

Kohl Johnson

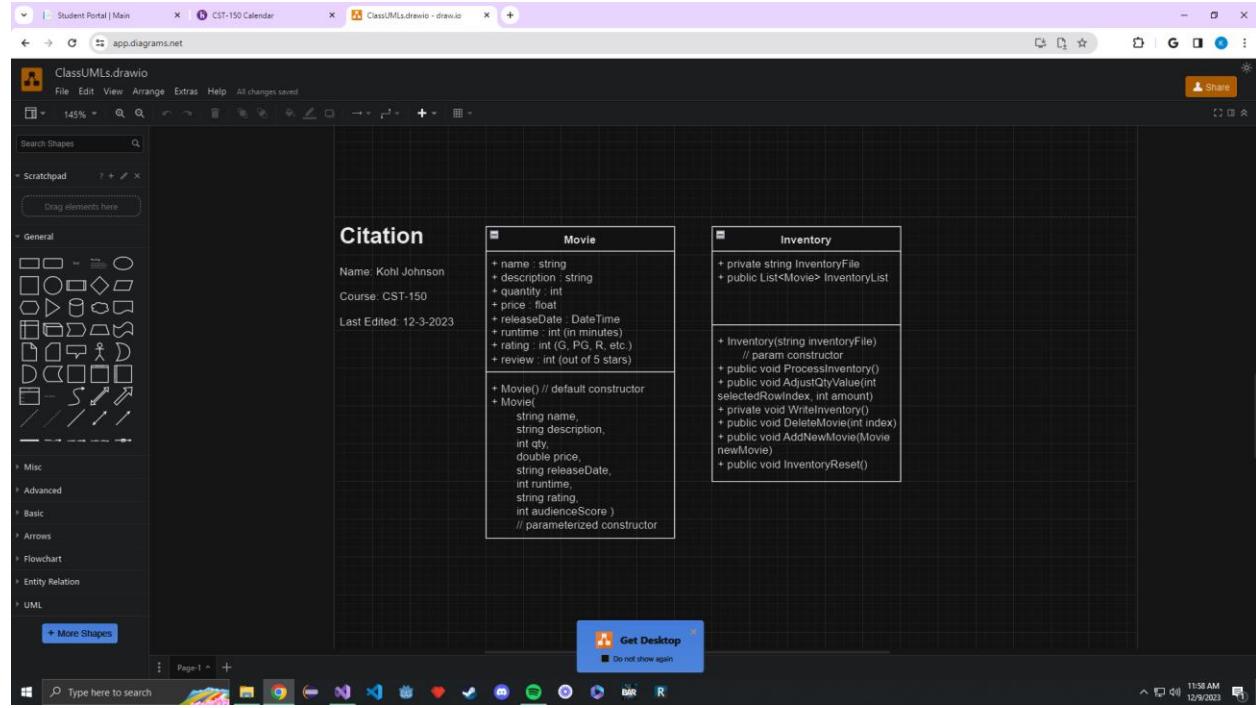
CST-150

12-09-2023

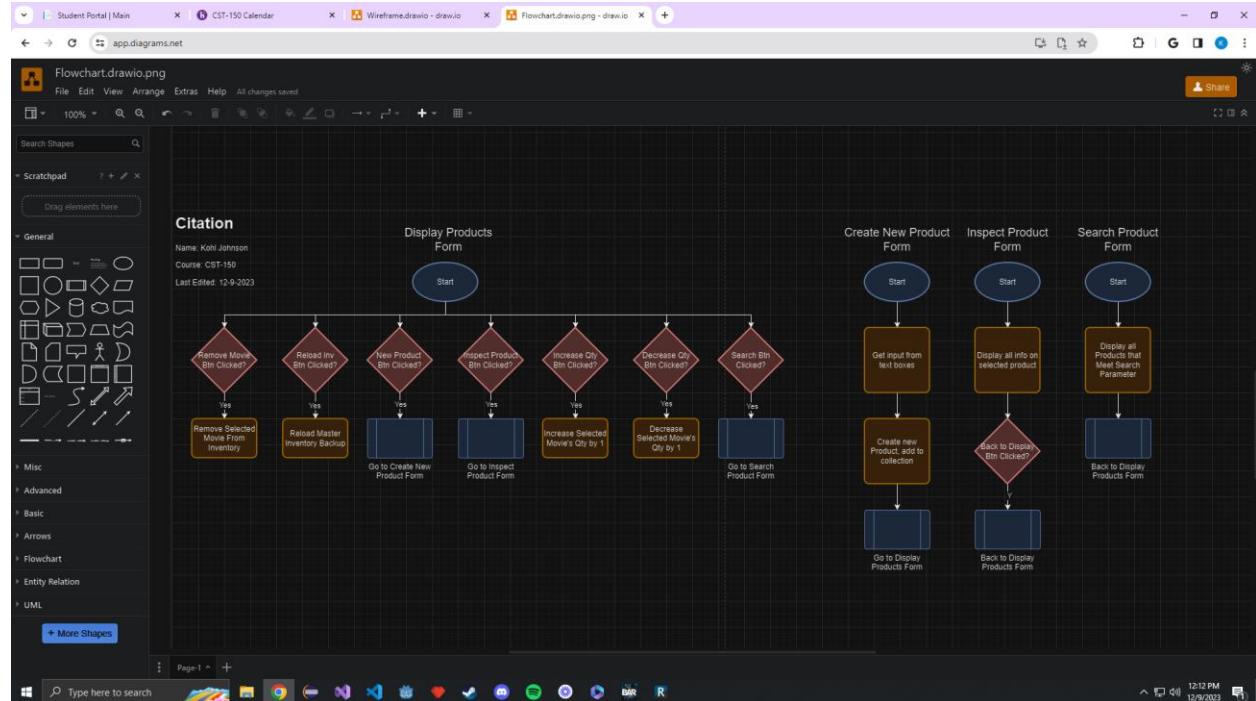
Milestone 7

**Loom Video Link:** <https://www.loom.com/share/ba4ea79413c1487baa74a369f5a50906?sid=3f481910-5dc1-44b9-8dae-cb0448c23659>

