Database System Project

GROUP MEMBERS:

221017 Aashfa Noor

221043 Lubna Arif

SUBMITTED TO:

Ms. Faryal Farooq

Cargo Management System

Introduction:

Our project aims to develop a comprehensive database system for managing cargo logistics operations for transportation companies and logistics providers. The database will serve as a central repository for storing and managing information related to vehicles, drivers, trips, cargo, clients, maintenance records, fuel logs, and warehouses.

Purpose:

The primary purpose of the database is to streamline and optimize cargo logistics operations, ensuring efficient transportation and distribution of goods. By centralizing data and automating key processes, the database will help improve operational efficiency, enhance decision-making capabilities, and facilitate better coordination among stakeholders.

Target Audience:

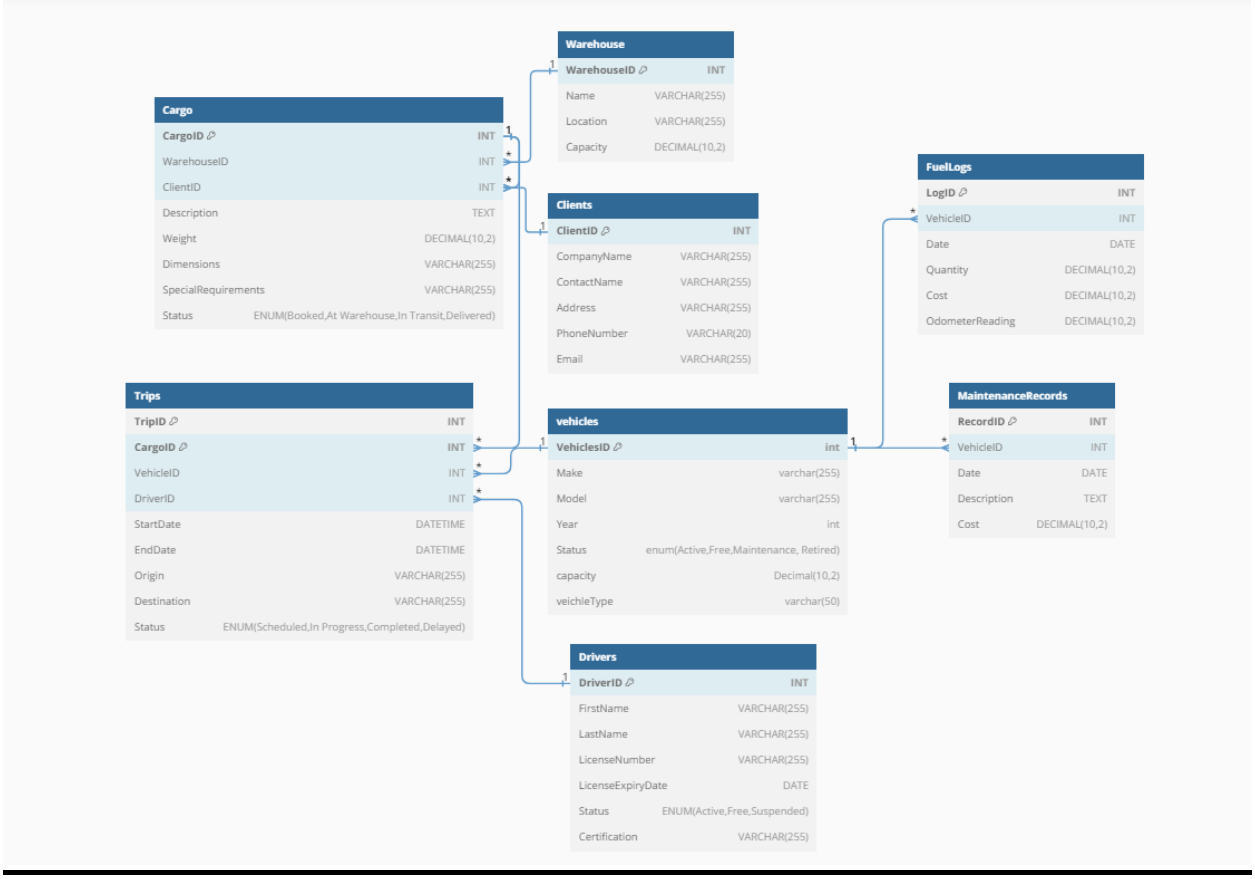
The target audience for the database includes:

Logistics Managers: Responsible for overseeing transportation and distribution operations, monitoring vehicle and driver assignments, tracking trip statuses, and managing cargo shipments to ensure timely deliveries.

Fleet Managers: Tasked with managing the company's fleet of vehicles, scheduling maintenance activities, tracking fuel usage, monitoring vehicle performance, and ensuring compliance with regulations.

Operations Staff: Involved in day-to-day operations such as trip coordination, resource allocation, client management, and warehouse inventory management.

