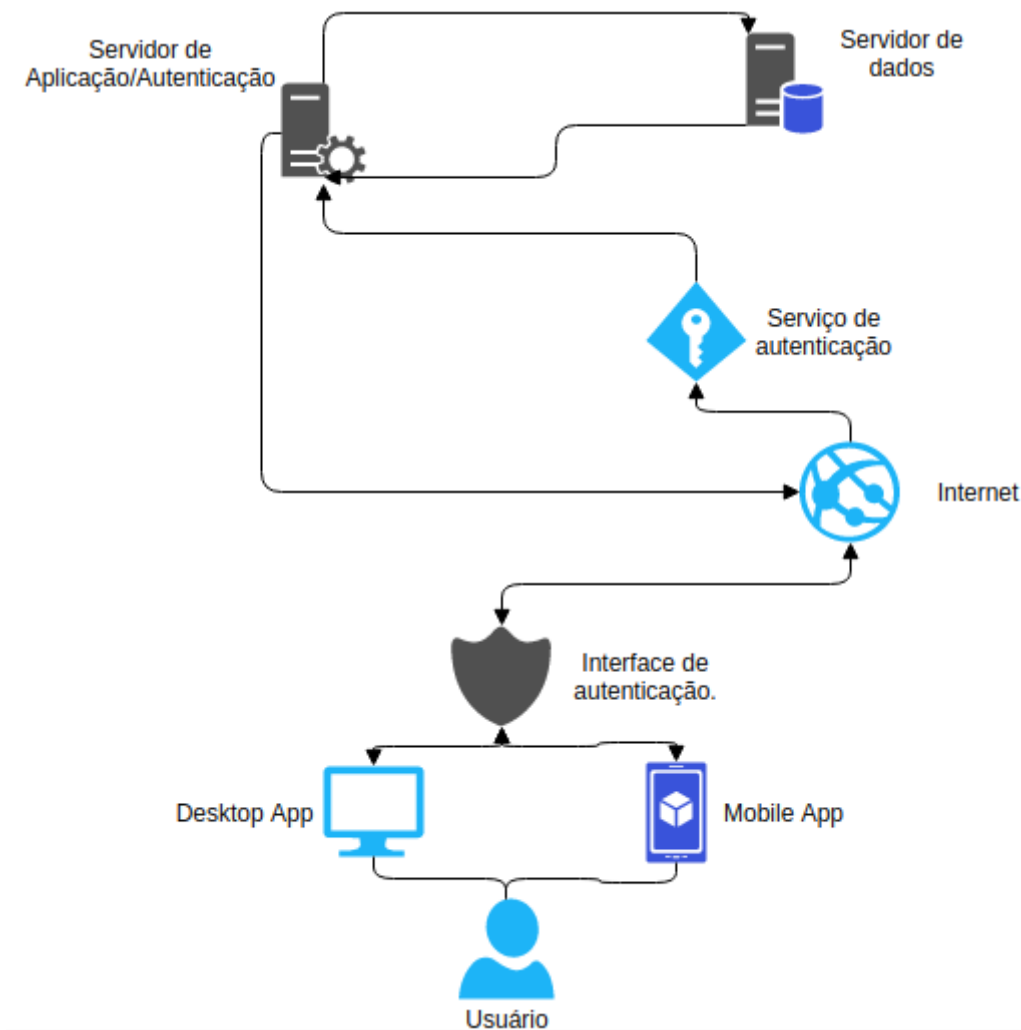


Presence APP

Introdução

O projeto consiste na idealização e implementação de uma API REST responsável por registrar presença em eventos e recolher feedback de usuários a cerca de eventos. No aspecto funcional, por via da leitura de um código gerado por quem ministra/organiza o evento, o usuário consegue utilizando um aplicativo mobile registrar sua presença no evento e fazer avaliações/comentários em tempo real sobre o evento.

O usuário é autenticado através de um serviço de login, que pode implementar mecanismos como OAuth ou registre/sing-in. Ao digitar suas credencias o sistema devolve um token que dali pra frente é utilizado para fornecer



Aparatos Tecnologicos

API

API (Application Protocol Interface - Protocolo de interface de aplicação) é um conjunto padrões que uma determinada aplicação estabelece para que outras aplicações consumam seus serviços sem a necessidade de saber detalhes sobre a implementação do software, disponibilizando uma interoperabilidade entre sistemas. Utilizando do protocolo HTTP o principal protocolo da Web, a fundação sobre qual toda a informação da internet trafega, e implementando suas convenções uma API fornece recursos através de contratos de informações, através de requisições e os envia sob uma promessa de resposta ao solicitante. Sobre esta premissa foi desenvolvido a aplicação, com o intuito de interoperabilidade e independência da camada de apresentação.

