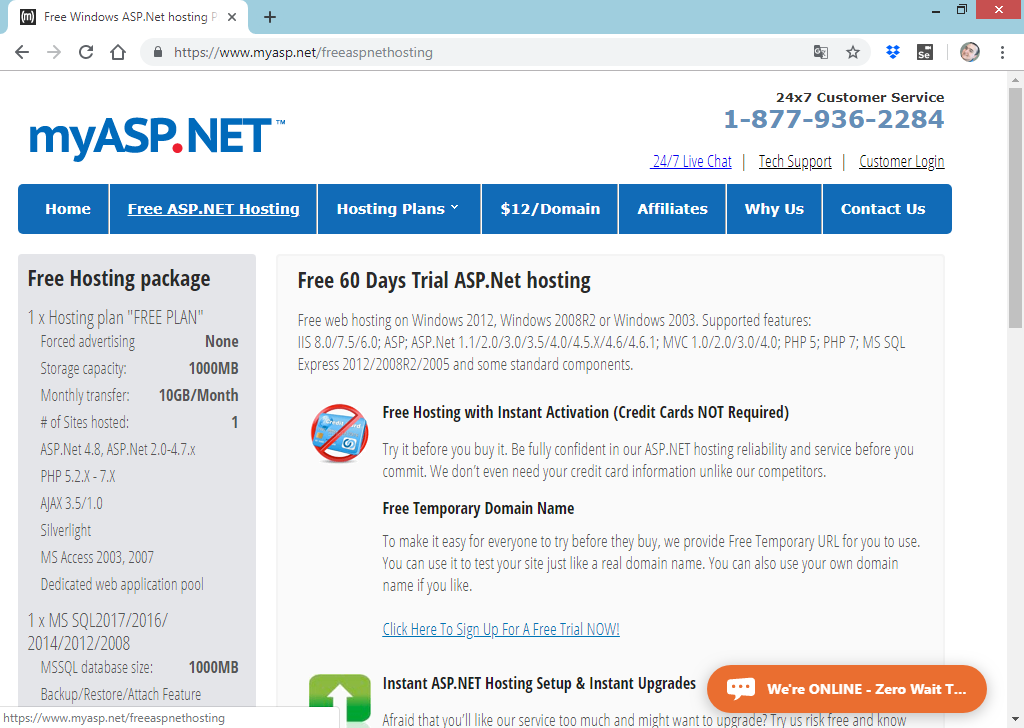
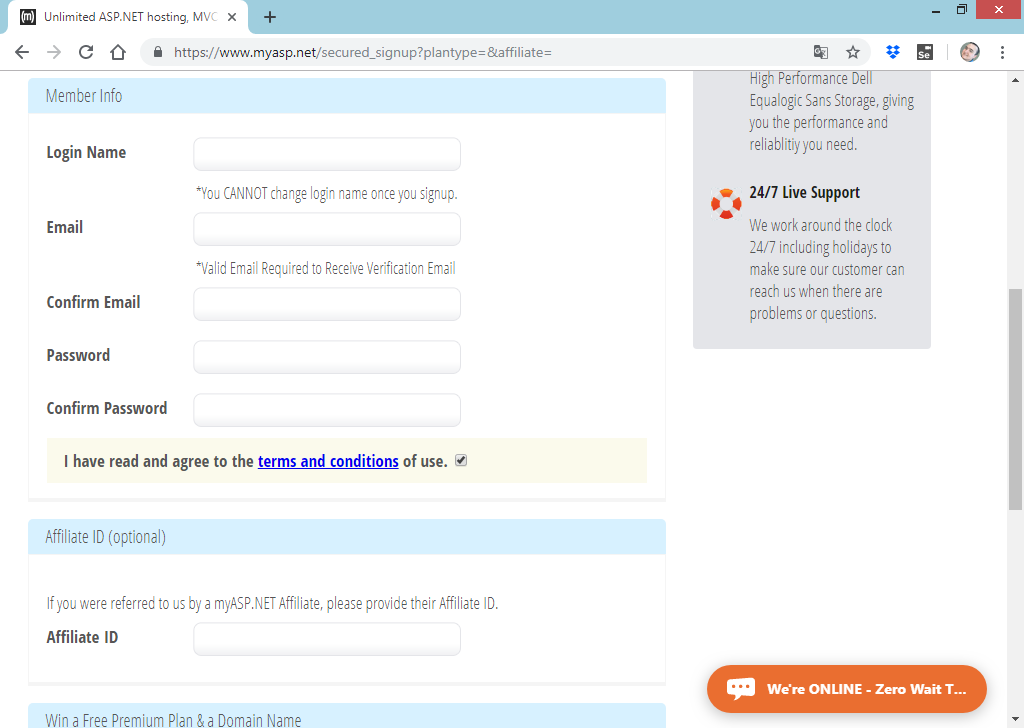
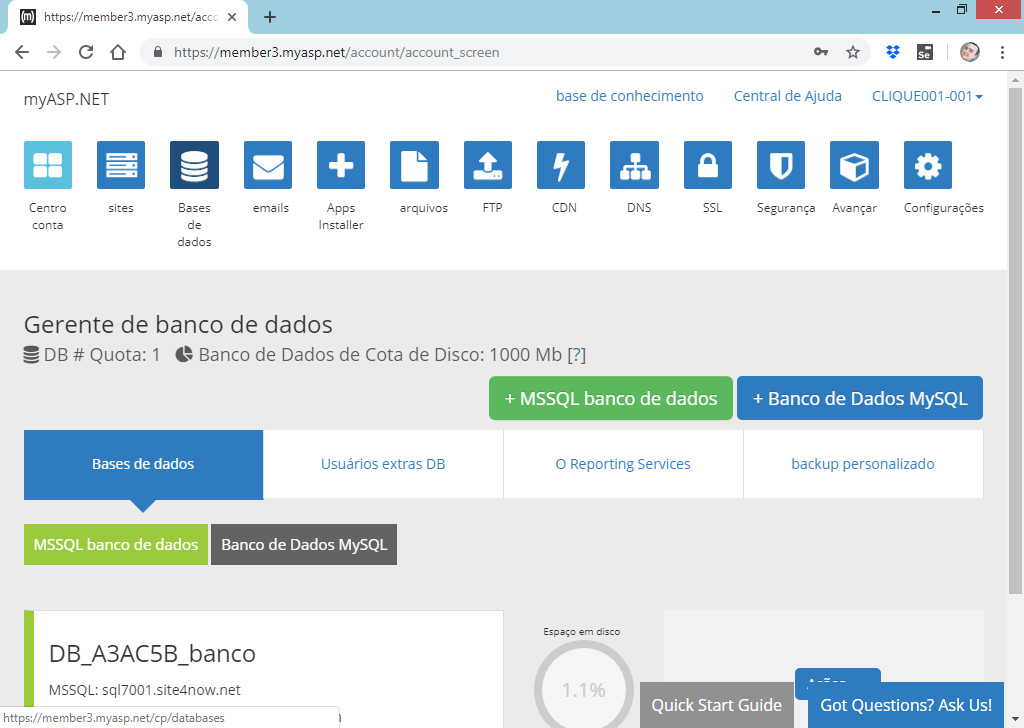
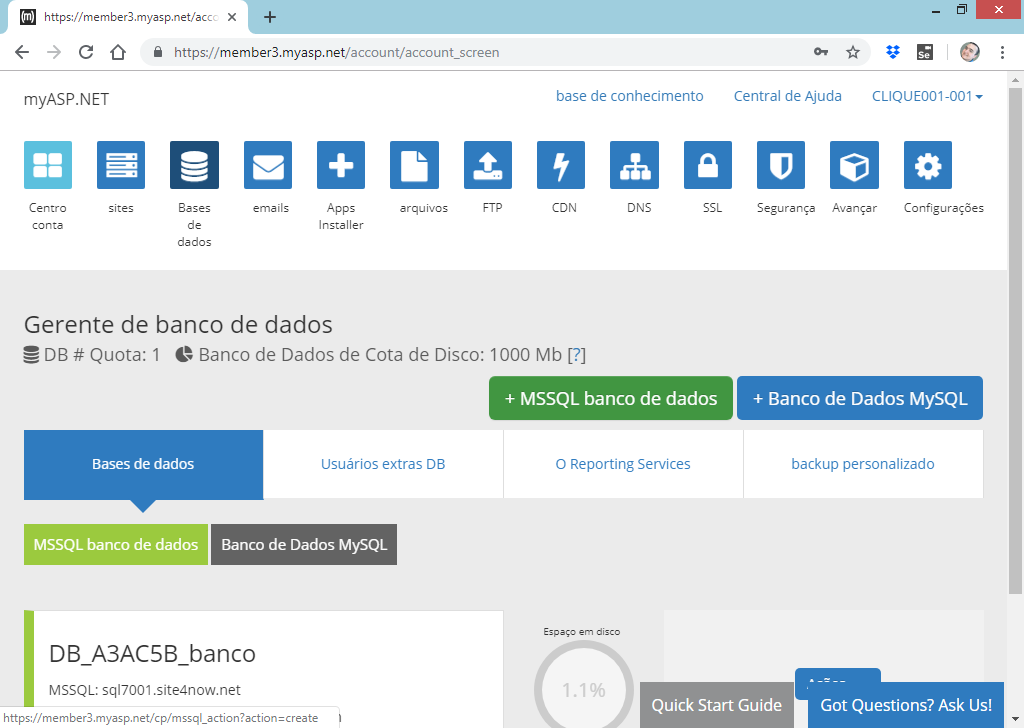
Criando uma conta de hospedagem no site: [**https://www.myasp.net/**](https://www.myasp.net/)