Entity Relationship Diagram



List of tables along with their attributes and data types, as well as primary and foreign keys:

1. Vehicles:

- VehiclesID: INT (Primary Key)
- Make: VARCHAR(255)
- Model: VARCHAR(255)
- Year: INT
- Status: ENUM ('Active', 'Maintenance', 'Retired')
- Capacity: DECIMAL(10,2)
- VeichleType: VARCHAR(255)

2. Drivers:

- DriverID: INT (Primary Key)
- FirstName: VARCHAR(255)

- LastName: VARCHAR(255)
- LicenseNumber: VARCHAR(255)
- LicenseExpiryDate: DATE
- Status: ENUM ('Active', 'Suspended')
- Certification: VARCHAR(255)

3. Trips:

- TripID: INT (Primary Key)
- CargoID: INT (Foreign Key references Cargo.CargoID)
- VehicleID: INT (Foreign Key references Vehicles.VehiclesID)
- DriverID: INT (Foreign Key references Drivers.DriverID)
- StartDate: DATETIME
- EndDate: DATETIME
- Origin: VARCHAR(255)
- Destination: VARCHAR(255)
- Status: ENUM ('Scheduled', 'In Progress', 'Completed', 'Delayed')

4. Cargo:

- CargoID: INT (Primary Key)
- WarehouseID: INT (Foreign Key references Warehouse.WarehouseID)
- ClientID: INT (Foreign Key references Clients.ClientID)
- Description: TEXT
- Weight: DECIMAL(10,2)
- Dimensions: VARCHAR(255)
- SpecialRequirements: VARCHAR(255)
- Status: ENUM ('At Warehouse', 'In Transit', 'Delivered')

5. Clients:

- ClientID: INT (Primary Key)
- CompanyName: VARCHAR(255)
- ContactName: VARCHAR(255)
- Address: VARCHAR(255)
- PhoneNumber: VARCHAR(20)

- Email: VARCHAR(255)

6. MaintenanceRecords:

- RecordID: INT (Primary Key)
- VehicleID: INT (Foreign Key references Vehicles.VehiclesID)
- Date: DATE
- Description: TEXT
- Cost: DECIMAL(10,2)

7. FuelLogs:

- LogID: INT (Primary Key)
- VehicleID: INT (Foreign Key references Vehicles.VehiclesID)
- Date: DATE
- Quantity: DECIMAL(10,2)
- Cost: DECIMAL(10,2)
- OdometerReading: DECIMAL(10,2)

8. Warehouse:

- WarehouseID: INT (Primary Key)
- Name: VARCHAR(255)
- Location: VARCHAR(255)
- Capacity: DECIMAL(10,2)

Features and Functionality:

The database will include the following features and functionality:

Vehicle Management: Tracking vehicle details, maintenance records, and fuel logs.

Driver Management: Managing driver information, licenses, and certifications.

Trip Management: Scheduling and tracking trips, monitoring trip statuses, and recording trip details.

Cargo Management: Managing cargo shipments, tracking cargo statuses, and recording cargo details.

Client Management: Storing client information, managing client relationships, and handling client inquiries.

Warehouse Management: Recording warehouse locations, capacities, and inventory.

Maintenance Records: Recording vehicle maintenance activities, costs, and schedules.

Fuel Logs: Logging fuel consumption, costs, and odometer readings for vehicles.

Benefits:

The database will offer several benefits, including:

Improved Operational Efficiency: Streamlining processes, reducing manual tasks, and enhancing data accuracy.

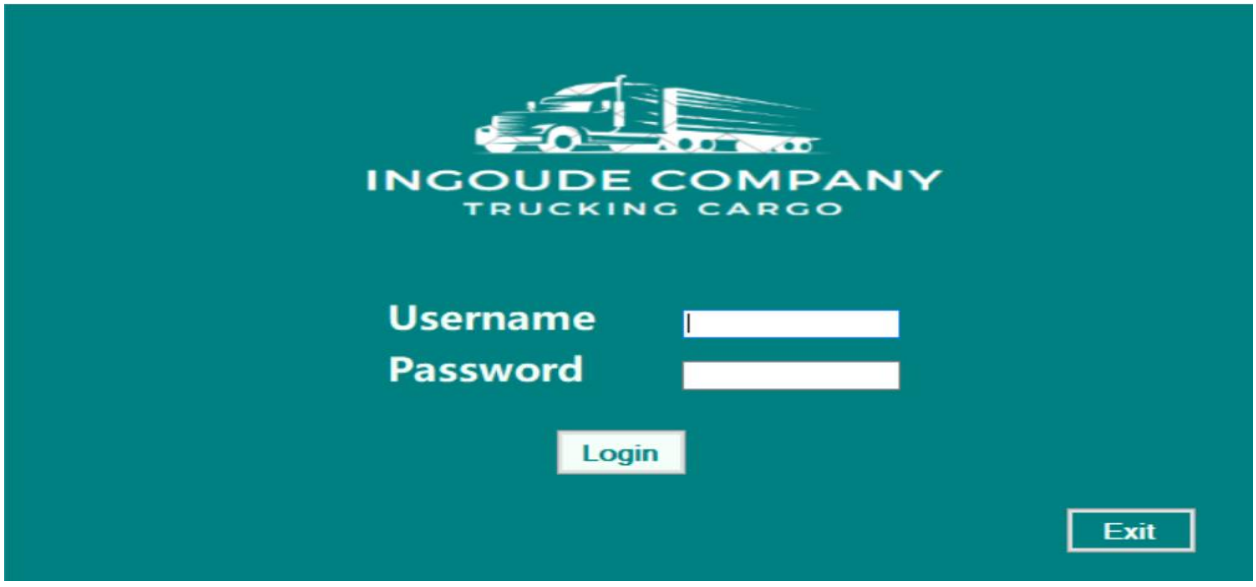
Enhanced Decision Making: Providing real-time insights, analytics, and reporting capabilities to support informed decision-making.

