

MAINFORMDESIGNER.cs

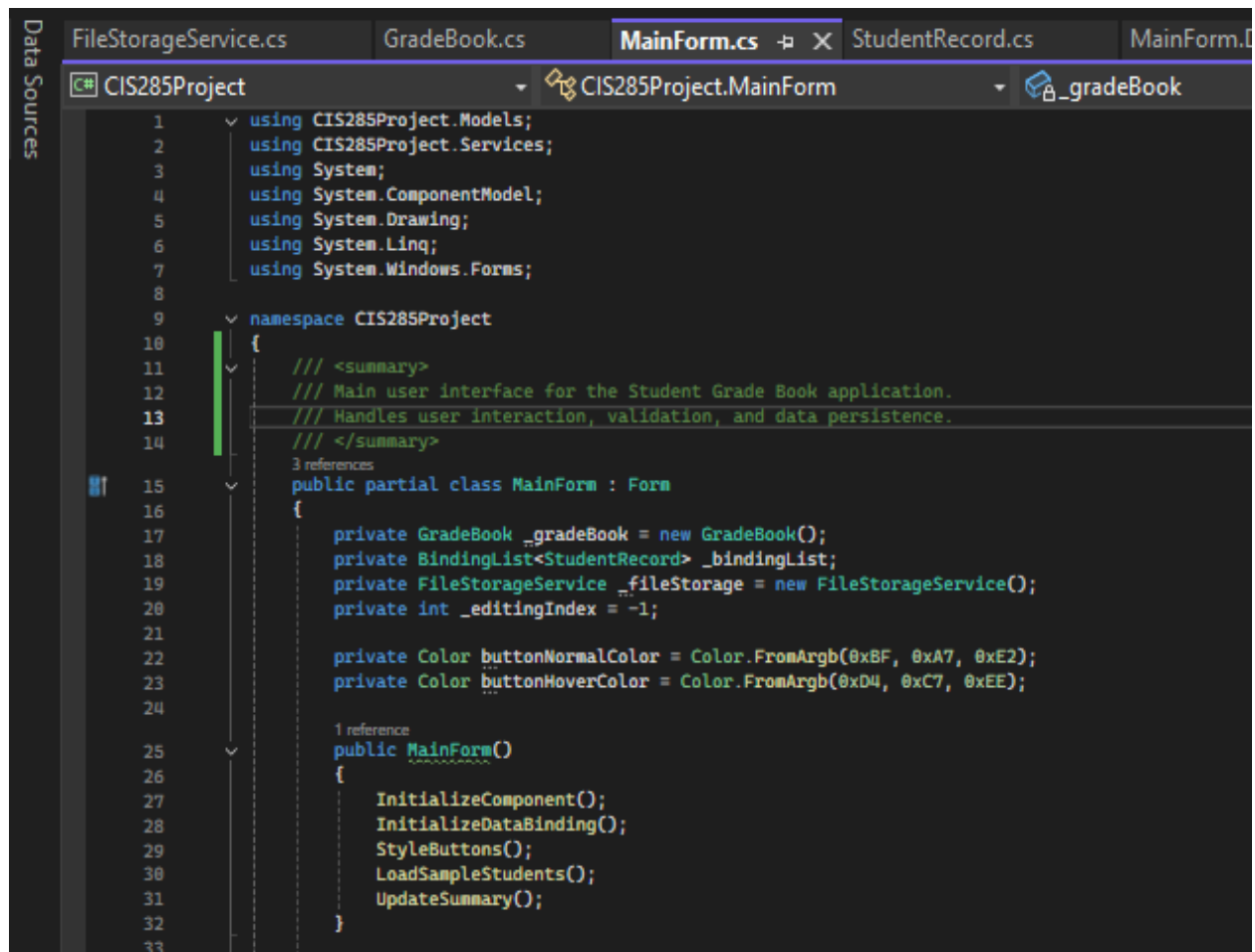
```

FileStorageService.cs  GradeBook.cs  StudentRecord.cs  MainForm.cs  MainForm.Designer.cs
CIS285Project
CIS285Project.MainForm

74
75 // DATAGRIDVIEW
76 this.dgvStudents.Location = new System.Drawing.Point(25, 25);
77 this.dgvStudents.Size = new System.Drawing.Size(800, 350);
78 this.dgvStudents.ReadOnly = true;
79 this.dgvStudents.RowHeadersVisible = false;
80 this.dgvStudents.SelectionMode = System.Windows.Forms.DataGridViewSelectionMode.FullRowSelect;
81 this.dgvStudents.AllowUserToAddRows = false;
82 this.dgvStudents.AllowUserToDeleteRows = false;
83 this.dgvStudents.MultiSelect = false;
84 this.dgvStudents.ScrollBars = System.Windows.Forms.ScrollBars.Both;
85 this.dgvStudents.EnableHeadersVisualStyles = false;
86 this.dgvStudents.ColumnHeadersDefaultCellStyle.BackColor =
87 System.Drawing.Color.FromArgb(0x8F, 0xA7, 0xE2);
88 this.dgvStudents.ColumnHeadersDefaultCellStyle.ForeColor =
89 System.Drawing.Color.White;
90 this.dgvStudents.BackgroundColor = System.Drawing.Color.White;
91
92 this.dgvStudents.Columns.AddRange(new System.Windows.Forms.DataGridViewColumn[] {
93     this.colName, this.colG1, this.colG2, this.colG3, this.colG4,
94     this.colG5, this.colG6, this.colG7, this.colG8, this.colAvg
95 });
96
97 this.dgvStudents.SelectionChanged += new System.EventHandler(this.dgvStudents_SelectionChanged);
98
99 // Column bindings
100 this.colName.DataPropertyName = "Name";
101 this.colG1.DataPropertyName = "Grade1";
102 this.colG2.DataPropertyName = "Grade2";
103 this.colG3.DataPropertyName = "Grade3";
104 this.colG4.DataPropertyName = "Grade4";
105 this.colG5.DataPropertyName = "Grade5";
106 this.colG6.DataPropertyName = "Grade6";
107 this.colG7.DataPropertyName = "Grade7";
108 this.colG8.DataPropertyName = "Grade8";
109 this.colAvg.DataPropertyName = "Average";
110
111 // Column headers
112 this.colName.HeaderText = "Student Name";
113 this.colName.Width = 140;
114
115 this.colG1.HeaderText = "Grade 1"; this.colG1.Width = 60;