## Class UMLs Screenshot

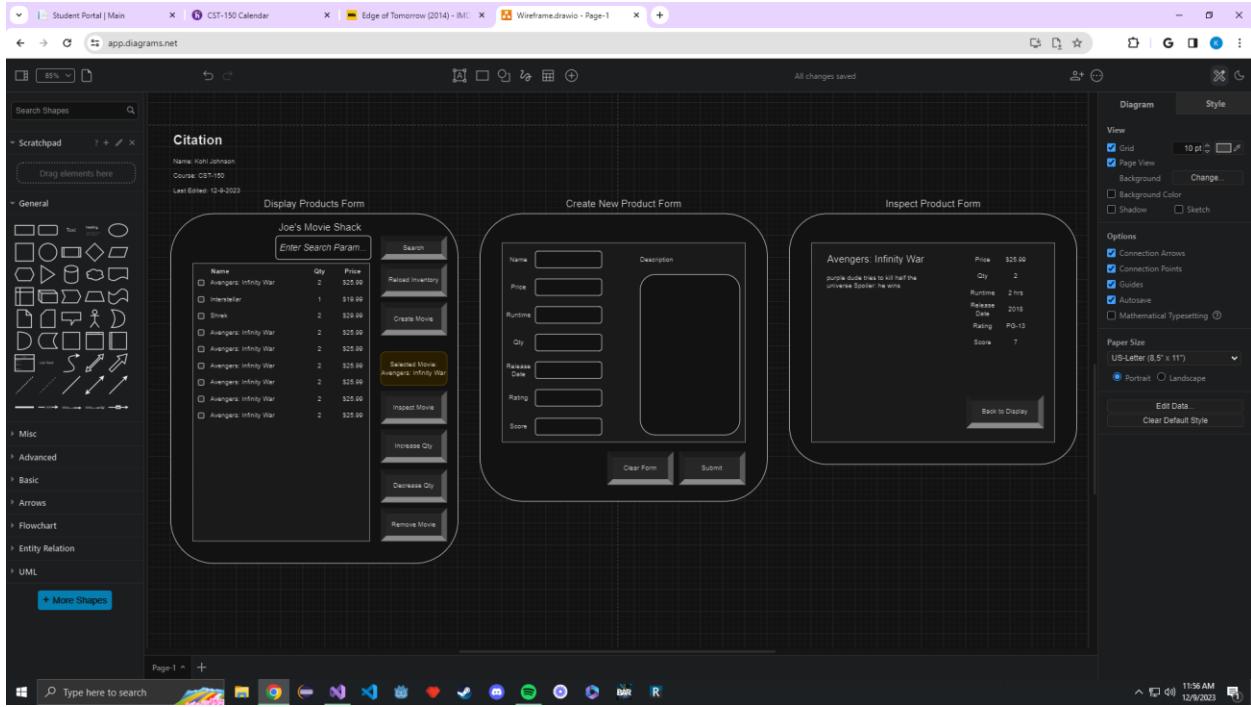


## Flowchart Screenshot



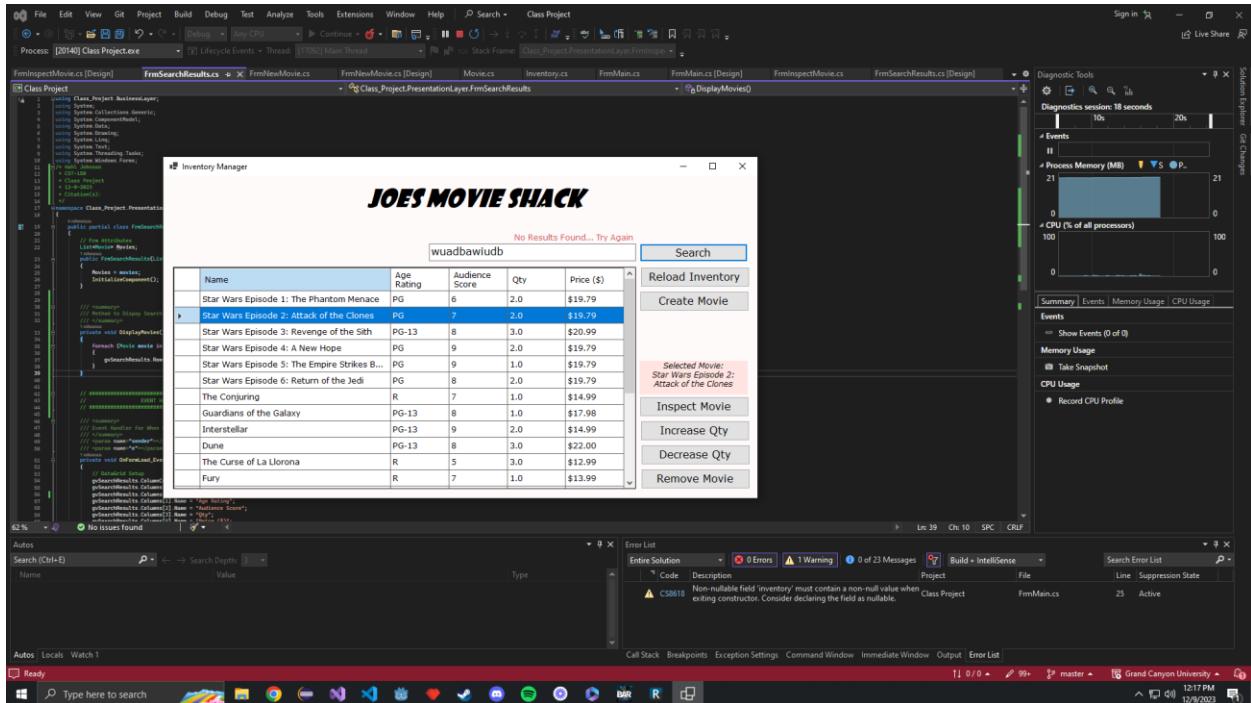