Regulatory Compliance: Ensuring compliance with industry regulations and standards through accurate record-keeping and reporting.

Cost Savings: Optimizing resource utilization, reducing downtime, and minimizing fuel wastage through efficient fleet management.

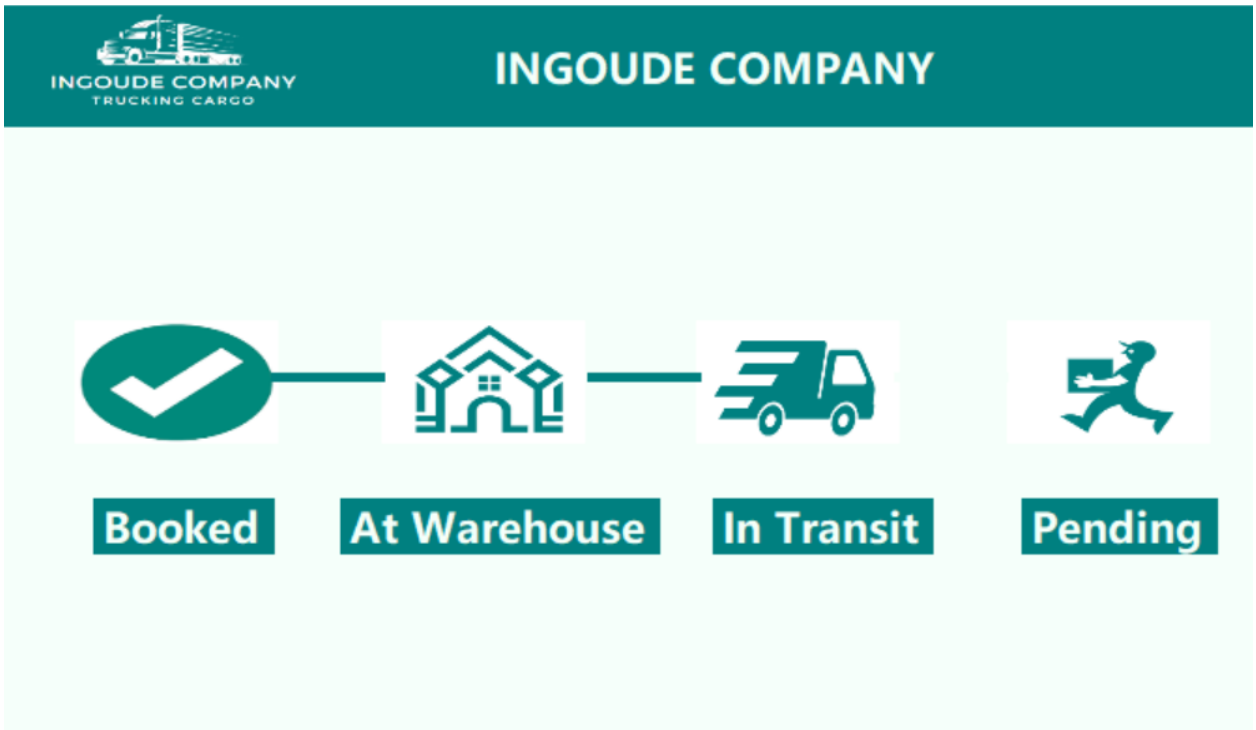
Interfaces:

1- Firstly, Login page appeared.

