School Management Console Application Documentation- AJJA ANASS

Table of Contents

I. Introduction 2
II. Getting Started 2
1. Prerequisites 2
2. Project Setup 2
3. Designing Classes and Objects 3
III. Application Overview 3
1. Main Menu 3
2. User Login 4
3. Weekly Schedules 5
4. Calendar 9
5. Course Details 10
6. User Management 11
7. Student Management 12
8. Teacher Management 13
9. Administrator Management 14
10. Course Management 15
11. Module Management 16
12. Exit 18
IV. Conclusion 18

I. Introduction

The School Management Console Application is a comprehensive solution designed to streamline administrative tasks and enhance the management of educational institutions. This console application provides users with a range of features, including user authentication, viewing schedules, managing users, students, teachers, courses, modules, and accessing important details related to courses.

Figure 1: Console Application Logo

II. Getting Started

1. Prerequisites

Before Developing and Using the School Management Console Application, ensure that you have the following prerequisites:

- .NET Framework installed.
- Access to the necessary system resources.
- Proper permissions to run the application.

2. Project Setup

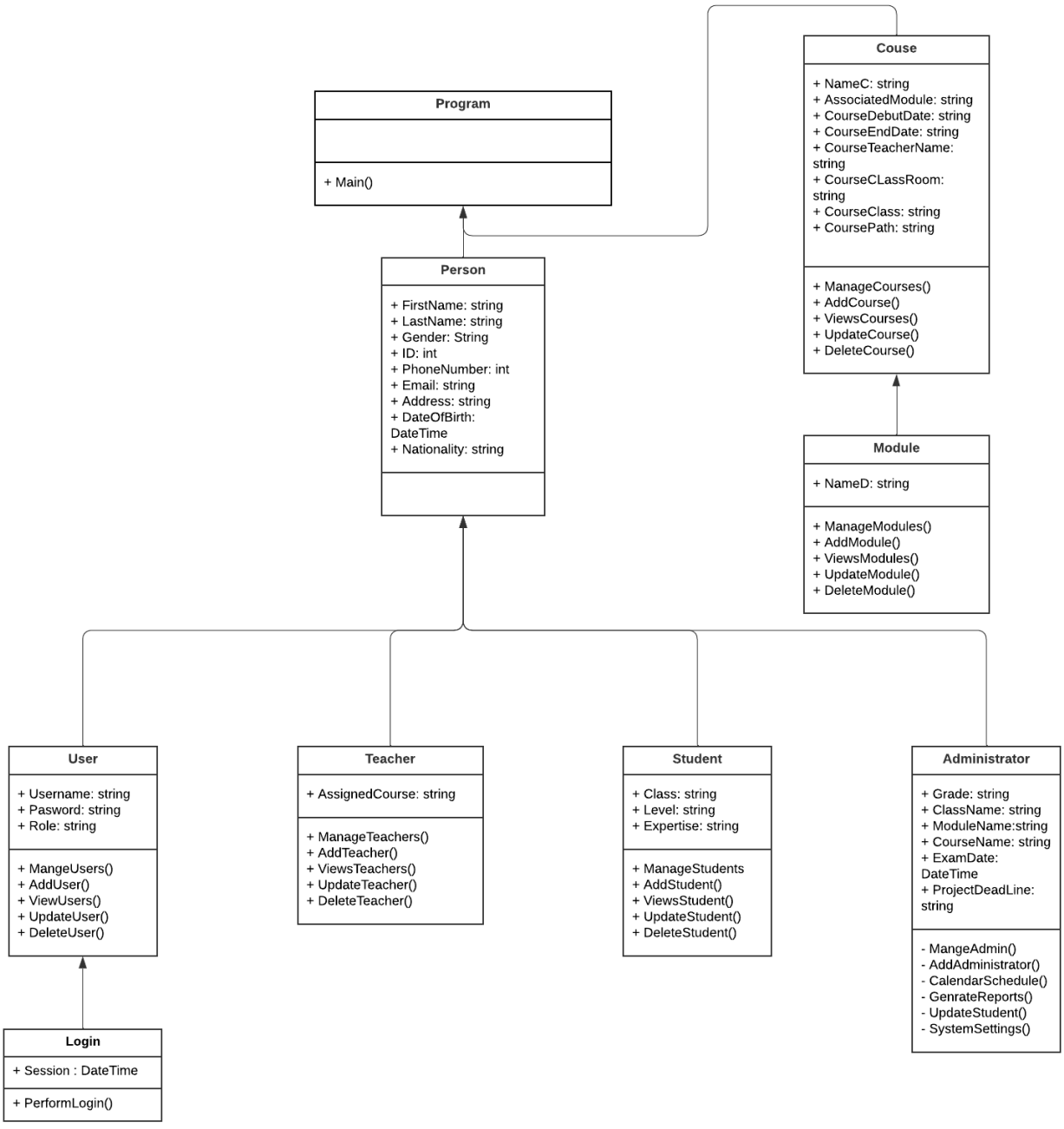
Initializing a new C# console application:

Creating a new console application project in Visual Studio or any C# development environment.

