Assessment

Disciplina: Desenvolvimento Web com .NET e Base de Dados

Professor: Orlando Fonseca

Aluno: Lucas de Castro Souza

Data: 20/06/2025

Models

Customer

```
namespace WebProjectAT.Models
{
    public class Customer
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Email { get; set; }
        public List<Reserve> Reservations { get; set; } = new List<Reserve>();
    }
}
```

Destiny

```
namespace WebProjectAT.Models
{
    public class Destiny
    {
        public int Id { get; set; }

        [BindProperty]
        [Required(ErrorMessage = "Destiny name is required")]
        [MinLength(3, ErrorMessage = "Destiny name must have at least 3 characters")]
        public string Name { get; set; }

        [BindProperty]
        [Required(ErrorMessage = "Country name is required")]
        public string Country { get; set; }

        public DateTime? DeletedAt { get; set; } = null;
    }
}
```

Reserve

```
namespace WebProjectAT.Models
{
    public class Reserve
    {
        delegate decimal CalculateDiscount(int y, decimal x);

        public int Id { get; set; }
        public int CustomerId { get; set; }
        public Customer Customer { get; set; }
        public int NumberOfNights { get; set; }
        public DateTime CreatedAt { get; set; } = DateTime.Now;
        public int TourPackageId { get; set; }
        public TourPackage TourPackage { get; set; }

        public decimal dailyPrice() {
            CalculateDelegate calculateDelegate = new CalculateDelegate();
            CalculateDiscount calculateDiscount = calculateDelegate.ApplyDiscount;

            decimal dailyPrice = calculateDiscount(NumberOfNights, TourPackage.DailyPrice);
            return dailyPrice;
        }

        public decimal totalPrice() {
            Func<decimal, int, decimal> reservePriceCalculator = (nightPrice, nights) => (decimal) nights * nightPrice;
            decimal totalPrice = reservePriceCalculator(dailyPrice(), NumberOfNights);
            return totalPrice;
        }
    }
}
```

TourPackage

```
namespace WebProjectAT.Models
{
    public class TourPackage
    {
        private CapacityObserver observer = new CapacityObserver();

        public int Id { get; set; }
        public string Title { get; set; }
        public DateTime StartDate { get; set; }
        public int MaxCapacity { get; set; }
        public decimal DailyPrice { get; set; }
        public List<Destiny> Destinations { get; set; } = new List<Destiny>();
        public List<Reserve> Reservations { get; set; } = new List<Reserve>();

        public TourPackage() {
            observer.CapacityReached += OnCapacityReachedHandler;
        }


        public bool hasReachedMaxCapacity()
        {
            if (Reservations.Count >= MaxCapacity)
            {
                observer.StartEvent();
                return true;
            }
            else {
                return false;
            }
        }

        static void OnCapacityReachedHandler(object sender, EventArgs e) {
            Console.WriteLine("ALERTA!! O número máximo de reservas do pacote foi atingido!");
        }
    }
}
```

Note

```
namespace WebProjectAT.Models
{
    public class Note
    {
        [Required(ErrorMessage = "Title is required")]
        [MinLength(3, ErrorMessage = "Title must have at least 3 characters")]
        public String Title { get; set; }

        [Required(ErrorMessage = "Note is required")]
        [MinLength(5, ErrorMessage = "Note must have at least 5 characters")]
        public String Message { get; set; }
    }
}
```

Data

WebProjectContext

```
namespace WebProjectAT.Data
{
    public class WebProjectContext: DbContext
    {
        public WebProjectContext(DbContextOptions<WebProjectContext> options) : base(options) { }

        public DbSet<Customer> Customers => Set<Customer>();
        public DbSet<Destiny> Destinations => Set<Destiny>();
        public DbSet<Reserve> Reservations => Set<Reserve>();
        public DbSet<TourPackage> TourPackages => Set<TourPackage>();

        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            modelBuilder.ApplyConfiguration(new CustomerConfiguration());
            modelBuilder.ApplyConfiguration(new DestinyConfiguration());
            modelBuilder.ApplyConfiguration(new ReserveConfiguration());
            modelBuilder.ApplyConfiguration(new TourPackageConfiguration());
        }
    }
}
```

CustomerConfiguration

```
namespace WebProjectAT.Data.Configurations
{
    public class CustomerConfiguration : IEntityTypeConfiguration<Customer>
    {
        public void Configure(EntityTypeBuilder<Customer> builder)
        {
            builder.HasKey(customer => customer.Id);

            builder.Property(customer => customer.Email)
                .IsRequired()
                .HasMaxLength(300);

            builder.HasData(
                new Customer { Id = 1, Name = "Alice Johnson", Email = "alice.j@example.com" },
                new Customer { Id = 2, Name = "Bob Williams", Email = "bob.w@example.com" },
                new Customer { Id = 3, Name = "Carol Davis", Email = "carol.d@example.com" },
                new Customer { Id = 4, Name = "David Brown", Email = "david.b@example.com" }
            );
        }
    }
}
```

