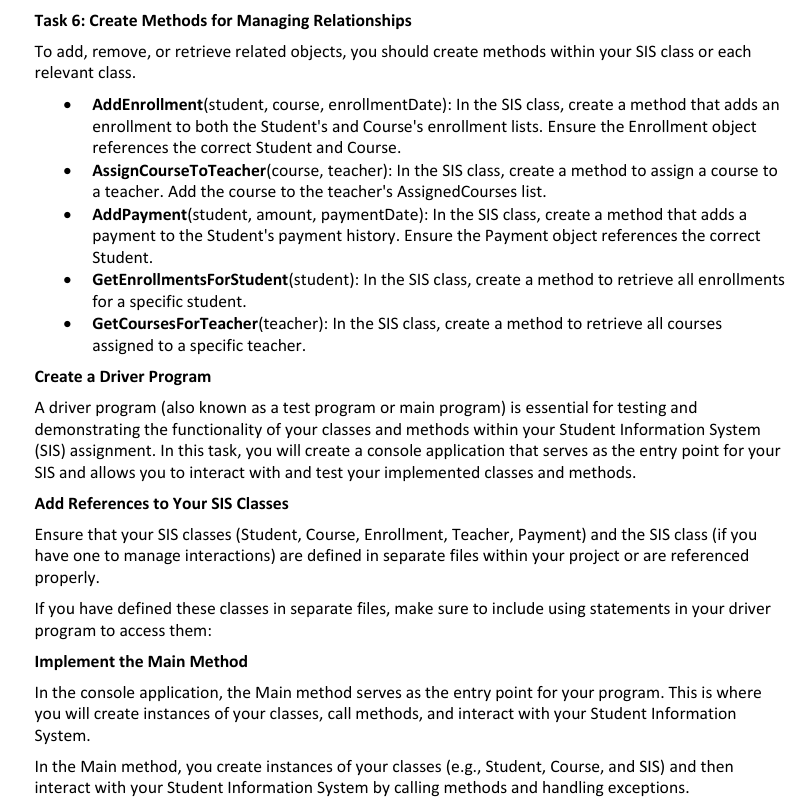
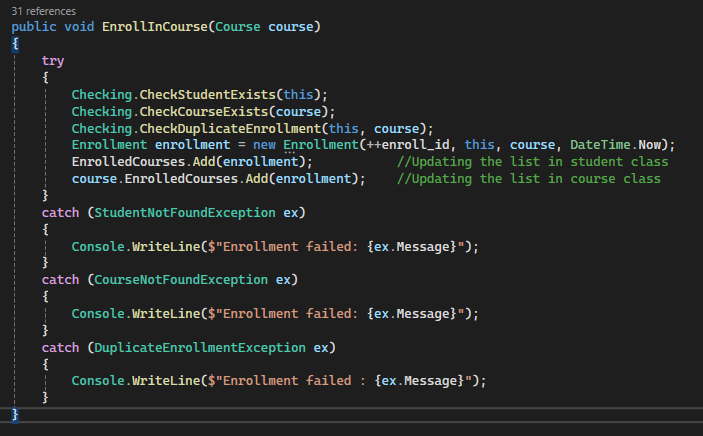
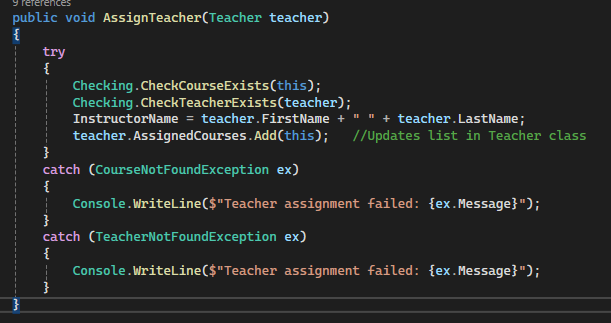
**Task 6 Questions**

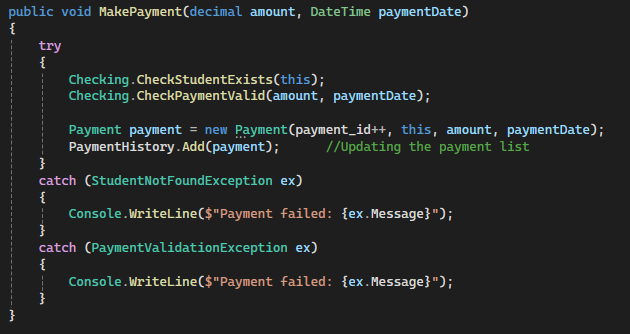


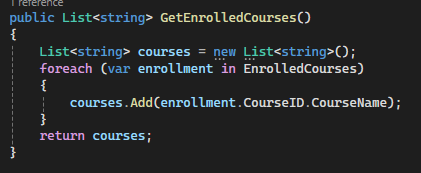
**Console application output screenshots are at the end**