Creating a basic menu system for navigation:

Implement a simple console menu system to allow users to navigate through different features.

3. Designing Classes and Objects



III. Application Overview

1. Main Menu

The main menu provides various options to users, including:

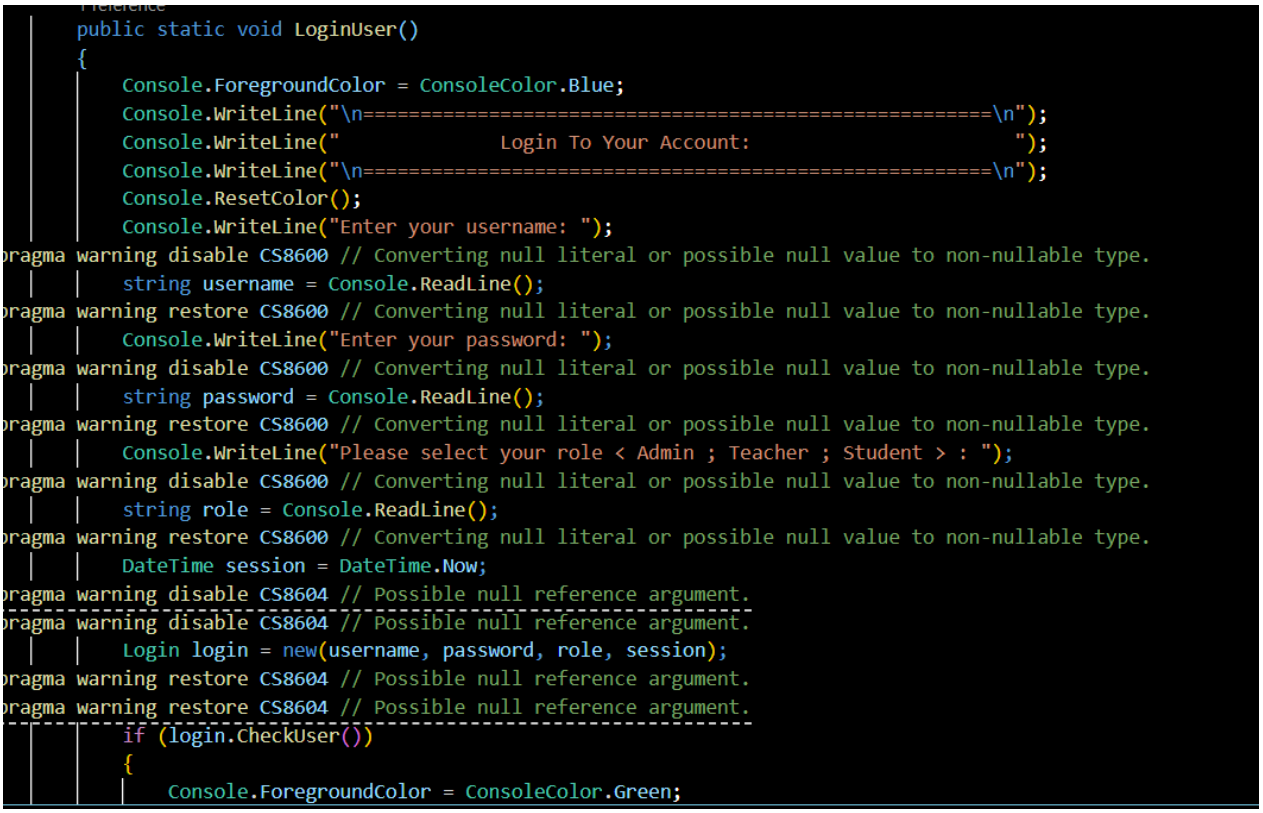
- **Login**: User authentication for accessing the application.
- **View Weekly Schedules**: Display weekly schedules for students.
- **Calendar**: Explore and interact with the academic calendar.
- **Display Course Details**: Access detailed information about courses.
- **User Management**: Administrative tools for managing users.
- **Student Management**: Tools for administrators to manage students.
- **Teacher Management**: Features to manage teachers and their information.
- **Course Management**: Tools for administrators and teachers to manage courses.
- **Module Management**: Functionality for managing course modules.
- **Exit**: Exit the application.

```
Console.ForegroundColor = ConsoleColor.Green;
Console.WriteLine("1. Login < ONLY FOR USERS! >\n");
Console.WriteLine("2. Display Weekly Schedules For Students \n");
Console.WriteLine("3. The Calendar \n");
Console.WriteLine("4. Display Course Details\n");
Console.WriteLine("5. Manage Users < ONLY ADMIN ACCESS ! >\n");
Console.WriteLine("6. Manage Students < ONLY ADMIN ACCESS ! > \n");
Console.WriteLine("7. Manage Teachers < ONLY ADMIN ACCESS ! >\n");
Console.WriteLine("8. Manage Administrators < ONLY ADMIN ACESS ! >\n");
Console.WriteLine("9. Manage Courses< ONLY ADMIN & TEACHERS ACCESS ! >\n");
Console.WriteLine("10. Manage Modules < ONLY ADMIN & TEACHERS ACCESS ! >\n");
Console.WriteLine("11. Exit\n");
Console.ResetColor(); // Reset color to default
Console.Write("Enter your choice: ");
```

