MARS COLONIZATION PROJECT TECHNICAL DOCUMENTATION

Appendix B

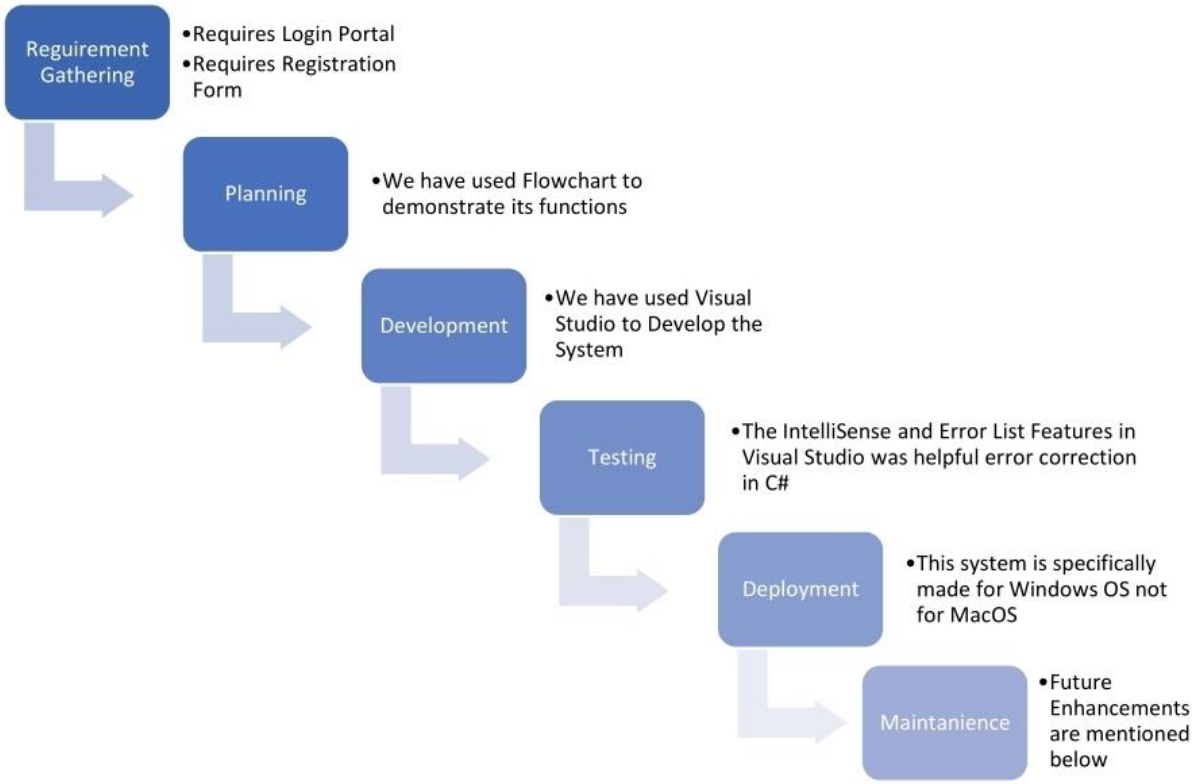
Contents

Introduction

This is an Official Technical Document to our Mars Colonization Management System made for managing Colonist, Dependent, E-Jets, Pilots, Mars Colony and Assignments between them. This Mars Colonization Management is an application created to reduce the work load of Data Entry Operator. This documentation provides a fully detailed explanation of how system works and its functionalities.

Development Process

I have used waterfall model to explain the Development.



Requirement Gathering

The requiremnt required by E-Spae Solutions for the Mars Colonization Project Management Systems are listed down below:

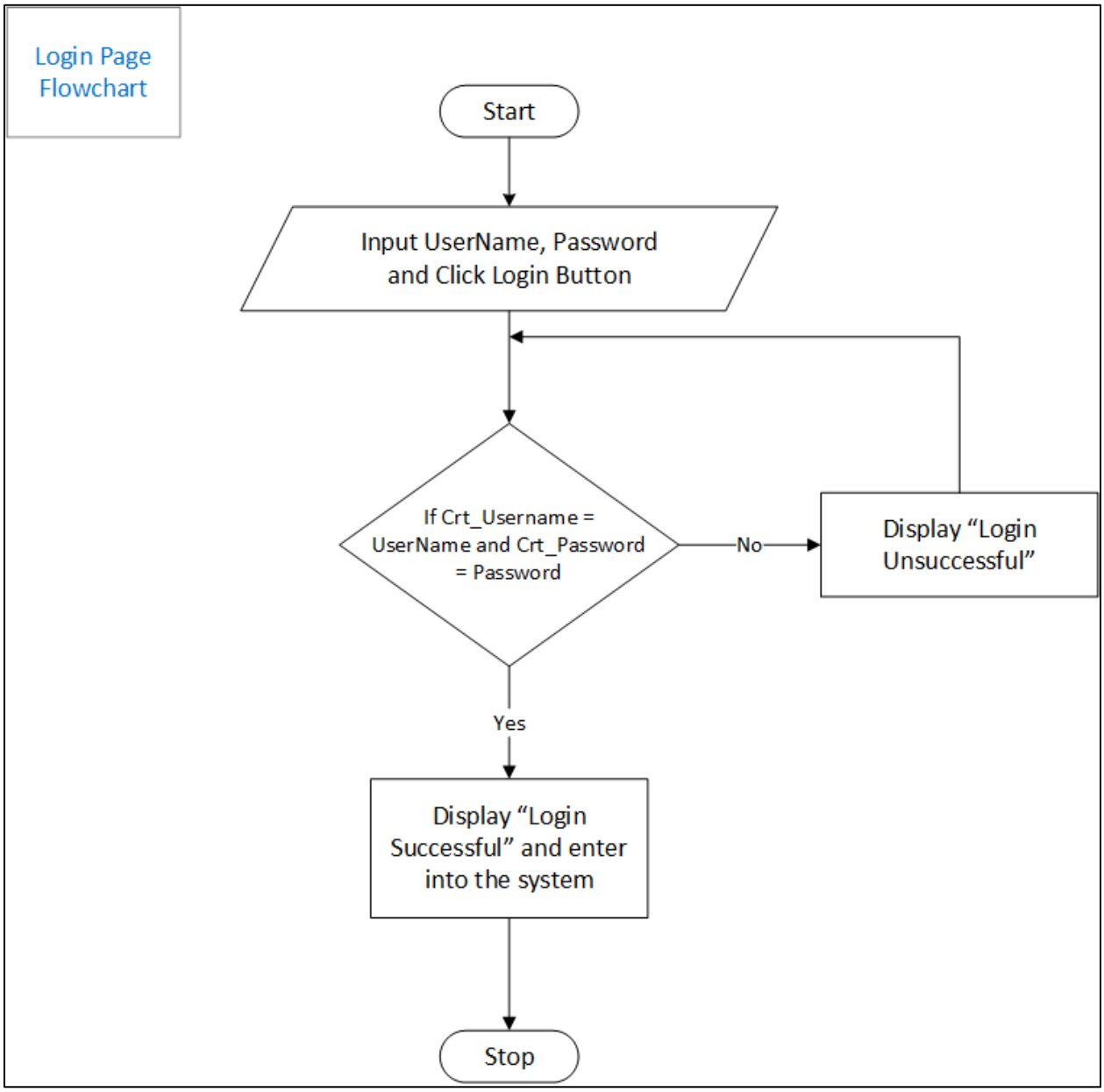
1. **Data Entry Specialist**
    a. Requirement
        i. Access database to enter and update the records
        ii. Ability to search and retrieve records.
2. **Colony Manager**
    a. Requirement
        i. Access Colony and Colonist Data and ability to edit and update it.
        ii. Tools to Generate Required Reports.
        iii. Ability to assign Job and Colony to Colonist.
3. **System Administrators**
    a. Requirement
        i. Access admin level controls for system monitoring
        ii. Tools to generate reports(Jet Report, Trip Report, Colonist Report, Colony Report and Job Report)
        iii. Ability to manage user accounts and permissions
4. **Pilots**
    a. Requirement
        i. View trip and passenger details.
        ii. Search function to find specific trips.
5. **CEO of E-Space Solutions**
    a. Requirements
        i. To view all the activities happening in Both Mars and Earth in Mars Colonization Project
        ii. Tools to Generate all the Reports.
        iii. To View all the Details of Colonist, Dependents and other Employee working in Mars Colonization Project

These're the major user's requirement and the database will be made to fulfil the requirement.

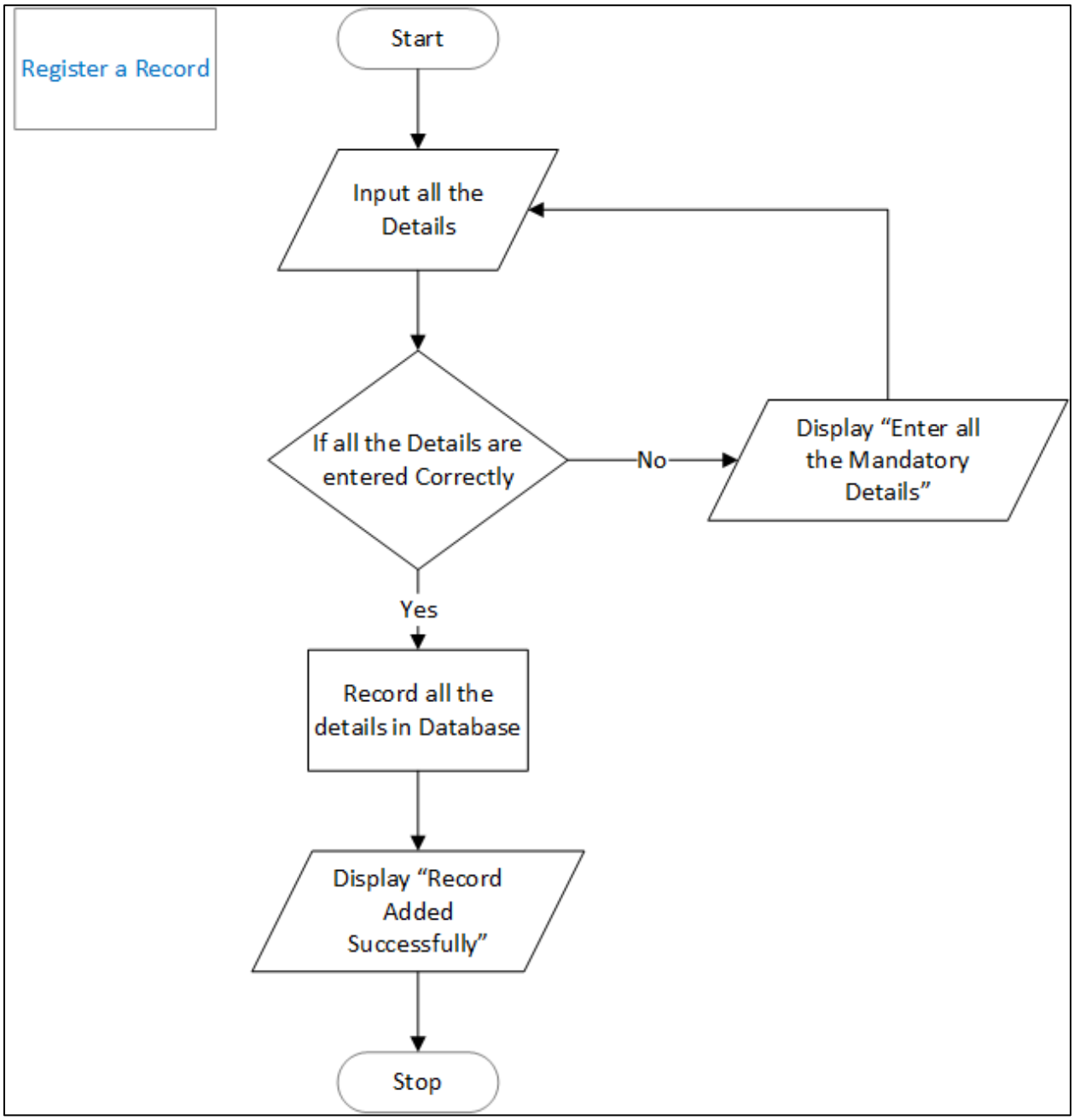
Flow charts

In this project there may be many entities to manage but basically there are only 4 functionalities to manage all the entities [they are register, Update, Delete and Search (this search function flow chart is also used to show data in reports also)] and login page, so under this topic I will be showing the illustrative diagram of flowcharts.

