# 4.1.4 Остале кључне особине

## 4.1.4.1 Миграције

Кључна способност коју нуди Entity Framework Core (EF Core) је инкрементална миграција података за ажурирање шеме базе података уз промене у доменском моделу апликације. EF Core пружа функцију миграција која омогућава систематско развијање шеме базе података у складу с променама направљеним у ентитетским класама у коду.

Миграције су обично *code-first*, што значи да је ентитетски модел у коду сматран извором истине и база података треба да буде ажурирана да би се успоставила сагласност с њим. Када се направе промене у ентитетским класама, као што су додавање или уклањање атрибута, EF може генерисати миграциону скрипту која садржи кораке потребне за ажурирање базе у складу с тим. Свака миграција има Up() методу која примењује промене шеме и Down() методу која их враћа, што омогућава да се, ефикасно, измењена шема базе података, врати на претходно стање.

```
namespace HospitalLibrary.Migrations
{
    public partial class CreatedFloorEntity : Migration
    {
        protected override void Up(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.CreateTable(
                name: "Floors",
                columns: table => new
                {
                    Id = table.Column<int>(type: "int", nullable: false)
                        .Annotation("SqlServer:Identity", "1, 1"),
                    Number = table.Column<int>(type: "int", nullable: false),
                    Purpose = table.Column<string>(type: "nvarchar(max)", nullable: true),
                    DateCreated = table.Column<DateTime>(type: "datetime2", nullable: false),
                    DateUpdated = table.Column<DateTime>(type: "datetime2", nullable: false),
                    Deleted = table.Column<bool>(type: "bit", nullable: false)
                },
                constraints: table =>
                {
                    table.PrimaryKey("PK_Floors", x => x.Id);
                });
        }

        protected override void Down(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.DropTable(
                name: "Floors");
        }
    }
}
```

Миграције садрже само инкременталне промене, а не целокупну шему. EF Core прати до сада примењене миграције преко историјске табеле. Пре него што примени даље

миграције, упоређује тренутни снимак (енг. snapshot) у историји са тренутним моделом да би применио само потребне промене. За ове потребе, Entity Framework садржи посебан фајл под називом *Snapshot* у којем се налази тренутно стање шеме базе података. Миграције могу креирати, брисати или модификовати објекте базе података попут табела, колона, индекса и ограничења да би се шема усагласила.

Командна линија EF Core dotnet ef пружа команде за генерисање, примену и враћање миграција. Миграције омогућавају развијање базе података на контролисан начин уз промене кода, са могућношћу да се тестирају миграције и укључе у систем контроле изворног кода. За постојеће базе података, EF Core такође пружа функционалност за инверзно инжењерство модела и миграција из шеме базе података. У глобалу, миграције омогућавају управљање променама шеме на систематизован, аутоматизован начин у различитим окружењима за релационе базе података као што су SQL Server, PostgreSQL и MySQL.

## 4.1.4.2 Котрола конкурентног присутпа

С обзиром на чињеницу да је база података дељени ресурс у времену, што значи да више корисничких процеса могу да јој приступају истовремено, Entity Framework пружа механизме којима се може контролисати вишекорисnички режим рада. Најчешћи метод је оптимистичка контрола конкуренције путем верзионисања реда у табели, што се постиже увођењем посебне колоне која представља верзију ентитета. Ово повезује временску ознаку или инкрементирајући број верзије реда са сваким записом, који се ажурира при свакој измени. При чувању, EF проверава да ли је верзија промењена од добављања, и уколико је то био случај, баци се DbUpdateConcurrencyException изузетак указујући на застарелу верзију података. Са друге стране, песимистичка контрола конкуренције користи закључавање базе података да би спречила конфликте ажурирања. Закључавање може бити имплицитно спроведено од стране Entity Framework-а или експлицитно наведено за одређене ентитете или упите. Ипак, треба узети у обзир да иако песимистичко закључавање спречава настајање конфликата ажурирања, оно умањује могућ ниво конкурентности при комуникацији са базом података.

```
public class Person
{
    public int PersonId { get; set; }
    public string FirstName { get; set;
}   public string LastName { get; set; }

    [Timestamp]
    public byte[] Version { get; set; }
}
```

Entity Framework представља снажан алат за управљање релационим базама података у .NET апликацијама. Кроз различите могућности као што су миграције података, контрола конкуренције и интегрисане оптимизације упита, EF омогућава развојним тимовима да ефикасно мењају шему базе података и управљају подацима у својим апликацијама. Са правилном имплементацијом и коришћењем најбољих пракси, максимално се могу искористити предности које овај ORM радни оквир нуди.