116 this.colG2.HeaderText = "Grade 2"; this.colG2.Width = 60;
117 this.colG3.HeaderText = "Grade 3"; this.colG3.Width = 60;
118 this.colG4.HeaderText = "Grade 4"; this.colG4.Width = 60;
119 this.colG5.HeaderText = "Grade 5"; this.colG5.Width = 60;
120 this.colG6.HeaderText = "Grade 6"; this.colG6.Width = 60;
121 this.colG7.HeaderText = "Grade 7"; this.colG7.Width = 60;
122 this.colG8.HeaderText = "Grade 8"; this.colG8.Width = 60;
123
124 this.colAvg.HeaderText = "Average";
125 this.colAvg.Width = 80;
126
127 // BUTTONS
128 this.btnAdd.Location = new System.Drawing.Point(850, 25);
129 this.btnAdd.Size = new System.Drawing.Size(220, 40);
130 this.btnAdd.Text = "Add Student";
131 this.btnAdd.Click += new System.EventHandler(this.btnAdd_Click);
132 this.btnAdd.MouseEnter += new System.EventHandler(this.Button_MouseEnter);
133 this.btnAdd.MouseLeave += new System.EventHandler(this.Button_MouseLeave);
134
135 this.btnEdit.Location = new System.Drawing.Point(850, 75);
136 this.btnEdit.Size = new System.Drawing.Size(220, 40);
137 this.btnEdit.Text = "Edit Student";
138 this.btnEdit.Click += new System.EventHandler(this.btnEdit_Click);
139 this.btnEdit.MouseEnter += new System.EventHandler(this.Button_MouseEnter);
140 this.btnEdit.MouseLeave += new System.EventHandler(this.Button_MouseLeave);
141
142 this.btnDelete.Location = new System.Drawing.Point(850, 125);
143 this.btnDelete.Size = new System.Drawing.Size(220, 40);
144 this.btnDelete.Text = "Delete Student";
145 this.btnDelete.Click += new System.EventHandler(this.btnDelete_Click);
146 this.btnDelete.MouseEnter += new System.EventHandler(this.Button_MouseEnter);
147
70 %  No issues found

```

MAINFORM.cs (MULTIPLE SCREENSHOTS)



```
1  using CIS285Project.Models;
2  using CIS285Project.Services;
3  using System;
4  using System.ComponentModel;
5  using System.Drawing;
6  using System.Linq;
7  using System.Windows.Forms;
8
9  namespace CIS285Project
10 {
11     /// <summary>
12     /// Main user interface for the Student Grade Book application.