Swagger

O swagger é uma projeto composto de ferramentas que auxiliam no desenvolvimento de API's REST. Com ele é possível modelar, gerar documentos legíveis da API e até mesmo gerar códigos de esqueletos para o cliente e servidor facilitando o desenvolvimento da aplicação. Em nosso projeto utilizamos somente a interface do Swagger UI para gerar e publicar a documentação dos serviços dos serviços da API juntamente com uma interface testável de suas funcionalidades e contrato.

O resultado pode ser verificado em no endereço [http\(s\)://api-root/swagger/index.html](http(s)://api-root/swagger/index.html)

Atividades

Google Chrome

qua, 20:47

(29) MySQL wor x | How to Install | x | Unable to conn x | Reset a MySQL x | https://localho x | Swagger UI x | past screen sho x | + | - | x

← → ↻

Não seguro | localhost:5001/swagger/index.html

☆ ▼ 🌙 🕒 🧑

Swagger

Powered by SMARTBEAR

Select a definition

Presence API

Presence API v1 OAS3

[v1/swagger.json](#)

Authorize

Identity

POST /api/v1/identity/register

POST /api/v1/identity/login

POST /api/v1/identity/refresh

Presencas

GET /api/v1/presencas

Atividades

Google Chrome

qua, 20:47

(29) MySQL wor x | How to Install | x | Unable to conn x | Reset a MySQL x | https://localho x | Swagger UI x | past screen sho x | + | - | x

← → ↻

Não seguro | localhost:5001/swagger/index.html

☆ ▼ 🌙 🕒 🧑

POST /api/v1/identity/register

Parameters

No parameters

Request body

application/json

Example Value | Schema

```
{  "email": "string",  "senha": "string"}
```

Responses

Code	Description	Links
200	Success	No links

POST /api/v1/identity/login

The top screenshot shows the Swagger UI interface for a REST API. The 'Request body' tab is selected, showing a JSON object for user registration:

```
{  "email": "gabriel.engsist@gmail.com.br",  "senha": "abcABC@123"}
```

The 'Execute' button is visible at the bottom of the request body section.

The bottom screenshot shows the 'Responses' tab, displaying the response for a successful registration (200 status code). The response body is a JSON object containing a token and a refresh token:

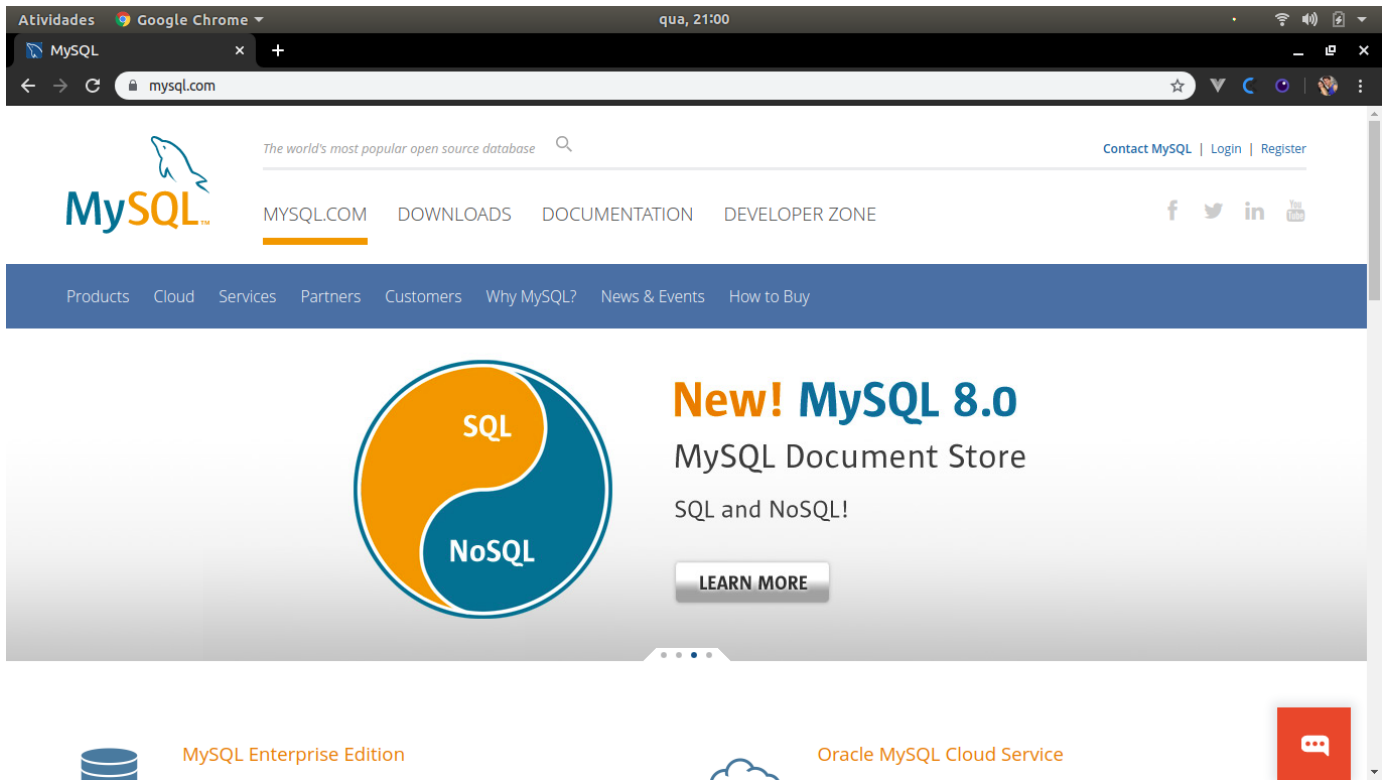
```
{  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJmYWRjaWVzLnVzZ3Npc3RAZ2Z1haWwY29tLmJyIiwianRpIjoiaWFianRpIiwiaWF0Ij01Z0UzNTBhYy1hYTALITQzMWVhYy1jdmOC1lNzQ4MmVlNzZxMDU1LjCjYmY0JlNzI0NzkyOTQsImV4cCI6MTU3HjQ3OTMzOSw1aWF0IjoNTcyNDc5Hj0fQ.rjmdpGUMWqJbT_k_if8mASAERGALuxCa2LW16nhm8",  "refreshToken": "cde14728-f437-4f26-9ddd-cc9b2968536b"}
```

The response headers are also displayed:

```
content-type: application/json; charset=utf-8date: Wed, 30 Oct 2019 23:48:14 GMTserver: Kestrelstatus: 200
```