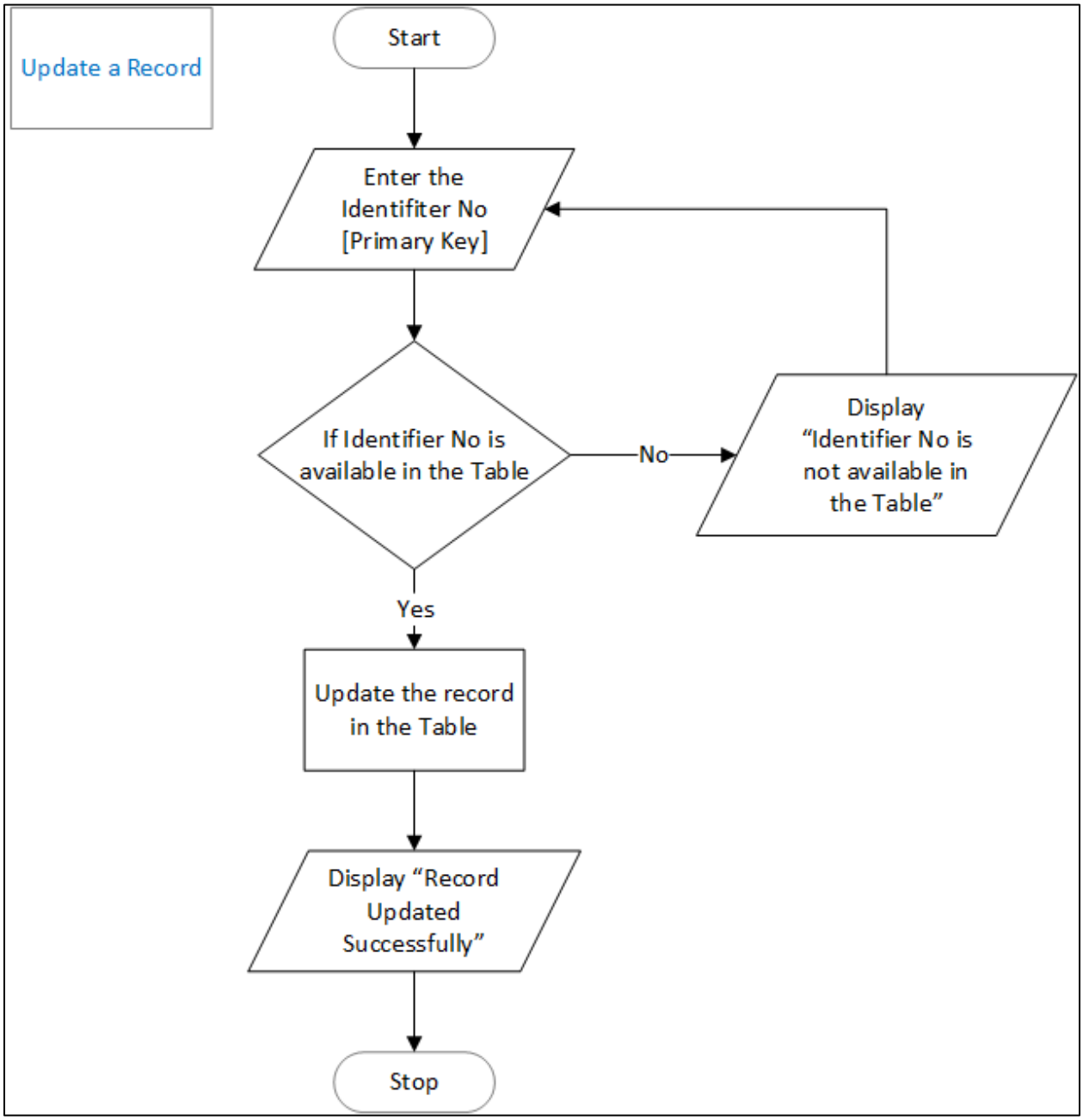
Login Page Flowchart



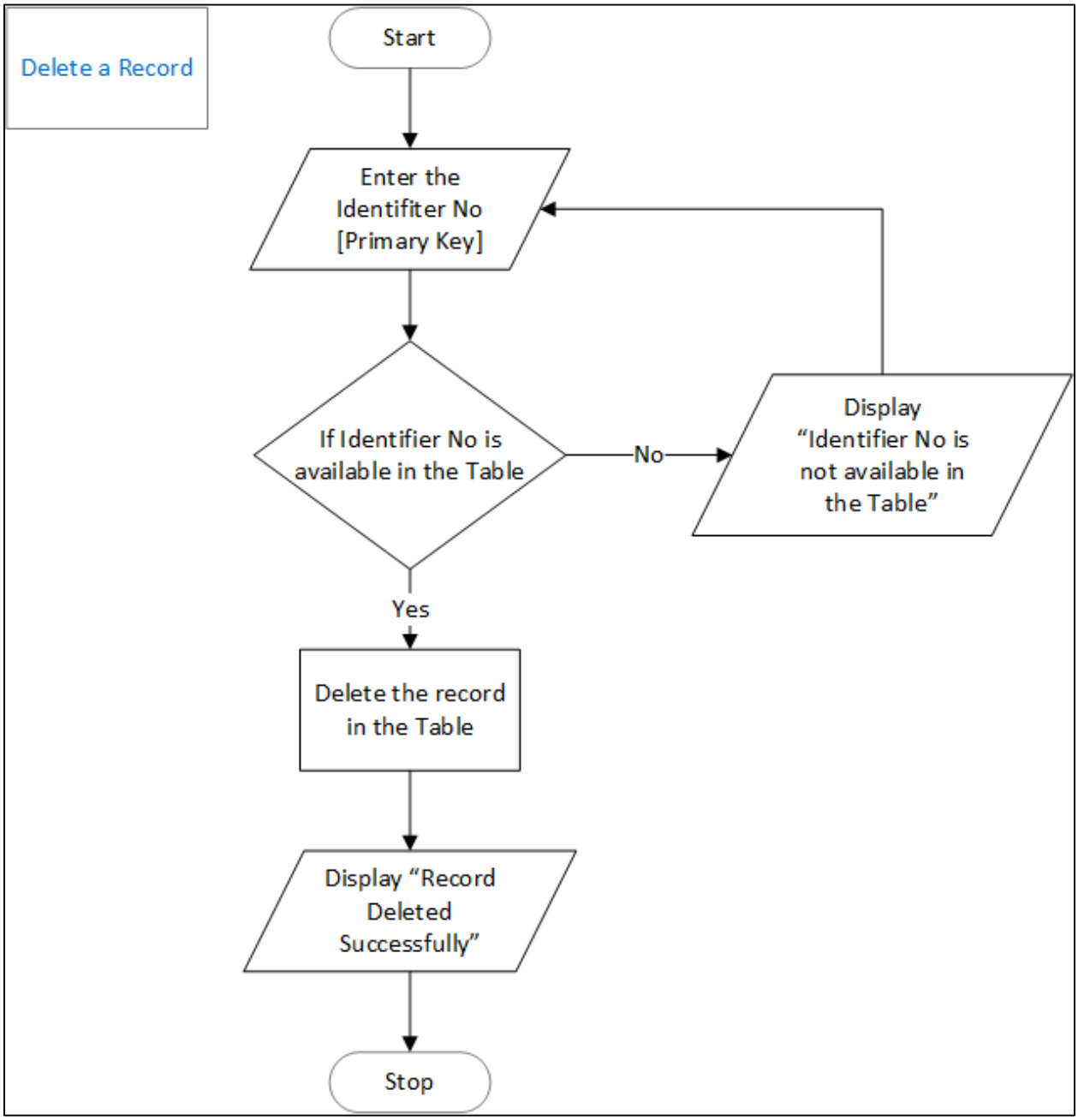
Registration Flowchart



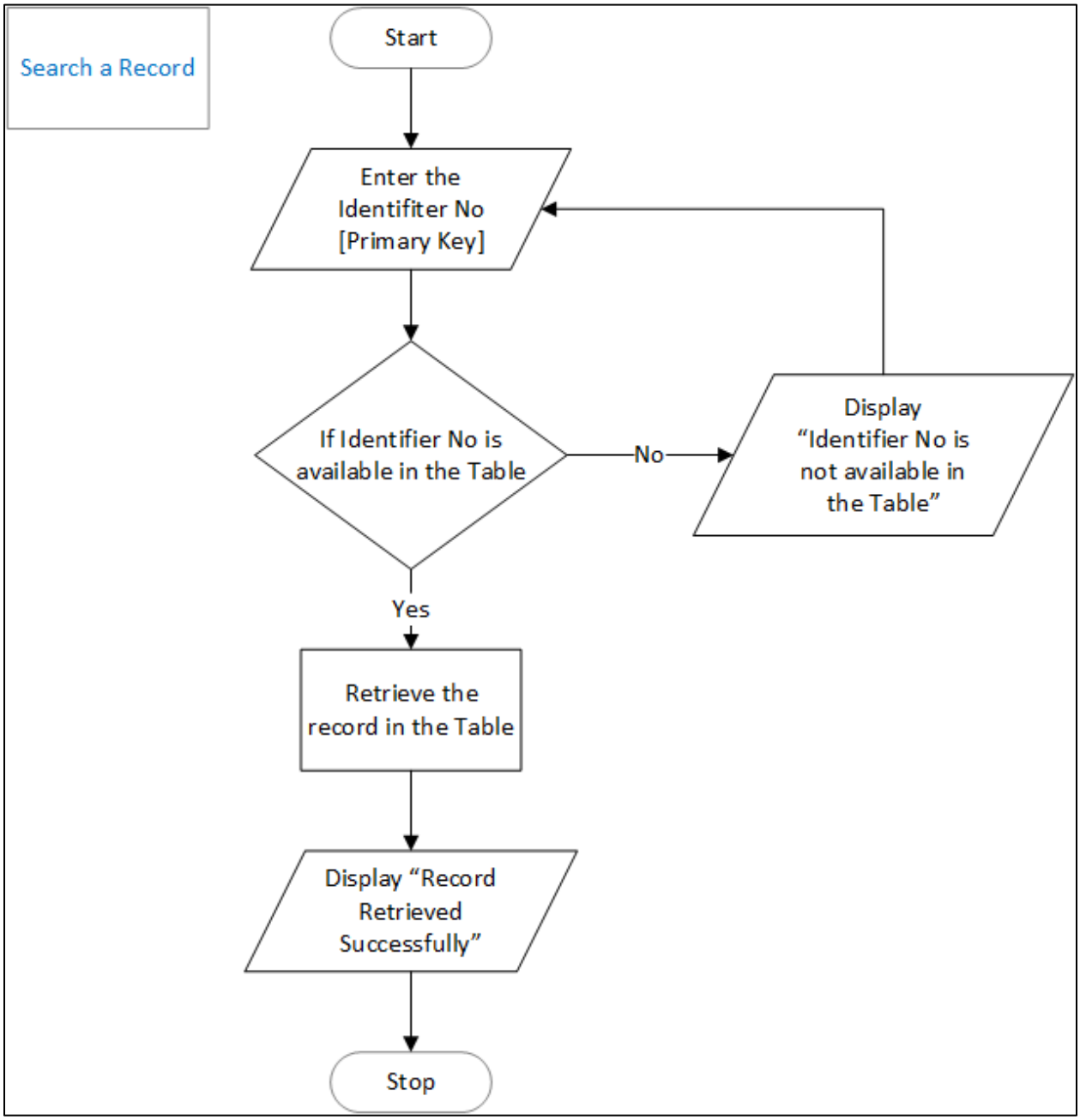
Update Record Flowchart



Delete Record Flowchart



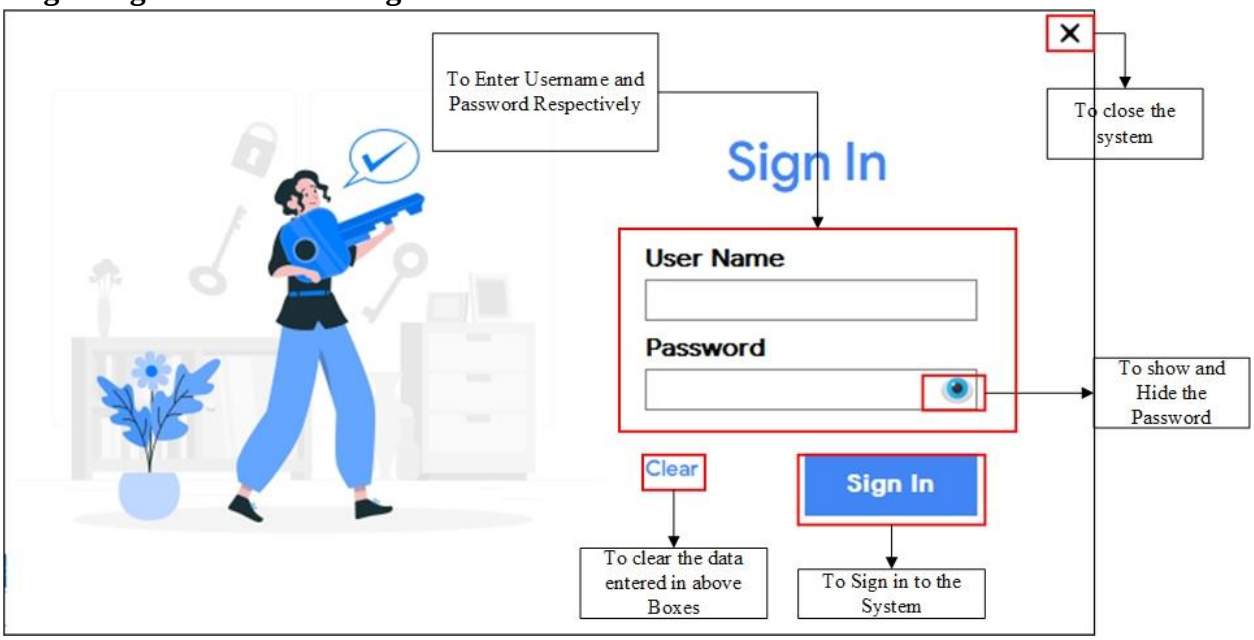
Search Record Flowchart



Illustrative Diagram and Respective Codes

Login Page

Login Page: Illustrative Diagram



Login Page: Code

```
1 reference
private void SignIn_Button_Click(object sender, EventArgs e)
{
    string Entered_Display_Name = UserName_TextBox.Text;
    string Entered_Password = Password_Textbox.Text;
    string connectionString = @"Data Source=Code-Maestro; Initial Catalog = Student; Integrated Security=True;";
    SqlConnection Connect = new SqlConnection(connectionString);
    Connect.Open();
    string search_query = "Select * from employee where  Display_Name = '" + Entered_Display_Name + "' and Password = '" + Entered_Password + "'
    SqlCommand Fetch = new SqlCommand(search_query, Connect);
    SqlDataReader row = Fetch.ExecuteReader();

    if (row.HasRows)
    {
        //To send user name to DashBoard_Form
        UserName = UserName_TextBox.Text;
        DateTime CTime = DateTime.Now;
        Time = CTime.ToString("HH:mm:ss");
        Hour = CTime.ToString("HH");
        this.Hide();
        DashBoard_Form Main_Page = new DashBoard_Form();
        Main_Page.Show();
        //
    }
    else
    {
        MessageBox.Show("Invalid Login Credentials, \nPlease Try Again", "Invalid Login Details", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

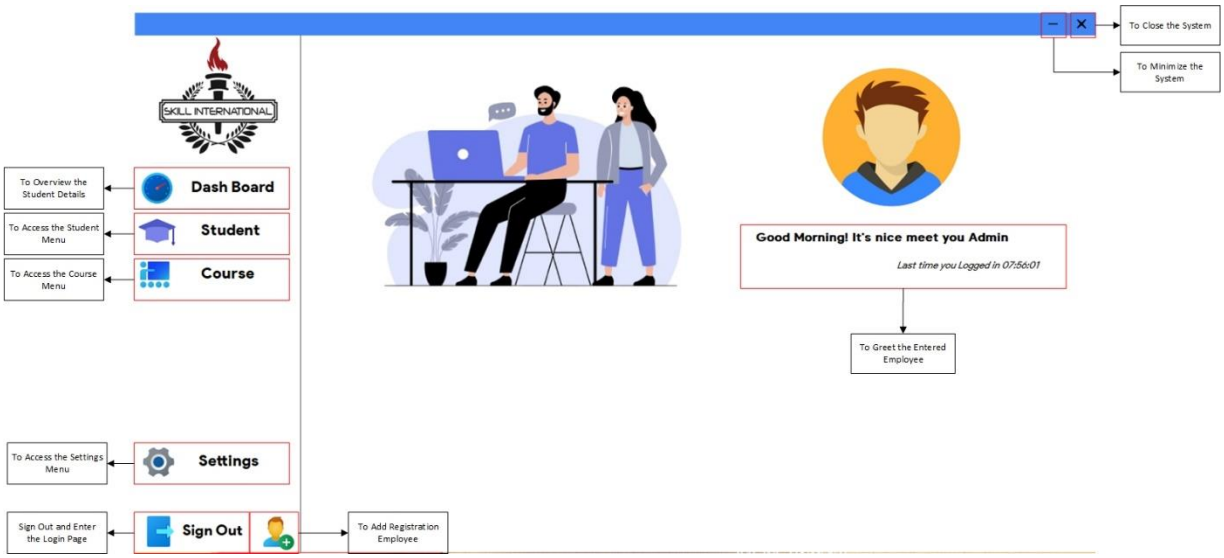
```
1 reference
private void Clear_Label_Click(object sender, EventArgs e)
{
    UserName_TextBox.Clear();
    Password_Textbox.Clear();
    UserName_TextBox.Focus();
}
```

```
1 reference
private void Close_Button_Click(object sender, EventArgs e)
{
    var result = MessageBox.Show("Are you sure, Do you really want to exit....?", "Exit", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
    if (result == DialogResult.Yes)
    {
        Application.Exit();
    }
}
```

```
1 reference
private void Hide_Button_Click(object sender, EventArgs e)
{
    Password_Textbox.UseSystemPasswordChar = true;
    Show_Button.BringToFront();
}

1 reference
private void Show_Button_Click(object sender, EventArgs e)
{
    Password_Textbox.UseSystemPasswordChar = false;
    Hide_Button.BringToFront();
}
```

Dashboard Page

Dashboard Page: Illustrative Diagram



Dashboard Page: Code

```
1 reference
private void Dashoard_Button_Click(object sender, EventArgs e)
{
    this.Form_Loader.Controls.Clear();
    DashBoard_Profile_Form DashBoard_Profile_Form_Vrb = new DashBoard_Profile_Form() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    DashBoard_Profile_Form_Vrb.FormBorderStyle = FormBorderStyle.None;
    this.Form_Loader.Controls.Add(DashBoard_Profile_Form_Vrb);
    DashBoard_Profile_Form_Vrb.Show();
}
```

```
1 reference
private void Management_Menu_Button_Click(object sender, EventArgs e)
{
    this.Form_Loader.Controls.Clear();
    Management_Form Management_Form_Vrb = new Management_Form() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Management_Form_Vrb.FormBorderStyle = FormBorderStyle.None;
    this.Form_Loader.Controls.Add(Management_Form_Vrb);
    Management_Form_Vrb.Show();

}
```

```
1 reference
private void Minimize_Button_Click(object sender, EventArgs e)
{
    WindowState = FormWindowState.Minimized;
}
```

```
1 reference
private void SignOut_Button_Click(object sender, EventArgs e)
{
    Login_Form SignUp_Page = new Login_Form();

    var result = MessageBox.Show("Are you sure, Do you really need to Sign Out....?", "Sign Out", MessageBoxButtons.YesNo, MessageBoxIcon.Questi
    if (result == DialogResult.Yes)
    {
        this.Close();
        SignUp_Page.Show();
    }

}
```

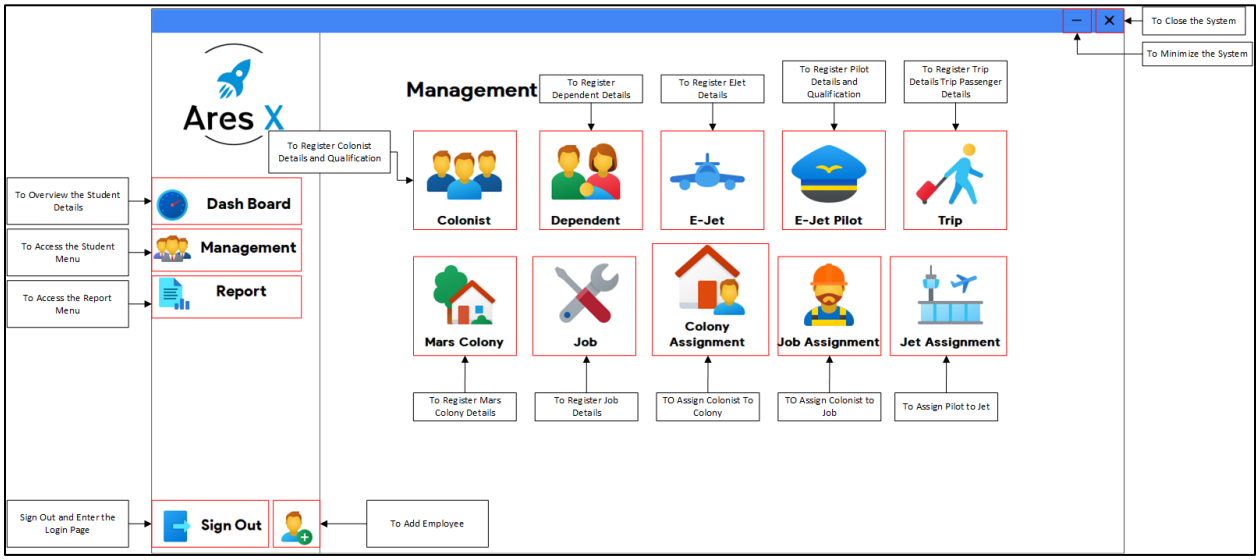
```
1 reference
private void Report_Menu_Button_Click(object sender, EventArgs e)
{
    this.Form_Loader.Controls.Clear();
    Report_Form Report_Form_Vrb = new Report_Form() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Report_Form_Vrb.FormBorderStyle = FormBorderStyle.None;
    this.Form_Loader.Controls.Add(Report_Form_Vrb);
    Report_Form_Vrb.Show();

}
```

```
1 reference
private void SignOut_Button_Click(object sender, EventArgs e)
{
    Login_Form SignUp_Page = new Login_Form();

    var result = MessageBox.Show("Are you sure, Do you really need to Sign Out....?", "Sign Out", MessageBoxButtons.YesNo, MessageBoxIcon.Questi
    if (result == DialogResult.Yes)
    {
        this.Close();
        SignUp_Page.Show();
    }

}
```

```
1 reference
private void AddMember_Button_Click(object sender, EventArgs e)
{
    AddMember_Form AddMember = new AddMember_Form();
    AddMember.Show();

}
```

Management Page

Management Page: Illustrative Diagram



Management Page: Code

```
1 reference
private void Colonist_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    Colonist_Management Colonist_Management_New = new Colonist_Management() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Colonist_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(Colonist_Management_New);
    Colonist_Management_New.Show();

}
```

```
1 reference
private void Colonist_Job_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    Colonist_Job_Management Colonist_Job_Management_New = new Colonist_Job_Management() { Dock = DockStyle.Fill, TopLevel = false, TopMost = t
    Colonist_Job_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(Colonist_Job_Management_New);
    Colonist_Job_Management_New.Show();

}
```

```
1 reference
private void Colony_Assignment_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    House_Assignment_Management House_Assignment_Management_New = new House_Assignment_Management() { Dock = DockStyle.Fill, TopLevel = false,
    House_Assignment_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(House_Assignment_Management_New);
    House_Assignment_Management_New.Show();


}
```

```
1 reference
private void Dependent_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    Dependent_Management Dependent_Management_New = new Dependent_Management() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Dependent_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(Dependent_Management_New);
    Dependent_Management_New.Show();

}
```

```
1 reference
private void EJet_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    EJet_Management EJet_Management_New = new EJet_Management() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    EJet_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(EJet_Management_New);
    EJet_Management_New.Show();

}
```

```
1 reference
private void Jet_Assignment_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    Jet_Assignment_Management Jet_Assignment_Management_New = new Jet_Assignment_Management() { Dock = DockStyle.Fill, TopLevel = false, TopMo
    Jet_Assignment_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(Jet_Assignment_Management_New);
    Jet_Assignment_Management_New.Show();

}
```

```
1 reference
private void Job_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    Job_Management Job_Management_New = new Job_Management() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Job_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(Job_Management_New);
    Job_Management_New.Show();

}
```

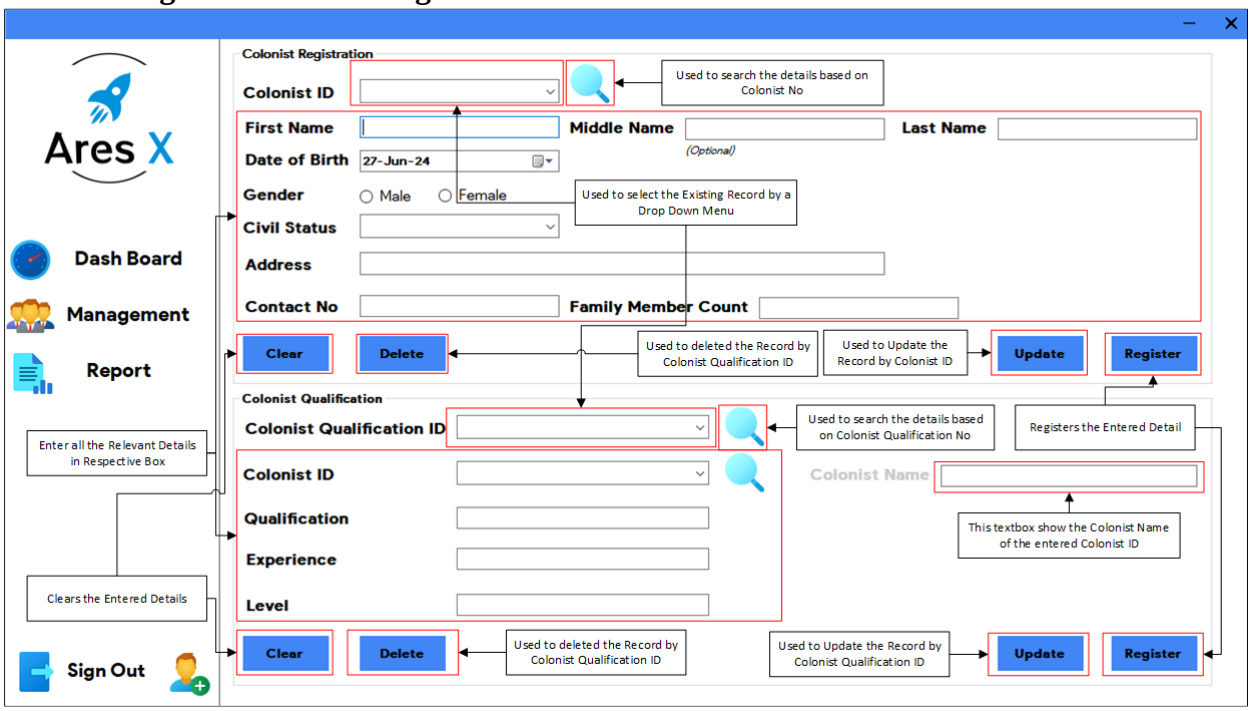
```
1 reference
private void Mars_Colony_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    Mars_Colony_Management Mars_Colony_Management_New = new Mars_Colony_Management() { Dock = DockStyle.Fill, TopLevel = false, TopMost
    Mars_Colony_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(Mars_Colony_Management_New);
    Mars_Colony_Management_New.Show();

}
```

```
1 reference
private void EJet_Pilot_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    EJet_Pilot_Management EJet_Pilot_Management_New = new EJet_Pilot_Management() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true }
    EJet_Pilot_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(EJet_Pilot_Management_New);
    EJet_Pilot_Management_New.Show();

}
```

```
1 reference
private void Trip_Button_Click(object sender, EventArgs e)
{
    Colonist_Button.Visible = false;
    Dependent_Button.Visible = false;
    EJet_Button.Visible = false;
    EJet_Pilot_Button.Visible = false;
    Trip_Button.Visible = false;
    Mars_Colony_Button.Visible = false;
    Job_Button.Visible = false;
    Colonist_Job_Button.Visible = false;
    Jet_Assignment_Button.Visible = false;
    Colony_Assignment_Button.Visible = false;

    this.Management_Form_Panel.Controls.Clear();
    Trip_Management Trip_Management_New = new Trip_Management() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Trip_Management_New.FormBorderStyle = FormBorderStyle.None;
    this.Management_Form_Panel.Controls.Add(Trip_Management_New);
    Trip_Management_New.Show();

}
```

Colonist Page

Colonist Page: Illustrative Diagram



Colonist Page: Code

```
1 reference
private void ColoReg_Clear_Button_Click(object sender, EventArgs e)
{
    ColonistID_Reg_ComboBox.Text = "";
    FName_TextBox.Clear();
    MName_TextBox.Clear();
    LName_TextBox.Clear();
    DateTime This_Day = DateTime.Today;
    DOB_Picker.Text = This_Day.ToString();
    Male_RBtn.Checked = false;
    Female_RBtn.Checked = false;
    CStatus_ComboBox.Text = "";
    Address_TextBox.Clear();
    ContactNo_Textbox.Clear();
    Family_Member_Count_TextBox.Clear();
    FName_TextBox.Focus();
}
```

```
1 reference
private void ColoRegDelete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Questi
        if (result == DialogResult.Yes)
        {
            string Colonist_ID = ColonistID_Reg_ComboBox.Text;
            if (string.IsNullOrEmpty(Colonist_ID))
            {
                MessageBox.Show("Enter a Colonist ID to Delete the record", "Enter Colonist ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamatic
            }
            else
            {
                string Colonist_Delete_Query = "DELETE from Colonist WHERE Colonist_ID = " + Colonist_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Colonist_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Have Been Deleted", "Colonist Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void Colonist_Management_Load(object sender, EventArgs e)
{
    try
    {
        FName_TextBox.Focus();
        con.Open();
        SqlCommand command = new SqlCommand("Select (Colonist_ID)from Colonist", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("Colonist_ID", typeof(int));
        table.Load(reader);
        ColonistID_Reg_ComboBox.ValueMember = "Colonist_ID";
        ColonistID_Reg_ComboBox.DataSource = table;
        con.Close();
        ColonistID_Reg_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command3 = new SqlCommand("Select (Colonist_ID)from Colonist", con);
        SqlDataReader reader3;
        reader3 = command3.ExecuteReader();
        DataTable table3 = new DataTable();
        table3.Columns.Add("Colonist_ID", typeof(int));
        table3.Load(reader3);
        ColonistID_Qua_ComboBox.ValueMember = "Colonist_ID";
```

```
        reader3 = command3.ExecuteReader();
        DataTable table3 = new DataTable();
        table3.Columns.Add("Colonist_ID", typeof(int));
        table3.Load(reader3);
        ColonistID_Qua_ComboBox.ValueMember = "Colonist_ID";
        ColonistID_Qua_ComboBox.DataSource = table3;
        con.Close();
        ColonistID_Qua_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command2 = new SqlCommand("Select (Colonist_Qualification_ID)from Colonist_Qualification", con);
        SqlDataReader reader2;
        reader2 = command2.ExecuteReader();
        DataTable table2 = new DataTable();
        table2.Columns.Add("Colonist_Qualification_ID", typeof(int));
        table2.Load(reader2);
        ColonistID_QuaID_ComboBox.ValueMember = "Colonist_Qualification_ID";
        ColonistID_QuaID_ComboBox.DataSource = table2;
        con.Close();
        ColonistID_QuaID_ComboBox.Text = "";

    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void ColoReg_Register_Button_Click(object sender, EventArgs e)
{
try {
    string Colonist_ID = ColonistID_Reg_ComboBox.Text;
    string First_Name = FName_TextBox.Text;
    string Middle_Name = MName_TextBox.Text;
    string Last_Name = LName_TextBox.Text;
    DOB_Picker.Format = DateTimePickerFormat.Custom;
    DOB_Picker.CustomFormat = "yyyy/MM/dd";
    string Date_Of_Birth = DOB_Picker.Text;
    string Gender;
    if (Male_RBtn.Checked)
    {
        Gender = "Male";
    }
    else if (Female_RBtn.Checked)
    {
        Gender = "Female";
    }
    else
    {
        Gender = "";
    }
    string Civil_Status = CStatus_ComboBox.Text;
    string Earth_Address = Address_TextBox.Text;
    string Contact_No = ContactNo_Textbox.Text;
    string Family_Members_Count = Family_Member_Count_TextBox.Text;
```

```
if (string.IsNullOrEmpty(Colonist_ID))
{
    if (First_Name == "" || Last_Name == "" || Date_Of_Birth == "" || Civil_Status == "" || Gender == "" || Earth_Address == "" || Contact_No
    {
        MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    else
    {
        string ColReg_Query_insert_1 = "INSERT INTO Colonist (First_Name, Middle_Name, Last_Name, Date_Of_Birth, Gender, Earth_Address, Contac
            "VALUES('" + First_Name + "','" + Middle_Name + "','" + Last_Name + "','" + Date_Of_Birth + "','" + Gender + "','" + Earth_Addres
        con.Open();
        SqlCommand Register = new SqlCommand(ColReg_Query_insert_1, con);
        int colonist_ID = Convert.ToInt32(Register.ExecuteScalar());
        con.Close();

        ColonistID_Reg_ComboBox.Text = colonist_ID.ToString();
        string Col_ID = ColonistID_Reg_ComboBox.Text;
        MessageBox.Show("Record Added Successfully \nRegistration No: " + Col_ID, "Register Student", MessageBoxButtons.OK, MessageBoxIcon.Inf

    }
}
```

```
else
{
    con.Open();
    // Enable IDENTITY_INSERT
    string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Colonist ON;";
    using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
    {
        enableIdentityInsertCommand.ExecuteNonQuery();
    }
    //----------------------------------------
    if (First_Name == "" || Last_Name == "" || Date_Of_Birth == "" || Civil_Status == "" || Gender == "" || Earth_Address == "" || Contact_No
    {
        MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    else
    {
        string ColReg_Query_insert_2 = "Insert into Colonist (Colonist_ID ,First_Name, Middle_Name, Last_Name, Date_Of_Birth, Gender, Earth_Ad
            "Values('"+ Colonist_ID +"','" + First_Name + "','" + Middle_Name + "','" + Last_Name + "','" + Date_Of_Birth + "','" + Gender +

        SqlCommand Register2 = new SqlCommand(ColReg_Query_insert_2, con);
        Register2.ExecuteNonQuery();

        MessageBox.Show("Record Added SucessFully", "Register Colonist", MessageBoxButtons.OK, MessageBoxIcon.Information);

    }


    //-------------------------------------
    // Disable IDENTITY_INSERT
```

```
    //-------------------------------------
    // Disable IDENTITY_INSERT
    string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Colonist OFF;";
    using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
    {
        disableIdentityInsertCommand.ExecuteNonQuery();
    }
    con.Close();
}


catch (SqlException ex)
{
    string message = "Insert Error:";
     message += ex.Message;
    MessageBox.Show(message);
    con.Close();
}
```

```
1 reference
private void ColReg_Search_Button_Click(object sender, EventArgs e)
{
    string Colonist_ID = ColonistID_Reg_ComboBox.Text;

    if (string.IsNullOrEmpty(Colonist_ID))
    {
        MessageBox.Show("Please enter a Colonist ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "
    {
        connection.Open();

        string query = "SELECT * FROM Colonist WHERE Colonist_ID = @Colonist_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Colonist_ID", Colonist_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    ColonistID_Reg_ComboBox.Text = reader["Colonist_ID"].ToString();
                    FName_TextBox.Text = reader["First_Name"].ToString();
                    MName_TextBox.Text = reader["Middle_Name"].ToString();
                    LName_TextBox.Text = reader["Last_Name"].ToString();
                    DOB_Picker.Format = DateTimePickerFormat.Custom;
                    DOB_Picker.CustomFormat = "yyyy/MM/dd";
```

