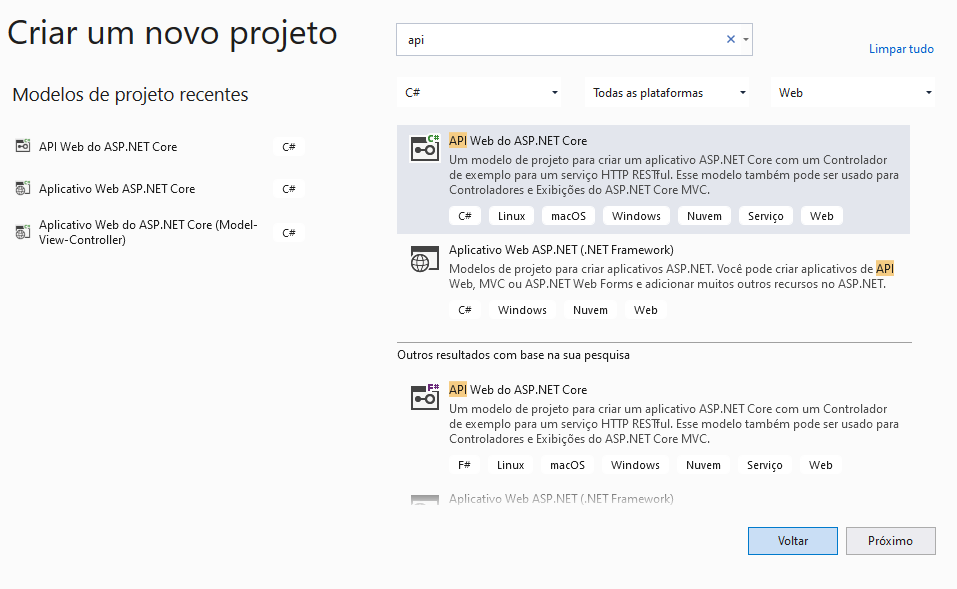
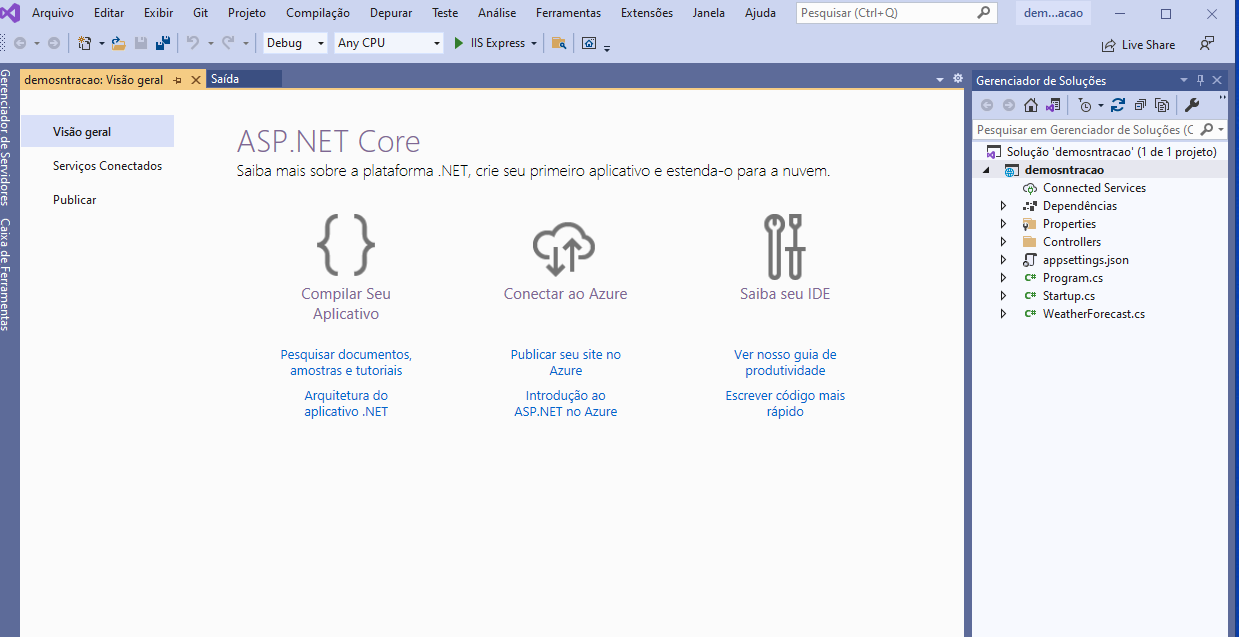
Blog vai voa!! 28/05

Contruindo uma api para criação de números de cartão de créditos.

