Unicom TIC Management

Introduction

The **Unicom TIC Management System (UMS)** is a **simple desktop-based application** built using **C# WinForms**, designed for managing the **basic operations of a college**. It is especially suitable for **beginners** who are learning how to build database-connected applications using the **MVC (Model-View-Controller)** architecture.

- The following things can be managed through this system
  - Courses
  - Subject
  - Students
  - Exams & Marks
  - Time table

Login System

There is a Login Form in the **Unicom Management System**. Users can log in by providing their **Username, Password**, and **Role** accurately.

- **Role are** :-
  - Admin
  - Student
  - Staff
  - Lecturer

Each user must select and enter their correct Username, correct Password, and correct Role.

Once the login is verified:

You can go to the appropriate Main Form for that Role:-

Admin => Admin Main Form

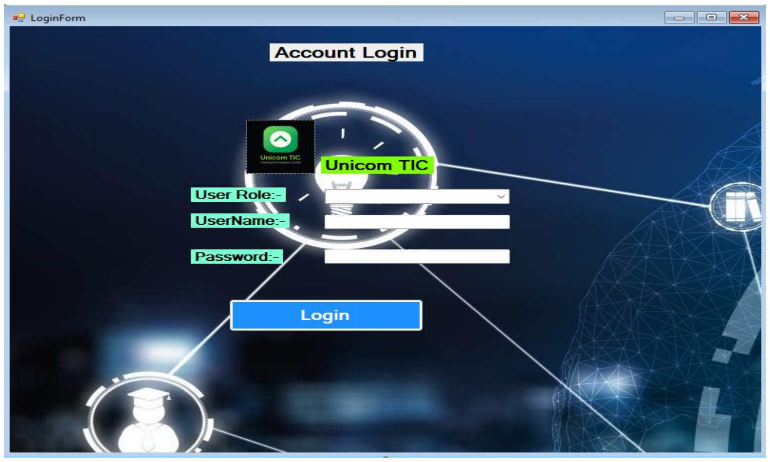
Student => Student Main Form

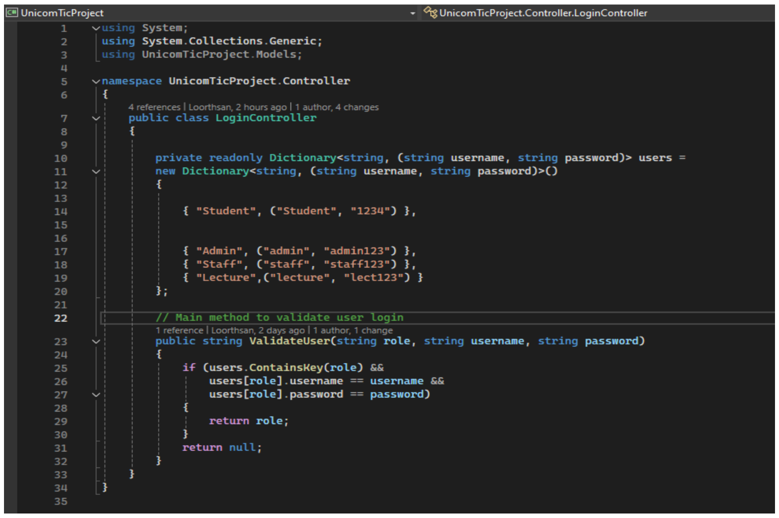
Staff => Staff Main Form

Lecturer => Lecturer Main Form

Unicom TIC Management

Login Form Design

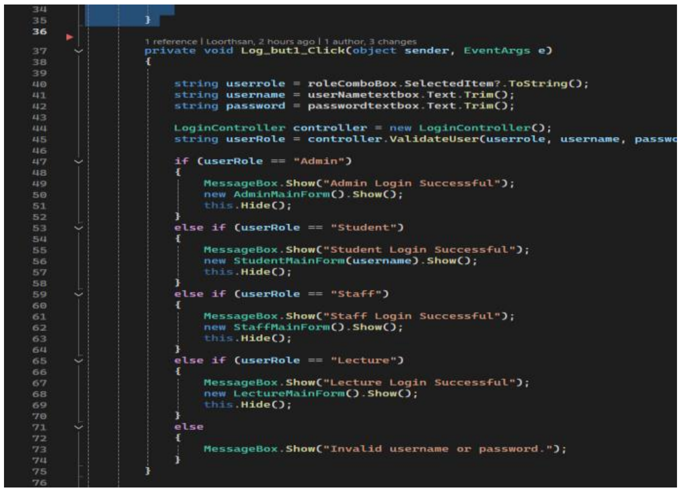




```
UnicomTicProject                                   UnicomTicProject.Controller.LoginController
using System;
using System.Collections.Generic;
using UnicomTicProject.Models;

namespace UnicomTicProject.Controller
{
    4 references | Loorthsan, 2 hours ago | 1 author, 4 changes
    public class LoginController
    {

        private readonly Dictionary<string, (string username, string password)> users =
        new Dictionary<string, (string username, string password)>()
        {

            { "Student", ("Student", "1234") },


            { "Admin", ("admin", "admin123") },
            { "Staff", ("staff", "staff123") },
            { "Lecture",("lecture", "lect123") }
        };

        // Main method to validate user login
        1 reference | Loorthsan, 2 days ago | 1 author, 1 change