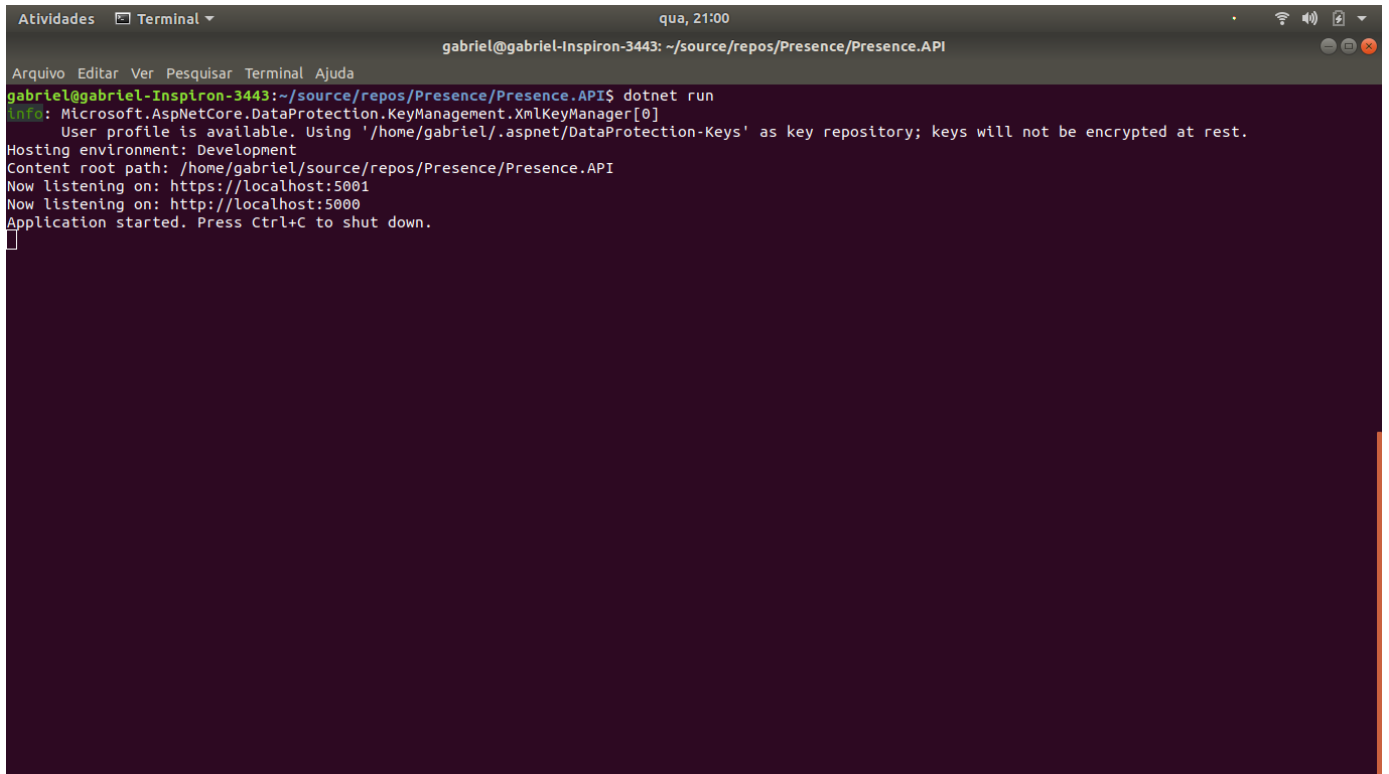
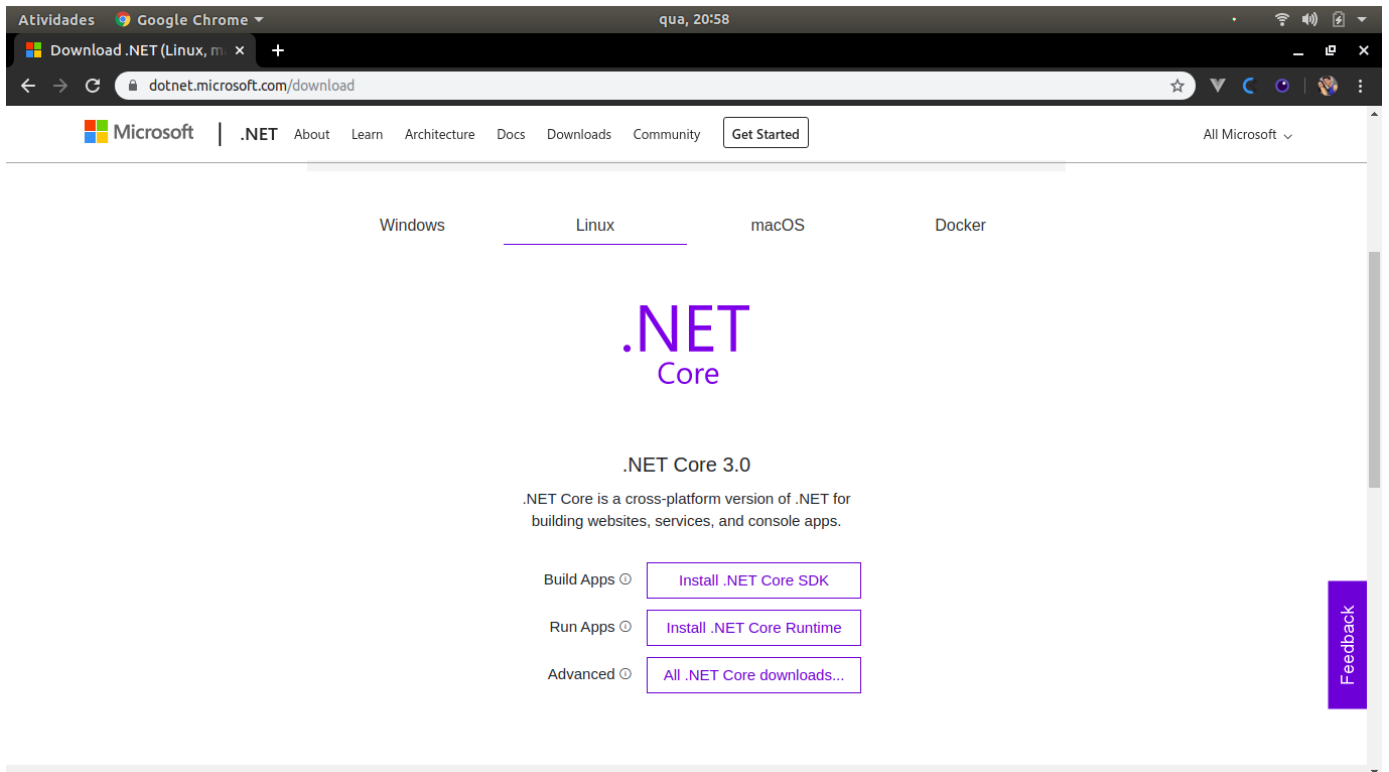
MySql