```
                    ColonistID_Reg_ComboBox.Text = reader["Colonist_ID"].ToString();
                    FName_TextBox.Text = reader["First_Name"].ToString();
                    MName_TextBox.Text = reader["Middle_Name"].ToString();
                    LName_TextBox.Text = reader["Last_Name"].ToString();
                    DOB_Picker.Format = DateTimePickerFormat.Custom;
                    DOB_Picker.CustomFormat = "yyyy/MM/dd";
                    DOB_Picker.Text = reader["Date_Of_Birth"].ToString();
                    if (reader["Gender"].ToString() == "Male")
                    {
                        Male_RBtn.Checked = true;
                    }
                    else
                    {
                        Female_RBtn.Checked = true;
                    }

                    Address_TextBox.Text = reader["Earth_Address"].ToString();
                    CStatus_ComboBox.Text = reader["Civil_Status"].ToString();
                    ContactNo_Textbox.Text = reader["Contact_No"].ToString();
                    Family_Member_Count_TextBox.Text = reader["Family_Members_Count"].ToString();


                }
                else
                {
                    MessageBox.Show("No record found for the given Colonist ID.");
                }
            }
        }
    }
```

```
1 reference
private void ColoReg_Update_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Colonist_ID = ColonistID_Reg_ComboBox.Text;
        string First_Name = FName_TextBox.Text;
        string Middle_Name = MName_TextBox.Text;
        string Last_Name = LName_TextBox.Text;
        DOB_Picker.Format = DateTimePickerFormat.Custom;
        DOB_Picker.CustomFormat = "yyyy/MM/dd";
        string Date_Of_Birth = DOB_Picker.Text;
        string Gender;
        if (Male_RBtn.Checked)
        {
            Gender = "Male";
        }
        else if (Female_RBtn.Checked)
        {
            Gender = "Female";
        }
        else
        {
            Gender = "";
        }
        string Civil_Status = CStatus_ComboBox.Text;
        string Earth_Address = Address_TextBox.Text;
        string Contact_No = ContactNo_Textbox.Text;
        string Family_Members_Count = Family_Member_Count_TextBox.Text;
```

```
string Colonist_Query_update = "UPDATE Colonist SET " +
      "First_Name = '" + First_Name + "', " +
      "Middle_Name = '" + Middle_Name + "', " +
      "Last_Name = '" + Last_Name + "', " +
      "Date_Of_Birth = '" + Date_Of_Birth + "', " +
      "Gender = '" + Gender + "', " +
      "Earth_Address = '" + Earth_Address + "', " +
      "Contact_No = '" + Contact_No + "', " +
      "Civil_Status = '" + Civil_Status + "', " +
      "Family_Members_Count = '" + Family_Members_Count + "' "+
      "WHERE Colonist_ID = '" + Colonist_ID + "'";
if (string.IsNullOrEmpty(Colonist_ID))
{
    MessageBox.Show("Enter a Colonist ID to update the record", "Enter Colonist ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}
else
{
    if (First_Name == "" || Last_Name == "" || Date_Of_Birth == "" || Civil_Status == "" || Gender == "" || Earth_Address == "" || Contact_No
    {
        MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    else
    {
        con.Open();
        SqlCommand Update = new SqlCommand(Colonist_Query_update, con);
        Update.ExecuteNonQuery();
        con.Close();
        MessageBox.Show("Record Updated SucessFully", "Update Colonist", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}
```

```
          "Civil_Status = '" + Civil_Status + "', " +
          "Family_Members_Count = '" + Family_Members_Count + "' "+
          "WHERE Colonist_ID = '" + Colonist_ID + "'";
    if (string.IsNullOrEmpty(Colonist_ID))
    {
        MessageBox.Show("Enter a Colonist ID to update the record", "Enter Colonist ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    else
    {
        if (First_Name == "" || Last_Name == "" || Date_Of_Birth == "" || Civil_Status == "" || Gender == "" || Earth_Address == "" || Conta
        {
            MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            con.Open();
            SqlCommand Update = new SqlCommand(Colonist_Query_update, con);
            Update.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("Record Updated SucessFully", "Update Colonist", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
}
catch (SqlException ex)
{
    string message = "Update Error:";
    message += ex.Message;
    MessageBox.Show(message);
    con.Close();
}
```

```
//Colonist_Qualification

1 reference
private void ColoQua_Clear_Button_Click(object sender, EventArgs e)
{
    ColonistID_QuaID_ComboBox.Text = "";
    ColonistID_Qua_ComboBox.Text = "";
    CName_Qualification_TextBox.Clear();
    CName_Experience_TextBox.Clear();
    CName_Level_TextBox.Clear();
    CName_Qua_Name_TextBox.Clear();
    ColonistID_Qua_ComboBox.Focus();
}
```

```
1 reference
private void ColoQua_Delete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Colonist_Qualification_ID = ColonistID_QuaID_ComboBox.Text;
            if (string.IsNullOrEmpty(Colonist_Qualification_ID))
            {
                MessageBox.Show("Enter a Colonist Qualification ID to Delete the record", "Enter Colonist Qualification ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Colonist_Qualification_Delete_Query = "DELETE from Colonist_Qualification WHERE Colonist_Qualification_ID = " + Colonist_Qualification_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Colonist_Qualification_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Have Been Deleted", "Colonist Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

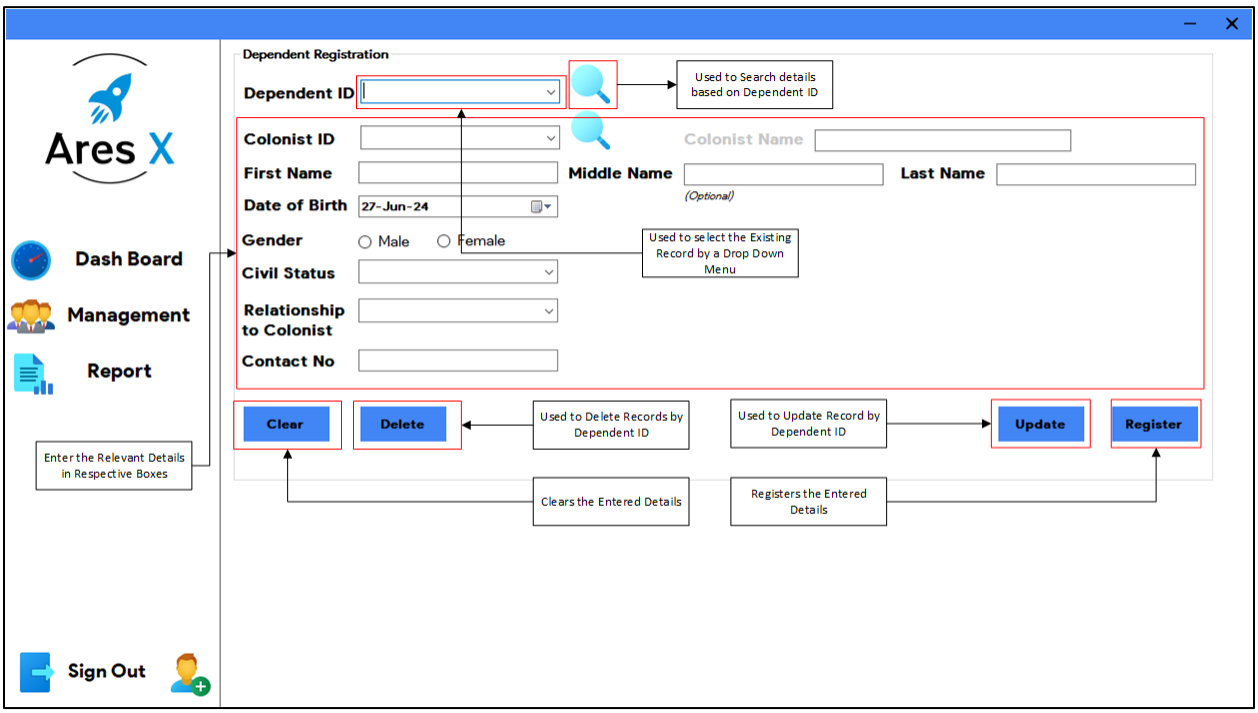
```
1 reference
private void ColoQua_Register_Button_Click(object sender, EventArgs e)
{
    try {
        string Colonist_Qualification_ID = ColonistID_QuaID_ComboBox.Text;
        string Qua_Colonist_ID = ColonistID_Qua_ComboBox.Text;
        string Colonist_Qualification = CName_Qualification_TextBox.Text;
        string Colonist_Experience = CName_Experience_TextBox.Text;
        string Colonist_Level = CName_Level_TextBox.Text;
        string Colonist_Name = CName_Qua_Name_TextBox.Text;

        if (string.IsNullOrEmpty(Colonist_Qualification_ID))
        {
            if (Qua_Colonist_ID == "" || Colonist_Name =="" || Colonist_Qualification == "" || Colonist_Experience == "" || Colonist_Level == "" )
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string ColQuaReg_Query_insert_1 = "INSERT INTO Colonist_Qualification (Colonist_ID, Qualification_Description, Colonist_Experience, Colonist_Level) " +
                    "VALUES('" + Qua_Colonist_ID + "','" + Colonist_Qualification + "','" + Colonist_Experience + "','" + Colonist_Level + "')" + "SELECT SCOPE_IDENTITY();";
                con.Open();
                SqlCommand Register = new SqlCommand(ColQuaReg_Query_insert_1, con);
                int Qua_Colo_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                ColonistID_QuaID_ComboBox.Text = Qua_Colo_ID.ToString();
                string ColQua_ID = ColonistID_QuaID_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + ColQua_ID, "Register Colonist Qualification", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
        }
        else
        {
            con.Open();
            // Enable IDENTITY_INSERT
            string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Colonist_Qualification ON;";
            using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
            {
                enableIdentityInsertCommand.ExecuteNonQuery();
            }
            //------------------------------------------
            if (Qua_Colonist_ID == "" || Colonist_Name == "" || Colonist_Qualification == "" || Colonist_Experience == "" || Colonist_Level == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string ColQuaReg_Query_insert_2 = "Insert into Colonist_Qualification (Colonist_Qualification_ID,Colonist_ID, Qualification_Description, Colonist_Experience, Colonist_Level) " +
                    "Values('"+ Colonist_Qualification_ID + "','" + Qua_Colonist_ID + "','" + Colonist_Qualification + "','" + Colonist_Experience + "','" + Colonist_Level + "')";

                SqlCommand Register2 = new SqlCommand(ColQuaReg_Query_insert_2, con);
                Register2.ExecuteNonQuery();

                MessageBox.Show("Record Added SucessFully", "Register Colonist Qualification", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
```

```
1 reference
private void CQuaID_Search_Button_Click(object sender, EventArgs e)
{
    string Colonist_Qualification_ID = ColonistID_QuaID_ComboBox.Text;

    if (string.IsNullOrEmpty(Colonist_Qualification_ID))
    {
        MessageBox.Show("Please enter a Colonist Qualification ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Colonist_Qualification WHERE Colonist_Qualification_ID = @Colonist_Qualification_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Colonist_Qualification_ID", Colonist_Qualification_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    ColonistID_QuaID_ComboBox.Text = reader["Colonist_Qualification_ID"].ToString();
                    ColonistID_Qua_ComboBox.Text = reader["Colonist_ID"].ToString();
                    CName_Qualification_TextBox.Text = reader["Qualification_Description"].ToString();
                    CName_Experience_TextBox.Text = reader["Colonist_Experience"].ToString();
                    CName_Level_TextBox.Text = reader["Colonist_Level"].ToString();
                    Colonist_CQua_Search_Button.PerformClick();
                }
                else
                {
                    MessageBox.Show("No record found for the given Colonist Qualification ID.");
                }
            }
        }
    }

}
```

```
1 reference
private void ColoQua_Update_Button_Click(object sender, EventArgs e)
{

    try
    {
        string Colonist_Qualification_ID = ColonistID_QuaID_ComboBox.Text;
        string Qua_Colonist_ID = ColonistID_Qua_ComboBox.Text;
        string Colonist_Qualification = CName_Qualification_TextBox.Text;
        string Colonist_Experience = CName_Experience_TextBox.Text;
        string Colonist_Level = CName_Level_TextBox.Text;
        string Colonist_Name = CName_Qua_Name_TextBox.Text;


        string Colonist_Query_update = "UPDATE Colonist_Qualification SET " +
              "Colonist_ID = '" + Qua_Colonist_ID + "', " +
              "Qualification_Description = '" + Colonist_Qualification + "', " +
              "Colonist_Experience = '" + Colonist_Experience + "', " +
              "Colonist_Level = '" + Colonist_Level + "'";


        if (string.IsNullOrEmpty(Colonist_Qualification_ID))
        {
            MessageBox.Show("Enter a Colonist Qualification ID to update the record", "Enter Colonist ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (Qua_Colonist_ID == "" || Colonist_Name == "" || Colonist_Qualification == "" || Colonist_Experience == "" || Colonist_Level == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(Colonist_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();
                //Colonist_CQua_Search_Button.PerformClick();
                MessageBox.Show("Record Updated SucessFully", "Update Colonist Qualification", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Update Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

Dependent Page

Dependent Page: Illustrative Diagram



Dependent Page: Code

```
1 reference
private void DepeReg_Clear_Button_Click(object sender, EventArgs e)
{

    DependentID_Reg_ComboBox.Text = "";
    Dependent_FName_TextBox.Clear();
    Dependent_MName_TextBox.Clear();
    Dependent_LName_TextBox.Clear();
    DateTime This_Day = DateTime.Today;
    Dependent_DOB_Picker.Text = This_Day.ToString();
    Dependent_Male_RBtn.Checked = false;
    Dependent_Female_RBtn.Checked = false;
    Dependent_Relationship_ComboBox.Text = "";
    ColonistID_Reg_ComboBox.Text = "";
    CName_Qua_Name_TextBox.Clear();
    Dependent_CStatus_ComboBox.Text = "";
    Dependent_ContactNo_Textbox.Clear();
    Dependent_FName_TextBox.Focus();
}
```

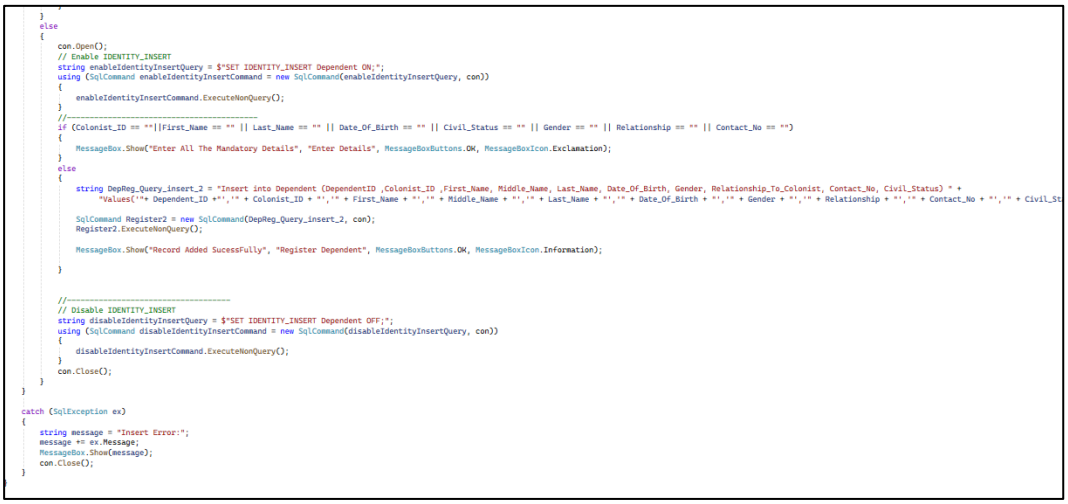
```
1 reference
private void DepRegDelete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Dependent_ID = DependentID_Reg_ComboBox.Text;
            if (string.IsNullOrEmpty(Dependent_ID))
            {
                MessageBox.Show("Enter a Dependent ID to Delete the record", "Enter Dependent ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Colonist_Delete_Query = "DELETE from Dependent WHERE DependentID = " + Dependent_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Colonist_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Have Been Deleted", "Colonist Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
}
```


```
1 reference
private void DepRegDelete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Dependent_ID = DependentID_Reg_ComboBox.Text;
            if (string.IsNullOrEmpty(Dependent_ID))
            {
                MessageBox.Show("Enter a Dependent ID to Delete the record", "Enter Dependent ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Colonist_Delete_Query = "DELETE from Dependent WHERE DependentID = " + Dependent_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Colonist_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Have Been Deleted", "Colonist Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
}
```

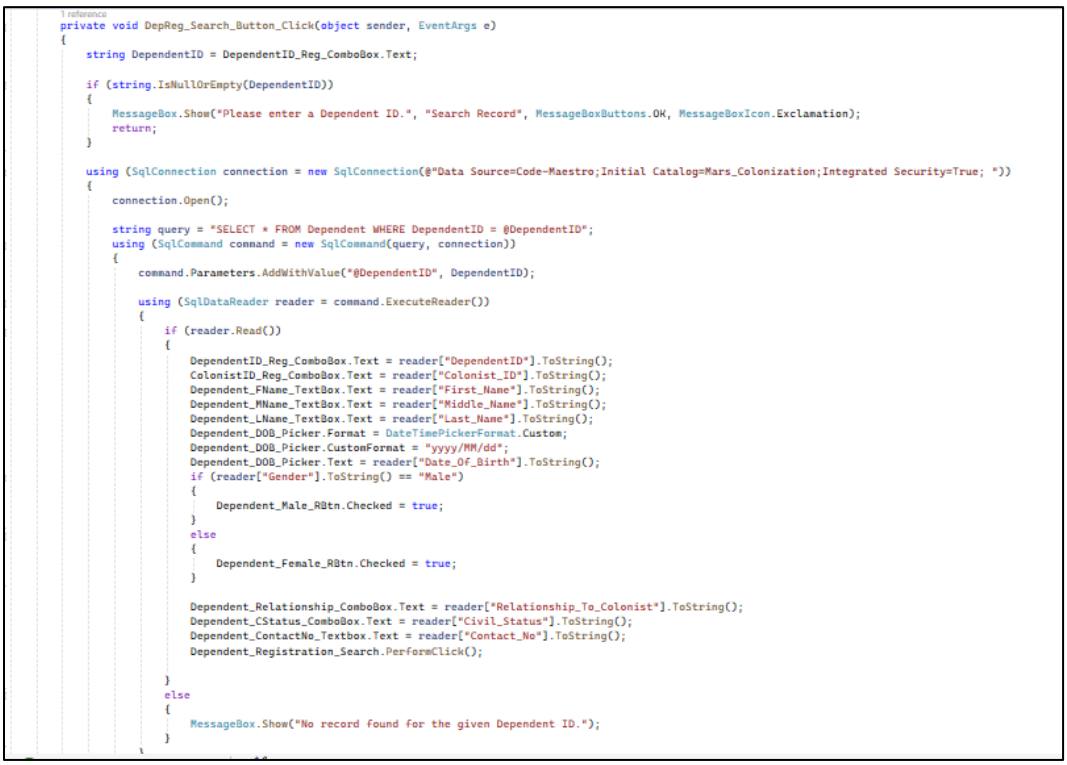
```
1 reference
private void Dependent_Management_Load(object sender, EventArgs e)
{
    try
    {
        con.Open();
        SqlCommand command = new SqlCommand("Select (Colonist_ID)from Colonist", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("Colonist_ID", typeof(int));
        table.Load(reader);
        ColonistID_Reg_ComboBox.ValueMember = "Colonist_ID";
        ColonistID_Reg_ComboBox.DataSource = table;
        con.Close();
        ColonistID_Reg_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command2 = new SqlCommand("Select (DependentID)from Dependent", con);
        SqlDataReader reader2;
        reader2 = command2.ExecuteReader();
        DataTable table2 = new DataTable();
        table2.Columns.Add("DependentID", typeof(int));
        table2.Load(reader2);
        DependentID_Reg_ComboBox.ValueMember = "DependentID";
        DependentID_Reg_ComboBox.DataSource = table2;
        con.Close();
        DependentID_Reg_ComboBox.Text = "";

    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

```
1 reference
private void DepReg_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Dependent_ID = DependentID_Reg_ComboBox.Text;
        string Colonist_ID = ColonistID_Reg_ComboBox.Text;
        string First_Name = Dependent_FName_TextBox.Text;
        string Middle_Name = Dependent_MName_TextBox.Text;
        string Last_Name = Dependent_LName_TextBox.Text;
        Dependent_DOB_Picker.Format = DateTimePickerFormat.Custom;
        Dependent_DOB_Picker.CustomFormat = "yyyy/MM/dd";
        string Date_Of_Birth = Dependent_DOB_Picker.Text;
        string Gender;
        if (Dependent_Male_RBtn.Checked)
        {
            Gender = "Male";
        }
        else if (Dependent_Female_RBtn.Checked)
        {
            Gender = "Female";
        }
        else
        {
            Gender = "";
        }
        string Civil_Status = Dependent_CStatus_ComboBox.Text;
        string Relationship = Dependent_Relationship_ComboBox.Text;
        string Contact_No = Dependent_ContactNo_Textbox.Text;


        if (string.IsNullOrEmpty(Dependent_ID))
        {
            if (Colonist_ID == "" || First_Name == "" || Last_Name == "" || Date_Of_Birth == "" || Civil_Status == "" || Gender == "" || Relationship == "" || Contact_No == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string DepReg_Query_insert_1 = "INSERT INTO Dependent (Colonist_ID, First_Name, Middle_Name, Last_Name, Date_Of_Birth, Gender, Relationship_To_Colonist, Contact_No, Civil_Status) " +
                    "VALUES('"+ Colonist_ID +"','" + First_Name + "','" + Middle_Name + "','" + Last_Name + "','" + Date_Of_Birth + "','" + Gender + "','" + Relationship + "','" + Contact_No + "','" + Civil_Status + "')";
                con.Open();
                SqlCommand Register = new SqlCommand(DepReg_Query_insert_1, con);
                int Dep_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                DependentID_Reg_ComboBox.Text = Dep_ID.ToString();
                string Depe_ID = DependentID_Reg_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + Depe_ID, "Register Dependent", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

```
1 reference
private void DepReg_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Dependent_ID = DependentID_Reg_ComboBox.Text;
        string Colonist_ID = ColonistID_Reg_ComboBox.Text;
        string First_Name = Dependent_FName_TextBox.Text;
        string Middle_Name = Dependent_MName_TextBox.Text;
        string Last_Name = Dependent_LName_TextBox.Text;
        Dependent_DOB_Picker.Format = DateTimePickerFormat.Custom;
        Dependent_DOB_Picker.CustomFormat = "yyyy/MM/dd";
        string Date_Of_Birth = Dependent_DOB_Picker.Text;
        string Gender;
        if (Dependent_Male_RBtn.Checked)
        {
            Gender = "Male";
        }
        else if (Dependent_Female_RBtn.Checked)
        {
            Gender = "Female";
        }
        else
        {
            Gender = "";
        }
        string Civil_Status = Dependent_CStatus_ComboBox.Text;
        string Relationship = Dependent_Relationship_ComboBox.Text;
        string Contact_No = Dependent_ContactNo_Textbox.Text;


        if (string.IsNullOrEmpty(Dependent_ID))
        {
            if (Colonist_ID == "" || First_Name == "" || Last_Name == "" || Date_Of_Birth == "" || Civil_Status == "" || Gender == "" || Relationship == "" || Contact_No == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string DepReg_Query_insert_1 = "INSERT INTO Dependent (Colonist_ID, First_Name, Middle_Name, Last_Name, Date_Of_Birth, Gender, Relationship_To_Colonist, Contact_No, Civil_Status) " +
                    "VALUES('"+ Colonist_ID +"','" + First_Name + "','" + Middle_Name + "','" + Last_Name + "','" + Date_Of_Birth + "','" + Gender + "','" + Relationship + "','" + Contact_No + "','" + Civil_Status + "')";
                con.Open();
                SqlCommand Register = new SqlCommand(DepReg_Query_insert_1, con);
                int Dep_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                DependentID_Reg_ComboBox.Text = Dep_ID.ToString();
                string Depe_ID = DependentID_Reg_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + Depe_ID, "Register Dependent", MessageBoxButtons.OK, MessageBoxIcon.Information);
```



```
    }
    else
    {
        con.Open();
        // Enable IDENTITY_INSERT
        string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Dependent ON;";
        using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
        {
            enableIdentityInsertCommand.ExecuteNonQuery();
        }
        //-----------------------------------------
        if (Colonist_ID == ""||First_Name == "" || Last_Name == "" || Date_Of_Birth == "" || Civil_Status == "" || Gender == "" || Relationship == "" || Contact_No == "")
        {
            MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            string DepReg_Query_insert_2 = "Insert into Dependent (DependentID ,Colonist_ID ,First_Name, Middle_Name, Last_Name, Date_Of_Birth, Gender, Relationship_To_Colonist, Contact_No, Civil_Status) " +
                "Values('"+ Dependent_ID +"','" + Colonist_ID + "','" + First_Name + "','" + Middle_Name + "','" + Last_Name + "','" + Date_Of_Birth + "','" + Gender + "','" + Relationship + "','" + Contact_No + "','" + Civil_St

