ANO
2025

Trabalho Acadêmico

Arquitetura e Desenvolvimento de APIs

V1.2

Wesllei de Brito Santos RU: 4913249

Prof. Rodrigo da S. do Nascimento

Prof. Osmar T. P. D. Junior

RELATÓRIO TÉCNICO

Sistema de Gerenciamento Empresarial – ApiEmpresa

1. Introdução

Este relatório técnico tem como objetivo descrever o desenvolvimento da **ApiEmpresa**, uma API REST criada como projeto acadêmico, voltada ao gerenciamento de informações empresariais, tais como empresas, funcionários, setores, habilidades e endereços.

O projeto foi desenvolvido utilizando **ASP.NET Core (.NET 9)**, com persistência de dados em **MySQL**, fazendo uso de boas práticas de arquitetura de software, versionamento, containerização e deploy em nuvem.

A aplicação foi publicada em ambiente de produção utilizando a infraestrutura da **Amazon Web Services (AWS)**, permitindo acesso público e validação prática do funcionamento do sistema.

2. Objetivos do Projeto

2.1 Objetivo Geral

Desenvolver uma API RESTful para gerenciamento empresarial, aplicando conceitos modernos de desenvolvimento backend, arquitetura em camadas e deploy em ambiente cloud.

2.2 Objetivos Específicos

- Implementar operações CRUD para as entidades do sistema;
- Aplicar validações e regras de negócio;
- Utilizar um ORM para mapeamento objeto-relacional;
- Containerizar a aplicação utilizando Docker;
- Realizar o deploy da aplicação em ambiente de produção na AWS.

3. Tecnologias Utilizadas

As seguintes tecnologias foram utilizadas no desenvolvimento do projeto:

- **C#**: Linguagem de programação principal;
- **ASP.NET Core Web API (.NET 9)**: Framework para construção da API;
- **Entity Framework Core**: ORM para comunicação com o banco de dados;
- **MySQL 8.x**: Sistema de gerenciamento de banco de dados relacional;
- **AutoMapper**: Mapeamento entre entidades e DTOs;
- **FluentValidation**: Validação de dados de entrada;
- **Swagger / OpenAPI**: Documentação e testes da API;
- **Docker e Docker Compose**: Containerização e orquestração dos serviços;
- **AWS EC2 (Ubuntu Server)**: Ambiente de deploy em produção.

4. Arquitetura da Aplicação

A ApiEmpresa foi desenvolvida seguindo o padrão de **arquitetura em camadas**, promovendo organização, manutenibilidade e separação de responsabilidades.

4.1 Camadas do Sistema

- **Controllers**: Responsáveis por receber as requisições HTTP e retornar as respostas adequadas;
- **DTOs (Data Transfer Objects)**: Utilizados para entrada e saída de dados, evitando exposição direta das entidades;
- **Services**: Contêm as regras de negócio da aplicação;
- **Repositories**: Responsáveis pelo acesso aos dados e comunicação com o banco;
- **Entities / Models**: Representam as tabelas do banco de dados;
- **Validators**: Aplicam validações utilizando FluentValidation;
- **Mapping**: Configurações do AutoMapper.

5. Modelagem do Banco de Dados

O banco de dados foi modelado de forma relacional, utilizando o MySQL. As principais entidades do sistema incluem:

- Empresa
- Funcionário
- Setor
- Habilidade
- Endereço

Os relacionamentos entre as entidades foram implementados utilizando o **Entity Framework Core**, e as tabelas são criadas

automaticamente através do mecanismo de **migrations**, garantindo versionamento do esquema do banco de dados.

6. Containerização da Aplicação

Para garantir portabilidade e padronização do ambiente, a aplicação foi containerizada utilizando Docker.

6.1 Dockerfile

O Dockerfile é responsável por gerar a imagem da API, contendo o runtime do .NET e o código da aplicação.

6.2 Docker Compose

O Docker Compose é utilizado para orquestrar os serviços do sistema:

- Container da API (.NET);
- Container do banco de dados MySQL;

Os containers se comunicam através de uma rede interna, e o banco de dados não é exposto publicamente, aumentando a segurança da aplicação.

7. Deploy em Ambiente Cloud (AWS)

O deploy da ApiEmpresa foi realizado utilizando uma instância **EC2** da Amazon Web Services, com sistema operacional **Ubuntu Server**.

7.1 Infraestrutura

- Tipo de instância: EC2 (Free Tier);
- Sistema operacional: Ubuntu Server 22.04 LTS;

- API exposta na porta 80;
- Banco de dados executado em container Docker, sem exposição externa.

7.2 Acesso à Aplicação

A aplicação está disponível publicamente através do seguinte endereço:

Swagger – Ambiente de Produção:

http://54.207.110.120/swagger

8. Testes e Validação

Os testes da aplicação foram realizados de forma manual, utilizando a interface do Swagger, que permite o envio de requisições HTTP e validação das respostas da API.

Os testes confirmaram o correto funcionamento dos endpoints, bem como a comunicação entre a API e o banco de dados, tanto em ambiente local quanto em produção.

9. Considerações Finais

O desenvolvimento da ApiEmpresa permitiu a aplicação prática de conceitos fundamentais de desenvolvimento backend, como arquitetura em camadas, ORM, validação de dados, containerização e deploy em nuvem.

O projeto atende aos objetivos propostos e pode ser expandido futuramente com melhorias como autenticação, autorização, testes automatizados e utilização de serviços gerenciados de banco de dados.

10. Trabalhos Futuros

- Implementação de autenticação e autorização (JWT);
- Criação de testes automatizados;
- Utilização de banco de dados gerenciado (AWS RDS);
- Implementação de CI/CD.