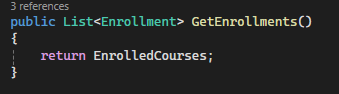
**Creating methods for managing relations in each class**











**The Sis Class**

*This class’s methods interact with the other classes to get the output*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using SIS.CoustomExceptions;

namespace SIS.Classes

{

internal class Sis

{

public void EnrollStudentInCourse(Student student, Course course)

{

try

{

student.EnrollInCourse(course);

}

catch (DuplicateEnrollmentException ex)

{

Console.WriteLine($"Enrollment failed : {ex.Message}");

}

}

public void AssignTeacherToCourse(Teacher teacher, Course course)

{

course.AssignTeacher(teacher);

}

public void RecordPayment(Student student, decimal amount, DateTime paymentDate)

{

student.MakePayment(amount, paymentDate);

}

public List<string> GenerateEnrollmentReport(Course course)

{

List<string> enrolledStudents = new List<string>();

foreach (var enrollment in course.GetEnrollments())

{

enrolledStudents.Add(enrollment.StudentID.FirstName + " " + enrollment.StudentID.LastName);

}

return enrolledStudents;

}

public List<string> GeneratePaymentReport(Student student)

{

List<string> paymentReport = new List<string>();

foreach (var payment in student.GetPaymentHistory())

{

paymentReport.Add($"Amount: {payment.Amount}, Date: {payment.PaymentDate.ToShortDateString()}");

}

return paymentReport;

}

public (int, decimal) CalculateCourseStatistics(Course course)

{

int enrollmentCount = 0;

decimal totalPayments = 0;

foreach (var enrollment in course.GetEnrollments())

{

enrollmentCount++;

foreach (var payment in enrollment.StudentID.GetPaymentHistory())

{

totalPayments += payment.Amount;

}

}

return (enrollmentCount, totalPayments);

}

public void AddStudent()

{

Console.Write("Enter First Name: ");

string firstName = Console.ReadLine();

Console.Write("Enter Last Name: ");

string lastName = Console.ReadLine();

Console.Write("Enter Date of Birth (yyyy-mm-dd): ");

DateTime dob = Convert.ToDateTime(Console.ReadLine());

Console.Write("Enter Email: ");

string email = Console.ReadLine();

Console.Write("Enter Phone Number: ");

string phone = Console.ReadLine();

int id = Student.AllStudents.Count + 1;

Student newStudent = new Student(id, firstName, lastName, dob, email, phone);

}

public void DisplayAllStudents()

{

foreach (var student in Student.AllStudents)

{

student.DisplayStudentInfo();

}

}

public void EnrollStudentInCourse()

{

Console.Write("Enter Student ID: ");

int studentId = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter Course ID: ");

int courseId = Convert.ToInt32(Console.ReadLine());

foreach (var student in Student.AllStudents)

{

if (student.StudentID == studentId)

{

foreach (var course in Course.AllCourses)

{

if (course.CourseID == courseId)

{

student.EnrollInCourse(course);

return;

}

}

}

}

Console.WriteLine("Enrollment failed: Student or Course not found.");

}

public void AddTeacher()

{

Console.Write("Enter First Name: ");

string firstName = Console.ReadLine();

Console.Write("Enter Last Name: ");

string lastName = Console.ReadLine();

Console.Write("Enter Email: ");

string email = Console.ReadLine();

Console.Write("Enter Area of Expertise: ");

string expertise = Console.ReadLine();

int id = Teacher.AllTeachers.Count + 1;

new Teacher(id, firstName, lastName, email, expertise);

}

public void DisplayAllTeachers()

{

foreach (var teacher in Teacher.AllTeachers)

{

teacher.DisplayTeacherInfo();

}

}

public void AddCourse()

{

Console.Write("Enter Course Name: ");

string name = Console.ReadLine();

Console.Write("Enter Course Code: ");

string code = Console.ReadLine();

Console.Write("Enter Instructor Name: ");

string instructor = Console.ReadLine();

int id = Course.AllCourses.Count + 1;

new Course(id, name, code, instructor);

}

public void DisplayAllCourses()

{

foreach (var course in Course.AllCourses)

{

course.DisplayCourseInfo();