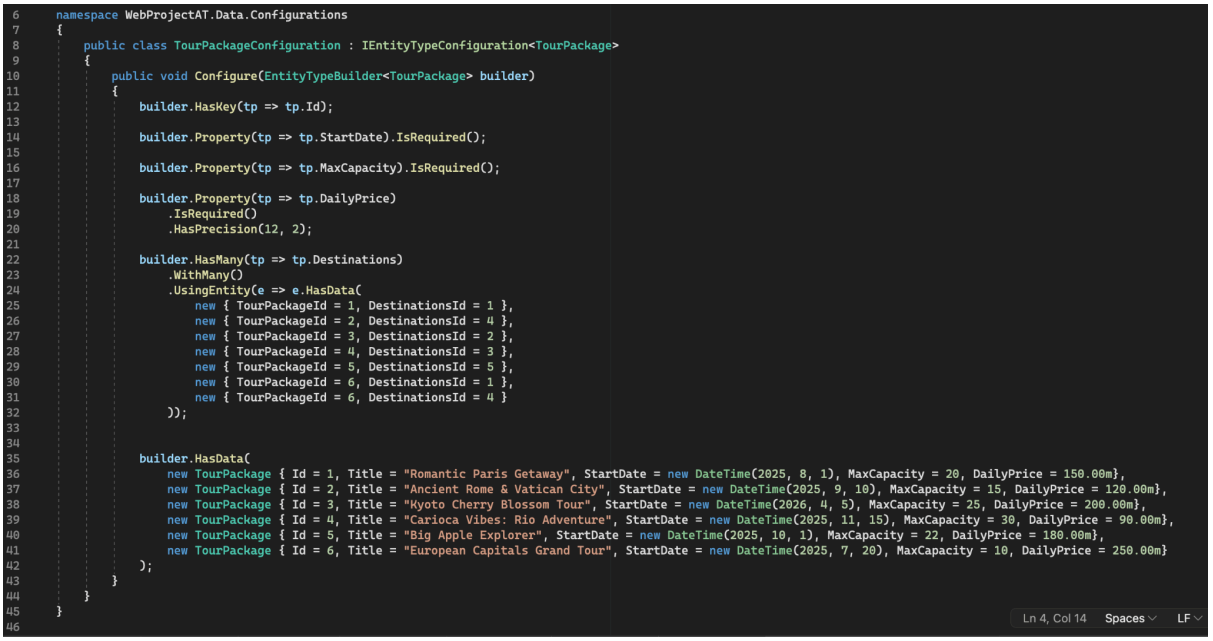
DestinyConfiguration

```
namespace WebProjectAT.Data.Configurations
{
    public class DestinyConfiguration : IEntityTypeConfiguration<Destiny>
    {
        public void Configure(EntityTypeBuilder<Destiny> builder)
        {
            builder.HasKey(destiny => destiny.Id);

            builder.Property(destiny => destiny.Name).IsRequired();

            builder.Property(destiny => destiny.Country).IsRequired();

            builder.HasData(
                new Destiny { Id = 1, Name = "Paris", Country = "France" },
                new Destiny { Id = 2, Name = "Kyoto", Country = "Japan" },
                new Destiny { Id = 3, Name = "Rio de Janeiro", Country = "Brazil" },
                new Destiny { Id = 4, Name = "Rome", Country = "Italy" },
                new Destiny { Id = 5, Name = "New York", Country = "USA" }
            );
        }
    }
}
```

ReserveConfiguration

```
namespace WebProjectAT.Data.Configurations
{
    public class ReserveConfiguration : IEntityTypeConfiguration<Reserve>
    {
        public void Configure(EntityTypeBuilder<Reserve> builder)
        {
            builder.HasKey(reserve => reserve.Id);

            builder.Property(reserve => reserve.NumberOfNights)
                .IsRequired()
                .HasDefaultValue(1);

            builder.Property(reserve => reserve.CreatedAt)
                .IsRequired();

            builder.HasOne(reserve => reserve.Customer)
                .WithMany(customer => customer.Reservations)
                .HasForeignKey(reserve => reserve.CustomerId)
                .OnDelete(DeleteBehavior.Restrict);

            builder.HasOne(reserve => reserve.TourPackage)
                .WithMany(tp => tp.Reservations)
                .HasForeignKey(reserve => reserve.TourPackageId)
                .OnDelete(DeleteBehavior.Restrict);

            builder.HasData(
                new Reserve { Id = 1, CustomerId = 1, NumberOfNights = 7, CreatedAt = DateTime.Now.AddDays(-30), TourPackageId = 1},
                new Reserve { Id = 2, CustomerId = 2, NumberOfNights = 5, CreatedAt = DateTime.Now.AddDays(-25), TourPackageId = 2},
                new Reserve { Id = 3, CustomerId = 1, NumberOfNights = 10, CreatedAt = DateTime.Now.AddDays(-15), TourPackageId = 3},
                new Reserve { Id = 4, CustomerId = 3, NumberOfNights = 4, CreatedAt = DateTime.Now.AddDays(-10), TourPackageId = 4},
                new Reserve { Id = 5, CustomerId = 4, NumberOfNights = 7, CreatedAt = DateTime.Now.AddDays(-5), TourPackageId = 1},
                new Reserve { Id = 6, CustomerId = 2, NumberOfNights = 3, CreatedAt = DateTime.Now.AddDays(-2), TourPackageId = 5}
            );
        }
    }
}

