

Uniwersytet WSB Merito

# Programowanie Zaawansowane

Projekt 1 – Asp.Net Core

Andrii Skoromnyi 144386

# Zadanie 1

**Podzadanie-1** (Internacjonalizacja wszystkich plików widoków dla Student dla PL oraz EN)

Name	Value
Age	Age
Are you sure	Are you sure you want to delete this?
Author	Author
Back to List	Back to List
BackToList	Back to List
Book	Book
Books	Books
building Web apps with ASP.NET Core	building Web apps with ASP.NET Core
Create	Create
Create New	Create New
Credits	Credits
Culture	Culture
Data	Data
Delete	Delete
Details	Details
Edit	Edit
Floor	Floor
Home	Home
Index of Books	Index of Books
Index of lecture rooms	Index of lecture rooms
Index of Lecturers	Index of Lecturers
Index of students	Index of students
Index of subjects	Index of subjects
Learn about	Learn about
Lecture Room	Lecture Room
Lecture Rooms	Lecture Rooms
Lecturer	Lecturer
Lecturers	Lecturers
Major	Major
Name	Name

Name	Value
Age	Wiek
Are you sure	Czy na pewno chcesz to usunąć?
Author	Autor
Back to List	Powrót do listy
BackToList	Powrót do listy
Book	Książka
Books	Książki
building Web apps with ASP.NET Core	tworzenie aplikacji internetowych przy użyciu platformy ASP.NET Core
Create	Tworzyć
Create New	Tworzyć nowe
Credits	Kredyty
Culture	Kultura
Data	Dane
Delete	Usuwać
Details	Detale
Edit	Edytować
Floor	Piętro
Home	Dom
Index of Books	Indeks książek
Index of lecture rooms	Indeks sal wykładowych
Index of Lecturers	Indeks wykładowców
Index of students	Indeks studentów
Index of subjects	Indeks tematyczny
Learn about	Uczyć się o
Lecture Room	Sala Wykładowa
Lecture Rooms	Sale wykładowe
Lecturer	Wykładowca
Lecturers	Wykładowcy
Major	Główny
..	..

## Index of students

[Create New](#)

Name and surname	Age	Major	
Andrii Skoromnyi	18	Informatyka	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

## Indeks studentów

[Tworzyć nowe](#)

Imię i nazwisko	Wiek	Główny	
Andrii Skoromnyi	18	Informatyka	<a href="#">Edytować</a>   <a href="#">Detale</a>   <a href="#">Usuwać</a>

**Podzadanie-2** (Internacjonalizacja wszystkich plików widoków dla Subject dla PL oraz EN)

## Indeks tematyczny

[Tworzyć nowe](#)

Nazwa	Kredyty	
Robotyka	5	<a href="#">Edytować</a>   <a href="#">Detale</a>   <a href="#">Usuwać</a>

## Index of subjects

[Create New](#)

Name	Credits	
Robotyka	5	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

**Podzadanie-3** (Internacjonalizacja całego GUI dla języków PL oraz EN)

Students.Web Dom Dane ▾ Prywatność Kultura ▾

Studenci

Przedmioty

Książki

Salę wykładowe

Wykładowcy

## Indeks studentów

[Tworzyć nowe](#)

Imię i nazwisko	Wiek	Główny	
Jan Skoromnyi	18	Informatyka	<a href="#">Edytować</a>   <a href="#">Detale</a>   <a href="#">Usuwać</a>