Everything was changed to be honest.

## Wireframe Screenshot



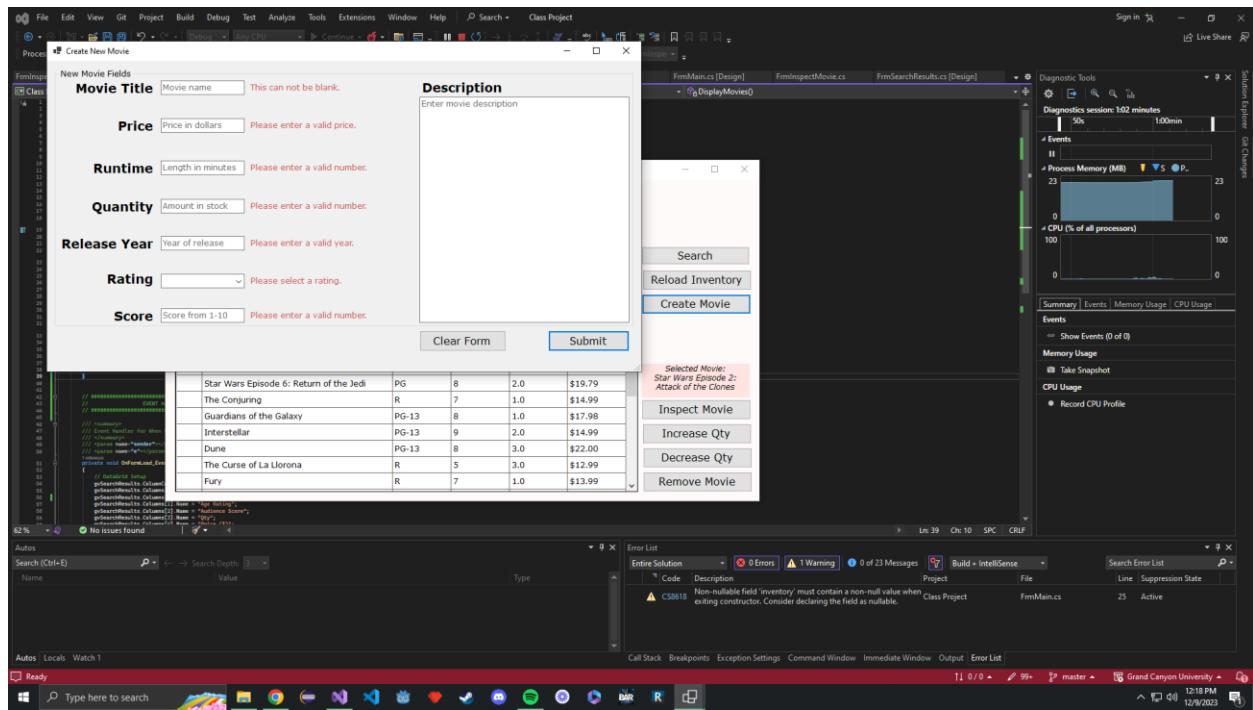
Added some more buttons on the Display Products Form