Figure 2: Main Menu

2. User Login

Users must log in to access the application. The login process includes authentication mechanisms to ensure secure access.



```
    public static void LoginUser()
    {
        Console.ForegroundColor = ConsoleColor.Blue;
        Console.WriteLine("\n=====================================================\n");
        Console.WriteLine("              Login To Your Account:                 ");
        Console.WriteLine("\n=====================================================\n");
        Console.ResetColor();
        Console.WriteLine("Enter your username: ");
#pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
        string username = Console.ReadLine();
#pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
        Console.WriteLine("Enter your password: ");
#pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
        string password = Console.ReadLine();
#pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
        Console.WriteLine("Please select your role < Admin ; Teacher ; Student > : ");
#pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
        string role = Console.ReadLine();
#pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
        DateTime session = DateTime.Now;
#pragma warning disable CS8604 // Possible null reference argument.
#pragma warning disable CS8604 // Possible null reference argument.
        Login login = new(username, password, role, session);
#pragma warning restore CS8604 // Possible null reference argument.
#pragma warning restore CS8604 // Possible null reference argument.
        if (login.CheckUser())
        {
            Console.ForegroundColor = ConsoleColor.Green;
```

3. Weekly Schedules

Users can view weekly schedules for students, providing an organized overview of academic activities.

```
public static void ViewShedule()
{
    Console.ForegroundColor = ConsoleColor.Blue;
    Console.WriteLine("**************************************\n");
    Console.WriteLine("           View  Weekly Shedule            ");
    Console.WriteLine("\n**************************************\n");
    Console.WriteLine("1. View Weekly Students Shedule\n");
    Console.WriteLine("2. View Weekly Teachers Shedule\n");
    Console.WriteLine("3. Back To Calendar and Schedule Menu\n");
    Console.ResetColor();
    Console.Write("Enter your choice: ");

    int choice = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine();

    switch (choice)
    {
        case 1:
            Student.ViewStudentShedule();
            break;

        case 2:
            Teacher.ViewTeacherShedule();
            break;

        case 3:
            return; // Return to the Calendar and Schedule menu

        default:
            Console.ForegroundColor = ConsoleColor.Red;
            Console.WriteLine("Invalid choice. Try again.\n");
            Console.ResetColor();
            break;
    }
}
```

Figure 3: ViewSchedule

```
421         public static void ViewStudentSchedule()
422         {
423             Console.ForegroundColor = ConsoleColor.Blue;
424             Console.WriteLine("*************************************\n");
425             Console.WriteLine("         View Weekly Student Schedule       ");
426             Console.WriteLine("\n*************************************\n");
427             Console.ResetColor();
428             Console.Write("Enter student Class: ");
429 #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
430             string cclass = Console.ReadLine();
431 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
432             Console.WriteLine("\n");
433 #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
434             Student student = Student.students.Find(s => s.Class == cclass);
435 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
436 #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
437             List<string> weeklySchedule = WeeklyStudentSchedule;
438 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
439 
440             if (weeklySchedule != null && weeklySchedule.Count > 0)
441             {
442                 Console.ForegroundColor = ConsoleColor.Green;
443 #pragma warning disable CS8602 // Dereference of a possibly null reference.
444                 Console.WriteLine($"Students Class: {student.Class}\n");
445 #pragma warning restore CS8602 // Dereference of a possibly null reference.
446                 Console.WriteLine("\nWeekly Schedule:\n");
447 
448                 foreach (var daySchedule in weeklySchedule)
449                 {
450                     Console.WriteLine(daySchedule);
451                 }
452                 Console.ResetColor();
453                 Console.WriteLine("\n");
454             }
455             else
456             {
457                 Console.ForegroundColor = ConsoleColor.Yellow;
458                 Console.WriteLine("No schedule available for this Class.\n");
459                 Console.ResetColor();
460             }
461         }
462 
```