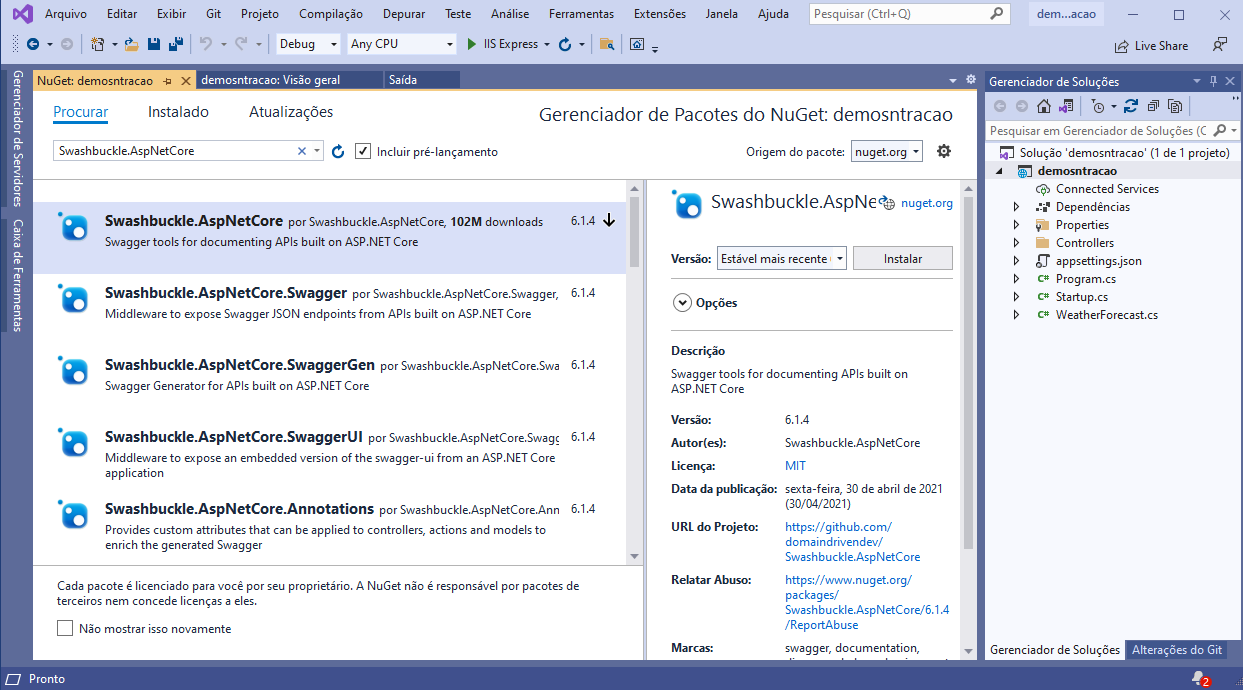
Olá, neste post iremos construir um api que gerará, a partir do seu email, números de cartão de créditos aleatórios. Bem, só de ver aquele monstro de dezesseis algarismos já dá medo e aos mais desavisados um -“deve ser um número qualquer”, mas não, eles tem regras e é o que também iremos aplicar aqui, pra isso vamos conhecer uma pessoa Hans Peter Luhn (https://en.wikipedia.org/wiki/Hans\_Peter\_Luhn) criador do algoritmo de cartões. Com base nesse algoritmo pegaremos um artigo dentre os inúmeros que se habilitam a explicá-lo do Nick Berry (<https://www.datagenetics.com/blog/july42013/index.html>) e usaremos também seu validador. Como ferramenta usaremos a linguagem C# e o Visual Studio Community. Outro detalhe é que faremos uso do protocolo HTTP.