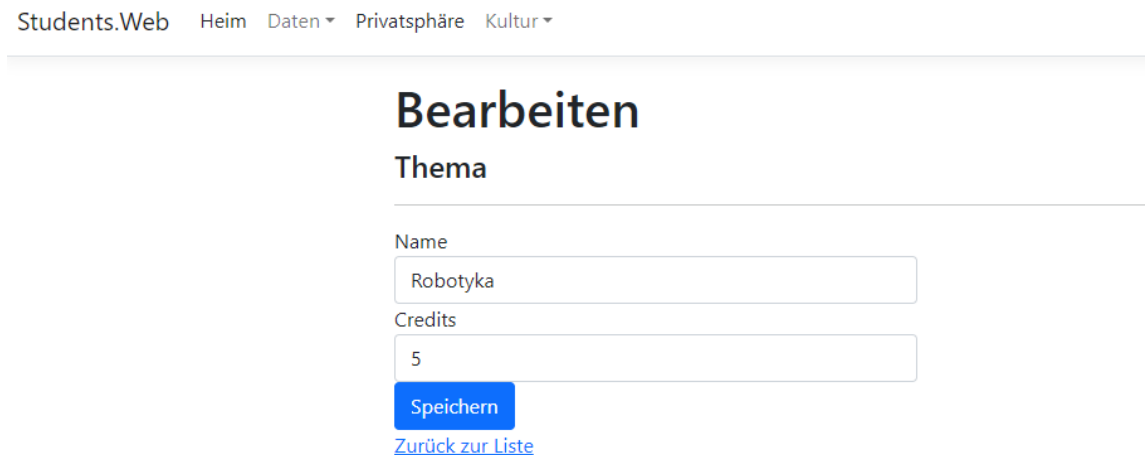
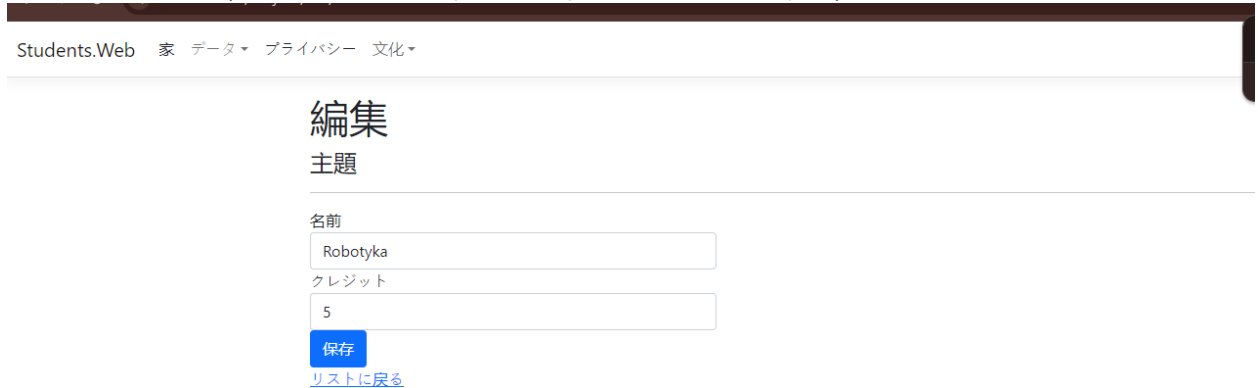
## Podzadanie-4 (Dodanie internacjonalizacji JP, DE dla Student)

Name	Value
Age	年
Are you sure	これを削除してもよろしいですか?
Author	著者
Back to List	リストに戻る
BackToList	リストに戻る
Book	本
Books	本
building Web apps with ASP.NET Core	ASP.NET Core を使用した Web アプリの構築
Create	作成する
Create New	新しく作る
Credits	クレジット
Culture	文化
Data	データ
Delete	消去
Details	詳細
Edit	編集
Floor	床
Home	家
Index of Books	書籍の索引
Index of lecture rooms	講義室インデックス
Index of Lecturers	講師一覧
Index of students	学生のインデックス
Index of subjects	科目の索引
Learn about	について学ぶ
Lecture Room	講義室
Lecture Rooms	講義室
Lecturer	講師
Lecturers	講師
Major	選考科目
Name	名前