        public string ValidateUser(string role, string username, string password)
        {
            if (users.ContainsKey(role) &&
                users[role].username == username &&
                users[role].password == password)
            {
                return role;
            }
            return null;
        }
    }
}
```

Login Controller

Login View Code



```
        }

        1 reference | Loorthsan, 2 hours ago | 1 author, 3 changes
        private void Log_but1_Click(object sender, EventArgs e)
        {

            string userrole = roleComboBox.SelectedItem?.ToString();
            string username = userNametextbox.Text.Trim();
            string password = passwordtextbox.Text.Trim();

            LoginController controller = new LoginController();
            string userRole = controller.ValidateUser(userrole, username, passwo

            if (userRole == "Admin")
            {
                MessageBox.Show("Admin Login Successful");
                new AdminMainForm().Show();
                this.Hide();
            }
            else if (userRole == "Student")
            {
                MessageBox.Show("Student Login Successful");
                new StudentMainForm(username).Show();
                this.Hide();
            }
            else if (userRole == "Staff")
            {
                MessageBox.Show("Staff Login Successful");
                new StaffMainForm().Show();
                this.Hide();
            }
            else if (userRole == "Lecture")
            {
                MessageBox.Show("Lecture Login Successful");
                new LectureMainForm().Show();
                this.Hide();
            }
            else
            {
                MessageBox.Show("Invalid username or password.");
            }
        }
```

Unicom TIC Management

Admin Main Form

Once the **Admin** logs in, they are redirected to the **Admin Dashboard/Main Form**, which contains the following key buttons (features):