Dando início vamos abrir o VSC (Visual Stuido Code) e criar um novo projeto com o arcabouço Api Web do ASP .NET Core esse projeto já nos fornece alguns arquivos de exemplo e um pequeno proheto montado que iremos aproveitar. Nesse segundo momento vamos baixar as dependências eu serão necessárias para o projeto.

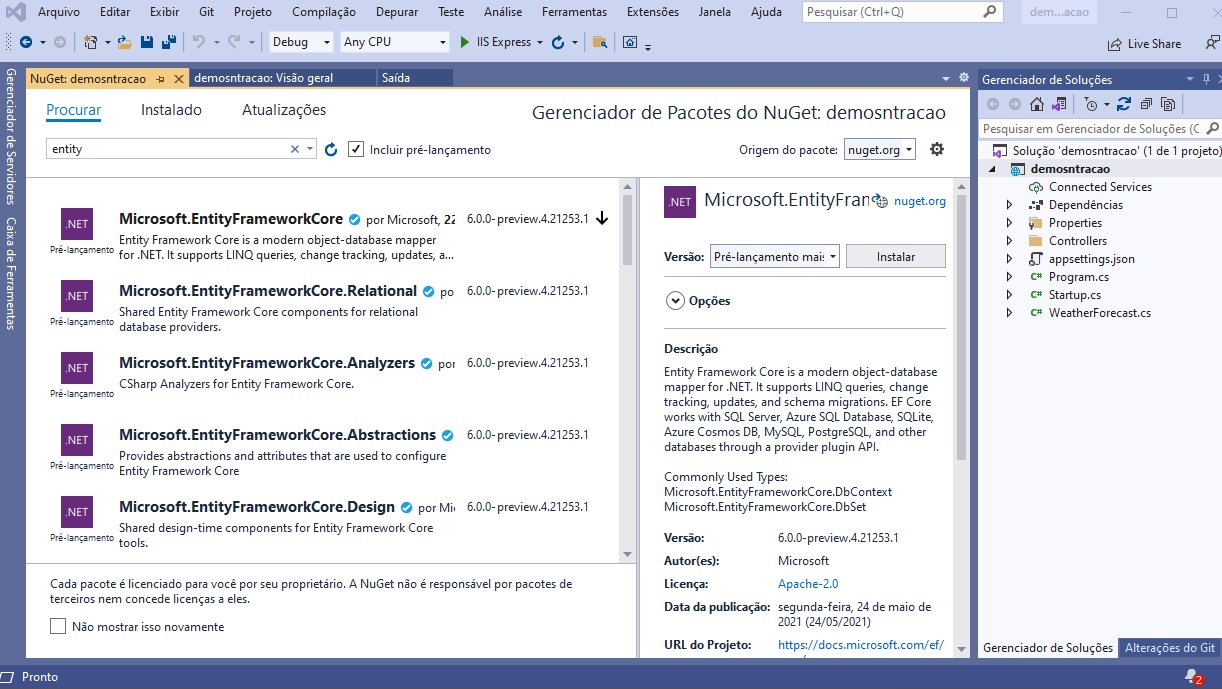




-Swagger (<https://docs.microsoft.com/pt-br/aspnet/core/tutorials/getting-started-with-swashbuckle?view=aspnetcore-5.0&tabs=visual-studio>)



-Entity Framework (<https://docs.microsoft.com/pt-br/ef/core/get-started/overview/first-app?tabs=netcore-cli>)



Um explicação simples sobre o Swagger ele é um framework que tem um bom conjunto de ferramentas nos ajudar no desenvolvimento e documentação da nossa api.

O Entity Framework é um ORM que nos dá um abstração não só das consultas no banco de dados mas também toda relação com ele e com objetos dentro do nosso sistema.