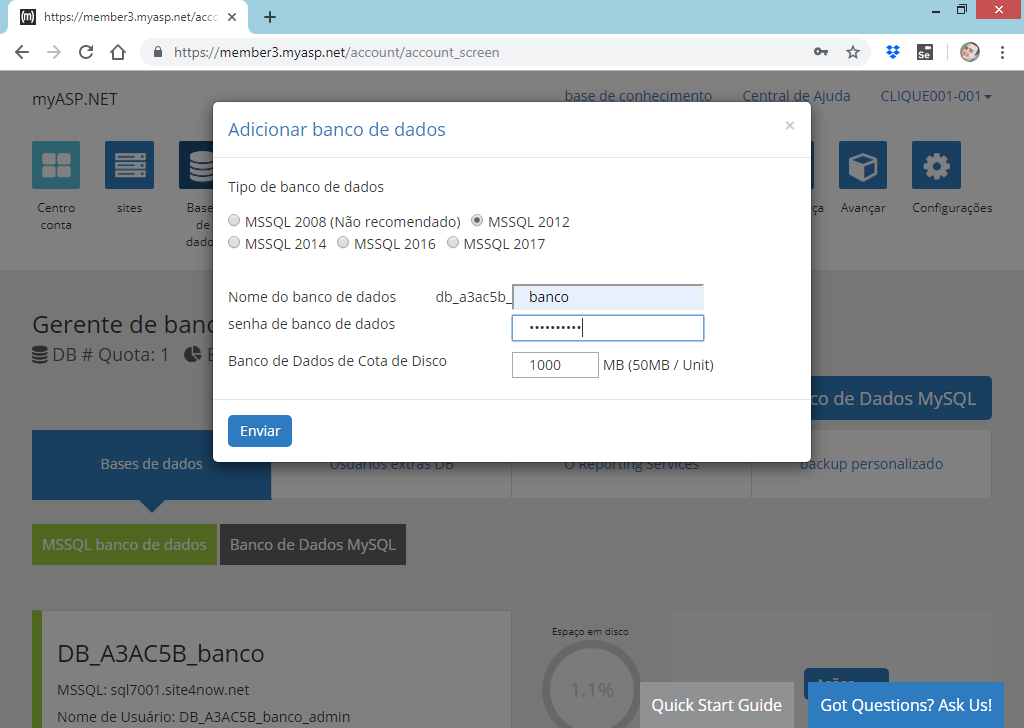




**Criando uma base de dados no site myasp.net**

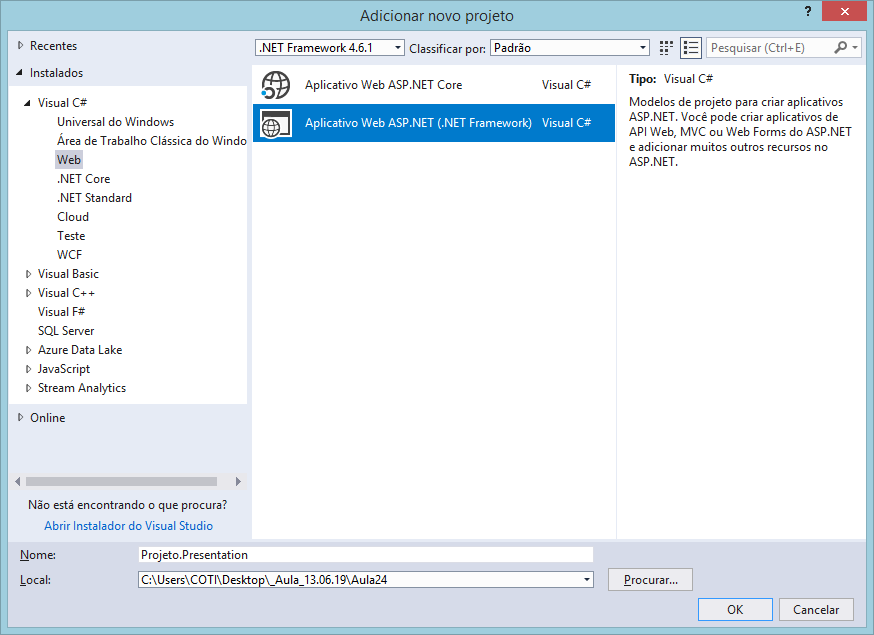


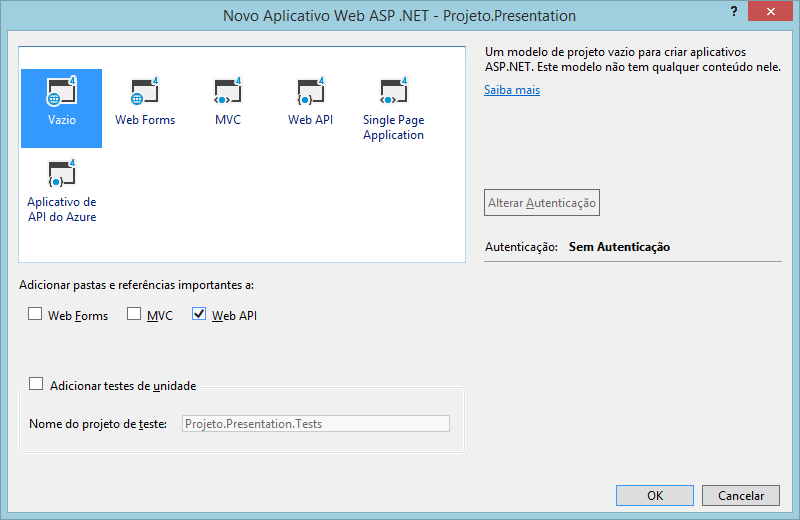




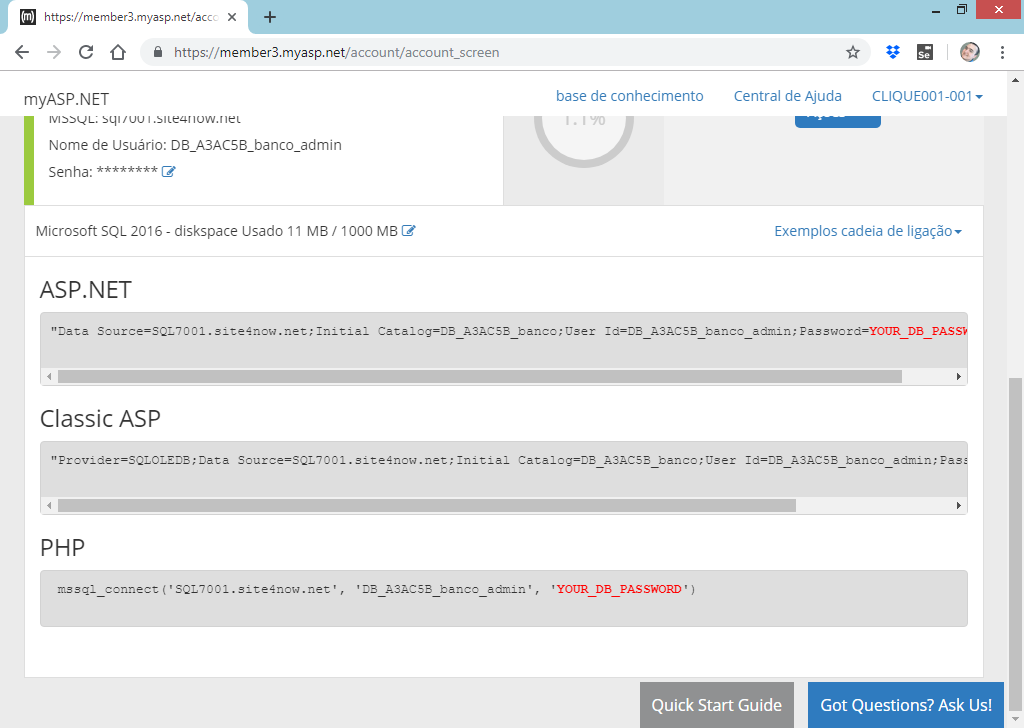
Criando o projeto Asp.Net WebApi

1 - Apresentação





**Obtendo a string de conexão no site myasp.net**



Data Source=SQL7001.site4now.net;Initial Catalog=DB\_A3AC5B\_banco;User Id=DB\_A3AC5B\_banco\_admin;Password=**YOUR\_DB\_PASSWORD**;

