AGRO-INDUSTRIAL ENTERPRISE MANAGEMENT SYSTEM WINDOWS APPLICATION UNDERSTANDING.

- Visual studio windows forms .net framework
- In each windows application there are 3 parts-
    - User interface
    - Back end
    - Database

USER INTERFACE

- User data entry
- Every User interaction
- Processed Output displaying
- Will define each part in accordance of each part purpose in the backend
- Textboxes, panels, Labels, Data grid view…. etc.
- Each item must be declared with a name in order to call them each by each in the backend.
- Name helps to manipulate action

BACKEND

- Define each item on UI in accordance of their desired purpose
- Event handling. (_click).
- Responsiveness for each event
- In each data entry field through this it will be look into data inputs. After that taking actions, resetting fields, sending or retrieving data from database such events will be handled here.

DATABASE

- All application data entry will be stored.
- User data entry through UI also be here.
- Store, Retrieve.
- This is an individual object. It's not attached to part of the application until the CONNECTION STRING of the database is being assigned to the backend.

- This string enables the path for communication between the backend and the database by query messages.
- First in namespace use sqlclient name space field.
- Define a new sqlconnection in the code and In each private method you have to-
    - open sql connection
    - give the desired command as an sql query
    - Execute command.
    - Close connection
- Insert data, update data, Delete data.

UNDERSTANDING THE APPLICATION.

- Design the UI to enter data. (forms, colors, navigation panel, DGV, labels)
- Name them to access in Backend.
- Create database and it's tables in accordance of the forms and data entry fields.
- Give data tables names, null or no null, be aware of naming database data entry fields(column names), data types (just as given in the back end).
- Add a new sqlconnection.
- Save button _click event.
    - Use if statement to check if all the data entry fields are filled or empty. If empty show a message box saying missing info.
    - If data entries are not empty do an exception handling.
    - Inside of a try block put the code that may raises and exception.
    - This is a private method. Open connection.
    - In the sql Command Write a query to Insert data into data table. (Because of save button).
    - ("`Purpose` `which Table` `Database fields` `placeholders` ",con).
    - Database fields are Column names
    - Placeholders (@....) carries inserted data fields to database fields. Assign each these inserted data fields to these placeholders.
    - After assigning each values execute the command.
    - Close connection.
    - Message box to show data added!

- Data Grid View is used to show data entered in to the database in the UI.
    - Open sql connection.
    - Query to select all the data entry rows from specific data table.

- To retrieve data from database towards the backend we use tool called SqlDataAdapter.
- Which goes through the connection string to the database and retrieves data and bring towards the back end.
- That retrieved data will be filled into a Dataset. Then SqldataAdapter go back to its' empty state.
- Data source field of the DGV is set to this data set.
- Close the connection.

➢ Sql Data Adapter- Retrieves data, populate data sets or data tables, work with disconnected data.

➢ Sql Data Reader- Reads data, one row at a time, doesn't load entire data in to the memory at once, forward only, Processes data sequentially.

➢ Declare a reset method.
- Assign each data input fields in to their empty states.

➢ In the end use a catch block to show if there is any error in the code during the process.

➢ In Edit event use Update data table query in Delete event use Delete from data table query.

➢ Data grid view _cell content click event

- Select a row in DGV.
- Retrieve data back from DGV turn them back into string or int values and load them back into their necessary data input fields.

➢ Buffer through the navigation menu
- Assign desired form as obj instance.
- Call show method.
- Call hide method to hide the current form so concurrent form can be showed up.

FORMS OF APPLICATION

➢ Home
➢ Inventory Management
➢ Sales and Marketing
➢ Crop Management
➢ Supply chain Management.
➢ Financial Analysis
➢ Employee Management
➢ Quality Control
➢ Equipment Maintenance
➢ Login

