up never gonna let you down never gonna.....
            try
            {
                conn.Open();
                String query = "insert into vehicle_details values('" + l_num + "','" + r_num +
                "','" + vclass + "','" + ftype + "',0)";
                cmd = new SqlCommand(query, conn);
                cmd.ExecuteNonQuery();
                conn.Close();
                MessageBox.Show("Registration Successful");
            }
            catch (Exception)
            {
                MessageBox.Show("Vehicle already exists");
            }
        }
    }

    private bool all_fine(string l_num, string r_num)
    {
        bool all_good = true;
        if (l_num.Length != 16)
        {
            l_error.Text = "Invalid License Number";
            all_good = false;
        }
        else l_error.Text = "";
        if (r_num.Length != 10)
        {
            r_error.Text = "Invalid Registration Number";
            all_good = false;
        }
        else r_error.Text = "";
        return all_good;
    }
}

```

supplier.cs

```

using System;
using System.Data.SqlClient;
namespace FMS
{
    public partial class supplier_page : UserControl
    {
        public string conn = @"Data Source=LAPTOP-LAPTOPNAME\SQLEXPRESS;Initial
        Catalog=dbms_miniproject;Persist Security Info=True;User ID=root;Password=password123";
        public string g_id { get; set; }

        public supplier_page()
        {
            InitializeComponent();
        }
    }
}

```

```

private void supplier_page_Load(object sender, EventArgs e)
{
}
private void get_supplier_details(object sender, EventArgs e)
{
    groupBox3.Hide();
    groupBox2.Show();
    groupBox2.BringToFront();
    String query = "select * from supplier join gas_station on g_id = '" + g_id + "' and
gas_station.supplier = supplier.supplier_name;";
    SqlDataAdapter sda = new SqlDataAdapter(query, conn);
    DataTable dtable = new DataTable();
    sda.Fill(dtable);
    DataRow[] g_rows = dtable.Select();
    supp_name.Text = g_rows[0]["supplier_name"].ToString();
    prate.Text = g_rows[0]["supplier_petrol_price"].ToString();
    drate.Text = g_rows[0]["supplier_diesel_price"].ToString();
    supp_email.Text = g_rows[0]["supplier_email"].ToString();
}
private void refill_button(object sender, EventArgs e)
{
    groupBox2.Hide();
    groupBox3.Show();
    groupBox3.BringToFront();
    groupBox2.BringToFront();
    String query = "select * from supplier join gas_station on g_id = '" + g_id + "' and
gas_station.supplier = supplier.supplier_name;";
    SqlDataAdapter sda = new SqlDataAdapter(query, conn);
    DataTable dtable = new DataTable();
    sda.Fill(dtable);
    DataRow[] g_rows = dtable.Select();
    int refill_status = int.Parse(g_rows[0]["refill"].ToString());
    string last_order_date = g_rows[0]["last_order_date"].ToString();
    DialogResult dialogResult = MessageBox.Show("Proceeding with yes will send an automated
Silo Refill Request", "Warning !", MessageBoxButtons.YesNo);
    if (dialogResult == DialogResult.Yes)
    {
        if (refill_status == 1)
        {
            status_info.Text = "You have already requested for a refill on '" +
last_order_date + "'";
        }
        else
        {
            status_info.Text = "Refill request will be attended to in 2-3 working days\n" +
            "Supplier Name : '" + g_rows[0]["supplier"].ToString() + "'\n" +
            "Request Status : Yet to Deliver";
            SqlConnection conn1 = new SqlConnection(conn);
            SqlCommand cmd;
            conn1.Open();
            String query1 = "update gas_station set last_order_date = '" +
DateTime.Now.ToString("yyyy-MM-dd") + "', refill = 1, petrol_curr = petrol_max, diesel_curr =
diesel_max where g_id = '" + g_id + "'";
            cmd = new SqlCommand(query1, conn1);
            cmd.ExecuteNonQuery();
            conn1.Close();
            MessageBox.Show("Refill request sent.");
        }
    }
    else
    {
        status_info.Text = " ";
    }
}
private void clear_refill_request(object sender, EventArgs e)

```