}

}

public void AssignTeacherToCourse()

{

Console.Write("Enter Teacher ID: ");

int teacherId = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter Course ID: ");

int courseId = Convert.ToInt32(Console.ReadLine());

foreach (var teacher in Teacher.AllTeachers)

{

if (teacher.TeacherID == teacherId)

{

foreach (var course in Course.AllCourses)

{

if (course.CourseID == courseId)

{

course.AssignTeacher(teacher);

return;

}

}

}

}

Console.WriteLine("Assignment failed: Teacher or Course not found.");

}

public void MakeStudentPayment()

{

Console.Write("Enter Student ID: ");

int studentId = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter Amount: ");

decimal amount = Convert.ToDecimal(Console.ReadLine());

Console.Write("Enter Date (yyyy-mm-dd): ");

DateTime date = Convert.ToDateTime(Console.ReadLine());

foreach (var student in Student.AllStudents)

{

if (student.StudentID == studentId)

{

student.MakePayment(amount, date);

return;

}

}

Console.WriteLine("Payment failed: Student not found.");

}

}

}

**The Main Class**

*The entry point of the project that interacts with Sis class*

using SIS.Task;

using SIS.Classes;

namespace SIS

{

internal class Program

{

static void Main(string[] args)

{

Sis sis = new Sis();

bool exit = false;

while (!exit)

{

Console.WriteLine("\n--- Student Information System ---");

Console.WriteLine("1. Add Student");

Console.WriteLine("2. Display All Students");

Console.WriteLine("3. Enroll Student in Course");

Console.WriteLine("4. Add Teacher");

Console.WriteLine("5. Display All Teachers");

Console.WriteLine("6. Add Course");

Console.WriteLine("7. Display All Courses");

Console.WriteLine("8. Assign Teacher to Course");

Console.WriteLine("9. Make Student Payment");

Console.WriteLine("0. Exit");

Console.Write("Select an option: ");

string choice = Console.ReadLine();

switch (choice)

{

case "1":

sis.AddStudent();

break;

case "2":

sis.DisplayAllStudents();

break;

case "3":

sis.EnrollStudentInCourse();

break;

case "4":

sis.AddTeacher();

break;

case "5":

sis.DisplayAllTeachers();

break;

case "6":

sis.AddCourse();

break;

case "7":

sis.DisplayAllCourses();

break;

case "8":

sis.AssignTeacherToCourse();

break;

case "9":

sis.MakeStudentPayment();

break;

case "0":

exit = true;

Console.WriteLine("Exiting the system");

break;

default:

Console.WriteLine("Invalid option. Please try again");

break;

