BANGLADESH UNIVERSITY OF BUSINESS & TECHNOLOGY

PROJECT REPORT

Course Code : CSE 200

Course Title : Software Development II

Project Name : Tutoring Center Management System

Submitted by

Name	ID
Md. Rahat Bin Israil	22234103**139**
Md. Shahidul Islam Parvez	22234103**123**
Md. Noor Uddin Yousuf Shanto	22234103**136**
Akkas Uddin Akash	22234103**137**
Fahim Sahriar	22234103**151**

Submitted To

Name : Anusha Aziz
Course Instructor
Lecturer, Department of : CSE
Bangladesh University of Business & Technology

TABLE OF CONTENTS

Declaration

We hereby declare that the project entitled "Tutoring Management System" submitted for Software Development II is our own work and has not been submitted for any other de gree or diploma. We further declare that all information sources used in this project have been duly acknowledged.

We also declare that we have acknowledged all the sources of information used in this report.

Dedication

We would like to dedicate this project report to our families, friends, and mentors who have supported and encouraged us throughout the development of the Tutoring Management System. Their unwavering support and guidance have been instrumental in the successful completion of this project.

Acknowledgement

We extend our sincere gratitude to our project supervisor Anusha Aziz, whose guidance and expertise have been invaluable in shaping this project. Their constructive feedback and insightful suggestions have significantly enhanced the quality of our work.

We would also like to thank our professors and mentors for their valuable input and encouragement throughout the duration of this project. Their wisdom and encouragement have been a constant source of motivation for us.

Additionally, we express our gratitude to Bangladesh University of Business and Technology for providing us with the resources and facilities necessary to undertake this project.

We are also grateful to all those who participated in the testing and evaluation of the system, providing us with valuable feedback that helped in refining the project.

Finally, we would like to acknowledge the contributions of our team members, Md. Rahat Bin Israil, Md. Shahidul Islam Parvez, Md. Noor Uddin Yousuf Shanto, Akkas Uddin Akash, Fahim Sahriar, for their collaborative efforts, ideas, and dedication in bringing the Tutoring Management System to fruition.

Lastly, we would like to acknowledge the support of our peers and friends, whose encouragement and camaraderie have made this journey enjoyable and memorable.

Abstract

The Tutoring Management System is a comprehensive software solution designed to streamline the operations and management of tutoring services. This project aims to develop a user-friendly platform that enables tutors, students, and administrators to efficiently manage tutoring sessions, track student progress, and facilitate seamless communication.

The system will provide a centralized database to store and manage student and tutor information, including profiles, schedules, and session details to enhance the overall tutoring experience.

The key objectives of this project are to improve the efficiency of tutoring services, enhance the communication and collaboration between tutors and students, and provide a robust data management system to support the growth and development of the tutoring program.

Through the implementation of the Tutoring Management System, we aim to streamline the tutoring process, improve student outcomes, and contribute to the overall success of the educational institution.

Chapter 1 INTRODUCTION

1.1 Introduction :

This report details the design and implementation of a database for a Tutoring Center Management System (TCMS). The TCMS aims to streamline and automate various administrative and operational tasks within a tutoring center, enhancing efficiency and providing a better experience for students.

1.2 Project Objective :

Our goal is to develop a comprehensive Tutoring Center Management System that simplifies administrative tasks, improves communication, and enhances the overall efficiency of tutoring centers.

Outline the specific objectives such as:

- Streamlining course offering.
- Managing student information.
- Manage course & see details.
- Generating student reports for analysis and decision-making.

1.3 Motivation :

Traditional methods of managing tutoring centers often involve manual processes, which are not only time-consuming but also errors and inconsistencies The motivation behind TCMS is to enhance the efficiency and effectiveness of educational delivery. By leveraging technology, the system aims to streamline administrative tasks, improve student engagement, and learning environments. improve the overall effectiveness of tutoring services.

1.4 Features :

Overview

The Tutoring Center Management System is a desktop application designed to streamline the management of a tutoring center. It provides a user-friendly interface for administrators to handle student records, course details, teacher assignments, student scores, and generate comprehensive reports. The system aims to enhance operational efficiency and provide valuable insights into the center's performance.

<u>Student Management :</u>

- **Registration:**
  - Add new students with personal details (name, address, contact information, gender, etc.).
  - Store student records securely.
- **Manage Student:**
  - Update student information.

- Search and filter students based on criteria like Address, Name.

- **Print:**
  - Print student information reports.

Course and Teacher Management :

- **Registered Courses:**
  - Add new courses with details (name, description, duration, teacher, etc.).
  - Assign tutors to specific courses.
  - Associate students with specific courses.
- **Manage Courses:**
  - Update course details.
  - Search and filter for courses by course name or student name.
  - View and manage course.
- **Print:**
  - Generate course reports with details, including enrolled students and assigned tutors and courses.

Score Management :

- **New Score:**
  - Record student scores for specific courses.
  - Associate scores with student ID and course details.
  - Add score description (e.g., A+, Very Good).
- **Manage Score:**
  - Update score details.
  - Search and filter scores by Name or course name.

- **Print:**
  - Generate comprehensive student result reports, scores and overall performance.

Dashboard :

- **Student Statistics:**
  - Display the total number of students enrolled.
  - Show student gender distribution (male/female).
- **Course Enrollment Trends:**
  - Track the number of students enrolled in each course.

Exit :

- Allow users to exit the application securely.

Chapter 2 LITERATURE REVIEW

2.1 Introduction :

The literature review section of a project report provides an overview of existing research, theories, and practices relevant to the project's subject matter. In the case of the Tutoring Management System, the literature review delves into existing systems, methodologies, and studies related to educational management systems, tutoring platforms, and similar domains.

2.2 Related work :

Numerous studies and systems exist in the domain of educational management and tutoring platforms. One notable area of related work is the development of Learning Management Systems (LMS) and Student Information Systems (SIS). LMS platforms such as

Moodle, Canvas, and Blackboard offer comprehensive solutions for course management, content delivery, and student interaction. These systems provide valuable insights into the design and functionality required for an effective tutoring management system.

Additionally, there are specialized tutoring platforms such as Khan Academy, Chegg Tutors, and Tutor.com, which focus specifically on connecting students with tutors for personalized learning experiences. These platforms offer features like scheduling, video conferencing, and progress tracking, which are essential components for our Tutoring Management System.

2.3 Problem Analysis :

The Tutoring Management System addresses several key challenges faced in traditional tutoring environments and existing online platforms:

- **Lack of Personalization:** Many existing tutoring platforms offer generic tutoring services without considering the individual learning needs and preferences of students. A lack of personalization can hinder the effectiveness of tutoring sessions and limit student engagement and motivation.
- **Difficulty in Monitoring Progress:** Without adequate tools for monitoring student progress and performance, tutors may struggle to assess the effectiveness of their teaching strategies and identify areas where students require additional support or intervention.
- **Scheduling Challenges:** Coordinating tutoring sessions between tutors and students can be cumbersome, especially when dealing with multiple students and varying schedules.

Manual scheduling processes are prone to errors and may result in conflicts or missed appointments.

- **Data Security and Privacy Concerns:** As the Tutoring Management System involves the collection and storage of sensitive student information, ensuring data security and privacy is paramount. Unauthorized access to student data or breaches in system security could have serious consequences for both students and tutors.

Chapter 3 REQUIREMENT SPECIFICATION

3.1 Hardware Requirement

3.2 Software Requirement

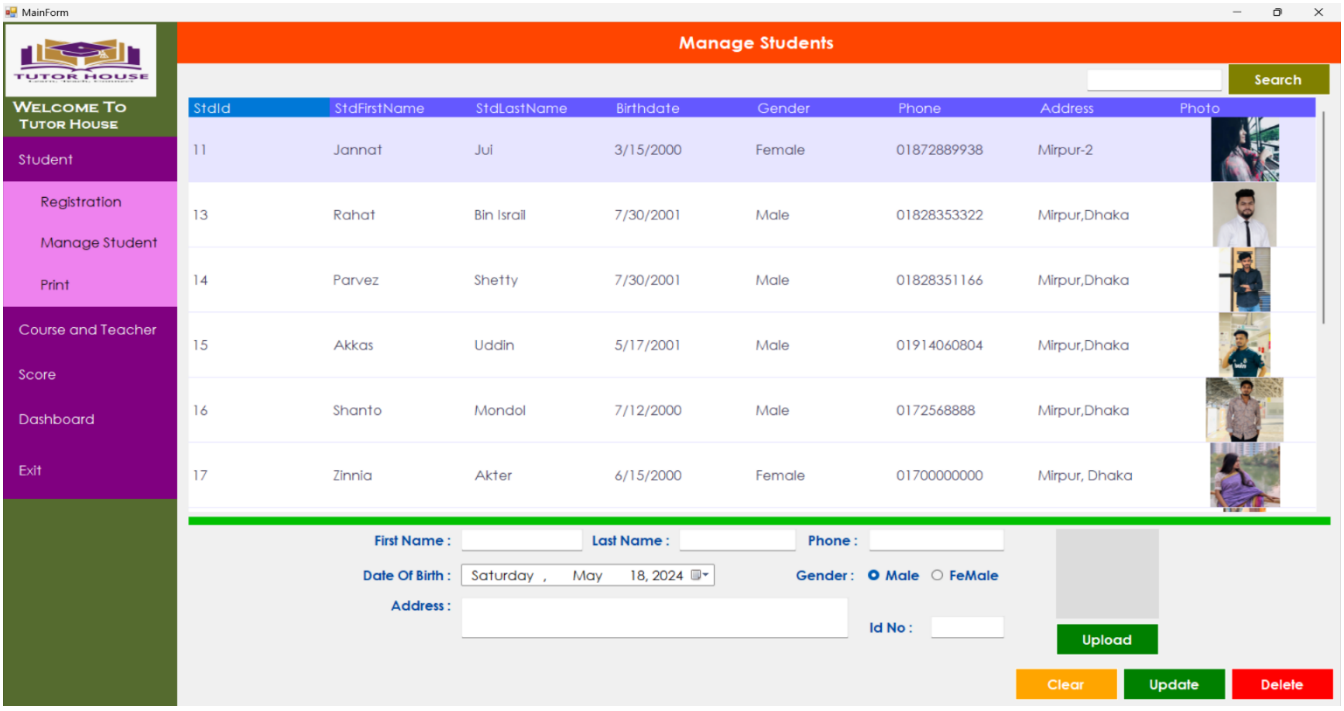
CPU	Core i3 4rth Generation or equivalent or better
Disk Space	200 MB or more
Ram	1 Gb or better
Monitor	15 VGA Colour or better
Operating system	Windows 7/8/9/10/11
Technology Used	ASP.NET
Backend Used	XAMPP SQL SERVER