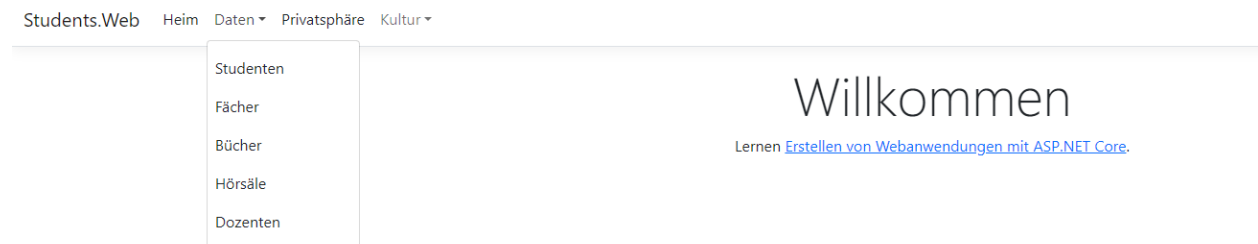
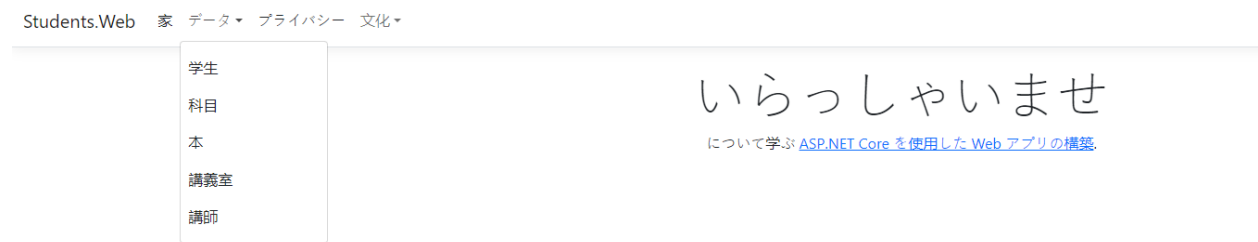
Name	Value	Comment
Age	Alter	
Are you sure	Sind Sie sicher, dass Sie dies löschen möchten?	
Author	Autor	
Back to List	Zurück zur Liste	
BackToList	Zurück zur Liste	
Book	Buch	
Books	Bücher	
building Web apps with ASP.NET Core	Erstellen von Webanwendungen mit ASP.NET Core	
Create	Erstellen	
Create New	Erstelle neu	
Credits	Credits	
Culture	Kultur	
Data	Daten	
Delete	Löschen	
Details	Einzelheiten	
Edit	Bearbeiten	
Floor	Boden	
Home	Heim	
Index of Books	Verzeichnis der Bücher	
Index of lecture rooms	Verzeichnis der Hörsäle	
Index of Lecturers	Dozentenverzeichnis	
Index of students	Index der Studierenden	
Index of subjects	Themenverzeichnis	
Learn about	Lernen	
Lecture Room	Hörsaal	
Lecture Rooms	Hörsäle	
Lecturer	Dozent	
Lecturers	Dozenten	
Major	Wesentlich	
Name	Name	



## Podzadanie-5 (Dodanie internacjonalizacji JP, DE dla Subject)



## Podzadanie-6 (Całkowita internacionalizacja JP, DE dla całego GUI)



# Zadanie 2

**Podzadanie-1** (Przeniesienie kodu z SubjectsController do DatabaseService)

Pokażę ci na przykładzie SubjectCreate.

```
183
184 #region Subject
185
186 public async Task<List<Subject>> SubjectList()...
191
192 public async Task<Subject?> SubjectDetailsDelete(int? id)...
197
198 public async Task<Subject?> SubjectCreate(Subject subject)
199 {
200     _context.Add(subject);
201     await _context.SaveChangesAsync();
202     return subject;
203 }
204
205 public async Task<Subject?> SubjectEditView(int? id)...
210
211 public async Task<Subject?> SubjectEdit(Subject subject)...
217
218 public async Task<bool> SubjectDeleteConfirmed(int id)...
230
231 #endregion Subject
232
```

```
0
1
2 // POST: Subjects/Create
3 // To protect from overposting attacks, enable the specific properties you want to bind to.
4 // For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
5 [HttpPost]
6 [ValidateAntiForgeryToken]
7 public async Task<IActionResult> Create([Bind("Id,Name,Credits")] Subject subject)
8 {
9     if (ModelState.IsValid)
10     {
11         await _databaseService.SubjectCreate(subject);
12         return RedirectToAction(nameof(Index));
13     }
14     return View(subject);
15 }
16
```