```

    {
        SqlConnection conn1 = new SqlConnection(conn);
        SqlCommand cmd;
        conn1.Open();
        String query1 = "update gas_station set refill = 0 where g_id = '" + g_id + "'";
        cmd = new SqlCommand(query1, conn1);
        cmd.ExecuteNonQuery();
        conn1.Close();
        MessageBox.Show("Your refill request has been cleared.");
        status_info.Text = " ";
    }
}

```

invoice.cs

```

using System;
using System.Data.SqlClient;
namespace FMS
{
    public partial class supplier_page : UserControl
    {
        public string conn = @"Data Source=LAPTOP-LAPTOPNAME\SQLEXPRESS;Initial
Catalog=dbms_miniproject;Persist Security Info=True;User ID=root;Password=password123";
        public string g_id { get; set; }

        public supplier_page()
        {
            InitializeComponent();
        }

        private void supplier_page_Load(object sender, EventArgs e)
        {
        }

        private void get_supplier_details(object sender, EventArgs e)
        {
            groupBox3.Hide();
            groupBox2.Show();
            groupBox2.BringToFront();
            String query = "select * from supplier join gas_station on g_id = '"+g_id+"' and
gas_station.supplier = supplier.supplier_name";
            SqlDataAdapter sda = new SqlDataAdapter(query, conn);
            DataTable dtable = new DataTable();
            sda.Fill(dtable);
            DataRow[] g_rows = dtable.Select();
            supp_name.Text = g_rows[0]["supplier_name"].ToString();
            prate.Text = g_rows[0]["supplier_petrol_price"].ToString();
            drate.Text = g_rows[0]["supplier_diesel_price"].ToString();
            supp_email.Text = g_rows[0]["supplier_email"].ToString();
        }

        private void refill_button(object sender, EventArgs e)
        {
            groupBox2.Hide();
            groupBox3.Show();
            groupBox3.BringToFront();
            groupBox2.BringToFront();
            String query = "select * from supplier join gas_station on g_id = '" + g_id + "' and
gas_station.supplier = supplier.supplier_name";
            SqlDataAdapter sda = new SqlDataAdapter(query, conn);
            DataTable dtable = new DataTable();
            sda.Fill(dtable);
            DataRow[] g_rows = dtable.Select();
        }
    }
}

```

```

int refill_status = int.Parse(g_rows[0]["refill"].ToString());
string last_order_date = g_rows[0]["last_order_date"].ToString();
DialogResult dialogResult = MessageBox.Show("Proceeding with yes will send an automated
Silo Refill Request", "Warning !", MessageBoxButtons.YesNo);
if (dialogResult == DialogResult.Yes)
{
    if (refill_status == 1)
    {
        status_info.Text = "You have already requested for a refill on '" +
last_order_date + "'";
    }
    else
    {
        status_info.Text = "Refill request will be attended to in 2-3 working days\n" +
        "Supplier Name : '" + g_rows[0]["supplier"].ToString() + "'\n" +
        "Request Status : Yet to Deliver";
        SqlConnection conn1 = new SqlConnection(conn);
        SqlCommand cmd;
        conn1.Open();
        String query1 = "update gas_station set last_order_date = '" +
DateTime.Now.ToString("yyyy-MM-dd") + "', refill = 1, petrol_curr = petrol_max, diesel_curr =
diesel_max where g_id = '" + g_id + "'";
        cmd = new SqlCommand(query1, conn1);
        cmd.ExecuteNonQuery();
        conn1.Close();
        MessageBox.Show("Refill request sent.");
    }
}
else
{
    status_info.Text = " ";
}
}
private void clear_refill_request(object sender, EventArgs e)
{
    SqlConnection conn1 = new SqlConnection(conn);
    SqlCommand cmd;
    conn1.Open();
    String query1 = "update gas_station set refill = 0 where g_id = '" + g_id + "'";
    cmd = new SqlCommand(query1, conn1);
    cmd.ExecuteNonQuery();
    conn1.Close();
    MessageBox.Show("Your refill request has been cleared.");
    status_info.Text = " ";
}
}
}
}

```

options.cs