EXAMPLE CODE TO HELP UNDERSTANDING

```
// Department Class


// Constructor
public Departments()
{
    InitializeComponent();
    showDep(); // Call showDep method to display departments on form load
}

// Method to display departments in the DataGridView
private void showDep()
{
    Con.Open(); // Open the database connection
    string Query = "select * from DepartmentTbl"; // SQL query to select all
departments
    SqlDataAdapter sda = new SqlDataAdapter(Query, Con); // Create a
SqlDataAdapter with the query and connection
    SqlCommandBuilder builder = new SqlCommandBuilder(sda); // Create a
SqlCommandBuilder to handle data updates
    var ds = new DataSet(); // Create a new DataSet
    sda.Fill(ds); // Fill the DataSet with data from the database
    DepDGV.DataSource = ds.Tables[0]; // Set the DataGridView's data source to
the first table in the DataSet
    Con.Close(); // Close the database connection
}

// Method to reset the form fields
```

```
private void Reset()
{
    DepNameTb.Text = ""; // Clear the Department Name textbox
    DepIntakeTb.Text = ""; // Clear the Department Intake textbox
    DepFeesTb.Text = ""; // Clear the Department Fees textbox
}

// Event handler for adding a new department
private void SaveBtn_Click(object sender, EventArgs e)
{
    // Check if any of the required fields are empty
    if (DepNameTb.Text == "" || DepIntakeTb.Text == "" || DepFeesTb.Text == "")
    {
        MessageBox.Show("Missing Information"); // Show an error message
    }
    else
    {
        try
        {
            Con.Open(); // Open the database connection
            // SQL command to insert a new department
            SqlCommand cmd = new SqlCommand("Insert into DepartmentTbl(DepName,
DepIntake, DepFees) values(@DN, @DI, @DF)", Con);
            cmd.Parameters.AddWithValue("@DN", DepNameTb.Text); // Add the
Department Name parameter
            cmd.Parameters.AddWithValue("@DI", DepIntakeTb.Text); // Add the
Department Intake parameter
            cmd.Parameters.AddWithValue("@DF", DepFeesTb.Text); // Add the
Department Fees parameter
            cmd.ExecuteNonQuery(); // Execute the SQL command
            MessageBox.Show("Department added"); // Show a success message
            Con.Close(); // Close the database connection
            ShowDep(); // Refresh the DataGridView with updated data
            Reset(); // Reset the form fields
        }
        catch (Exception Ex)
        {
            MessageBox.Show(Ex.Message); // Show any error message
        }
    }
}

// Variable to store the selected department's ID
int Key = 0;
```

```
// Event handler for selecting a department in the DataGridView
private void DepDGV_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
    // Populate the form fields with the selected department's data
    DepNameTb.Text = DepDGV.SelectedRows[0].Cells[1].Value.ToString();
    DepIntakeTb.Text = DepDGV.SelectedRows[0].Cells[2].Value.ToString();
    DepFeesTb.Text = DepDGV.SelectedRows[0].Cells[3].Value.ToString();

    // Check if a department is selected
    if (DepNameTb.Text == "")
    {
        Key = 0; // If no department is selected, set Key to 0
    }
    else
    {
        // Otherwise, set Key to the selected department's ID
        Key = Convert.ToInt32(DepDGV.SelectedRows[0].Cells[0].Value.ToString());
    }
}

// Event handler for editing a department
private void EditBtn_Click(object sender, EventArgs e)
{
    // Check if any of the required fields are empty
    if (DepNameTb.Text == "" || DepIntakeTb.Text == "" || DepFeesTb.Text == "")
    {
        MessageBox.Show("Missing Information"); // Show an error message
    }
    else
    {
        try
        {
            Con.Open(); // Open the database connection
            // SQL command to update an existing department
            SqlCommand cmd = new SqlCommand("Update DepartmentTbl set
DepName=@DN, DepIntake=@DI, DepFees=@DF where DepNum=@DKey", Con);
            cmd.Parameters.AddWithValue("@DN", DepNameTb.Text); // Add the
Department Name parameter
            cmd.Parameters.AddWithValue("@DI", DepIntakeTb.Text); // Add the
Department Intake parameter
            cmd.Parameters.AddWithValue("@DF", DepFeesTb.Text); // Add the
Department Fees parameter
            cmd.Parameters.AddWithValue("@DKey", Key); // Add the Department ID
parameter
            cmd.ExecuteNonQuery(); // Execute the SQL command
```

```
            MessageBox.Show("Department Updated"); // Show a success message
            Con.Close(); // Close the database connection
            ShowDep(); // Refresh the DataGridView with updated data
            Reset(); // Reset the form fields
        }
        catch (Exception Ex)
        {
            MessageBox.Show(Ex.Message); // Show any error message
        }
    }