## Podzadanie-2 (Przeniesienie kodu z StudentController do DatabaseService)

Pokażę ci na przykładzie StudentCreate.

```
private readonly StudentsContext _context;
private readonly ILogger<DatabaseService> _logger;

public DatabaseService(
    ILogger<DatabaseService> logger,
    StudentsContext context)...

#endregion // Ctor and Properties

#region Students

public async Task<Student?> EditStudentView(int? id)...
public async Task<Student> StudentEdit(Student student, int[] subjectIdDst)...
public Student? DisplayStudent(int? id)...
public async Task<List<Student>> StudentList()...
public async Task<Student?> StudentDetails(int? id)...
public async Task<Student> StudentCreate(Student student, int[] subjectIdDst)...
public async Task<Student?> DeleteStudentView(int? id)...
public async Task<Student> StudentCreateView()...
public async Task<bool> StudentDeleteConfirmed(int id)...

#endregion //Students

Subject
Book
Lecturer
LectureRooms
}
```

```
131 public async Task<Student> StudentCreate(Student student, int[] subjectIdDst)
132 {
133     var chosenSubjects = _context.Subject
134         .Where(s => subjectIdDst.Contains(s.Id))
135         .ToList();
136     var availableSubjects = _context.Subject
137         .Where(s => !subjectIdDst.Contains(s.Id))
138         .ToList();
139     student.AvailableSubjects = availableSubjects;
140
141     foreach (var chosenSubject in chosenSubjects)
142     {
143         student.AddSubject(chosenSubject);
144     }
145     _context.Add(student);
146     await _context.SaveChangesAsync();
147     return student;
148 }
149
150
```

```

96
97 // POST: Students/Create
98 // To protect from overposting attacks, enable the specific properties you want to bind to.
99 // For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
100 [HttpPost]
101 [ValidateAntiForgeryToken]
102 public async Task<IActionResult> Create([Bind("Id, Name, Age, Major, PostalCode ")] Student student, int[] subjectIdDst)
103 {
104     IActionResult result = await Create();
105     if (ModelState.IsValid)
106     {
107         student = await _databaseService.StudentCreate(student, subjectIdDst);
108         result = RedirectToAction(nameof(Index));
109     }
110     return result;
111 }
112
113 // GET: Students/Edit/5

```

# Zadanie 3

Do trzeciego zadania dodałem nowe obiekty takie jak: Książka, Pracownik naukowy i Sala wykładowa.

Od samego początku będę pokazywać realizację książek

**Podzadanie-1** (Wybrać co najmniej dwie z pięciu encji. Oprogramować je od początku do końca w domyślnym języku EN)

```
using System.ComponentModel.DataAnnotations;

namespace Students.Common.Models;

public class Book
{
    public int Id { get; set; }

    [Required]
    public string Name { get; set; } = string.Empty;

    [NameValidation]
    public string Author { get; set; } = string.Empty;

    public Book()
    {
    }

    public Book(string name, string author)
    {
        Name = name;
        Author = author;
    }
}
```

```

Students.Common
Students.Common.Models.Lecturer
1  using Students.Common.Attributes;
2  using Students.Common.ValidationAttributes;
3  using System.ComponentModel.DataAnnotations;
4
5  namespace Students.Common.Models;
6
7  public class Lecturer
8  {
9
10     public int Id { get; set; }
11
12     [Required]
13     [NameValidation]
14     public string Name { get; set; } = string.Empty;
15
16     [Required]
17     public string Subject { get; set; }
18     public Lecturer()
19     {
20     }
21
22     public Lecturer(string name, string subject)
23     {
24         Name = name;
25         Subject = subject;
26     }
27 }
28

```

## Podzadanie-2 (Internacjonalizacja GUI dla wskazanych z podzadania 1 encji)

Pokażę Ci na przykładzie z książki.

```

4
5  namespace Students.Common.Migrations
6  {
7      /// <inheritdoc />
8      public partial class BookAdd : Migration
9      {
10         /// <inheritdoc />
11         protected override void Up(MigrationBuilder migrationBuilder)
12         {
13             migrationBuilder.CreateTable(
14                 name: "Book",
15                 columns: table => new
16                 {
17                     Id = table.Column<int>(type: "int", nullable: false)
18                         .Annotation("SqlServer:Identity", "1, 1"),
19                     Name = table.Column<string>(type: "nvarchar(max)", nullable: false),
20                     Author = table.Column<string>(type: "nvarchar(max)", nullable: false)
21                 },
22                 constraints: table =>
23                 {
24                     table.PrimaryKey("PK_Book", x => x.Id);
25                 });
26         }
27
28         /// <inheritdoc />
29         protected override void Down(MigrationBuilder migrationBuilder)
30         {
31             migrationBuilder.DropTable(
32                 name: "Book");
33         }
34     }
35 }