Figure 4: ViewStudentSchedule

And Admins can Manage schedules for students and teachers.

```
257         private static void ManageStudentSchedule()
258         {
259             Console.ForegroundColor = ConsoleColor.Blue;
260             Console.WriteLine("*****************************************************\n");
261             Console.WriteLine("            Manage Weekly Students Schedule            ");
262             Console.WriteLine("\n*****************************************************\n");
263             Console.WriteLine("Enter Weekly Students Schedule Class: ");
264 #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
265             string className = Console.ReadLine();
266 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
267             Console.WriteLine("\n");
268             foreach (var day in WeekDays)
269             {
270                 Console.ForegroundColor = ConsoleColor.Green;
271                 Console.WriteLine($"Enter Weekly Students Schedule Details for {day}: ");
272                 Console.WriteLine("\n");
273                 Console.WriteLine("Enter Weekly Students Schedule Module: ");
274     #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
275                 string moduleName = Console.ReadLine();
276     #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
277                 Console.WriteLine("\n");
278                 Console.WriteLine("Enter Weekly Students Schedule Course: ");
279                 #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
280                 string courseName = Console.ReadLine();
281                 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
282                 Console.WriteLine("\n");
283                 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
284                 Console.WriteLine("\n");
285                 Console.WriteLine("Enter Weekly Students Schedule Debut Time: ");
286                 #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
287                 DateTime DebutTime;
288                 DateTime.TryParse(Console.ReadLine(), out DebutTime);
289                 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
290                 Console.WriteLine("\n");
291                 Console.WriteLine("Enter Weekly Students Schedule End Time: ");
292                 #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
293                 DateTime EndTime;
294                 DateTime.TryParse(Console.ReadLine(), out EndTime);
295                 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
296                 Console.WriteLine("\n");
297                 Console.WriteLine("Enter Weekly Students Schedule Teacher: ");
298                 #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
299                 string teacher = Console.ReadLine();
300                 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
301                 Console.WriteLine("\n");
302                 Console.WriteLine("Enter Weekly Students Schedule ClassRoom: ");
303                 #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
304                 string ClassRoom = Console.ReadLine();
305                 #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
306                 Console.WriteLine("\n");
307 #pragma warning disable CS8602 // Dereference of a possibly null reference.
308                 WeeklyStudentSchedule.Add($"{day}: {moduleName} {courseName} {DebutTime} {EndTime} {teacher} {ClassRoom}");
309 #pragma warning restore CS8602 // Dereference of a possibly null reference.
310                 Console.ResetColor();
311             }
312 #pragma warning disable CS8604 // Possible null reference argument.
313 #pragma warning disable CS8602 // Dereference of a possibly null reference.
314             WeeklyStudentSchedule.Add(className);
315 #pragma warning restore CS8602 // Dereference of a possibly null reference.
316 #pragma warning restore CS8604 // Possible null reference argument.
317             Console.ForegroundColor = ConsoleColor.Green;
318             Console.WriteLine("\nWeekly Students Schedule Added Successfully!\n");
319             Console.ResetColor();
320         }
```

