Implementacja DDD .NET - Zadanie 5

Bottega IT Minds

1. Zadanie 5 - Agregat i Optimistic Locking

1.1. Wprowadzenie

Aby zapewnić atomową spójność podczas zapisu oraz gwarancję zachowania niezmienników nawet w przypadku problemów związanych ze współbieżnością, w module *Scoring* zostało dodane pojęcie *Agregatu*.

Każdy Agregat musi posiadać *wersję* czyli tzw *concurrency token*.

Bazowa klasa agregatu wygląda następująco:

```
public abstract class AggregateRootBase : Entity, IAggregateRoot
{
    private int _versionId;

    public void IncreaseVersion()
    {
        _versionId++;
    }
}
```

W każdym Agregacie należy zamapować pole wersji oznaczając, że jest to `IsConcurrencyToken()`:

```
public class CustomerScoringEntityTypeConfiguration :
IEntityTypeConfiguration<CustomerScoring>
{
    public void Configure(EntityTypeBuilder<CustomerScoring> builder)
    {
        builder.ToTable("customer_scorings");

        builder.HasKey(x => x.CustomerId);
        builder.Property(x => x.CustomerId).ValueGeneratedNever();


builder.Property("_versionId").HasColumnName("version_id").IsConcurrencyToken();

        builder.OwnsOne<Money>("_loanLimit", property =>
        {
            property.Property(x => x.Amount).HasColumnName("loan_limit_value");
            property.Property(x =>
x.CurrencyCode).HasColumnName("loan_limit_currency");
        });
    }
}
```

Za podnoszenie wersji agregatu odpowiedzialna jest infrastruktura, która bazuje na *Zdarzeniach*

Domenowych:

```
public class UnitOfWork : IUnitOfWork
{
    private readonly DbContext _context;

    public UnitOfWork(
        DbContext context)
    {
        _context = context;
    }

    public async Task<int> CommitAsync(CancellationToken cancellationToken)
    {
        IncreaseChangedAggregateVersions();

        var result = await _context.SaveChangesAsync(cancellationToken);

        return result;
    }

    private void IncreaseChangedAggregateVersions()
    {
        var aggregates = _context.ChangeTracker
            .Entries<AggregateRootBase>()
            .Where(x => x.Entity.DomainEvents.Any())
            .Select(x => x.Entity).ToList();

        foreach (var entity in aggregates)
        {
            entity.IncreaseVersion();
        }
    }
}
```

1.2. Treść Zadania

Na bazie modułu *Scoring* zamień encję `Resource` na Agregat i dodaj w nim obsługę współbieżności.

Spróbuj zasymulować problem współbieżności używając testu integracyjnego (2 komendy wysłane równocześnie).