```

## Index of Books

[Create New](#)

Name	Author	
Informatyka	Andrii Skoromnyi	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

## 書籍の索引

[新しく作る](#)

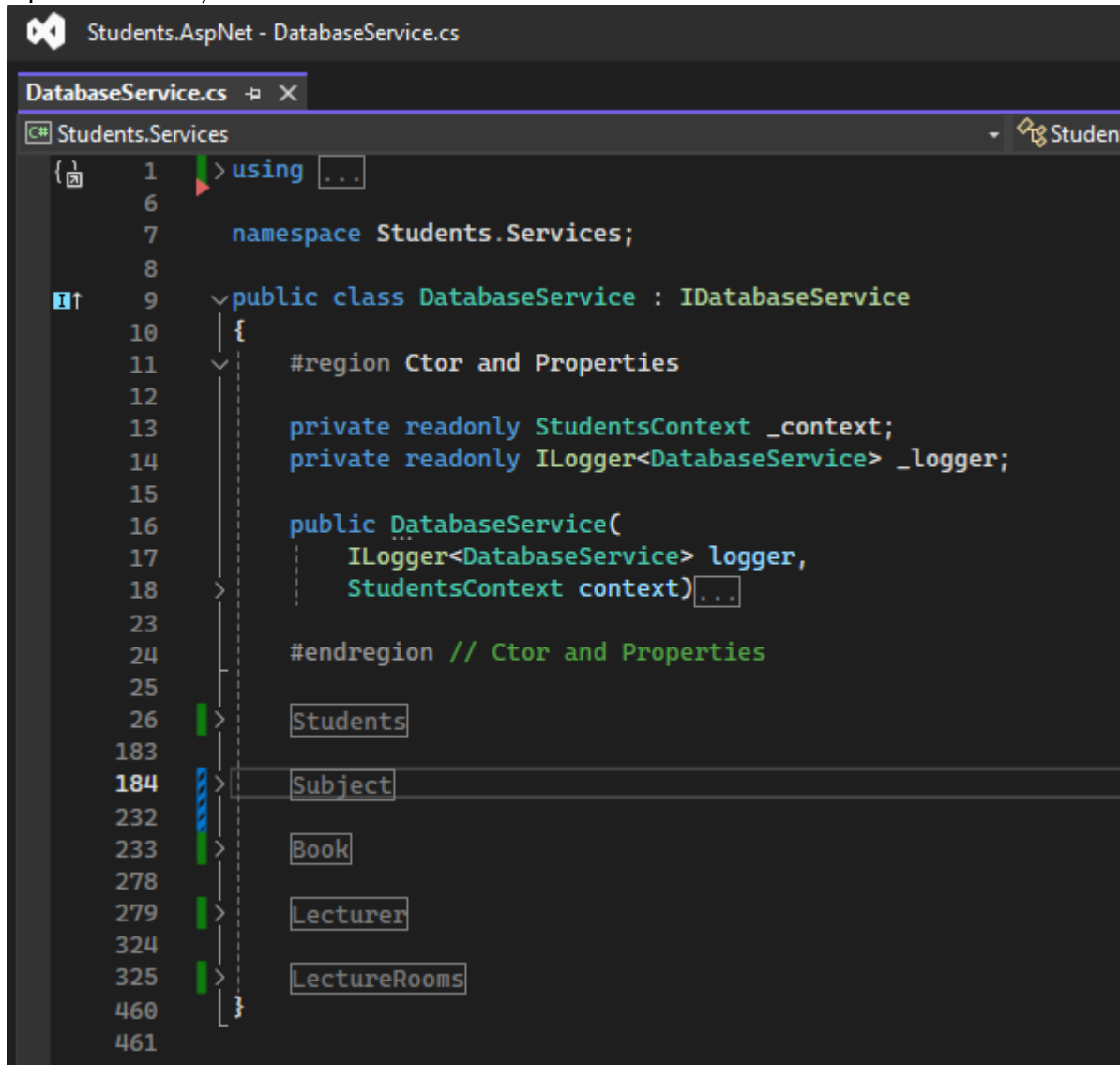
名前	著者	
Informatyka	Andrii Skoromnyi	<a href="#">編集</a>   <a href="#">詳細</a>   <a href="#">消去</a>