\Web.config.xml

**<connectionStrings>**

**<add**

**name="projeto"**

**connectionString="Data Source=SQL7001.site4now.net;**

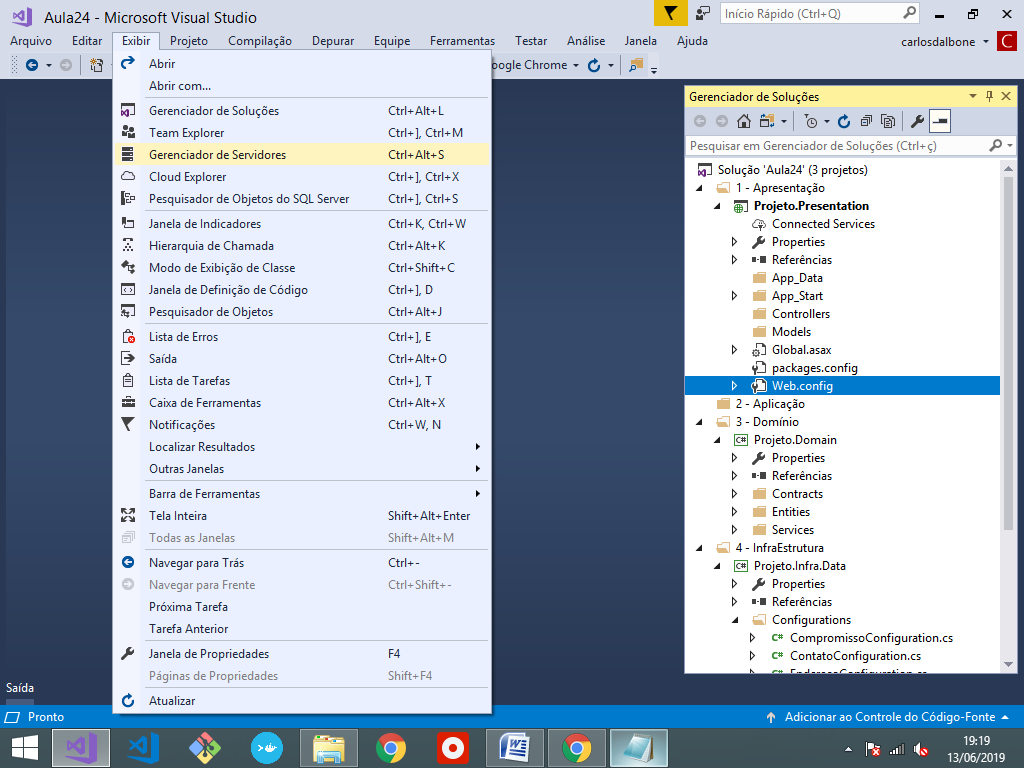
**Initial Catalog=DB\_A3AC5B\_banco;User Id=DB\_A3AC5B\_banco\_admin;**