13     /// Handles user interaction, validation, and data persistence.
14     /// </summary>
15     3 references
16     public partial class MainForm : Form
17     {
18         private GradeBook _gradeBook = new GradeBook();
19         private BindingList<StudentRecord> _bindingList;
20         private FileStorageService _fileStorage = new FileStorageService();
21         private int _editingIndex = -1;
22
23         private Color buttonNormalColor = Color.FromArgb(0xBF, 0xA7, 0xE2);
24         private Color buttonHoverColor = Color.FromArgb(0xD4, 0xC7, 0xEE);
25
26         1 reference
27         public MainForm()
28         {
29             InitializeComponent();
30             InitializeDataBinding();
31             StyleButtons();
32             LoadSampleStudents();
33             UpdateSummary();
34         }
35     }
36 }
```

```
FileStorageService.cs  GradeBook.cs  MainForm.cs  StudentRecord.cs  MainForm.Designer.cs
CIS285Project  CIS285Project.MainForm  _gradeBook

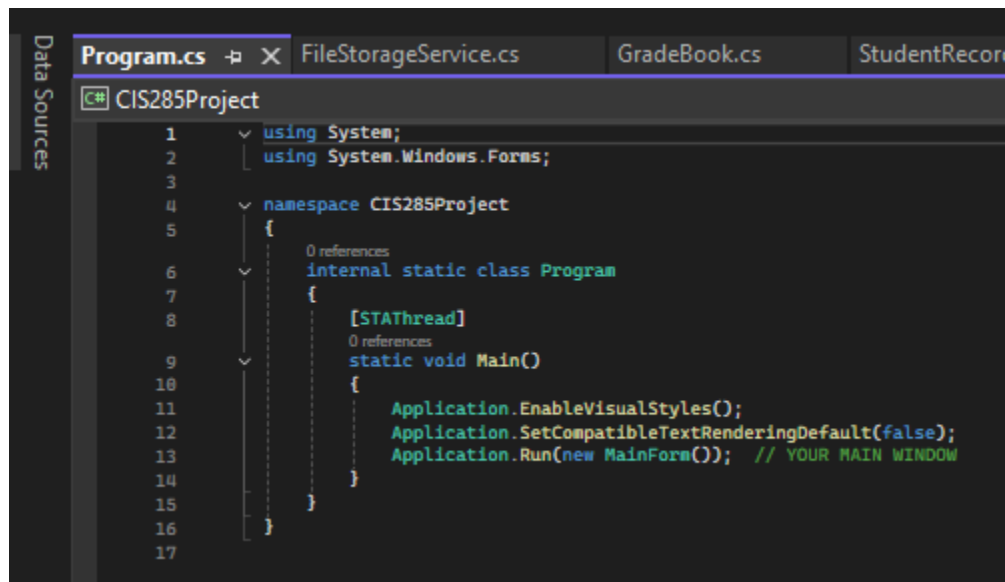
65
66
67
68 // Live validation for grade textboxes
69 8 references
70 private void GradeTextBox_TextChanged(object sender, EventArgs e)
71 {
72     if (sender is TextBox tb)
73     {
74         if (string.IsNullOrEmpty(tb.Text))
75         {
76             tb.BackColor = Color.White;
77             return;
78         }
79         if (double.TryParse(tb.Text, out double value) && value >= 0 && value <= 100)
80             tb.BackColor = Color.White;
81         else
82             tb.BackColor = Color.MistyRose;
83     }
84 }
85
86 // -----
87 // LOAD 19 SAMPLE STUDENTS (8 GRADES)
88 // -----
89 1 reference
90 private void LoadSampleStudents()
91 {
92     _gradeBook.Students.Clear();
93     _gradeBook.AddStudent(new StudentRecord
94     {
95         Name = "Jake Arrieta",
96         Grade1 = 85,
97         Grade2 = 92,
98         Grade3 = 88,
99         Grade4 = 91,
100        Grade5 = 89,
101        Grade6 = 94,
102        Grade7 = 90,
103        Grade8 = 93
104    });
105     _gradeBook.AddStudent(new StudentRecord
106     {
107         Name = "Anthony Rizzo",
108         Grade1 = 78,
109         Grade2 = 82,
110         Grade3 = 80,
111         Grade4 = 75,
112         Grade5 = 79,
113         Grade6 = 81,
114         Grade7 = 77,
115         Grade8 = 83
116     });
117     _gradeBook.AddStudent(new StudentRecord
118     {
119         Name = "Troy Polanalu",
120         Grade1 = 94,
121         Grade2 = 90,
122         Grade3 = 96,
123         Grade4 = 98,
124         Grade5 = 99,
125         Grade6 = 97,
126         Grade7 = 95,
127         Grade8 = 99
128     });
129     _gradeBook.AddStudent(new StudentRecord
130     {
131         Name = "Hines Ward",
132         Grade1 = 67,
133         Grade2 = 72,
```

```
340         Grade7 = 87,
341         Grade8 = 100,
342
343     });
344
345     _bindingList = new BindingList<StudentRecord>(_gradeBook.Students);
346     dgvStudents.DataSource = _bindingList;
347
348     lblStatus.Text = "Loaded 19 active students.";
349 }
350
351 /// <summary>
352 /// Updates all class summary statistics displayed in the UI,
353 /// including averages and passing counts.