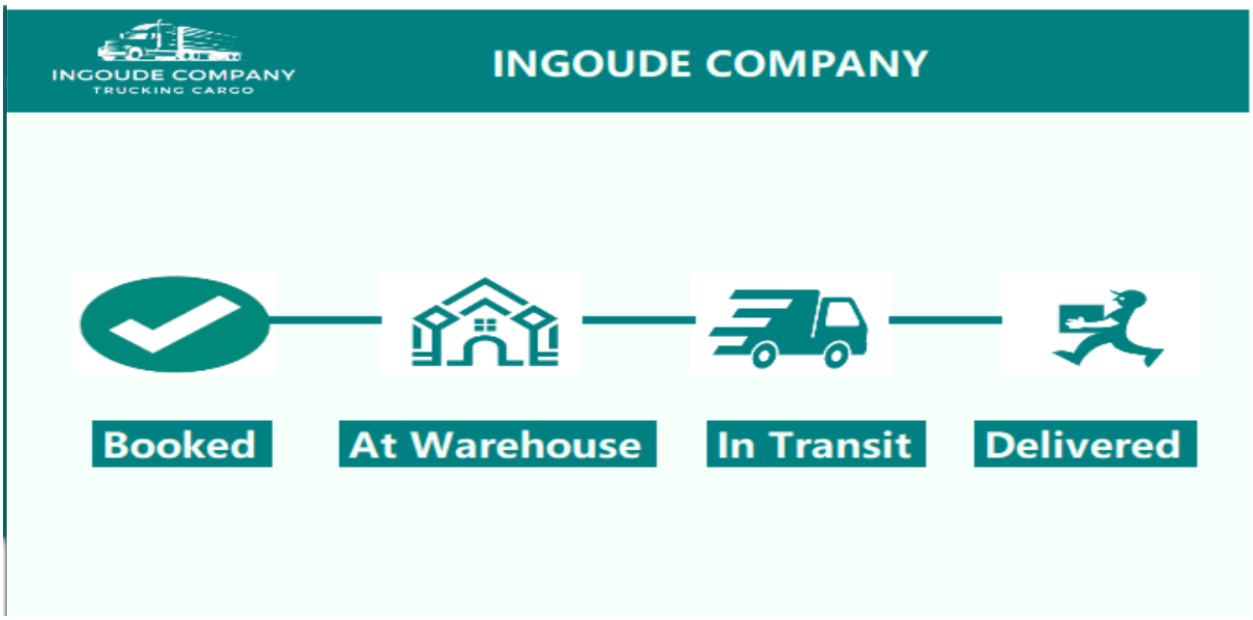


2- After login main page appear with menu on side and by click on track cargo we can track cargo process on the basis of cargo ID.

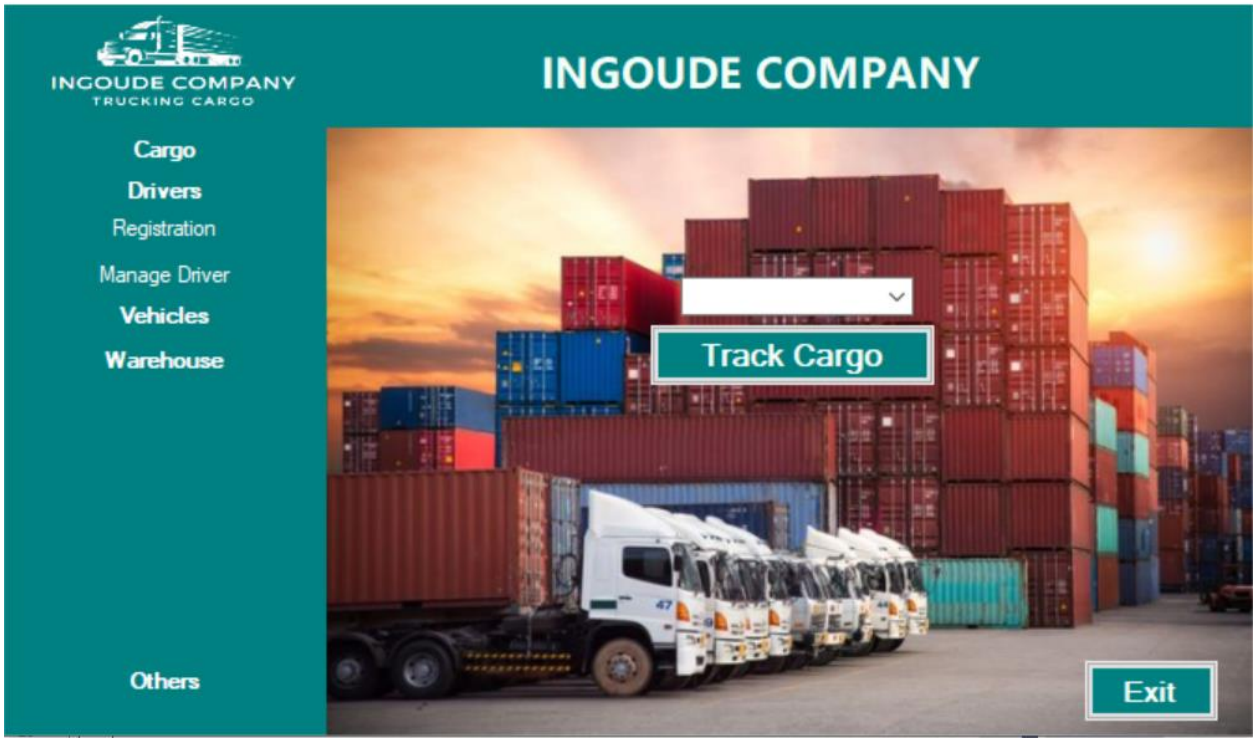
3- After clicking on track cargo button an intaerface appear which show cargo status in transit state and delivery is pending. (By seeing teal color line and text box under the symbols)