Chapter 4 : SYSTEM MANUAL

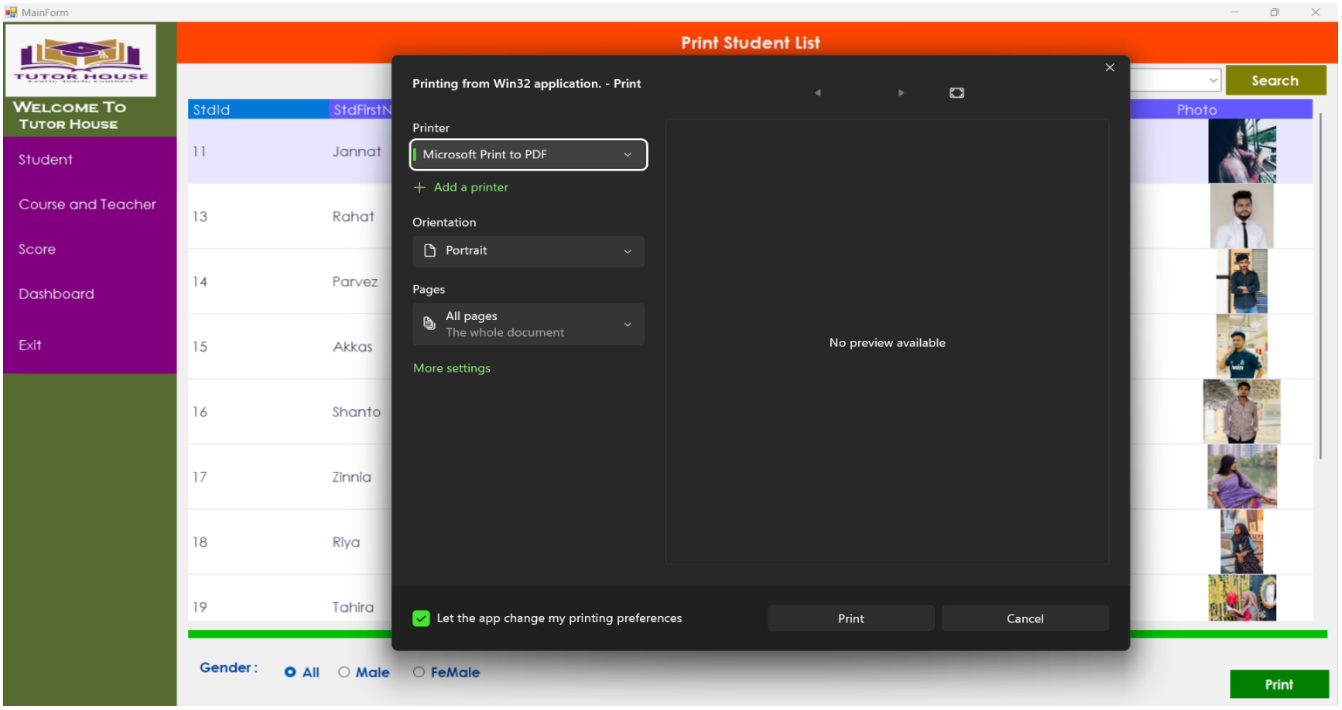
4.1 Student Management Part

Registration

Manage Students

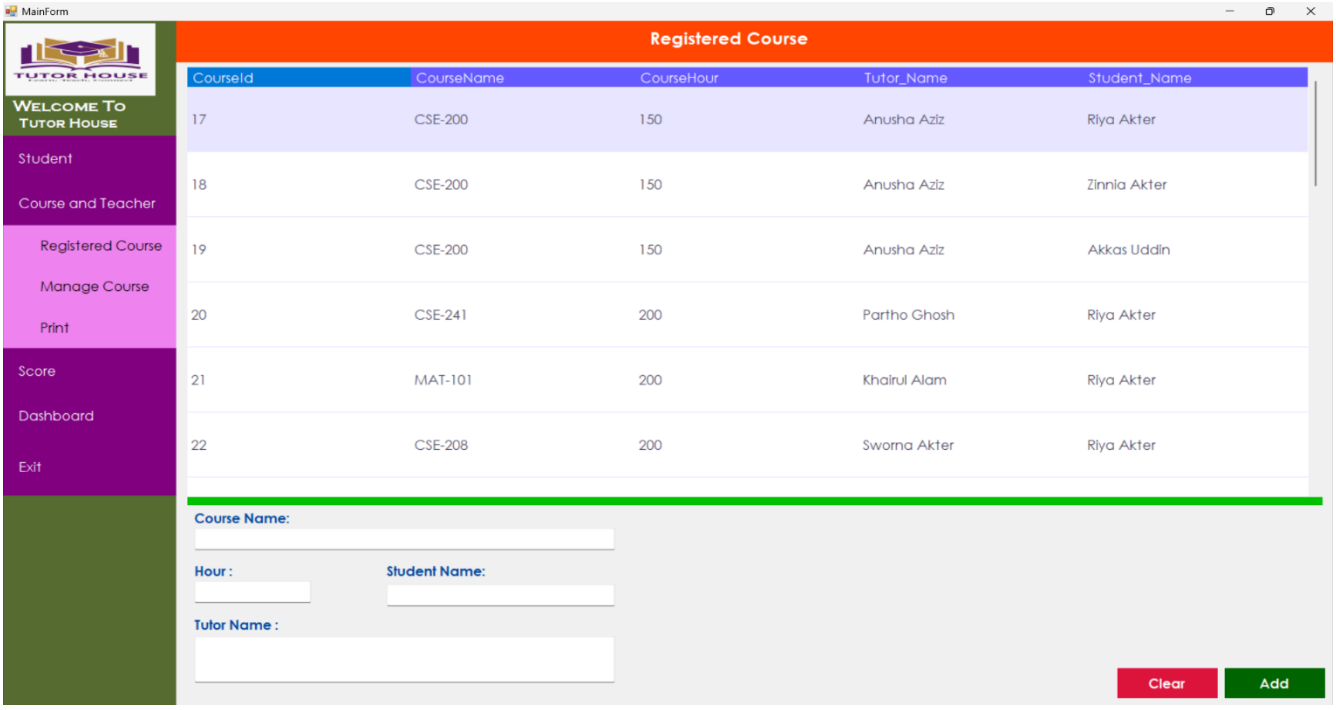


Print Student List

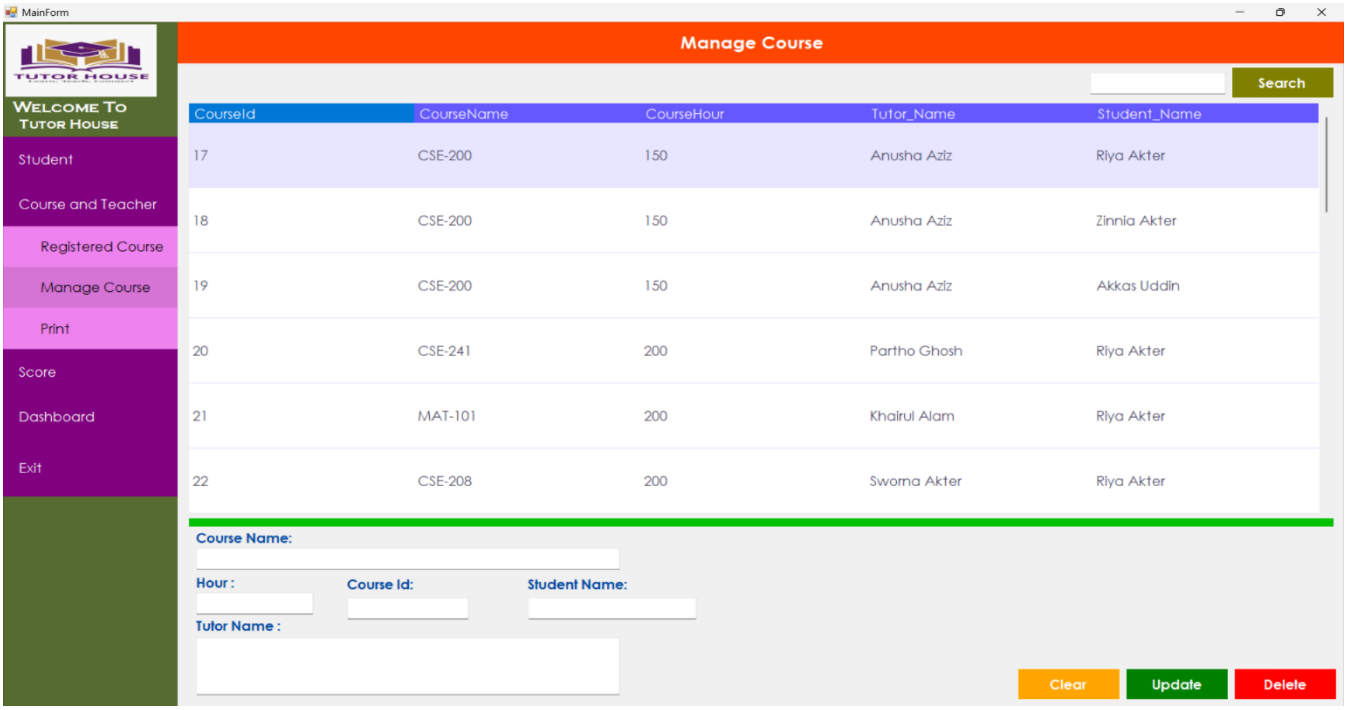


4.2 Course and Teacher Management Part

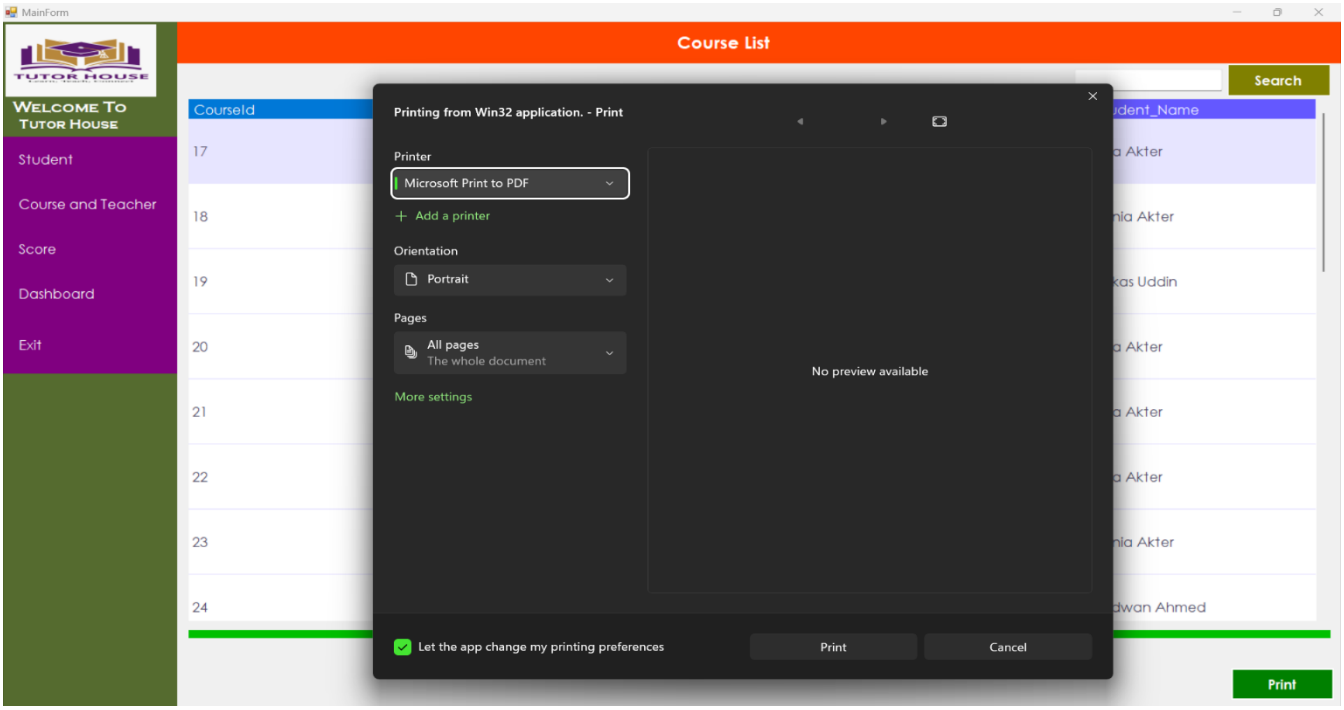
Registered Course



Manage Course

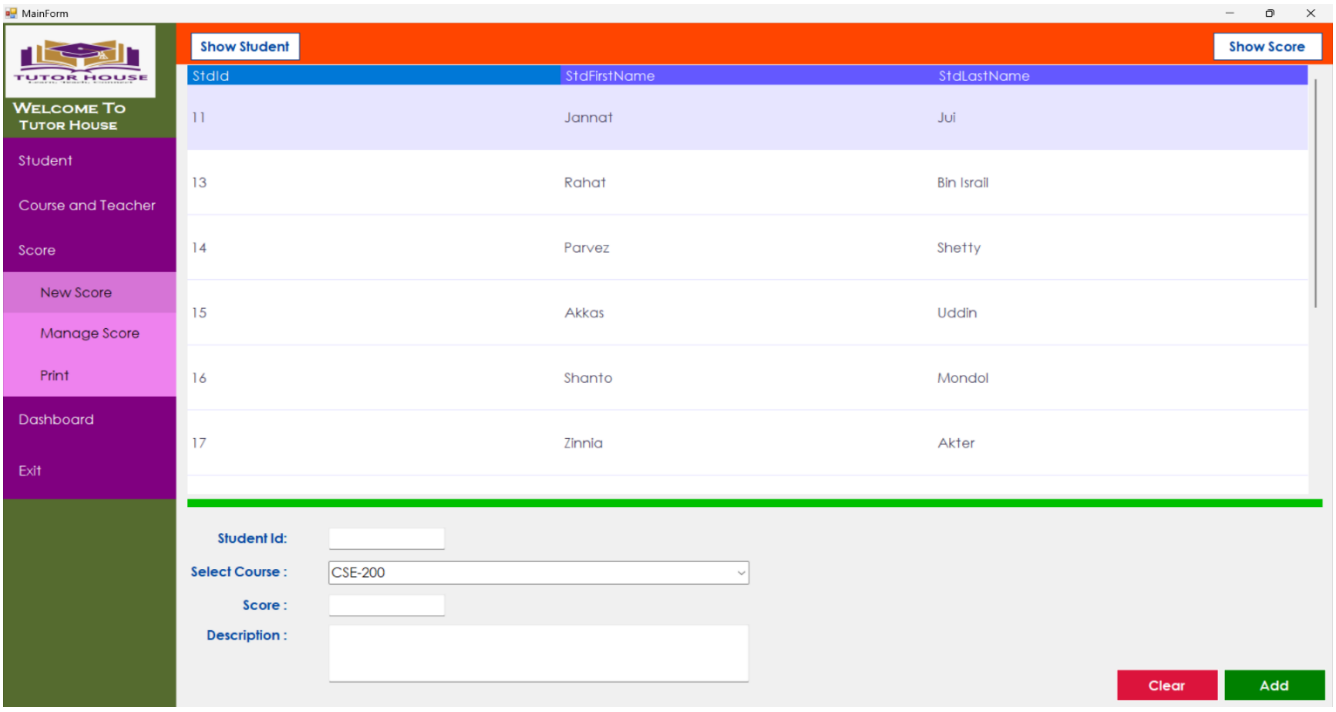


Print Course List

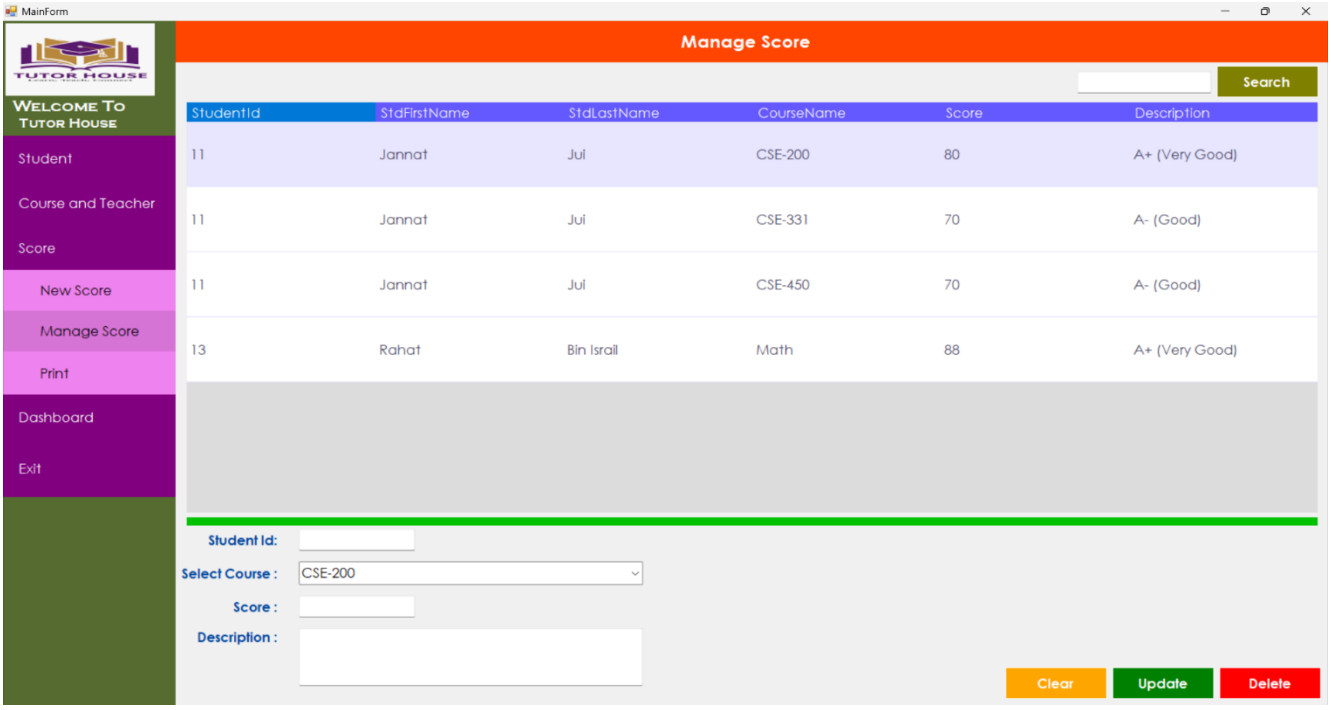


4.3 Score Management Part

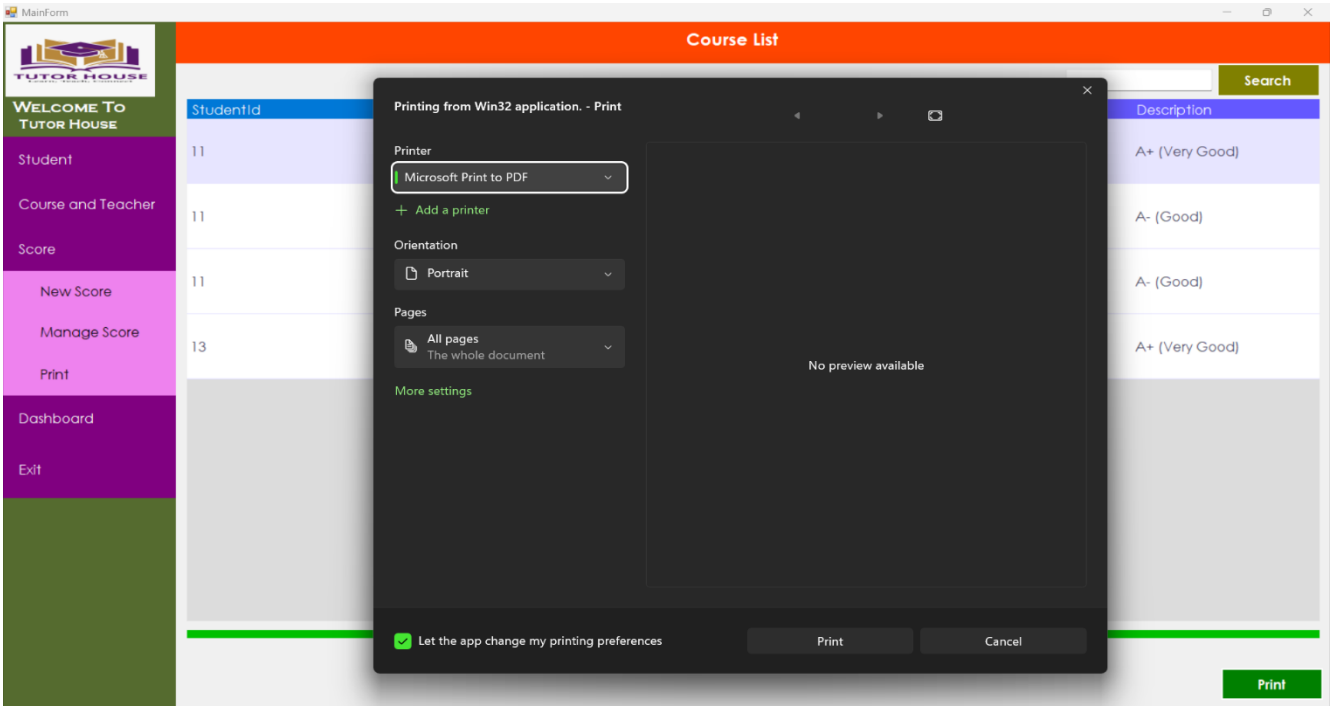
New Score



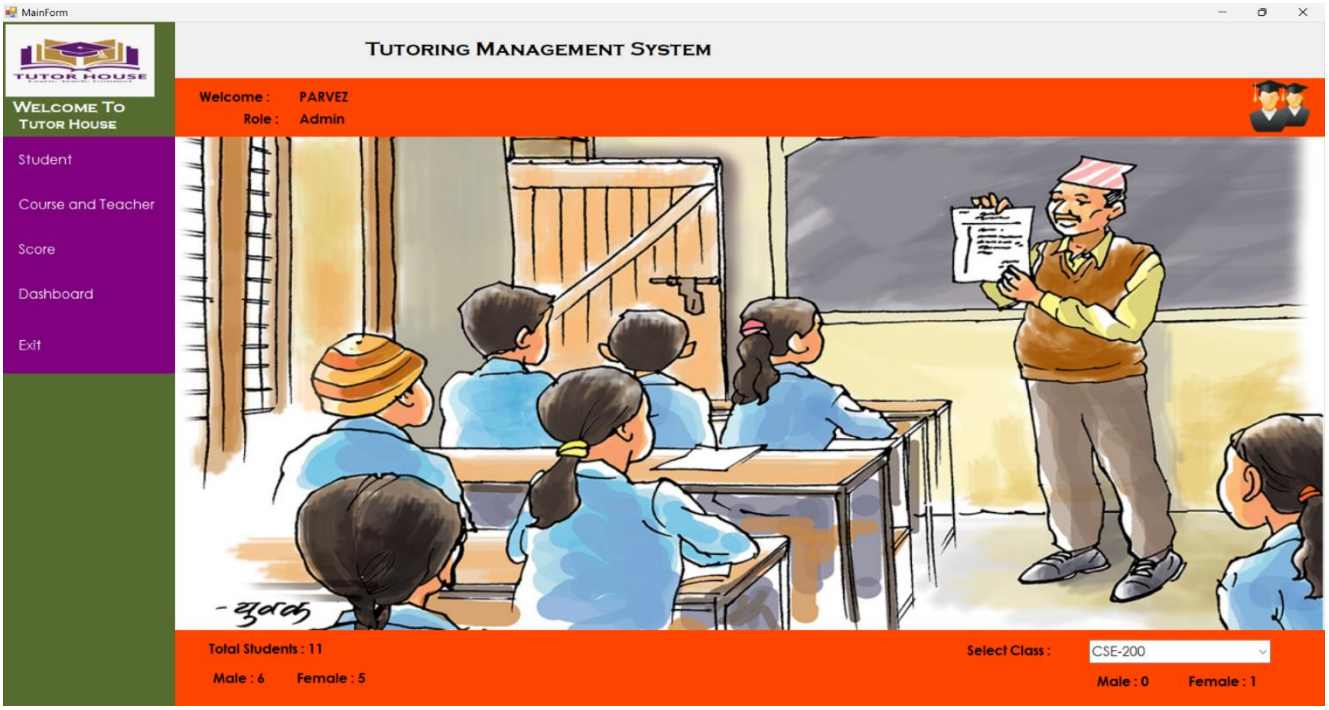
Manage Score



Print Score List



4.4 Dashboard Part



Chapter 5 : CONCLUSION & FUTURE WORK

5.1 Conclusion

The Tutoring Center Management System represents a significant advancement in the field of educational management software. By automating routine tasks, optimizing resource allocation, and providing valuable insights, TCMS empowers tutoring centers to operate more efficiently and effectively. With its array of features and user-friendly interface, TCMS is poised to revolutionize the way tutoring centers manage their operations, ultimately enhancing the educational experience for students.

5.2 Future Work/Plan

To enhance the professionalism and effectiveness of the Tutoring Management System project in the future, consider implementing the following ideas:

1. **Enhanced User Interface**: Update the user interface to be more intuitive, visually appealing, and user-friendly. Incorporate modern design principles to improve user experience and engagement.
2. **Mobile Application**: Develop a mobile application version of the system to allow users to access tutoring services on-the-go. This will increase accessibility and convenience for both tutors and students.