            SqlCommand Register2 = new SqlCommand(DepReg_Query_insert_2, con);
            Register2.ExecuteNonQuery();

            MessageBox.Show("Record Added SucessFully", "Register Dependent", MessageBoxButtons.OK, MessageBoxIcon.Information);

        }


        //-----------------------------------
        // Disable IDENTITY_INSERT
        string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Dependent OFF;";
        using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
        {
            disableIdentityInsertCommand.ExecuteNonQuery();
        }
        con.Close();
    }
}

catch (SqlException ex)
{
    string message = "Insert Error:";
    message += ex.Message;
    MessageBox.Show(message);
    con.Close();
}
}
```



```
1 reference
private void DepReg_Search_Button_Click(object sender, EventArgs e)
{
    string DependentID = DependentID_Reg_ComboBox.Text;

    if (string.IsNullOrEmpty(DependentID))
    {
        MessageBox.Show("Please enter a Dependent ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Dependent WHERE DependentID = @DependentID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@DependentID", DependentID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    DependentID_Reg_ComboBox.Text = reader["DependentID"].ToString();
                    ColonistID_Reg_ComboBox.Text = reader["Colonist_ID"].ToString();
                    Dependent_FName_TextBox.Text = reader["First_Name"].ToString();
                    Dependent_MName_TextBox.Text = reader["Middle_Name"].ToString();
                    Dependent_LName_TextBox.Text = reader["Last_Name"].ToString();
                    Dependent_DOB_Picker.Format = DateTimePickerFormat.Custom;
                    Dependent_DOB_Picker.CustomFormat = "yyyy/MM/dd";
                    Dependent_DOB_Picker.Text = reader["Date_Of_Birth"].ToString();
                    if (reader["Gender"].ToString() == "Male")
                    {
                        Dependent_Male_RBtn.Checked = true;
                    }
                    else
                    {
                        Dependent_Female_RBtn.Checked = true;
                    }

                    Dependent_Relationship_ComboBox.Text = reader["Relationship_To_Colonist"].ToString();
                    Dependent_CStatus_ComboBox.Text = reader["Civil_Status"].ToString();
                    Dependent_ContactNo_Textbox.Text = reader["Contact_No"].ToString();
                    Dependent_Registration_Search.PerformClick();

                }
                else
                {
                    MessageBox.Show("No record found for the given Dependent ID.");
                }
            }
```

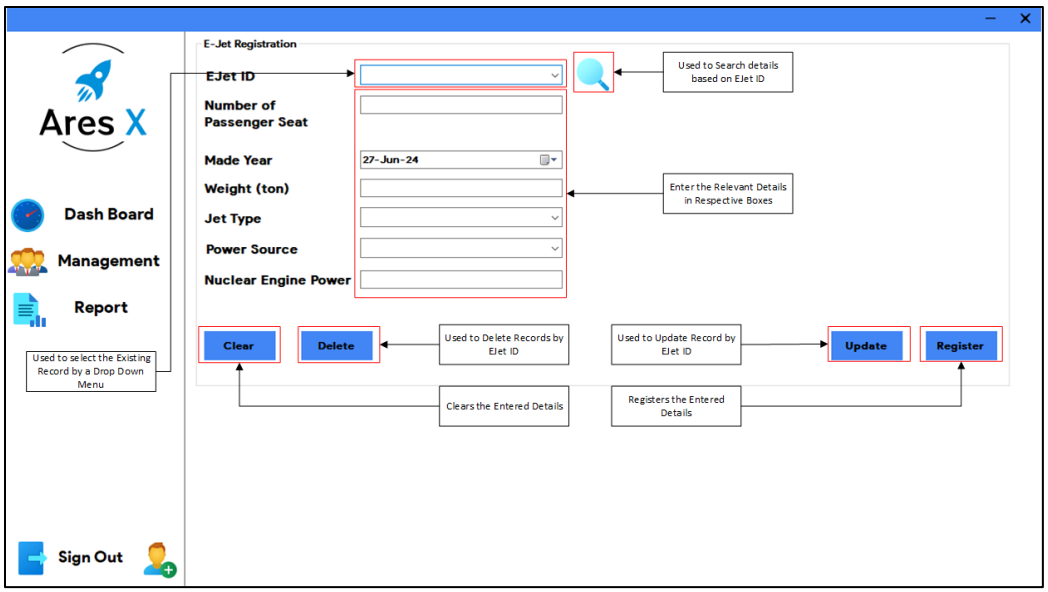
```
1 reference
private void Dependent_Registration_Search_Click(object sender, EventArgs e)
{
    string Colonist_ID = ColonistID_Reg_ComboBox.Text;

    if (string.IsNullOrEmpty(Colonist_ID))
    {
        MessageBox.Show("Please enter a Colonist ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Colonist WHERE Colonist_ID = @Colonist_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Colonist_ID", Colonist_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    CName_Qua_Name_TextBox.Text = reader["First_Name"].ToString();
                }
                else
                {
                    MessageBox.Show("No record found for the given Colonist ID.");
                }
            }
        }
    }

}
```

```
1 reference
private void DepReg_Update_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Dependent_ID = DependentID_Reg_ComboBox.Text;
        string Colonist_ID = ColonistID_Reg_ComboBox.Text;
        string First_Name = Dependent_FName_TextBox.Text;
        string Middle_Name = Dependent_MName_TextBox.Text;
        string Last_Name = Dependent_LName_TextBox.Text;
        Dependent_DOB_Picker.Format = DateTimePickerFormat.Custom;
        Dependent_DOB_Picker.CustomFormat = "yyyy/MM/dd";
        string Date_Of_Birth = Dependent_DOB_Picker.Text;
        string Gender;
        if (Dependent_Male_RBtn.Checked)
        {
            Gender = "Male";
        }
        else if (Dependent_Female_RBtn.Checked)
        {
            Gender = "Female";
        }
        else
        {
            Gender = "";
        }
        string Civil_Status = Dependent_CStatus_ComboBox.Text;
        string Relationship = Dependent_Relationship_ComboBox.Text;
        string Contact_No = Dependent_ContactNo_Textbox.Text;


        string Dependent_Query_update = "UPDATE Dependent SET " +
              "Colonist_ID = '" +Colonist_ID+ "', " +
              "First_Name = '" + First_Name + "', " +
              "Middle_Name = '" + Middle_Name + "', " +
              "Last_Name = '" + Last_Name + "', " +
              "Date_Of_Birth = '" + Date_Of_Birth + "', " +
              "Gender = '" + Gender + "', " +
```

```
        string Dependent_Query_update = "UPDATE Dependent SET " +
              "Colonist_ID = '" +Colonist_ID+ "', " +
              "First_Name = '" + First_Name + "', " +
              "Middle_Name = '" + Middle_Name + "', " +
              "Last_Name = '" + Last_Name + "', " +
              "Date_Of_Birth = '" + Date_Of_Birth + "', " +
              "Gender = '" + Gender + "', " +
              "Relationship_To_Colonist = '" + Relationship + "', " +
              "Contact_No = '" + Contact_No + "', " +
              "Civil_Status = '" + Civil_Status + "'" +
              "WHERE DependentID = '" + Dependent_ID + "'";
        if (string.IsNullOrEmpty(Dependent_ID))
        {
            MessageBox.Show("Enter a Dependent ID to update the record", "Enter Dependent ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (Colonist_ID == "" || First_Name == "" || Last_Name == "" || Date_Of_Birth == "" || Civil_Status == "" || Gender == "" || Relationship == "" || Contact_No == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(Dependent_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Updated SucessFully", "Update Dependent", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Update Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

E-Jet Page

E-Jet Page: Illustrative Diagram



E-Jet Page: Code

```
1 reference
private void EJet_Clear_Button_Click(object sender, EventArgs e)
{
    EJetID_Reg_ComboBox.Text = "";
    EJet_PassSeat_TextBox.Clear();
    DateTime This_Day = DateTime.Today;
    EJet_MadeYear_Picker.Text = This_Day.ToString();
    EJet_Weight_TextBox.Clear();
    Jet_Type_ComboBox.Text = "";
    Power_Source_ComboBox.Text = "";
    Nuclear_Engine_Power_Textbox.Clear();
    EJet_PassSeat_TextBox.Focus();
}
```

```
1 reference
private void EJet_Delete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string EJet_ID = EJetID_Reg_ComboBox.Text;
            if (string.IsNullOrEmpty(EJet_ID))
            {
                MessageBox.Show("Enter a EJet ID to Delete the record", "Enter EJet ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string EJet_Delete_Query = "DELETE from EJet WHERE EJet_ID = " + EJet_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(EJet_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                EJet_Clear_Button.PerformClick();
                MessageBox.Show("Record Have Been Deleted", "EJet Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

```
1 reference
private void EJet_Management_Load(object sender, EventArgs e)
{
    try
    {
        con.Open();
        SqlCommand command = new SqlCommand("Select (EJet_ID) from EJet", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("EJet_ID", typeof(int));
        table.Load(reader);
        EJetID_Reg_ComboBox.ValueMember = "EJet_ID";
        EJetID_Reg_ComboBox.DataSource = table;
        con.Close();
        EJetID_Reg_ComboBox.Text = "";


    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void EJet_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string EJet_ID = EJetID_Reg_ComboBox.Text;
        string Pass_Seat = EJet_PassSeat_TextBox.Text;
        EJet_MadeYear_Picker.Format = DateTimePickerFormat.Custom;
        EJet_MadeYear_Picker.CustomFormat = "yyyy/MM/dd";
        string MadeYear = EJet_MadeYear_Picker.Text;
        string EJet_Weight = EJet_Weight_TextBox.Text;
        string EJet_Type = Jet_Type_ComboBox.Text;
        string Power_Source = Power_Source_ComboBox.Text;
        string Nuclear_Engine = Nuclear_Engine_Power_Textbox.Text;

        if (string.IsNullOrEmpty(EJet_ID))
        {
            if (Pass_Seat == "" || MadeYear == "" || EJet_Weight == "" || EJet_Type == "" || Power_Source == "" || Nuclear_Engine == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string EJetReg_Query_insert_1 = "INSERT INTO EJet (Passenger_Seats, EJet_Type, Made_Year, Weight, Nuclear_Engine_Power, Power_Source) " +
                    "VALUES('" + Pass_Seat + "','" + EJet_Type + "','" + MadeYear + "','" + EJet_Weight + "','" + Nuclear_Engine + "','" + Power_Source + "')" +
                con.Open();
                SqlCommand Register = new SqlCommand(EJetReg_Query_insert_1, con);
                int ejet_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                EJetID_Reg_ComboBox.Text = ejet_ID.ToString();
                string jet_ID = EJetID_Reg_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + jet_ID, "Register EJet", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
```

```
else
{
    con.Open();
    // Enable IDENTITY_INSERT
    string enableIdentityInsertQuery = $"SET IDENTITY_INSERT EJet ON;";
    using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
    {
        enableIdentityInsertCommand.ExecuteNonQuery();
    }
    //-----------------------------------------
    if (Pass_Seat == "" || MadeYear == "" || EJet_Weight == "" || EJet_Type == "" || Power_Source == "" || Nuclear_Engine == "")
    {
        MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    else
    {
        string EJetReg_Query_insert_2 = "Insert into EJet (EJet_ID ,Passenger_Seats, EJet_Type, Made_Year, Weight, Nuclear_Engine_Power, Power_Source) " +
            "Values('" + EJet_ID + "','" + Pass_Seat + "','" + EJet_Type + "','" + MadeYear + "','" + EJet_Weight + "','" + Nuclear_Engine + "','" + Power_Source + "')";

        SqlCommand Register2 = new SqlCommand(EJetReg_Query_insert_2, con);
        Register2.ExecuteNonQuery();

        MessageBox.Show("Record Added SucessFully", "Register EJet", MessageBoxButtons.OK, MessageBoxIcon.Information);

    }


    //-------------------------------------
    // Disable IDENTITY_INSERT
    string disableIdentityInsertQuery = $"SET IDENTITY_INSERT EJet OFF;";
    using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
    {
        disableIdentityInsertCommand.ExecuteNonQuery();
    }
    con.Close();
}
```

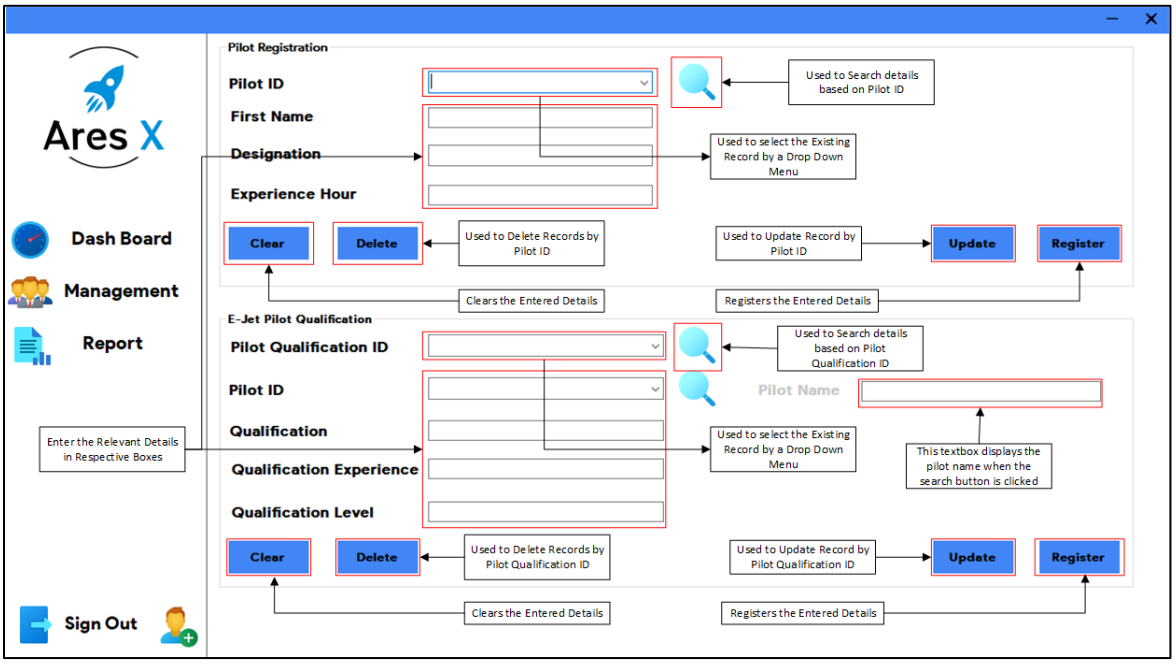
```
1 reference
private void EJet_Search_Button_Click(object sender, EventArgs e)
{
    string EJet_ID = EJetID_Reg_ComboBox.Text;

    if (string.IsNullOrEmpty(EJet_ID))
    {
        MessageBox.Show("Please enter a EJet ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM EJet WHERE EJet_ID = @EJet_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@EJet_ID", EJet_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    EJetID_Reg_ComboBox.Text = reader["EJet_ID"].ToString();
                    EJet_PassSeat_TextBox.Text = reader["Passenger_Seats"].ToString();
                    Jet_Type_ComboBox.Text = reader["EJet_Type"].ToString();
                    EJet_MadeYear_Picker.Text = reader["Made_Year"].ToString();
                    EJet_MadeYear_Picker.Format = DateTimePickerFormat.Custom;
                    EJet_MadeYear_Picker.CustomFormat = "yyyy/MM/dd";
                    EJet_MadeYear_Picker.Text = reader["Made_Year"].ToString();
                    EJet_Weight_TextBox.Text = reader["Weight"].ToString();
                    Nuclear_Engine_Power_Textbox.Text = reader["Nuclear_Engine_Power"].ToString();
                    Power_Source_ComboBox.Text = reader["Power_Source"].ToString();



                }
                else
                {
                    MessageBox.Show("No record found for the given EJet ID.");
                }
            }
```

```
1 reference
private void EJet_Update_Button_Click(object sender, EventArgs e)
{
    try
    {
        string EJet_ID = EJetID_Reg_ComboBox.Text;
        string Pass_Seat = EJet_PassSeat_TextBox.Text;
        EJet_MadeYear_Picker.Format = DateTimePickerFormat.Custom;
        EJet_MadeYear_Picker.CustomFormat = "yyyy/MM/dd";
        string MadeYear = EJet_MadeYear_Picker.Text;
        string EJet_Weight = EJet_Weight_TextBox.Text;
        string EJet_Type = Jet_Type_ComboBox.Text;
        string Power_Source = Power_Source_ComboBox.Text;
        string Nuclear_Engine = Nuclear_Engine_Power_Textbox.Text;


        string EJet_Query_update = "UPDATE EJet SET " +
              "Passenger_Seats = '" + Pass_Seat + "', " +
              "EJet_Type = '" + EJet_Type + "', " +
              "Made_Year = '" + MadeYear + "', " +
              "Weight = '" + EJet_Weight + "', " +
              "Nuclear_Engine_Power = '" + Nuclear_Engine + "', " +
              "Power_Source = '" + Power_Source + "' " +
              "WHERE EJet_ID = '" + EJet_ID + "'";
        if (string.IsNullOrEmpty(EJet_ID))
        {
            MessageBox.Show("Enter a EJet ID to update the record", "Enter EJet ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (Pass_Seat == "" || MadeYear == "" || EJet_Weight == "" || EJet_Type == "" || Power_Source == "" || Nuclear_Engine == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
```

```
              "Passenger_Seats = '" + Pass_Seat + "', " +
              "EJet_Type = '" + EJet_Type + "', " +
              "Made_Year = '" + MadeYear + "', " +
              "Weight = '" + EJet_Weight + "', " +
              "Nuclear_Engine_Power = '" + Nuclear_Engine + "', " +
              "Power_Source = '" + Power_Source + "' " +
              "WHERE EJet_ID = '" + EJet_ID + "'";
        if (string.IsNullOrEmpty(EJet_ID))
        {
            MessageBox.Show("Enter a EJet ID to update the record", "Enter EJet ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (Pass_Seat == "" || MadeYear == "" || EJet_Weight == "" || EJet_Type == "" || Power_Source == "" || Nuclear_Engine == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(EJet_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Updated SucessFully", "Update EJet", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
catch (SqlException ex)
{
    string message = "Update Error:";
    message += ex.Message;
    MessageBox.Show(message);
    con.Close();
}
```

E-Jet Pilot Page

E-Jet Pilot Page: Illustrative Diagram



E-Jet Pilot Page: Code

```
1 reference
private void PilotReg_Clear_Button_Click(object sender, EventArgs e)
{
    PilotID_Reg_ComboBox.Text = "";
    FName_TextBox.Clear();
    Designation_TextBox.Clear();
    ExpHour_Textbox.Clear();
    FName_TextBox.Focus();

}
```

```
1 reference
private void PilotRegDelete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Pilot_ID = PilotID_Reg_ComboBox.Text;
            if (string.IsNullOrEmpty(Pilot_ID))
            {
                MessageBox.Show("Enter a Pilot ID to Delete the record", "Enter Pilot ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Pilot_Delete_Query = "DELETE from Pilot WHERE Pilot_ID = " + Pilot_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Pilot_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                PilotReg_Clear_Button.PerformClick();
                MessageBox.Show("Record Have Been Deleted", "Pilott Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
}
```

```
1 reference
private void EJet_Pilot_Management_Load(object sender, EventArgs e)
{
    try
    {
        con.Open();
        SqlCommand command = new SqlCommand("Select (Pilot_ID)from Pilot", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("Pilot_ID", typeof(int));
        table.Load(reader);
        PilotID_Reg_ComboBox.ValueMember = "Pilot_ID";
        PilotID_Reg_ComboBox.DataSource = table;
        con.Close();
        PilotID_Reg_ComboBox.Text = "";

        ////////////////////////////////////////////

        con.Open();
        SqlCommand command3 = new SqlCommand("Select (Pilot_ID)from Pilot", con);
        SqlDataReader reader3;
        reader3 = command3.ExecuteReader();
        DataTable table3 = new DataTable();
        table3.Columns.Add("Pilot_ID", typeof(int));
        table3.Load(reader3);
        PilotID_Qua_ComboBox.ValueMember = "Pilot_ID";
        PilotID_Qua_ComboBox.DataSource = table3;
        con.Close();
        PilotID_Qua_ComboBox.Text = "";
        ////////////////////////////////////////////

        con.Open();
        SqlCommand command2 = new SqlCommand("Select (Pilot_Qualification_ID)from Pilot_Qualification", con);
```

```
        reader3 = command3.ExecuteReader();
        DataTable table3 = new DataTable();
        table3.Columns.Add("Pilot_ID", typeof(int));
        table3.Load(reader3);
        PilotID_Qua_ComboBox.ValueMember = "Pilot_ID";
        PilotID_Qua_ComboBox.DataSource = table3;
        con.Close();
        PilotID_Qua_ComboBox.Text = "";
        ////////////////////////////////////////////

        con.Open();
        SqlCommand command2 = new SqlCommand("Select (Pilot_Qualification_ID)from Pilot_Qualification", con);
        SqlDataReader reader2;
        reader2 = command2.ExecuteReader();
        DataTable table2 = new DataTable();
        table2.Columns.Add("Pilot_Qualification_ID", typeof(int));
        table2.Load(reader2);
        PilotID_QuaID_ComboBox.ValueMember = "Pilot_Qualification_ID";
        PilotID_QuaID_ComboBox.DataSource = table2;
        con.Close();
        PilotID_QuaID_ComboBox.Text = "";

    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void ColReg_Search_Button_Click(object sender, EventArgs e)
{
    string Pilot_ID = PilotID_Reg_ComboBox.Text;

    if (string.IsNullOrEmpty(Pilot_ID))
    {
        MessageBox.Show("Please enter a Pilot ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Pilot WHERE Pilot_ID = @Pilot_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Pilot_ID", Pilot_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    PilotID_Reg_ComboBox.Text = reader["Pilot_ID"].ToString();
                    FName_TextBox.Text = reader["First_Name"].ToString();
                    Designation_TextBox.Text = reader["Designation"].ToString();
                    ExpHour_Textbox.Text = reader["Experience"].ToString();


                }
                else
                {
                    MessageBox.Show("No record found for the given Pilot ID.");
                }
            }
```

```
1 reference
private void PilotReg_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Pilot_ID = PilotID_Reg_ComboBox.Text;
        string First_Name = FName_TextBox.Text;
        string Designation = Designation_TextBox.Text;
        string ExpHour = ExpHour_Textbox.Text;

        if (string.IsNullOrEmpty(Pilot_ID))
        {
            if (First_Name == "" || Designation == "" || ExpHour == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string PilotReg_Query_insert_1 = "INSERT INTO Pilot (First_Name, Experience, Designation) " +
                    "VALUES('" + First_Name + "','" + ExpHour + "','" + Designation + "')" ;
                con.Open();
                SqlCommand Register = new SqlCommand(PilotReg_Query_insert_1, con);
                int pilot_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                PilotID_Reg_ComboBox.Text = pilot_ID.ToString();
                string Pil_ID = PilotID_Reg_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + Pil_ID, "Register Pilot", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
        }
        else
        {
            con.Open();
            // Enable IDENTITY_INSERT
```

```
con.Open();
// Enable IDENTITY_INSERT
string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Pilot ON;";
using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
{
    enableIdentityInsertCommand.ExecuteNonQuery();
}
//-----------------------------------------
if (First_Name == "" || Designation == "" || ExpHour == "")
{
    MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}
else
{
    string PilReg_Query_insert_2 = "Insert into Pilot (Pilot_ID ,First_Name ,Experience, Designation) " +
        "Values('" + Pilot_ID + "','" + First_Name + "','" + ExpHour + "','" + Designation + "')";

    SqlCommand Register2 = new SqlCommand(PilReg_Query_insert_2, con);
    Register2.ExecuteNonQuery();

    MessageBox.Show("Record Added SucessFully", "Register Pilot", MessageBoxButtons.OK, MessageBoxIcon.Information);

}


//-------------------------------------
// Disable IDENTITY_INSERT
string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Pilot OFF;";
using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
{
    disableIdentityInsertCommand.ExecuteNonQuery();
}
con.Close();
```

