Lab Activities

Lab Activity 1: Creating a Welcome Console App

- **Objective**: Learn basic C# syntax and Visual Studio setup.
- **Description**: Write a console program to display a welcome message for the Unicom TIC Management System and read user input.
- **Steps**:
    1. Open Visual Studio, create a new **C# Console App** project named “UMSConsoleTest”.
    2. In Program.cs, use Console.WriteLine to print “Welcome to Unicom TIC Management System!”.
    3. Use Console.ReadLine to read a user’s name and print “Hello, [name]!”.
    4. Debug the program using a breakpoint to check the name variable.
- **Expected Outcome**: Console outputs “Welcome to Unicom TIC Management System!” followed by “Hello, Karthik!” (for input “Karthik”).

Lab Activity 2: Validating Login Input with Conditionals

- **Objective**: Use C# conditionals and methods to validate user input.
- **Description**: Create a console program to check if a username and password meet basic requirements for Unicom TIC Management System login.
- **Steps**:
    1. Create a new Console App project named “UMSLoginCheck”.
    2. Define a method IsValidLogin that takes username and password as parameters.
    3. Use if-else to check if username is not empty and password is at least 6 characters long, returning true or false.
    4. In Main, read user input for username and password, call IsValidLogin, and print “Valid login” or “Invalid login”.
- **Expected Outcome**: Input “admin1” and “pass123” outputs “Valid login”; input “” and “pass” outputs “Invalid login”.

Lab Activity 3: Building a Basic Login Form

- **Objective**: Create a WinForms form with basic controls and event handling.
- **Description**: Design a login form for the Unicom TIC Management System with username and password fields.
- **Steps**:
    1. Create a new **C# Windows Forms App** project named "UMSLoginForm".
    2. In Form Designer, add two Labels ("Username," "Password"), two TextBoxes, and a Button ("Login").
    3. Set the password TextBox's PasswordChar to '*' for hidden input.
    4. Add a button click event to check if both TextBoxes are non-empty, showing a MessageBox: "Login attempt!".
- **Expected Outcome**: Clicking "Login" with text in both fields shows "Login attempt!"; empty fields show no message.

Lab Activity 4: Displaying Courses in a ComboBox

- **Objective**: Use ComboBox to display a list of items.
- **Description**: Create a WinForms form to show a list of courses in a ComboBox for the Unicom TIC Management System Course module.
- **Steps**:
    1. Create a new WinForms project named "UMSCourseCombo".
    2. Add a ComboBox and a Button ("Show Selection") to a form.
    3. In the form's Load event, add three course names to the ComboBox (e.g., "BSc Computer Science," "Diploma in IT," "BSc Math").
    4. On button click, show a MessageBox with the selected course (e.g., "Selected: BSc Computer Science").
- **Expected Outcome**: ComboBox lists courses; button click shows the selected course in a MessageBox.

Lab Activity 5: Creating a Room Class

- **Objective**: Apply OOP encapsulation to create a data class.
- **Description**: Define a Room class for Unicom TIC Management System Timetable module with properties for computer labs and lecture halls.
- **Steps**:
  1. Create a Console App project named “UMSRoomClass”.
  2. Create a Room class with private fields roomID, roomName, roomType and public properties RoomID, RoomName, RoomType.
  3. Add a constructor to set all properties.
  4. In Main, create two Room objects (e.g., “Lab 1, Lab” and “Hall A, Hall”) and print their details.
- **Expected Outcome**: Console outputs “Room: Lab 1, Type: Lab” and “Room: Hall A, Type: Hall”.

Lab Activity 6: Setting Up a SQLite Users Table

- **Objective**: Create and populate a SQLite table.
- **Description**: Set up a Users table for Unicom TIC Management System login and insert sample users.
- **Steps**:
  1. Create a Console App project named “UMSUsersTable”.
  2. Install System.Data.SQLite via NuGet.
  3. Create a unicomtic.db file and a Users table with columns UserID (INTEGER PRIMARY KEY), Username (TEXT), Password (TEXT), Role (TEXT).
  4. Insert two users (e.g., “admin1, pass123, Admin” and “john123, student1, Student”).
  5. Select and print all users.
- **Expected Outcome**: Console shows two users’ details from Users table.