Vamos configurar o Swagger no arquivo no arquivos Startup colocaremos duas configurções na primeira iremos no método configureServies adicionar essas linhas

services.AddSwaggerGen(c => { c.SwaggerDoc("v1", new OpenApiInfo { Title = "Curso Api MVC", Version = "v1" }); });

var xmlFile = $"{Assembly.GetExecutingAssembly().GetName().Name}.xml";

var xmlPath = Path.Combine(AppContext.BaseDirectory, xmlFile);

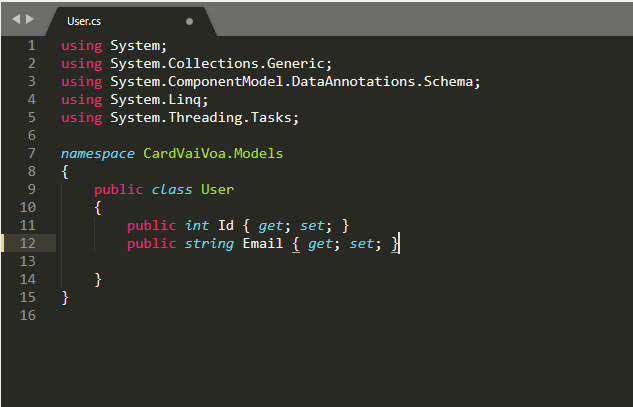
E no seugndo método que é o configure adicionaremos essas linhas:

app.UseSwagger();

app.UseSwaggerUI(c => { c.SwaggerEndpoint("/swagger/v1/swagger.json", "Curso API MVC"); });

Agora vamos dar início ao nosso banco de dados nesse exemplo usaremos o médo code first que consiste em crirar o código primerio , crie uma pasta models no nosso projeto logo em seguida um arquivo que conterá uma classe modelo para o EF (Entity Framework) Crie o nome da classe de

maneria a descrever o que ela faz aqui o primero é o User com Id e Email.

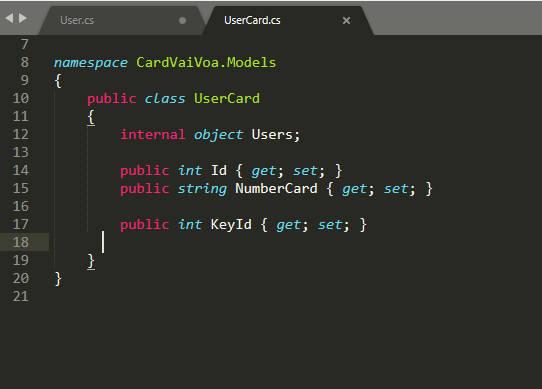
r

Agora vamos criar nosso banco e nossa primeira tabela abra o terminal e digite os comandos

Add-Migration InitialUserCreate cria as migrations

Update-Database cria as tabelas no banco

Repita esses paços para a tabela de cartão dos usuário ela deve ficar assim

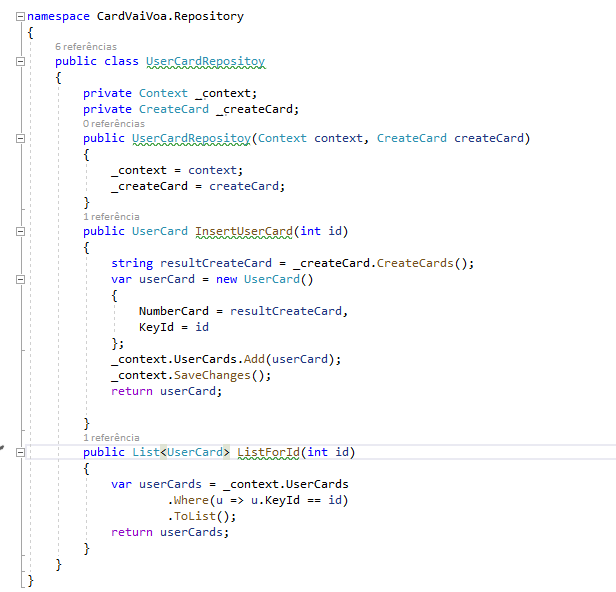


Ok tabelas criadas vamos a nossa controller que é o arquivo q roteará a nossa aplicação. Ela conterá dois métodos HTTP dois do tipo POST que irá inserir e devolverá o usuário com nosso número de cartão e o outro que listará a partir do emai os cartões do usuário e outro GET que apenas listará os usuários.

