Persisting Payroll

In this lab, we will modify our payroll system to support the database.

Entity framework

Add these packages to your project via NuGet

- Microsoft.EntityFrameworkCore
- Microsoft.EntityFrameworkCore.SqlServer
- Microsoft.EntityFrameworkCore.Tools
- Microsoft.EntityFrameworkCore.Proxies

Model Updates

Entity framework doesn't do well with interface type properties. Rename *IPayable* to *Payable*

- change from interface to abstract class
- change Pay to public abstract
- add an *Id* property

Company

- add default ctor
- add **[NotMapped]** to Tenure
- change the IEnumerable Resources to ICollection {get;set;}

HumanResource

- remove *Pay()*

Employee

- LocalTaxFunc not mapped
- add default ctor

Contractor & Intern

- add default ctor

PayDbContext

- Create a new class *PayDbContext : DbContext*
- Create a DbSet for each entity (company, employee, contractor, intern)
- Override OnConfiguring
  - add sqlserver
  - use proxies
- Create a database in *Sql Server Object Explorer*
- Copy the connection string into db context
- create migration
- update database

```
public class PayDbContext : DbContext
{
    public PayDbContext()
    { }
    public PayDbContext(DbContextOptions<PayDbContext> opt)
        : base(opt) { }

    public DbSet<Company> Companies => Set<Company>();
    public DbSet<Employee> Employees => Set<Employee>();
    public DbSet<Contractor> Contractors => Set<Contractor>();
    public DbSet<Intern> Interns => Set<Intern>();

    protected override void OnConfiguring(DbContextOptionsBuilder opts)
    {
        if (!opts.IsConfigured)
        {
            opts.UseSqlServer(@"*connection string*")
                .UseLazyLoadingProxies();
        }
        base.OnConfiguring(opts);
    }
}
```

CompanyDbTest

Add a new class for testing the database. See below for a simple test class.

```
public class CompanyDbTest : IDisposable
{
    PayDbContext ctx;
    public CompanyDbTest()
    {
        ctx = new PayDbContext();
        ctx.Database.EnsureDeleted();
        ctx.Database.EnsureCreated();
        Seed(ctx);
    }
    [Fact]
    public void TestGetAllCompanies()
    {
        var c = ctx.Companies.First();
        Assert.Equal(3, c.Resources.Count());
    }

    public void Dispose()
    {
        ctx.Dispose();
    }
    public static void Seed(PayDbContext ctx)
    {
        Company c1 = new Company("Acme", "12-3456");
        Employee e1 = new Employee("Hank", "Hill", 200, DateTime.Today.AddYears(-16
        Contractor ctr = new Contractor("Peggy", "Hill", 50, DateTime.Parse("2005-1
        Intern intern = new Intern("Luann", "Platter", DateTime.Today.AddMonths(-18
        c1.Hire(e1);
        c1.Hire(ctr);
        c1.Hire(intern);
        ctx.Companies.Add(c1);
        ctx.SaveChanges();


    }
}
```