```

TourPackageConfiguration



```
namespace WebProjectAT.Data.Configurations
{
    public class TourPackageConfiguration : IEntityTypeConfiguration<TourPackage>
    {
        public void Configure(EntityTypeBuilder<TourPackage> builder)
        {
            builder.HasKey(tp => tp.Id);

            builder.Property(tp => tp.StartDate).IsRequired();

            builder.Property(tp => tp.MaxCapacity).IsRequired();

            builder.Property(tp => tp.DailyPrice)
                .IsRequired()
                .HasPrecision(12, 2);

            builder.HasMany(tp => tp.Destinations)
                .WithMany()
                .UsingEntity(e => e.HasData(
                    new { TourPackageId = 1, DestinationsId = 1 },
                    new { TourPackageId = 2, DestinationsId = 4 },
                    new { TourPackageId = 3, DestinationsId = 2 },
                    new { TourPackageId = 4, DestinationsId = 3 },
                    new { TourPackageId = 5, DestinationsId = 5 },
                    new { TourPackageId = 6, DestinationsId = 1 },
                    new { TourPackageId = 6, DestinationsId = 4 }
                ));


            builder.HasData(
                new TourPackage { Id = 1, Title = "Romantic Paris Getaway", StartDate = new DateTime(2025, 8, 1), MaxCapacity = 20, DailyPrice = 150.00m},
                new TourPackage { Id = 2, Title = "Ancient Rome & Vatican City", StartDate = new DateTime(2025, 9, 10), MaxCapacity = 15, DailyPrice = 120.00m},
                new TourPackage { Id = 3, Title = "Kyoto Cherry Blossom Tour", StartDate = new DateTime(2026, 4, 5), MaxCapacity = 25, DailyPrice = 200.00m},
                new TourPackage { Id = 4, Title = "Carioca Vibes: Rio Adventure", StartDate = new DateTime(2025, 11, 15), MaxCapacity = 30, DailyPrice = 90.00m},
                new TourPackage { Id = 5, Title = "Big Apple Explorer", StartDate = new DateTime(2025, 10, 1), MaxCapacity = 22, DailyPrice = 180.00m},
                new TourPackage { Id = 6, Title = "European Capitals Grand Tour", StartDate = new DateTime(2025, 7, 20), MaxCapacity = 10, DailyPrice = 250.00m}
            );
        }
    }
}
```

Ln 4, Col 14 Spaces LF

Delegates

CalculateDelegate

```
namespace WebProjectAT.Common.Delegates
{
    public class CalculateDelegate
    {
        public decimal ApplyDiscount(int numberOfNights, decimal price) {
            if (numberOfNights > 3) {
                return price * (decimal) 0.9;
            } else {
                return price;
            }
        }
    }
}
```

CreateReserveDelegate

```
namespace WebProjectAT.Common.Delegates
{
    public class CreateReserveDelegate
    {
        public Reserve createReserve(Reserve newReserve) {
            Reserve createdReserve = null;

            Action<string> logMessage;
            Logger logger = new Logger();

            logMessage = logger.LogToConsole;
            logMessage += logger.LogToDatabase;
            logMessage += logger.LogToFile;

            // TODO: Lógica para criar reserva no banco de dados reserva

            if (createdReserve != null)
            {
                logMessage("Reserva " + createdReserve.Id);
            }
            else {
                logMessage("Ocorreu um erro, Reserva " + newReserve.Id + " não");
            }


            return createdReserve;
        }
    }
}
```

Logger

```
namespace WebProjectAT.Common.Delegates
{
    public class Logger
    {
        public void LogToConsole(string description)
        {
            Console.WriteLine(description + " foi registrada no Console");
        }

        public void LogToFile(string description)
        {
            Console.WriteLine(description + " foi registrada em Arquivo");
        }

        public void LogToDatabase(string description)
        {
            Console.WriteLine(description + " foi registrada no Banco de Dados");
        }
    }
}
```

Events

CapacityObserver

```
namespace WebProjectAT.Common.Events
{
    public class CapacityObserver
    {
        public event EventHandler CapacityReached;

        protected virtual void OnCapacityReached()
        {
            CapacityReached?.Invoke(this, EventArgs.Empty);
        }

        public void StartEvent() {
            OnCapacityReached();
        }
    }
}
```