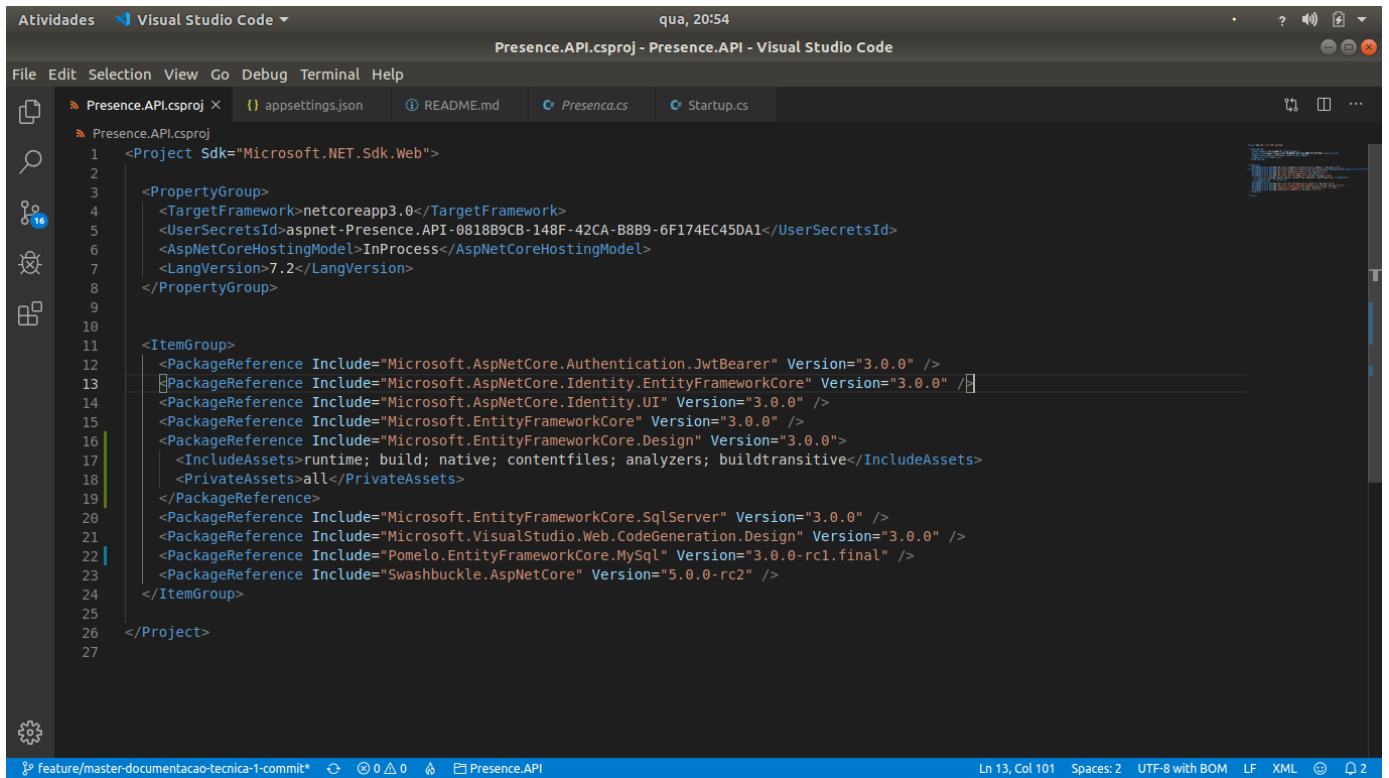
O MySql server é um sistema de gerenciamento de banco de dados popularmente conhecido e difundido. É através dele que salvamos toda a estrutura de dados de nossa aplicação e organizamos as apresentações e persistencia de informação para o usuário.