```
1 reference
private void PilotReg_Update_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Pilot_ID = PilotID_Reg_ComboBox.Text;
        string First_Name = FName_TextBox.Text;
        string Designation = Designation_TextBox.Text;
        string ExpHour = ExpHour_Textbox.Text;


        string Pilot_Query_update = "UPDATE Pilot SET " +
              "First_Name = '" + First_Name + "', " +
              "Experience = '" + ExpHour + "', " +
              "Designation = '" + Designation + "' " +
              "WHERE Pilot_ID = '" + Pilot_ID + "'";
        if (string.IsNullOrEmpty(Pilot_ID))
        {
            MessageBox.Show("Enter a Pilot ID to update the record", "Enter Pilot ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (First_Name == "" || Designation == "" || ExpHour == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(Pilot_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Updated SucessFully", "Update Pilot", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Update Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void PilotQua_Clear_Button_Click(object sender, EventArgs e)
{
    PilotID_QuaID_ComboBox.Text = "";
    PilotID_Qua_ComboBox.Text = "";
    Pilot_Qua_Name_TextBox.Clear();
    Pilot_Qualification_TextBox.Clear();
    Pilot_Experience_TextBox.Clear();
    Pilot_Level_TextBox.Clear();
    PilotID_Qua_ComboBox.Focus();

}
```

```
1 reference
private void PilotQua_Delete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Pilot_Qualification_ID = PilotID_QuaID_ComboBox.Text;
            if (string.IsNullOrEmpty(Pilot_Qualification_ID))
            {
                MessageBox.Show("Enter a Pilot Qualification ID to Delete the record", "Enter Pilot Qualification ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Pilot_Qualification_Delete_Query = "DELETE from Pilot_Qualification WHERE Pilot_Qualification_ID = " + Pilot_Qualification_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Pilot_Qualification_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                PilotQua_Clear_Button.PerformClick();
                MessageBox.Show("Record Have Been Deleted", "Pilot Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

```
1 reference
private void PilotQua_Update_Button_Click(object sender, EventArgs e)
{

    try
    {
        string Pilot_Qualification_ID = PilotID_QuaID_ComboBox.Text;
        string Qua_Pilot_ID = PilotID_Qua_ComboBox.Text;
        string Pilot_Qualification = Pilot_Qualification_TextBox.Text;
        string Pilot_Experience = Pilot_Experience_TextBox.Text;
        string Pilot_Level = Pilot_Level_TextBox.Text;
        string Pilot_Name = Pilot_Qua_Name_TextBox.Text;


        string Pilot_Query_update = "UPDATE Pilot_Qualification SET " +
              "Pilot_ID = '" + Qua_Pilot_ID + "', " +
              "Qualification_Description = '" + Pilot_Qualification + "', " +
              "Pilot_Experience = '" + Pilot_Experience + "', " +
              "Pilot_Level = '" + Pilot_Level + "'";

        if (string.IsNullOrEmpty(Pilot_Qualification_ID))
        {
            MessageBox.Show("Enter a Pilot Qualification ID to update the record", "Enter Pilot ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (Qua_Pilot_ID == "" || Pilot_Name == "" || Pilot_Qualification == "" || Pilot_Experience == "" || Pilot_Level == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(Pilot_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Updated SucessFully", "Update Pilot", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Update Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void PilotQua_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Pilot_Qualification_ID = PilotID_QuaID_ComboBox.Text;
        string Qua_Pilot_ID = PilotID_Qua_ComboBox.Text;
        string Pilot_Qualification = Pilot_Qualification_TextBox.Text;
        string Pilot_Experience = Pilot_Experience_TextBox.Text;
        string Pilot_Level = Pilot_Level_TextBox.Text;
        string Pilot_Name = Pilot_Qua_Name_TextBox.Text;

        if (string.IsNullOrEmpty(Pilot_Qualification_ID))
        {
            if (Qua_Pilot_ID == "" || Pilot_Name == "" || Pilot_Qualification == "" || Pilot_Experience == "" || Pilot_Level == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string PilotQuaReg_Query_insert_1 = "INSERT INTO Pilot_Qualification (Pilot_ID, Qualification_Description, Pilot_Experience, Pilot_L
                    "VALUES('" + Qua_Pilot_ID + "','" + Pilot_Qualification + "','" + Pilot_Experience + "','" + Pilot_Level + "')";
                con.Open();
                SqlCommand Register = new SqlCommand(PilotQuaReg_Query_insert_1, con);
                int Qua_Pilo_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                PilotID_QuaID_ComboBox.Text = Qua_Pilo_ID.ToString();
                string PilQua_ID = PilotID_QuaID_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + PilQua_ID, "Register Pilot Qualification", MessageBoxButtons.OK, M
```

```
else
{
    con.Open();
    // Enable IDENTITY_INSERT
    string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Pilot_Qualification ON;";
    using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
    {
        enableIdentityInsertCommand.ExecuteNonQuery();
    }
    //-----------------------------------------
    if (Qua_Pilot_ID == "" || Pilot_Name == "" || Pilot_Qualification == "" || Pilot_Experience == "" || Pilot_Level == "")
    {
        MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    else
    {
        string PilotQuaReg_Query_insert_2 = "Insert into Pilot_Qualification (Pilot_Qualification_ID, Pilot_ID, Qualification_Description, Pilot_Experience, Pilot_Level) " +
            "Values('" + Pilot_Qualification_ID + "','" + Qua_Pilot_ID + "','" + Pilot_Qualification + "','" + Pilot_Experience + "','" + Pilot_Level + "')";

        SqlCommand Register2 = new SqlCommand(PilotQuaReg_Query_insert_2, con);
        Register2.ExecuteNonQuery();

        MessageBox.Show("Record Added SucessFully", "Register Pilot Qualification", MessageBoxButtons.OK, MessageBoxIcon.Information);

    }


    //-------------------------------------
    // Disable IDENTITY_INSERT
    string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Pilot_Qualification OFF;";
    using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
    {
        disableIdentityInsertCommand.ExecuteNonQuery();
    }
    con.Close();
}
```

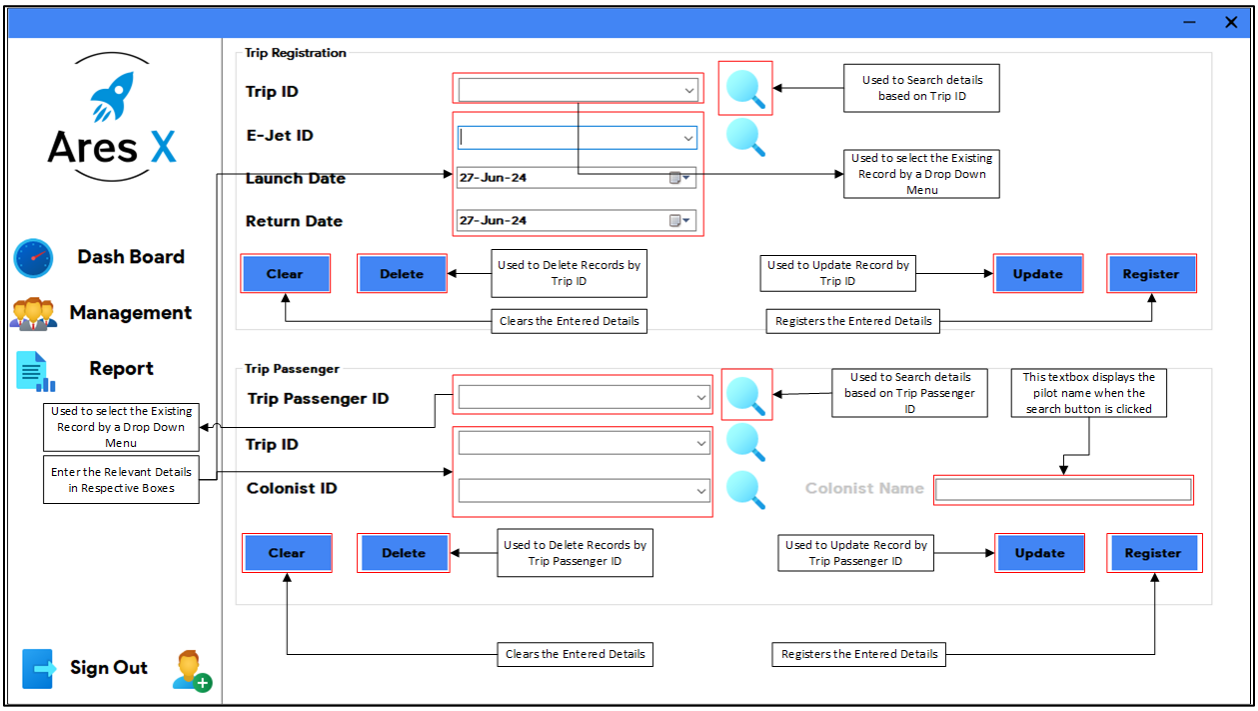
```
            //-----------------------------------------
            if (Qua_Pilot_ID == "" || Pilot_Name == "" || Pilot_Qualification == "" || Pilot_Experience == "" || Pilot_Level == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string PilotQuaReg_Query_insert_2 = "Insert into Pilot_Qualification (Pilot_Qualification_ID, Pilot_ID, Qualification_Description, Pilot_Experience, Pilot_Level) " +
                    "Values('" + Pilot_Qualification_ID + "','" + Qua_Pilot_ID + "','" + Pilot_Qualification + "','" + Pilot_Experience + "','" + Pilot_Level + "')";

                SqlCommand Register2 = new SqlCommand(PilotQuaReg_Query_insert_2, con);
                Register2.ExecuteNonQuery();

                MessageBox.Show("Record Added SucessFully", "Register Pilot Qualification", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }


            //-------------------------------------
            // Disable IDENTITY_INSERT
            string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Pilot_Qualification OFF;";
            using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
            {
                disableIdentityInsertCommand.ExecuteNonQuery();
            }
            con.Close();
        }

    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
}
```

```
1 reference
private void PilotID_PQua_Search_Button_Click(object sender, EventArgs e)
{
    string Pilot_ID = PilotID_Qua_ComboBox.Text;

    if (string.IsNullOrEmpty(Pilot_ID))
    {
        MessageBox.Show("Please enter a Pilot ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Pilot WHERE Pilot_ID = @Pilot_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Pilot_ID", Pilot_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    Pilot_Qua_Name_TextBox.Text = reader["First_Name"].ToString();
                }
                else
                {
                    MessageBox.Show("No record found for the given Pilot ID.");
                }
            }
        }
    }

}
```

```
1 reference
private void Pilot_QuaID_Search_Button_Click(object sender, EventArgs e)
{
    string Pilot_Qualification_ID = PilotID_QuaID_ComboBox.Text;

    if (string.IsNullOrEmpty(Pilot_Qualification_ID))
    {
        MessageBox.Show("Please enter a Pilot Qualification ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Pilot_Qualification WHERE Pilot_Qualification_ID = @Pilot_Qualification_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Pilot_Qualification_ID", Pilot_Qualification_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    PilotID_QuaID_ComboBox.Text = reader["Pilot_Qualification_ID"].ToString();
                    PilotID_Qua_ComboBox.Text = reader["Pilot_ID"].ToString();
                    Pilot_Qualification_TextBox.Text = reader["Qualification_Description"].ToString();
                    Pilot_Experience_TextBox.Text = reader["Pilot_Experience"].ToString();
                    Pilot_Level_TextBox.Text = reader["Pilot_Level"].ToString();
                    PilotID_PQua_Search_Button.PerformClick();
                }
                else
                {
                    MessageBox.Show("No record found for the given Pilot Qualification ID.");
                }
            }
        }
    }
```

Trip Page

Trip Page: Illustrative Diagram



Trip Page: Code

```
1 reference
private void TripReg_Clear_Button_Click(object sender, EventArgs e)
{
    TripID_Reg_ComboBox.Text = "";
    Trip_EJetID_ComboBox.Text = "";
    DateTime This_Day = DateTime.Today;
    Trip_LaunchDate_Picker.Text = This_Day.ToString();
    Trip_ReturnDate_Picker.Text = This_Day.ToString();
    Trip_EJetID_ComboBox.Focus();

}
```

```
1 reference
private void TripRegDelete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Trip_ID = TripID_Reg_ComboBox.Text;
            if (string.IsNullOrEmpty(Trip_ID))
            {
                MessageBox.Show("Enter a Trip ID to Delete the record", "Enter Trip ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Trip_Delete_Query = "DELETE from Trip WHERE Trip_ID = " + Trip_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Trip_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                TripReg_Clear_Button.PerformClick();
                MessageBox.Show("Record Have Been Deleted", "Trip Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

```
1 reference
private void Trip_Management_Load(object sender, EventArgs e)
{
    try
    {
        Trip_EJetID_ComboBox.Focus();
        con.Open();
        SqlCommand command = new SqlCommand("Select (Trip_ID)from Trip", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("Trip_ID", typeof(int));
        table.Load(reader);
        TripID_Reg_ComboBox.ValueMember = "Trip_ID";
        TripID_Reg_ComboBox.DataSource = table;
        con.Close();
        TripID_Reg_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command2 = new SqlCommand("Select (EJet_ID)from EJet", con);
        SqlDataReader reader2;
        reader2 = command2.ExecuteReader();
        DataTable table2 = new DataTable();
        table2.Columns.Add("EJet_ID", typeof(int));
        table2.Load(reader2);
        Trip_EJetID_ComboBox.ValueMember = "EJet_ID";
        Trip_EJetID_ComboBox.DataSource = table2;
        con.Close();
        Trip_EJetID_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command3 = new SqlCommand("Select (Trip_Passenger_ID)from Trip_Passenger", con);
        SqlDataReader reader3;
        reader3 = command3.ExecuteReader();
        DataTable table3 = new DataTable();
        table3.Columns.Add("Trip_Passenger_ID", typeof(int));
        table3.Load(reader3);
```

```
        TripPass_TripPassID_ComboBox.DataSource = table3;
        con.Close();
        TripPass_TripPassID_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command4 = new SqlCommand("Select (Trip_ID)from Trip", con);
        SqlDataReader reader4;
        reader4 = command4.ExecuteReader();
        DataTable table4 = new DataTable();
        table4.Columns.Add("Trip_ID", typeof(int));
        table4.Load(reader4);
        TripPass_TripID_ComboBox.ValueMember = "Trip_ID";
        TripPass_TripID_ComboBox.DataSource = table4;
        con.Close();
        TripPass_TripID_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command5 = new SqlCommand("Select (Colonist_ID) from Colonist", con);
        SqlDataReader reader5;
        reader5 = command5.ExecuteReader();
        DataTable table5 = new DataTable();
        table5.Columns.Add("Colonist_ID", typeof(int));
        table5.Load(reader5);
        TripPass_ColonistID_ComboBox.ValueMember = "Colonist_ID";
        TripPass_ColonistID_ComboBox.DataSource = table5;
        con.Close();
        TripPass_ColonistID_ComboBox.Text = "";

    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void Trip_TripReg_Search_Button_Click(object sender, EventArgs e)
{
    string Trip_ID = TripID_Reg_ComboBox.Text;

    if (string.IsNullOrEmpty(Trip_ID))
    {
        MessageBox.Show("Please enter a Trip ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Trip WHERE Trip_ID = @Trip_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Trip_ID", Trip_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    TripID_Reg_ComboBox.Text = reader["Trip_ID"].ToString();
                    Trip_EJetID_ComboBox.Text = reader["EJet_ID"].ToString();
                    Trip_LaunchDate_Picker.Text = reader["Launch_Date"].ToString();
                    Trip_ReturnDate_Picker.Text = reader["Return_Date"].ToString();


                }
                else
                {
                    MessageBox.Show("No record found for the given Trip ID.");
                }
            }
        }
    }
}
```

```
1 reference
private void TripReg_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Trip_ID = TripID_Reg_ComboBox.Text;
        string EJet_ID = Trip_EJetID_ComboBox.Text;
        Trip_LaunchDate_Picker.Format = DateTimePickerFormat.Custom;
        Trip_LaunchDate_Picker.CustomFormat = "yyyy/MM/dd";
        string Launch_Date = Trip_LaunchDate_Picker.Text;
        Trip_ReturnDate_Picker.Format = DateTimePickerFormat.Custom;
        Trip_ReturnDate_Picker.CustomFormat = "yyyy/MM/dd";
        string Return_Date = Trip_ReturnDate_Picker.Text;


        if (string.IsNullOrEmpty(Trip_ID))
        {
            if (EJet_ID == "" || Launch_Date == "" || Return_Date == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string TripReg_Query_insert_1 = "INSERT INTO Trip (EJet_ID, Launch_Date, Return_Date) " +
                        "VALUES('" + EJet_ID + "','" + Launch_Date + "','" + Return_Date + "')";
                con.Open();
                SqlCommand Register = new SqlCommand(TripReg_Query_insert_1, con);
                int trip_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                TripID_Reg_ComboBox.Text = trip_ID.ToString();
                string Tri_ID = TripID_Reg_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + Tri_ID, "Register Trip", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
        }
        else
        {
            con.Open();
            // Enable IDENTITY_INSERT
            string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Trip ON;";
```

```
    else
    {
        con.Open();
        // Enable IDENTITY_INSERT
        string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Trip ON;";
        using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
        {
            enableIdentityInsertCommand.ExecuteNonQuery();
        }
        //-----------------------------------------
        if (EJet_ID == "" || Launch_Date == "" || Return_Date == "")
        {
            MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            string TripReg_Query_insert_2 = "Insert into Trip (Trip_ID ,EJet_ID, Launch_Date, Return_Date) " +
                "Values('" + Trip_ID + "','" + EJet_ID + "','" + Launch_Date + "','" + Return_Date + "')";

            SqlCommand Register2 = new SqlCommand(TripReg_Query_insert_2, con);
            Register2.ExecuteNonQuery();

            MessageBox.Show("Record Added SucessFully", "Register Trip", MessageBoxButtons.OK, MessageBoxIcon.Information);

        }


        //-------------------------------------
        // Disable IDENTITY_INSERT
        string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Trip OFF;";
        using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
        {
            disableIdentityInsertCommand.ExecuteNonQuery();
        }
        con.Close();
    }
}

catch (SqlException ex)
{
    string message = "Insert Error:";
    message += ex.Message;
    MessageBox.Show(message);
    con.Close();
}
```

```
private void TripReg_Update_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Trip_ID = TripID_Reg_ComboBox.Text;
        string EJet_ID = Trip_EJetID_ComboBox.Text;
        Trip_LaunchDate_Picker.Format = DateTimePickerFormat.Custom;
        Trip_LaunchDate_Picker.CustomFormat = "yyyy/MM/dd";
        string Launch_Date = Trip_LaunchDate_Picker.Text;
        Trip_ReturnDate_Picker.Format = DateTimePickerFormat.Custom;
        Trip_ReturnDate_Picker.CustomFormat = "yyyy/MM/dd";
        string Return_Date = Trip_ReturnDate_Picker.Text;


        string Trip_Query_update = "UPDATE Trip SET " +
              "EJet_ID = '" + EJet_ID + "', " +
              "Launch_Date = '" + Launch_Date + "', " +
              "Return_Date = '" + Return_Date + "' " +
              "WHERE Trip_ID = '" + Trip_ID + "'";
        if (string.IsNullOrEmpty(Trip_ID))
        {
            MessageBox.Show("Enter a Trip ID to update the record", "Enter Trip ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (EJet_ID == "" || Launch_Date == "" || Return_Date == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(Trip_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Updated SucessFully", "Update Trip", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Update Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void TripPass_Clear_Button_Click(object sender, EventArgs e)
{
    TripPass_TripPassID_ComboBox.Text = "";
    TripPass_TripID_ComboBox.Text = "";
    TripPass_ColonistID_ComboBox.Text = "";
    Trip_ID_Finder_Label.Text = "";
    Colonist_Name_TextBox.Clear();
    TripPass_TripID_ComboBox.Focus();
}
```

```
1 reference
private void TripPass_ColonistID_Search_Button_Click(object sender, EventArgs e)
{
    string Colonist_ID = TripPass_ColonistID_ComboBox.Text;

    if (string.IsNullOrEmpty(Colonist_ID))
    {
        MessageBox.Show("Please enter a Colonist ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Colonist WHERE Colonist_ID = @Colonist_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Colonist_ID", Colonist_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    Colonist_Name_TextBox.Text = reader["First_Name"].ToString();
                }
                else
                {
                    MessageBox.Show("No record found for the given Colonist ID.");
                }
            }
        }
    }

}
```

```
1 reference
private void TripPass_Delete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Trip_Passenger_ID = TripPass_TripPassID_ComboBox.Text;
            if (string.IsNullOrEmpty(Trip_Passenger_ID))
            {
                MessageBox.Show("Enter a Trip Passenger ID to Delete the record", "Enter Colonist Qualification ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Trip_Passenger_Delete_Query = "DELETE from Trip_Passenger WHERE Trip_Passenger_ID = " + Trip_Passenger_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Trip_Passenger_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Have Been Deleted", "Passenger Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

```
1 reference
private void TripPass_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Trip_Passenger_ID = TripPass_TripPassID_ComboBox.Text;
        string Trip_ID = TripPass_TripID_ComboBox.Text;
        string Colonist_ID = TripPass_ColonistID_ComboBox.Text;
        string Colonist_Name = Colonist_Name_TextBox.Text;

        if (string.IsNullOrEmpty(Trip_Passenger_ID))
        {
            if (Trip_ID == "" || Colonist_ID == "" || Colonist_Name == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string TripPassengerReg_Query_insert_1 = "INSERT INTO Trip_Passenger (Trip_ID, Colonist_ID) " +
                     "VALUES('" + Trip_ID + "','"+ Colonist_ID +"')";
                con.Open();
                SqlCommand Register = new SqlCommand(TripPassengerReg_Query_insert_1, con);
                int Tri_Pass_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                TripPass_TripPassID_ComboBox.Text = Tri_Pass_ID.ToString();
                string TriPass_ID = TripPass_TripPassID_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + TriPass_ID, "Register Trip Passenger", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
        }
        else
        {
            con.Open();
            // Enable IDENTITY_INSERT
            string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Trip_Passenger ON;";
            using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
```

```
            con.Open();
            // Enable IDENTITY_INSERT
            string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Trip_Passenger ON;";
            using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
            {
                enableIdentityInsertCommand.ExecuteNonQuery();
            }
            //-----------------------------------------
            if (Trip_ID == "" || Colonist_ID == "" || Colonist_Name == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string TripPassengerReg_Query_insert_2 = "Insert into Trip_Passenger (Trip_Passenger_ID,Trip_ID, Colonist_ID) " +
                     "Values('" + Trip_Passenger_ID + "','" + Trip_ID + "','" + Colonist_ID + "')";

                SqlCommand Register2 = new SqlCommand(TripPassengerReg_Query_insert_2, con);
                Register2.ExecuteNonQuery();

                MessageBox.Show("Record Added SucessFully", "Register Trip Passenger", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }


            //-------------------------------------
            // Disable IDENTITY_INSERT
            string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Trip_Passenger OFF;";
            using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
            {
                disableIdentityInsertCommand.ExecuteNonQuery();
            }
            con.Close();
        }


    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

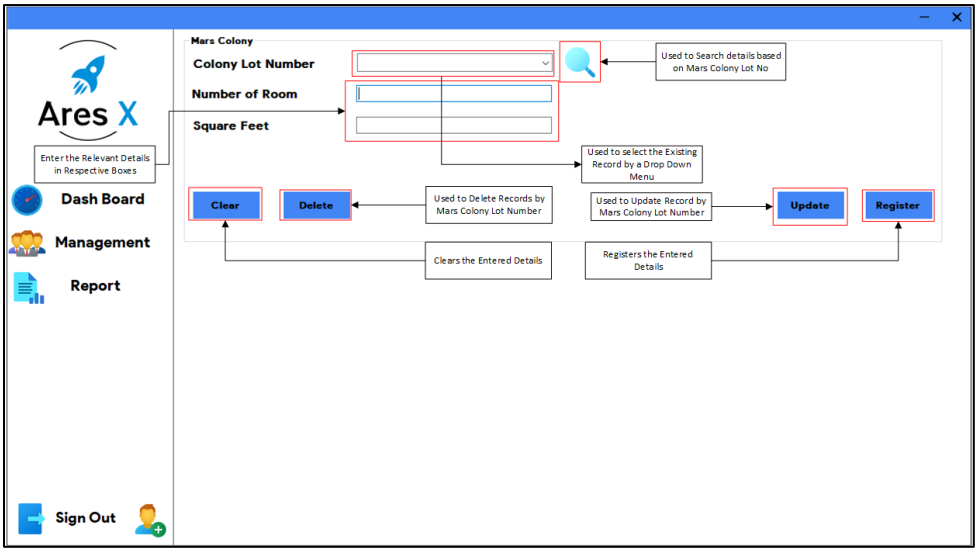
```
1 reference
private void TripPass_TripID_Search_Button_Click(object sender, EventArgs e)
{
    string Trip_ID = TripPass_TripID_ComboBox.Text;

    if (string.IsNullOrEmpty(Trip_ID))
    {
        Trip_ID_Finder_Label.Text = "";
        MessageBox.Show("Please enter a Trip ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Trip WHERE Trip_ID = @Trip_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Trip_ID", Trip_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    Trip_ID_Finder_Label.Text = "Available";
                    Trip_ID_Finder_Label.ForeColor = Color.LimeGreen;
                }
                else
                {
                    Trip_ID_Finder_Label.Text = "Not Available";
                    Trip_ID_Finder_Label.ForeColor = Color.Red;
                    MessageBox.Show("No record found for the given Trip ID.");
                }
            }
        }
    }
```