1. Manage Student

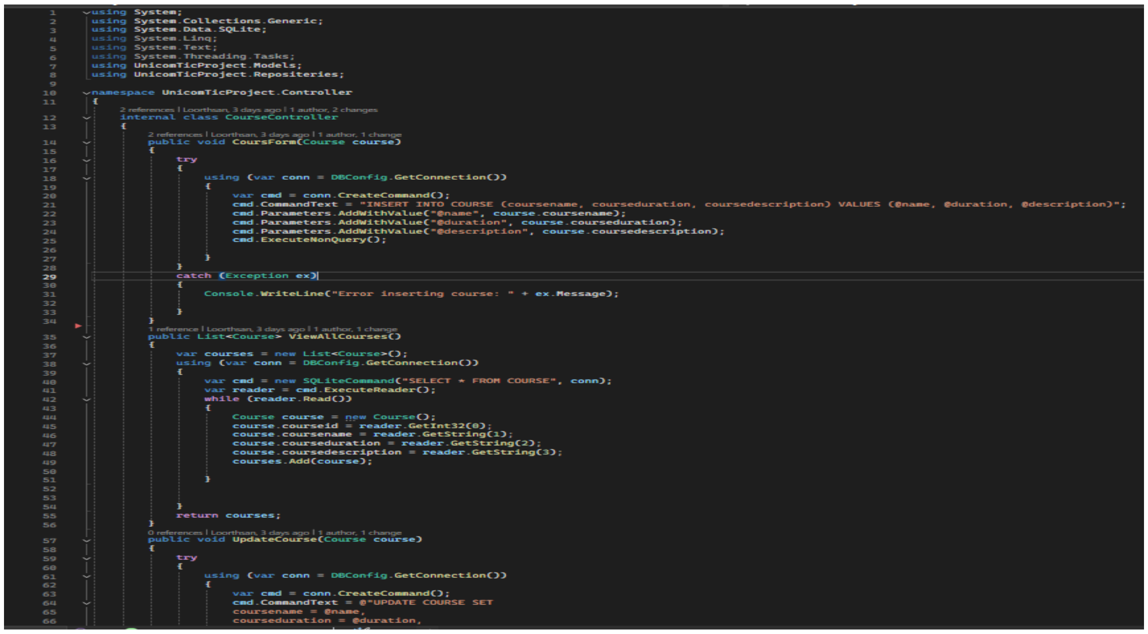
- Clicking this button opens the All Students Form.
- The admin can add new students to the college by entering details like:
  - Name
  - Date of Birth (DOB)
  - Course
  - And other necessary details
- Existing student records can be updated if any information changes.
- The admin can also delete/remove a student from the system if needed.

2. Course

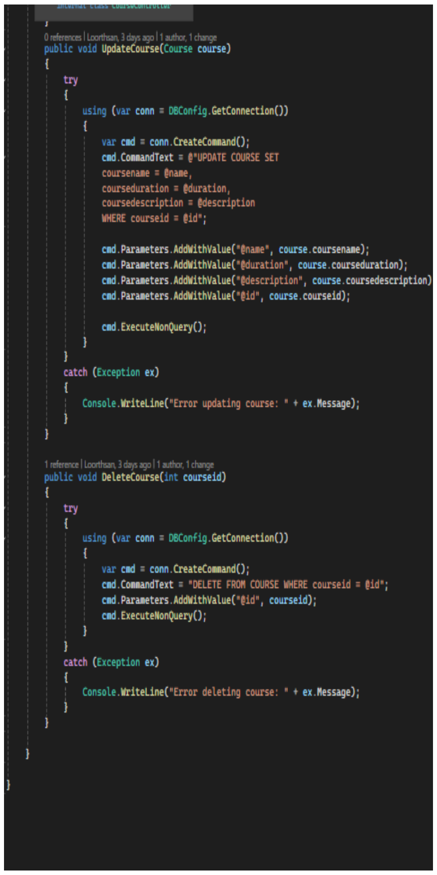
- Clicking this button opens the Course Form.
- In this form, the admin can enter:
  - Course Name
  - Duration
  - Description
- Admin can:
  - Add a new course
  - Update existing course details
  - Delete a course if it is no longer offered