## Indeks książek

[Tworzyć nowe](#)

Nazwa	Autor	
Informatyka	Andrii Skoromnyi	<a href="#">Edytować</a>   <a href="#">Detale</a>   <a href="#">Usuwać</a>

**Podzadanie-3** (Przeniesienie kodu bazodanowego z kontrolera do DatabaseService dla encji z podzadania 1.)



```
Students.AspNet - DatabaseService.cs
DatabaseService.cs
C# Students.Services
1 > using ...
6
7 namespace Students.Services;
8
9 public class DatabaseService : IDatabaseService
10 {
11     #region Ctor and Properties
12
13     private readonly StudentsContext _context;
14     private readonly ILogger<DatabaseService> _logger;
15
16     public DatabaseService(
17         ILogger<DatabaseService> logger,
18         StudentsContext context) ...
23
24     #endregion // Ctor and Properties
25
26     > Students
183
184 > Subject
232
233 > Book
278
279 > Lecturer
324
325 > LectureRooms
460
461 }
```

```

#region Book
public async Task<List<Book>> BookList()...

public async Task<Book?> BookDetailsDelete(int? id)...

public async Task<Book?> BookCreate(Book book)
{
    _context.Add(book);
    await _context.SaveChangesAsync();
    return book;
}

public async Task<Book?> BookEditView(int? id)...

public async Task<Book?> BookEdit(Book book)...

public async Task<bool> BookDeleteConfirmed(int id)...
#endregion

```

```

38
39 //Books
40 public Task<List<Book>> BookList();
41
42 public Task<Book?> BookDetailsDelete(int? id);
43
44 public Task<Book?> BookCreate(Book book);
45
46 public Task<Book?> BookEditView(int? id);
47
48 public Task<Book?> BookEdit(Book book);
49
50 public Task<bool> BookDeleteConfirmed(int id);
51
52 //Lecturers
53 public Task<List<Lecturer>> LecturerList();
54
55 public Task<Lecturer?> LecturerDetailsDelete(int? id);
56
57 public Task<Lecturer?> LecturerCreate(Lecturer lecturer);
58
59 public Task<Lecturer?> LecturerEditView(int? id);
60
61 public Task<Lecturer?> LecturerEdit(Lecturer lecturer);
62
63 public Task<bool> LecturerDeleteConfirmed(int id);
64
65 //LectureRooms
66 public Task<List<LectureRoom>> LectureRoomList();
67
68 public Task<LectureRoom?> LectureRoomDetailsDelete(int? id);
69
70 public Task<LectureRoom?> LectureRoomCreate(LectureRoom lectureRoom, int[] subjectIdDst);

```

```

// POST: Books/Create
// To protect from overposting attacks, enable the specific properties you want to bind to.
// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Create([Bind("Id,Name,Author")] Book book)
{
    if (ModelState.IsValid)
    {
        await _databaseService.BookCreate(book);
        return RedirectToAction(nameof(Index));
    }
    return View(book);
}

// GET: Books/Edit/5
public async Task<IActionResult> Edit(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var book = await _databaseService.BookEditView(id);
    if (book == null)
    {
        return NotFound();
    }
    return View(book);
}

// POST: Books/Edit/5
// To protect from overposting attacks, enable the specific properties you want to bind to