Program.cs

```
var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddRazorPages();

builder.Services.AddDbContext<WebProjectContext>(options =>
  options.UseSqlite(builder.Configuration.GetConnectionString("DefaultConnection")));

builder.Services.AddAuthentication(CookieAuthenticationDefaults.AuthenticationScheme)
    .AddCookie( options => {
        options.LoginPath = "/login";
    });

var app = builder.Build();

// Configure the HTTP request pipeline.
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Error");
    // The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.
    app.UseHsts();
}

using (var scope = app.Services.CreateScope())
{
    var context = scope.ServiceProvider.GetRequiredService<WebProjectContext>();
    context.Database.Migrate();
}

app.UseHttpsRedirection();
app.UseStaticFiles();

app.UseRouting();

app.UseAuthentication();
app.UseAuthorization();

app.MapRazorPages();

app.Run();
```

Pages and PageModels

Login

```razor
@page
@model WebProjectAT.Pages.LoginModel
@{
}

<div class="col-4">
    <form method="post">
        <div class="mb-3">
            <label class="form-label" asp-for="UserName"></label>
            <input class="form-control" asp-for="UserName" />
        </div>
        <div class="mb-3">
            <label class="form-label" asp-for="Password"></label>
            <input class="form-control" asp-for="Password" />
        </div>
        <div class="mb-3">
            <button class="btn btn-outline-primary">Sign in</button>
        </div>
    </form>
</div>
```

```razor
namespace WebProjectAT.Pages
{
    public class LoginModel : PageModel
    {
        [BindProperty, Display(Name = "User name")]
        public string UserName { get; set; }

        [BindProperty]
        public string Password { get; set; }

        public string ErrorMessage { get; set; }

        public async Task<IActionResult> OnPostAsync(string returnUrl = null)
        {
            var claims = new List<Claim>
            {
                new Claim(ClaimTypes.Name, UserName)
            };

            if (UserName == "admin" && Password == "admin123")
            {
                claims.Add(new Claim(ClaimTypes.Role, "Admin"));
            }
            else if (UserName == "user" && Password == "user123")
            {
                claims.Add(new Claim(ClaimTypes.Role, "User"));
            }
            else
            {
                ErrorMessage = "Usuário ou senha inválidos.";
                return Page();
            }

            var identity = new ClaimsIdentity(claims, CookieAuthenticationDefaults.AuthenticationScheme);
            var principal = new ClaimsPrincipal(identity);

            await HttpContext.SignInAsync(principal);

            if (string.IsNullOrEmpty(returnUrl))
                return RedirectToPage("Index");

            return LocalRedirect(returnUrl);
        }
    }
}
```

Logout

```razor
@page
@model WebProjectAT.Pages.LogoutModel
@{
    Layout = null;
}
```

```
namespace WebProjectAT.Pages
{
    public class LogoutModel : PageModel
    {
        public async Task<IActionResult> OnGetAsync()
        {
            await HttpContext.SignOutAsync();
            return RedirectToPage("/Login");
        }
    }
}
```

ViewNotes

```
@page
@model WebProjectAT.Pages.ViewNotesModel
@{
}

<div class="text-center">
    <h1 class="display-4">Notes</h1>
</div>

<form method="post">
    <div class="form-group">
        <label asp-for="NewNote.Title" class="control-label"></label>
        <input asp-for="NewNote.Title" class="form-control" />
        <span asp-validation-for="NewNote.Title" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="NewNote.Message" class="control-label">Add Note</label>
        <textarea asp-for="NewNote.Message" class="form-control"></textarea>
        <span asp-validation-for="NewNote.Message" class="text-danger"></span>
    </div>
    <br />
    <button type="submit" class="btn btn-primary">Submit</button>
</form>

<br/>
```

```
<table class="table">
    <thead>
        <tr>
            <th>
                File
            </th>
            <th>
                Note
            </th>
            <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach (var note in Model.Notes)
        {
            <tr>
                <td>
                    @Html.DisplayFor(modelItem => note.Title)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => note.Message)
                </td>
            </tr>
        }
    </tbody>
</table>

@section Scripts {
    <partial name="_ValidationScriptsPartial" />
}
```

