**Unicom TIC Management System**

## Key Features by Module

### User Account Management

* Admin:
* Must first create user accounts (username, password, role) before adding Student, Staff, or Lecturer information.
* Can view, update, and delete all user accounts.

### Course & Subject Management

* Admin: Can add, update, delete, and view courses and subjects.
* Staff, Students, Lecturers: No access to manage or view this section.

### Student Management

* Admin: Can add, update, and delete student records.
* Student details are added after their user account is created.
* Students: Can view their own details only.

### Staff Management

* Admin: Can add, update, and delete staff records.
* Staff details include name and role.
* Staff user account must be created first.

### Lecturer Management

* Admin: Can add, update, delete, and view lecturer records.
* Lecturers are assigned to courses.
* Lecturer user account must be created first.

### Admin Management

* Admin: Can add other admin users using a dedicated form.

### Exam & Marks Management

* Admin / Staff / Lecturers: Can add and update exams and student marks.
* Students: Can view their own marks.

### Timetable Management

* Admin: Can add, update, and delete timetable entries (subject, time, room).
* Staff: Can add and update timetable entries (no delete).
* Lecturers / Students: Can view timetables (including labs/halls).

### Room Management

* Admin: Can add, update, and delete rooms (labs and lecture halls).

### Feedback System

* Students: Can submit anonymous feedback.
* Admin / Staff / Lecturers: Can view feedback.

## Technologies Used

* Visual Studio – for development and designing forms
* C# – main programming language used
* WinForms – used to build the graphical user interface
* SQLite – database engine to store all application data
* System.Data.SQLite (NuGet package) – integration between SQLite and C#
* Manual MVC Pattern – separation of UI, business logic, and data layers

## Challenges Faced and Solutions

* **Challenge**- Handling many-to-many table updating caused issues with duplicate data.
* **Solution**-Solved by adding conditional checks to prevent duplicate entries during updates.