Lab Activity 7: Adding Courses to SQLite

- **Objective**: Implement SQLite INSERT operation.
- **Description**: Create a WinForms form to add courses to a Courses table for the Unicom TIC Management System.
- **Steps**:
    1. Create a WinForms project named “UMSCourseAdd”.
    2. Create a Courses table in unicomtic.db with CourseID (INTEGER PRIMARY KEY) and CourseName (TEXT).
    3. Design a form with a TextBox (course name), Button (“Add”), and MessageBox feedback.
    4. On button click, insert the TextBox value into Courses (e.g., “BSc Computer Science”) and show “Course added!”.
- **Expected Outcome**: Entering “Diploma in IT” and clicking “Add” saves it to Courses and shows a success message.

Lab Activity 8: Displaying Students in DataGridView

- **Objective**: Read SQLite data and display in a DataGridView.
- **Description**: Show a list of students from the Students table in a WinForms form.
- **Steps**:
    1. Create a WinForms project named “UMSStudentView”.
    2. Create a Students table with StudentID (INTEGER PRIMARY KEY), Name (TEXT), CourseID (INTEGER).
    3. Insert two sample students (e.g., “John Doe, 1,” “Jane Smith, 2”).
    4. Design a form with a DataGridView and set columns: StudentID, Name.
    5. On form load, select all students and display in the DataGridView.
- **Expected Outcome**: DataGridView shows “John Doe” and “Jane Smith” with their IDs.

Lab Activity 9: Async Loading of Rooms

- **Objective**: Use async/await for database operations.
- **Description**: Load rooms into a ComboBox asynchronously for the Unicom TIC Management System Timetable module.
- **Steps**:
    1. Create a WinForms project named “UMSRoomAsync”.
    2. Create a Rooms table with RoomID (INTEGER PRIMARY KEY), RoomName (TEXT), RoomType (TEXT), and insert “Lab 1, Lab,” “Hall A, Hall”.
    3. Design a form with a ComboBox and a Button (“Load Rooms”).
    4. Write an async method to select all rooms and populate the ComboBox.
    5. On button click, call the async method and show “Rooms loaded!”.
- **Expected Outcome**: Clicking “Load Rooms” populates ComboBox with “Lab 1,” “Hall A” and shows a message.

Lab Activity 10: Updating Exam Details

- **Objective**: Implement SQLite UPDATE operation.
- **Description**: Create a form to update an exam’s name in the Exams table.
- **Steps**:
    1. Create a WinForms project named “UMSExamUpdate”.
    2. Create an Exams table with ExamID (INTEGER PRIMARY KEY), ExamName (TEXT), SubjectID (INTEGER).
    3. Insert a sample exam (e.g., “Midterm, 1”).
    4. Design a form with a TextBox (new name), ComboBox (select ExamID), and Button (“Update”).
    5. On button click, update the exam’s name and show “Exam updated!”.
- **Expected Outcome**: Changing “Midterm” to “Final” for ExamID 1 updates the table and shows a message.

Lab Activity 11: Creating a Course Model

- **Objective**: Define a Model class for MVC.
- **Description**: Create a Course Model class and test it in a console app.
- **Steps**:
    1. Create a Console App project named “UMSCourseModel”.
    2. Create a Course class with properties CourseID and CourseName.
    3. Add a method GetDetails to return a string (e.g., “Course: BSc Computer Science”).
    4. In Main, create a Course object and call GetDetails, printing the result.
- **Expected Outcome**: Console outputs “Course: Diploma in IT” for a course object.

Lab Activity 12: Building a Course View

- **Objective**: Create a WinForms View for MVC.
- **Description**: Design a CourseForm to display courses for the Unicom TIC Management System Course module.
- **Steps**:
    1. Create a WinForms project named “UMSCourseView”.
    2. Design a form with a DataGridView (columns: CourseID, CourseName) and a Button (“Refresh”).
    3. Hardcode a list of two courses (e.g., “BSc Computer Science,” “Diploma in IT”).
    4. On button click, populate the DataGridView with the course list.
- **Expected Outcome**: Clicking “Refresh” shows two courses in the DataGridView.

Lab Activity 13: Creating a Course Controller

- **Objective**: Implement a Controller for MVC.
- **Description**: Create a CourseController to add a course in a WinForms form.
- **Steps**:
  1. Create a WinForms project named “UMSCourseController”.
  2. Create a Course class with CourseID and CourseName.
  3. Create a CourseController class with a method AddCourse(Course course) that stores the course in a list.
  4. Design a form with a TextBox (course name), Button (“Add”), and DataGridView.
  5. On button click, call AddCourse and refresh the DataGridView.
- **Expected Outcome**: Adding “BSc Math” shows it in the DataGridView.

