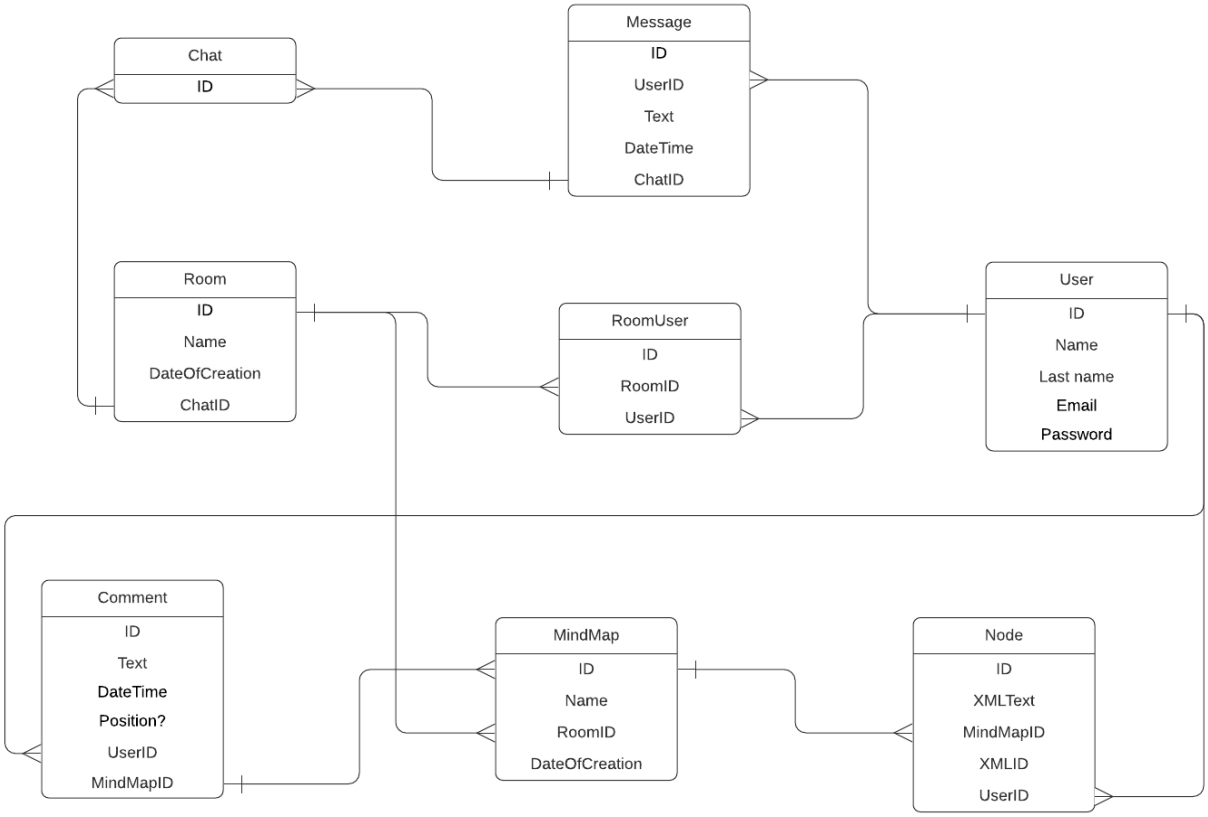
MindMaps Faza 2

Model podataka i model perzistencije

Model podataka i model perzistencije

U projektu MindMaps koristimo relacionu bazu podataka (SQL Server) i ORM alat Entity Framework Core.

Na sledećoj slici je prikazan UML class diagram:



Za pristup bazi implementiran je Repository patern na sledeći način:

1. Repository templejtski interfejs koji ima osnovne CRUD operacije

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using MindMaps.Data.Entities;

namespace MindMaps.Repository
{
    1 reference
    public interface IRepository<T> where T : class, IEntity
    {
        9 references
        Task<List<T>> GetAll();
        17 references
        Task<T> Get(int id);
        9 references
        Task<T> Add(T entity);
        9 references
        Task<T> Update(T entity);
        9 references
        Task<T> Delete(int id);
    }
}
```

2. Abrstraktna templejtska klasa EfCoreRepository koja implementira sve ove operacije u zavisnosti od entiteta

```
public class EfCoreRepository<TEntity> : IRepository<TEntity>
     where TEntity : class, IEntity
{

    private readonly MindMapsContext context;

    8 references
    public EfCoreRepository(MindMapsContext context)
    {
        this.context = context;
    }

    9 references
    public async Task<TEntity> Add(TEntity entity)
    {
        context.Set<TEntity>().Add(entity);
        await context.SaveChangesAsync();
        return entity;
    }
```

```
9 references
public async Task<TEntity> Delete(int id)
{
    var entity = await context.Set<TEntity>().FindAsync(id);
    if (entity == null)
    {
        return entity;
    }

    context.Set<TEntity>().Remove(entity);
    await context.SaveChangesAsync();

    return entity;
}

17 references
public async Task<TEntity> Get(int id)
{
    return await context.Set<TEntity>().FindAsync(id);
}

9 references
public async Task<List<TEntity>> GetAll()
{
    return await context.Set<TEntity>().ToListAsync();
}

9 references
public async Task<TEntity> Update(TEntity entity)
{
    context.Entry(entity).State = EntityState.Modified;
    await context.SaveChangesAsync();
    return entity;
}
```

3. Repozitorijumska klasa za svaki entitet nasleđuje apstraktnu klasu EfCoreRepository i implementira interfejs IRepository

```
using MindMaps.Data.Context;
using MindMaps.Data.Entities;

namespace MindMaps.Repository
{
    4 references
    public class UserRepository : EfCoreRepository<User>
    {
        0 references
        public UserRepository(MindMapsContext context) : base(context)
        {
        }
    }
}
```