Figure 5: ManageStudentSchedule

```
        private static void ManageTeacherSchedule()
        {
            Console.ForegroundColor = ConsoleColor.Blue;
            Console.WriteLine("**************************************************\n");
            Console.WriteLine("           Manage Weekly Teachers Schedule           ");
            Console.WriteLine("\n**************************************************\n");
            Console.WriteLine("Enter Weekly Teachers Schedule Class: ");
#pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
            string className = Console.ReadLine();
#pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
            Console.WriteLine("\n");
            foreach (var day in WeekDays)
            {
                Console.ForegroundColor = ConsoleColor.Green;
                Console.WriteLine($"Enter Weekly Teachers Schedule Details for {day}: ");
                Console.WriteLine("\n");
                Console.WriteLine("Enter Weekly Teachers Schedule Module: ");
                #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
                string moduleName = Console.ReadLine();
                #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
                Console.WriteLine("\n");
                Console.WriteLine("Enter Weekly Teachers Schedule Course: ");
                #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
                string courseName = Console.ReadLine();
                #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
                Console.WriteLine("\n");
                Console.WriteLine("Enter Weekly Teachers Schedule Debut Time: ");
                #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
                DateTime DebutTime;
                DateTime.TryParse(Console.ReadLine(), out DebutTime);
                #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
                Console.WriteLine("\n");
                Console.WriteLine("Enter Weekly Teachers Schedule End Time: ");
                #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
                DateTime EndTime;
                DateTime.TryParse(Console.ReadLine(), out EndTime);
                #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
                Console.WriteLine("\n");
                Console.WriteLine("Enter Weekly Teachers Schedule Teacher: ");
                #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
                string teacher = Console.ReadLine();
                #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
                Console.WriteLine("\n");
                Console.WriteLine("Enter Weekly Teachers Schedule ClassRoom: ");
                #pragma warning disable CS8600 // Converting null literal or possible null value to non-nullable type.
                string ClassRoom = Console.ReadLine();
                #pragma warning restore CS8600 // Converting null literal or possible null value to non-nullable type.
                Console.WriteLine("\n");
#pragma warning disable CS8602 // Dereference of a possibly null reference.
                WeeklyTeacherSchedule.Add($"{day}: {moduleName} {courseName} {DebutTime} {EndTime} {teacher} {ClassRoom}");
#pragma warning restore CS8602 // Dereference of a possibly null reference.
                Console.ResetColor();
            }
#pragma warning disable CS8604 // Possible null reference argument.
#pragma warning disable CS8602 // Dereference of a possibly null reference.
            WeeklyTeacherSchedule.Add(className);
#pragma warning restore CS8602 // Dereference of a possibly null reference.
#pragma warning restore CS8604 // Possible null reference argument.
            Console.ForegroundColor = ConsoleColor.Green;
            Console.WriteLine("\nWeekly Teachers Schedule Added Successfully!\n");
            Console.ResetColor();
        }
```

Figure 6: ManageTeacherSchedule

4. Calendar

The calendar feature enables users to explore and interact with the academic calendar, highlighting important dates and events.

```
504         public static void ViewCalendar()
505         {
506             Console.ForegroundColor = ConsoleColor.Blue;
507             Console.WriteLine("*****************************************\n");
508             Console.WriteLine("            View Calendar             ");
509             Console.WriteLine("\n*****************************************\n");
510             Console.WriteLine("1. View Exam Dates and Deadlines\n");
511             Console.WriteLine("2. View Upcoming Modules and Courses\n");
512             Console.WriteLine("3. Back To Calendar and Schedule Menu\n");
513             Console.ResetColor();
514             Console.Write("Enter your choice: ");
515 
516             int choice = Convert.ToInt32(Console.ReadLine());
517             Console.WriteLine();
518 
519             switch (choice)
520             {
521                 case 1:
522                     ViewExamDatesAndDeadlines();
523                     break;
524 
525                 case 2:
526                     ViewUpcomingModulesAndCourses();
527                     break;
528 
529                 case 3:
530                     return; // Return to the Calendar and Schedule menu
531 
532                 default:
533                     Console.ForegroundColor = ConsoleColor.Red;
534                     Console.WriteLine("Invalid choice. Try again.\n");
535                     Console.ResetColor();
536                     break;
537             }
538         }
539 
            1 reference
540         private static void ManageCalender()
541         {
542             Console.ForegroundColor = ConsoleColor.Blue;
543             Console.WriteLine("*****************************************\n");
544             Console.WriteLine("            Manage Calendar             ");
545             Console.WriteLine("\n*****************************************\n");
546             Console.WriteLine("1. Manage Exam Dates and Deadlines\n");
547             Console.WriteLine("2. Manage Upcoming Modules and Courses\n");
548             Console.WriteLine("3. Back To Calendar and Schedule Menu\n");
549             Console.ResetColor();
550             Console.Write("Enter your choice: ");
551 
552             int choice = Convert.ToInt32(Console.ReadLine());
553             Console.WriteLine();
554 
555             switch (choice)
556             {
557                 case 1:
558                     ManageExamDatesAndDeadlines();
559                     break;
560 
561                 case 2:
562                     ManageUpcomingModulesAndCourses();
563                     break;
564 
565                 case 3:
566                     return; // Return to the Calendar and Schedule menu
567 
568                 default:
569                     Console.ForegroundColor = ConsoleColor.Red;
570                     Console.WriteLine("Invalid choice. Try again.\n");
571                     Console.ResetColor();
572                     break;
573             }
574         }
```