# 1.Login

Figure Login Form UI

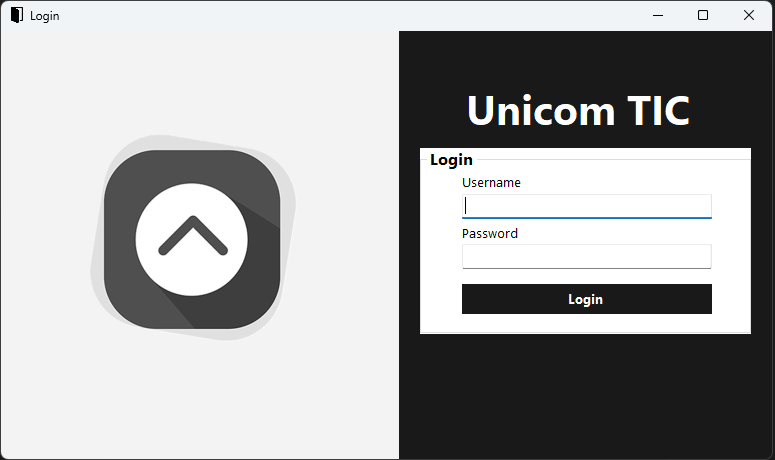


Figure Login Form Code

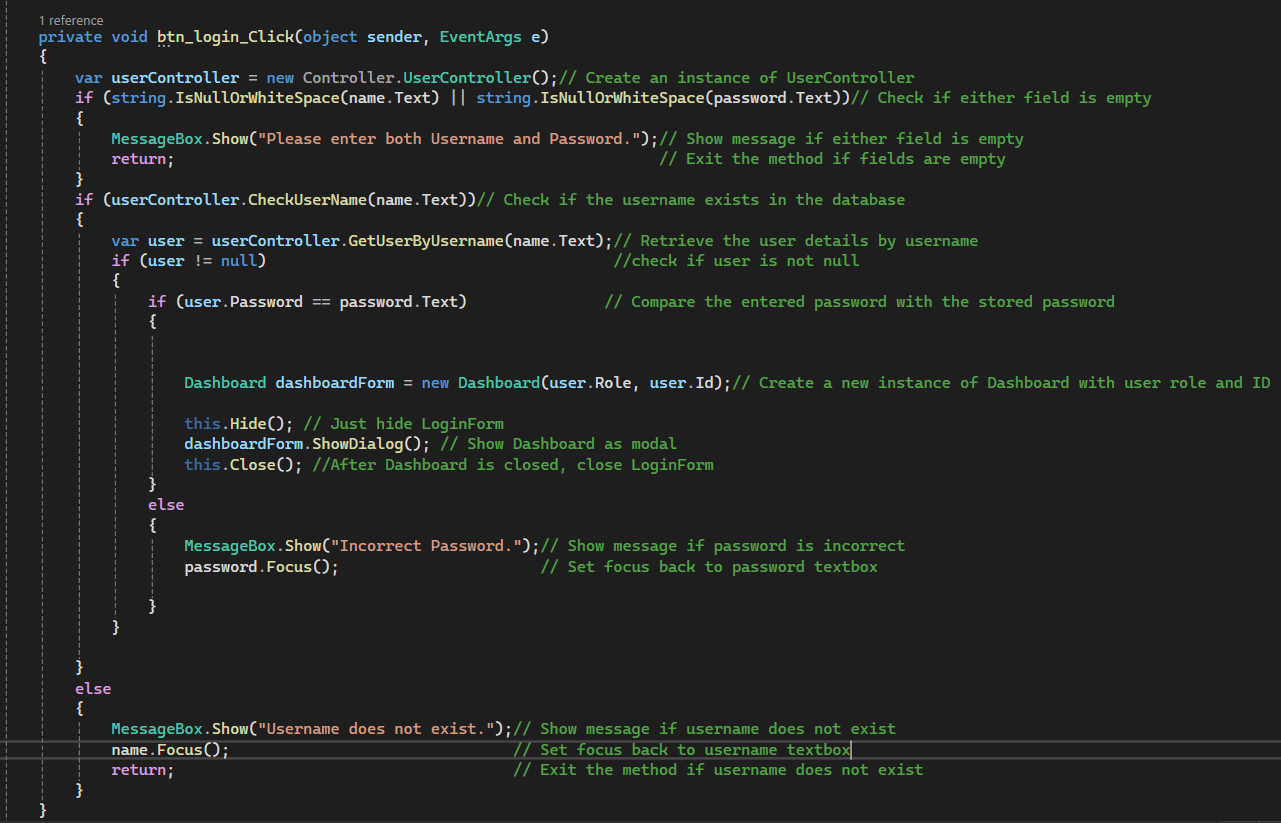
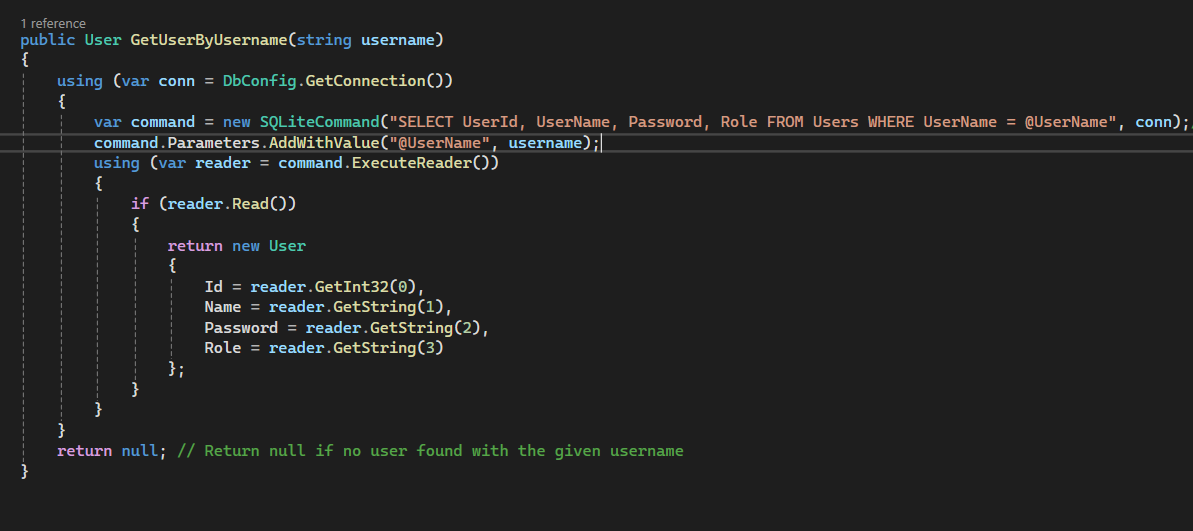