}

// Event handler for deleting a department
private void DeleteBtn_Click(object sender, EventArgs e)
{
    // Check if a department is selected
    if (Key == 0)
    {
        MessageBox.Show("Select the Department"); // Show an error message if no
department is selected
    }
    else
    {
        try
        {
            Con.Open(); // Open the database connection
            // SQL command to delete a department
            SqlCommand cmd = new SqlCommand("Delete from DepartmentTbl where
DepNum=@DKey", Con);
            cmd.Parameters.AddWithValue("@DKey", Key); // Add the Department ID
parameter
            cmd.ExecuteNonQuery(); // Execute the SQL command
            MessageBox.Show("Department Deleted"); // Show a success message
            Con.Close(); // Close the database connection
            ShowDep(); // Refresh the DataGridView with updated data
            Reset(); // Reset the form fields
        }
        catch (Exception Ex)
        {
            MessageBox.Show(Ex.Message); // Show any error message
        }
    }
}

// Event handler for navigating to the Student form
```

```
private void label2_Click(object sender, EventArgs e)
{
    Students Obj = new Students(); // Create a new instance of the Students form
    Obj.Show(); // Show the Students form
    this.Hide(); // Hide the current form
}

// Event handler for navigating to the Departments form
private void label4_Click(object sender, EventArgs e)
{
    Departments Obj = new Departments(); // Create a new instance of the
Departments form
    Obj.Show(); // Show the Departments form
    this.Hide(); // Hide the current form
}

// Student Class

// Method to display students in the DataGridView
private void showStudent()
{
    Con.Open(); // Open the database connection
    string Query = "select * from StudentTbl"; // SQL query to select all
students
    SqlDataAdapter sda = new SqlDataAdapter(Query, Con); // Create a
SqlDataAdapter with the query and connection
    SqlCommandBuilder builder = new SqlCommandBuilder(sda); // Create a
SqlCommandBuilder to handle data updates
    var ds = new DataSet(); // Create a new DataSet
    sda.Fill(ds); // Fill the DataSet with data from the database
    stdDGV.DataSource = ds.Tables[0]; // Set the DataGridView's data source to
the first table in the DataSet
    Con.Close(); // Close the database connection
}

// Method to reset the form fields
private void Reset()
{
    StdTb.Text = ""; // Clear the Student Name textbox
    StdGenCb.SelectedIndex = -1; // Reset the Student Gender combobox
    PhoneTb.Text = ""; // Clear the Student Phone textbox
    StdAddTb.Text = ""; // Clear the Student Address textbox
    DepNameTb.Text = ""; // Clear the Department Name textbox
    DepIdCb.SelectedIndex = -1; // Reset the Department ID combobox
}
```

```
// Method to populate the Department ID combobox
private void GetDepId()
{
    Con.Open(); // Open the database connection
    SqlCommand cmd = new SqlCommand("Select DepNum from DepartmentTbl", Con); //
SQL query to select department IDs
    SqlDataReader Rdr; // Create a SqlDataReader object
    Rdr = cmd.ExecuteReader(); // Execute the SQL query and get the result set
    DataTable dt = new DataTable(); // Create a new DataTable
    dt.Columns.Add("DepNum", typeof(int)); // Add a column to the DataTable for
the department ID
    dt.Load(Rdr); // Load the result set into the DataTable
    DepIdCb.ValueMember = "DepNum"; // Set the value member of the combobox to
the department ID column
    DepIdCb.DataSource = dt; // Set the data source of the combobox to the
DataTable
    Con.Close(); // Close the database connection
}

// Method to populate the Department Name textbox based on the selected
Department ID
private void GetDepName()
{
    Con.Open(); // Open the database connection
    // SQL query to select the department name for the selected department ID
    string Query = "Select DepName from DepartmentTbl where DepNum=" +
DepIdCb.SelectedValue.ToString();
    SqlCommand cmd = new SqlCommand(Query, Con); // Create a SqlCommand with the
query and connection
    DataTable dt = new DataTable(); // Create a new DataTable
    SqlDataAdapter sda = new SqlDataAdapter(cmd); // Create a SqlDataAdapter with
the SqlCommand
    sda.Fill(dt); // Fill the DataTable with the result set
    foreach (DataRow dr in dt.Rows) // Iterate through the rows of the DataTable
    {
        DepNameTb.Text = dr["DepName"].ToString(); // Set the Department Name
textbox to the department name
    }
    Con.Close(); // Close the database connection
}

// Event handler for changing the selected Department ID in the combobox
private void DepIdCb_SelectionChangeCommitted(object sender, EventArgs e)
{
```

```
    GetDepName(); // Call the GetDepName method to populate the Department Name