CreateDestiny

```razor
@page
@model WebProjectAT.Pages.Destinations.CreateModel

@{
    ViewData["Title"] = "CreateDestiny";
}

<h1>Create</h1>

<h4>Destiny</h4>
<hr />
<div class="row">
    <div class="col-md-4">
        <form method="post">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="Destiny.Name" class="control-label"></label>
                <input asp-for="Destiny.Name" class="form-control" />
                <span asp-validation-for="Destiny.Name" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Destiny.Country" class="control-label"></label>
                <input asp-for="Destiny.Country" class="form-control" />
                <span asp-validation-for="Destiny.Country" class="text-danger"></span>
            </div>
            <br />
            <div class="form-group">
                <input type="submit" value="Create" class="btn btn-primary" />
                <a asp-page="Index" class="btn btn-danger">Cancel</a>
            </div>
        </form>
    </div>
</div>

@section Scripts {
    @{await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}
```

```razor
namespace WebProjectAT.Pages.Destinations
{
    [Authorize(Roles = "Admin")]
    public class CreateModel : PageModel
    {
        private readonly WebProjectAT.Data.WebProjectContext _context;

        public CreateModel(WebProjectAT.Data.WebProjectContext context)
        {
            _context = context;
        }

        public IActionResult OnGet()
        {
            return Page();
        }

        [BindProperty]
        public Destiny Destiny { get; set; } = default!;

        public async Task<IActionResult> OnPostAsync()
        {
          if (!ModelState.IsValid || _context.Destinations == null || Destiny == null)
            {
                return Page();
            }

            _context.Destinations.Add(Destiny);
            await _context.SaveChangesAsync();

            return RedirectToPage("./Index");
        }
    }
}
```

Delete

```razor
@page
@model WebProjectAT.Pages.Destinations.DeleteModel

@{
    ViewData["Title"] = "Delete";
}

<h1>Delete Destiny</h1>

<h4>Double check the destiny and muke sure you are deleting the right one!</h4>
<div>
    <hr />
    <dl class="row">
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Destiny.Name)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Destiny.Name)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Destiny.Country)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Destiny.Country)
        </dd>
    </dl>

    <form method="post">
        <input type="hidden" asp-for="Destiny.Id" />
        <a asp-page="./Index" class="btn btn-primary">&#8592; Back to List</a>
        <input type="submit" value="Delete" class="btn btn-danger" />
    </form>
</div>
```

```razor
namespace WebProjectAT.Pages.Destinations
{
    [Authorize(Roles = "Admin")]
    public class DeleteModel : PageModel
    {
        private readonly WebProjectAT.Data.WebProjectContext _context;

        public DeleteModel(WebProjectAT.Data.WebProjectContext context)
        {
            _context = context;
        }

        [BindProperty]
        public Destiny Destiny { get; set; } = default!;

        public async Task<IActionResult> OnGetAsync(int? id)
        {
            if (id == null || _context.Destinations == null)
            {
                return NotFound();
            }

            var destiny = await _context.Destinations.FirstOrDefaultAsync(m => m.Id == id);

            if (destiny == null)
            {
                return NotFound();
            }
            else
            {
                Destiny = destiny;
            }
            return Page();
        }
```

```razor

        public async Task<IActionResult> OnPostAsync(int? id)
        {
            if (id == null || _context.Destinations == null)
            {
                return NotFound();
            }
            var destiny = await _context.Destinations.FindAsync(id);

            if (destiny != null && destiny.DeletedAt == null)
            {
                destiny.DeletedAt = DateTime.UtcNow;
                await _context.SaveChangesAsync();
            }

            return RedirectToPage("./Index");
        }
    }
}
```

Edit

```razor
@page
@model WebProjectAT.Pages.Destinations.EditModel

@{
    ViewData["Title"] = "Edit Destiny";
}

<h1>Edit Destiny</h1>
<hr />
<div class="row">
    <div class="col-md-4">
        <form method="post">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <input type="hidden" asp-for="Destiny.Id" />
            <div class="form-group">
                <label asp-for="Destiny.Name" class="control-label"></label>
                <input asp-for="Destiny.Name" class="form-control" />
                <span asp-validation-for="Destiny.Name" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Destiny.Country" class="control-label"></label>
                <input asp-for="Destiny.Country" class="form-control" />
                <span asp-validation-for="Destiny.Country" class="text-danger"></span>
            </div>
            <br/>
            <div class="form-group">
                <input type="submit" value="Save" class="btn btn-primary" />
                <a asp-page="./Index" class="btn btn-danger">Cancel</a>
            </div>
        </form>
    </div>
</div>

@section Scripts {
    @await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}
```

```razor
namespace WebProjectAT.Pages.Destinations
{
    [Authorize(Roles = "Admin")]
    public class EditModel : PageModel
    {
        private readonly WebProjectAT.Data.WebProjectContext _context;

        public EditModel(WebProjectAT.Data.WebProjectContext context)
        {
            _context = context;
        }

        [BindProperty]
        public Destiny Destiny { get; set; } = default!;

        public async Task<IActionResult> OnGetAsync(int? id)
        {
            if (id == null || _context.Destinations == null)
            {
                return NotFound();
            }

            var destiny =  await _context.Destinations.FirstOrDefaultAsync(m => m.Id == id);
            if (destiny == null)
            {
                return NotFound();
            }
            Destiny = destiny;
            return Page();
        }
```

```
        public async Task<IActionResult> OnPostAsync()
        {
            if (!ModelState.IsValid)
            {
                return Page();
            }

            _context.Attach(Destiny).State = EntityState.Modified;

            try
            {
                await _context.SaveChangesAsync();
            }
            catch (DbUpdateConcurrencyException)
            {
                if (!DestinyExists(Destiny.Id))
                {
                    return NotFound();
                }
                else
                {
                    throw;
                }
            }

            return RedirectToPage("./Index");
        }

        private bool DestinyExists(int id)
        {
          return (_context.Destinations?.Any(e => e.Id == id)).GetValueOrDefault();
        }
    }
}
```