```

using System;
using System.Data.SqlClient;
namespace FMS
{
    public partial class options_page : UserControl
    {
        public string conn_string = @"Data Source=LAPTOP-LAPTOPNAME\SQLEXPRESS;Initial
Catalog=dbms_miniproject;Persist Security Info=True;User ID=root;Password=password123";

        SqlCommand cmd;
        public void clear_text()
        {
            new_pass_err.Text = "";
            confirm_new_pass_err.Text = "";
            curr_pass_err.Text = "";
        }
    }
}

```

```
}
public bool all_good()
{
    bool check = true;
    if(new_pass.Text.Length < 5)
    {
        new_pass_err.Text = "Password length must be greater than 5 characters";
        check = false;
    }
    if(new_pass.Text != confirm_new_pass.Text)
    {
        confirm_new_pass_err.Text = "Passwords not matching";
        check = false;
    }
    if(curr_pass.Text == "")
    {
        check = false;
        curr_pass_err.Text = "Current password is empty";
    }

    return check;
}

public string g_id { get; set; }
public double g_curr_petrol { get; set; }
public double g_curr_diesel { get; set; }
public options_page()
{
    InitializeComponent();
}
private void change_pass(object sender, EventArgs e)
{
    clear_text();
    if (all_good())
    {
        using (SqlConnection conn = new SqlConnection(conn_string))
        {
            String query = "Select * from gas_station where g_id = '" + g_id + "'";
            SqlDataAdapter sda = new SqlDataAdapter(query, conn);
            DataTable dtable = new DataTable();
            sda.Fill(dtable);
            DataRow[] g_rows = dtable.Select();
            if (curr_pass.Text == g_rows[0]["g_pass"].ToString())
            {
                //update
                conn.Open();
                String query = "update gas_station set g_pass = '" + new_pass.Text + "' where
g_id = '" + g_id + "'";
                cmd = new SqlCommand(query, conn);
                cmd.ExecuteNonQuery();
                conn.Close();
                pass_change.Text = "Password changed";
            }
            else
            {
                curr_pass_err.Text = "Incorrect Password";
            }
        }
    }
    else
    {
        return;
    }
}
private void options_page_Load(object sender, EventArgs e)
```

```
{
    update_petrol_err.Text = "";
    update_diesel_err.Text = "";
}
private void update_pmax(object sender, EventArgs e)
{
    double value;
    if (update_petrol.Text == "")
    {
        update_petrol_err.ForeColor = Color.Red;
        update_petrol_err.Text = "Invalid Input";
    }
    else
    {
        using (SqlConnection conn = new SqlConnection(conn_string))
        {
            try
            {
                value = double.Parse(update_petrol.Text);

                conn.Open();
                String query = "update gas_station set petrol_max = '" + value + "' where
g_id = '" + g_id + "'";
                cmd = new SqlCommand(query, conn);
                cmd.ExecuteNonQuery();
                conn.Close();
                update_petrol_err.ForeColor = Color.Green;
                update_petrol_err.Text = "Capacity updated";
            }
            catch (Exception)
            {
                update_petrol_err.ForeColor = Color.Red;
                update_petrol_err.Text = "Invalid Input";
            }
        }
    }
}
private void update_dmax(object sender, EventArgs e)
{
    double value;
    if (update_diesel.Text == "")
    {
        update_diesel_err.ForeColor = Color.Red;
        update_diesel_err.Text = "Invalid Input";
    }
    else
    {
        using (SqlConnection conn = new SqlConnection(conn_string))
        {
            try
            {
                value = double.Parse(update_diesel.Text);

                conn.Open();
                String query = "update gas_station set diesel_max = '" + value + "' where
g_id = '" + g_id + "'";
                cmd = new SqlCommand(query, conn);
                cmd.ExecuteNonQuery();
                conn.Close();
                update_diesel_err.ForeColor = Color.Green;
                update_diesel_err.Text = "Capacity updated";
            }
            catch (Exception)
            {
                update_diesel_err.ForeColor = Color.Red;
                update_diesel_err.Text = "Invalid Input";
            }
        }
    }
}
```