3. **Integration with Learning Management Systems**: Integrate the Tutoring Management System with popular Learning Management Systems (LMS) to streamline data sharing, scheduling, and progress tracking for a more cohesive educational experience.

4. **AI-Powered Recommendations**: Implement artificial intelligence algorithms to provide personalized tutor recommendations based on student learning styles, preferences, and performance data.
5. **Virtual Tutoring Features**: Introduce virtual tutoring capabilities such as video conferencing, virtual whiteboards, and screen sharing to facilitate remote tutoring sessions and enhance collaboration between tutors and students.
6. **Data Analytics and Reporting**: Incorporate advanced data analytics tools to generate insights on tutoring effectiveness, student progress, and areas for improvement. Develop customizable reports for administrators, tutors, and students.
7. **Gamification Elements**: Introduce gamification elements such as badges, rewards, and progress tracking to motivate student engagement and participation in tutoring sessions.
8. **Feedback and Rating System**: Implement a feedback and rating system to gather input from students on tutor performance, session quality, and overall satisfaction. Use this data to continuously improve the tutoring experience.
9. **Integration with Payment Gateways**: Enable seamless payment processing within the system for tutoring services, allowing for secure transactions and automated invoicing.

10. **Continuous Training and Support**: Provide ongoing training and support for tutors on using the system effectively, incorporating best practices in tutoring, and adapting to new features and updates.

11. **Teacher & Student Singup :** Teacher & Student can be Singup separately & handle there information, course by own.