Unicom TIC Management

Course Controller Codding



```
using System;
using System.Collections.Generic;
using System.Data.SQLite;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using UnicomTicProject.Models;
using UnicomTicProject.Repositeries;

namespace UnicomTicProject.Controller
{
    2 references | Loorthsan, 3 days ago | 1 author, 2 changes
    internal class CourseController
    {
        2 references | Loorthsan, 3 days ago | 1 author, 1 change
        public void CoursForm(Course course)
        {
            try
            {
                using (var conn = DBConfig.GetConnection())
                {
                    var cmd = conn.CreateCommand();
                    cmd.CommandText = "INSERT INTO COURSE (coursename, courseduration, coursedescription) VALUES (@name, @duration, @description)";
                    cmd.Parameters.AddWithValue("@name", course.coursename);
                    cmd.Parameters.AddWithValue("@duration", course.courseduration);
                    cmd.Parameters.AddWithValue("@description", course.coursedescription);
                    cmd.ExecuteNonQuery();

                }
            }
            catch (Exception ex)
            {
                Console.WriteLine("Error inserting course: " + ex.Message);

            }
        }
        1 reference | Loorthsan, 3 days ago | 1 author, 1 change
        public List<Course> ViewAllCourses()
        {
            var courses = new List<Course>();
            using (var conn = DBConfig.GetConnection())
            {
                var cmd = new SQLiteCommand("SELECT * FROM COURSE", conn);
                var reader = cmd.ExecuteReader();
                while (reader.Read())
                {
                    Course course = new Course();
                    course.courseid = reader.GetInt32(0);
                    course.coursename = reader.GetString(1);
                    course.courseduration = reader.GetString(2);
                    course.coursedescription = reader.GetString(3);
                    courses.Add(course);

                }


            }
            return courses;
        }
        0 references | Loorthsan, 3 days ago | 1 author, 1 change
        public void UpdateCourse(Course course)
        {
            try
            {
                using (var conn = DBConfig.GetConnection())
                {
                    var cmd = conn.CreateCommand();
                    cmd.CommandText = @"UPDATE COURSE SET
                    coursename = @name,
                    courseduration = @duration,
```

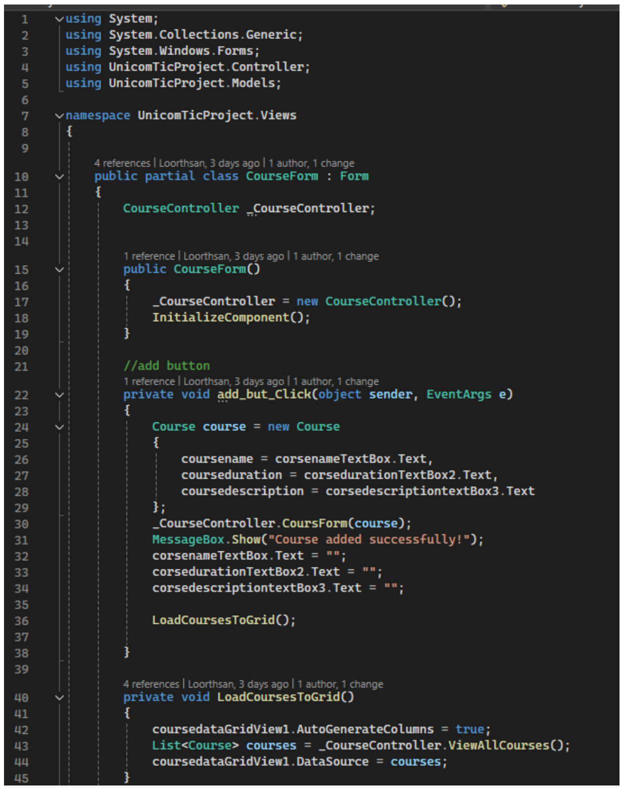


```
internal class CourseController
}
0 references | Loorthsan, 3 days ago | 1 author, 1 change
public void UpdateCourse(Course course)
{
    try
    {
        using (var conn = DBConfig.GetConnection())
        {
            var cmd = conn.CreateCommand();
            cmd.CommandText = @"UPDATE COURSE SET
            coursename = @name,
            courseduration = @duration,
            coursedescription = @description
            WHERE courseid = @id";

            cmd.Parameters.AddWithValue("@name", course.coursename);
            cmd.Parameters.AddWithValue("@duration", course.courseduration);
            cmd.Parameters.AddWithValue("@description", course.coursedescription)
            cmd.Parameters.AddWithValue("@id", course.courseid);

            cmd.ExecuteNonQuery();
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error updating course: " + ex.Message);
    }
}
1 reference | Loorthsan, 3 days ago | 1 author, 1 change
public void DeleteCourse(int courseid)
{
    try
    {
        using (var conn = DBConfig.GetConnection())
        {
            var cmd = conn.CreateCommand();
            cmd.CommandText = "DELETE FROM COURSE WHERE courseid = @id";
            cmd.Parameters.AddWithValue("@id", courseid);
            cmd.ExecuteNonQuery();
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error deleting course: " + ex.Message);
    }
}

}

}
```