A controller ficará assim

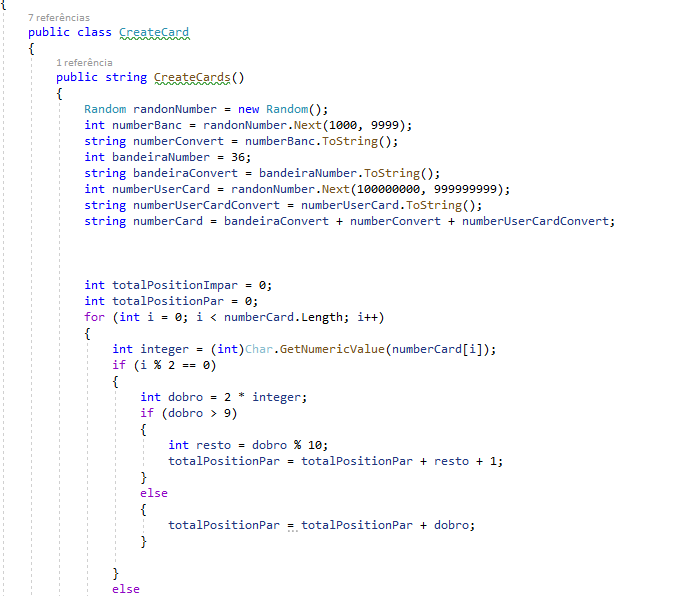


Para não ecessarmos o banco a partir da controller adicionaremos uma cmada que vai acessar o banco: crie uma pasta Repository lá criaremos duas classes repositórios elas ficam assim:





Bem falta o principal a régra de negócio e colocaremos ela na pasta RuleDomain e depois desse paço dois arquios serão criados um com a nossa regra da criação do cartão q ficará assim.

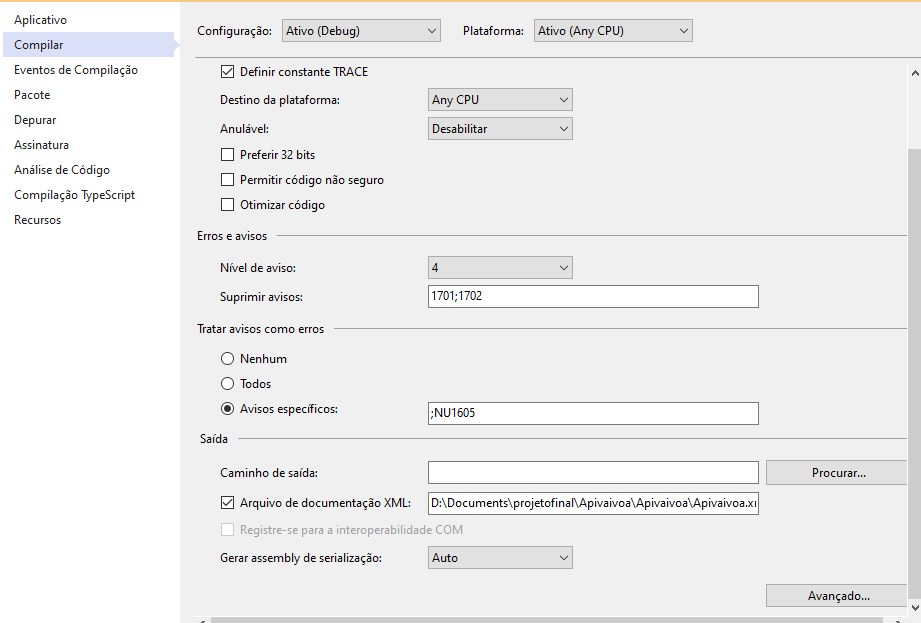


Nessa regra usamos a classe Randon para gerar números e trabalhos bastante com conversão de tipos como string para char de char int.