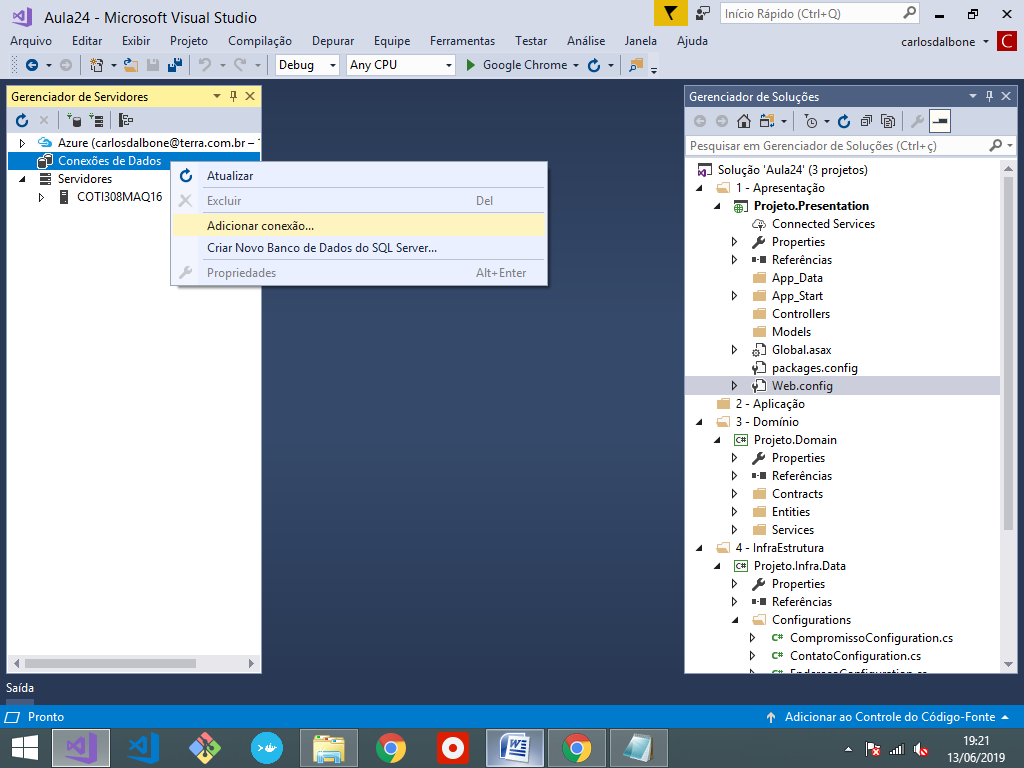
**Password=admin123456;"**

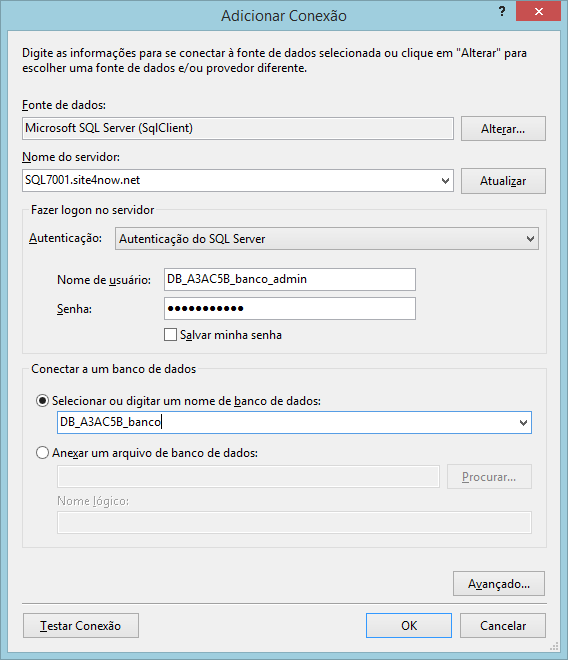
**/>**

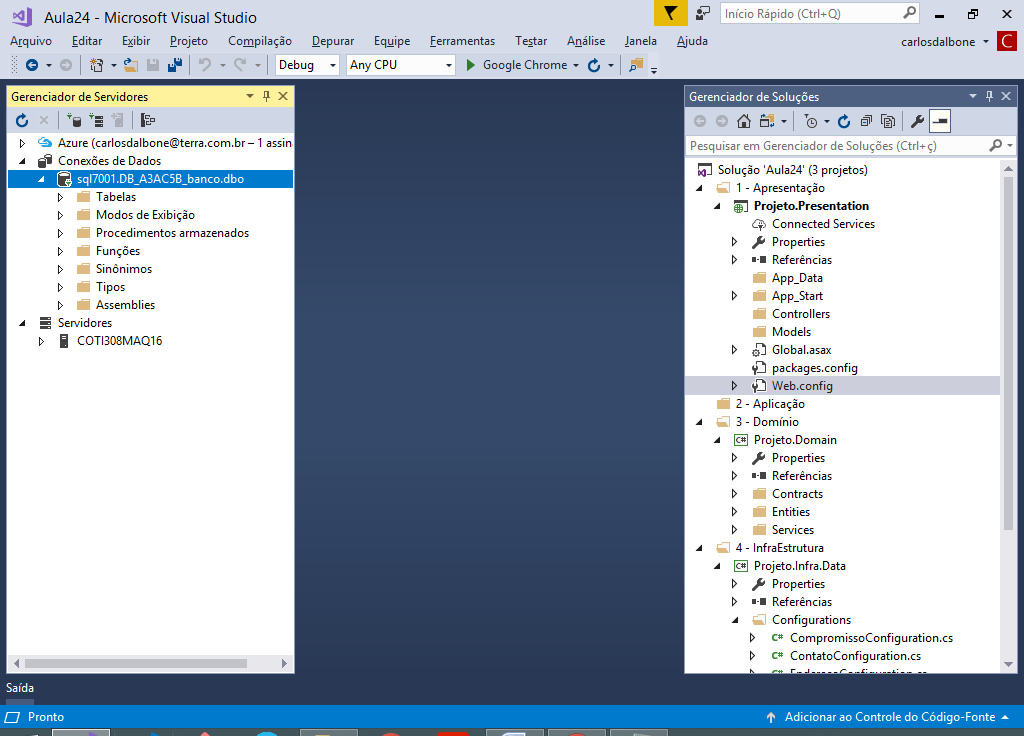
**</connectionStrings>**

Conectando no banco de dados do servidor atraves do VisualStudio









**Criando a classe DataContext**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Configuration; //importando

using System.Data.Entity; //importando

using Projeto.Infra.Data.Configurations; //importando

using Projeto.Domain.Entities; //importando

namespace Projeto.Infra.Data.Context

{

//REGRA 1) herdar DbContext

public class DataContext : DbContext

{

//REGRA 2) construtor que envie para o DbContext

//o caminho da connectionstring

public DataContext()

: base(ConfigurationManager

.ConnectionStrings["projeto"].ConnectionString)

{

}

//REGRA 3) sobrescrever o método OnModelCreating

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

//adicionar cada classe de mapeamento

modelBuilder.Configurations.Add(new ContatoConfiguration());

modelBuilder.Configurations.Add(new CompromissoConfiguration());

modelBuilder.Configurations.Add(new EnderecoConfiguration());

}

//Regra 4) criar uma propriedade DbSet para cada entidade

public DbSet<Contato> Contato { get; set; }

