E-Players

Projeto de aprendizagem MVC com ASP Net Core

Parte 1 - Models

> Criamos nossos models Equipe, Jogador, Partida e Notícias com suas devidas propriedades

```
public class Equipe
{
    public int IdEquipe { get; set; }
    public string Nome { get; set; }
    public string Imagem { get; set; }
}


public class Jogador
{
    public int IdJogador { get; set; }
```

```
        public string Nome { get; set; }
        public int IdEquipe { get; set; }
    }


    public class Partida
    {
        public int IdPartida { get; set; }
        public int IdJogador1 { get; set; }
        public int IdJogador2 { get; set; }
        public DateTime HorarioInicio { get; set; }
        public DateTime HorarioTermino { get; set; }
    }


    public class Noticias
    {
        public int IdNoticia { get; set; }
        public string Titulo { get; set; }
        public string Texto { get; set; }
        public string Imagem { get; set; }
    }
```

> Criamos uma superclasse chamada EPlayersBase para abstrair alguns métodos de manipulação do CSV

```
    public class EplayersBase
    {
        public void CreateFolderAndFile(string _path){

            string folder   = _path.Split("/")[0];
            string file     = _path.Split("/")[1];

            if(!Directory.Exists(folder)){
                Directory.CreateDirectory(folder);
            }

            if(!File.Exists(_path)){
                File.Create(_path).Close();
            }
        }
    }
```

> Também criamos uma interface para Equipe para solidificar nossa estrutura:

```
using E_Players.Models;

namespace E_Players.Interfaces
```

```
{
    public interface IEquipe
    {
        void Create(Equipe e);

        List<Equipe> ReadAll();

        void Update(Equipe e);

        void Delete(int id);
    }
}
```

> Com a interface criada, herdamos em Equipe a classe EplayersBase e a interface IEquipe

```
public class Equipe : EplayersBase , IEquipe
```

> Implementamos a interface e começamos o desenvolvimento do método Create:

```
using System;
using System.IO;
using E_Players.Interfaces;

namespace E_Players.Models
{
    public class Equipe : EplayersBase , IEquipe
    {
        public int IdEquipe { get; set; }
        public string Nome { get; set; }
        public string Imagem { get; set; }
        private const string PATH = "Database/equipe.csv";

        public Equipe(){
            CreateFolderAndFile(PATH);
        }

        public void Create(Equipe e){
            string[] linha = { Prepare(e) };
            File.AppendAllLines(PATH, linha);
        }

        private string Prepare(Equipe e){
            return $"{e.IdEquipe};{e.Nome};{e.Imagem}";
        }

        public List<Equipe> ReadAll()
```

```
        {
            throw new NotImplementedException();
        }

        public void Update(Equipe e)
        {
            throw new NotImplementedException();
        }

        public void Delete(int id)
        {
            throw new NotImplementedException();
        }
    }
}
```

> Depois, dentro de Equipe, criamos as funcionalidades do método ReadAll

```
        public List<Equipe> ReadAll()
        {
            List<Equipe> equipes = new List<Equipe>();
            string[] linhas = File.ReadAllLines(PATH);
            foreach (var item in linhas)
            {
                string[] linha = item.Split(";");
                Equipe equipe = new Equipe();
                equipe.IdEquipe = Int32.Parse(linha[0]);
                equipe.Nome = linha[1];
                equipe.Imagem = linha[2];

                equipes.Add(equipe);
            }
            return equipes;
        }
```

> Dentro de EplayersBase, criamos o método que vai nos ajudar a ler todas as linhas de um csv, e o outro para re-escrevê-lo. Lembre-se de que eles poderão ser aplicados em qualquer classe que herdar de EplayersBase:

```
        public List<string> ReadAllLinesCSV(string PATH){

            List<string> linhas = new List<string>();
            using(StreamReader file = new StreamReader(PATH))
            {
                string linha;
```

```
            while((linha = file.ReadLine()) != null)
            {
                linhas.Add(linha);
            }
        }
        return linhas;
    }

    public void RewriteCSV(string PATH, List<string> linhas)
    {
        using(StreamWriter output = new StreamWriter(PATH))
        {
            foreach (var item in linhas)
            {
                output.Write(item + "\n");
            }
        }
    }
```