}

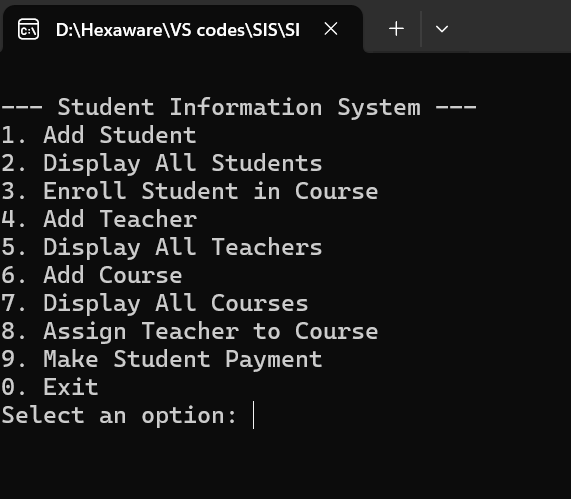
}

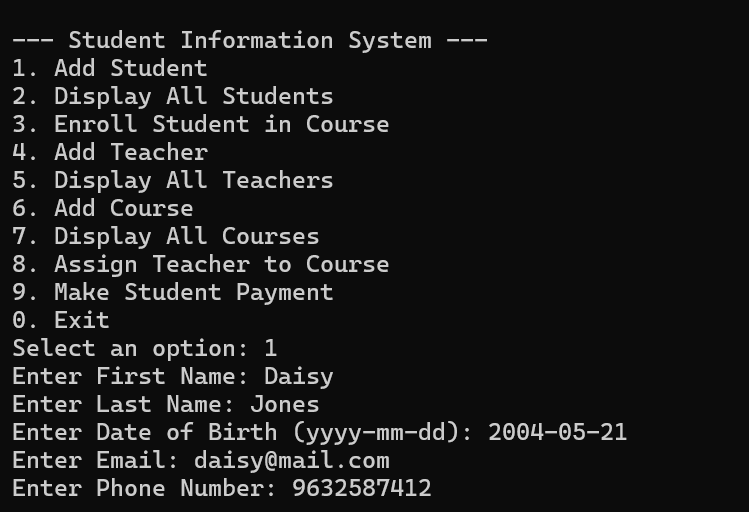
}

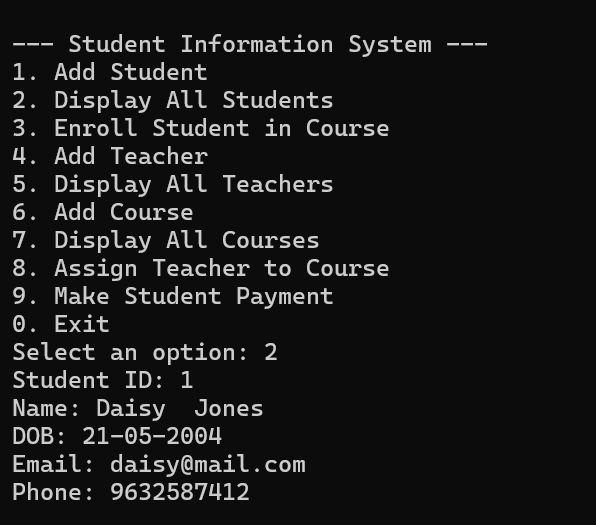
}

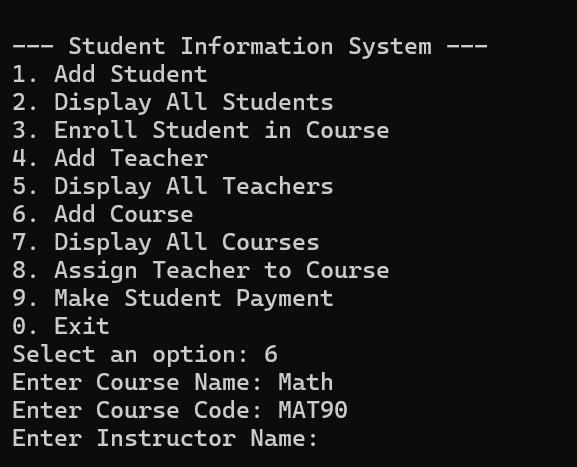
}

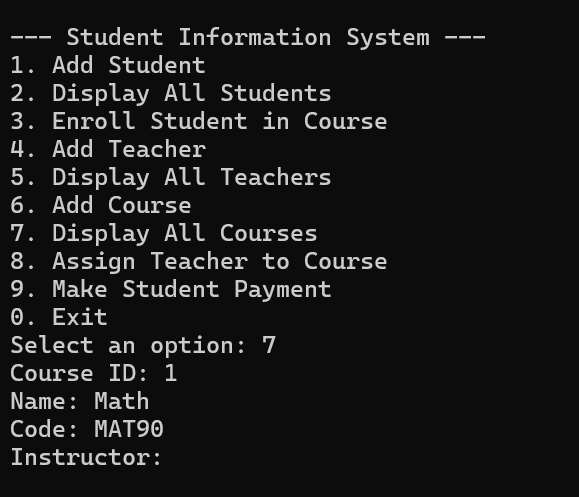
**Console Application Output**

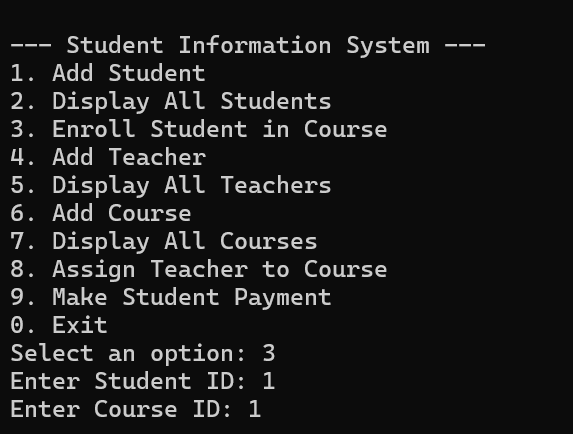


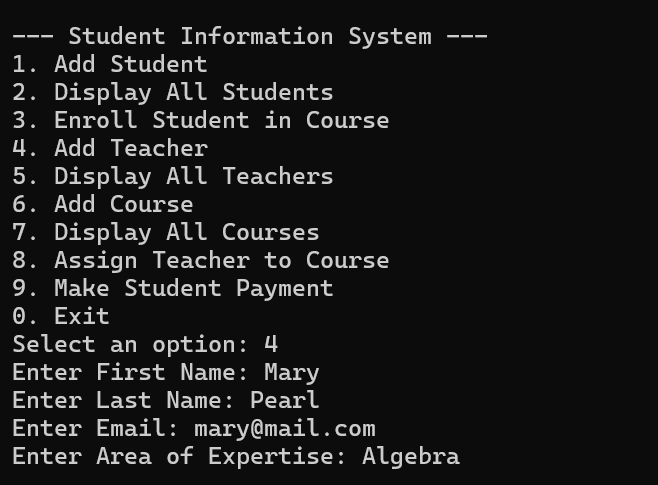


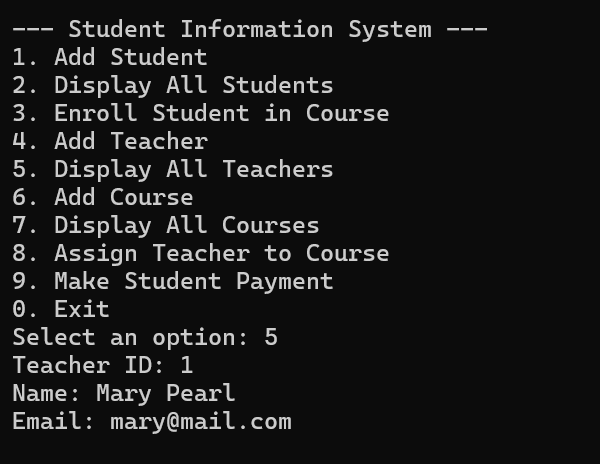