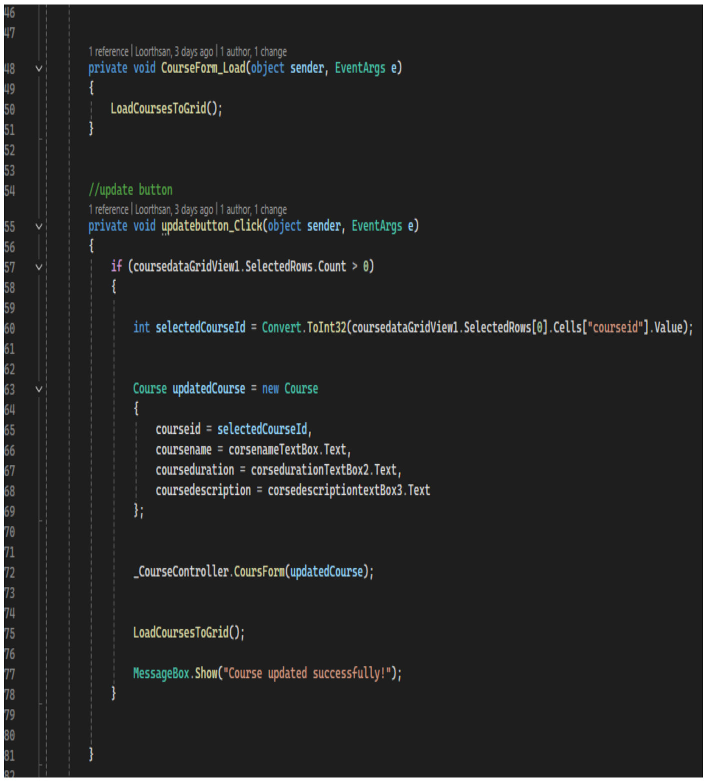
```
internal class CourseController
}
0 references | Loorthsan, 3 days ago | 1 author, 1 change
public void UpdateCourse(Course course)
{
    try
    {
        using (var conn = DBConfig.GetConnection())
        {
            var cmd = conn.CreateCommand();
            cmd.CommandText = @"UPDATE COURSE SET
            coursename = @name,
            courseduration = @duration,
            coursedescription = @description
            WHERE courseid = @id";

            cmd.Parameters.AddWithValue("@name", course.coursename);
            cmd.Parameters.AddWithValue("@duration", course.courseduration);
            cmd.Parameters.AddWithValue("@description", course.coursedescription)
            cmd.Parameters.AddWithValue("@id", course.courseid);

            cmd.ExecuteNonQuery();
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error updating course: " + ex.Message);
    }
}
1 reference | Loorthsan, 3 days ago | 1 author, 1 change
public void DeleteCourse(int courseid)
{
    try
    {
        using (var conn = DBConfig.GetConnection())
        {
            var cmd = conn.CreateCommand();
            cmd.CommandText = "DELETE FROM COURSE WHERE courseid = @id";
            cmd.Parameters.AddWithValue("@id", courseid);
            cmd.ExecuteNonQuery();
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error deleting course: " + ex.Message);
    }
}

}

}
```