By incorporating these future plan ideas, the Tutoring Management System project can evolve into a professional, efficient, and innovative platform that enhances the tutoring experience for both tutors and students.

Chapter 6 : PROJECT CODE

6.1 Student Class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MySql.Data.MySqlClient;
using System.Data;
using System.Data.SqlClient;

namespace Transparent_Form
{
   class StudentClass
   {
      DBconnect connect = new DBconnect();
      //create a function to add a new students to the database

      public bool insertStudent(string fname, string lname, DateTime
bdate, string gender, string phone, string address, byte[] img)
      {
         MySqlCommand command = new MySqlCommand("INSERT
INTO `student`(`StdFirstName`, `StdLastName`, `Birthdate`, `Gender`,
`Phone`, `Address`, `Photo`) VALUES(@fn, @ln, @bd, @gd, @ph,
@adr, @img)",connect.getconnection);

         //@fn, @ln, @bd, @gd, @ph, @adr, @img
```

```
        command.Parameters.Add("@fn",
MySqlDbType.VarChar).Value = fname;
        command.Parameters.Add("@ln",
MySqlDbType.VarChar).Value = lname;
        command.Parameters.Add("@bd",   MySqlDbType.Date).Value
= bdate;
        command.Parameters.Add("@gd",
MySqlDbType.VarChar).Value = gender;
        command.Parameters.Add("@ph",
MySqlDbType.VarChar).Value = phone;
        command.Parameters.Add("@adr",
MySqlDbType.VarChar).Value = address;
        command.Parameters.Add("@img", MySqlDbType.Blob).Value
= img;

        connect.openConnect();
        if (command.ExecuteNonQuery() == 1)
        {
           connect.closeConnect();
           return true;
        }
        else
        {
           connect.closeConnect();
           return false;
        }

    }
    // to get student table
    public DataTable getStudentlist(MySqlCommand command)
```