> Com esses métodos base criados, nossos métodos restantes, Update e Delete ficam mais fáceis de serem implantados:

```
    public void Update(Equipe e)
    {
        List<string> linhas = ReadAllLinesCSV(PATH);
        linhas.RemoveAll(x => x.Split(";")[0] == e.IdEquipe.ToString());
        linhas.Add( Prepare(e) );
        RewriteCSV(PATH, linhas);
    }

    public void Delete(int id)
    {
        List<string> linhas = ReadAllLinesCSV(PATH);
        linhas.RemoveAll(x => x.Split(";")[0] == id.ToString());
        RewriteCSV(PATH, linhas);
    }
```

Parte 2 - Controllers

> Criamos a classe ***EquipeController***, com a rota geral "Equipe" para toda a classe

```
using System;
using System.Collections.Generic;
```

```
using System.Diagnostics;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using E_Players.Models;

namespace E_Players.Controllers
{
    [Route("Equipe")]
    public class EquipeController : Controller
    {
        public IActionResult Index()
        {
            return View();
        }
    }
}
```

> Dentro da classe instanciamos nosso model de Equipe e passamos ele como retorno da View, para conseguir acessá-lo via Razor

```
        Equipe equipeModel = new Equipe();

        public IActionResult Index()
        {
            ViewBag.Equipes = equipeModel.ReadAll();
            return View();
        }
```

> Criamos o método *Cadastrar()*, passando como argumento um *IFormCollection*:

```
    public IActionResult Cadastrar(IFormCollection form)
    {

    }
```

> Depois capturamos as informações que virão do formulário e passamos para um objeto do tipo **Equipe**

```
        public IActionResult Cadastrar(IFormCollection form)
        {
            Equipe novaEquipe   = new Equipe();
            novaEquipe.IdEquipe = Int32.Parse(form["IdEquipe"]);
            novaEquipe.Nome     = form["Nome"];
            novaEquipe.Imagem   = form["Imagem"];

            equipeModel.Create(novaEquipe);
```

```
            ViewBag.Equipes = equipeModel.ReadAll();

            return LocalRedirect("~/Equipe");
        }
```

Parte 3 - Views

> Dentro da pasta Views, criamos um diretório chamado Equipe, e dentro dele um arquivo chamado Index.cshtml
> Dentro deste arquivo chamamos nossa model e mudamos a ViewData do Title

```
@model Equipe
@{
    ViewData["Title"] = "Equipes";
}
```

> Logo em baixo criamos um form bem simples para testar, implementando a ação via Razor:

```
<form method="POST" action='@Url.Action("Cadastrar")'>
    <label>ID</label>
    <input type="text" name="IdEquipe" />

    <label>Nome</label>
    <input type="text" name="Nome" />

    <label>Imagem</label>
    <input type="text" name="Imagem" />

    <button type="submit">Cadastrar</button>
</form>

<table class="table table-striped table-responsive">
    <thead>
        <th>ID</th>
        <th>Nome</th>
        <th>Foto</th>
    </thead>
    <tbody>
        @foreach(Equipe e in ViewBag.Equipes){
            <tr>
                <td>@e.IdEquipe</td>
                <td>@e.Nome</td>
                <td>@e.Imagem</td>
```

```
            </tr>
        }
    </tbody>
</table>
```

> Dentro da pasta Views/Home/Indesx.cshtml, adicionamos um link para abrir nossa página:

```
<a class="navbar-brand" asp-area="" asp-controller="Equipe" asp-action="Index"
```