DotNet Core

Não menos importante e o motor de todo este engenho, está o DotNet Core. Responsável por servir e disponibilizar toda a estrutura funcional da aplicação. O DotNet Core consiste de uma solução de software livre multiplataforma 🖥️🍏🐧 que disponibiliza vários recursos que possibilitam a criação de aplicações modernas e robustas. Ele serve como um interpretador e compilador de código além de um servidor de aplicação.





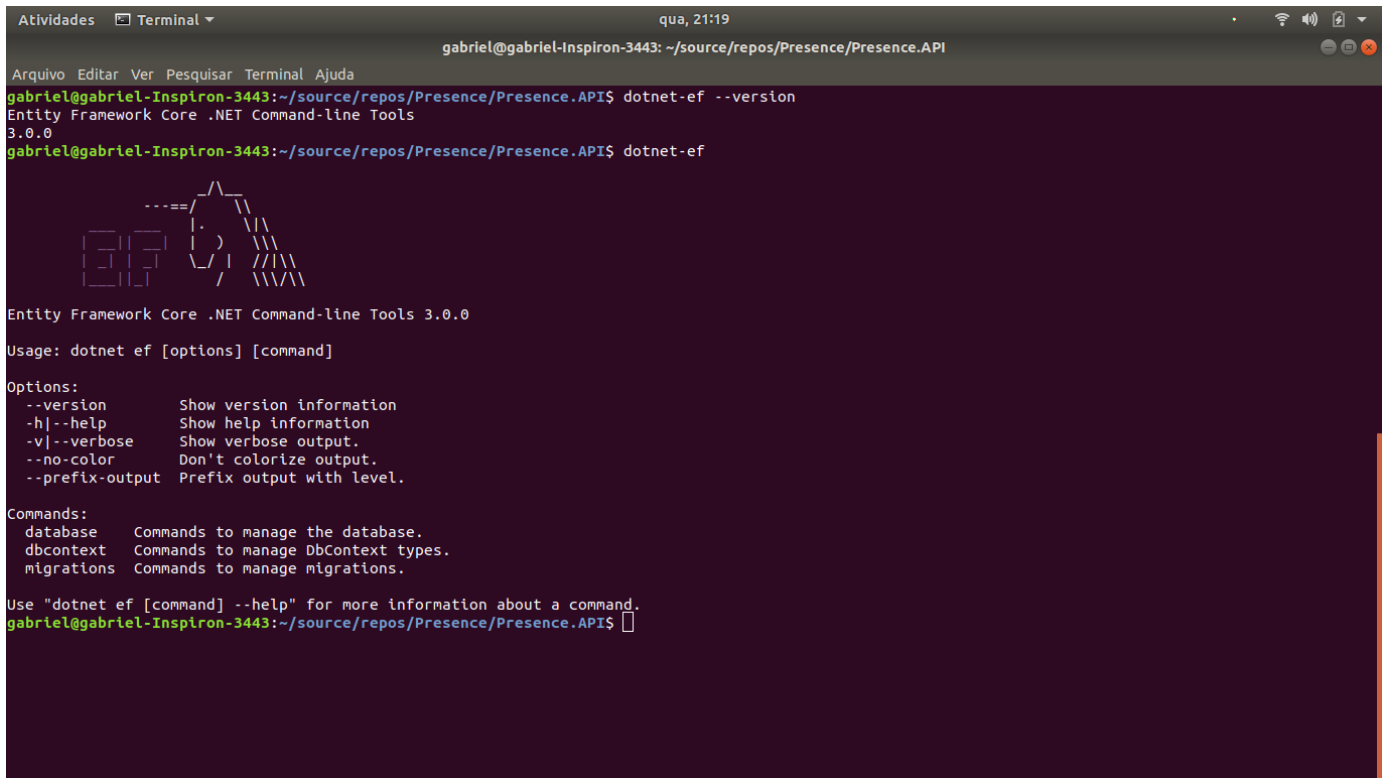
```
1 <Project Sdk="Microsoft.NET.Sdk.Web">
2
3   <PropertyGroup>
4     <TargetFramework>netcoreapp3.0</TargetFramework>
5     <UserSecretsId>aspnet-Presence.API-0818B9CB-148F-42CA-B8B9-6F174EC45DA1</UserSecretsId>
6     <AspNetCoreHostingModel>InProcess</AspNetCoreHostingModel>
7     <LangVersion>7.2</LangVersion>
8   </PropertyGroup>
9
10
11   <ItemGroup>
12     <PackageReference Include="Microsoft.AspNetCore.Authentication.JwtBearer" Version="3.0.0" />
13     <PackageReference Include="Microsoft.AspNetCore.Identity.EntityFrameworkCore" Version="3.0.0" />
14     <PackageReference Include="Microsoft.AspNetCore.Identity.UI" Version="3.0.0" />
15     <PackageReference Include="Microsoft.EntityFrameworkCore" Version="3.0.0" />
16     <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="3.0.0">
17       <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
18       <PrivateAssets>all</PrivateAssets>
19     </PackageReference>
20     <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="3.0.0" />
21     <PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="3.0.0" />
22     <PackageReference Include="Pomelo.EntityFrameworkCore.MySql" Version="3.0.0-rc1.final" />
23     <PackageReference Include="Swashbuckle.AspNetCore" Version="5.0.0-rc2" />
24   </ItemGroup>
25
26 </Project>
27
```