Figure CheckUserName Method in User Controller

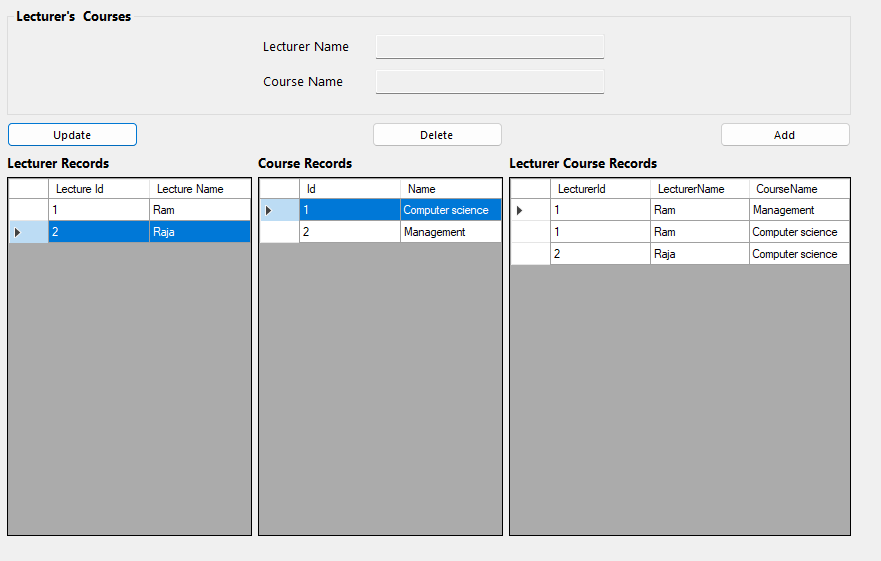


Figure GetuserByUserName method in User Controller



# 2. Assigning Lecturers