## FrmMain Post Pop Screenshot



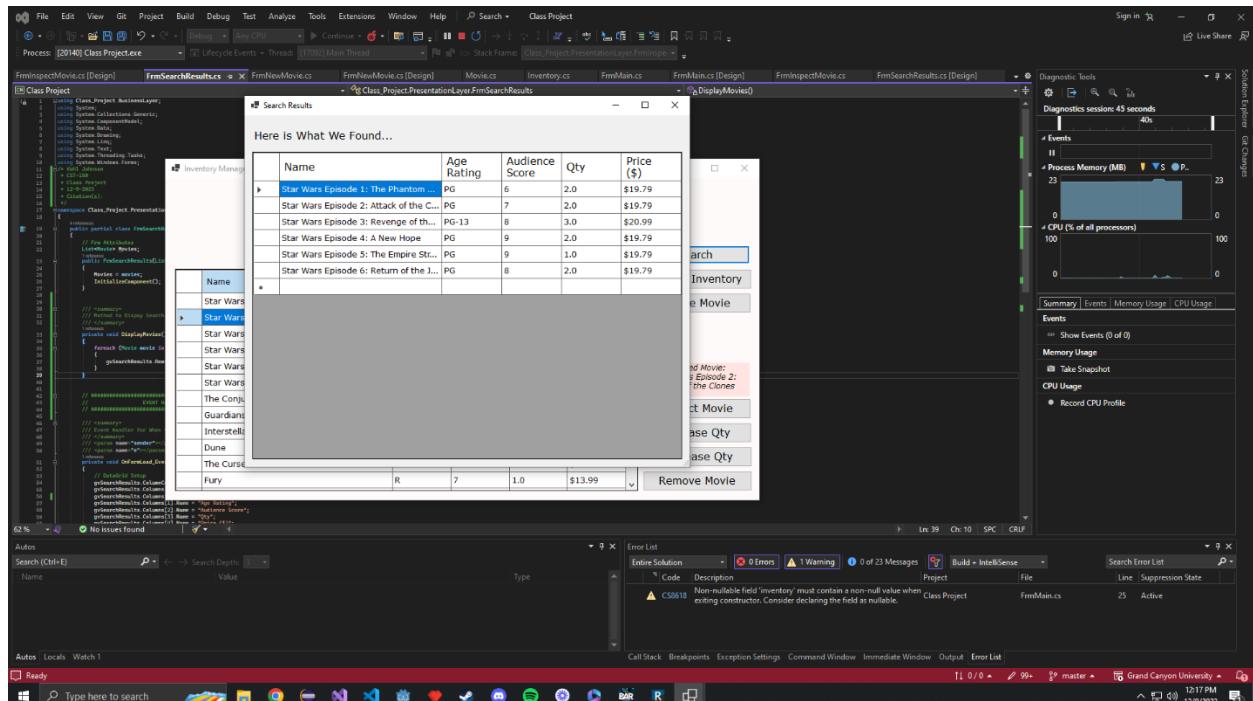
Shows the result of search parameter not meeting any of the movie names. Displays the text above the search text box saying “No Results Found...”