```
    {
        command.Connection=connect.getconnection;
        MySqlDataAdapter adapter = new MySqlDataAdapter(command);
        DataTable table = new DataTable();
        adapter.Fill(table);
        return table;
    }
    // Create a function to execute the count query(total, male , female)
    public string exeCount(string query)
    {
        MySqlCommand command = new MySqlCommand(query, connect.getconnection);
        connect.openConnect();
        string count = command.ExecuteScalar().ToString();
        connect.closeConnect();
        return count;
    }
    //to get the total student
    public string totalStudent()
    {
        return exeCount("SELECT COUNT(*) FROM student");
    }
    // to get the male student count
    public string maleStudent()
    {
        return exeCount("SELECT COUNT(*) FROM student WHERE `Gender`='Male'");
    }
```

```
    // to get the female student count
    public string femaleStudent()
    {
      return exeCount("SELECT COUNT(*) FROM student WHERE `Gender`='Female'");
    }


  //create a function search for student (first name, last name, address)
  public DataTable searchStudent(string searchdata)
    {
      MySqlCommand command = new MySqlCommand("SELECT * FROM `student` WHERE CONCAT(`StdFirstName`,`StdLastName`,`Address`) LIKE '%"+ searchdata +"%'", connect.getconnection);
      MySqlDataAdapter adapter = new MySqlDataAdapter(command);
      DataTable table = new DataTable();
      adapter.Fill(table);
      return table;
    }
    //create a function edit for student
    public bool updateStudent(int id,string fname, string lname, DateTime bdate, string gender, string phone, string address, byte[] img)
    {
      MySqlCommand command = new MySqlCommand("UPDATE `student` SET `StdFirstName`=@fn,`StdLastName`=@ln,`Birthdate`=@bd,`Gender`
```

```
=@gd,`Phone`=@ph,`Address`=@adr,`Photo`=@img WHERE
`StdId`= @id", connect.getconnection);

        //@id,@fn, @ln, @bd, @gd, @ph, @adr, @img
        command.Parameters.Add("@id",   MySqlDbType.Int32).Value
= id;
        command.Parameters.Add("@fn",
MySqlDbType.VarChar).Value = fname;
        command.Parameters.Add("@ln",
MySqlDbType.VarChar).Value = lname;
        command.Parameters.Add("@bd",   MySqlDbType.Date).Value
= bdate;
        command.Parameters.Add("@gd",
MySqlDbType.VarChar).Value = gender;
        command.Parameters.Add("@ph",
MySqlDbType.VarChar).Value = phone;
        command.Parameters.Add("@adr",
MySqlDbType.VarChar).Value = address;
        command.Parameters.Add("@img", MySqlDbType.Blob).Value
= img;

        connect.openConnect();
        if (command.ExecuteNonQuery() == 1)
        {
           connect.closeConnect();
           return true;
        }
        else
        {
           connect.closeConnect();
```

```
            return false;
        }

    }
    //Create a function to delete data
    //we need only id
    public bool deleteStudent(int id)
    {
        MySqlCommand command = new MySqlCommand("DELETE
FROM `student` WHERE `StdId`=@id", connect.getconnection);

        //@id
        command.Parameters.Add("@id",   MySqlDbType.Int32).Value
= id;

        connect.openConnect();
        if (command.ExecuteNonQuery() == 1)
        {
            connect.closeConnect();
            return true;
        }
        else
        {
            connect.closeConnect();
            return false;
        }

    }
    // create a function for any command in studentDb
    public DataTable getList(MySqlCommand command)
```

```
      {
         command.Connection = connect.getconnection;
         MySqlDataAdapter adapter = new
MySqlDataAdapter(command);
         DataTable table = new DataTable();
         adapter.Fill(table);
         return table;
      }
   }
}
```

6.2 Manage Student Form

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using MySql.Data.MySqlClient;

namespace Transparent_Form
{
   public partial class ManageStudentForm : Form
   {
      StudentClass student = new StudentClass();
```

```
public ManageStudentForm()
{
    InitializeComponent();
}

private void ManageStudentForm_Load(object sender, EventArgs e)
{
    showTable();
}
// To show student list in DatagridView
public void showTable()
{
    DataGridView_student.DataSource = student.getStudentlist(new MySqlCommand("SELECT * FROM `student`"));
    DataGridViewImageColumn imageColumn = new DataGridViewImageColumn();
    imageColumn = (DataGridViewImageColumn)DataGridView_student.Columns[7];
    imageColumn.ImageLayout = DataGridViewImageCellLayout.Zoom;
}

//Display student data from student to textbox
private void DataGridView_student_Click(object sender, EventArgs e)
{
    textBox_id.Text = DataGridView_student.CurrentRow.Cells[0].Value.ToString();
```

```
        textBox_Fname.Text =
DataGridView_student.CurrentRow.Cells[1].Value.ToString();
        textBox_Lname.Text =
DataGridView_student.CurrentRow.Cells[2].Value.ToString();

        dateTimePicker1.Value =
(DateTime)DataGridView_student.CurrentRow.Cells[3].Value;
        if
(DataGridView_student.CurrentRow.Cells[4].Value.ToString() ==
"Male")
            radioButton_male.Checked = true;

        textBox_phone.Text =
DataGridView_student.CurrentRow.Cells[5].Value.ToString();
        textBox_address.Text =
DataGridView_student.CurrentRow.Cells[6].Value.ToString();
        byte[] img =
(byte[])DataGridView_student.CurrentRow.Cells[7].Value;
        MemoryStream ms = new MemoryStream(img);
        pictureBox_student.Image = Image.FromStream(ms);
    }

    private void button_clear_Click(object sender, EventArgs e)
    {
        textBox_id.Clear();
        textBox_Fname.Clear();
        textBox_Lname.Clear();
        textBox_phone.Clear();
        textBox_address.Clear();
        radioButton_male.Checked = true;
```

```
        dateTimePicker1.Value = DateTime.Now;
        pictureBox_student.Image = null;
    }

    private void button_upload_Click(object sender, EventArgs e)
    {
        // browse photo from your computer
        OpenFileDialog opf = new OpenFileDialog();
        opf.Filter = "Select Photo(*.jpg;*.png;*.gif)|*.jpg;*.png;*.gif";

        if (opf.ShowDialog() == DialogResult.OK)
            pictureBox_student.Image = Image.FromFile(opf.FileName);
    }

    private void button_search_Click(object sender, EventArgs e)
    {
        DataGridView_student.DataSource =
student.searchStudent(textBox_search.Text);
        DataGridViewImageColumn imageColumn = new
DataGridViewImageColumn();
        imageColumn =
(DataGridViewImageColumn)DataGridView_student.Columns[7];
        imageColumn.ImageLayout =
DataGridViewImageCellLayout.Zoom;
    }
    //create a function to verify
    bool verify()
    {
        if ((textBox_Fname.Text == "") || (textBox_Lname.Text == "") ||
            (textBox_phone.Text == "") || (textBox_address.Text == "") ||
```

```
            (pictureBox_student.Image == null))
        {
            return false;
        }
        else
            return true;
    }

    private void button_update_Click(object sender, EventArgs e)
    {
        // update student record
        int id = Convert.ToInt32(textBox_id.Text);
        string fname = textBox_Fname.Text;
        string lname = textBox_Lname.Text;
        DateTime bdate = dateTimePicker1.Value;
        string phone = textBox_phone.Text;
        string address = textBox_address.Text;
        string gender = radioButton_male.Checked ? "Male" : "Female";


        //we need to check student age between 10 and 100

        int born_year = dateTimePicker1.Value.Year;
        int this_year = DateTime.Now.Year;
        if ((this_year - born_year) < 10 || (this_year - born_year) > 100)
        {
            MessageBox.Show("The student age must be between 10 and
100", "Invalid Birthdate", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
```

```
        else if (verify())
        {
            try
            {
                // to get photo from picture box
                MemoryStream ms = new MemoryStream();
                pictureBox_student.Image.Save(ms,
pictureBox_student.Image.RawFormat);
                byte[] img = ms.ToArray();
                if (student.updateStudent(id, fname, lname, bdate, gender,
phone, address, img))
                {
                    showTable();
                    MessageBox.Show("Student data update", "Update
Student", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    button_clear.PerformClick();
                }
            }
            catch (Exception ex)

            {
                MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        else
        {
            MessageBox.Show("Empty Field", "Update Student",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
```

```
        }

        private void button_delete_Click(object sender, EventArgs e)
        {
            //remove the selected Student
            int id = Convert.ToInt32(textBox_id.Text);
            //Show a confirmation message before delete the student
            if (MessageBox.Show("Are you sure you want to remove this
student", "Remove Student", MessageBoxButtons.YesNo,
MessageBoxIcon.Question) == DialogResult.Yes)
            {
                if (student.deleteStudent(id))
                {
                    showTable();
                    MessageBox.Show("Student Removed", "Remove student",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                    button_clear.PerformClick();
                }
            }
        }

        private void textBox_search_TextChanged(object sender,
EventArgs e)
        {

        }
    }
}
```

6.3 Print Student Form

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using DGVPrinterHelper;

namespace Transparent_Form
{
    public partial class PrintStudent : Form
    {
        StudentClass student = new StudentClass();
        DGVPrinter printer = new DGVPrinter();

        public PrintStudent()
        {
            InitializeComponent();
        }

        private void PrintStudent_Load(object sender, EventArgs e)
        {
            showData(new MySqlCommand("SELECT * FROM `student`"));
```

```
    }

    // create a function to show the student list in datagridview
    public void showData(MySqlCommand command)
    {
        DataGridView_student.ReadOnly = true;
        DataGridViewImageColumn imageColumn = new
DataGridViewImageColumn();
        DataGridView_student.DataSource =
student.getList(command);
        // column 7 is the image column index
        imageColumn =
(DataGridViewImageColumn)DataGridView_student.Columns[7];
        imageColumn.ImageLayout =
DataGridViewImageCellLayout.Zoom;
    }

    private void button_search_Click(object sender, EventArgs e)
    {
        //check the radio button
        string selectQuery;
        if (radioButton_all.Checked)
        {
            selectQuery = "SELECT* FROM `student`";
        }
        else if (radioButton_male.Checked)
        {
            selectQuery = "SELECT * FROM `student` WHERE
`Gender`='Male'";
        }
```