```

    }
    }
}
private void groupBox1_Enter(object sender, EventArgs e)
{
}
}
}

```

employee.cs

```

using System;
using System.Data.SqlClient;
namespace FMS
{
    public partial class employee_page : UserControl
    {
        public string g_id { get; set; }
        string time = DateTime.Now.ToString("HH:mm:ss");

        public employee_page()
        {
            InitializeComponent();
        }

        double salary;
        string con_string = @"Data Source=LAPTOP-LAPTOPNAME\SQLEXPRESS;Initial
Catalog=dbms_miniproject;Persist Security Info=True;User ID=root;Password=password123";
        SqlConnection conn = new SqlConnection(@"Data Source=LAPTOP-LAPTOPNAME\SQLEXPRESS;Initial
Catalog=dbms_miniproject;Persist Security Info=True;User ID=root;Password=password123");
        SqlCommand cmd;
        private void employee_page_Load(object sender, EventArgs e)
        {
            delete_raise_group.Hide();
            add_group.Hide();
            designation.SelectedIndex=1;
        }
        private void refresh_data(object sender, EventArgs e)
        {
            clear_textfield();
            get_employee_list();
            employee_incharge();
        }

        public void employee_incharge()
        {
            try
            {
                String queryry = "select * from employee where g_id = '" + g_id + "' and desig =
'pump_attendant' and '" + time + "' between duty_start and duty_end;";
                SqlDataAdapter sda = new SqlDataAdapter(queryry, con_string);
                DataTable dtable = new DataTable();
                sda.Fill(dtable);
                DataRow[] g_rows = dtable.Select();
                curr_emp.Text = "[" + g_rows[0]["emp_id"].ToString() + "]" + " " +
g_rows[0]["fname"].ToString() + " " + g_rows[0]["lname"].ToString();
            }
            catch (Exception)
            {
                curr_emp.Text = "No employee assigned for now";
            }
        }
    }
}

```

```
public void get_employee_list()
{
    try
    {
        SqlConnection con = new SqlConnection(con_string);
        SqlCommand cmd = new SqlCommand("select * from employee where g_id = '" + g_id +
";", con);
        DataTable dt = new DataTable();
        con.Open();
        SqlDataReader sdr = cmd.ExecuteReader();
        dt.Load(sdr);
        con.Close();
        dataGridView1.DataSource = dt;
    }
    catch (Exception)
    {
        MessageBox.Show("An error has occurred in retrieving data ");
    }
}

private void add_employee(object sender, EventArgs e)
{
    delete_raise_group.Hide();
    delete_raise_group.SendToBack();
    add_group.Text = "Add Employee";
    add_group.Show();
    add_group.BringToFront();
    clear_textfield();
    confirm_and_update.Hide();
    confirm_and_add.Show();
    confirm_and_add.BringToFront();
    update2.Hide();
}

public void clear_textfield()
{
    e_id.Clear();
    fname.Clear();
    lname.Clear();
    cnum.Clear();
    dstart.Clear();
    dend.Clear();
    sal.Clear();

    e_id_err.Text = " ";
    fname_err.Text = " ";
    lname_err.Text = " ";
    cnum_err.Text = " ";
    dstart_err.Text = " ";
    dend_err.Text = " ";
    sal_err.Text = " ";
}

public bool all_good()
{
    bool all_right = true;
    String query = "Select * from employee where g_id = '" + g_id + "' AND emp_id = '" +
e_id.Text + "'";
    SqlDataAdapter sda = new SqlDataAdapter(query, con_string);
    DataTable dtable = new DataTable();
    sda.Fill(dtable);
    if (dtable.Rows.Count > 0 || e_id.Text == "")
    {
        e_id_err.Text = "X";
        all_right = false;
    }
}
```