## FrmNewProduct Post Pop Screenshot



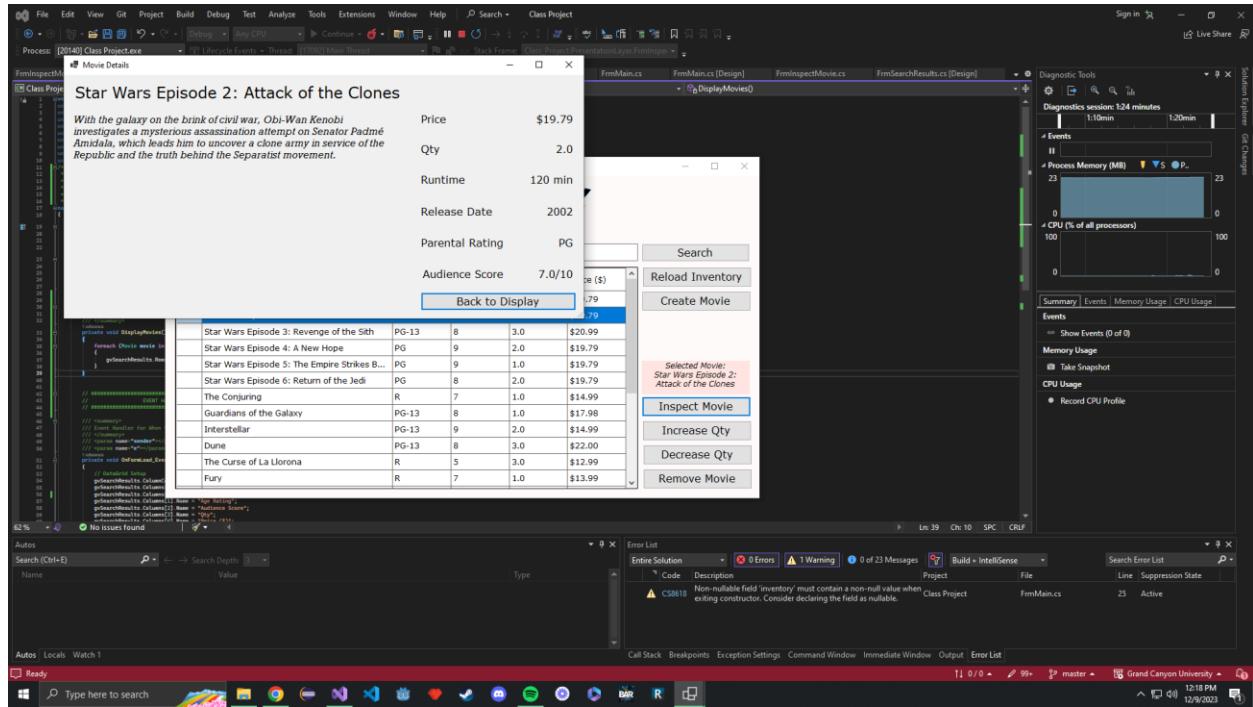
Shows most of the error messages that could pop up when creating a new movie object.

## FrmSearch Post Pop Screenshot



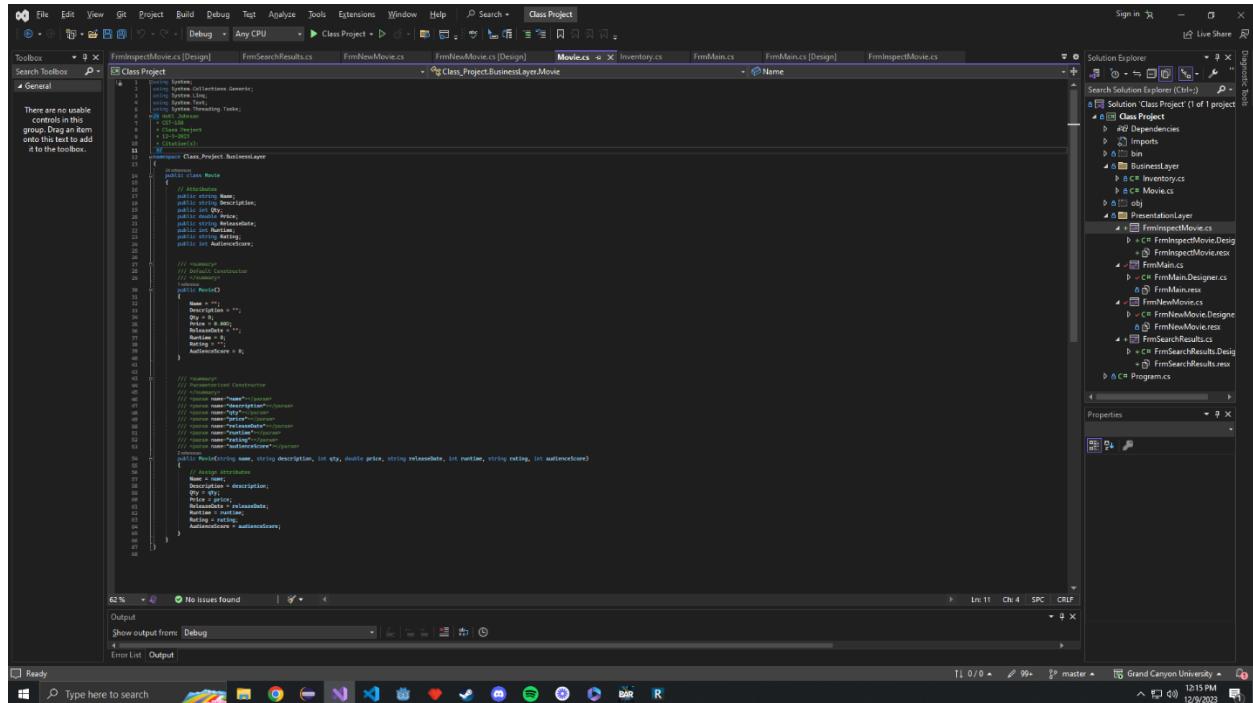
Shows the result of searching for "St", of course, only Star Wars movies pop up.