354 /// </summary>
355 5 references
356 private void UpdateSummary()
357 {
358     int count = _gradeBook.Students.Count;
359     lblTotalStudents.Text = $"Total Students: {count}";
360     lblSummaryClassAvg.Text = $"Class Avg: {_gradeBook.CalculateClassAverage():F2}";
361
362     if (count > 0)
363     {
364         lblHighestAvg.Text = $"Highest Avg: {_gradeBook.Students.Max(s => s.Average):F2}";
365         lblLowestAvg.Text = $"Lowest Avg: {_gradeBook.Students.Min(s => s.Average):F2}";
366         lblPassingCount.Text = $"Passing (>=70): {_gradeBook.Students.Count(s => s.Average >= 70)}";
367     }
368     else
369     {
370         lblHighestAvg.Text = "Highest Avg: N/A";
371         lblLowestAvg.Text = "Lowest Avg: N/A";
372         lblPassingCount.Text = "Passing: 0";
373     }
374 }
375
376 // -----
377 // BUTTON CLICK EVENTS
378 // -----
379 1 reference
380 private void btnAdd_Click(object sender, EventArgs e)
381 {
382     ClearInputs();
383     _editingIndex = -1;
384     lblStatus.Text = "Adding new student...";
385 }
386
387 1 reference
388 private void btnEdit_Click(object sender, EventArgs e)
389 {
390     if (dgvStudents.CurrentRow == null)
391     {
392         MessageBox.Show("Select a student to edit.");
393         return;
394     }
395
396     _editingIndex = dgvStudents.CurrentRow.Index;
397     var s = _gradeBook.Students[_editingIndex];
398
399     txtName.Text = s.Name;
400     txtGrade1.Text = s.Grade1?.ToString();
401     txtGrade2.Text = s.Grade2?.ToString();
402     txtGrade3.Text = s.Grade3?.ToString();
403     txtGrade4.Text = s.Grade4?.ToString();
404     txtGrade5.Text = s.Grade5?.ToString();
405     txtGrade6.Text = s.Grade6?.ToString();
406     txtGrade7.Text = s.Grade7?.ToString();
407     txtGrade8.Text = s.Grade8?.ToString();
408     txtNotes.Text = s.Notes;
409
410     lblStatus.Text = "Editing student...";
411 }
```

```
FileStorageService.cs  GradeBook.cs  StudentRecord.cs  MainForm.cs  X  Mair
C# CIS285Project  CIS285P

529
530
531 1 reference
532 private void btnAverage_Click(object sender, EventArgs e)
533 {
534     lblClassAverage.Text = _gradeBook.CalculateClassAverage().ToString("F2");
535     lblStatus.Text = "Class average updated.";
536     UpdateSummary();
537 }
538
539 1 reference
540 private void btnSaveFile_Click(object sender, EventArgs e)
541 {
542     SaveFileDialog dialog = new SaveFileDialog();
543     dialog.Filter = "XML Files|*.xml";
544
545     if (dialog.ShowDialog() == DialogResult.OK)
546     {
547         _fileStorage.Save(dialog.FileName, _gradeBook.Students);
548         lblStatus.Text = "File saved.";
549     }
550
551 1 reference
552 private void btnSaveNotes_Click(object sender, EventArgs e)
553 {
554     if (dgvStudents.CurrentRow == null)
555     {
556         MessageBox.Show("Select a student first.");
557         return;
558     }
559
560     int index = dgvStudents.CurrentRow.Index;
561     _gradeBook.Students[index].Notes = txtNotes.Text;
562
563     lblStatus.Text = "Notes saved.";
564 }
565
566 1 reference
567 private void btnLoadFile_Click(object sender, EventArgs e)
568 {
569     OpenFileDialog dialog = new OpenFileDialog();
570     dialog.Filter = "XML Files|*.xml";
571
572     if (dialog.ShowDialog() == DialogResult.OK)
573     {
574         var list = _fileStorage.Load(dialog.FileName);
575
576         _gradeBook.Students.Clear();
577         _gradeBook.Students.AddRange(list);
578
579         _bindingList = new BindingList<StudentRecord>(_gradeBook.Students);
580         dgvStudents.DataSource = _bindingList;
581
582         UpdateSummary();
583         lblStatus.Text = "File loaded.";
584     }
585 }
```

PROGRAM.cs



```
1  using System;
2  using System.Windows.Forms;
3
4  namespace CIS285Project
5  {
6      internal static class Program
7      {
8          [STAThread]
9          static void Main()
10         {
11             Application.EnableVisualStyles();