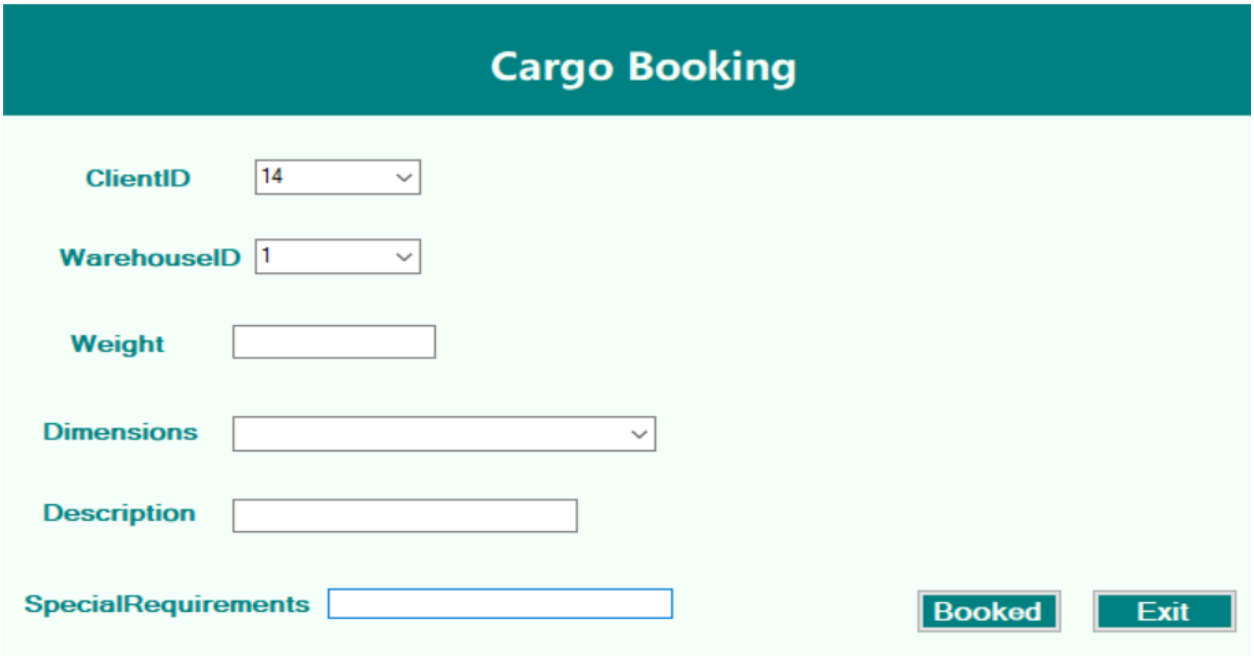
4- This interface showing that cargo delivered.



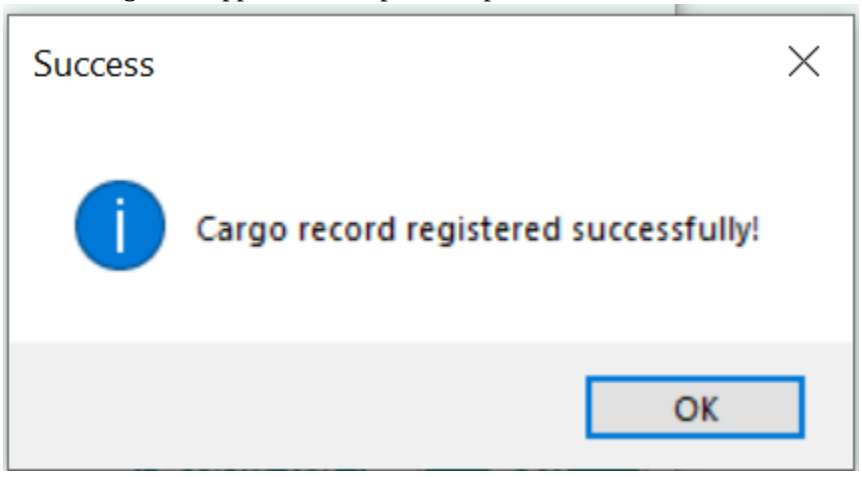
5- By clicking on side menu button submenu also appear.(in this picture driver submenu appear by clicking on driver button).



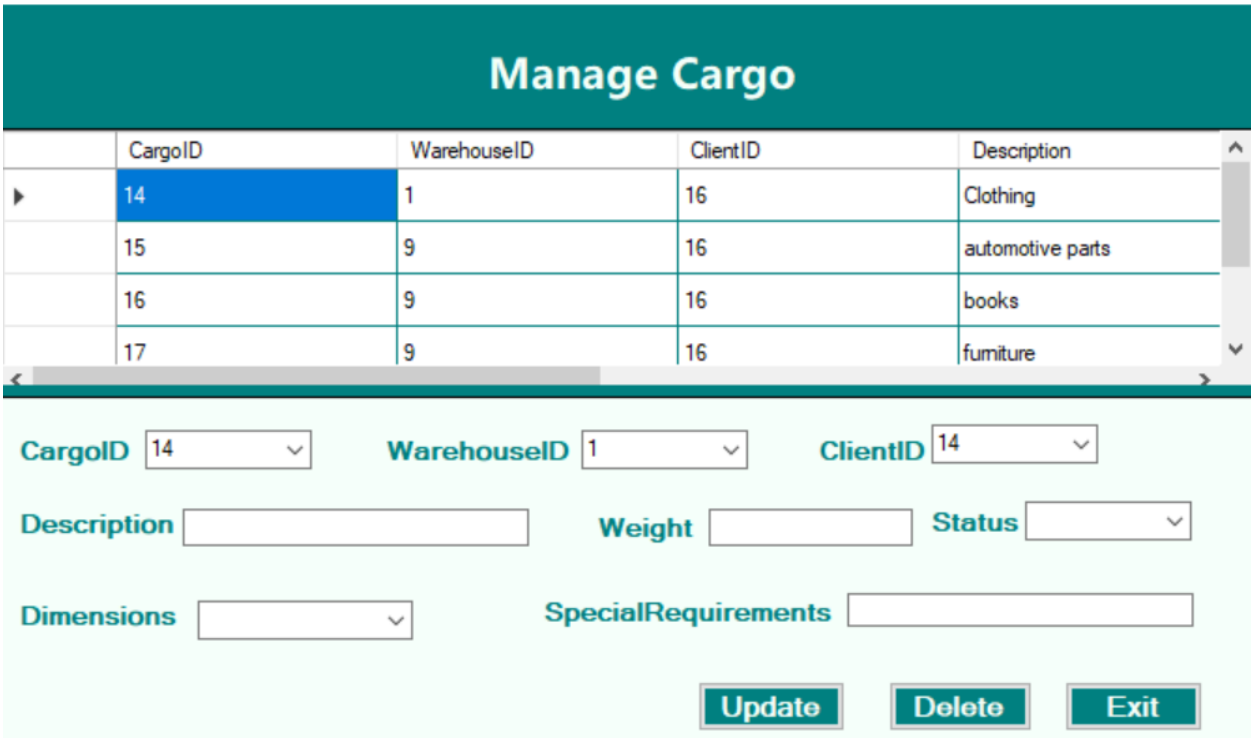
6-By filling this form we book cargo at billing on booked button status of cargo automatically set to “Booked”.