> Rodamos e testamos nossa aplicação com:

```
dotnet run
```

Parte 4 - Upload de Imagem

> No método Cadastrar colocamos todo o comportamento necessário para realizar o upload de imagem, com a rota diferente da padrão:

```
        [Route("Cadastrar")]
        public IActionResult Cadastrar(IFormCollection form)
        {
            Equipe novaEquipe = new Equipe();
            novaEquipe.IdEquipe = Int32.Parse( form["IdEquipe"] );
            novaEquipe.Nome = form["Nome"];

            if(form.Files.Count > 0)
            {
                // Upload Início
                var file    = form.Files[0];
                var folder  = Path.Combine(Directory.GetCurrentDirectory(), "w

                if(!Directory.Exists(folder)){
                    Directory.CreateDirectory(folder);
                }

                var path = Path.Combine(Directory.GetCurrentDirectory(), "wwwr
                using (var stream = new FileStream(path, FileMode.Create))
                {
                    file.CopyTo(stream);
                }
                novaEquipe.Imagem   = file.FileName;
            }
```

```
            else
            {
                novaEquipe.Imagem    = "padrao.png";
            }
            // Upload Final

            equipeModel.Create(novaEquipe);

            ViewBag.Equipes = equipeModel.ReadAll();
            return LocalRedirect("~/Equipe");
        }
```

> Adicionamos o ***enctype*** para permitir a inserção de arquivos e mudamos o tipo do input *Imagem* para ***file***:

```
<form method="POST" action='@Url.Action("Cadastrar")' enctype="multipart/form-

    <label>ID</label>
    <input type="text" name="IdEquipe" />

    <label>Nome</label>
    <input type="text" name="Nome" />

    <label>Imagem</label>
    <input type="file" name="Imagem" />

    <button type="submit">Cadastrar</button>

</form>
```

Parte 5 - Excluir

> Em EquipeController adicionamos o metodo Excluir

```
        [Route("Equipe/{id}")]
        public IActionResult Excluir(int id)
        {
            equipeModel.Delete(id);
            ViewBag.Equipes = equipeModel.ReadAll();
            return LocalRedirect("~/Equipe");
        }
```

> E na tabela do Index.cshtml adicionamos mais uma coluna com o link para excluir:

```
<table class="table table-striped table-responsive">
    <thead>
        <th>ID</th>
        <th>Nome</th>
        <th>Foto</th>
    </thead>
    <tbody>
        @foreach(Equipe e in ViewBag.Equipes){
            <tr>
                <td>@e.IdEquipe</td>
                <td>@e.Nome</td>
                <td><img src="img/Equipes/@e.Imagem" alt="Imagem da equipe @e.
                <td><a asp-area="" asp-controller="Equipe" asp-action="Excluir
            </tr>
        }
    </tbody>
</table>
```

Parte 6 - Aplicando o design base

> Colocamos dentro da raiz nosso diretório base do e-players
> Copiamos as pastas ***css*** e ***images*** do **E-PlayersBase** para a pasta ***wwwroot***
> Dentro de Views > Shared > _Layout.cshtml, importamos o css e removemos o que vem por padrão no projeto, alterando também o lang para pt-br:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - E_Players</title>
    <!-- Novo CSS e JS -->
    <link href="https://fonts.googleapis.com/css?family=Titillium+Web:400,700&
    <script src="https://kit.fontawesome.com/2faae0d38b.js" crossorigin="anony
    <link rel="stylesheet" href="~/css/style.css" />

</head>
```

> Depois trocamos para nosso Header da base e atualizamos os links para asp net:

```
    <header>
        <div class="container cabecalho">
```

```
            <div>

                <a href="#" onclick="botaoMenu()">
                    <i class="fas fa-bars"></i>
                </a>

                <a asp-area="" asp-controller="Home" asp-action="Index">
                <img src="Images/Logo.svg" alt="Logo do site E-Players">
            </a>
            </div>
            <nav id="menu">
                <ul>
                    <li><a asp-area="" asp-controller="Home" asp-action="Index
                    <li><a href="#horarios" title="Ir até a sessão de horários
                    <li><a href="#resultados" title="Ir até a sessão de result
                    <li><a href="#noticias" title="Ir até a sessão de notícias
                    <li><a href="#" title="buscar conteudo no site"><i class="
                </ul>
            </nav>
        </div>
    </header>
```

> Fazemos o mesmo com o footer, removendo as bibliotecas de jquery desnecessárias nesse projeto:

```
    <footer>
        <div class="container">
            <div>
                <h5>INSTITUCIONAL</h5>
                <nav>
                    <ul>
                        <li><a href="#">Fale Conosco</a></li>
                        <li><a href="#">Regras</a></li>
                        <li><a href="#">Suporte</a></li>
                        <li><a href="#">Política de Privacidade</a></li>
                        <li><a href="#">Termos de Uso</a></li>
                        <li><a href="#">Anuncie</a></li>
                    </ul>
                </nav>
            </div>
            <div class="mob_logo_sociais">
                <img src="Images/Logo.svg" alt="Logo do projeto E-Players">
                <div class="sociais">
                    <i class="fab fa-facebook-square"></i>
                    <i class="fab fa-instagram-square"></i>
                    <i class="fab fa-twitter-square"></i>
                    <i class="fab fa-youtube-square"></i>
                </div>
            </div>
        <div>
```

```
            <h5>PRINCIPAIS LIGAS</h5>
            <h6>LIGA CBL</h6>
            <h6>LIGA LCK</h6>
            <h6>LIGA PROFISSIONAL</h6>
            <h6>LIGA WCG</h6>
        </div>
    </div>
        <p>© Copyrigth 2077 - Todos os direitos reservados ao sena-code</p>
    </footer>

    <script src="~/js/site.js" asp-append-version="true"></script>
```

> Depois vamos para a pasta Home > Index e adicionamos nosso main lá dentro:

```
@{
    ViewData["Title"] = "Home Page";
}

<div class="banner">
    <a asp-area="" asp-controller="Jogador" asp-action="Index" class="btn grad
</div>

<div class="sobre">
    <div class="container">
        <div>
            <h1>O QUE É MUNDO EM COMUM ?</h1>
            <p> Lorem ipsum dolor sit amet consectetur adipisicing elit. Offic
                asperiores officia, quisquam eius optio itaque nobis vel. Amet
                at facere eaque adipisci laborum. Lorem ipsum dolor sit amet c
                ipsum, veniam ad, illum corrupti nisi fugiat repellat harum as
                voluptatibus nulla a excepturi facere id non nemo totam! Lorem
                adipisicing elit. Cum numquam expedita architecto quia debitis
                itaque illum aut officiis, quasi tempora esse id incidunt cons
                dolor, sit amet consectetur adipisicing elit. Natus, beatae ul
                excepturi tempore a unde, ipsa facilis aut et tempora nam quae
                ipsum dolor sit amet consectetur adipisicing elit. Tempore ab
                molestias aliquid eligendi laboriosam sint optio dolor, accusa
                harum veniam fuga sequi. Lorem ipsum dolor sit amet consectetu
        </div>
    </div>
</div>
<img src="Images/9a6e4f50b0d94f75ce274348f06a6d56.png" alt="Imagem de um
campeão do lol" class="champion">

<section id="horarios">

    <div class="container">
        <div class="ao_vivo">
            <h2>AO VIVO</h2>
            <div class="partida">
```

```
                <strong>SK</strong>
                <img src="Images/SK.png" alt="Logo da equipe SK">
                <I>VS</I>
                <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
                <strong>FLA</strong>
            </div>
        </div>

        <div class="em_breve">
            <h2>EM BREVE</h2>
            <div class="partida">
                <strong>SK</strong>
                <img src="Images/SK.png" alt="Logo da equipe SK">
                <I>VS</I>
                <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
                <strong>FLA</strong>
            </div>
            <div class="partida">
                <strong>SK</strong>
                <img src="Images/SK.png" alt="Logo da equipe SK">
                <I>VS</I>
                <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
                <strong>FLA</strong>
            </div>
            <div class="partida">
                <strong>SK</strong>
                <img src="Images/SK.png" alt="Logo da equipe SK">
                <I>VS</I>
                <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
                <strong>FLA</strong>
            </div>
        </div>
    </div>
</section>

<div class="container content">
    <section class="noticias">
        <div class="card">
            <img src="Images/Teams/T1.jpeg" alt="Notícia sobra assunto 1">
            <div>
            <h4>Título da Noticia</h4>
            <p> Lorem, ipsum dolor sit amet consectetur adipisicing elit. Quos
                velit vel sapiente nemo obcaecati veritatis dolorum odio facer
                Aspernatur quam reiciendis ratione dolorem!</p>
                </div>
        </div>


        <div class="card">
            <img src="Images/Teams/T2.jpg" alt="Notícia sobra assunto 1">
            <div>
            <h4>Título da Noticia</h4>
```