```
}
else
{
    e_id_err.Text = "";
}
try
{
    salary = double.Parse(sal.Text);
}
catch (Exception)
{
    sal_err.Text = "X";
    all_right = false;
}
if(fname.Text == "" && lname.Text == "")
{
    fname_err.Text = "X";
    lname_err.Text = "X";
    all_right = false;
}
else
{
    fname_err.Text = "";
    lname_err.Text = "";
}
query = "Select * from employee where contact = '" + cnum.Text + "'";
SqlDataAdapter sda1 = new SqlDataAdapter(query, con_string);
DataTable dtable1 = new DataTable();
sda.Fill(dtable1);
if (dtable1.Rows.Count > 0 || cnum.Text == "")
{
    cnum_err.Text = "X";
    all_right = false;
}
else
{
    cnum_err.Text = "";
}
return all_right;
}
private void add_employee_group(object sender, EventArgs e)
{
    if (!all_good())
        return;
    using (SqlConnection conn1 = new SqlConnection(con_string)) {
        try {
            conn1.Open();
            String query = "insert into employee values('" + e_id.Text + "','" +
fname.Text + "','" + lname.Text + "','" + g_id + "','" + salary + "','" + designation.Text + "','" +
cnum.Text + "','" + dstart.Text + "','" + dend.Text + "')";
            cmd = new SqlCommand(query, conn1);
            cmd.ExecuteNonQuery();
            conn1.Close();
            MessageBox.Show("Data successfully inserted");
            button1.PerformClick();
        }
        catch (Exception)
        {
            MessageBox.Show("Invalid input");
        }
    }
}
private void update_employee(object sender, EventArgs e)
{
}
```

```

        delete_raise_group.Hide();
        delete_raise_group.SendToBack();
        add_group.Show();
        add_group.BringToFront();
        clear_textfield();
        add_group.Text = "Update Employee";
        update2.Hide();
        confirm_and_add.Hide();
        confirm_and_update.Show();
        confirm_and_update.BringToFront();
    }
    private void update_employee_group(object sender, EventArgs e)
    {
        String query = "Select * from employee where g_id = '" + g_id + "' AND emp_id = '"
+e_id.Text+ "'";
        SqlDataAdapter sda = new SqlDataAdapter(query, con_string);
        DataTable dtable = new DataTable();
        sda.Fill(dtable);
        if (e_id.Text == "" || dtable.Rows.Count == 0)
        {
            clear_textfield();
            e_id_err.Text = "X";
            update2.Hide();
            return;
        }
        e_id_err.Text = "";
        String query2 = "Select * from employee where g_id = '" + g_id + "' AND e_id = '" +
e_id.Text + "'";
        SqlDataAdapter sda2 = new SqlDataAdapter(query2, conn);
        DataTable dtable2 = new DataTable();
        sda.Fill(dtable2);
        DataRow[] g_rows = dtable.Select();
        fname.Text = g_rows[0]["fname"].ToString();
        lname.Text = g_rows[0]["lname"].ToString();
        designation.Text = g_rows[0]["desig"].ToString();
        cnum.Text = g_rows[0]["contact"].ToString();
        dstart.Text = g_rows[0]["duty_start"].ToString();
        dend.Text = g_rows[0]["duty_end"].ToString();
        sal.Text = g_rows[0]["salary"].ToString();
        update2.Show();
    }
    private void update_for_real(object sender, EventArgs e)
    {
        using (SqlConnection conn1 = new SqlConnection(con_string))
        {
            try
            {
                double salary = double.Parse(sal.Text);
                conn1.Open();
                String query = "update employee set fname = '" + fname.Text + "', lname = '" +
lname.Text + "', salary = '" + salary + "', desig = '" + designation.Text + "', contact = '" +
cnum.Text + "', duty_start = '" + dstart.Text + "', duty_end = '" + dend.Text + "' where g_id = '" +
g_id + "' and emp_id = '" + e_id.Text + "'";
                cmd = new SqlCommand(query, conn1);
                cmd.ExecuteNonQuery();
                conn1.Close();
                MessageBox.Show("Employee details updated succesfully");
                button1.PerformClick();
            }
            catch (Exception)
            {
                MessageBox.Show("Invalid input");
            }
        }
    }
}

```