```



#### Podzadanie-4 (Dodanie kolejnej encji plus internacjonalizacji GUI i kod bazodanowych w DatabaseService)

```
public async Task<LectureRoom?> LectureRoomCreate(LectureRoom lectureRoom, int[] subjectIdDst)
{
    var chosenSubjects = _context.Subject
        .Where(s => subjectIdDst.Contains(s.Id))
        .ToList();

    var availableSubjects = _context.Subject
        .Where(s => !subjectIdDst.Contains(s.Id))
        .ToList();
    lectureRoom.AvailableSubjects = availableSubjects;

    foreach (var chosenSubject in chosenSubjects)
    {
        lectureRoom.Subjects.Add(chosenSubject);
    }

    _context.Add(lectureRoom);
    await _context.SaveChangesAsync();
    return lectureRoom;
}
```

```
// GET: LectureRooms/Create
public async Task<IActionResult> Create()
{
    IActionResult result = View();
    try
    {
        var newStudent = await _databaseService.LectureRoomCreateView();
        result = View(newStudent);
    }
    catch (Exception ex)
    {
        _logger.LogError("Exception caught: " + ex.Message);
    }

    return result;
}

// POST: LectureRooms/Create
// To protect from overposting attacks, enable the specific properties you want to bind to.
// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Create([Bind("Id,Number,Floor")] LectureRoom lectureRoom, int[] subjectIdDst)
{
    if (ModelState.IsValid)
    {
        await _databaseService.LectureRoomCreate(lectureRoom, subjectIdDst);
        return RedirectToAction(nameof(Index));
    }

    return View(lectureRoom);
}

// GET: LectureRooms/Edit/5
```

```
1  using Students.Common.Attributes;
2  using System.ComponentModel.DataAnnotations;
3  using System.ComponentModel.DataAnnotations.Schema;
4
5  namespace Students.Common.Models;
6
7  public class LectureRoom
8  {
9
10     public int Id { get; set; }
11
12     [Required]
13     public string Number { get; set; } = string.Empty;
14     public int Floor { get; set; }
15
16     [NotMapped]
17     public ICollection<Subject> AvailableSubjects { get; set; } = new List<Subject>();
18     public ICollection<Subject> Subjects { get; set; } = new List<Subject>();
19
20     public LectureRoom()
21     {
22     }
23
24     public LectureRoom(string number, int floor)
25     {
26         Number = number;
27         Floor = floor;
28     }
29 }
30
```

# Zadanie 4

Podzadanie-1 (Wybrać co najmniej jeden atrybut od 1 do 3)

```
[AttributeUsage(AttributeTargets.Property | AttributeTargets.Field | AttributeTargets.Parameter, AllowMultiple = false)]
public class NameValidation : ValidationAttribute
{
    public override bool IsValid(object value)
    {
        if (value == null)
            return true; // Null values are considered valid.

        string name = value as string;
        if (string.IsNullOrEmpty(name))
            return false; // Empty or whitespace-only names are not valid.

        Regex regex = new Regex(@"^[A-Z][a-z]* [A-Z][a-z]*$");

        return regex.IsMatch(name);
    }

    public override string FormatErrorMessage(string name)
    {
        return "Name must start with a capital letter, have only one space, and not contain digits.";
    }
}
```

Name

AndriiSkoromnyi

Name must start with a capital  
letter, have only one space,  
and not contain digits.

## Podzadanie-2 (Wybrać dodatkowy atrybut od 1 do 3)

```
namespace Students.Common.ValidationAttributes
{
    public class PolishPostalCodeAttribute : ValidationAttribute
    {
        public override bool IsValid(object value)
        {
            if (value == null)
                return true; // Null values are considered valid.

            string postalCode = value.ToString();

            Regex regex = new Regex(@"^\d{2}-\d{3}$");

            return regex.IsMatch(postalCode);
        }
    }
}
```

Postal Code

77

The field PostalCode is invalid.

## Podzadanie-3 (Wybrać dodatkowy atrybut od 4 do 6)

```
[StringLength(100)]
public string Major { get; set; } = string.Empty;

[Required]
[PolishPostalCode]
public string PostalCode { get; set; } = string.Empty;
```

```
namespace Students.Common.ValidationAttributes
{
    public class PolishPostalCodeAttribute : ValidationAttribute
    {
        public override bool IsValid(object value)
        {
            if (value == null)
                return true; // Null values are considered valid.

            string postalCode = value.ToString();

            Regex regex = new Regex(@"^\d{2}-\d{3}$");

            return regex.IsMatch(postalCode);
        }
    }
}
```

Postal Code

77

The field PostalCode is invalid.