Unicom TIC Management

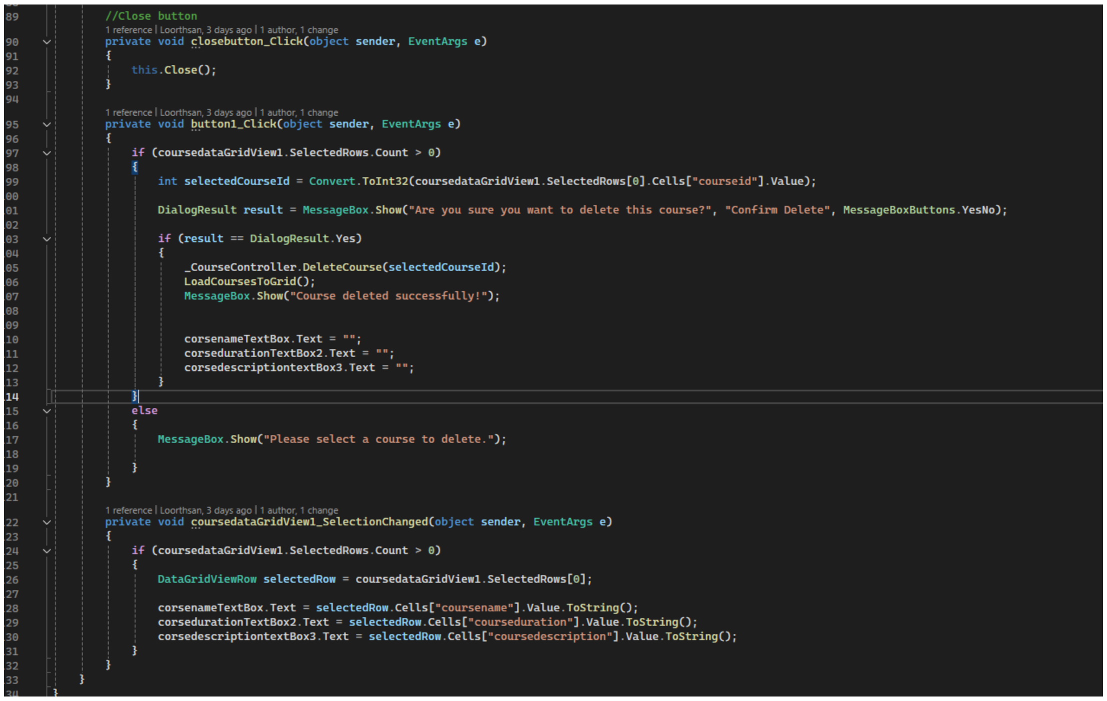
Course View code



```csharp
using System;
using System.Collections.Generic;
using System.Windows.Forms;
using UnicomTicProject.Controller;
using UnicomTicProject.Models;

namespace UnicomTicProject.Views
{

    4 references | Loorthsan, 3 days ago | 1 author, 1 change
    public partial class CourseForm : Form
    {
        CourseController _CourseController;


        1 reference | Loorthsan, 3 days ago | 1 author, 1 change
        public CourseForm()
        {
            _CourseController = new CourseController();
            InitializeComponent();
        }

        //add button
        1 reference | Loorthsan, 3 days ago | 1 author, 1 change
        private void add_but_Click(object sender, EventArgs e)
        {
            Course course = new Course
            {
                coursename = corsenameTextBox.Text,
                courseduration = corsedurationTextBox2.Text,
                coursedescription = corsedescriptiontextBox3.Text
            };
            _CourseController.CoursForm(course);
            MessageBox.Show("Course added successfully!");
            corsenameTextBox.Text = "";
            corsedurationTextBox2.Text = "";
            corsedescriptiontextBox3.Text = "";

            LoadCoursesToGrid();

        }

        4 references | Loorthsan, 3 days ago | 1 author, 1 change
        private void LoadCoursesToGrid()
        {
            coursedataGridView1.AutoGenerateColumns = true;
            List<Course> courses = _CourseController.ViewAllCourses();
            coursedataGridView1.DataSource = courses;
        }
```



```csharp


        1 reference | Loorthsan, 3 days ago | 1 author, 1 change
        private void CourseForm_Load(object sender, EventArgs e)
        {
            LoadCoursesToGrid();
        }


        //update button
        1 reference | Loorthsan, 3 days ago | 1 author, 1 change
        private void updatebutton_Click(object sender, EventArgs e)
        {
            if (coursedataGridView1.SelectedRows.Count > 0)
            {

                int selectedCourseId = Convert.ToInt32(coursedataGridView1.SelectedRows[0].Cells["courseid"].Value);


                Course updatedCourse = new Course
                {
                    courseid = selectedCourseId,
                    coursename = corsenameTextBox.Text,
                    courseduration = corsedurationTextBox2.Text,
                    coursedescription = corsedescriptiontextBox3.Text
                };


                _CourseController.CoursForm(updatedCourse);


                LoadCoursesToGrid();

                MessageBox.Show("Course updated successfully!");
            }



        }
```



```csharp
        //Close button
        1 reference | Loorthsan, 3 days ago | 1 author, 1 change
        private void closebutton_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        1 reference | Loorthsan, 3 days ago | 1 author, 1 change
        private void button1_Click(object sender, EventArgs e)
        {
            if (coursedataGridView1.SelectedRows.Count > 0)
            {
                int selectedCourseId = Convert.ToInt32(coursedataGridView1.SelectedRows[0].Cells["courseid"].Value);

                DialogResult result = MessageBox.Show("Are you sure you want to delete this course?", "Confirm Delete", MessageBoxButtons.YesNo);

                if (result == DialogResult.Yes)
                {
                    _CourseController.DeleteCourse(selectedCourseId);
                    LoadCoursesToGrid();
                    MessageBox.Show("Course deleted successfully!");


                    corsenameTextBox.Text = "";
                    corsedurationTextBox2.Text = "";
                    corsedescriptiontextBox3.Text = "";
                }
            }
            else
            {
                MessageBox.Show("Please select a course to delete.");

            }
        }

        1 reference | Loorthsan, 3 days ago | 1 author, 1 change
        private void coursedataGridView1_SelectionChanged(object sender, EventArgs e)
        {
            if (coursedataGridView1.SelectedRows.Count > 0)
            {
                DataGridViewRow selectedRow = coursedataGridView1.SelectedRows[0];

                corsenameTextBox.Text = selectedRow.Cells["coursename"].Value.ToString();
                corsedurationTextBox2.Text = selectedRow.Cells["courseduration"].Value.ToString();
                corsedescriptiontextBox3.Text = selectedRow.Cells["coursedescription"].Value.ToString();
            }
        }
    }
}
```

