Gestione Esercizi | Diario di lavoro - 04.10.2019

Gabriele Alessi

Canobbio, 04.10.2019

Lavori svolti

Durante questa giornata ho continuato con l'implementazione del progetto completando in contemporanea anche la documentazione riguardo al lavoro sviluppato.
Penso di aver concluso la parte della libreria di classi, quindi tutto ciò che ha a che fare con i dati e il database, ma prima di cominciare con l'App WPF vorrei consultarmi con il docente per verificare quanto svolto finora.

AppDbContext è la classe che praticamente mette insieme i modelli di dati (Models) e le interfacce con il database (Services) per impostare la base di dati configurando le raccolte delle entità e il percorso di memorizzazione SQLite.

```
public class AppDbContext : DbContext
{
    public DbSet<Classe> Classi { get; set; }
    public DbSet<Modulo> Moduli { get; set; }
    public DbSet<Tematica> Tematiche { get; set; }
    public DbSet<Anno> Anno { get; set; }
    public DbSet<Esercizio> Esercizi { get; set; }
    public DbSet<EsercizioProva> EserciziProva { get; set; }
    public DbSet<Prova> Prove { get; set; }

    public AppDbContext() : base()
    {
    }

    public AppDbContext(DbContextOptions<AppDbContext> options) : base(options)
    {
    }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        if (!optionsBuilder.IsConfigured)
        {
            optionsBuilder.UseSqlite("Data Source=D:\\Temp\\GestioneEsercizi.db");
        }
    }
}
```

DbDataRepository è la classe che implementa l'interfaccia IDataRepository e viene usata principalmente come base per i repository degli altri modelli di dati implementando i metodi relativi al database.

```
protected C context;

protected DbDataRepository(C ctx)
{
    context = ctx;
}

public T Get(int id)
{
    return Get().SingleOrDefault(be => be.Id == id);
}

public virtual IQueryable<T> Get()
{
    return context.Set<T>();
}

public virtual T Insert(T entity)
{
    context.Set<T>().Add(entity);
    context.SaveChanges();
    return entity;
}

public virtual void Update(T entity)
{
    context.Entry(entity).State = EntityState.Modified;
    context.SaveChanges();
}

public virtual void Delete(T entity)
{
    context.Set<T>().Remove(entity);
    context.SaveChanges();
}

public virtual IQueryable<T> Where(Expression<Func<T, bool>> predicate)
{
    return context.Set<T>().Where(predicate);
}
```

Riguardo questa classe dovrei informarmi dell'utilità del metodo Where() in quanto guardando i progetti del modulo dell'anno scorso risulta un po' ambiguo.

IEsercizioRepository è un'interfaccia figlia di IDataRepository relativa al modello di dati dell'esercizio.

```
/// <summary>
/// Interfaccia che implementa i metodi relativi al database
/// sul modello di dati dell'esercizio.
/// </summary>
public interface IEsercizioRepository : IDataRepository<Esercizio>
{
}
```

Orario	Lavori svolti
13:15 - 16:30	Implementazione e Documentazione

Problemi riscontrati e soluzioni adottate

Nessun problema riscontrato.

Punto della situazione rispetto alla pianificazione

Abbastanza in anticipo rispetto alla pianificazione.

Programma di massima per la prossima giornata di lavoro

Implementazione, documentazione.