```
1 reference
private void TripPass_TripPassID_Search_Button_Click(object sender, EventArgs e)
{
    string Trip_Passenger_ID = TripPass_TripPassID_ComboBox.Text;

    if (string.IsNullOrEmpty(Trip_Passenger_ID))
    {
        MessageBox.Show("Please enter a Trip Passenger ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Trip_Passenger WHERE Trip_Passenger_ID = @Trip_Passenger_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Trip_Passenger_ID", Trip_Passenger_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    TripPass_TripPassID_ComboBox.Text = reader["Trip_Passenger_ID"].ToString();
                    TripPass_TripID_ComboBox.Text = reader["Trip_ID"].ToString();
                    TripPass_ColonistID_ComboBox.Text = reader["Colonist_ID"].ToString();
                    TripPass_ColonistID_Search_Button.PerformClick();
                }
                else
                {
                    MessageBox.Show("No record found for the given Trip Passenger ID.");
                }
            }
        }
    }
```

```
1 reference
private void TripPass_Update_Button_Click(object sender, EventArgs e)
{

    try
    {
        string Trip_Passenger_ID = TripPass_TripPassID_ComboBox.Text;
        string Trip_ID = TripPass_TripID_ComboBox.Text;
        string Colonist_ID = TripPass_ColonistID_ComboBox.Text;
        string Colonist_Name = Colonist_Name_TextBox.Text;

        string TripPassenger_Query_update = "UPDATE Trip_Passenger SET " +
              "Trip_ID = '" + Trip_ID + "', " +
              "Colonist_ID = '" + Colonist_ID + "'";


        if (string.IsNullOrEmpty(Trip_Passenger_ID))
        {
            MessageBox.Show("Enter a Trip Passenger ID to update the record", "Enter Colonist ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (Trip_ID == "" || Colonist_ID == "" || Colonist_Name == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(TripPassenger_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();

                MessageBox.Show("Record Updated SucessFully", "Update Trip Passenger", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Update Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

Mars Colony Page

Mars Colonization Page: Illustrative Diagram

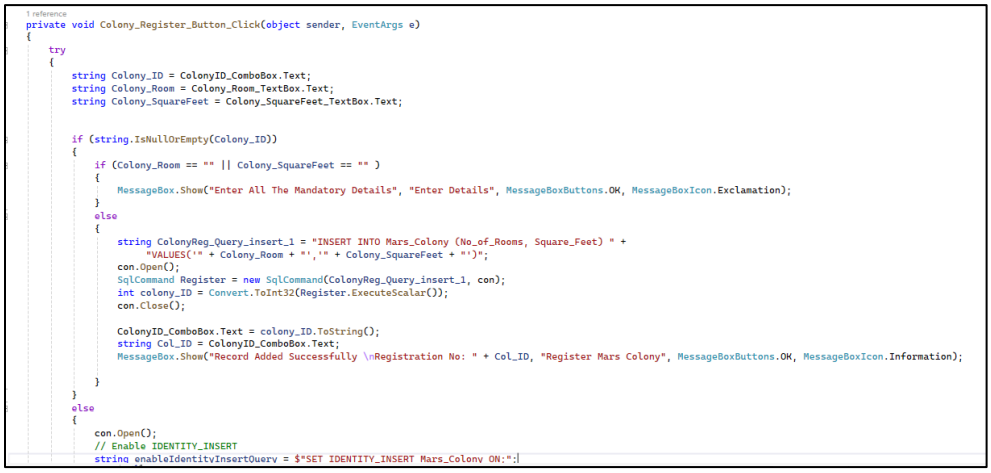


Mars Colonization Page: Code

```
1 reference
private void Colony_Clear_Button_Click(object sender, EventArgs e)
{
    ColonyID_ComboBox.Text = "";
    Colony_Room_TextBox.Clear();
    Colony_SquareFeet_TextBox.Clear();
    Colony_Room_TextBox.Focus();

}
```

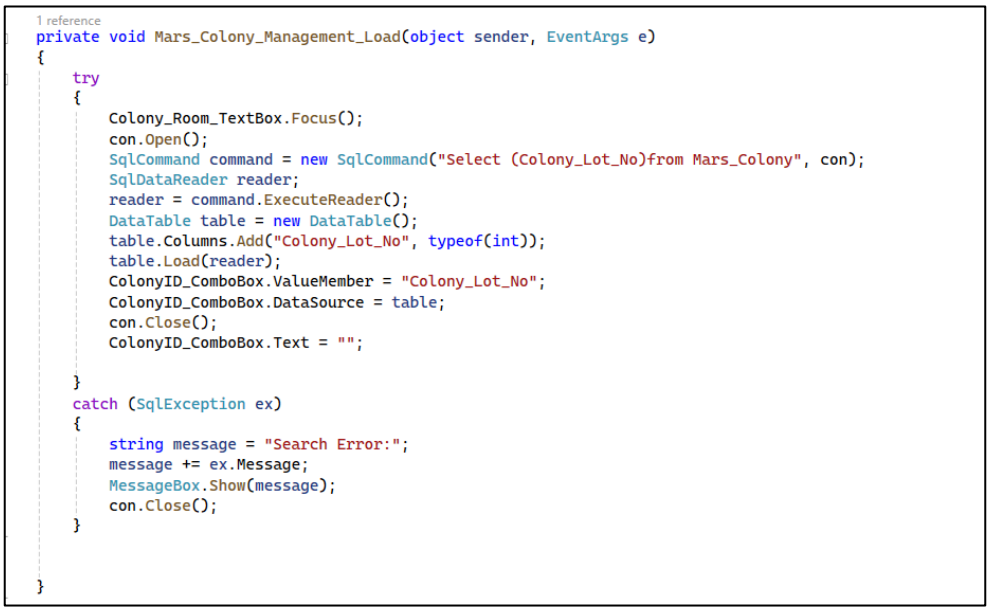
```
1 reference
private void Colony_Delete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Colony_Lot_No = ColonyID_ComboBox.Text;
            if (string.IsNullOrEmpty(Colony_Lot_No))
            {
                MessageBox.Show("Enter a Colony_Lot_No to Delete the record", "Enter Colony Lot No", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Colony_Delete_Query = "DELETE from Mars_Colony WHERE Colony_Lot_No = " + Colony_Lot_No + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Colony_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                Colony_Clear_Button.PerformClick();
                MessageBox.Show("Record Have Been Deleted", "Colony Lot No Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
}
```

```
1 reference
private void Colony_Update_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Colony_Lot_No = ColonyID_ComboBox.Text;
        string Colony_Room = Colony_Room_TextBox.Text;
        string Colony_SquareFeet = Colony_SquareFeet_TextBox.Text;

        string Colony_Query_update = "UPDATE Mars_Colony SET " +
              "No_of_Rooms = '" + Colony_Room + "', " +
              "Square_Feet = '" + Colony_SquareFeet + "' " +
              "WHERE Colony_Lot_No = '" + Colony_Lot_No + "'";
        if (string.IsNullOrEmpty(Colony_Lot_No))
        {
            MessageBox.Show("Enter a Colonist ID to update the record", "Enter Colonist ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (Colony_Room == "" || Colony_SquareFeet == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(Colony_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Updated SucessFully", "Update Mars Colony", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Update Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```



```
1 reference
private void Colony_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Colony_ID = ColonyID_ComboBox.Text;
        string Colony_Room = Colony_Room_TextBox.Text;
        string Colony_SquareFeet = Colony_SquareFeet_TextBox.Text;


        if (string.IsNullOrEmpty(Colony_ID))
        {
            if (Colony_Room == "" || Colony_SquareFeet == "" )
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string ColonyReg_Query_insert_1 = "INSERT INTO Mars_Colony (No_of_Rooms, Square_Feet) " +
                     "VALUES('" + Colony_Room + "','" + Colony_SquareFeet + "')";
                con.Open();
                SqlCommand Register = new SqlCommand(ColonyReg_Query_insert_1, con);
                int colony_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                ColonyID_ComboBox.Text = colony_ID.ToString();
                string Col_ID = ColonyID_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + Col_ID, "Register Mars Colony", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
        }
        else
        {
            con.Open();
            // Enable IDENTITY_INSERT
            string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Mars_Colony ON;";
```

```
            // Enable IDENTITY_INSERT
            string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Mars_Colony ON;";
            using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
            {
                enableIdentityInsertCommand.ExecuteNonQuery();
            }
            //-----------------------------------------
            if (Colony_Room == "" || Colony_SquareFeet == "" )
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string ColonyReg_Query_insert_2 = "Insert into Mars_Colony (Colony_Lot_No ,No_of_Rooms, Square_Feet) " +
                     "Values('" + Colony_ID + "','" + Colony_Room + "','" + Colony_SquareFeet + "')";

                SqlCommand Register2 = new SqlCommand(ColonyReg_Query_insert_2, con);
                Register2.ExecuteNonQuery();

                MessageBox.Show("Record Added SucessFully", "Register Mars Colony", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }


            //-------------------------------------
            // Disable IDENTITY_INSERT
            string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Mars_Colony OFF;";
            using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
            {
                disableIdentityInsertCommand.ExecuteNonQuery();
            }
            con.Close();
        }
    }

    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

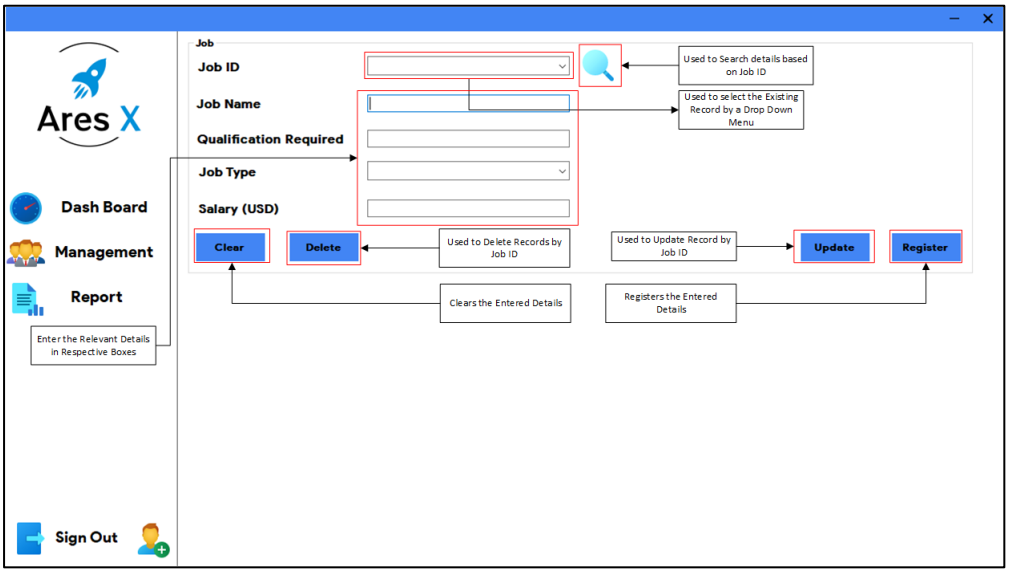


```
1 reference
private void Mars_Colony_Management_Load(object sender, EventArgs e)
{
    try
    {
        Colony_Room_TextBox.Focus();
        con.Open();
        SqlCommand command = new SqlCommand("Select (Colony_Lot_No)from Mars_Colony", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("Colony_Lot_No", typeof(int));
        table.Load(reader);
        ColonyID_ComboBox.ValueMember = "Colony_Lot_No";
        ColonyID_ComboBox.DataSource = table;
        con.Close();
        ColonyID_ComboBox.Text = "";

    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }


}
```

```
1 reference
private void ColonyID_Search_Button_Click(object sender, EventArgs e)
{
    string Colony_Lot_No = ColonyID_ComboBox.Text;

    if (string.IsNullOrEmpty(Colony_Lot_No))
    {
        MessageBox.Show("Please enter a Colony Lot No.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Mars_Colony WHERE Colony_Lot_No = @Colony_Lot_No";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Colony_Lot_No", Colony_Lot_No);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    ColonyID_ComboBox.Text = reader["Colony_Lot_No"].ToString();
                    Colony_Room_TextBox.Text = reader["No_of_Rooms"].ToString();
                    Colony_SquareFeet_TextBox.Text = reader["Square_Feet"].ToString();


                }
                else
                {
                    MessageBox.Show("No record found for the given Colony Lot No.");
                }
            }
        }
    }

}
```

Job Page

Job Page: Illustrative Diagram



Job Page: Code

```
1 reference
private void Job_Clear_Button_Click(object sender, EventArgs e)
{
    JobID_ComboBox.Text = "";
    Job_Name_TextBox.Clear();
    Qualification_Req_TextBox.Clear();
    Job_Type_ComboBox.Text = "";
    Job_Salary_TextBox.Clear();
    Job_Name_TextBox.Focus();
}
```

```
1 reference
private void Job_Delete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Job_ID = JobID_ComboBox.Text;
            if (string.IsNullOrEmpty(Job_ID))
            {
                MessageBox.Show("Enter a Job ID to Delete the record", "Enter Job ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Job_Delete_Query = "DELETE from Job WHERE Job_ID = " + Job_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Job_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                Job_Clear_Button.PerformClick();
                MessageBox.Show("Record Have Been Deleted", "Job Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

```
private void JobID_Search_Button_Click(object sender, EventArgs e)
{
    string Job_ID = JobID_ComboBox.Text;

    if (string.IsNullOrEmpty(Job_ID))
    {
        MessageBox.Show("Please enter a Job ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Job WHERE Job_ID = @Job_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Job_ID", Job_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    JobID_ComboBox.Text = reader["Job_ID"].ToString();
                    Job_Name_TextBox.Text = reader["Job_Title"].ToString();
                    Qualification_Req_TextBox.Text = reader["Job_Qualification"].ToString();
                    Job_Type_ComboBox.Text = reader["Job_Type"].ToString();
                    Job_Salary_TextBox.Text = reader["Job_Salary"].ToString();

                }
                else
                {
                    MessageBox.Show("No record found for the given Job ID.");
                }
            }
```

```
1 reference
private void Job_Management_Load(object sender, EventArgs e)
{
    try
    {
        Job_Name_TextBox.Focus();
        con.Open();
        SqlCommand command = new SqlCommand("Select (Job_ID)from Job", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("Job_ID", typeof(int));
        table.Load(reader);
        JobID_ComboBox.ValueMember = "Job_ID";
        JobID_ComboBox.DataSource = table;
        con.Close();
        JobID_ComboBox.Text = "";


    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

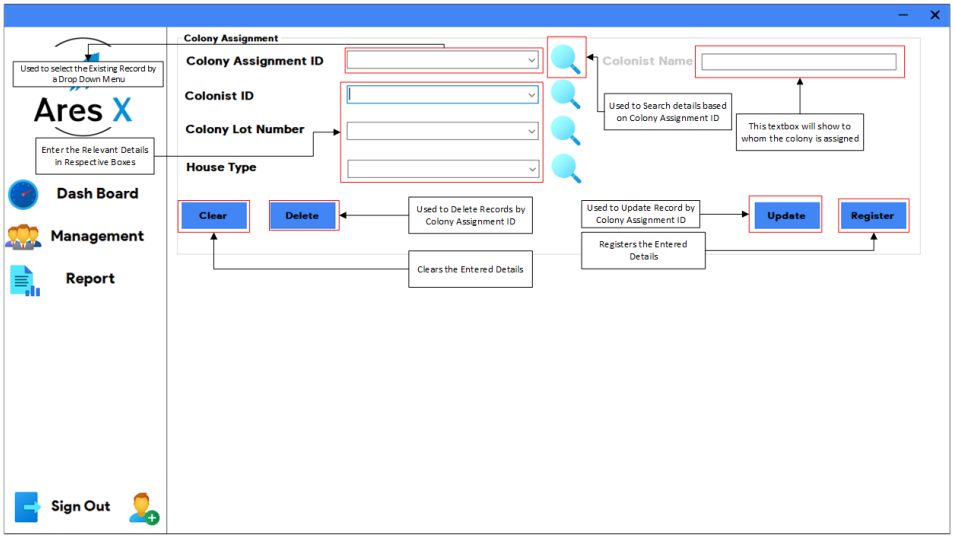
```
private void Job_Update_Button_Click(object sender, EventArgs e)
{
    try
    {

        string Job_ID = JobID_ComboBox.Text;
        string Job_Name = Job_Name_TextBox.Text;
        string Job_Qualification = Qualification_Req_TextBox.Text;
        string Job_Type = Job_Type_ComboBox.Text;
        string Job_Salary = Job_Salary_TextBox.Text;

        string Job_Query_update = "UPDATE Job SET " +
              "Job_Title = '" + Job_Name + "', " +
              "Job_Qualification = '" + Job_Qualification + "', " +
              "Job_Type = '" + Job_Type + "', " +
              "Job_Salary = '" + Job_Salary + "' " +
              "WHERE Job_ID = '" + Job_ID + "'";
        if (string.IsNullOrEmpty(Job_ID))
        {
            MessageBox.Show("Enter a Job ID to update the record", "Enter Job ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (Job_Name == "" || Job_Qualification == "" || Job_Type == "" || Job_Salary == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(Job_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Updated SucessFully", "Update Job", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Update Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
private void Job_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Job_ID = JobID_ComboBox.Text;
        string Job_Name = Job_Name_TextBox.Text;
        string Job_Qualification = Qualification_Req_TextBox.Text;
        string Job_Type = Job_Type_ComboBox.Text;
        string Job_Salary = Job_Salary_TextBox.Text;

        if (string.IsNullOrEmpty(Job_ID))
        {
            if (Job_Name == "" || Job_Qualification == "" || Job_Type == "" || Job_Salary == "" )
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string JobReg_Query_insert_1 = "INSERT INTO job (Job_Title, Job_Qualification, Job_Type, Job_Salary) " +
                     "VALUES('" + Job_Name + "','" + Job_Qualification + "','" + Job_Type + "','" + Job_Salary + "')";
                con.Open();
                SqlCommand Register = new SqlCommand(JobReg_Query_insert_1, con);
                int job_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                JobID_ComboBox.Text = job_ID.ToString();
                string jo_ID = JobID_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + jo_ID, "Register Job", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
        }
        else
        {
            con.Open();
            // Enable IDENTITY_INSERT
```

```
            con.Open();
            // Enable IDENTITY_INSERT
            string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Job ON;";
            using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
            {
                enableIdentityInsertCommand.ExecuteNonQuery();
            }
            //-----------------------------------------
            if (Job_Name == "" || Job_Qualification == "" || Job_Type == "" || Job_Salary == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string JobReg_Query_insert_2 = "Insert into Job (Job_ID ,Job_Title, Job_Qualification, Job_Type, Job_Salary) " +
                     "Values('" + Job_ID + "','" + Job_Name + "','" + Job_Qualification + "','" + Job_Type + "','" + Job_Salary + "')";

                SqlCommand Register2 = new SqlCommand(JobReg_Query_insert_2, con);
                Register2.ExecuteNonQuery();

                MessageBox.Show("Record Added SucessFully", "Register Job", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }


            //-------------------------------------
            // Disable IDENTITY_INSERT
            string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Job OFF;";
            using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
            {
                disableIdentityInsertCommand.ExecuteNonQuery();
            }
            con.Close();
        }
    }
```

Colony Assignment Page

Colony Assignment Page: Illustrative Diagram



Colony Assignment Page: Code

```
1 reference
private void Colony_Clear_Button_Click(object sender, EventArgs e)
{
    ColonyID_ComboBox.Text = "";
    Colony_Room_TextBox.Clear();
    Colony_SquareFeet_TextBox.Clear();
    Colony_Room_TextBox.Focus();

}
```

```
1 reference
private void Colony_Delete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Colony_Lot_No = ColonyID_ComboBox.Text;
            if (string.IsNullOrEmpty(Colony_Lot_No))
            {
                MessageBox.Show("Enter a Colony_Lot_No to Delete the record", "Enter Colony Lot No", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Colony_Delete_Query = "DELETE from Mars_Colony WHERE Colony_Lot_No = " + Colony_Lot_No + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Colony_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                Colony_Clear_Button.PerformClick();
                MessageBox.Show("Record Have Been Deleted", "Colony Lot No Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
}
```

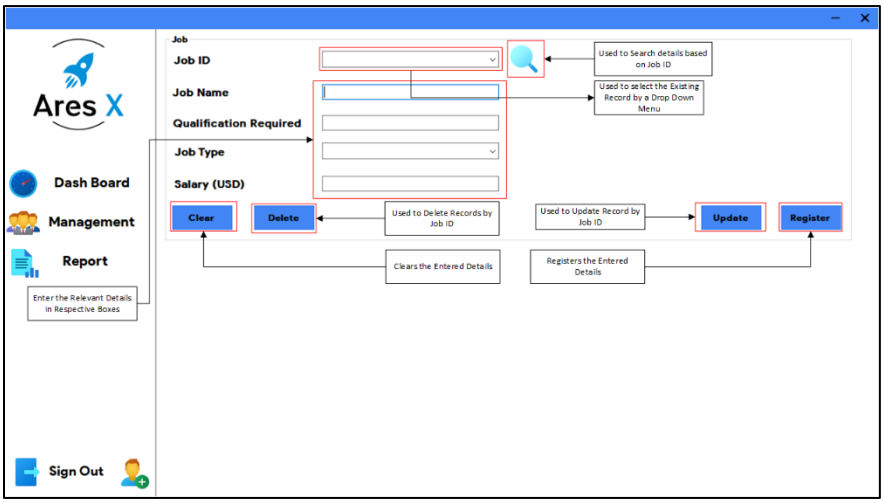
```
1 reference
private void Colony_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Colony_ID = ColonyID_ComboBox.Text;
        string Colony_Room = Colony_Room_TextBox.Text;
        string Colony_SquareFeet = Colony_SquareFeet_TextBox.Text;


        if (string.IsNullOrEmpty(Colony_ID))
        {
            if (Colony_Room == "" || Colony_SquareFeet == "" )
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string ColonyReg_Query_insert_1 = "INSERT INTO Mars_Colony (No_of_Rooms, Square_Feet) " +
                     "VALUES('" + Colony_Room + "','" + Colony_SquareFeet + "')" + "SELECT SCOPE_IDENTITY();";
                con.Open();
                SqlCommand Register = new SqlCommand(ColonyReg_Query_insert_1, con);
                int colony_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                ColonyID_ComboBox.Text = colony_ID.ToString();
                string Col_ID = ColonyID_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + Col_ID, "Register Mars Colony", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
        }
        else
        {
            con.Open();
            // Enable IDENTITY_INSERT
```

```
1 reference
private void Mars_Colony_Management_Load(object sender, EventArgs e)
{
    try
    {
        Colony_Room_TextBox.Focus();
        con.Open();
        SqlCommand command = new SqlCommand("Select (Colony_Lot_No)from Mars_Colony", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("Colony_Lot_No", typeof(int));
        table.Load(reader);
        ColonyID_ComboBox.ValueMember = "Colony_Lot_No";
        ColonyID_ComboBox.DataSource = table;
        con.Close();
        ColonyID_ComboBox.Text = "";

    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }


}
}
```

```
1 reference
private void ColonyID_Search_Button_Click(object sender, EventArgs e)
{
    string Colony_Lot_No = ColonyID_ComboBox.Text;

    if (string.IsNullOrEmpty(Colony_Lot_No))
    {
        MessageBox.Show("Please enter a Colony Lot No.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Mars_Colony WHERE Colony_Lot_No = @Colony_Lot_No";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Colony_Lot_No", Colony_Lot_No);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    ColonyID_ComboBox.Text = reader["Colony_Lot_No"].ToString();
                    Colony_Room_TextBox.Text = reader["No_of_Rooms"].ToString();
                    Colony_SquareFeet_TextBox.Text = reader["Square_Feet"].ToString();


                }
                else
                {
                    MessageBox.Show("No record found for the given Colony Lot No.");
                }
            }
```