Figure 7: View & Manage Calendar

5. Course Details

Users can display detailed information about courses, providing insights into course content and structure.

```
2 references
public static void ManageCourses()
{
    Console.ForegroundColor = ConsoleColor.Green;
    Console.WriteLine("\n=====================================\n");
    Console.WriteLine("           Courses Management System       ");
    Console.WriteLine("\n=====================================\n");
    Console.WriteLine("1. Add Course\n");
    Console.WriteLine("2. View Courses\n");
    Console.WriteLine("3. Display Course Details\n");
    Console.WriteLine("4. Update Course\n");
    Console.WriteLine("5. Delet Course\n");
    Console.WriteLine("6. Back to main menu\n");
    Console.ResetColor();
    Console.Write("Enter your choice: ");

    int choice = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine('\n');

    switch (choice)
    {
        case 1:
            AddCourse();
            break;

        case 2:
            ViewCourses();
            break;

        case 3:
#pragma warning disable CS8604 // Possible null reference argument.
            DisplayCourseDetails(CoursePath);
#pragma warning restore CS8604 // Possible null reference argument.
            break;

        case 4:
            UpdateCourse();
            break;

        case 5:
            DeleteCourse();
            break;

        case 6:
```

Figure 8: Manage Courses

6. User Management

Administrators have tools to manage users, including adding, updating, and removing user accounts.

```
137         public static void ManageUsers()
138         {
139             Console.ForegroundColor = ConsoleColor.Green;
140             Console.WriteLine("\n=====================================\n");
141             Console.WriteLine("          User Management System           ");
142             Console.WriteLine("\n=====================================\n");
143             Console.WriteLine("User Management");
144             Console.WriteLine("1. Add User");
145             Console.WriteLine("2. View Users");
146             Console.WriteLine("3. Update User");
147             Console.WriteLine("4. Delete User");
148             Console.WriteLine("5. Back to main Menu");
149             Console.ResetColor();
150             Console.Write("Enter your choice: ");
151 
152             int choice = Convert.ToInt32(Console.ReadLine());
153             Console.WriteLine("\n");
154 
155             switch (choice)
156             {
157                 case 1:
158                     AddUser();
159                     break;
160 
161                 case 2:
162                     ViewUsers();
163                     break;
164 
165                 case 3:
166                     UpdateUser();
167                     break;
168 
169                 case 4:
170                     DeleteUser();
171                     break;
172 
173                 case 5:
174                     return; // Return to the main menu
175 
176                 default:
177                     Console.ForegroundColor = ConsoleColor.Red;
178                     Console.WriteLine("Invalid choice. Try again.\n");
179                     Console.ResetColor();
180                     break;
```

Figure 9: Manage Users

7. Student Management

Administrators can manage student information, including enrollment, personal details, and academic records.

```
public static void ManageStudents()
{
    Console.ForegroundColor = ConsoleColor.Green;
    Console.WriteLine("\n=====================================\n");
    Console.WriteLine("          Student Management System        ");
    Console.WriteLine("\n=====================================\n");
    Console.WriteLine("1. Add Student\n");
    Console.WriteLine("2. View Students\n");
    Console.WriteLine("3. Update Student\n");
    Console.WriteLine("4. Delete Student\n");
    Console.WriteLine("5. Back to main menu\n");
    Console.ResetColor(); // Reset color to default
    Console.Write("Enter your choice: ");

    int choice = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine('\n');

    switch (choice)
    {
        case 1:
            AddStudent();
            break;

        case 2:
            ViewStudents();
            break;

        case 3:
            UpdateStudent();
            break;

        case 4:
            DeleteStudent();
            break;

        case 5:
            return; // Retuns to main menu

        default:
            Console.ForegroundColor = ConsoleColor.Red;
            Console.WriteLine("Invalid choice. Try again.\n");
            Console.ResetColor();
            break;
    }
```