CSharp

CSharp é uma linguagem Orientada a Objetos criada pela Microsoft, é com ela que transformamos nossas idéias em uma aplicação funcional. Interpretada pelo DotNet Core é ela que contém toda a lógica por trás do funcionamento de nossa solução.

Entity Framework Core

O Entity Framework Core é o ORM (Object Relational Mapper - Mapeador de objeto-relacional) escolhido para fazer a gestão e persistencia da informação, ela serve como tradutora entre o código e serviço de banco de dados relacional escolhido. Utilizando da terminologia CodeFirst(Primeiro o código), serve como um facilitador traduzindo, gerando os scripts e a estrutura do banco de dados, de uma forma em que o programador se preocupe somente em escrever o código.



```
Atividades Terminal qua, 21:19
gabriel@gabriel-Inspiron-3443: ~/source/repos/Presence/Presence.API

Arquivo Editar Ver Pesquisar Terminal Ajuda
gabriel@gabriel-Inspiron-3443:~/source/repos/Presence/Presence.API$ dotnet-ef --version
Entity Framework Core .NET Command-line Tools
3.0.0
gabriel@gabriel-Inspiron-3443:~/source/repos/Presence/Presence.API$ dotnet-ef

Entity Framework Core .NET Command-line Tools 3.0.0

Usage: dotnet ef [options] [command]

Options:
  --version          Show version information
  -h|--help          Show help information
  -v|--verbose       Show verbose output.
  --no-color         Don't colorize output.
  --prefix-output    Prefix output with level.

Commands:
  database           Commands to manage the database.
  dbcontext          Commands to manage DbContext types.
  migrations         Commands to manage migrations.

Use "dotnet ef [command] --help" for more information about a command.
gabriel@gabriel-Inspiron-3443:~/source/repos/Presence/Presence.API$
```

Aplicativo

Flutter

Outras Ferramentas

- Visual Studio Code - Editor de texto/IDE maleável que é utilizada no desenvolvimento do Front-End e do Back-end
- Post Man - Ferramenta para testes de conectividade da API, com ela é possível verificar respostas e efetuar testes de integração na aplicação.
- Visual Studio IDE - Ambiente de desenvolvimento gráfico que facilita o desenvolvimento de aplicações utilizando o DotNet Core.
- MySql Workbench - Interface para a modelagem e manipulação do banco de dados