```
            <p> Lorem, ipsum dolor sit amet consectetur adipisicing elit. Quos
                velit vel sapiente nemo obcaecati veritatis dolorum odio facer
                Aspernatur quam reiciendis ratione dolorem!</p>
                </div>
        </div>

        <div class="card">
            <img src="Images/Teams/T3.jpeg" alt="Notícia sobra assunto 1">
            <div>
            <h4>Título da Noticia</h4>
            <p> Lorem, ipsum dolor sit amet consectetur adipisicing elit. Quos
                velit vel sapiente nemo obcaecati veritatis dolorum odio facer
                Aspernatur quam reiciendis ratione dolorem!</p>
        </div>
        </div>

        <div class="card">
            <img src="Images/Teams/T9.jpeg" alt="Notícia sobra assunto 1">
            <div>
            <h4>Título da Noticia</h4>
            <p> Lorem, ipsum dolor sit amet consectetur adipisicing elit. Quos
                velit vel sapiente nemo obcaecati veritatis dolorum odio facer
                Aspernatur quam reiciendis ratione dolorem!</p>
                </div>
        </div>

        <div class="card">
            <img src="Images/Teams/T5.jpeg" alt="Notícia sobra assunto 1">
            <div>
            <h4>Título da Noticia</h4>
            <p> Lorem, ipsum dolor sit amet consectetur adipisicing elit. Quos
                velit vel sapiente nemo obcaecati veritatis dolorum odio facer
                Aspernatur quam reiciendis ratione dolorem!</p>
                </div>
        </div>

        <div class="card">
            <img src="Images/Teams/T7.jpeg" alt="Notícia sobra assunto 1">
            <div>
            <h4>Título da Noticia</h4>
            <p> Lorem, ipsum dolor sit amet consectetur adipisicing elit. Quos
                velit vel sapiente nemo obcaecati veritatis dolorum odio facer
                Aspernatur quam reiciendis ratione dolorem!</p>
                </div>
        </div>
    </section>
    <section class="recentes" id="resultados">
        <h3>PARTIDAS RECENTES</h3>

        <div class="resultado">
            <div class="team">
                <strong>SK</strong>
```

```
            <img src="Images/SK.png" alt="Logo da equipe SK">
            <I>VS</I>
            <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
            <strong>FLA</strong>
        </div>
        <div class="placar">
            <P>3 x 2</p>
        </div>
    </div>
    <div class="resultado">
        <div class="team">
            <strong>SK</strong>
            <img src="Images/SK.png" alt="Logo da equipe SK">
            <I>VS</I>
            <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
            <strong>FLA</strong>
        </div>
        <div class="placar">
            <P>3 x 2</p>
        </div>
    </div>
    <div class="resultado">
        <div class="team">
            <strong>SK</strong>
            <img src="Images/SK.png" alt="Logo da equipe SK">
            <I>VS</I>
            <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
            <strong>FLA</strong>
        </div>
        <div class="placar">
            <P>3 x 2</p>
        </div>
    </div>
    <div class="resultado">
        <div class="team">
            <strong>SK</strong>
            <img src="Images/SK.png" alt="Logo da equipe SK">
            <I>VS</I>
            <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
            <strong>FLA</strong>
        </div>
        <div class="placar">
            <P>3 x 2</p>
        </div>
    </div>
    <div class="resultado">
        <div class="team">
            <strong>SK</strong>
            <img src="Images/SK.png" alt="Logo da equipe SK">
            <I>VS</I>
            <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
            <strong>FLA</strong>
```

```
            </div>
            <div class="placar">
                <P>3 x 2</p>
            </div>
        </div>
        <div class="resultado">
            <div class="team">
                <strong>SK</strong>
                <img src="Images/SK.png" alt="Logo da equipe SK">
                <I>VS</I>
                <img src="Images/FLA.png" alt="Logo da equipe do Flamengo">
                <strong>FLA</strong>
            </div>
            <div class="placar">
                <P>3 x 2</p>
            </div>
        </div>
        <a href="todos_os_resultados.html" class="btn gradient">mais resultado
    </section>
</div>
```

> Na pasta JS, no arquivo site.js, colocamos nosso script que faz o menu funcionar corretamente no modo responsivo:

```
// Please see documentation at https://docs.microsoft.com/aspnet/core/client-s
// for details on configuring this project to bundle and minify static web ass

// Write your JavaScript code.
function botaoMenu(){

    var m = document.getElementById("menu")
    if(m.style.display == "block"){
        m.style.display = "none";
    }else{
        m.style.display = "block"
    }
}
```

> Tiramos a div .container que envolvia o main, pra não prejudicar nosso css, ficando assim:

```
    </header>

    <main role="main" class="pb-3">
        @RenderBody()
    </main>

    <footer>
```

> Testamos nossa aplicação:

```
dotnet run
```

> Em ***Equipe*** > ***Index.cshtml*** Ajustamos nosso form para ficar com a aparência definida no *style.css*:

```
<div class="titulo_pagina">
    <h1>cadastro de equipe</h1>
</div>

<form method="POST" action='@Url.Action("Cadastrar")' enctype="multipart/form-

    <div class="campo">
        <label for="IdEquipe">ID</label>
        <input type="text" name="IdEquipe" id="IdEquipe" />
    </div>

    <div class="campo">
        <label for="Nome">Nome</label>
        <input type="text" name="Nome" id="Nome" />
    </div>

    <div class="campo">
        <label for="Imagem">Imagem</label>
        <input type="file" name="Imagem" for="Imagem" />
    </div>

    <button class="gradient btn" type="submit">Cadastrar</button>

</form>
```

Parte 7 - Login - Parte 1

> Colocamos os atributos de email e senha em Jogador

```
namespace E_Players_AspNETCore.Models
{
```

```
    public class Jogador
    {
        public int IdJogador { get; set; }
        public string Nome { get; set; }
        public int IdEquipe { get; set; }
        // Login
        public string Email { get; set; }
        public string Senha { get; set; }

    }
}
```

> Herdamos EplayersBase em jogador

```
    public class Jogador : EPlayersBase
```

> Criamos o método construtor para gerar o arquivo

```
    private const string PATH = "Database/Jogador.csv";

    public Jogador()
    {
        CreateFolderAndFile(PATH);
    }
```

> Criamos a interface para Jogador -> *IJogador*

```
using System.Collections.Generic;
using E_Players_AspNETCore.Models;

namespace E_Players_AspNETCore.Interfaces
{
    public interface IJogador
    {
        //Criar
        void Create(Jogador j);
        //Ler
        List<Jogador> ReadAll();
        //Alterar
        void Update(Jogador j);
        //Excluir
        void Delete(int id);
    }
}
```

> Implementamos a Interface

```
public class Jogador : EPlayersBase , IJogador
```

> Implementamos os métodos de CRUD no model

```
        /// <summary>
        /// Adiciona uma Jogador ao CSV
        /// </summary>
        /// <param name="j">Jogador</param>
        public void Create(Jogador j)
        {
            string[] linha = { PrepararLinha(j) };
            File.AppendAllLines(PATH, linha);
        }

        /// <summary>
        /// Prepara a linha para a estrutura do objeto Jogador
        /// </summary>
        /// <param name="e">Objeto "Jogador"</param>
        /// <returns>Retorna a linha em formato de .csv</returns>
        private string PrepararLinha(Jogador j)
        {
            return $"{j.IdJogador};{j.Nome};{j.Email};{j.Senha}";
        }

        /// <summary>
        /// Exclui uma Jogador
        /// </summary>
        /// <param name="idJogador"></param>
        public void Delete(int idJogador)
        {
            List<string> linhas = ReadAllLinesCSV(PATH);
            // 1;FLA;fla.png
            linhas.RemoveAll(x => x.Split(";")[0] == idJogador.ToString());
            RewriteCSV(PATH, linhas);
        }

        /// <summary>
        /// Lê todos as linhas do csv
        /// </summary>
        /// <returns>Lista de Jogadors</returns>
        public List<Jogador> ReadAll()
        {
            List<Jogador> jogadores = new List<Jogador>();
            string[] linhas = File.ReadAllLines(PATH);

            foreach (var item in linhas)
            {
                string[] linha = item.Split(";");
```