## **FrmlnSpectMovie Post Pop Screenshot**



Shows the form for inspecting a single movie. Displays all the attributes associated with the Movie class.

## Movie Class Code Screenshot



## Inventory Class Screenshots

The screenshot shows a Microsoft Visual Studio interface with the following details:

- Project:** Class Project
- Code Editor:** FfrmInspectMovie.cs [Design] (C#)
- Code Content:** The code handles reading CSV files for inventory and movies, and updates a database. It includes imports for System, System.Collections.Generic, System.Linq, System.Text, System.Threading.Tasks, and System.IO.
- Solution Explorer:** Shows the project structure:
  - Solution Class Project (1 of 1 project)
  - Class Project
    - Dependencies
    - Imports
    - BusinessLayer
      - Inventory.cs
      - Movies.cs
    - obj
    - PresentationLayer
      - FrmInspectMovie.cs
      - FrmInspectMovie.csproj
      - FrmMain.cs
      - FrmMain.csproj
      - FrmNewMovie.cs
      - FrmNewMovie.csproj
      - FrmSearchResults.cs
      - FrmSearchResults.csproj
- Properties:** Available on the right side of the interface.
- Output:** Shows build output and error logs.
- Task List:** Shows no issues found.
- Toolbox:** General tab is selected.
- Status Bar:** Displays file paths, line numbers, and other system information.



## FrmMain Code Screenshots

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search. The title bar indicates "Class Project". The left sidebar has sections for Toolbox, Search Toolbox, General, and a note about unavailable controls. The main area is the code editor with FrmMain.cs selected. The code is a Windows application using the Windows Forms framework. It includes methods for handling various events such as button clicks (e.g., BtInspectMovie\_Click) and list item selection changes (e.g., LstMovie\_SelectedIndexChanged). The code uses delegates and events to interact with other forms like FrmSearchResults and FrmNewMovie. The Solution Explorer on the right shows the project structure with files like FrmInspectMovie.cs, FrmSearchResults.cs, and FrmNewMovie.cs. The status bar at the bottom right shows the date as 12/29/2023.

## FrmNewProduct Code Screenshots

```

using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Threading.Tasks;
using Class_Project.BusinessLayer;
using Class_Project.PresentationLayer;
using System.Windows.Forms;
public partial class FrmNewMovie : Form
{
    public FrmNewMovie()
    {
        InitializeComponent();
        Initialize();
    }
    private void Initialize()
    {
        // Set Error Messages to Hidden
        UnderlineTitle.Visible = false;
        UnderlineDescription.Visible = false;
        UnderlinePrice.Visible = false;
        UnderlineQuantity.Visible = false;
        UnderlineRating.Visible = false;
        UnderlineGenre.Visible = false;
    }
    // ***** METHODS *****
    // ***** DON'T REMOVE METHODS *****
    // ***** DO NOT MOVE ANYTHING *****

    // Method to handle all logic with creating new Movie (will break apart into separate methods later on)
    // ***** DO NOT MOVE ANYTHING *****

    protected void btnSubmit_Click(object sender, EventArgs e)
    {
        // Variable Declaration
        string name = "";
        string description = "";
        float price;
        int quantity;
        string rating = "";
        string genre = "";
        int errorCount = 0;
        // Validation
        // Price Rule
        if (txtTitle.Text != "")
        {
            if (string.IsNullOrWhiteSpace(txtTitle.Text))
            {
                UnderlineTitle.Visible = true;
                errorCount++;
            }
            else
            {
                UnderlineTitle.Text = "This can not be blank.";
                UnderlineTitle.Visible = false;
                errorCount--;
            }
        }
        else
        {
            UnderlineTitle.Text = "Please enter a valid title";
            UnderlineTitle.Visible = true;
            errorCount++;
        }
        // Movie Description
        if (txtDescription.Text == "")
        {
            UnderlineDescription.Visible = true;
            errorCount++;
        }
        else
        {
            UnderlineDescription.Text = "No description given";
            UnderlineDescription.Visible = false;
            errorCount--;
        }
        // Movie Price
        if (!float.TryParse(txtPrice.Text, out price))
        {
            UnderlinePrice.Text = "Please enter a valid price";
            UnderlinePrice.Visible = true;
            errorCount++;
        }
        else
        {
            UnderlinePrice.Visible = false;
        }
        // Movie Rating
        if (!float.TryParse(txtRating.Text, out rating))
        {
            UnderlineRating.Text = "Please enter a valid number";
            UnderlineRating.Visible = true;
            errorCount++;
        }
        else
        {
            UnderlineRating.Visible = false;
        }
        // Movie Quantity
        if (!int.TryParse(txtQuantity.Text, out quantity))
        {
            UnderlineQuantity.Text = "Please enter a valid number";
            UnderlineQuantity.Visible = true;
            errorCount++;
        }
        else
        {
            UnderlineQuantity.Visible = false;
        }
    }
}