Details

```
@page "{destinyId}"
@model WebProjectAT.Pages.Destinations.DetailsModel

@{
    ViewData["Title"] = "Details";
}

<h1>Destiny Details</h1>

<div>
    <hr />
    <dl class="row">
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Destiny.Name)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Destiny.Name)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Destiny.Country)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Destiny.Country)
        </dd>
    </dl>
</div>
<div>
    <a asp-page="./Index" class="btn btn-primary">&#8592; Back to List</a>
    <a asp-page="./Edit" asp-route-id="@Model.Destiny?.Id" class="btn btn-warning">Edit</a>
</div>
```

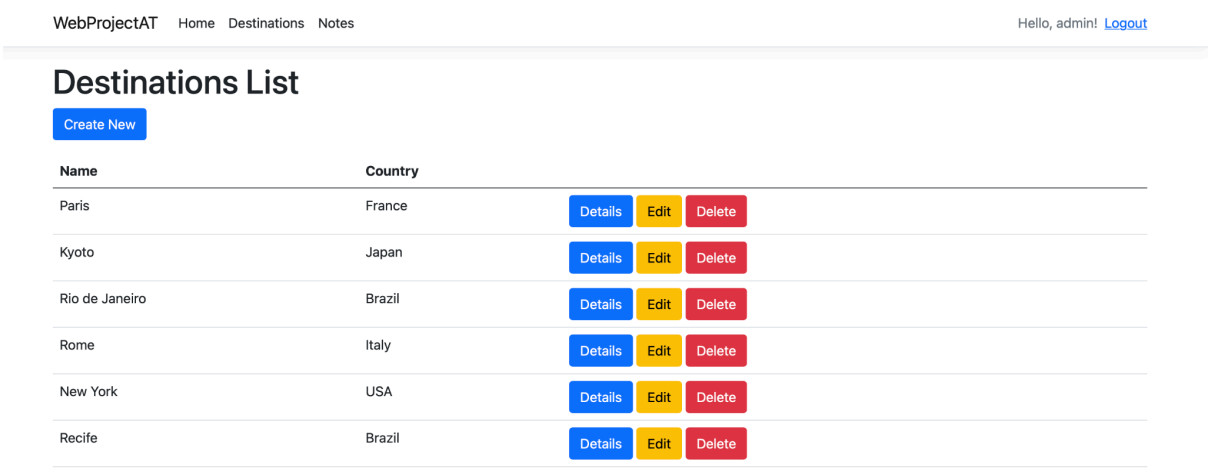
```
namespace WebProjectAT.Pages.Destinations
{
    public class DetailsModel : PageModel
    {
        private readonly WebProjectAT.Data.WebProjectContext _context;

        public DetailsModel(WebProjectAT.Data.WebProjectContext context)
        {
            _context = context;
        }

        public Destiny Destiny { get; set; } = default!;

        public async Task<IActionResult> OnGetAsync(int? destinyId)
        {
            if (destinyId == null || _context.Destinations == null)
            {
                return NotFound();
            }

            var destiny = await _context.Destinations.FirstOrDefaultAsync(m => m.Id == destinyId);
            if (destiny == null)
            {
                return NotFound();
            }
            else
            {
                Destiny = destiny;
            }
            return Page();
        }
    }
}
```