```
        else
        {
          selectQuery = "SELECT * FROM `student` WHERE `Gender`='Female'";
        }
        showData(new MySqlCommand(selectQuery));
    }

    private void button_print_Click(object sender, EventArgs e)
    {
        //We need DGVprinter helper for print pdf file
        printer.Title = "Mdemy Students list";
        printer.SubTitle = string.Format("Date: {0}", DateTime.Now.Date);
        printer.SubTitleFormatFlags = StringFormatFlags.LineLimit | StringFormatFlags.NoClip;
        printer.PageNumbers = true;
        printer.PageNumberInHeader = false;
        printer.PorportionalColumns = true;
        printer.HeaderCellAlignment = StringAlignment.Near;
        printer.Footer = "foxlearn";
        printer.FooterSpacing = 15;
        printer.printDocument.DefaultPageSettings.Landscape = true;
        printer.PrintDataGridView(DataGridView_student);
    }

    private void radioButton_female_CheckedChanged(object sender, EventArgs e)
    {
```

```
    }
  }
}
```

6.4 Course Class

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MySql.Data.MySqlClient;

namespace Transparent_Form
{
  class CourseClass
  {
    DBconnect connect = new DBconnect();
    //create a function to insert course
    public bool insetCourse(string cName, int hr, string tn, string sn)
    {
      MySqlCommand command = new MySqlCommand("INSERT INTO `course`(`CourseName`, `CourseHour`, `Tutor_Name`, `Student_Name`) VALUES (@cn,@ch,@tn,@sn)", connect.getconnection);
      //@cn,@ch,@tn,@sn
```

```
        command.Parameters.Add("@cn",
MySqlDbType.VarChar).Value = cName;
        command.Parameters.Add("@ch",  MySqlDbType.Int32).Value
= hr;
        command.Parameters.Add("@tn",
MySqlDbType.VarChar).Value = tn;
        command.Parameters.Add("@sn",
MySqlDbType.VarChar).Value = sn;
        connect.openConnect();
        if (command.ExecuteNonQuery() == 1)
        {
           connect.closeConnect();
           return true;
        }
        else
        {
           connect.closeConnect();
           return false;
        }
     }

     //create a function to get course list
     public DataTable getCourse(MySqlCommand command)
     {
        command.Connection = connect.getconnection;
        MySqlDataAdapter adapter = new
MySqlDataAdapter(command);
        DataTable table = new DataTable();
        adapter.Fill(table);
        return table;
```

```
    }

    //create a update function for course edit
    public bool updateCourse(int id, string cName, int hr, string tn,
string sn)
    {
      MySqlCommand command = new MySqlCommand("UPDATE
`course`
SET`CourseName`=@cn,`CourseHour`=@ch,`Tutor_Name`=@tn,
`Student_Name`=@sn WHERE `SerialID`=@id",
connect.getconnection);
      //@id,@cn,@ch,@desc
      command.Parameters.Add("@id", MySqlDbType.Int32).Value
= id;
      command.Parameters.Add("@cn",
MySqlDbType.VarChar).Value = cName;
      command.Parameters.Add("@ch", MySqlDbType.Int32).Value
= hr;
      command.Parameters.Add("@tn",
MySqlDbType.VarChar).Value = tn;
      command.Parameters.Add("@sn",
MySqlDbType.VarChar).Value = sn;
      connect.openConnect();
      if (command.ExecuteNonQuery() == 1)
      {
        connect.closeConnect();
        return true;
      }
      else
      {
```

```
            connect.closeConnect();
            return false;
        }
    }
    //Create a function to delete a course
    //we only need course id
    public bool deletCourse(int id)
    {
        MySqlCommand command = new MySqlCommand("DELETE
FROM `course` WHERE `SerialID`=@id", connect.getconnection);
        command.Parameters.Add("@id",   MySqlDbType.Int32).Value
= id;
        connect.openConnect();
        if (command.ExecuteNonQuery() == 1)
        {
            connect.closeConnect();
            return true;
        }
        else
        {
            connect.closeConnect();
            return false;
        }
    }
  }
}
```

6.5 Course Form

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;


namespace Transparent_Form
{
   public partial class CourseForm : Form
   {
      CourseClass course = new CourseClass();
      public CourseForm()
      {
         InitializeComponent();
      }

      private void button_add_Click(object sender, EventArgs e)
      {
         if (textBox_Cname.Text == "" || textBox_Chour.Text == "")
         {
            MessageBox.Show("Need Course data", "Field Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
```

```
        }
        else
        {

            string cName = textBox_Cname.Text;
            int chr = Convert.ToInt32(textBox_Chour.Text);
            string tn = textBox_description.Text;
            string sn = textBox_sn.Text;

            if (course.insetCourse(cName, chr, tn, sn))
            {
                showData();
                button_clear.PerformClick();
                MessageBox.Show("New course inserted", "Add Course",
MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
            else
            {
                MessageBox.Show("Course not insert", "Add Course",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }

    private void button_clear_Click(object sender, EventArgs e)
    {
        textBox_Cname.Clear();
        textBox_Chour.Clear();
        textBox_description.Clear();
```

```
        textBox_sn.Clear();
    }

    private void CourseForm_Load(object sender, EventArgs e)
    {
        showData();
    }
    private void showData()
    {
        //to show course list on datagridview
        DataGridView_course.DataSource = course.getCourse(new
MySqlCommand("SELECT * FROM `course`"));
    }

    private void label7_Click(object sender, EventArgs e)
    {

    }

    private void textBox_Cname_TextChanged(object sender,
EventArgs e)
    {

    }

    private void label1_Click(object sender, EventArgs e)
    {

    }
```

```
        private void label2_Click(object sender, EventArgs e)
        {

        }

        private void label5_Click(object sender, EventArgs e)
        {

        }

        private void textBox1_TextChanged(object sender, EventArgs e)
        {

        }

        private void DataGridView_course_CellContentClick(object
sender, DataGridViewCellEventArgs e)
        {

        }

        private void textBox_description_TextChanged(object sender,
EventArgs e)
        {

        }
    }
}
```

6.6 Manage Course

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Transparent_Form
{
   public partial class ManageCourseForm : Form
   {
      CourseClass course = new CourseClass();
      public ManageCourseForm()
      {
         InitializeComponent();
      }

      private void ManageCourseForm_Load(object sender, EventArgs
e)
      {
         showData();
```

```
    }
    // Show data of the course
    private void showData()
    {
        //to show course list on datagridview
        DataGridView_course.DataSource = course.getCourse(new
MySqlCommand("SELECT * FROM `course`"));
    }

    private void button_clear_Click(object sender, EventArgs e)
    {
        textBox_id.Clear();
        textBox_Cname.Clear();
        textBox_Chour.Clear();
        textBox_description.Clear();
        textBox_sn.Clear();
    }

    private void button_Update_Click(object sender, EventArgs e)
    {
        if (textBox_Cname.Text == "" || textBox_Chour.Text == ""||
textBox_id.Text.Equals(""))
        {
            MessageBox.Show("Need Course data", "Field Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        else
        {
            int id = Convert.ToInt32(textBox_id.Text);
```

```
            string cName = textBox_Cname.Text;
            int chr = Convert.ToInt32(textBox_Chour.Text);
            string desc = textBox_description.Text;
            string sn = textBox_sn.Text;

            if (course.updateCourse(id, cName, chr, desc, sn))
            {
                showData();
                button_clear.PerformClick();
                MessageBox.Show("course update successfuly", "Update
Course", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
            else
            {
                MessageBox.Show("Error-Course not Edit", "Update
Course", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }

    private void button_delete_Click(object sender, EventArgs e)
    {
        if (textBox_id.Text.Equals(""))
        {
            MessageBox.Show("Need Course Id", "Field Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        else
        {
```

```
            try
            {
                int id = Convert.ToInt32(textBox_id.Text);
                if (course.deletCourse(id))
                {
                    showData();
                    button_clear.PerformClick();
                    MessageBox.Show("course Deleted", "Removed Course",
MessageBoxButtons.OK, MessageBoxIcon.Information);

                }
            }
            catch (Exception ex)

            {
                MessageBox.Show(ex.Message, "Removed Course",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }

    private void DataGridView_course_Click(object sender,
EventArgs e)
    {
        textBox_id.Text =
DataGridView_course.CurrentRow.Cells[0].Value.ToString();
        textBox_Cname.Text =
DataGridView_course.CurrentRow.Cells[1].Value.ToString();
        textBox_Chour.Text =
DataGridView_course.CurrentRow.Cells[2].Value.ToString();
```

```
        textBox_description.Text =
DataGridView_course.CurrentRow.Cells[3].Value.ToString();
        textBox_sn.Text =
DataGridView_course.CurrentRow.Cells[4].Value.ToString();
    }

    private void button_search_Click(object sender, EventArgs e)
    {
        //To Search course and show on datagridview
        DataGridView_course.DataSource = course.getCourse(new
MySqlCommand("SELECT * FROM course WHERE
CONCAT(CourseName, ' ', Student_Name)LIKE '%" +
textBox_search.Text + "%'"));
        textBox_search.Clear();
    }

    private void DataGridView_course_CellContentClick(object
sender, DataGridViewCellEventArgs e)
    {

    }

    private void label7_Click(object sender, EventArgs e)
    {

    }

    private void label5_Click(object sender, EventArgs e)
    {
```

```
        }

        private void label3_Click(object sender, EventArgs e)
        {

        }

        private void textBox_id_TextChanged(object sender, EventArgs
e)
        {

        }

        private void label2_Click(object sender, EventArgs e)
        {

        }
    }
}
```