```

```

using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Threading.Tasks;
using Class_Project.BusinessLayer;
using Class_Project.PresentationLayer;
using System.Windows.Forms;
public partial class FrmNewMovie : Form
{
    public FrmNewMovie()
    {
        InitializeComponent();
        Initialize();
    }
    private void Initialize()
    {
        // Set Error Messages to Hidden
        UnderlineTitle.Visible = false;
        UnderlineDescription.Visible = false;
        UnderlinePrice.Visible = false;
        UnderlineQuantity.Visible = false;
        UnderlineRating.Visible = false;
        UnderlineGenre.Visible = false;
    }
    // ***** METHODS *****
    // ***** DON'T REMOVE METHODS *****
    // ***** DO NOT MOVE ANYTHING *****

    // Method to handle all logic with creating new Movie (will break apart into separate methods later on)
    // ***** DO NOT MOVE ANYTHING *****

    protected void btnSubmit_Click(object sender, EventArgs e)
    {
        // Variable Declaration
        string name = "";
        string description = "";
        float price;
        int quantity;
        string rating = "";
        string genre = "";
        int errorCount = 0;
        // Validation
        // Price Rule
        if (txtTitle.Text != "")
        {
            if (string.IsNullOrWhiteSpace(txtTitle.Text))
            {
                UnderlineTitle.Visible = true;
                errorCount++;
            }
            else
            {
                UnderlineTitle.Text = "This can not be blank.";
                UnderlineTitle.Visible = false;
                errorCount--;
            }
        }
        else
        {
            UnderlineTitle.Text = "Please enter a valid title";
            UnderlineTitle.Visible = true;
            errorCount++;
        }
        // Movie Description
        if (txtDescription.Text == "")
        {
            UnderlineDescription.Visible = true;
            errorCount++;
        }
        else
        {
            UnderlineDescription.Text = "No description given";
            UnderlineDescription.Visible = false;
            errorCount--;
        }
        // Movie Price
        if (!float.TryParse(txtPrice.Text, out price))
        {
            UnderlinePrice.Text = "Please enter a valid price";
            UnderlinePrice.Visible = true;
            errorCount++;
        }
        else
        {
            UnderlinePrice.Visible = false;
        }
        // Movie Rating
        if (!float.TryParse(txtRating.Text, out rating))
        {
            UnderlineRating.Text = "Please enter a valid number";
            UnderlineRating.Visible = true;
            errorCount++;
        }
        else
        {
            UnderlineRating.Visible = false;
        }
        // Movie Quantity
        if (!int.TryParse(txtQuantity.Text, out quantity))
        {
            UnderlineQuantity.Text = "Please enter a valid number";
            UnderlineQuantity.Visible = true;
            errorCount++;
        }
        else
        {
            if (quantity < 0)
            {
                UnderlineQuantity.Text = "Movie must be greater than 0";
                UnderlineQuantity.Visible = true;
                errorCount++;
            }
            else
            {
                UnderlineQuantity.Visible = false;
            }
        }
    }
}

```

The screenshot shows a Microsoft Visual Studio interface with the following details:

- Solution Explorer:** Displays the solution structure for "Class Project" (1 of 1 project). It includes the Class Project, BusinessLayer, Inventory.cs, Movies.cs, MainForm.cs, and PresentationLayer. The PresentationLayer folder contains F# files: F#inspectMovie.cs, F#inspectMovie.designer.cs, F#inspectMovie.resx, F#MainDesigner.cs, F#NewMovie.cs, F#NewMovie.designer.cs, F#SearchResults.cs, F#SearchResults.designer.cs, and Program.cs.
- Toolbox:** Shows the "General" tab with a note: "There are no enabled controls in this group. Drag an item onto this text to add it to the toolbox."
- Code Editor:** Displays F# code for the F#inspectMovie.cs file. The code includes methods for movie creation, clearing form, and handling button click events. It uses F# types like Movie and InventoryItem, and functions like AddMovie and GetMovie.
- Status Bar:** Shows the current file as F#inspectMovie.cs [Design], line 14, character 13, and the commit message "12/15/2023".