```
                Jogador jogador = new Jogador();
                jogador.IdJogador = int.Parse(linha[0]);
                jogador.Nome = linha[1];
                jogador.Email = linha[2];
                jogador.Senha = linha[3];

                jogadores.Add(jogador);
            }
            return jogadores;
        }

        /// <summary>
        /// Altera uma Jogador
        /// </summary>
        /// <param name="j">Jogador alterada</param>
        public void Update(Jogador j)
        {
            List<string> linhas = ReadAllLinesCSV(PATH);
            linhas.RemoveAll(x => x.Split(";")[0] == j.IdJogador.ToString());
            linhas.Add( PrepararLinha(j) );
            RewriteCSV(PATH, linhas);
        }
```

> Criamos o controller *JogadorController.cs*, com o método de retorno pra view da index:

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;

namespace E_Players_AspNETCore.Controllers
{
    [Route("Jogador")]
    public class JogadorController : Controller
    {
        public IActionResult Index()
        {
            return View();
        }
    }
}
```

> Instanciamos Jogador dentro da classe e chamamos uma ViewBag como apoio e retorno para listar os jogadores disponíveis:

```
        Jogador jogadorModel = new Jogador();

        public IActionResult Index()
        {
            ViewBag.Jogadores = jogadorModel.ReadAll();
            return View();
        }
```

> Criamos o método Cadastrar(), passando como argumento um IFormCollection:

```
        public IActionResult Cadastrar(IFormCollection form)
        {
            Jogador novoJogador     = new Jogador();
            novoJogador.IdJogador   = Int32.Parse(form["IdJogador"]);
            novoJogador.Nome        = form["Nome"];
            novoJogador.Email       = form["Email"];
            novoJogador.Senha       = form["Senha"];

            jogadorModel.Create(novoJogador);
            ViewBag.Jogadores = jogadorModel.ReadAll();

            return LocalRedirect("~/Jogador");
        }
```

> Dentro da pasta Views, criamos um diretório chamado Jogador, e dentro dele um arquivo chamado Index.cshtml Dentro deste arquivo chamamos nossa model e mudamos a ViewData do Title:

```
    @model Jogador
    @{
        ViewData["Title"] = "Jogadores";
    }
```

> Logo em baixo criamos um form bem simples para testar, implementando a ação via Razor:

```
@model Jogador
@{
    ViewData["Title"] = "Jogadores";
}

<div class="titulo_pagina">
    <h1>cadastro de @ViewData["Title"]</h1>
</div>

<form method="POST" action='@Url.Action("Cadastrar")' class="cadastro">
```

```
    <div class="campo">
        <label>ID Jogador</label>
        <input type="text" name="IdJogador" />
    </div>

    <div class="campo">
        <label>ID Equipe</label>
        <input type="text" name="IdEquipe" />
    </div>

    <div class="campo">
        <label>Nome</label>
        <input type="text" name="Nome" />
    </div>

    <div class="campo">
        <label>Email</label>
        <input type="text" name="Email" />
    </div>

    <div class="campo">
        <label>Senha</label>
        <input type="password" name="Senha" />
    </div>

    <button class="gradient btn" type="submit">Cadastrar</button>
</form>

<table class="table table-striped table-responsive">
    <thead>
        <th>ID</th>
        <th>Nome</th>
        <th>Email</th>
    </thead>
    <tbody>
        @foreach(Jogador j in ViewBag.Jogadores){
            <tr>
                <td>@j.IdJogador</td>
                <td>@j.Nome</td>
                <td>@j.Email</td>
            </tr>
        }
    </tbody>
</table>
```

> Dentro da pasta Views/Home/Indesx.cshtml, adicionamos um link para abrir nossa página:

```
<a class="navbar-brand" asp-area="" asp-controller="Jogador" asp-action="Index
```