3. Exam List / Mark List

- Clicking this button opens the Exam Form.
- In this form, the admin can manage exam-related data including:
  - Student Name
  - Exam ID
  - Subject Name
  - Course Name
  - Marks
  - Exam Name
- Admin can:
  - Add new exam entries
  - Update existing exam records
  - **Delete** exams if necessary

4. Timetable

- Clicking this button opens the Timetable Form.
- In this form, the admin can:
  - Create class schedules/timetables
  - Allocate rooms such as labs or lecture halls
  - Assign subjects to specific times for organizing the daily academic schedule

Lecture Main Form

- When a lecturer logs in, they are taken to the **Lecturer Main Form**.
- Available buttons:
  - **All Students** – View list of students
  - **Exam List** – View all exam entries
  - **Mark List** – View students' marks
  - **Subject List** – View subjects assigne
  - **Timetable** – View class schedule

- Lecturers can **view and manage** some data, depending on the system permissions

Unicom TIC Management

Students Main Form

- When a student logs in, they are taken to the **Student Main Form**.
- The student can:
  - **View their Profile** (by clicking a button)
  - **View Marks** (Mark Form opens)
  - **View Timetable** (Timetable Form opens)
- The student can **only view** these forms – they **cannot add, update, or delete** any data.

Staff Main Form

- Like lecturers, staff also have their own **Staff Main Form**.
- It includes buttons similar to:
  - **Student List**
  - **Course List**
  - **Timetable**
- Mostly, Staff have **view and support roles** – helping with leave approval, student info support, etc.

- **Git hup My Project Link**

  https://github.com/Loorthsan/UMS-project.git

Summary

"My project includes role-based login for Admin, Student, Lecturer, and Staff. Each role has a separate main form with access to different features. Admin can manage everything, while other roles have limited viewing access. I have currently completed the major forms and functionalities like Course, Student, Exam, and Mark, and I'm still working on completing the full features."