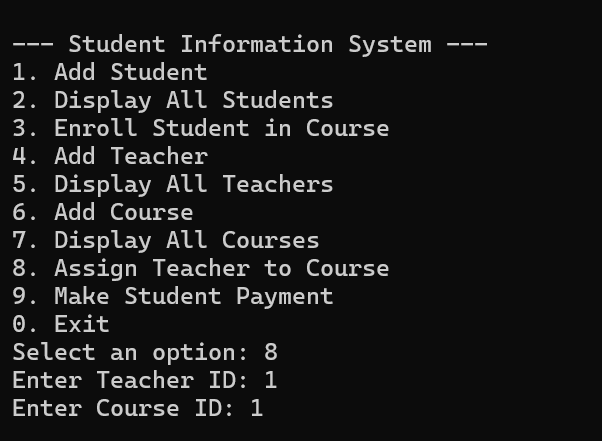


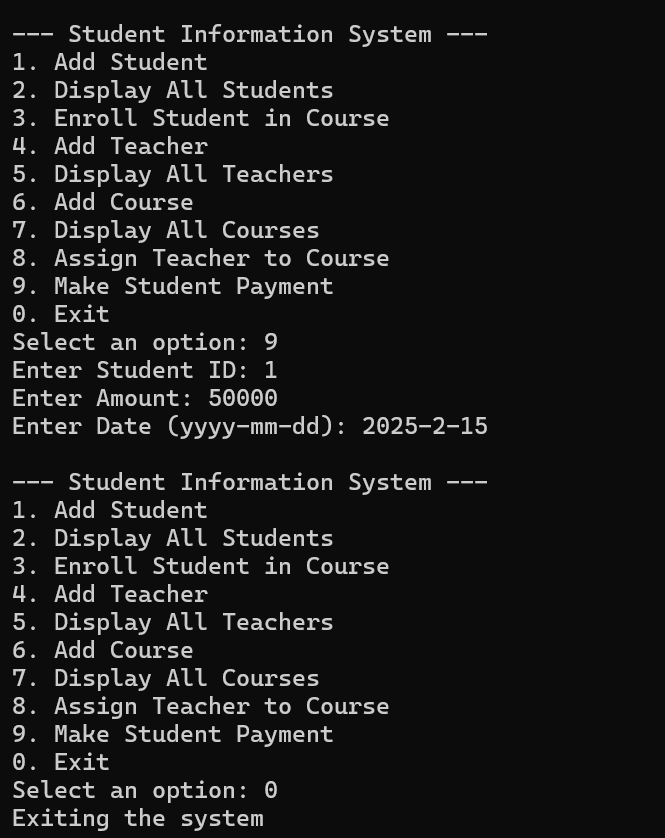












*I have implemented the basic functions and methods of the project, like creating, updating, and viewing the data. There are other methods present within the projects that can perform additional functions. Considering the simplicity, I went with the basic functions for the console application.*