textbox
}

// Event handler for adding a new student
private void SaveBtn_Click(object sender, EventArgs e)
{
    // Check if any of the required fields are empty or not selected
    if (StdTb.Text == "" || StdAddTb.Text == "" || DepNameTb.Text == "" ||
DepIdCb.SelectedIndex == -1 || PhoneTb.Text == "" || SemCb.SelectedIndex == -1 ||
StdGenCb.SelectedIndex == -1)
    {
        MessageBox.Show("Missing Information"); // Show an error message
    }
    else
    {
        try
        {
            Con.Open(); // Open the database connection
            // SQL command to insert a new student
            SqlCommand cmd = new SqlCommand("Insert into StudentTbl(StName,
StDOB, StGen, StAddress, StDepId, StDepName, StPhone, StSem) values(@SN, @SD,
@SG, @SA, @SDI, @SDN, @SP, @SS)", Con);
            cmd.Parameters.AddWithValue("@SN", StdTb.Text); // Add the Student
Name parameter
            cmd.Parameters.AddWithValue("@SD", StdDOB.Value.Date); // Add the
Student Date of Birth parameter
            cmd.Parameters.AddWithValue("@SG", StdGenCb.SelectedItem.ToString());
// Add the Student Gender parameter
            cmd.Parameters.AddWithValue("@SA", StdAddTb.Text); // Add the Student
Address parameter
            cmd.Parameters.AddWithValue("@SDI",
DepIdCb.SelectedValue.ToString()); // Add the Student Department ID parameter
            cmd.Parameters.AddWithValue("@SDN", DepNameTb.Text); // Add the
Student Department Name parameter
            cmd.Parameters.AddWithValue("@SP", PhoneTb.Text); // Add the Student
Phone parameter
            cmd.Parameters.AddWithValue("@SS", SemCb.SelectedItem.ToString()); //
Add the Student Semester parameter
            cmd.ExecuteNonQuery(); // Execute the SQL command
            MessageBox.Show("Student added"); // Show a success message
            Con.Close(); // Close the database connection
            ShowStudents(); // Refresh the DataGridView with updated data
            Reset(); // Reset the form fields
        }
```

```
        catch (Exception Ex)
        {
            MessageBox.Show(Ex.Message); // Show any error message
        }
    }
}

// Variable to store the selected student's ID
int Key = 0;

// Event handler for selecting a student in the DataGridView
private void StdDGV_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
    // Populate the form fields with the selected student's data
    StdTb.Text = StdDGV.SelectedRows[0].Cells[1].Value.ToString();
    StdDOB.Text = StdDGV.SelectedRows[0].Cells[2].Value.ToString();
    StdGenCb.SelectedItem = StdDGV.SelectedRows[0].Cells[3].Value.ToString();
    StdAddTb.Text = StdDGV.SelectedRows[0].Cells[4].Value.ToString();
    DepIdCb.SelectedValue = StdDGV.SelectedRows[0].Cells[5].Value.ToString();
    DepNameTb.Text = StdDGV.SelectedRows[0].Cells[6].Value.ToString();
    PhoneTb.Text = StdDGV.SelectedRows[0].Cells[7].Value.ToString();
    SemCb.SelectedItem = StdDGV.SelectedRows[0].Cells[8].Value.ToString();

    // Check if a student is selected
    if (StdTb.Text == "")
    {
        Key = 0; // If no student is selected, set Key to 0
    }
    else
    {
        // Otherwise, set Key to the selected student's ID
        Key = Convert.ToInt32(StdDGV.SelectedRows[0].Cells[0].Value.ToString());
    }
}

// Event handler for editing a student
private void EditBtn_Click(object sender, EventArgs e)
{
    // Check if any of the required fields are empty or not selected
    if (StdTb.Text == "" || StdAddTb.Text == "" || DepNameTb.Text == "" ||
DepIdCb.SelectedIndex == -1 || PhoneTb.Text == "" || SemCb.SelectedIndex == -1 ||
StdGenCb.SelectedIndex == -1)
    {
        MessageBox.Show("Missing Information"); // Show an error message
    }
```

```
    else
    {
        try
        {
            Con.Open(); // Open the database connection
            // SQL command to update an existing student
            SqlCommand cmd = new SqlCommand("Update StudentTbl set StName=@SN,
StDOB=@SD, StGen=@SG, StAddress=@SA, StDepId=@SDI, StDepName=@SDN, StPhone=@SP,
StSem=@SS where StNum=@SKey", Con);
            cmd.Parameters.AddWithValue("@SN", StdTb.Text); // Add the Student
Name parameter
            cmd.Parameters.AddWithValue("@SD", StdDOB.Value.Date); // Add the
Student Date of Birth parameter
            cmd.Parameters.AddWithValue("@SG", StdGenCb.SelectedItem.ToString());
// Add the Student Gender parameter
            cmd.Parameters.AddWithValue("@SA", StdAddTb.Text); // Add the Student
Address parameter
            cmd.Parameters.AddWithValue("@SDI",
DepIdCb.SelectedValue.ToString()); // Add the Student Department ID parameter
            cmd.Parameters.AddWithValue("@SDN", DepNameTb.Text); // Add the
Student Department Name parameter
            cmd.Parameters.AddWithValue("@SP", PhoneTb.Text); // Add the Student
```