Rozszerzenie walidacji

Problem

```
public class CustomerController(IValidator validator) : ControllerBase
{
    [HttpPost]
    public IActionResult Post([FromBody] Customer customer)
    {
        if (!_validator.Validate(customer))
        {
            return BadRequest("Validation failed.");
        }

        // Tu normalnie byłby kod dodający kontrahenta do bazy danych
        return Created();
    }
}
```

Rozwiązanie

```
public class CompositeValidator : IValidator
{
    private readonly IEnumerable<IValidator> _validators;

    public CompositeValidator(IEnumerable<IValidator> validators)
    {
        _validators = validators;
    }

    public bool Validate(Customer customer)
    {
        foreach (var validator in _validators)
        {
            if (!validator.Validate(customer))
                return false;
        }

        return true;
    }
}
```

- Przykład użycia

```
var validator = new CompositeValidator(new IValidator[]
{
    new TaxNumberValidator(),
    new RegonValidator(),
    new EmailValidator()
});

var controller = new CustomerController(validator);
controller.Validate(new Customer { TaxNumber = "123", Regon = "456", Email =
"test@example.com" });
```

- Testy jednostkowe

```
using Xunit;

public class ValidatorTests
{
    [Fact]
    public void CompositeValidator_AllValidatorsPass_ReturnsTrue()
    {
        // Arrange
        var customer = new Customer
        {
            TaxNumber = "1234567890",
            Regon = "0987654321",
            Email = "john@example.com"
        };

        var validator = new CompositeValidator(new IValidator[]
        {
            new TaxNumberValidator(),
            new RegonValidator(),
            new EmailValidator()
        });

        // Act
        var result = validator.Validate(customer);

        // Assert
        Assert.True(result);
    }
```

```
    [Fact]
    public void CompositeValidator_OneValidatorFails_ReturnsFalse()
    {
        var customer = new Customer
        {
            TaxNumber = "1234567890",
            Regon = "", // Błąd
            Email = "john@example.com"
        };

        var validator = new CompositeValidator(new IValidator[]
        {
            new TaxNumberValidator(),
            new RegonValidator(),
            new EmailValidator()
        });

        var result = validator.Validate(customer);

        Assert.False(result);
    }

    [Fact]
    public void DecoratorValidator_AllChecksPass_ReturnsTrue()
    {
        var customer = new Customer
        {
            TaxNumber = "1234567890",
            Regon = "0987654321",
            Email = "john@example.com"
        };

        IValidator validator = new EmailValidatorDecorator(
            new RegonValidatorDecorator(
                new TaxNumberValidator()
            )
        );

        var result = validator.Validate(customer);

        Assert.True(result);
    }

    [Fact]
    public void DecoratorValidator_InnerFails_ReturnsFalse()
    {
```

```
        var customer = new Customer
        {
            TaxNumber = "", // Błąd
            Regon = "0987654321",
            Email = "john@example.com"
        };

        IValidator validator = new EmailValidatorDecorator(
            new RegonValidatorDecorator(
                new TaxNumberValidator()
            )
        );

        var result = validator.Validate(customer);

        Assert.False(result);
    }
}
```

Równolegle

```
public interface IValidator
{
    Task<bool> ValidateAsync(Customer customer);
}
```

- Tworzymy kompozyt

```
public class CompositeValidator : IValidator
{
    private readonly IEnumerable<IValidator> _validators;

    public CompositeValidator(params IValidator[] validators)
    {
        _validators = validators;
    }

    public async Task<bool> ValidateAsync(Customer customer)
    {
        var tasks = _validators.Select(v => v.ValidateAsync(customer));
        var results = await Task.WhenAll(tasks);

        return results.All(result => result);
```

```
    }
}
```

- Użycie

```
var validator = new CompositeValidator(
    new TaxNumberValidator(),
    new RegonValidator(),
    new EmailValidator()
);
```