Job Assignment Page

Job Assignment Page: Illustrative Diagram



Job Assignment Page: Code

```
1 reference
private void CJobAss_Clear_Button_Click(object sender, EventArgs e)
{
    CJobAss_CJobID_ComboBox.Text = "";
    CJobAss_JobID_ComboBox.Text = "";
    CJobAss_CID_ComboBox.Text = "";
    CJobAss_JobID_ComboBox.Focus();

}
```

```
1 reference
private void CJobAss_Delete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Colonist_Job_ID = CJobAss_CJobID_ComboBox.Text;
            if (string.IsNullOrEmpty(Colonist_Job_ID))
            {
                MessageBox.Show("Enter a Colonist Job ID to Delete the record", "Enter Colonist Job ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string Colonist_Job_Delete_Query = "DELETE from Colonist_Job WHERE Colonist_Job_ID = " + Colonist_Job_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(Colonist_Job_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                CJobAss_Clear_Button.PerformClick();
                MessageBox.Show("Record Have Been Deleted", "Colonist Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
}
```

```
1 reference
private void CJobAss_Update_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Colonist_Job_ID = CJobAss_CJobID_ComboBox.Text;
        string Job_ID = CJobAss_JobID_ComboBox.Text;
        string Colonist_ID = CJobAss_CID_ComboBox.Text;

        string ColonistJob_Query_update = "UPDATE Colonist_Job SET " +
              "Job_ID = '" + Job_ID + "', " +
              "Colonist_ID = '" + Colonist_ID + "' " +
              "WHERE Colonist_Job_ID = '" + Colonist_Job_ID + "'";
        if (string.IsNullOrEmpty(Colonist_Job_ID))
        {
            MessageBox.Show("Enter a Colonist Job ID to update the record", "Enter Colonist Job ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            if (Job_ID == "" || Colonist_ID == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                con.Open();
                SqlCommand Update = new SqlCommand(ColonistJob_Query_update, con);
                Update.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Updated SucessFully", "Update Colonist Job", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Update Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void Colonist_Job_Management_Load(object sender, EventArgs e)
{
    try
    {
        CJobAss_JobID_ComboBox.Focus();
        con.Open();
        SqlCommand command = new SqlCommand("Select (Colonist_Job_ID)from Colonist_Job", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("Colonist_Job_ID", typeof(int));
        table.Load(reader);
        CJobAss_CJobID_ComboBox.ValueMember = "Colonist_Job_ID";
        CJobAss_CJobID_ComboBox.DataSource = table;
        con.Close();
        CJobAss_CJobID_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command3 = new SqlCommand("Select (Colonist_ID)from Colonist", con);
        SqlDataReader reader3;
        reader3 = command3.ExecuteReader();
        DataTable table3 = new DataTable();
        table3.Columns.Add("Colonist_ID", typeof(int));
        table3.Load(reader3);
        CJobAss_CID_ComboBox.ValueMember = "Colonist_ID";
        CJobAss_CID_ComboBox.DataSource = table3;
        con.Close();
        CJobAss_CID_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command2 = new SqlCommand("Select (Job_ID)from Job", con);
        SqlDataReader reader2;
        reader2 = command2.ExecuteReader();
        DataTable table2 = new DataTable();
```

```
        SqlCommand command3 = new SqlCommand("Select (Colonist_ID)from Colonist", con);
        SqlDataReader reader3;
        reader3 = command3.ExecuteReader();
        DataTable table3 = new DataTable();
        table3.Columns.Add("Colonist_ID", typeof(int));
        table3.Load(reader3);
        CJobAss_CID_ComboBox.ValueMember = "Colonist_ID";
        CJobAss_CID_ComboBox.DataSource = table3;
        con.Close();
        CJobAss_CID_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command2 = new SqlCommand("Select (Job_ID)from Job", con);
        SqlDataReader reader2;
        reader2 = command2.ExecuteReader();
        DataTable table2 = new DataTable();
        table2.Columns.Add("Job_ID", typeof(int));
        table2.Load(reader2);
        CJobAss_JobID_ComboBox.ValueMember = "Job_ID";
        CJobAss_JobID_ComboBox.DataSource = table2;
        con.Close();
        CJobAss_JobID_ComboBox.Text = "";

    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

```
1 reference
private void CJobAss_JobID_Search_Button_Click(object sender, EventArgs e)
{
    string Job_ID = CJobAss_JobID_ComboBox.Text;

    if (string.IsNullOrEmpty(Job_ID))
    {
        MessageBox.Show("Please enter a Job ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Job WHERE Job_ID = @Job_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Job_ID", Job_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    Job_Name_Textbox.Text = reader["Job_Title"].ToString();
                }
                else
                {
                    MessageBox.Show("No record found for the given Job ID.");
                }
            }
        }
    }
```

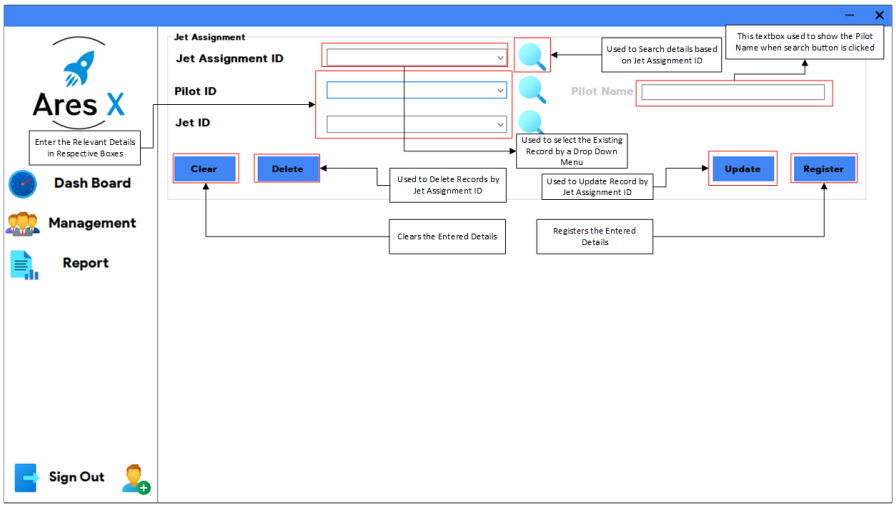
```
1 reference
private void CJobAss_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Colonist_Job_ID = CJobAss_CJobID_ComboBox.Text;
        string Job_ID = CJobAss_JobID_ComboBox.Text;
        string Colonist_ID = CJobAss_CID_ComboBox.Text;

        if (string.IsNullOrEmpty(Colonist_Job_ID))
        {
            if (Job_ID == "" || Colonist_ID == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string JobAss_Query_insert_1 = "INSERT INTO Colonist_Job (Colonist_ID, Job_ID) " +
                    "VALUES('" + Colonist_ID + "','" + Job_ID + "')";
                con.Open();
                SqlCommand Register = new SqlCommand(JobAss_Query_insert_1, con);
                int JobAss_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                CJobAss_CJobID_ComboBox.Text = JobAss_ID.ToString();
                string Job_Ass_ID = CJobAss_CJobID_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + Job_Ass_ID, "Assign Job", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
        }
        else
        {
            con.Open();
            // Enable IDENTITY_INSERT
```

```
            string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Colonist_Job ON;";
            using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
            {
                enableIdentityInsertCommand.ExecuteNonQuery();
            }
            //-----------------------------------------
            if (Job_ID == "" || Colonist_ID == "")
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string JobAss_Query_insert_2 = "Insert into Colonist_Job (Colonist_Job_ID ,Colonist_ID, Job_ID) " +
                     "Values('" + Colonist_Job_ID + "','" + Colonist_ID + "','" + Job_ID + "')";

                SqlCommand Register2 = new SqlCommand(JobAss_Query_insert_2, con);
                Register2.ExecuteNonQuery();

                MessageBox.Show("Record Added SucessFully", "Assign Job", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }


            //-------------------------------------
            // Disable IDENTITY_INSERT
            string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Colonist_Job OFF;";
            using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
            {
                disableIdentityInsertCommand.ExecuteNonQuery();
            }
            con.Close();
        }
    }

    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void CJobAss_CJobID_Search_Button_Click(object sender, EventArgs e)
{
    string Colonist_Job_ID = CJobAss_CJobID_ComboBox.Text;

    if (string.IsNullOrEmpty(Colonist_Job_ID))
    {
        MessageBox.Show("Please enter a Colonist Job ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Colonist_Job WHERE Colonist_Job_ID = @Colonist_Job_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Colonist_Job_ID", Colonist_Job_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    CJobAss_CJobID_ComboBox.Text = reader["Colonist_Job_ID"].ToString();
                    CJobAss_JobID_ComboBox.Text = reader["Job_ID"].ToString();
                    CJobAss_CID_ComboBox.Text = reader["Colonist_ID"].ToString();
                    CJobAss_JobID_Search_Button.PerformClick();
                    CJobAss_CID_Search_Button.PerformClick();
                }
                else
                {
                    MessageBox.Show("No record found for the given Colonist Qualification ID.");
                }
            }
```

Jet Assignment Page

Jet Assignment Page: Illustrative Diagram



Jet Assignment Page: Code

```
1 reference
private void JetAss_Clear_Button_Click(object sender, EventArgs e)
{
    JetAss_JAssID_ComboBox.Text = "";
    JetAss_PilotID_ComboBox.Text = "";
    JetAss_JetID_ComboBox.Text = "";
    JetAss_PilotID_ComboBox.Focus();
}
```

```
1 reference
private void JetAss_Delete_Button_Click(object sender, EventArgs e)
{
    try
    {
        var result = MessageBox.Show("Are You Sure You Need To Delete This Record...?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            string Jet_Assignment_ID = JetAss_JAssID_ComboBox.Text;
            if (string.IsNullOrEmpty(Jet_Assignment_ID))
            {
                MessageBox.Show("Enter a Jet Assignment ID to Delete the record", "Enter Jet Assignment ID", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string JetAss_Delete_Query = "DELETE from Jet_Assignment WHERE Jet_Assignment_ID = " + Jet_Assignment_ID + " ";
                con.Open();
                SqlCommand Command = new SqlCommand(JetAss_Delete_Query, con);
                Command.ExecuteNonQuery();
                con.Close();
                MessageBox.Show("Record Have Been Deleted", "Jet Assignment Deleted", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
    catch (SqlException ex)
    {
        string message = "Insert Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }

}
```

```
private void Jet_Assignment_Management_Load(object sender, EventArgs e)
{
    try
    {
        JetAss_PilotID_ComboBox.Focus();
        con.Open();
        SqlCommand command = new SqlCommand("Select (Jet_Assignment_ID)from Jet_Assignment", con);
        SqlDataReader reader;
        reader = command.ExecuteReader();
        DataTable table = new DataTable();
        table.Columns.Add("Jet_Assignment_ID", typeof(int));
        table.Load(reader);
        JetAss_JAssID_ComboBox.ValueMember = "Jet_Assignment_ID";
        JetAss_JAssID_ComboBox.DataSource = table;
        con.Close();
        JetAss_JAssID_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command3 = new SqlCommand("Select (Pilot_ID)from Pilot", con);
        SqlDataReader reader3;
        reader3 = command3.ExecuteReader();
        DataTable table3 = new DataTable();
        table3.Columns.Add("Pilot_ID", typeof(int));
        table3.Load(reader3);
        JetAss_PilotID_ComboBox.ValueMember = "Pilot_ID";
        JetAss_PilotID_ComboBox.DataSource = table3;
        con.Close();
        JetAss_PilotID_ComboBox.Text = "";
        ////////////////////////////////////////////
        con.Open();
        SqlCommand command2 = new SqlCommand("Select (EJet_ID)from EJet", con);
        SqlDataReader reader2;
        reader2 = command2.ExecuteReader();
        DataTable table2 = new DataTable();
        table2.Columns.Add("EJet_ID", typeof(int));
        table2.Load(reader2);
        JetAss_JetID_ComboBox.ValueMember = "EJet_ID";
        JetAss_JetID_ComboBox.DataSource = table2;
        con.Close();
        JetAss_JetID_ComboBox.Text = "";

    }
    catch (SqlException ex)
    {
        string message = "Search Error:";
        message += ex.Message;
        MessageBox.Show(message);
        con.Close();
    }
```

```
1 reference
private void JetAss_Register_Button_Click(object sender, EventArgs e)
{
    try
    {
        string Jet_Assignment_ID = JetAss_JAssID_ComboBox.Text;
        string Pilot_ID = JetAss_PilotID_ComboBox.Text;
        string EJet_ID = JetAss_JetID_ComboBox.Text;


        if (string.IsNullOrEmpty(Jet_Assignment_ID))
        {
            if (Pilot_ID == "" || EJet_ID == "" )
            {
                MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {
                string JetAss_Query_insert_1 = "INSERT INTO Jet_Assignment (Pilot_ID, EJet_ID) " +
                     "VALUES('" + Pilot_ID + "','" + EJet_ID + "')";
                con.Open();
                SqlCommand Register = new SqlCommand(JetAss_Query_insert_1, con);
                int jetass_ID = Convert.ToInt32(Register.ExecuteScalar());
                con.Close();

                JetAss_JAssID_ComboBox.Text = jetass_ID.ToString();
                string jet_ass_ID = JetAss_JAssID_ComboBox.Text;
                MessageBox.Show("Record Added Successfully \nRegistration No: " + jet_ass_ID, "Register Jet Assignment", MessageBoxButtons.OK, MessageBoxIcon.In

            }
        }
        else
        {
            con.Open();
            // Enable IDENTITY_INSERT
```

```
        con.Open();
        // Enable IDENTITY_INSERT
        string enableIdentityInsertQuery = $"SET IDENTITY_INSERT Jet_Assignment ON;";
        using (SqlCommand enableIdentityInsertCommand = new SqlCommand(enableIdentityInsertQuery, con))
        {
            enableIdentityInsertCommand.ExecuteNonQuery();
        }
        //-----------------------------------------
        if (Pilot_ID == "" || EJet_ID == "")
        {
            MessageBox.Show("Enter All The Mandatory Details", "Enter Details", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            string JetAss_Query_insert_2 = "Insert into Jet_Assignment (Jet_Assignment_ID ,Pilot_ID, EJet_ID) " +
                 "Values('" + Jet_Assignment_ID + "','" + Pilot_ID + "','" + EJet_ID + "')";

            SqlCommand Register2 = new SqlCommand(JetAss_Query_insert_2, con);
            Register2.ExecuteNonQuery();

            MessageBox.Show("Record Added SucessFully", "Register Jet Assignment", MessageBoxButtons.OK, MessageBoxIcon.Information);

        }


        //-------------------------------------
        // Disable IDENTITY_INSERT
        string disableIdentityInsertQuery = $"SET IDENTITY_INSERT Jet_Assignment OFF;";
        using (SqlCommand disableIdentityInsertCommand = new SqlCommand(disableIdentityInsertQuery, con))
        {
            disableIdentityInsertCommand.ExecuteNonQuery();
        }
        con.Close();
    }
}
```

```
1 reference
private void JetAss_JAssID_Search_Button_Click(object sender, EventArgs e)
{
    string Jet_Assignment_ID = JetAss_JAssID_ComboBox.Text;

    if (string.IsNullOrEmpty(Jet_Assignment_ID))
    {
        MessageBox.Show("Please enter a Jet Assignment ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Jet_Assignment WHERE Jet_Assignment_ID = @Jet_Assignment_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Jet_Assignment_ID", Jet_Assignment_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    JetAss_JAssID_ComboBox.Text = reader["Jet_Assignment_ID"].ToString();
                    JetAss_PilotID_ComboBox.Text = reader["Pilot_ID"].ToString();
                    JetAss_JetID_ComboBox.Text = reader["EJet_ID"].ToString();
                    JetAss_PilotID_Search_Button.PerformClick();
                }
                else
                {
                    MessageBox.Show("No record found for the given Jet Assignment ID.");
                }
            }
        }
    }
```

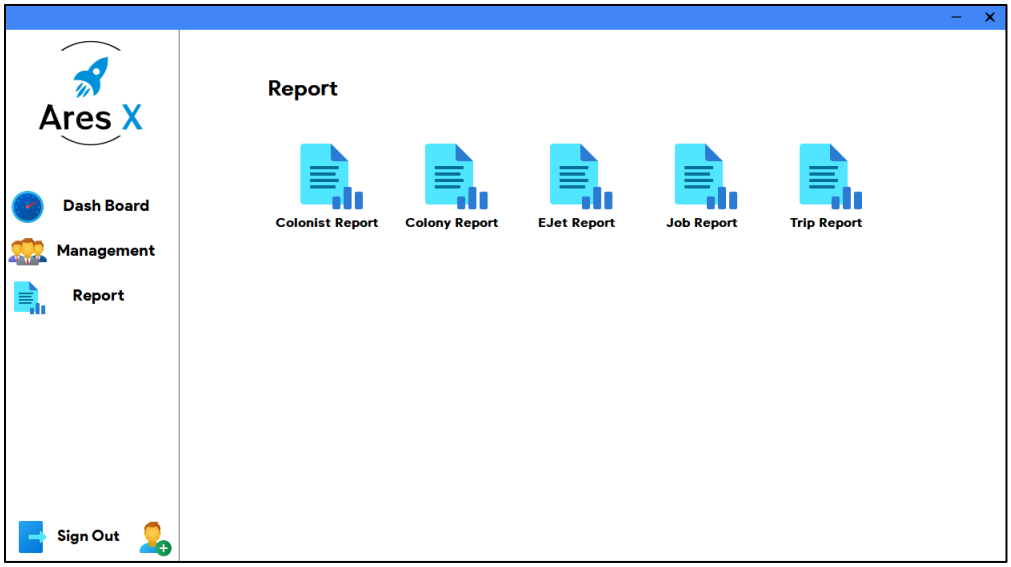
```
1 reference
private void JetAss_JAssID_Search_Button_Click(object sender, EventArgs e)
{
    string Jet_Assignment_ID = JetAss_JAssID_ComboBox.Text;

    if (string.IsNullOrEmpty(Jet_Assignment_ID))
    {
        MessageBox.Show("Please enter a Jet Assignment ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Jet_Assignment WHERE Jet_Assignment_ID = @Jet_Assignment_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Jet_Assignment_ID", Jet_Assignment_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    JetAss_JAssID_ComboBox.Text = reader["Jet_Assignment_ID"].ToString();
                    JetAss_PilotID_ComboBox.Text = reader["Pilot_ID"].ToString();
                    JetAss_JetID_ComboBox.Text = reader["EJet_ID"].ToString();
                    JetAss_PilotID_Search_Button.PerformClick();
                }
                else
                {
                    MessageBox.Show("No record found for the given Jet Assignment ID.");
                }
            }
        }
    }
```

```
1 reference
private void JetAss_JetID_Search_Button_Click(object sender, EventArgs e)
{
    string EJet_ID = JetAss_JetID_ComboBox.Text;

    if (string.IsNullOrEmpty(EJet_ID))
    {
        Jet_ID_Finder_Label.Text = "";
        MessageBox.Show("Please enter a EJet ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM EJet WHERE EJet_ID = @EJet_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@EJet_ID", EJet_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    Jet_ID_Finder_Label.Text = "Available";
                    Jet_ID_Finder_Label.ForeColor = Color.LimeGreen;
                }
                else
                {
                    Jet_ID_Finder_Label.Text = "Not Available";
                    Jet_ID_Finder_Label.ForeColor = Color.Red;
                    MessageBox.Show("No record found for the given Jet ID.");
                }
            }
        }
```

```
1 reference
private void JetAss_PilotID_Search_Button_Click(object sender, EventArgs e)
{
    string Pilot_ID = JetAss_PilotID_ComboBox.Text;

    if (string.IsNullOrEmpty(Pilot_ID))
    {
        MessageBox.Show("Please enter a Pilot ID.", "Search Record", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    using (SqlConnection connection = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
    {
        connection.Open();

        string query = "SELECT * FROM Pilot WHERE Pilot_ID = @Pilot_ID";
        using (SqlCommand command = new SqlCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Pilot_ID", Pilot_ID);

            using (SqlDataReader reader = command.ExecuteReader())
            {
                if (reader.Read())
                {
                    Pilot_Name_TextBox.Text = reader["First_Name"].ToString();

                }
                else
                {
                    MessageBox.Show("No record found for the given Pilot ID.");
                }
            }
        }
    }

}
```

Report Management Page

Report Management Page: Illustrative Diagram



Report Management Page: Code

```
1 reference
private void Colonist_Report_Button_Click(object sender, EventArgs e)
{
    Colonist_Report_Button.Visible = false;
    Colony_Report_Button.Visible = false;
    Jet_Report_Button.Visible = false;
    Job_Report_Button.Visible = false;
    Trip_Report_Button.Visible = false;

    this.Report_Form_Panel.Controls.Clear();
    Colonist_Report Colonist_Report_New = new Colonist_Report() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Colonist_Report_New.FormBorderStyle = FormBorderStyle.None;
    this.Report_Form_Panel.Controls.Add(Colonist_Report_New);
    Colonist_Report_New.Show();

}
```

```
1 reference
private void Colony_Report_Button_Click(object sender, EventArgs e)
{
    Colonist_Report_Button.Visible = false;
    Colony_Report_Button.Visible = false;
    Jet_Report_Button.Visible = false;
    Job_Report_Button.Visible = false;
    Trip_Report_Button.Visible = false;

    this.Report_Form_Panel.Controls.Clear();
    Colony_Report Colony_Report_New = new Colony_Report() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Colony_Report_New.FormBorderStyle = FormBorderStyle.None;
    this.Report_Form_Panel.Controls.Add(Colony_Report_New);
    Colony_Report_New.Show();


}
```

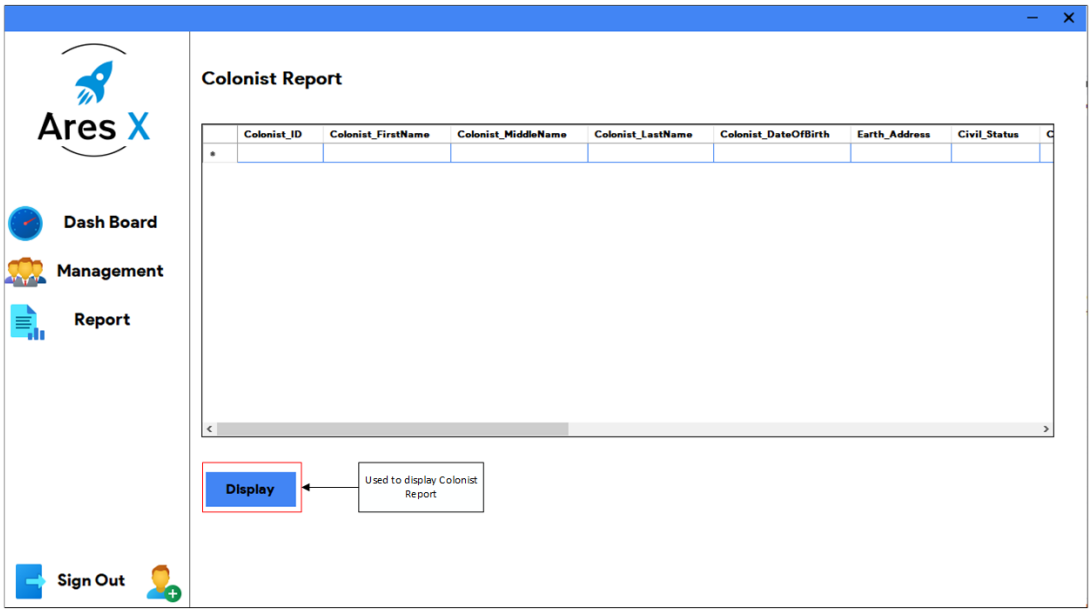
```
1 reference
private void Jet_Report_Button_Click(object sender, EventArgs e)
{
    Colonist_Report_Button.Visible = false;
    Colony_Report_Button.Visible = false;
    Jet_Report_Button.Visible = false;
    Job_Report_Button.Visible = false;
    Trip_Report_Button.Visible = false;

    this.Report_Form_Panel.Controls.Clear();
    Jet_Report Jet_Report_New = new Jet_Report() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Jet_Report_New.FormBorderStyle = FormBorderStyle.None;
    this.Report_Form_Panel.Controls.Add(Jet_Report_New);
    Jet_Report_New.Show();
}
```