Parte 7 - Login - Parte 2

> Na pasta ***Views*** criamos uma nova pasta chamada ***Login*** e dentro de login colocamos um arquivo *Index.cshtml*

```
@{
    ViewData["Title"] = "Login";
}

<div class="titulo_pagina">
    <h1>@ViewData["Title"]</h1>
</div>

<form method="POST" action='@Url.Action("Logar")' class="cadastro">

    <div class="campo">
        <label>Email</label>
        <input type="text" name="Email" required />
    </div>

    <div class="campo">
        <label>Senha</label>
        <input type="password" name="Senha" required />
    </div>

    <button class="gradient btn" type="submit">Entrar</button>
</form>
```

> Também criamos dentro da pasta Controllers a classe ***LoginController.cs*** , fazemos ela herdar de *Controller*, deixamos a Route no topo da classe como Login, e chamamos um model de Jogador para acessar as propriedades *Email* e *Senha*

```
using E_Players_AspNETCore.Models;
using Microsoft.AspNetCore.Mvc;

namespace E_Players_AspNETCore.Controllers
{
    [Route("Login")]
    public class LoginController : Controller
    {
        Jogador jogadorModel = new Jogador();

        public IActionResult Index()
```

```
        {
            return View();
        }
    }
}
```

> Dentro do menu principal colocamos um link para a tela de login:

```
<li><a asp-controller="Login" asp-action="Index" title="voltar para a página i
```

> Criamos o método para Logar e testamos se está redirecionando para a Home ao enviar o formulário:

```
        [Route("Logar")]
        public IActionResult Logar(IFormCollection form)
        {
            return LocalRedirect("~/");
        }
```

> Criamos no CSS uma classe para os erros:

```
.erro{
    color: #fc1e8b;
}
```

> Dentro da classe login, utilizamos o recurso TempData para armazenar as mensagens de erro na aplicação:

```
[TempData]
public string Mensagem { get; set; }
```

> Criamos o método Logar utilizando todos os parâmetros necessários

```
public IActionResult Logar(IFormCollection form)
{
    // Lemos todos os arquivos do CSV
    List<string> csv = jogadorModel.ReadAllLinesCSV("Database/Jogador.csv");

    // Verificamos se as informações passadas existe na lista de string
    var logado =
    csv.Find(
        x =>
        x.Split(";")[2] == form["Email"] &&
        x.Split(";")[3] == form["Senha"]
```

```
        );


        // Redirecionamos o usuário logado caso encontrado
        if(logado != null)
        {
            return LocalRedirect("~/");
        }

        Mensagem = "Dados incorretos, tente novamente...";
        return LocalRedirect("~/Login");
    }
```

> Dentro da div de erro chamamos nossa TempData

```
<div class="erro">@TempData.Peek("Mensagem")</div>
```

> Vamos salvar os dados na sessão do navegador utilizando a documentação da microsoft: https://docs.microsoft.com/pt-br/aspnet/core/fundamentals/app-state?view=aspnetcore-5.0

> Em Startup.cs, no método de serviços adicionamos:

```
        services.AddDistributedMemoryCache();

        services.AddSession(options =>
        {
            options.IdleTimeout = TimeSpan.FromHours(2);
            options.Cookie.HttpOnly = true;
            options.Cookie.IsEssential = true;
        });
```

> No método *Configure* adicione o seguinte método, em cima de *app.UseEndpoints*:

```
app.UseSession();
```

> Dentro de *logado* , antes de retornar o login com sucesso, salvamos os dados do usuário em uma Sessão:

```
if(logado != null)
{
    // Definimos os valores a serem salvos na sessão
    HttpContext.Session.SetString("_UserName", logado.Split(";")[1]);
```

```
    return LocalRedirect("~/");
}
```

> Na Index da HomeController, colocamos uma ViewBag, pegando as informações da sessão:

```
ViewBag.UserName = HttpContext.Session.GetString("_UserName");
```

> No menu principal no **_Layout.cshtml**, incrementamos a seguinte condicional:

```
@if(ViewBag.UserName != null)
{
    <li style="color: #d5379b">Olá @ViewBag.UserName!</li>
    <li><a asp-controller="Login" asp-action="Logout" title="voltar para a pág
}else{

    <li><a asp-controller="Login" asp-action="Index" title="voltar para a pági
}
```

> Fazemos o método de Logout:

```
[Route("Logout")]
public IActionResult Logout()
{
    HttpContext.Session.Remove("_UserName");
    return LocalRedirect("~/");
}
```