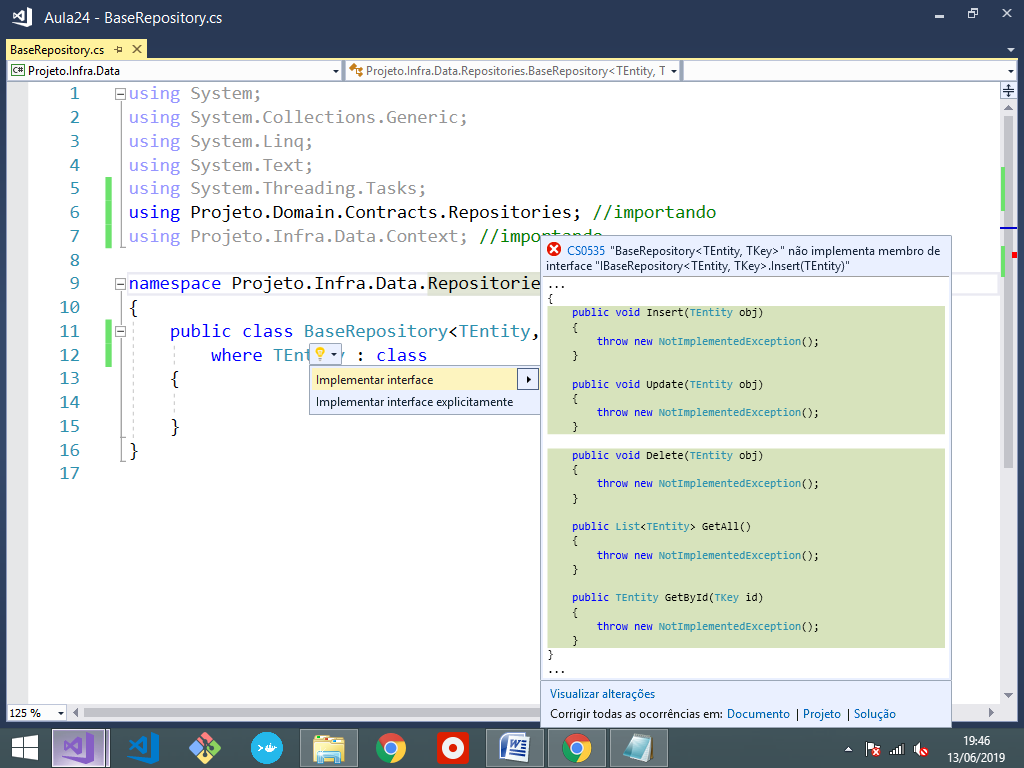
public DbSet<Compromisso> Compromisso { get; set; }

public DbSet<Endereco> Endereco { get; set; }

}

}

**Implementando os contratos do repositório:**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto.Domain.Contracts.Repositories; //importando

using Projeto.Infra.Data.Context; //importando

using System.Data.Entity; //importando

namespace Projeto.Infra.Data.Repositories

{

public class BaseRepository<TEntity, TKey> : IBaseRepository<TEntity, TKey>

where TEntity : class

{

public virtual void Insert(TEntity obj)

{

using (var ctx = new DataContext())

{

ctx.Entry(obj).State = EntityState.Added;

ctx.SaveChanges();

}

}

public virtual void Update(TEntity obj)

{

using (var ctx = new DataContext())

{

ctx.Entry(obj).State = EntityState.Modified;

ctx.SaveChanges();

}

}

public virtual void Delete(TEntity obj)

{

using (var ctx = new DataContext())

{

ctx.Entry(obj).State = EntityState.Deleted;

ctx.SaveChanges();

}

}

public virtual List<TEntity> GetAll()

{

using (var ctx = new DataContext())

{

return ctx.Set<TEntity>().ToList();

}

}

public virtual TEntity GetById(TKey id)

{

using (var ctx = new DataContext())

{

return ctx.Set<TEntity>().Find(id);