6.7 Print Course

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;
using System.Windows.Forms;
using DGVPrinterHelper;

namespace Transparent_Form
{
   public partial class PrintCourseForm : Form
   {
      CourseClass course = new CourseClass();
      DGVPrinter printer = new DGVPrinter();
      public PrintCourseForm()
      {
         InitializeComponent();
      }

      private void button_search_Click(object sender, EventArgs e)
      {
         //To Search course and show on datagridview
         DataGridView_student.DataSource = course.getCourse(new
MySqlCommand("SELECT * FROM `course` WHERE
CONCAT(`CourseName`)LIKE '%" + textBox_search.Text + "%'"));
         textBox_search.Clear();
      }

      private void PrintCourseForm_Load(object sender, EventArgs e)
      {
         DataGridView_student.DataSource = course.getCourse(new
MySqlCommand("SELECT * FROM `course`"));
      }
```

```
private void button_print_Click(object sender, EventArgs e)
{
    //We need DGVprinter helper for print pdf file
    printer.Title = "Mdemy Course list";
    printer.SubTitle = string.Format("Date: {0}",
DateTime.Now.Date);
    printer.SubTitleFormatFlags = StringFormatFlags.LineLimit |
StringFormatFlags.NoClip;
    printer.PageNumbers = true;
    printer.PageNumberInHeader = false;
    printer.PorportionalColumns = true;
    printer.HeaderCellAlignment = StringAlignment.Near;
    printer.Footer = "Mdemy";
    printer.FooterSpacing = 15;
    printer.printDocument.DefaultPageSettings.Landscape = true;
    printer.PrintDataGridView(DataGridView_student);
}

private void panel3_Paint(object sender, PaintEventArgs e)
{

}

private void DataGridView_student_CellContentClick(object
sender, DataGridViewCellEventArgs e)
{

}

private void label7_Click(object sender, EventArgs e)
```

```
        {

        }

        private void panel2_Paint(object sender, PaintEventArgs e)
        {

        }

        private void panel1_Paint(object sender, PaintEventArgs e)
        {

        }

        private void textBox_search_TextChanged(object sender,
EventArgs e)
        {

        }
    }
}
```

6.8 Score Class

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;

namespace Transparent_Form
{
   class ScoreClass
   {
      DBconnect connect = new DBconnect();
      //create a function add score
      public bool insertScore(int stdid, string courName, double scor,
string desc)
      {
         MySqlCommand command = new MySqlCommand("INSERT
INTO `score`(`StudentId`, `CourseName`, `Score`, `Description`)
VALUES (@stid,@cn,@sco,@desc)", connect.getconnection);
         //@stid,@cn,@sco,@desc
         command.Parameters.Add("@stid",
MySqlDbType.Int32).Value = stdid;
         command.Parameters.Add("@cn",
MySqlDbType.VarChar).Value = courName;
         command.Parameters.Add("@sco",
MySqlDbType.Double).Value = scor;
         command.Parameters.Add("@desc",
MySqlDbType.VarChar).Value = desc;
         connect.openConnect();
         if (command.ExecuteNonQuery() == 1)
         {
            connect.closeConnect();
            return true;
         }
```

```
        else
        {
            connect.closeConnect();
            return false;
        }
    }
    //create a functon to get list
    public DataTable getList(MySqlCommand command)
    {
        command.Connection = connect.getconnection;
        MySqlDataAdapter adapter = new
MySqlDataAdapter(command);
        DataTable table = new DataTable();
        adapter.Fill(table);
        return table;
    }

    // create a function to check already course score
    public bool checkScore(int stdId, string cName)
    {
        DataTable table = getList(new MySqlCommand("SELECT *
FROM `score` WHERE `StudentId`= '" + stdId + "' AND `CourseName`=
'" + cName + "'"));
        if (table.Rows.Count > 0)
        { return true; }
        else
        { return false; }
    }
    // Create A function to edit score data
```

```
    public bool updateScore(int stdid,string scn, double scor, string
desc)
    {
        MySqlCommand command = new MySqlCommand("UPDATE
`score` SET `Score`=@sco,`Description`=@desc WHERE
`StudentId`=@stid AND `CourseName`=@scn",
connect.getconnection);
        //@stid,@sco,@desc
        command.Parameters.Add("@scn",
MySqlDbType.VarChar).Value = scn;
        command.Parameters.Add("@stid",
MySqlDbType.Int32).Value = stdid;
        command.Parameters.Add("@sco",
MySqlDbType.Double).Value = scor;
        command.Parameters.Add("@desc",
MySqlDbType.VarChar).Value = desc;
        connect.openConnect();
        if (command.ExecuteNonQuery() == 1)
        {
            connect.closeConnect();
            return true;
        }
        else
        {
            connect.closeConnect();
            return false;
        }
    }
    //Create a function to delete a score data
    public bool deleteScore(int id)
```

```
    {
        MySqlCommand command = new MySqlCommand("DELETE
FROM `score` WHERE `StudentId`=@id", connect.getconnection);

        //@id
        command.Parameters.Add("@id",   MySqlDbType.Int32).Value
= id;

        connect.openConnect();
        if (command.ExecuteNonQuery() == 1)
        {
            connect.closeConnect();
            return true;
        }
        else
        {
            connect.closeConnect();
            return false;
        }
    }
  }
}
```

6.9 Score Form

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```

```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Transparent_Form
{
   public partial class ScoreForm : Form
   {
      CourseClass course = new CourseClass();
      StudentClass student = new StudentClass();
      ScoreClass score = new ScoreClass();
      public ScoreForm()
      {
         InitializeComponent();
      }
      //create a function to show data on datagridview score
      private void showScoe()
      {
         DataGridView_student.DataSource = score.getList(new
MySqlCommand("SELECT
score.StudentId,student.StdFirstName,student.StdLastName,score.C
ourseName,score.Score,score.Description FROM student INNER JOIN
score ON score.StudentId=student.StdId"));
      }

      private void ScoreForm_Load(object sender, EventArgs e)
      {
         //populate the combobox with courses name
```

```
        comboBox_course.DataSource = course.getCourse(new MySqlCommand("SELECT * FROM `course`"));
        comboBox_course.DisplayMember = "CourseName";
        comboBox_course.ValueMember = "CourseName";
        // to show data on score datagridview

        //To Display the student list on Datagridview
        DataGridView_student.DataSource = student.getList(new MySqlCommand("SELECT `StdId`,`StdFirstName`,`StdLastName` FROM `student`"));
    }

    private void button_add_Click(object sender, EventArgs e)
    {
        if (textBox_stdId.Text == "" || textBox_score.Text == "")
        {
            MessageBox.Show("Need score data", "Field Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        else
        {
            int stdId = Convert.ToInt32(textBox_stdId.Text);
            string cName = comboBox_course.Text;
            double scor = Convert.ToInt32(textBox_score.Text);
            string desc = textBox_description.Text;
            if (!score.checkScore(stdId, cName))
            {

                if (score.insertScore(stdId, cName, scor, desc))
                {
```

```
                showScoe();
                button_clear.PerformClick();
                MessageBox.Show("New score added", "Add Score",
MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
            else
            {
                MessageBox.Show("Score not added", "Add Score",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        else
        {
            MessageBox.Show("The score for this course are alerady
exists", "Add Score", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
    }
}

private void button_clear_Click(object sender, EventArgs e)
{
    textBox_stdId.Clear();
    textBox_score.Clear();
    comboBox_course.SelectedIndex = 0;
    textBox_description.Clear();
}
```

```
    private void DataGridView_student_Click(object sender, EventArgs e)
    {
        textBox_stdId.Text = DataGridView_student.CurrentRow.Cells[0].Value.ToString();
    }

    private void button_sStudent_Click(object sender, EventArgs e)
    {
        DataGridView_student.DataSource = student.getList(new MySqlCommand("SELECT `StdId`,`StdFirstName`,`StdLastName` FROM `student`"));
    }

    private void button_sScore_Click(object sender, EventArgs e)
    {
        showScoe();
    }
  }
}
```

6.10 Manage Score

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Transparent_Form
{
   public partial class ManageScoreForm : Form
   {
      CourseClass course = new CourseClass();
      ScoreClass score = new ScoreClass();
      public ManageScoreForm()
      {
         InitializeComponent();
      }

      private void ManageScoreForm_Load(object sender, EventArgs
e)
      {
         //populate the combobox with courses name
         comboBox_course.DataSource = course.getCourse(new
MySqlCommand("SELECT * FROM `course`"));
         comboBox_course.DisplayMember = "CourseName";
         comboBox_course.ValueMember = "CourseName";
         // to show score data on datagridview
         showScore();
      }
      public void showScore()
      {
         DataGridView_score.DataSource = score.getList(new
MySqlCommand("SELECT
```

```
score.StudentId,student.StdFirstName,student.StdLastName,score.C
ourseName,score.Score,score.Description FROM student INNER JOIN
score ON score.StudentId=student.StdId"));
    }

    private void button_Update_Click(object sender, EventArgs e)
    {
      if (textBox_stdId.Text == "" || textBox_score.Text == "")
      {
        MessageBox.Show("Need score data", "Field Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
      }
      else
      {
        int stdId = Convert.ToInt32(textBox_stdId.Text);
        string cName = comboBox_course.Text;
        double scor = Convert.ToInt32(textBox_score.Text);
        string desc = textBox_description.Text;

        if (score.updateScore(stdId,cName,scor, desc))
        {
            showScore();
            button_clear.PerformClick();
            MessageBox.Show("Score Edited Complete", "Update
Score", MessageBoxButtons.OK, MessageBoxIcon.Information);

        }
        else
        {
```

```
                MessageBox.Show("Score not edit", "Update Score",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }

        }
    }

    private void button_delete_Click(object sender, EventArgs e)
    {

        if (textBox_stdId.Text == "")
        {
            MessageBox.Show("Field Error- we need student id", "Delete
Score", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        else
        {
            int id = Convert.ToInt32(textBox_stdId.Text);
            if (MessageBox.Show("Are you sure you want to remove this
score", "Delete Score", MessageBoxButtons.YesNo,
MessageBoxIcon.Question) == DialogResult.Yes)
            {
                if (score.deleteScore(id))
                {
                    showScore();
                    MessageBox.Show("Score Removed", "Delete Score",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                    button_clear.PerformClick();
                }
            }
```