7- Meassage box appear when operation performed on database like delete, update, insert e-t-c.



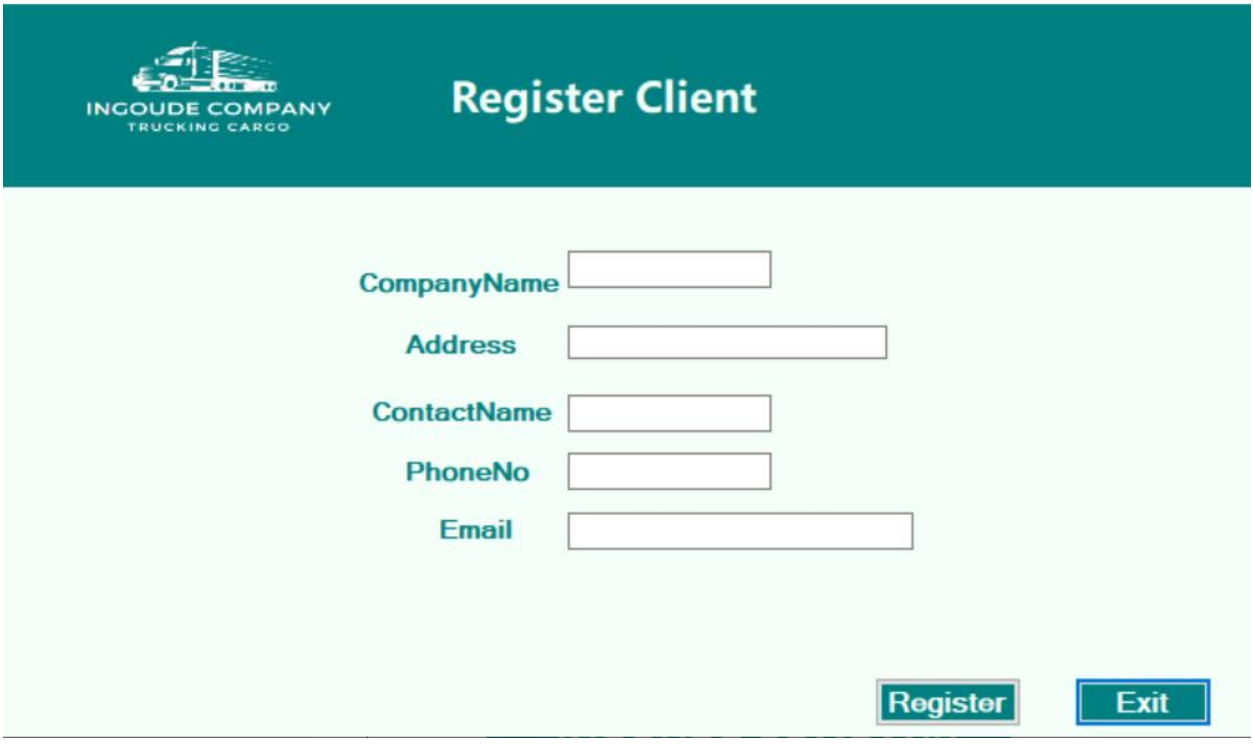
8-This interface showing cargo table info in grid view. By clicking on cell form under the gridview cell automaticlly filled with data relevant to that row and we can easily update by changing the attributes we want to update.



9- For those variable in which we restrict data combobox appear which show some different options to select. (These type of form also appear for all tables where we can perform add, update and delete actions on database)

10- By clicking on other button this interface appear .

