**Tới phần code**

**Code từ tầng thấp nhất tới cao nhất DAO đầu tiên > Repos > Services**

1. **Viết account DAO trước copy này quăng vô thay lại cái DAO và Context**

**> đây là singleton**

**public class PremierLeagueAccountDAO**

**{**

**private static PremierLeagueAccountDAO instance = null;**

**private readonly EnglishPremierLeague2024DbContext context;**

**private PremierLeagueAccountDAO()**

**{**

**context = new EnglishPremierLeague2024DbContext();**

**}**

**public static PremierLeagueAccountDAO Instance**

**{**

**get**

**{**

**if (instance == null)**

**{**

**instance = new PremierLeagueAccountDAO();**

**}**

**return instance;**

**}**

**}**

**}**

**Sau đó viết hàm login trong class đó luôn**

**public async Task <PremierLeagueAccount> Login(string email, string password)**

**{**

**var account = await context.PremierLeagueAccounts.FirstOrDefaultAsync(account => account.EmailAddress == email && account.Password == password);**

**return account;**

**}**

**Tạo 1 cái DAO nữa ( của cái bảng có nhiều ) ví dụ player, pet ,…**

**Copy cái singleton qua rồi sửa DAO lại**

**Viết CRUD ở cái DAO đó luôn**

**GET all**

**public async Task<List<FootballPlayer>> GetPlayers()**

**{**

**var players = await context.FootballPlayers.Include(p => p.FootballClub).ToListAsync();**

**return players;**

**}**

**GET one**

**public async Task<Pet> GetPetById(string id)**

**{**

**var pet = await context.Pets.FirstOrDefaultAsync(p => p.PetId == id);**

**return pet;**

**}**

**Add**

**public async Task<FootballPlayer> AddPlayer(FootballPlayer player)**

**{**

**context.FootballPlayers.Add(player);**

**await context.SaveChangesAsync();**

**return player;**

**}**

Riêng update thay đổi **Update**

public async Task<FootballPlayer> UpdatePlayer(FootballPlayer player)

{

var playerToUpdate = await context.FootballPlayers.FirstOrDefaultAsync(p => p.FootballPlayerId==player.FootballPlayerId);

if(playerToUpdate == null)

{

throw new Exception("Player not found");

}

playerToUpdate.FullName = player.FullName;

playerToUpdate.Achievements = player.Achievements;

playerToUpdate.Birthday = player.Birthday;

playerToUpdate.PlayerExperiences = player.PlayerExperiences;

playerToUpdate.Nomination = player.Nomination;

playerToUpdate.FootballClubId = player.FootballClubId;

await context.SaveChangesAsync();

return playerToUpdate;

}

Delete

public async Task<FootballAthlete> DeletePlayer (string id)

{

var playerToDelete = await context.FootballPlayers.FirstOrDefaultAsync(p => p.FootballPlayerId.Equals(id));

if (playerToDelete == null)

{

throw new Exception("Player not found");

}

context.FootballPlayers.Remove(playerToDelete);

await context.SaveChangesAsync();

return playerToDelete;

}

Get all cái bảng khóa ngoại

public async Task<List<FootballClub>> GetFootballClubs()

{

var clubs = await context.FootballClubs.ToListAsync();

return clubs;

}