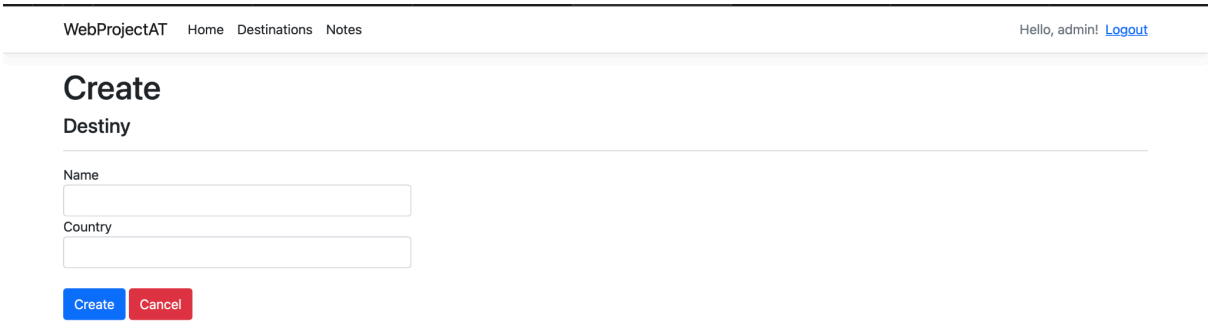
Destinations - Index

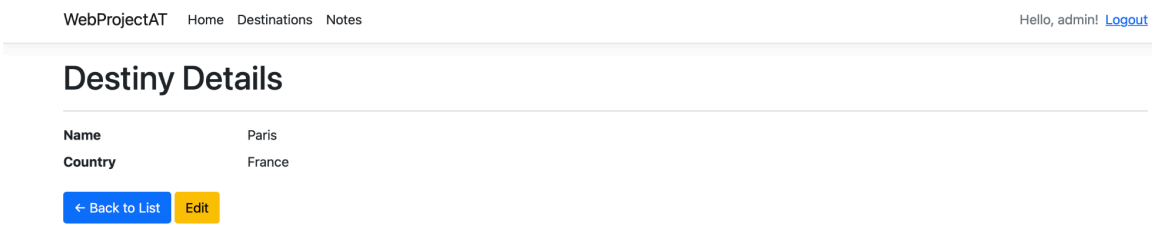
```
@page
@model WebProjectAT.Pages.Destinations.IndexModel

@{
    ViewData["Title"] = "Destinations";
}

<h1>Destinations List</h1>

<p>
    <a asp-page="CreateDestiny" class="btn btn-primary">Create New</a>
</p>
<table class="table">
    <thead>
        <tr>
            <th>
                @Html.DisplayNameFor(model => model.Destiny[0].Name)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.Destiny[0].Country)
            </th>
            <th></th>
        </tr>
    </thead>
    <tbody>
@foreach (var item in Model.Destiny) {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => item.Name)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Country)
            </td>
            <td>
                <a asp-page="./Details" asp-route-destinyId="@item.Id" class="btn btn-primary">Details</a>
                <a asp-page="./Edit" asp-route-id="@item.Id" class="btn btn-warning">Edit</a>
                <a asp-page="./Delete" asp-route-id="@item.Id" class="btn btn-danger">Delete</a>
            </td>
        </tr>
}
    </tbody>
</table>
```

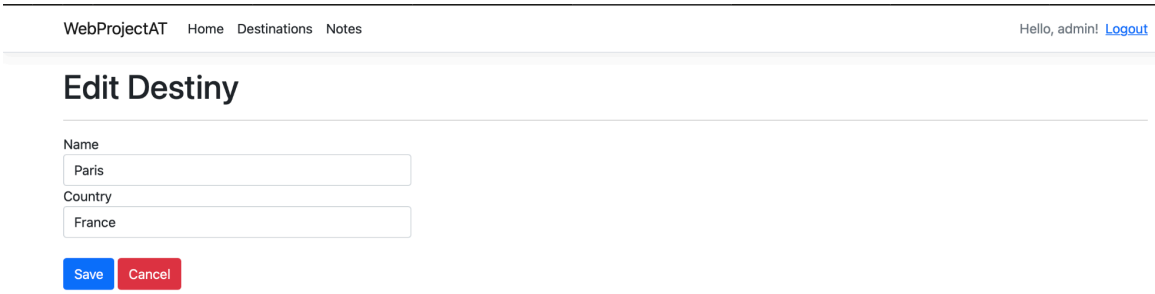
```
4
5   namespace WebProjectAT.Pages.Destinations
6   {
7       public class IndexModel : PageModel
8       {
9           private readonly WebProjectAT.Data.WebProjectContext _context;
10
11          public IndexModel(WebProjectAT.Data.WebProjectContext context)
12          {
13              _context = context;
14          }
15
16          public IList<Destiny> Destiny { get;set; } = default!;
17
18          public async Task OnGetAsync()
19          {
20              if (_context.Destinations != null)
21              {
22                  Destiny = await _context.Destinations
23                      .Where(d=> d.DeletedAt == null)
24                      .ToListAsync();
25              }
26          }
27      }
28  }
```