to Courses

Figure Lecturer's Courses Form UI



## Lecturer's Courses Form Code

Figure Selecting Lecturer from Lecturer Records

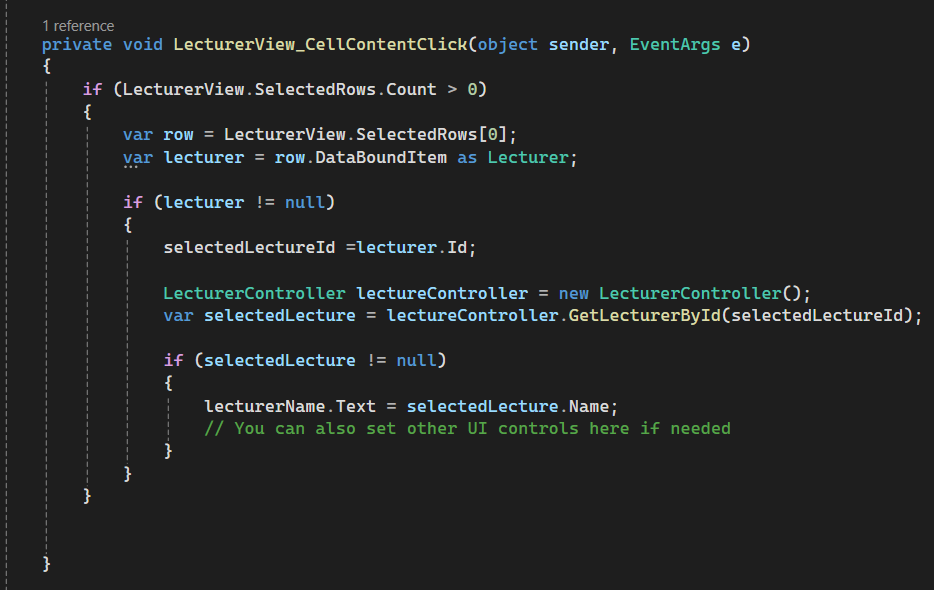