12             Application.SetCompatibleTextRenderingDefault(false);
13             Application.Run(new MainForm()); // YOUR MAIN WINDOW
14         }
15     }
16 }
17
```

FILESTORAGESERVICE.cs

```
File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Debug Any CPU Start
Data Sources
FileStorageService.cs GradeBook.cs MainForm.cs StudentRecord.cs
CIS285Project CIS285Project.Services.FileStorageService Save(string filePath, List<StudentRecord> students)

10 namespace CIS285Project.Services
11 {
12     /// <summary>
13     /// Provides functionality for saving and loading student records
14     /// to and from XML files.
15     /// </summary>
16     2 references
17     public class FileStorageService
18     {
19         /// <summary>
20         /// Saves a list of student records to an XML file.
21         /// </summary>
22         /// <param name="filePath">The file path where the XML file will be saved.</param>
23         /// <param name="students">The list of student records to save.</param>
24         1 reference
25         public void Save(string filePath, List<StudentRecord> students)
26         {
27             var doc = new XDocument(
28                 new XElement("Students",
29                     students.Select(s =>
30                         new XElement("Student",
31                             new XAttribute("Name", s.Name ?? ""),
32                             new XAttribute("Grade1", s.Grade1?.ToString() ?? ""),
33                             new XAttribute("Grade2", s.Grade2?.ToString() ?? ""),
34                             new XAttribute("Grade3", s.Grade3?.ToString() ?? ""),
35                             new XAttribute("Grade4", s.Grade4?.ToString() ?? ""),
36                             new XAttribute("Grade5", s.Grade5?.ToString() ?? ""),
37                             new XAttribute("Grade6", s.Grade6?.ToString() ?? ""),
38                             new XAttribute("Grade7", s.Grade7?.ToString() ?? ""),
39                             new XAttribute("Grade8", s.Grade8?.ToString() ?? ""),
40                             new XElement("Notes", s.Notes ?? "")
41                         )
42                     )
43                 );
44             doc.Save(filePath);
45         }
46         /// <summary>
47         /// Loads student records from an XML file.
48         /// </summary>
49         /// <param name="filePath">The file path of the XML file to load.</param>
50         /// <returns>A list of student records loaded from the file.</returns>
51         1 reference
52         public List<StudentRecord> Load(string filePath)
53         {
54             var result = new List<StudentRecord>();
55             if (!File.Exists(filePath))
56                 return result;
57             var doc = XDocument.Load(filePath);
58             foreach (var elem in doc.Root.Elements("Student"))
59             {
60                 var s = new StudentRecord
61                 {
62                     Name = (string)elem.Attribute("Name")
63                 };
64                 s.Grade1 = ParseNullableDouble((string)elem.Attribute("Grade1"));
65                 s.Grade2 = ParseNullableDouble((string)elem.Attribute("Grade2"));
66                 s.Grade3 = ParseNullableDouble((string)elem.Attribute("Grade3"));
67                 s.Grade4 = ParseNullableDouble((string)elem.Attribute("Grade4"));
68                 s.Grade5 = ParseNullableDouble((string)elem.Attribute("Grade5"));
69                 s.Grade6 = ParseNullableDouble((string)elem.Attribute("Grade6"));
70                 s.Grade7 = ParseNullableDouble((string)elem.Attribute("Grade7"));
71                 s.Grade8 = ParseNullableDouble((string)elem.Attribute("Grade8"));
72                 s.Notes = (string)elem.Element("Notes") ?? "";
73                 result.Add(s);
74             }
75         }
76     }
77 }
78
79
```

GRADEBOOK.cs


```
FileStorageService.cs  GradeBook.cs  MainForm.cs  StudentRecord.cs  MainF...