11- Client registration form by entering values we can register client



Code:

Some main functions and module code is given below :

----- Program.cs-----

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

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

----MainForm.cs----

```
using System;
using System.Collections.Generic;
using System.Collections.Specialized;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Cargo_Management_system
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            customizeDesign ();
        }
        private void Form1_Load(object sender, EventArgs e)
        {
```

```
            // TODO: This line of code loads data into the
'cargo_Management_SystemDataSet.Cargo' table. You can move, or remove it, as needed.
            this.cargoTableAdapter.Fill(this.cargo_Management_SystemDataSet.Cargo);

        }


        private void customizeDesign ()
        {
            Cargopanel.Visible= false;
            driverpanel.Visible = false;
            veichlepanel.Visible = false;
            Warehousepanel.Visible = false;
        }
        private void hidesubmenu()
        {
            if(Cargopanel.Visible == true)
            {
                Cargopanel.Visible = false;
            }
            if (driverpanel.Visible == true)
            {
                driverpanel.Visible = false;
            }
            if (veichlepanel.Visible == true)
            {
                veichlepanel.Visible = false;
            }
        }
        private void showSubmenu (Panel submenu)
        {
            if (submenu.Visible == false)
            {
                hidesubmenu();
                submenu.Visible = true;
            }
            else
            {
                submenu.Visible = false;
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            BookingForm Form = new BookingForm();
            Form.Show();
        }

        private void cargobutton_Click(object sender, EventArgs e)
        {
            showSubmenu(Cargopanel);
        }

        private void Driversbutton_Click(object sender, EventArgs e)
        {
            showSubmenu(driverpanel);
        }
```

```
        private void veichlesbutton_Click(object sender, EventArgs e)
        {
            showSubmenu(veichlepanel);
        }

        private void vregisterationbutton_Click(object sender, EventArgs e)
        {
            VeichleRegisterationForm registrationForm = new
VeichleRegisterationForm();
            // Show the form
            registrationForm.Show();
        }

        private void Managevehiclesbutton_Click(object sender, EventArgs e)
        {
            managevehicleForm Form = new managevehicleForm();
            // Show the form
            Form.Show();
        }

        private void exitbutton_Click(object sender, EventArgs e)
        {
            OthersForm2 Form = new OthersForm2();
            Form.Show();
        }

        private void warehousebutton_Click_1(object sender, EventArgs e)
        {
            showSubmenu(Warehousepanel);
        }

        private void WarehouseRegisterationbutton_Click(object sender, EventArgs e)
        {
            WarehouseForm Form = new WarehouseForm();
            // Show the form
            Form.Show();
        }

        private void ManageWarehousebutton_Click(object sender, EventArgs e)
        {
            ManageWarehouseForm3 Form = new ManageWarehouseForm3();
            // Show the form
            Form.Show();
        }

        private void Dregistrationbutton_Click(object sender, EventArgs e)
        {
            DriverRegisterationForm Form = new DriverRegisterationForm();
            // Show the form
            Form.Show();
        }

        private void Managedriverbutton_Click(object sender, EventArgs e)
        {
            ManageDriverForm Form = new ManageDriverForm();
            Form.Show();
        }
```

```
        private void fuelLogsbutton_Click(object sender, EventArgs e)
        {
            FuelLogsForm Form = new FuelLogsForm();
            Form.Show();
        }

        private void maintainnaceRecordbutton_Click(object sender, EventArgs e)
        {
            MaintenanceRecordForm Form = new MaintenanceRecordForm();
            Form.Show();
        }

        private void ManageCargobutton_Click(object sender, EventArgs e)
        {

            ManageCargoForm Form = new ManageCargoForm();
            Form.Show();
        }

        private void Exitbutton_Click_1(object sender, EventArgs e)
        {
            this.Close();
        }



        private void trackbutton_Click(object sender, EventArgs e)
        {
            string connectionString = "Data Source=DESKTOP-
04KFGU9\\SQLSERVER2022;Initial Catalog=\"Cargo Management System\";Integrated
Security=True;Encrypt=False";
            string cargoID = CargoIDcomboBox.Text;

            using (SqlConnection con = new SqlConnection(connectionString))
            {
                con.Open(); // Open the connection


                string statusQuery = "SELECT Status FROM Cargo WHERE CargoID =
@CargoID";
                using (SqlCommand cmd = new SqlCommand(statusQuery, con))
                {
                    cmd.Parameters.AddWithValue("@CargoID", cargoID);


                    string status = cmd.ExecuteScalar()?.ToString();


                    switch (status)
                    {
                        case "Booked":
                            BookedForm bookedForm = new BookedForm();
                            bookedForm.Show();
                            break;
                        case "At Warehouse":
                            AtWarehouseForm warehouseForm = new AtWarehouseForm();
                            warehouseForm.Show();
                            break;
```

```
                        case "In Transit":
                            InTransitForm inTransitForm = new InTransitForm();
                            inTransitForm.Show();
                            break;
                        case "Delivered":
                            DeliveredForm deliveredForm = new DeliveredForm();
                            deliveredForm.Show();
                            break;
                        default:
                            MessageBox.Show("Invalid status or CargoID.");
                            break;
                    }
                }
            }
        }

    }
}
```