```
        }
    }

    private void button_clear_Click(object sender, EventArgs e)
    {
        textBox_stdId.Clear();
        textBox_score.Clear();
        textBox_description.Clear();
        textBox_search.Clear();
    }

    private void DataGridView_course_Click(object sender,
EventArgs e)
    {
        textBox_stdId.Text =
DataGridView_score.CurrentRow.Cells[0].Value.ToString();
        comboBox_course.Text =
DataGridView_score.CurrentRow.Cells[3].Value.ToString();
        textBox_score.Text =
DataGridView_score.CurrentRow.Cells[4].Value.ToString();
        textBox_description.Text =
DataGridView_score.CurrentRow.Cells[5].Value.ToString();
    }

    private void button_search_Click(object sender, EventArgs e)
    {
        DataGridView_score.DataSource = score.getList(new
MySqlCommand("SELECT score.StudentId, student.StdFirstName,
student.StdLastName, score.CourseName, score.Score,
```

```
score.Description FROM student INNER JOIN score ON score.StudentId = student.StdId WHERE CONCAT(student.StdFirstName, student.StdLastName, score.CourseName)LIKE '%"+textBox_search.Text+"%'"));

        }

        private void textBox_search_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
```

6.11 Print Score

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using DGVPrinterHelper;
```

```
namespace Transparent_Form
{
   public partial class PrintScoreForm : Form
   {
      ScoreClass score = new ScoreClass();
      DGVPrinter printer = new DGVPrinter();
      public PrintScoreForm()
      {
         InitializeComponent();
      }

      private void button_search_Click(object sender, EventArgs e)
      {
         DataGridView_score.DataSource = score.getList(new
MySqlCommand("SELECT score.StudentId, student.StdFirstName,
student.StdLastName, score.CourseName, score.Score,
score.Description FROM student INNER JOIN score ON
score.StudentId = student.StdId WHERE
CONCAT(student.StdFirstName, student.StdLastName,
score.CourseName)LIKE '%" + textBox_search.Text + "%'"));
      }

      private void button_print_Click(object sender, EventArgs e)
      {
         //We need DGVprinter helper for print pdf file
         printer.Title = "Mdemy Student score list";
         printer.SubTitle = string.Format("Date: {0}",
DateTime.Now.Date);
         printer.SubTitleFormatFlags = StringFormatFlags.LineLimit |
StringFormatFlags.NoClip;
```

```
        printer.PageNumbers = true;
        printer.PageNumberInHeader = false;
        printer.PorportionalColumns = true;
        printer.HeaderCellAlignment = StringAlignment.Near;
        printer.Footer = "Mdemy";
        printer.FooterSpacing = 15;
        printer.printDocument.DefaultPageSettings.Landscape = true;
        printer.PrintDataGridView(DataGridView_score);
    }

    private void PrintScoreForm_Load(object sender, EventArgs e)
    {
        showScore();
    }
    //to show score list
    public void showScore()
    {
        DataGridView_score.DataSource = score.getList(new
MySqlCommand("SELECT
score.StudentId,student.StdFirstName,student.StdLastName,score.C
ourseName,score.Score,score.Description FROM student INNER JOIN
score ON score.StudentId=student.StdId"));
    }
  }
}
```

6.12 DashBoard

```
using MySql.Data.MySqlClient;
using System;
```

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

namespace Transparent_Form
{
   public partial class MainForm : Form
   {
      StudentClass student = new StudentClass();
      CourseClass course = new CourseClass();
      public MainForm()
      {
         InitializeComponent();
         customizeDesign();
      }

      private void MainForm_Load(object sender, EventArgs e)
      {
         studentCount();
         //populate the combobox with courses name
         comboBox_course.DataSource = course.getCourse(new
MySqlCommand("SELECT * FROM `course`"));
         comboBox_course.DisplayMember = "CourseName";
         comboBox_course.ValueMember = "CourseName";
```

```
    }

    //create a function to display student count
    private void studentCount()
    {
        //Display the values
        label_totalStd.Text = "Total Students : " +
student.totalStudent();
        label_maleStd.Text = "Male : " + student.maleStudent();
        label_femaleStd.Text = "Female : " + student.femaleStudent();

    }


    private void customizeDesign()
    {
        panel_stdsubmenu.Visible = false;
        panel_courseSubmenu.Visible = false;
        panel_scoreSubmenu.Visible = false;

    }

    private void hideSubmenu()
    {
        if (panel_stdsubmenu.Visible == true)
            panel_stdsubmenu.Visible = false;
        if (panel_courseSubmenu.Visible == true)
            panel_courseSubmenu.Visible = false;
        if (panel_scoreSubmenu.Visible == true)
            panel_scoreSubmenu.Visible = false;
```

```
    }

    private void showSubmenu(Panel submenu)
    {
        if (submenu.Visible == false)
        {
            hideSubmenu();
            submenu.Visible = true;
        }
        else
            submenu.Visible = false;
    }

    private void button_std_Click(object sender, EventArgs e)
    {
        showSubmenu(panel_stdsubmenu);
    }
    #region StdSubmenu
    private void button_registration_Click(object sender, EventArgs
e)
    {
        openChildForm(new RegisterForm());

        hideSubmenu();

    }

    private void button_manageStd_Click(object sender, EventArgs
e)
    {
```

```
    openChildForm(new ManageStudentForm());
    //...
    //.. TutorHouse
    //...
    hideSubmenu();
}

private void button_status_Click(object sender, EventArgs e)
{
    //...
    //.. TutorHouse
    //...
    hideSubmenu();
}

private void button_stdPrint_Click(object sender, EventArgs e)
{
    openChildForm(new PrintStudent());
    //...
    //.. TutorHouse          //...
    hideSubmenu();
}

#endregion StdSubmenu
private void button_course_Click(object sender, EventArgs e)
{
    showSubmenu(panel_courseSubmenu);
}
#region CourseSubmenu
```

```
    private void button_newCourse_Click(object sender, EventArgs
e)
    {
        openChildForm(new CourseForm());
        //...
        //..TutorHouse
        //...
        hideSubmenu();
    }

    private void button_manageCourse_Click(object sender,
EventArgs e)
    {
        openChildForm(new ManageCourseForm());
        //...
        //..TutorHouse
        //...
        hideSubmenu();
    }

    private void button_coursePrint_Click(object sender, EventArgs
e)
    {
        openChildForm(new PrintCourseForm());
        //...
        //..TutorHouse
        //...
        hideSubmenu();
    }
    #endregion CourseSubmenu
```

```
private void button_score_Click(object sender, EventArgs e)
{
   showSubmenu(panel_scoreSubmenu);
}
#region ScoreSubmenu
private void button_newScore_Click(object sender, EventArgs e)
{
   openChildForm(new ScoreForm());
   //...
   //..Your code
   //...
   hideSubmenu();
}

private void button_manageScore_Click(object sender, EventArgs e)
{
   openChildForm(new ManageScoreForm());

   hideSubmenu();
}

private void button_scorePrint_Click(object sender, EventArgs e)
{
   openChildForm(new PrintScoreForm());
   hideSubmenu();
}
```

```
#endregion ScoreSubmenu

//to show register form in mainform
private Form activeForm = null;
private void openChildForm(Form childForm)
{
   if (activeForm != null)
      activeForm.Close();
   activeForm = childForm;
   childForm.TopLevel = false;
   childForm.FormBorderStyle = FormBorderStyle.None;
   childForm.Dock = DockStyle.Fill;
   panel_main.Controls.Add(childForm);
   panel_main.Tag = childForm;
   childForm.BringToFront();
   childForm.Show();

}

private void button_exit_Click(object sender, EventArgs e)
{
   if (activeForm != null)
      activeForm.Close();
   panel_main.Controls.Add(panel_cover);
   studentCount();
}

private void button_exit_Click_1(object sender, EventArgs e)
{
   LoginForm login = new LoginForm();
```

```
        this.Hide();
        login.Show();
    }

    private void comboBox_course_SelectedIndexChanged(object sender, EventArgs e)
    {
        label_cmale.Text = "Male : " + student.exeCount("SELECT COUNT(*) FROM student INNER JOIN score ON score.StudentId = student.StdId WHERE score.CourseName = '"+comboBox_course.Text+"' AND student.Gender = 'Male'");
        label_cfemale.Text = "Female : " + student.exeCount("SELECT COUNT(*) FROM student INNER JOIN score ON score.StudentId = student.StdId WHERE score.CourseName = '" + comboBox_course.Text + "' AND student.Gender = 'Female'");
    }


    private void label12_Click(object sender, EventArgs e)
    {

    }

    private void label4_Click(object sender, EventArgs e)
    {

    }

    private void label2_Click(object sender, EventArgs e)
    {
```

```
        }

        private void panel_logo_Paint(object sender, PaintEventArgs e)
        {

        }

        private void label3_Click(object sender, EventArgs e)
        {

        }

        private void pictureBox2_Click(object sender, EventArgs e)
        {

        }

        private void label1_Click(object sender, EventArgs e)
        {

        }
    }
}
```

6.13 Database Connection

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using MySql.Data.MySqlClient;
namespace Transparent_Form
{
    /*
     * In this class Create the connection between application and mysql
database
     * we need to install xampp and mysql connector to this project
     * we need to create the student database
     */

    class DBconnect
    {
        //to create connection
        MySqlConnection connect = new
MySqlConnection("datasource=localhost;port=3306;username=root
;password=;database=studentdb");

        //to get connection
        public MySqlConnection getconnection
        {
            get
            {
                return connect;
            }
        }

        //create a function to Open conncetion
        public void openConnect()
```

```
        {
            if (connect.State == System.Data.ConnectionState.Closed)
                connect.Open();
        }

        //Create a fuction to close connection
        public void closeConnect()
        {
            if (connect.State == System.Data.ConnectionState.Open)
                connect.Close();
        }
    }
}
```

Chapter 7 : ER-DIAGRAM

7.1 ER-Diagram :