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.Entity; //importando

using Projeto.Infra.Data.Context; //importando

using Projeto.Domain.Entities; //importando

using Projeto.Domain.Contracts.Repositories; //importando

namespace Projeto.Infra.Data.Repositories

{

public class CompromissoRepository

: BaseRepository<Compromisso, int>, ICompromissoRepository

{

public List<Compromisso> GetByDatas

(DateTime dataInicio, DateTime dataFim)

{

using (var ctx = new DataContext())

{

return ctx.Compromisso

.Where(c => c.DataHora >= dataInicio

&& c.DataHora <= dataFim)

.ToList();

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.Entity; //importando

using Projeto.Infra.Data.Context; //importando

using Projeto.Domain.Entities; //importando

using Projeto.Domain.Contracts.Repositories; //importando

namespace Projeto.Infra.Data.Repositories

{

public class ContatoRepository

: BaseRepository<Contato, int>, IContatoRepository

{

public Contato GetByEmail(string email)

{

using (var ctx = new DataContext())

{

return ctx.Contato

.FirstOrDefault(c => c.Email.Equals(email));

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.Entity; //importando

using Projeto.Infra.Data.Context; //importando

using Projeto.Domain.Entities; //importando

using Projeto.Domain.Contracts.Repositories; //importando

namespace Projeto.Infra.Data.Repositories

{

public class EnderecoRepository

: BaseRepository<Endereco, int>, IEnderecoRepository

{

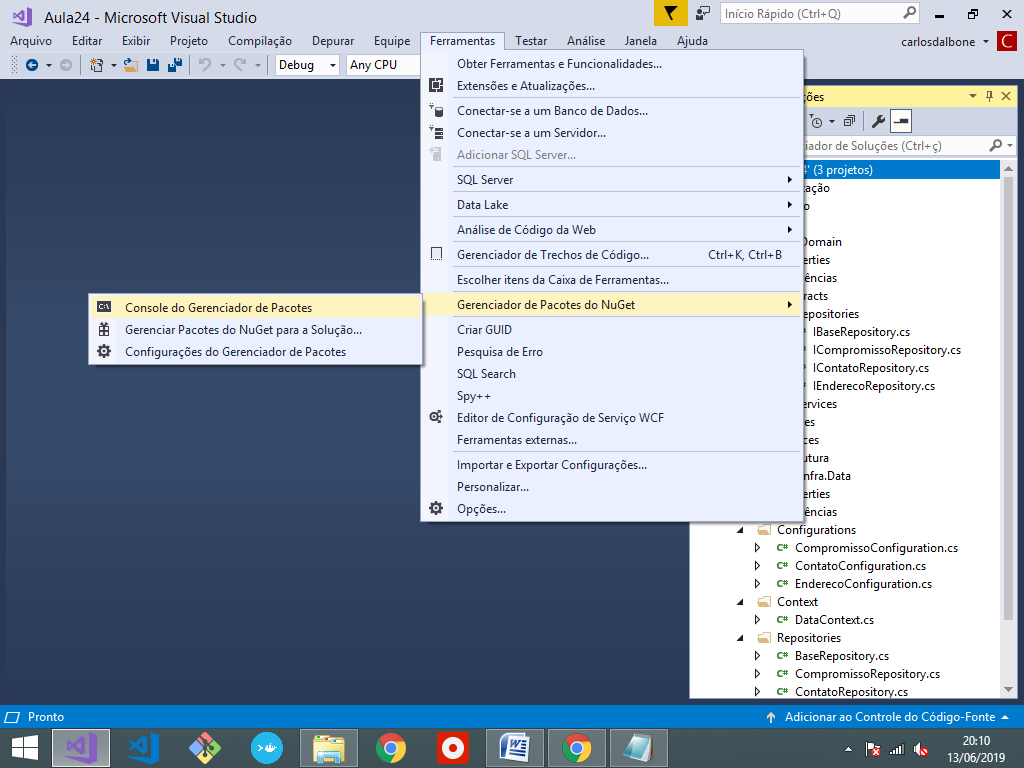
}

}

--------------------------

Migrations

Habilitando o recurso de migrations no projeto



**PM> enable-migrations**

namespace Projeto.Infra.Data.Migrations

{

using System;

using System.Data.Entity;

using System.Data.Entity.Migrations;

using System.Linq;

internal sealed class Configuration : DbMigrationsConfiguration

<Projeto.Infra.Data.Context.DataContext>

{

public Configuration()

{

**AutomaticMigrationsEnabled = true;**

}

protected override void Seed

(Projeto.Infra.Data.Context.DataContext context)

{

// This method will be called after migrating

//to the latest version.

// You can use the DbSet<T>.AddOrUpdate()

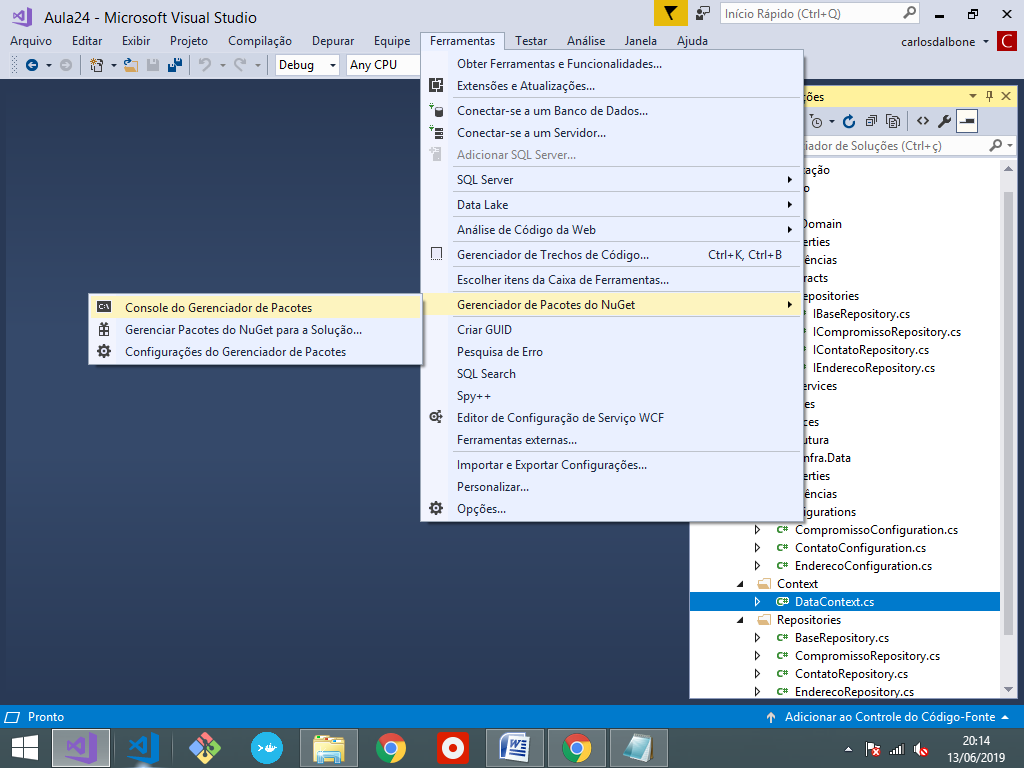
//helper extension method

// to avoid creating duplicate seed data.