Figure Selecting Course from Course Records

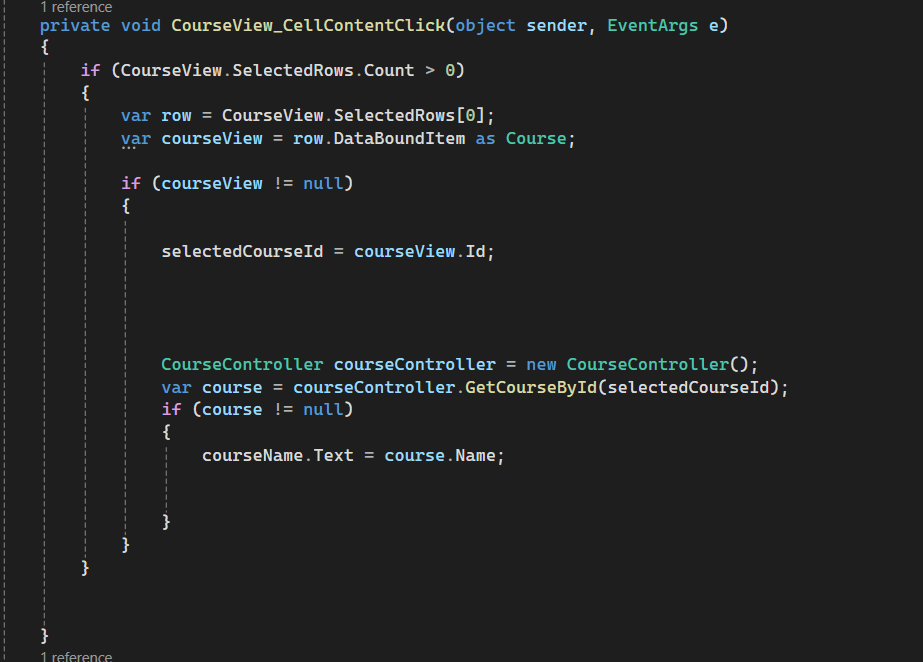
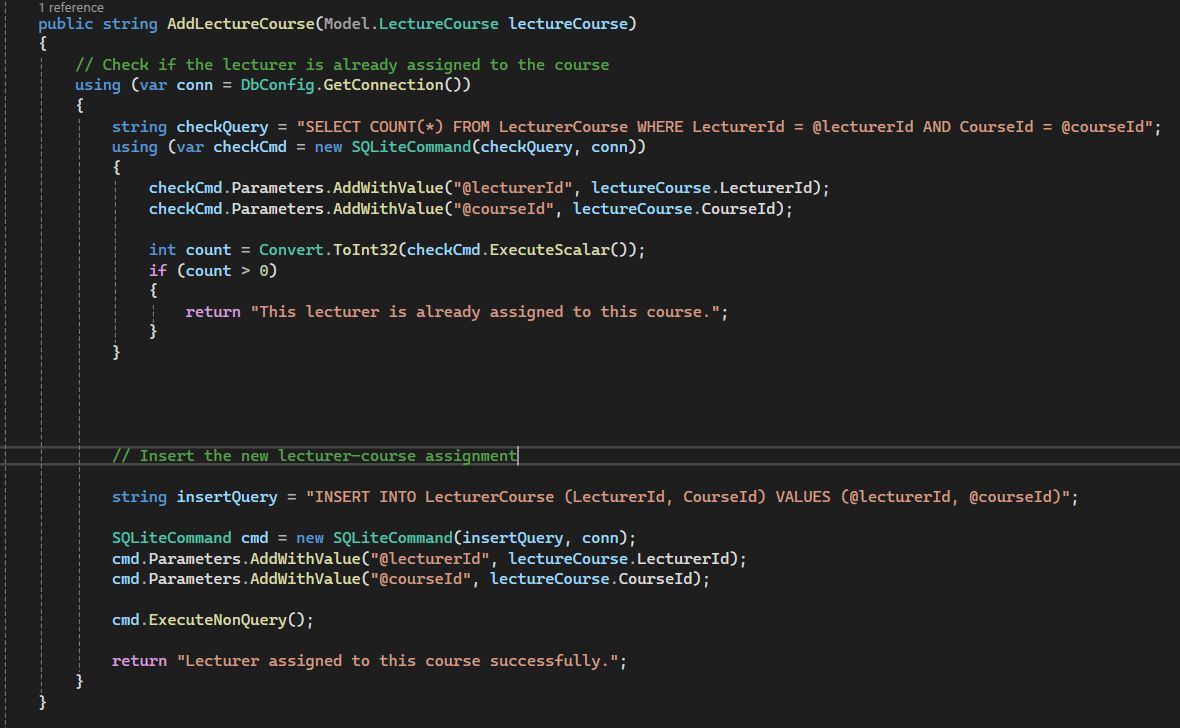
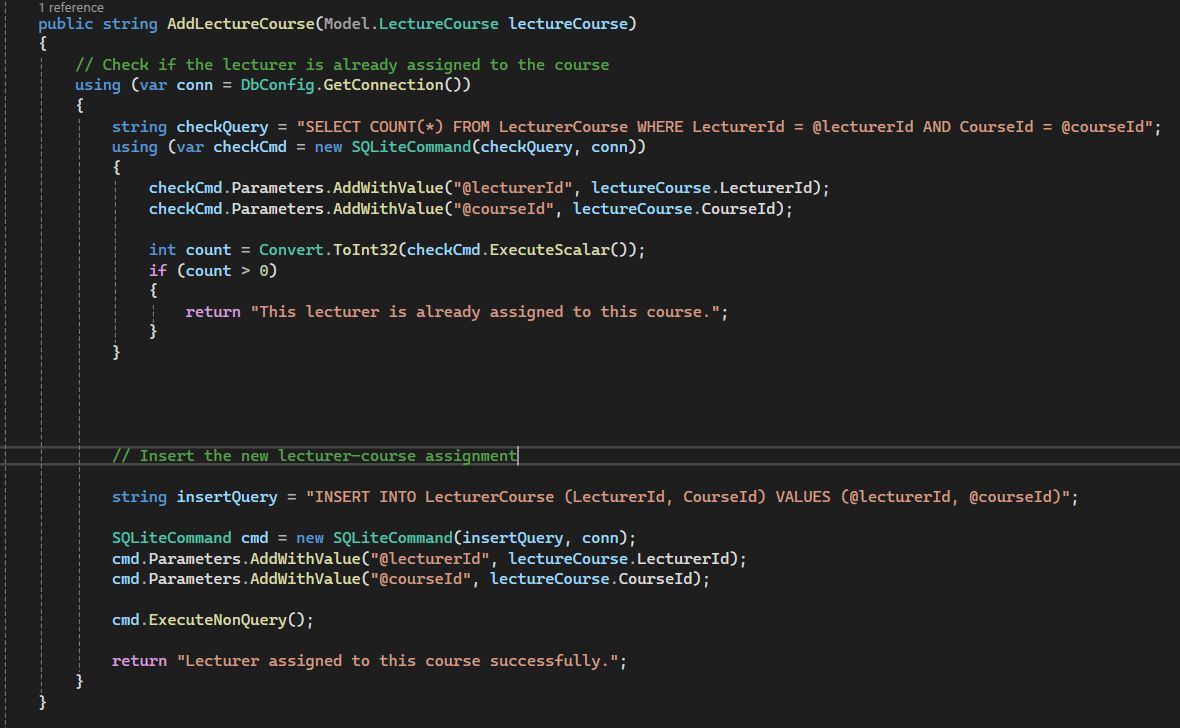