```
private void delete_employee(object sender, EventArgs e)
{
    add_group.SendToBack();
    add_group.Hide();
    delete_raise_group.Show();
    delete_raise_group.BringToFront();
}
private void delete_employee_group(object sender, EventArgs e)
{
    using (SqlConnection conn1 = new SqlConnection(con_string))
    {
        try
        {
            conn1.Open();
            String query = "delete from employee where g_id = '" + g_id + "' and emp_id = '"
+ delete_emp.Text + "'";
            cmd = new SqlCommand(query, conn1);
            cmd.ExecuteNonQuery();
            conn1.Close();
            MessageBox.Show("Employee details deleted succesfully");
            button1.PerformClick();
        }
        catch (Exception)
        {
            MessageBox.Show("Invalid input");
        }
    }
}
private void update_salary(object sender, EventArgs e)
{
}
}
```

SQL Triggers

```
begin
    declare @fuel_amount float, @petrol_curr float, @g_id varchar(16), @ig_id int, @diesel_curr float,
    @v_type varchar(2), @number_plate varchar(16);
    set @fuel_amount = (select fuel_amount from inserted);
    set @petrol_curr = (select petrol_curr from gas_station, inserted where gas_station.g_id = inserted.g_id);
    set @diesel_curr = (select diesel_curr from gas_station, inserted where gas_station.g_id = inserted.g_id);
    set @g_id = (select g_id from inserted);
    set @ig_id = (select invoice.invoice_id from invoice, inserted where invoice.invoice_id =
    inserted.invoice_id);
    set @v_type = (select p_or_d from vehicle_details, inserted where vehicle_details.number_plate =
    inserted.number_plate);
    set @number_plate = (select vehicle_details.number_plate from vehicle_details, inserted where
    vehicle_details.number_plate = inserted.number_plate);
        if not exists(select * from vehicle_details where vehicle_details.number_plate = @number_plate)
            begin
                print 'No such vehicle number exists'
            end
        else
            begin
                if(@v_type = 'P')
                    begin
                        if(@petrol_curr - @fuel_amount < 0)
                            begin
                                delete from invoice where invoice.invoice_id = @ig_id;
                            end
                        else
                            begin
                                update gas_station set petrol_curr = @petrol_curr - @fuel_amount where gas_station.g_id = @g_id;
                                update vehicle_details set total_fuel_consumed = total_fuel_consumed + @fuel_amount where
                                vehicle_details.number_plate = @number_plate;
                            end
                        end
                    end
                else
                    begin
                        if(@diesel_curr - @fuel_amount < 0)
                            begin
                                delete from invoice where invoice.invoice_id = @ig_id;
                            end
                        else
                            begin
                                update gas_station set diesel_curr = @diesel_curr - @fuel_amount where gas_station.g_id = @g_id;
                                update vehicle_details set total_fuel_consumed = total_fuel_consumed +
                                @fuel_amount where vehicle_details.number_plate = @number_plate;
                            end
                        end
                    end
                end end
            end
go
```

Chapter 6

SCREENSHOTS

Fuel Management System

Gas Station ID : G3
Ramnagar Circle
Maharashtra - 400001

8:57:44 PM
09-02-2022
Wednesday

License Number

Registration Number

Vehicle Class LMV


Fuel Type D

Register

Home
Register Vehicle
Employees
Supplier
Invoice History
Options

Log out Exit

Fig:6.1 Register section



Home

Register Vehicle

Employees

Supplier

Invoice History

Options

Log out

Exit

Gas Station ID : G3
Ramnagar Circle
Maharashtra - 400001

8:58:00 PM