```
1 reference
private void Job_Report_Button_Click(object sender, EventArgs e)
{
    Colonist_Report_Button.Visible = false;
    Colony_Report_Button.Visible = false;
    Jet_Report_Button.Visible = false;
    Job_Report_Button.Visible = false;
    Trip_Report_Button.Visible = false;

    this.Report_Form_Panel.Controls.Clear();
    Job_Report Job_Report_New = new Job_Report() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Job_Report_New.FormBorderStyle = FormBorderStyle.None;
    this.Report_Form_Panel.Controls.Add(Job_Report_New);
    Job_Report_New.Show();
}
```

```
1 reference
private void Trip_Report_Button_Click(object sender, EventArgs e)
{
    Colonist_Report_Button.Visible = false;
    Colony_Report_Button.Visible = false;
    Jet_Report_Button.Visible = false;
    Job_Report_Button.Visible = false;
    Trip_Report_Button.Visible = false;

    this.Report_Form_Panel.Controls.Clear();
    Trip_Report Trip_Report_New = new Trip_Report() { Dock = DockStyle.Fill, TopLevel = false, TopMost = true };
    Trip_Report_New.FormBorderStyle = FormBorderStyle.None;
    this.Report_Form_Panel.Controls.Add(Trip_Report_New);
    Trip_Report_New.Show();


}
```

Colonist Report Page

Colonist Report Page: Illustrative Diagram

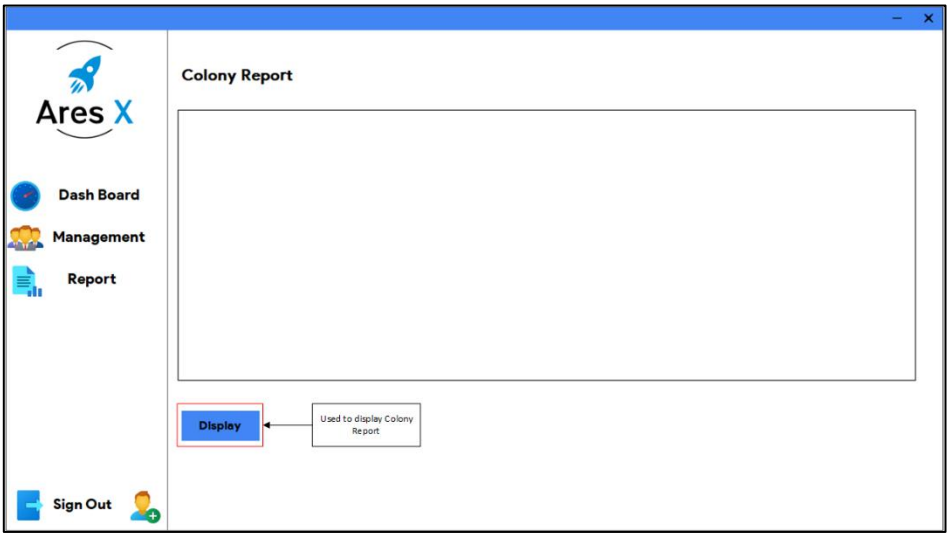


Colonist Report Page: Code

```
1 reference
private void Display_Button_Click(object sender, EventArgs e)
{
    try
    {

        using (SqlConnection sqlCon = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
        {
            sqlCon.Open();


            SqlDataAdapter sqlDa = new SqlDataAdapter(...., sqlCon);

            DataTable dtbl = new DataTable();
            sqlDa.Fill(dtbl);


            Colonist_DGV.DataSource = dtbl;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
```

Colony Report Page

Colony Report Page: Illustrative Diagram

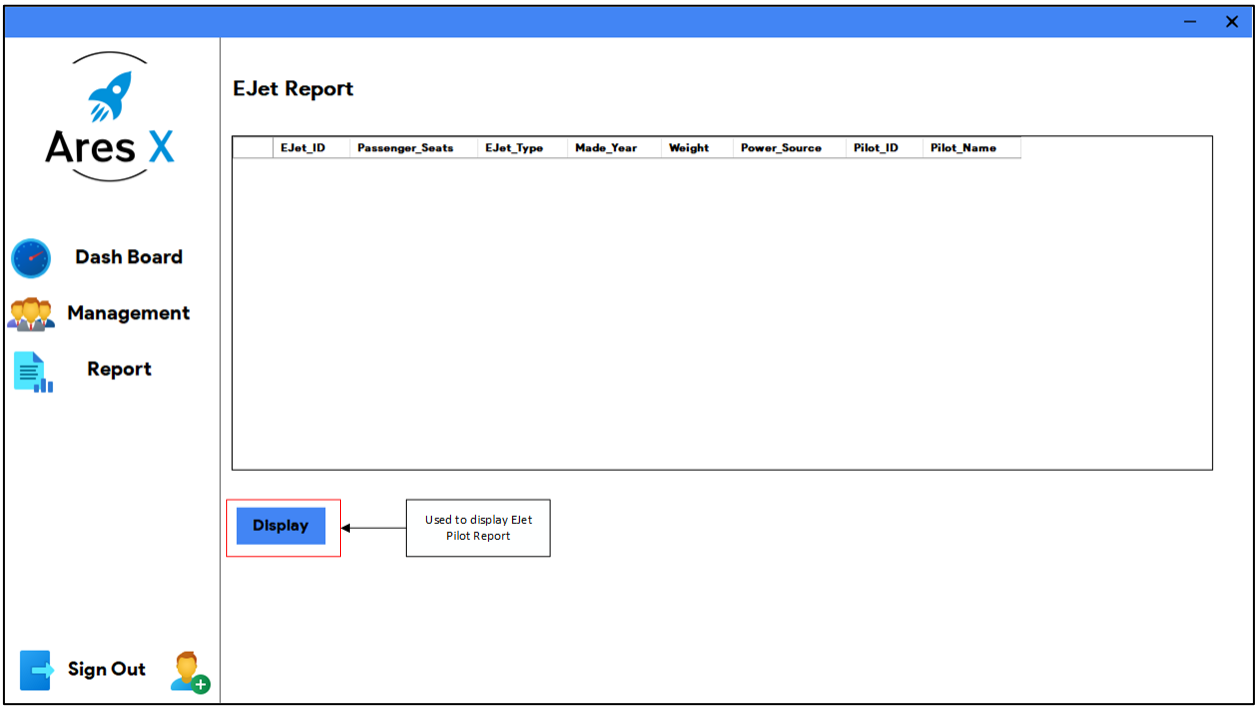


Colony Report Page: Code

```
1 reference
private void Display_Button_Click(object sender, EventArgs e)
{
    try
    {

        using (SqlConnection sqlCon = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
        {
            sqlCon.Open();

            // Prepare SQL query and data adapter
            SqlDataAdapter sqlDa = new SqlDataAdapter(..., sqlCon);


            DataTable dtbl = new DataTable();
            sqlDa.Fill(dtbl);


            Colony_DGV.DataSource = dtbl;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }

}
}
```

E-Jet Report Page

E-Jet Report Page: Illustration Page

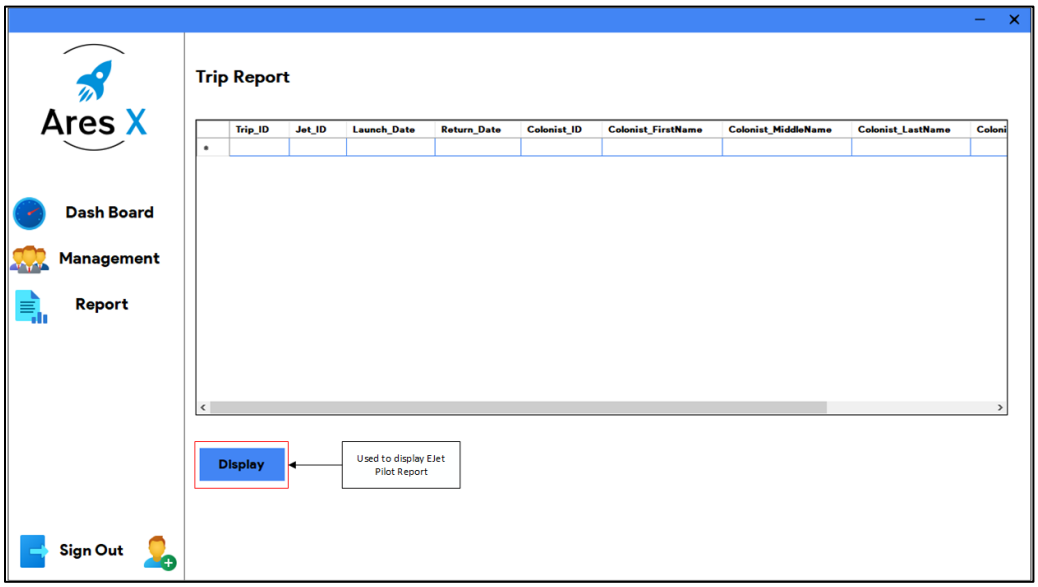


E-Jet Report Page: Code

```
1 reference
private void Display_Button_Click(object sender, EventArgs e)
{
    try
    {

        using (SqlConnection sqlCon = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
        {
            sqlCon.Open();


        SqlDataAdapter sqlDa = new SqlDataAdapter(@"
            SELECT
                EJet.EJet_ID,
                EJet.Passenger_Seats,
                EJet.EJet_Type,
                EJet.Made_Year,
                EJet.Weight,
                EJet.Power_Source,
                Jet_Assignment.Pilot_ID,
                Pilot.First_Name + ' ' + Pilot.Designation AS Pilot_Name
            FROM
                EJet
            LEFT JOIN
                Jet_Assignment ON EJet.EJet_ID = Jet_Assignment.EJet_ID
            LEFT JOIN
                Pilot ON Jet_Assignment.Pilot_ID = Pilot.Pilot_ID;", sqlCon);

            DataTable dtbl = new DataTable();
            sqlDa.Fill(dtbl);


            EJet_DGV.DataSource = dtbl;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }

}
}
```

Job Report Page

Job Report Page: Illustration Diagram

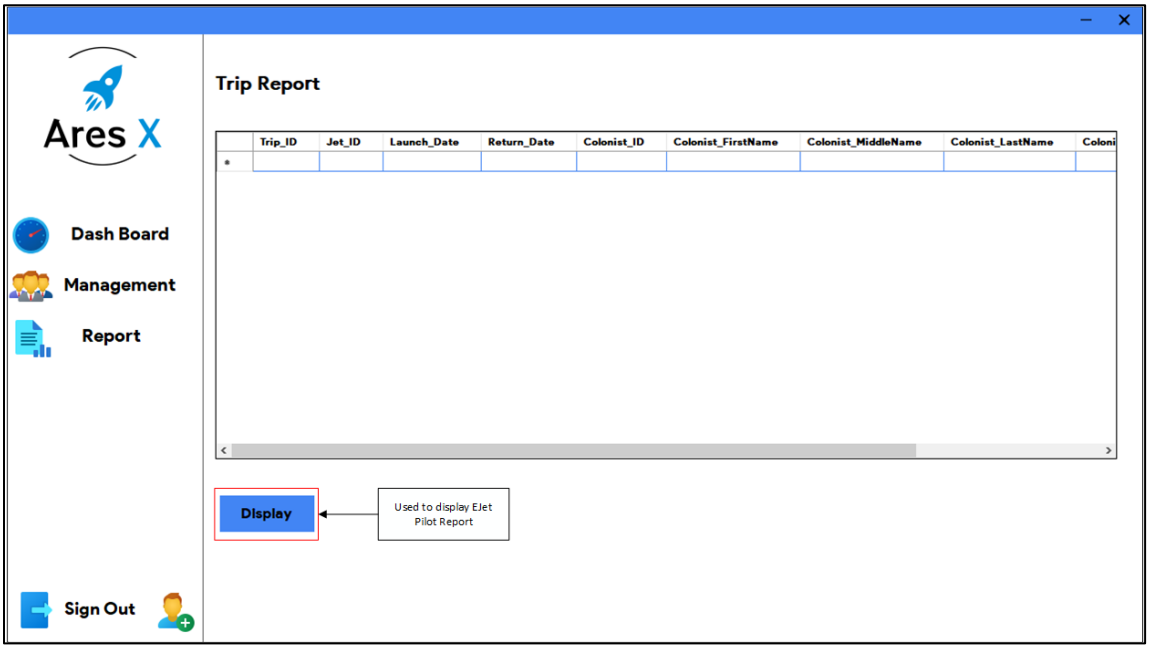


Job Report Page: Code

```
1 reference
private void Display_Button_Click(object sender, EventArgs e)
{
    try
    {

        using (SqlConnection sqlCon = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
        {
            sqlCon.Open();

            // Prepare SQL query and data adapter
            SqlDataAdapter sqlDa = new SqlDataAdapter(@"
            SELECT
            Colonist.Colonist_ID,
            Colonist.First_Name AS Colonist_FirstName,
            Colonist.Middle_Name AS Colonist_MiddleName,
            Colonist.Last_Name AS Colonist_LastName,
            Colonist.Date_Of_Birth AS Colonist_DateOfBirth,
            Job.Job_Title AS Job_Assigned,
            Job.Job_Qualification,
            Job.Job_Type,
            Job.Job_Salary
        FROM
            Colonist
        JOIN
            Colonist_Job ON Colonist.Colonist_ID = Colonist_Job.Colonist_ID
        JOIN
            Job ON Colonist_Job.Job_ID = Job.Job_ID", sqlCon);


            DataTable dtbl = new DataTable();
            sqlDa.Fill(dtbl);


            Job_DGV.DataSource = dtbl;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
```

Trip Report Page

Trip Report Page: Illustration Diagram

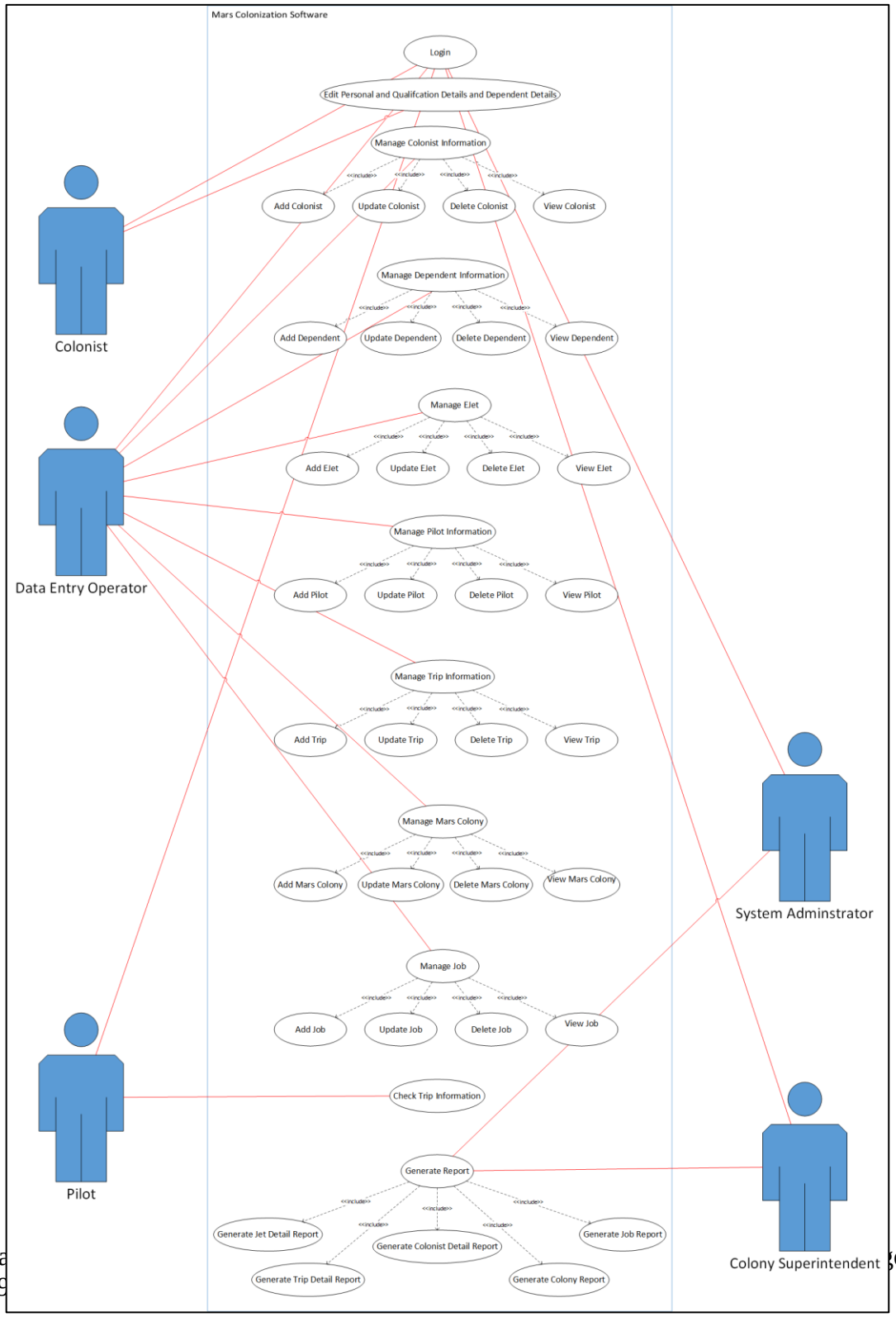


Trip Report Page: Code

```
1 reference
private void Display_Button_Click(object sender, EventArgs e)
{
    try
    {

        using (SqlConnection sqlCon = new SqlConnection(@"Data Source=Code-Maestro;Initial Catalog=Mars_Colonization;Integrated Security=True; "))
        {
            sqlCon.Open();

            // Prepare SQL query and data adapter
            SqlDataAdapter sqlDa = new SqlDataAdapter(..., sqlCon);


            DataTable dtbl = new DataTable();
            sqlDa.Fill(dtbl);


            Trip_DGV.DataSource = dtbl;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }

}
}
```

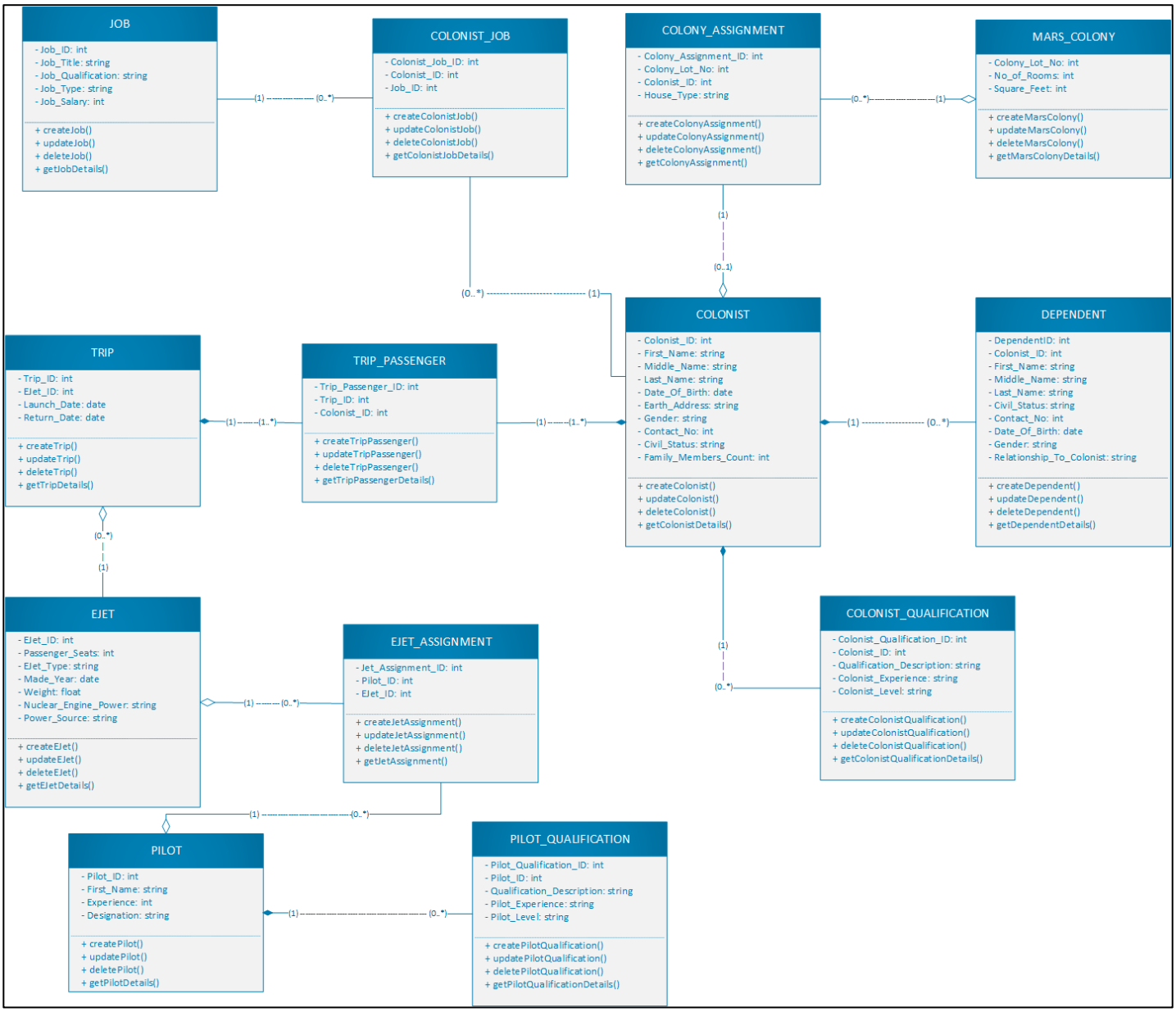
Use Case Diagram

Use case Diagram is a type of UML diagram that shows the interactions between actors and a system using graphical representation. It focuses on the functionalities of the system as a perspective of its users. (Visual Paradigm)



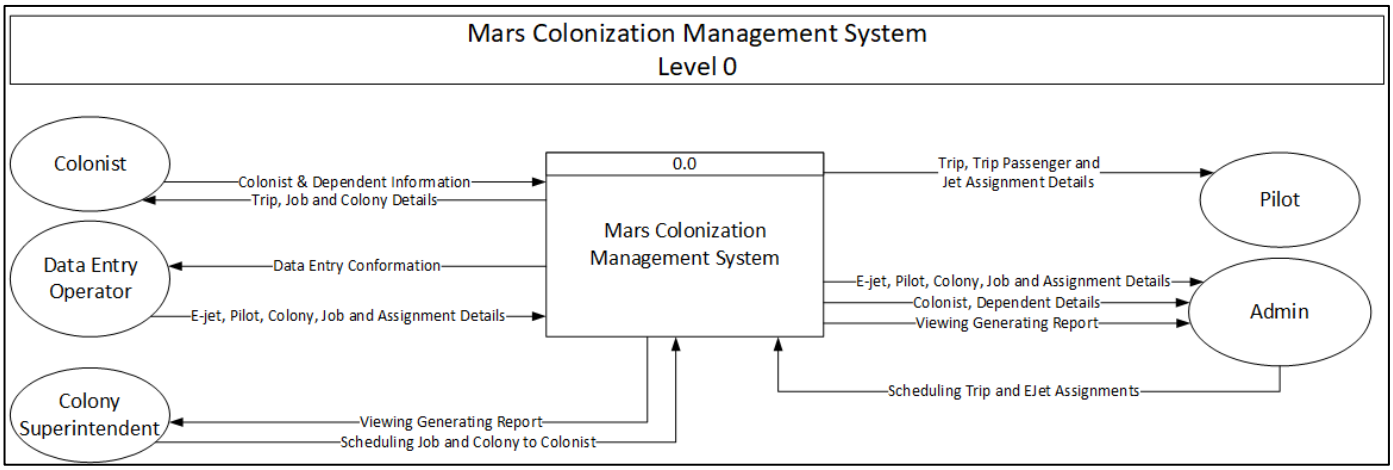
Class Diagram

Class Diagram is a type of UML Diagram which visually represents the structure and relationships of classes and interfaces in a system. It is also a fundamental tool in OOP analysis (Visual Paradigm, *What is Class Diagram*)



Data Flow Diagram (DFD)

DFD Level 0



DFD Level 1