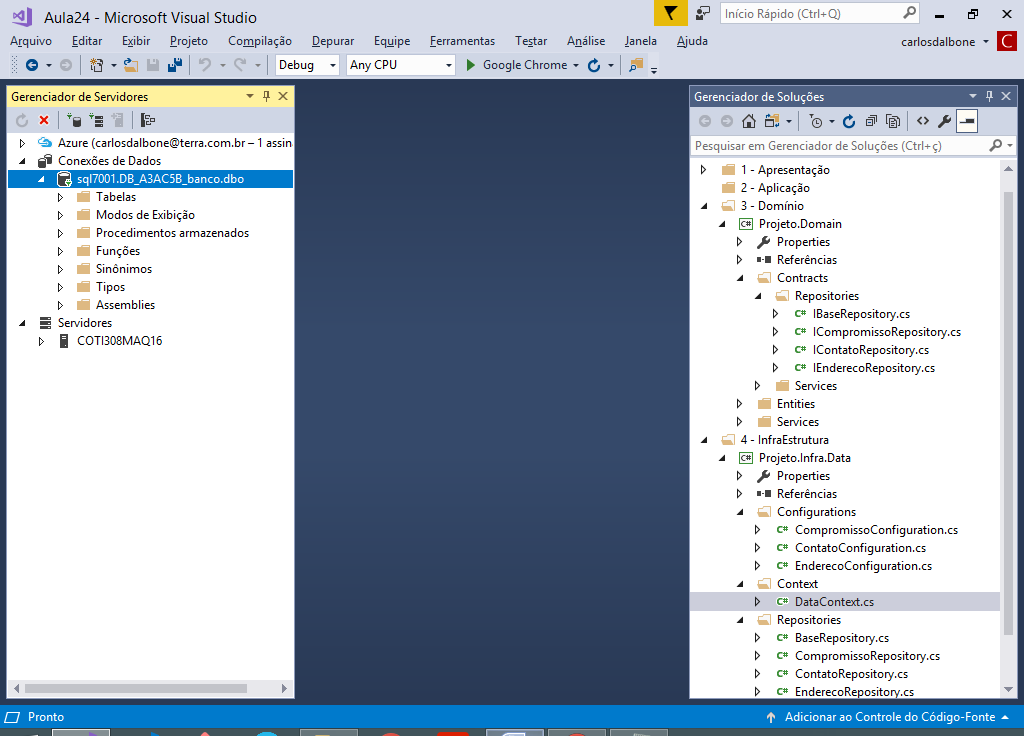
}

}

}



**PM> update-database -verbose**



**Implementando as classes DomainService:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto.Domain.Contracts.Services;

using Projeto.Domain.Contracts.Repositories;

namespace Projeto.Domain.Services

{

public abstract class BaseDomainService<TEntity, TKey>

: IBaseDomainService<TEntity, TKey>

where TEntity : class

{

//atributo

private readonly IBaseRepository<TEntity, TKey> repository;

//construtor para injeção de dependência

public BaseDomainService(IBaseRepository<TEntity, TKey> repository)

{

this.repository = repository;

}

public virtual void Cadastrar(TEntity obj)

{

repository.Insert(obj);

}

public virtual void Atualizar(TEntity obj)

{

repository.Update(obj);

}

public virtual void Excluir(TEntity obj)

{

repository.Delete(obj);

}

public virtual List<TEntity> ConsultarTodos()

{

return repository.GetAll();

}

public virtual TEntity ConsultarPorId(TKey id)

{

return repository.GetById(id);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto.Domain.Entities;

using Projeto.Domain.Contracts.Services;

using Projeto.Domain.Contracts.Repositories;

namespace Projeto.Domain.Services

{

public class CompromissoDomainService

: BaseDomainService<Compromisso, int>, ICompromissoDomainService

{

private readonly ICompromissoRepository repository;

public CompromissoDomainService(ICompromissoRepository repository)

: base(repository)

{

this.repository = repository;

}

public List<Compromisso> ConsultarPorDatas

(DateTime dataInicio, DateTime dataFim)

{

return repository.GetByDatas(dataInicio, dataFim);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto.Domain.Entities;

using Projeto.Domain.Contracts.Services;

using Projeto.Domain.Contracts.Repositories;

namespace Projeto.Domain.Services

{

public class ContatoDomainService

: BaseDomainService<Contato, int>, IContatoDomainService

{

private readonly IContatoRepository repository;

public ContatoDomainService(IContatoRepository repository)

: base(repository)

{

this.repository = repository;

}

public Contato ConsultarPorEmail(string email)

{

return repository.GetByEmail(email);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto.Domain.Entities;

using Projeto.Domain.Contracts.Services;

using Projeto.Domain.Contracts.Repositories;

namespace Projeto.Domain.Services

{

public class EnderecoDomainService

: BaseDomainService<Endereco, int>, IEnderecoDomainService

{

private readonly IEnderecoRepository repository;

public EnderecoDomainService(IEnderecoRepository repository)

: base(repository)

{

this.repository = repository;