C# CIS285Project  CIS285Project.Models.GradeBook  UpdateStuden...
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace CIS285Project.Models
8  {
9      /// <summary>
10     /// Manages a collection of student records and provides
11     /// methods for modifying students and calculating class statistics.
12     /// </summary>
13     4 references
14     public class GradeBook
15     {
16         /// <summary>
17         /// Gets the list of students in the grade book.
18         /// </summary>
19         24 references
20         public List<StudentRecord> Students { get; private set; }
21
22         /// <summary>
23         /// Initializes a new instance of the <see cref="GradeBook"/> class.
24         /// </summary>
25         1 reference
26         public GradeBook()
27         {
28             Students = new List<StudentRecord>();
29         }
30
31         /// <summary>
32         /// Adds a new student to the grade book.
33         /// </summary>
34         /// <param name="s">The student record to add.</param>
35         20 references
36         public void AddStudent(StudentRecord s) => Students.Add(s);
37
38         /// <summary>
39         /// Updates an existing student at the specified index.
40         /// </summary>
41         /// <param name="index">The index of the student to update.</param>
42         /// <param name="s">The updated student record.</param>
43         1 reference
44         public void UpdateStudent(int index, StudentRecord s)
45         {
46             if (index >= 0 && index < Students.Count)
47                 Students[index] = s;
48         }
49
50         /// <summary>
51         /// Removes a student from the grade book at the specified index.
52         /// </summary>
53         /// <param name="index">The index of the student to remove.</param>
54         1 reference
55         public void RemoveStudent(int index)
56         {
57             if (index >= 0 && index < Students.Count)
58                 Students.RemoveAt(index);
59         }
60
61         /// <summary>
62         /// Calculates the average grade across all students.
63         /// </summary>
64         /// <returns>The class average.</returns>
65         2 references
66         public double CalculateClassAverage()
67         {
68             if (Students.Count == 0)
69                 return 0;
70
71             return Students.Average(s => s.Average);
72         }
73     }
74 }
```

STUDENTRECORD.cs

```
FileStorageService.cs  GradeBook.cs  MainForm.cs  StudentRecord.cs  Mai
Data Sources
CIS285Project  CIS285Project.Models.StudentRecord  Grade4

4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace CIS285Project.Models
8  {
9      /// <summary>
10     /// Represents a student record containing the student's name,
11     /// grades, notes, and calculated average.
12     /// </summary>
13     35 references
14     public class StudentRecord
15     {
16         /// <summary>
17         /// The student's full name.
18         /// </summary>
19         23 references
20         public string Name { get; set; }
21
22         /// <summary>First grade.</summary>
23         24 references
24         public double? Grade1 { get; set; }
25
26         /// <summary>Second grade.</summary>
27         24 references
28         public double? Grade2 { get; set; }
29
30         /// <summary>Third grade.</summary>
31         24 references
32         public double? Grade3 { get; set; }
33
34         /// <summary>Fourth grade.</summary>
35         24 references
36         public double? Grade4 { get; set; }
37
38         /// <summary>Fifth grade.</summary>
39         24 references
40         public double? Grade5 { get; set; }
41
42         /// <summary>Sixth grade.</summary>
43         24 references
44         public double? Grade6 { get; set; }
45
46         /// <summary>Seventh grade.</summary>
47         24 references
48         public double? Grade7 { get; set; }
49
50         /// <summary>Eighth grade.</summary>
51         24 references
52         public double? Grade8 { get; set; }
53
54         /// <summary>
55         /// Optional notes associated with the student.
56         /// </summary>
57         6 references
58         public string Notes { get; set; }
59
60         /// <summary>
61         /// Calculates the student's average grade based on all
62         /// non-null grade values.
63         /// </summary>
64         4 references
65         public double Average
66         {
67             get
68             {
69                 var grades = new double?[]
70                 {
71                     Grade1, Grade2, Grade3, Grade4,
72                     Grade5, Grade6, Grade7, Grade8
73                 };
74
75                 var valid = grades.Where(g => g.HasValue).Select(g => g.Value);
76                 return valid.Any() ? valid.Average() : 0;
77             }
78         }
79     }
80 }
```

Source Code Reference:

This project was developed by our group and was not adapted from any external repository or tutorial project.