----Insert----

```
private void AddFuelLogbutton_Click(object sender, EventArgs e)
{
    string connectionString = "Data Source=DESKTOP-04KFGU9\\SQLSERVER2022;Initial
Catalog=\"Cargo Management System\";Integrated Security=True;Encrypt=False"; //
Replace with your actual connection string
    string insertQuery = @"INSERT INTO FuelLogs (VehicleID, Date, Quantity, Cost,
OdometerReading)
                           VALUES (@VehicleID, @Date, @Quantity, @Cost,
@OdometerReading)";

    using (SqlConnection con = new SqlConnection(connectionString))
    {
        using (SqlCommand cmd = new SqlCommand(insertQuery, con))
        {

            cmd.Parameters.AddWithValue("@VehicleID", VehicleIDcomboBox.Text);
            cmd.Parameters.AddWithValue("@Date",
DateTime.Parse(FuelLogdateTimePicker.Text));
            cmd.Parameters.AddWithValue("@Quantity",
decimal.Parse(QuantitytextBox.Text));
            cmd.Parameters.AddWithValue("@Cost", decimal.Parse(CosttextBox.Text));
            cmd.Parameters.AddWithValue("@OdometerReading",
decimal.Parse(OdometerReadingtextBox.Text));

            try
            {
                con.Open();
                cmd.ExecuteNonQuery();
                con.Close();

                // Show a success message
                MessageBox.Show("Fuel log data inserted successfully!", "Success",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            catch (Exception ex)
            {
                // Show an error message
```

```
                MessageBox.Show($"Error inserting fuel log data: {ex.Message}",
"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }
    this.Close();

}
```

----Update-----

```
private void Updatebutton_Click(object sender, EventArgs e)
{
    string connectionString = "Data Source=DESKTOP-04KFGU9\\SQLSERVER2022;Initial
Catalog=\"Cargo Management System\";Integrated Security=True;Encrypt=False";

    string updateDriverQuery = @"UPDATE Drivers SET FirstName = @FirstName,LastName
= @LastName,LicenseNumber = @LicenseNumber,LicenseExpiryDate = @LicenseExpiryDate,
                                Status = @Status,Certification = @Certification WHERE
DriverID = @DriverID";

    using (SqlConnection con = new SqlConnection(connectionString))
    {
        using (SqlCommand cmd = new SqlCommand(updateDriverQuery, con))
        {
            cmd.Parameters.AddWithValue("@DriverID",
int.Parse(DriverIDcomboBox.Text));
            cmd.Parameters.AddWithValue("@FirstName", DriverFirstNametextBox.Text);
            cmd.Parameters.AddWithValue("@LastName", DriverLastNametextBox.Text);
            cmd.Parameters.AddWithValue("@LicenseNumber",
DriverLicenseNotextBox.Text);
            cmd.Parameters.AddWithValue("@LicenseExpiryDate",
DateTime.Parse(DriverLicencedateTimePicker.Text));
            cmd.Parameters.AddWithValue("@Status", DriverStatuscombobox.Text);
            cmd.Parameters.AddWithValue("@Certification",
DriverCertificationtextBox.Text);

            try
            {
                con.Open();
                cmd.ExecuteNonQuery();
                con.Close();

                // Show a success message
                MessageBox.Show("Driver data updated successfully!", "Success",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            catch (Exception ex)
            {
                // Show an error message
                MessageBox.Show($"Error updating driver data: {ex.Message}",
"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }
    this.Close();
```

```
}
```

----Delete-----

```
private void Vdeletebutton_Click(object sender, EventArgs e)
 {
     string deleteDriverQuery = @"DELETE FROM Drivers WHERE DriverID = @DriverID";

     using (SqlConnection con = new SqlConnection("Data Source=DESKTOP-
04KFGU9\\SQLSERVER2022;Initial Catalog=\"Cargo Management System\";Integrated
Security=True;Encrypt=False"))
     {
         using (SqlCommand cmd = new SqlCommand(deleteDriverQuery, con))
         {
             cmd.Parameters.AddWithValue("@DriverID",
int.Parse(DriverIDcomboBox.Text));

             try
             {
                 con.Open();
                 cmd.ExecuteNonQuery();
                 con.Close();

                 // Show a success message
                 MessageBox.Show("Driver data deleted successfully!", "Success",
MessageBoxButtons.OK, MessageBoxIcon.Information);
             }
             catch (Exception ex)
             {
                 // Show an error message
                 MessageBox.Show($"Error deleting driver data: {ex.Message}",
"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
             }
         }
     }
     this.Close();

 }
                                ----GridView----
private void MVdataGridView_CellContentClick(object sender,
DataGridViewCellEventArgs e)
 {
     if (e.RowIndex >= 0)
     {
         DriverIDcomboBox.Text =
DriversDataGridView.Rows[e.RowIndex].Cells[0].Value.ToString();
         DriverFirstNametextBox.Text =
DriversDataGridView.Rows[e.RowIndex].Cells[1].Value.ToString();
         DriverLastNametextBox.Text =
DriversDataGridView.Rows[e.RowIndex].Cells[2].Value.ToString();
         DriverLicenseNotextBox.Text =
DriversDataGridView.Rows[e.RowIndex].Cells[3].Value.ToString();
         DriverLicencedateTimePicker.Text =
DriversDataGridView.Rows[e.RowIndex].Cells[4].Value.ToString();
         DriverStatuscombobox.Text =
DriversDataGridView.Rows[e.RowIndex].Cells[5].Value.ToString();
```

```
         DriverCertificationtextBox.Text =
DriversDataGridView.Rows[e.RowIndex].Cells[6].Value.ToString();
     }

 }
```

Conclusion:

In conclusion, the proposed cargo logistics database will serve as a valuable tool for transportation companies and logistics providers to streamline operations, improve efficiency, and deliver superior service to clients. By centralizing data and providing insights for decision-making, the database will empower stakeholders to achieve operational excellence in the dynamic and competitive logistics industry.