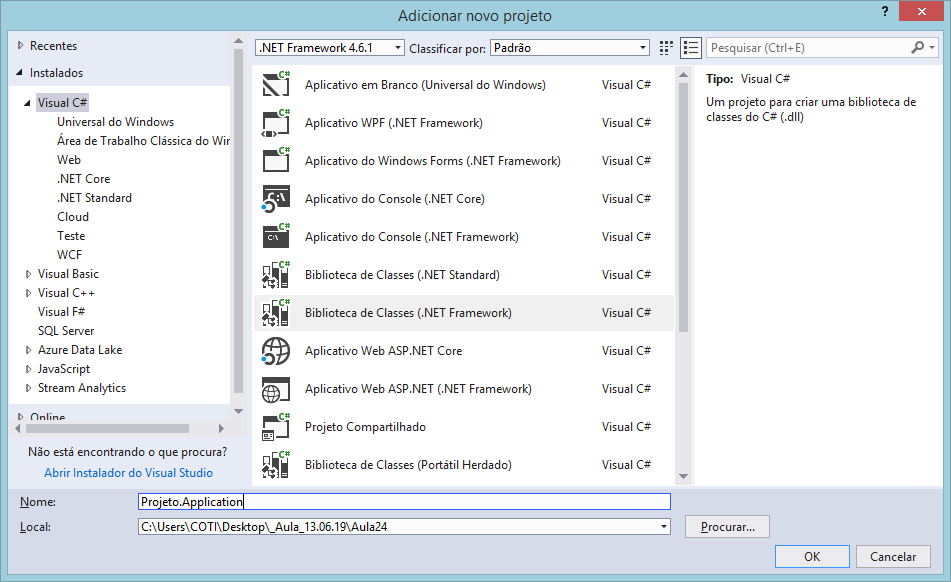
}

}

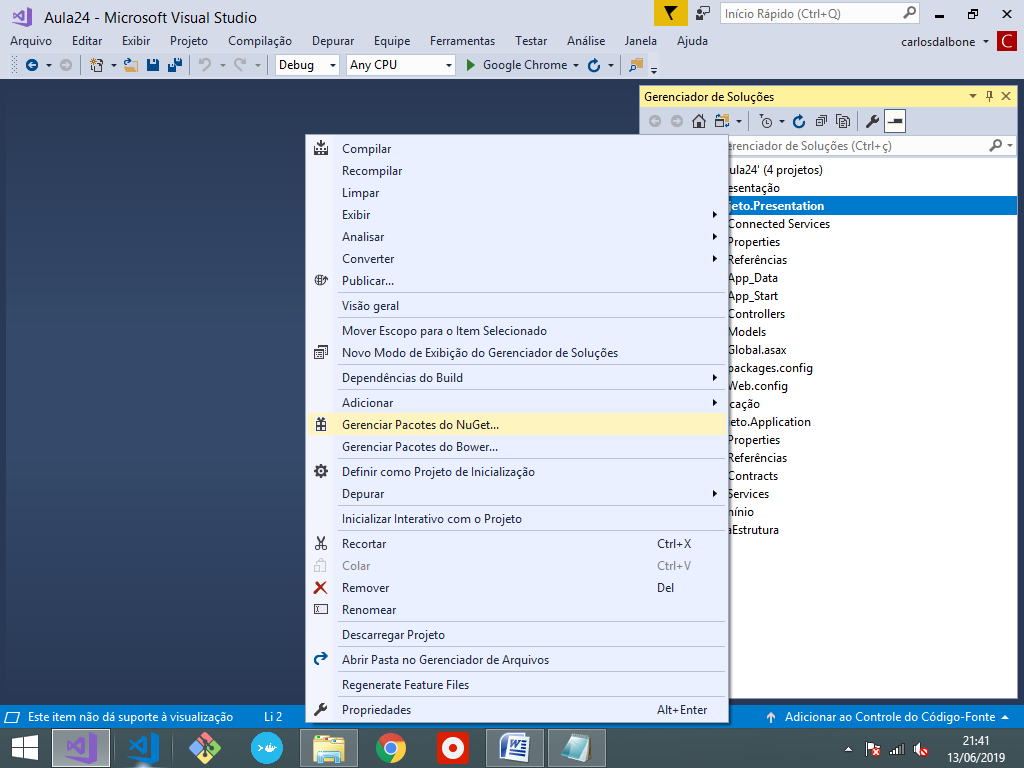
}

-------------------------------------------

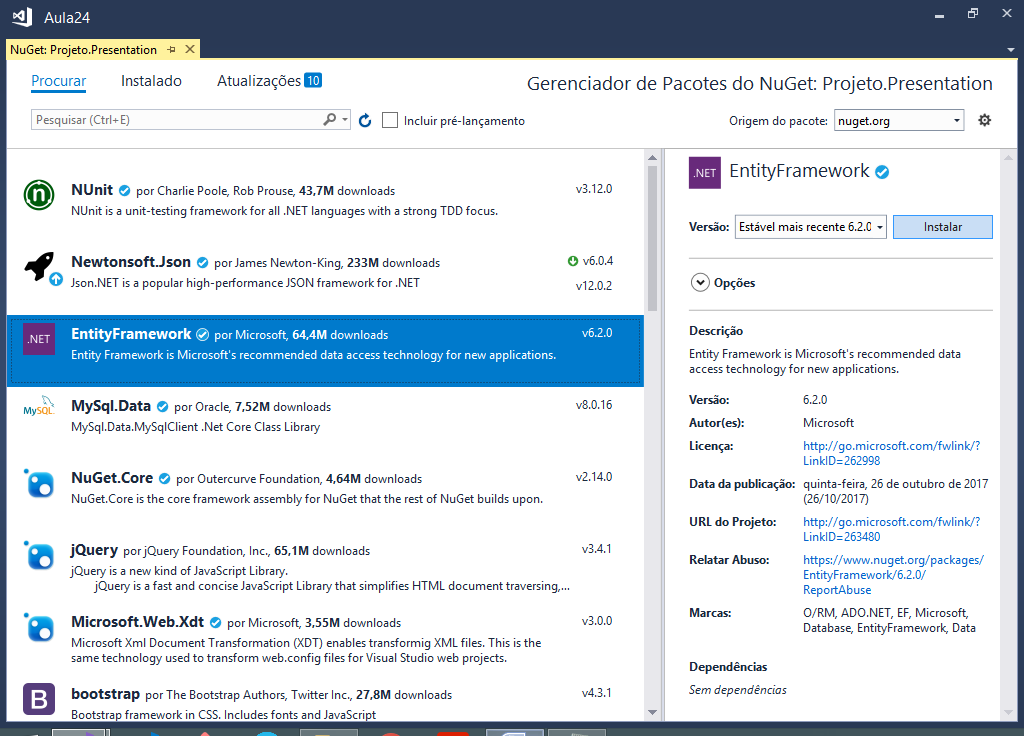
2 - Aplicação



**Instalando os frameworks necessários no projeto Presentation:**



**EntityFramework:**



**Simple Injector para WebApi**