Figure 10: Manage Students

8. Teacher Management

Administrators can manage teacher details, ensuring accurate and up-to-date information about faculty members.

```
207         public static void ManageTeachers()
208         {
209             Console.ForegroundColor = ConsoleColor.Green;
210             Console.WriteLine("\n=====================================\n");
211             Console.WriteLine("         Teacher Management System         ");
212             Console.WriteLine("\n=====================================\n");
213             Console.WriteLine("1. Add Teacher\n");
214             Console.WriteLine("2. View Teachers\n");
215             Console.WriteLine("3. Update Teacher\n");
216             Console.WriteLine("4. Delete Teacher\n");
217             Console.WriteLine("5. Back to main menu\n");
218             Console.ResetColor(); // Reset color to default
219             Console.Write("Enter your choice: ");
220 
221             int choice = Convert.ToInt32(Console.ReadLine());
222             Console.WriteLine('\n');
223 
224             switch (choice)
225             {
226                 case 1:
227                     AddTeacher();
228                     break;
229 
230                 case 2:
231                     ViewTeachers();
232                     break;
233 
234                 case 3:
235                     UpdateTeacher();
236                     break;
237 
238                 case 4:
239                     DeleteTeacher();
240                     break;
241 
242                 case 5:
243                     break; // Returns to main menu
244 
245                 default:
246                     Console.ForegroundColor = ConsoleColor.Red;
247                     Console.WriteLine("Invalid choice. Try again.\n");
248                     Console.ResetColor();
249                     break;
```

Figure 11: Mange Teachers

9. Administrator Management

Administrator Management allows authorized users to manage administrators' accounts, providing tools for adding, updating, and removing administrators.

```
48      public static void ManageAdministration()
49      {
50          Console.ForegroundColor = ConsoleColor.Green;
51          Console.WriteLine("\n=================================================\n");
52          Console.WriteLine("                  Administrator Access");
53          Console.WriteLine("\n=================================================\n");
54          Console.WriteLine("1. Manage Students < ONLY ADMIN ACCESS ! >\n");
55          Console.WriteLine("2. Manage Teachers < ONLY ADMIN ACCESS ! >\n");
56          Console.WriteLine("3. Manage Modules < ONLY ADMIN ACCESS ! >\n");
57          Console.WriteLine("4. Manage Courses < ONLY ADMIN ACCESS ! >\n");
58          Console.WriteLine("5. Add Admininstrator < ONLY ADMIN ACCESS ! >\n");
59          Console.WriteLine("6. Calendar and Schedule < ONLY ADMIN ACCESS ! >\n");
60          Console.WriteLine("7. Generate Reports < ONLY ADMIN ACCESS ! >\n");
61          Console.WriteLine("8. System Settings < ONLY ADMIN ACCESS ! >\n");
62          Console.WriteLine("9. Back to main menu\n");
63          Console.ResetColor(); // Reset color to default
64          Console.Write("Enter your choice: ");
65
66          int choice = Convert.ToInt32(Console.ReadLine());
67          Console.WriteLine('\n');
68
69          switch (choice)
70          {
71              case 1:
72                  Student.ManageStudents();
73                  break;
74
75              case 2:
76                  Teacher.ManageTeachers();
77                  break;
78
79              case 3:
80                  Module.ManageModules();
81                  break;
82
83              case 4:
84                  Course.ManageCourses();
85                  break;
86
87              case 5:
88                  AddAdministrator();
89                  break;
90
91              case 6:
```

Figure 12: Manage Administration

10. Course Management

Both administrators and teachers can manage courses, including creating, updating, and removing courses.

```
2 references
public static void ManageCourses()
{
    Console.ForegroundColor = ConsoleColor.Green;
    Console.WriteLine("\n=====================================\n");
    Console.WriteLine("           Courses Management System       ");
    Console.WriteLine("\n=====================================\n");
    Console.WriteLine("1. Add Course\n");
    Console.WriteLine("2. View Courses\n");
    Console.WriteLine("3. Display Course Details\n");
    Console.WriteLine("4. Update Course\n");
    Console.WriteLine("5. Delet Course\n");
    Console.WriteLine("6. Back to main menu\n");
    Console.ResetColor();
    Console.Write("Enter your choice: ");

    int choice = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine('\n');

    switch (choice)
    {
        case 1:
            AddCourse();
            break;

        case 2:
            ViewCourses();
            break;

        case 3:
#pragma warning disable CS8604 // Possible null reference argument.
            DisplayCourseDetails(CoursePath);
#pragma warning restore CS8604 // Possible null reference argument.
            break;

        case 4:
            UpdateCourse();
            break;

        case 5:
            DeleteCourse();
            break;

        case 6:
```