Figure Lecturer's Courses Form Code for Add button



## LecturerCourse Controller Code

Figure Add Lecturer’s Course



# 3. Update Lecturer’s Course

## Lecturer's Courses Form Code

Figure Selecting Lecturer-Course Pair from Lecturer Course Records

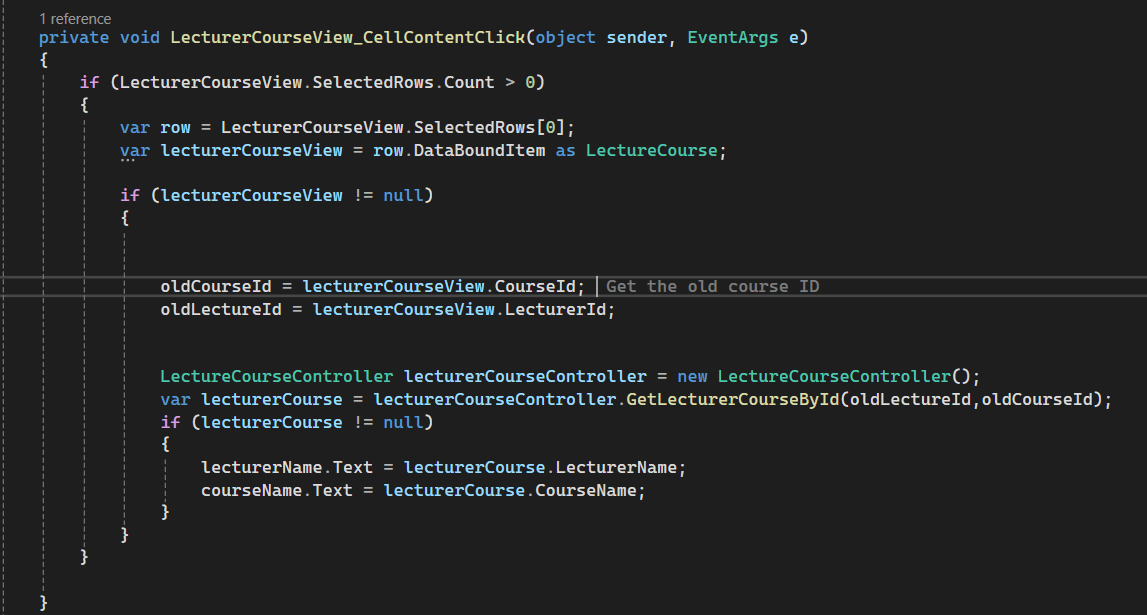


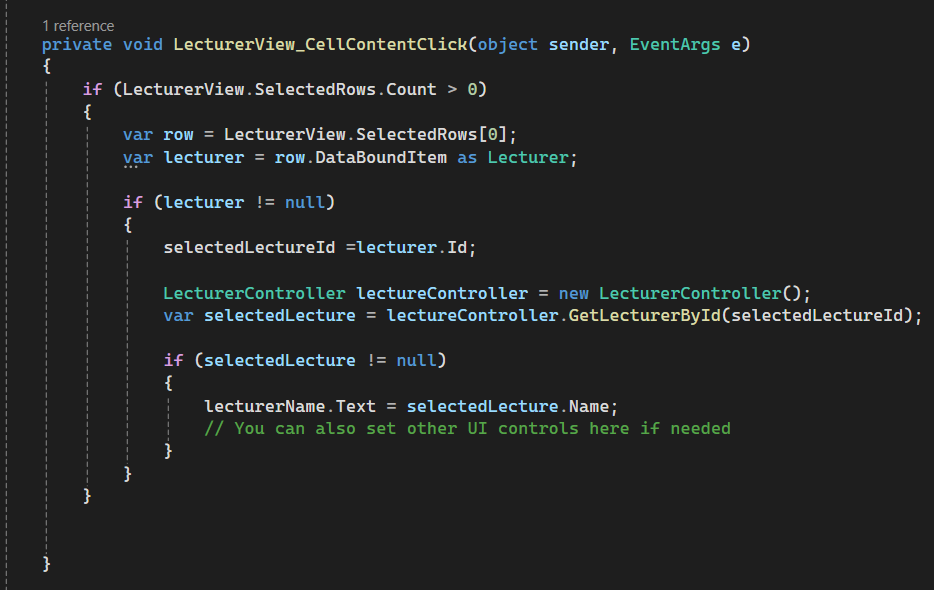
Figure Select Lecturer from Records (If Changes Are Needed)

Figure Select Course from Records (If Changes Are Needed)