Metodologia

Scrum

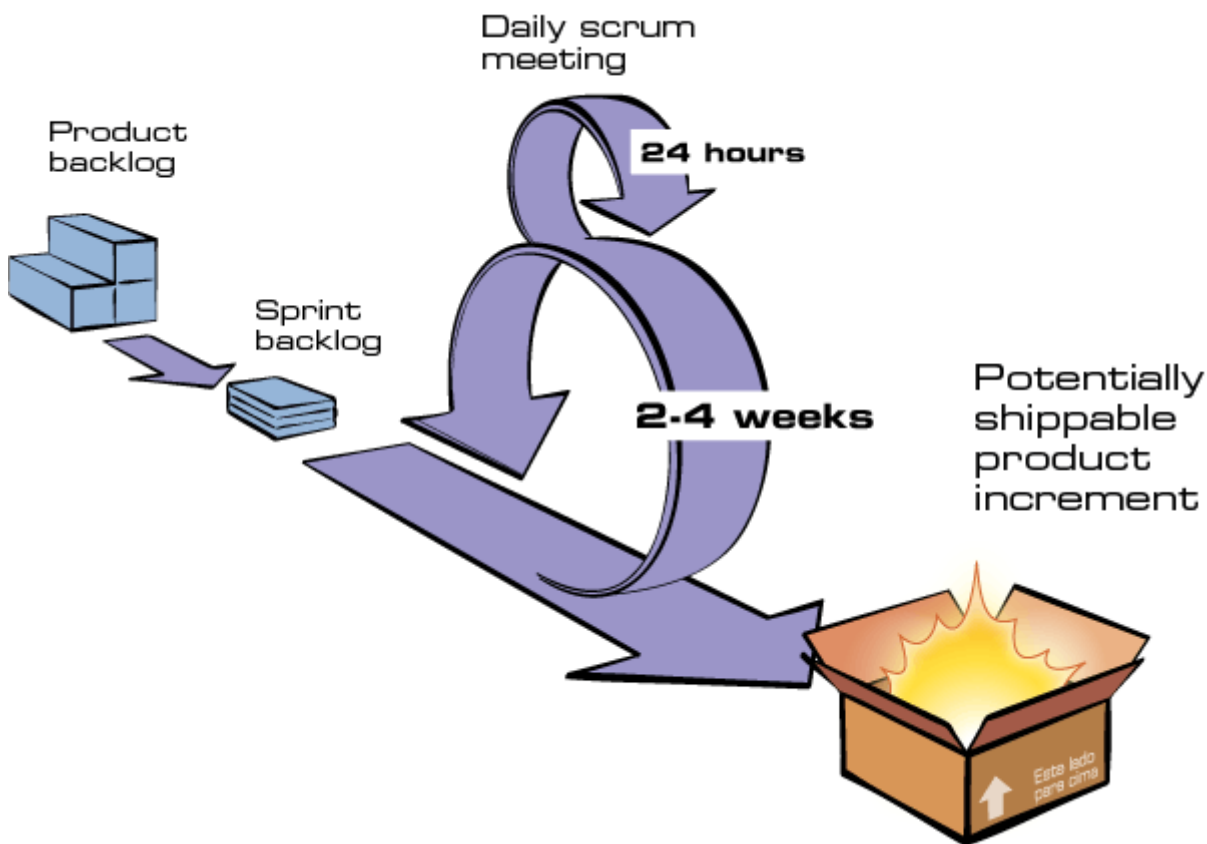
Scrum é uma metodologia ágil para gestão e planejamento de projetos de software.

No Scrum, os projetos são divididos em ciclos (tipicamente mensais) chamados de Sprints. O Sprint representa um Time Box dentro do qual um conjunto de atividades deve ser executado. Metodologias ágeis de desenvolvimento de software são iterativas, ou seja, o trabalho é dividido em iterações, que são chamadas de Sprints no caso do Scrum.

As funcionalidades a serem implementadas em um projeto são mantidas em uma lista que é conhecida como Product Backlog. No início de cada Sprint, faz-se um Sprint Planning Meeting, ou seja, uma reunião de planejamento na qual o Product Owner prioriza os itens do Product Backlog e a equipe seleciona as atividades que ela será capaz de implementar durante o Sprint que se inicia. As tarefas alocadas em um Sprint são transferidas do Product Backlog para o Sprint Backlog.

A cada dia de uma Sprint, a equipe faz uma breve reunião (normalmente de manhã), chamada Daily Scrum. O objetivo é disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho do dia que se inicia.

Ao final de um Sprint, a equipe apresenta as funcionalidades implementadas em uma Sprint Review Meeting. Finalmente, faz-se uma Sprint Retrospective e a equipe parte para o planejamento do próximo Sprint. Assim reinicia-se o ciclo.



 Retirado de:

Documentação

Para gerar a documentação utilizamos o Docsify, que é responsável por indexar e disponibilizar a documentação da aplicação. Ele funciona como um site estático que interpreta Arquivos do Tipo Markdown e faz uma linda camada de apresentação para o usuário.