Figure 13: Manage Courses

11. Module Management

The application provides functionality for managing course modules, allowing for efficient organization of course content.

```
138         public static void ManageModules()
139         {
140             Console.ForegroundColor = ConsoleColor.Green;
141             Console.WriteLine("\n=============================================\n");
142             Console.WriteLine("              Modules Management System              ");
143             Console.WriteLine("\n=============================================\n");
144             Console.WriteLine("1. Add Module\n");
145             Console.WriteLine("2. Add Course To Module \n");
146             Console.WriteLine("3. Display Module Details\n");
147             Console.WriteLine("4. Delete Module\n");
148             Console.WriteLine("5. Back to main menu\n");
149             Console.ResetColor(); // Reset color to default
150             Console.Write("Enter your choice: ");
151 
152             int choice = Convert.ToInt32(Console.ReadLine());
153 
154             switch (choice)
155             {
156                 case 1:
157                     AddModule();
158                     break;
159 
160                 case 2:
161                     AddCourseToModule();
162                     break;
163 
164                 case 3:
165                     ViewModules();
166                     break;
167 
168                 case 4:
169                     DeleteModule();
170                     break;
171 
172                 case 5:
173                     return;
174 
175                 default:
176                     Console.ForegroundColor = ConsoleColor.Red;
177                     Console.WriteLine("Invalid choice. Try again.\n");
178                     Console.ResetColor();
179                     break;
180             }
```

Figure 14: Manage Modules

12. Data Persistence

Store and retrieve data (Upload data) from JSON files using OOP Lists.

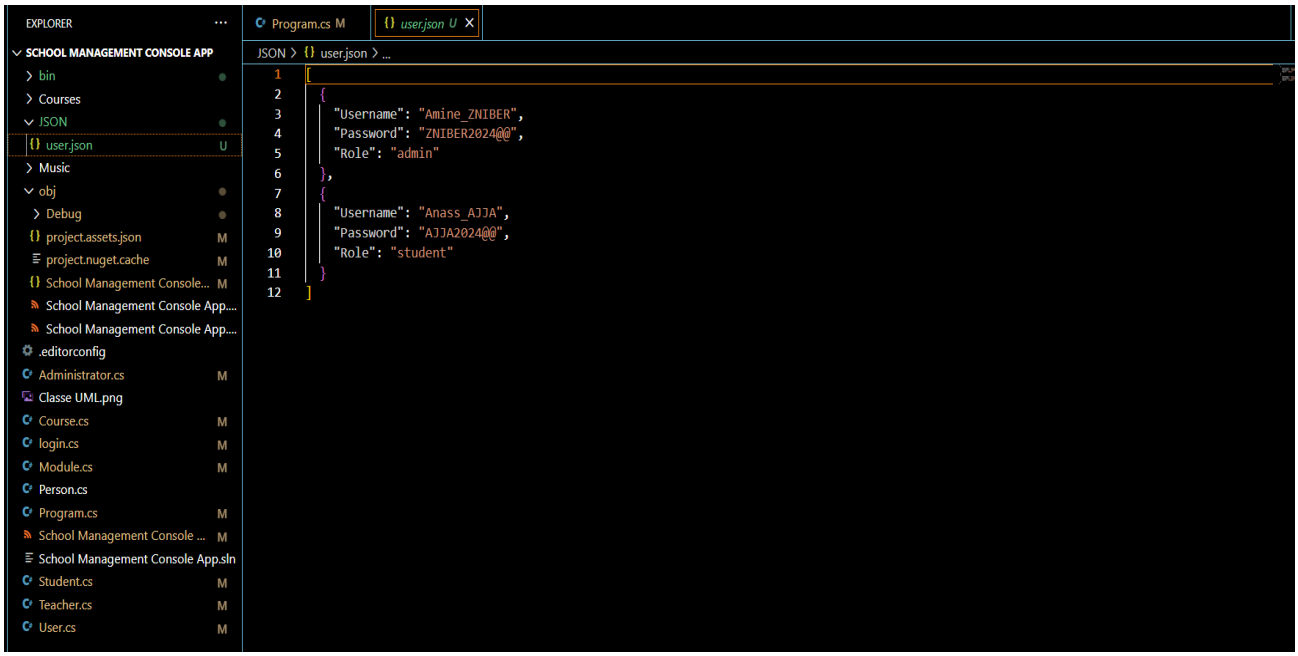


Figure 15: user.json

13. Exit

Users can exit the application at any time using the exit option in the main menu.

IV. Conclusion

In conclusion, the School Management Console Application offers a user-friendly and feature-rich solution for efficiently managing various aspects of educational institutions. Whether it's handling user accounts, overseeing student information, or managing course details, this application provides a comprehensive set of tools to streamline administrative processes.