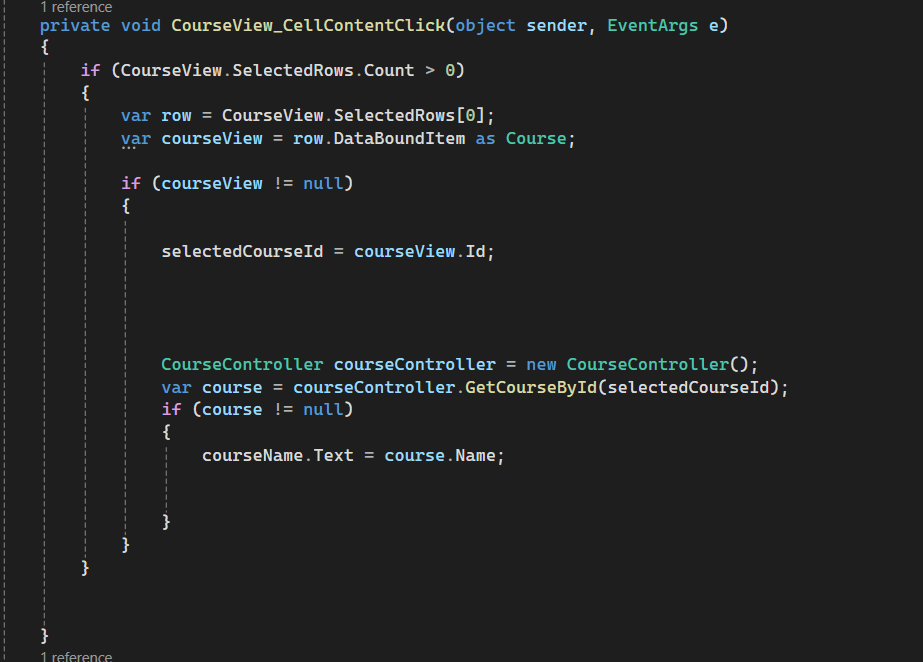
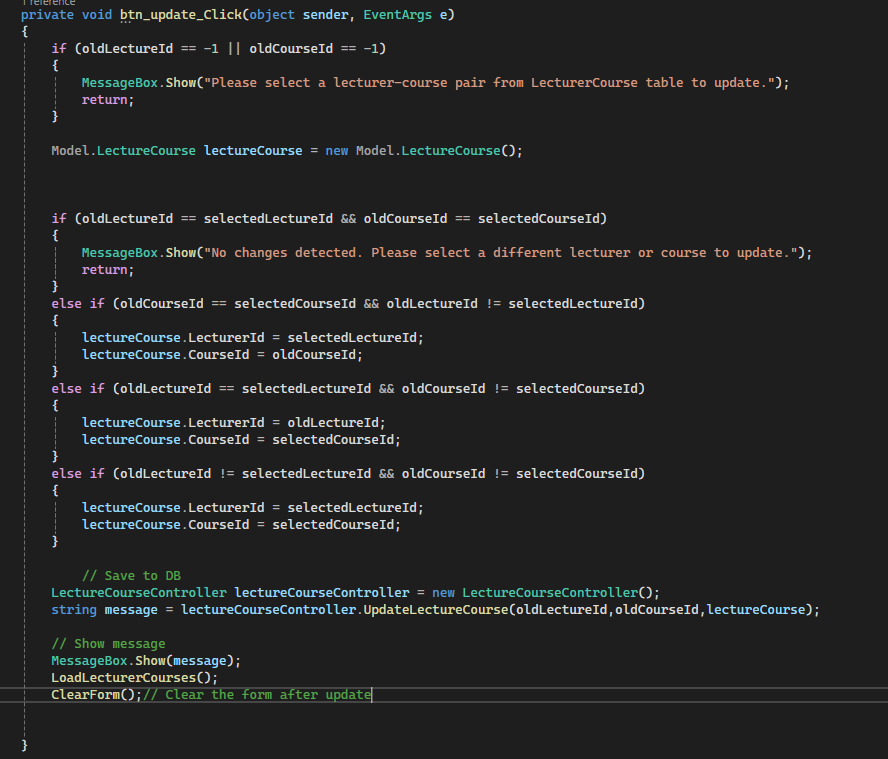


Figure Code for Update Button



## LecturerCourse Controller Code

Figure Update Lecturer's Course