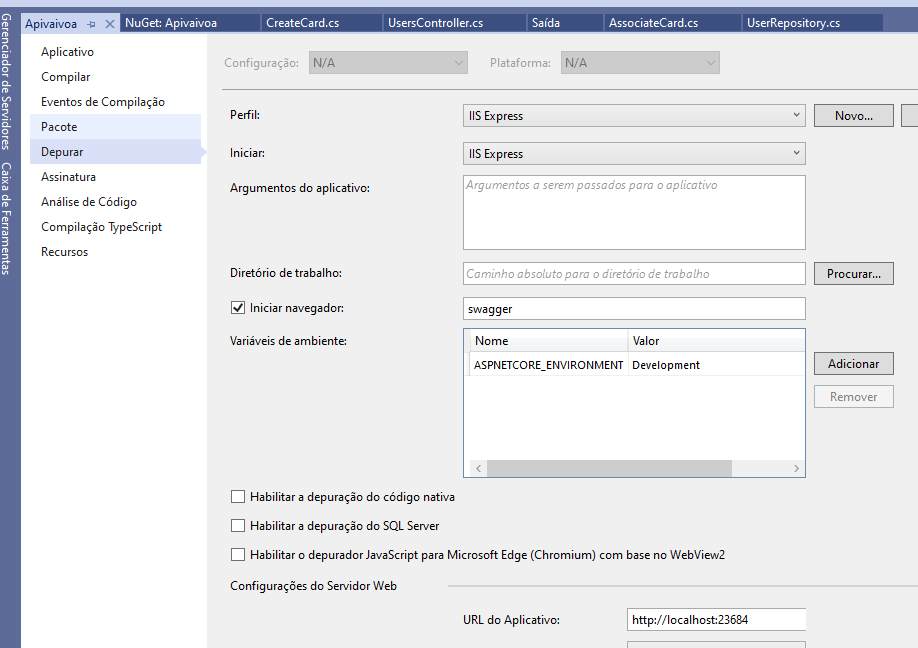
Logo em seguida criaremos outro arquivo onde ficara a classe que associa os o cartão ao usuário.



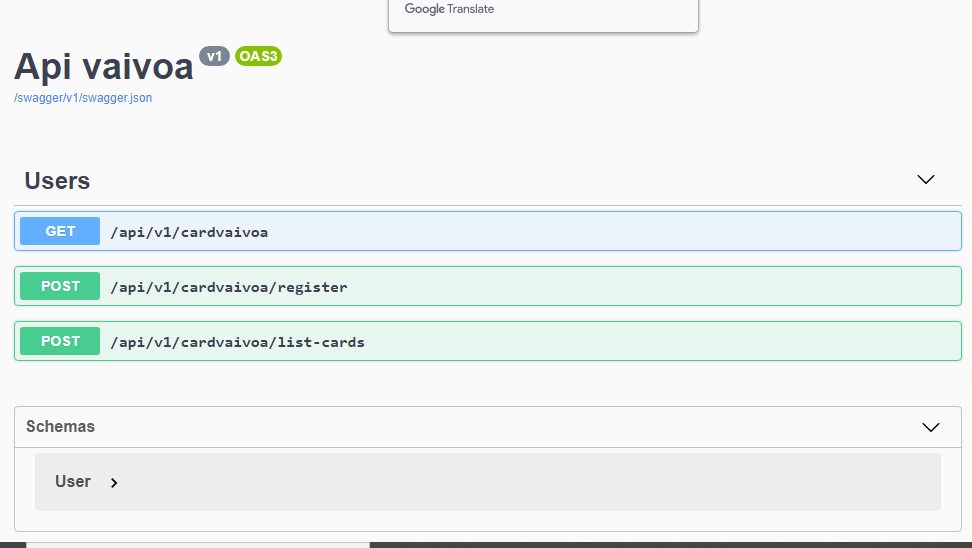
Nessa classe também se trata de uma cmada para não colocarmos a regra de negócios na controller.

A ligação entre todas as camdas está feita só falta uma configurção do swagger para abrirmos à partir dele nossa plicação, click com botão direito no nosso projeto vá em propriedades vai abrir uma caixa de diálogo, click em compilar marque a caixa arquivo de documentação XML.

Agora na mesa caixa de diálogo depurar e onde esta escrito iniciar no navegador escreva swagger.



Pronto agora é só salvar e usar a plicação ela deve abrir uma janela assim.



Está pronta nossa api é só usar os métodos.