09-02-2022
Wednesday

Retrieve/Refresh

Add Employee

Update Employee

Delete Employee

ID	First	Last	Salary	Designation	Contact no	Duty Start	Duty End
e11	UniKrishnan		10000	pump_attendant	9485736274	16:00:00	23:59:00
e12	Madhav	Vishnu	10000	pump_attendant	989516156	00:00:00	12:00:00

Current Pump Attendant Incharge : [e11] UniKrishnan

Add Employee

Employee ID

Contact Number

First Name

Duty Start Time

Last Name

Duty End Time


Designation

Salary

Manager

Confirm

Fig:6.2 Employee section



Home

Register Vehicle

Employees

Supplier

Invoice History

Options

Log out

Exit

Gas Station ID : G3

Ramnagar Circle

Maharashtra - 400001

8:58:14 PM

09-02-2022

Wednesday

Supplier Details

Refill Silo

Supplier Name : Hindustan Petrol

Current Rates

Petrol

73.5

Diesel

80.5

Supplier email : contact@hpetrol.in

Fig:6.3 Supplier section

Fuel Management System

Gas Station ID : G3
Ramnagar Circle
Maharashtra - 400001

8:58:32 PM
09-02-2022
Wednesday

Retrieve/Refresh All Invoices Invoice ID/R no/License no Find through Invoice ID Find through Registration Number Find through License Number

Invoice ID	Number Plate	Fuel Volume (L)	Cost (₹)	Time	Date
20	KA51HQ4789	157	18086.4	21:59:35	17-01-2022
19	KA51HQ4789	100	11520	21:59:26	17-01-2022
18	MH02KQ7456	4.6598322460...	500	21:58:56	17-01-2022
17	MH02KQ7456	4.6598322460...	500	21:58:54	17-01-2022
16	MH02KQ7456	4.6598322460...	500	21:58:52	17-01-2022
15	MH02KQ7456	4.6598322460...	500	21:58:50	17-01-2022
14	MH02KQ7456	4.6598322460...	500	21:58:48	17-01-2022
13	MH02KQ7456	4.6598322460...	500	21:58:47	17-01-2022
12	MH02KQ7456	4.6598322460...	500	21:58:45	17-01-2022
11	MH02KQ7456	4.6598322460...	500	21:58:42	17-01-2022
10	MH02KQ7456	4.6598322460...	500	21:58:37	17-01-2022
9	TN34IL5402	4.3402777777...	500	21:50:25	17-01-2022
8	KL12YU8337	12	1382.4	21:15:23	17-01-2022

Home Register Vehicle Employees Supplier Invoice History Options Log out Exit

Fig:6.4 Invoice section

Gas Station ID : G3
Ramnagar Circle
Maharashtra - 400001

8:58:48 PM
09-02-2022
Wednesday

Change Password

New Password
Confirm New Password
Current Password

Change Password

Update Fuel Capacity

Petrol Capacity Update
Diesel Capacity Update

Log out Exit

Fig:6.5 options section


Username
Password

Login

Create new account

Exit

Fig:6.6 Login section



Home

Register Vehicle

Employees

Supplier

Invoice History

Options

Log out

Exit

Gas Station ID : G3

Ramnagar Circle

Maharashtra - 400001

8:57:28 PM

09-02-2022

Wednesday

Registration Number

Amount

L ▾

Refill

Current Rates

Petrol 115.2

Diesel 107.3

Petrol Available 300.00

Diesel Available 400.00

Refresh Rates

Fig:6.7 Home section

Chapter 7

CONCLUSION & FUTURE SCOPE

This project is built keeping in mind that it is to be used by consumer. It is built for managing Consumer Details. According to the requested requirement the pump attendant can add the consumer, view the consumer details. Similarly. The required records can be easily viewed by the admin and build real world systems using the data recorded in the database. Numerous validations implemented would enable the entry of the consumer data to be accurate. The main objective of this framework is to save time, make the system cost effective and management records efficiently.

Future scope of the work:

- ✓ The option to print the records in future.
- ✓ The system can be developed in such a way that its existing features can be modified to better versions.

BIBLIOGRAPHY

- During the course of this project, reference to the following materials were made
- [1] <https://docs.microsoft.com/en-us/dotnet/>
 - [2] <https://docs.microsoft.com/en-us/sql/?view=sql-server-ver15>
 - [3] <https://docs.microsoft.com/en-us/dotnet/desktop/winforms/?view=netdesktop-6.0>
 - [4] <https://docs.microsoft.com/en-us/dotnet/csharp/>
 - [5] Fundamentals of Database Systems, Ramez Elmasri and Shamkant B. Navathe, 7th Edition, 2017, Pearson
 - [6] Database management systems, Ramakrishnan, and Gehrke, 3rd Edition, 2014, McGraw Hill