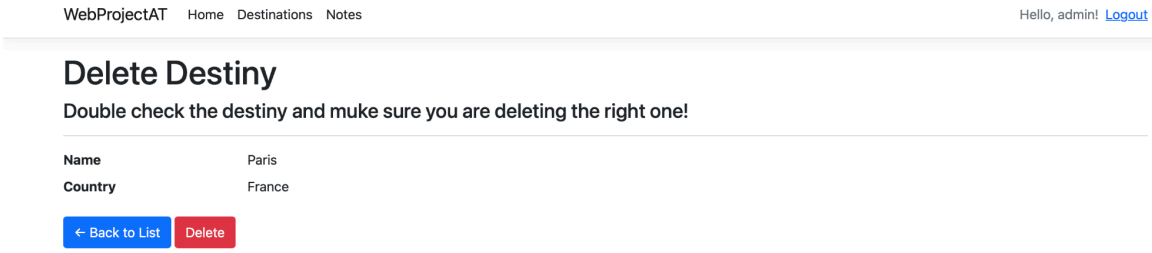
Project

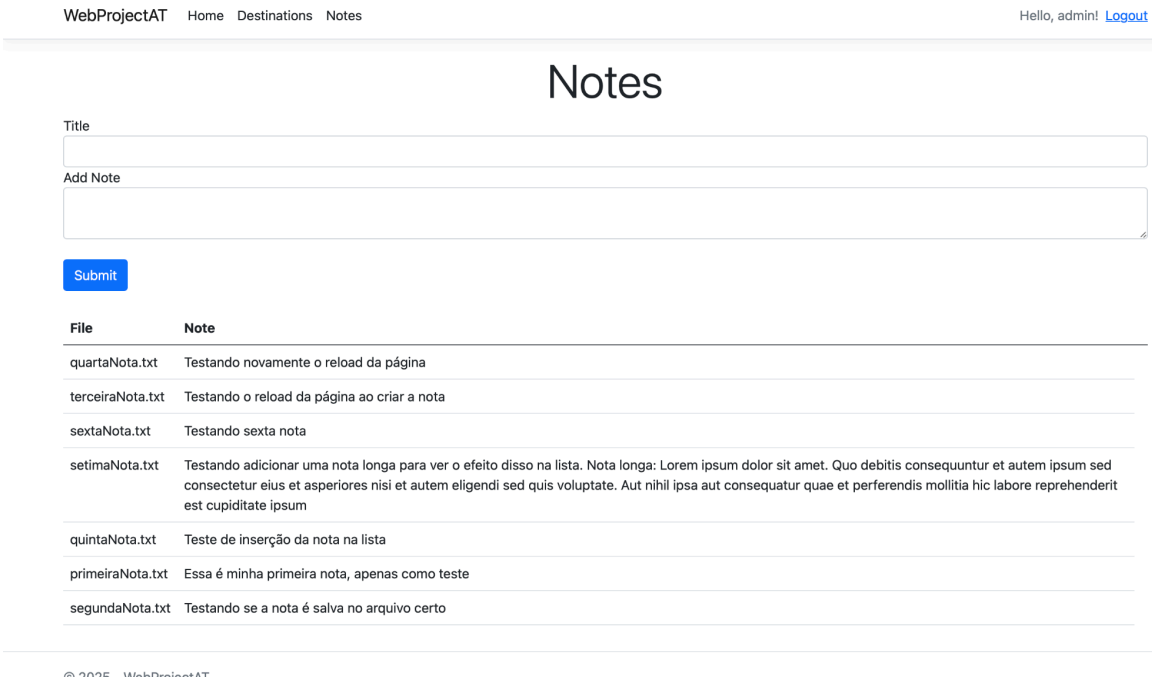


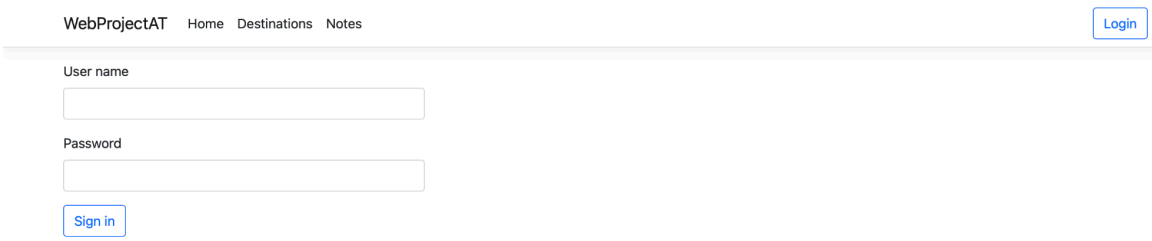



WebProjectAT
Home
Destinations
Notes
Hello, admin!
Logout
Destiny Details
Name
Paris
Country
France
← Back to List
Edit


WebProjectAT
Home
Destinations
Notes
Hello, admin!
Logout
Edit Destiny
Name
Paris
Country
France
Save
Cancel


WebProjectAT
Home
Destinations
Notes
Hello, admin!
Logout
Delete Destiny
Double check the destiny and muke sure you are deleting the right one!
Name
Paris
Country
France
← Back to List
Delete


WebProjectAT
Home
Destinations
Notes
Hello, admin!
Logout
Notes
Title
Add Note
Submit
File
Note
quartaNota.txt
Testando novamente o reload da página
terceiraNota.txt
Testando o reload da página ao criar a nota
sextaNota.txt
Testando sexta nota
setimaNota.txt
Testando adicionar uma nota longa para ver o efeito disso na lista. Nota longa: Lorem ipsum dolor sit amet. Quo debitis consequuntur et autem ipsum sed consectetur eius et asperiores nisi et autem eligendi sed quis voluptate. Aut nihil ipsa aut consequatur quae et perferendis mollitia hic labore reprehenderit est cupiditate ipsum
quintaNota.txt
Teste de inserção da nota na lista
primeiraNota.txt
Essa é minha primeira nota, apenas como teste
segundaNota.txt
Testando se a nota é salva no arquivo certo



User name

Password

Sign in