Lab Activity 14: Hardcoded Role-Based Login

- **Objective**: Implement a basic role-based login system.
- **Description**: Create a login form that checks hardcoded credentials and shows a role-specific dashboard.
- **Steps**:
  1. Create a WinForms project named “UMSLoginRole”.
  2. Design a login form with TextBoxes (username, password) and a Button (“Login”).
  3. Hardcode a user: “admin1, pass123, Admin”.
  4. On login, check credentials and open a dashboard form with a Label (“Welcome, Admin!”) and a Button (“Manage Courses”).
- **Expected Outcome**: Entering “admin1, pass123” opens a dashboard with an Admin button; wrong credentials show “Invalid login”.

Lab Activity 15: Database-Driven Login

- **Objective**: Validate login using SQLite.
- **Description**: Create a login form that checks credentials against the Users table.
- **Steps**:
    1. Create a WinForms project named “UMSDatabaseLogin”.
    2. Use the Users table from Lab 6.
    3. Design a login form with TextBoxes (username, password) and a Button (“Login”).
    4. On button click, query Users for matching username and password, showing “Login successful!” or “Invalid login” in a MessageBox.
- **Expected Outcome**: “admin1, pass123” shows “Login successful!”; incorrect input shows “Invalid login”.

Lab Activity 16: Adding a Timetable Entry

- **Objective**: Create a form to add timetable entries with rooms.
- **Description**: Build a TimetableForm to add a timetable entry to SQLite.
- **Steps**:
    1. Create a WinForms project named “UMSTimetableAdd”.
    2. Create Subjects (SubjectID, SubjectName) and Timetables (TimetableID, SubjectID, TimeSlot, RoomID) tables, linking to Rooms.
    3. Insert sample data: subjects (“Programming”), rooms (“Lab 1, Lab”).
    4. Design a form with ComboBoxes (subjects, rooms), TextBox (time slot), and Button (“Add”).
    5. On button click, save to Timetables and show “Timetable added!”.
- **Expected Outcome**: Adding “Programming, Monday 10 AM, Lab 1” saves to the table and shows a message.

Lab Activity 17: Viewing Timetable with Rooms

- **Objective**: Display timetable entries in a DataGridView.
- **Description**: Show timetable entries with computer labs or lecture halls in a form.
- **Steps**:
  1. Create a WinForms project named “UMSTimetableView”.
  2. Use tables from Lab 16, with sample timetable data.
  3. Design a form with a DataGridView (columns: Subject, Time Slot, Room).
  4. On form load, select all timetable entries and display in the DataGridView, joining with Subjects and Rooms.
- **Expected Outcome**: DataGridView shows “Programming, Monday 10 AM, Lab 1”.

Lab Activity 18: Adding Student Marks

- **Objective**: Implement a form to add marks for the Unicom TIC Management System.
- **Description**: Create a form to save student marks to the Marks table.
- **Steps**:
  1. Create a WinForms project named “UMSMarksAdd”.
  2. Create Marks (MarkID, StudentID, ExamID, Score) and Exams tables, with sample data.
  3. Design a form with ComboBoxes (student, exam), TextBox (score), and Button (“Add”).
  4. On button click, save to Marks (e.g., “John Doe, Midterm, 85”) and show “Mark added!”.
- **Expected Outcome**: Adding a mark saves to the table and shows a message.

Lab Activity 19: Role-Based Dashboard

- **Objective**: Show role-specific buttons on a dashboard.
- **Description**: Create a dashboard form that displays buttons based on user role.
- **Steps**:
    1. Create a WinForms project named “UMSDashboardRole”.
    2. Use the Users table from Lab 6.
    3. Design a login form (from Lab 15) and a dashboard with buttons: “Manage Courses” (Admin), “View Timetable” (all roles).
    4. After login, check the user’s role and show only allowed buttons (e.g., Admin sees “Manage Courses”).
- **Expected Outcome**: Admin sees both buttons; Student sees only “View Timetable”.

Lab Activity 20: Testing the Full UMS

- **Objective**: Test all Unicom TIC Management System modules together.
- **Description**: Run the Unicom TIC Management System, testing login, course, student, exam, and timetable functionality.
- **Steps**:
    1. Use a WinForms project combining prior labs or create a new one named “UMSFullTest”.
    2. Ensure all tables (Users, Courses, Students, Exams, Marks, Rooms, Timetables) are set up.
    3. Log in as Admin, add a course, student, exam, and timetable entry with “Lab 1”.
    4. Log in as Student, view the timetable and marks.
    5. Test an error (e.g., empty room selection) and verify a MessageBox appears.
- **Expected Outcome**: All modules work; Admin adds data, Student views it, errors are handled.