#### Podzadanie-4 (Dodać unit testy dla atrybutu z podzadania 1)

```
using Xunit;
using Students.Common.ValidationAttributes;

namespace Students.Tests
{
    public class NameValidationTests
    {
        [Theory]
        [InlineData("John Doe")]
        [InlineData("Alice Smith")]
        public void IsValid_ValidNames_ReturnsTrue(string name)
        {
            // Arrange
            var attribute = new NameValidation();

            // Act
            bool isValid = attribute.IsValid(name);

            // Assert
            Assert.True(isValid);
        }

        [Theory]
        [InlineData("")]
        [InlineData(" ")]
        [InlineData("john doe")]
        [InlineData("John123")]
        [InlineData("JohnDoe1")]
        [InlineData("John_Doe")]
        [InlineData("John-Doe")]
        [InlineData("John Doe")]
        [InlineData("JohnDoe Smith")]
        [InlineData("John Doe Smith")]
        public void IsValid_InvalidNames_ReturnsFalse(string name)
        {
            // Arrange
            var attribute = new NameValidation();
        }
    }
}
```

### Podzadanie-5 (Dodać unit testy dla atrybutu z podzadania 3)

```
public class PolishPostalCodeAttributeTests
{
    [Fact]
    public void IsValid_NullValue_ReturnsTrue()
    {
        // Arrange
        var attribute = new PolishPostalCodeAttribute();

        // Act
        var isValid = attribute.IsValid(null);

        // Assert
        Assert.True(isValid);
    }

    [Theory]
    [InlineData("12-345")]
    [InlineData("99-999")]
    public void IsValid_ValidPostalCodes_ReturnsTrue(string postalCode)
    {
        // Arrange
        var attribute = new PolishPostalCodeAttribute();

        // Act
        var isValid = attribute.IsValid(postalCode);

        // Assert
        Assert.True(isValid);
    }

    [Theory]
    [InlineData("12345")]
    [InlineData("123-456")]
    [InlineData("AB-123")]
    [InlineData("12-3456")]
    [InlineData("12-3A5")]
    public void IsValid_InvalidPostalCodes_ReturnsFalse(string postalCode)
```

# Zadanie 5

Do tego zadania zdecydowałam się wykorzystać model, którego istotą będzie to, abyś mógł wybrać przedmiot do biura

**Podzadanie-1** (Wykonać migrację bazy danych oraz zmiany w modelach)

```
public ICollection<StudentSubject> StudentSubjects { get; set; } = new List<StudentSubject>();

public LectureRoom? LectureRoom { get; set; }
public Subject()
{
}

[NotMapped]
public ICollection<Subject> AvailableSubjects { get; set; } = new List<Subject>();
public ICollection<Subject> Subjects { get; set; } = new List<Subject>();

public LectureRoom()
{
}
```

**Podzadanie-2** (Wykonać zmiany w odpowiednich GUI.)

```
@Html.DisplayFor(model => model.LectureRoom)
</dd>
<dt class="col-sm-2">
    @sharedResourcesService.GetString("Subjects", cultureInfo)
</dt>
<dd class="col-sm-10">
    <ul>
```

```
</div>

<div class="form-group">
    <partial name="_LectureRoomChoicePartial" model="Model"/>
</div>

<div class="form-group">
    <input type="submit" value="@sharedResourcesService.GetString("Create", cultureInfo)" class="btn btn-primary" />
</div>
```

# Edit

## Lecture Room

Number

17-B

Floor

2

Available subjects

Robotyka

Operations

==>

<==

Chosen subjects

Save

[Back to List](#)

**Podzadanie-4** (Przetestować i uruchomić)

# Details

## Lecture Room

**Number** 17-B

**Floor** 2

**Subjects** • Robotyka

[Edit](#) | [Back to List](#)



# Tabela

Zadanie	Ilość punktów
1.1	5
1.2	5
1.3	15
1.4	5
1.5	5
1.6.	15
2.1	10
2.2	20
3.1	10
3.2	5
3.3	10
3.4	25
4.1	5
4.2	5
4.3	10
4.4	5
4.5	10
5.1	10
5.2	20
5.3	0
5.4	10
Całkowita suma	205