## FrmSearch Code Screenshot

The screenshot shows a Microsoft Visual Studio interface with the following details:

- File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help** menu items.
- Search** button.
- Class Project** tab is selected in the tabs bar.
- Toolbox** window is open, showing a general category with controls like `Panel`, `Image`, `Label`, etc.
- Search Toolbox** dropdown is open, showing results for "Movie".
- There are no usable controls in this group. Drag them onto the form to add it to the toolbox.** message.
- Code Editor**: The file `FrmInspectMovies.cs` is open, showing C# code for a Windows Form application. The code includes imports for `System`, `System.Collections.Generic`, `System.Linq`, `System.Threading.Tasks`, and `System.Windows.Forms`. It defines a `Form1` class with methods like `DisplayMovies` and `DisplayMovie`.
- Solution Explorer**: Shows the `Class Project` solution with files like `FrmMain.cs`, `FrmMain.Designer.cs`, `FrmNewMovie.cs`, `FrmNewMovie.Designer.cs`, `FrmSearchResults.cs`, `FrmSearchResults.Designer.cs`, and `Program.cs`.
- Properties** window is visible on the right.
- Status Bar**: Shows "Ready", "Type here to search", and system icons.
- Taskbar**: Shows browser, file explorer, and other application icons.

## FrmlnSpectMovie Code Screenshot

The screenshot shows the Microsoft Visual Studio IDE interface. The title bar displays "File Edit View Gb Project Build Debug Test Analyze Tools Extensions Window Help Search Class Project". The main area is the code editor with the file "FrmInspectMovie.cs" open, showing C# code for a Windows Form application. The code includes imports for System, System.Collections.Generic, System.ComponentModel, System.Drawing, System.Text, System.Threading.Tasks, and System.Windows.Forms. It defines a class "FrmInspectMovie" with methods like "DisplayMovieDetails" and event handlers for "FormLoad" and "Button1Click". The right side of the interface features the Solution Explorer, which lists the project structure: "Solution Class Project (1 of 1 project)" containing "FrmInspectMovie.cs", "FrmMain.cs", "FrmSearchResults.cs", and "Program.cs". Below the Solution Explorer is the Properties window. At the bottom, the Task List, Status Bar (showing "Ready" and "12:12 PM 12/9/2023"), and the Start button are visible.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Class_Project.PresentationLayer
{
    public partial class FrmInspectMovie : Form
    {
        Movie InspectedMovie;
        public FrmInspectMovie(Movie movie)
        {
            InspectedMovie = movie;
            InitializeComponent();
        }

        ///////////////////////////////////////////////////
        // Method to Display Movie Details
        ///////////////////////////////////////////////////
        private void DisplayMovieDetails()
        {
            lblName.Text = InspectedMovie.Name;
            lblGenre.Text = InspectedMovie.Genre;
            lblRating.Text = String.Format("({0}) {1}", InspectedMovie.Rating);
            lblDuration.Text = String.Format("{0} min", InspectedMovie.Duration);
            lblActing.Text = InspectedMovie.Actors;
            lblCritic.Text = InspectedMovie.Rating;
            lblScore.Text = String.Format("({0}/10)", InspectedMovie.AudienceScore);
        }

        ///////////////////////////////////////////////////
        // EVENT HANDLER METHODS
        ///////////////////////////////////////////////////

        ///////////////////////////////////////////////////
        // Event Handler for Form Loading
        ///////////////////////////////////////////////////
        private void FrmInspectMovie_Load(object sender, EventArgs e)
        {
            DisplayMovieDetails();
        }

        ///////////////////////////////////////////////////
        // Event Handler for Button Click
        ///////////////////////////////////////////////////
        private void Button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

### **Analysis on Global Awareness, Perspectives, and Ethics**

Learning common naming conventions, formatting, and “flow” I think tie well into global awareness and perspectives. Learning the industry standards will help me not only with having a successful interview, but also with working in a team and ensuring that my code is easily readable and comprehensible for other developers. With the ethics part, even though is not necessarily applicable to this class project, ensuring the safety of user information and being honest with what data we are collecting is paramount. Another thing with ethics that also ties into using industry standards, is writing good quality code in terms of security.