

Licenciatura em Engenharia Informática

Cloud Computing

Azure Web Application



45273 - Rafael Alves
45137 - Diogo Gonçalves

Resumo

O projeto final de Cloud Computing foi realizado no âmbito de aplicar os conhecimentos adquiridos em sala de aulas, como para adquirir conhecimentos adicionais.

O projeto tem como base a implementação do Portal Azure, onde o backend, frontend e SQL database pode ser escolhido e desenvolvido por nós.

Assim construímos um projeto Cloud, utilizando recursos como Function App, SQL Server, Resource Group e Storage Account.

Web Application URL → <https://webprojectcloudcompaf41.z13.web.core.windows.net/>

Tecnologias Utilizadas

Para a realização do projeto final da disciplina de Cloud Computing, foram utilizadas as seguintes tecnologias:

Backend corresponde à lógica da web application.

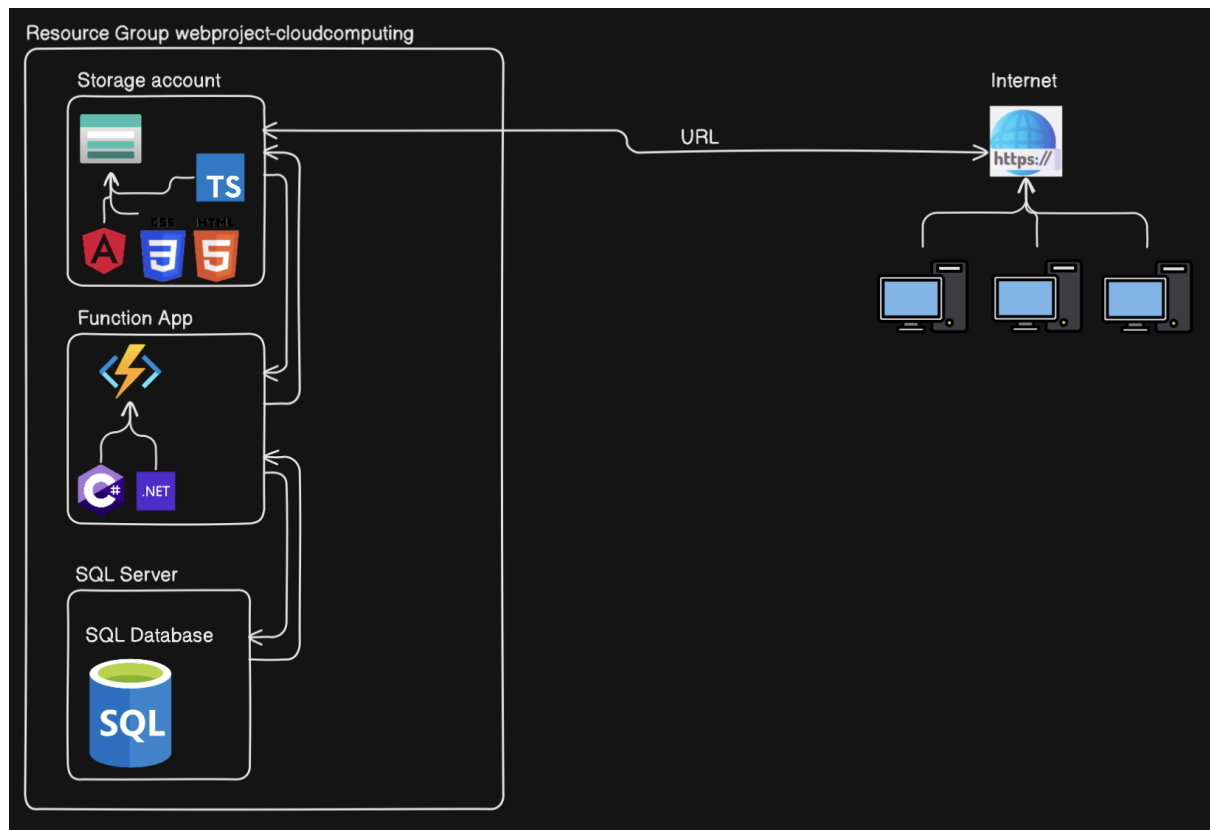
Frontend corresponde ao visual, e design da web application.

SQL corresponde a database da web application.

A **Cloud** optada para realizar a hospedagem foi o **Azure**.

	Linguagens de Programação	Serviços Azure
Backend	C# .NET8, Azure Functions	Function App
Frontend	Angular, Typescript, Html, Css	Storage Account
Database	SQL	SQL Server

Diagrama Arquitetural



No diagrama arquitetural apresentado, é possível verificar e confirmar a arquitetura da web application hospedada no Azure.

O backend foi totalmente criado em C# .NET, onde foi alocado no Azure Function App. O acesso ao backend é apenas possível a partir do frontend, garantindo assim uma maior segurança.

O frontend foi codificado em Angular, com auxílio do Typescript, Html e Css, onde foi hospedado no Azure Storage Account. Apenas o frontend é público, todos os dados e lógica necessários para o frontend, são pedidos ao backend.

O SQL database foi criado em um Azure SQL Server, a database é totalmente em SQL. A database é apenas conectada via String Connection ao backend.

Implementações

Portal Azure

1. Criação do Resource Group

The screenshot shows the Azure Portal interface for a Resource Group named 'webproject-cloudcomputing'. The 'Overview' tab is active, showing a list of resources. The table below represents the data visible in the screenshot:

Name	Type	Location
Application Insights Smart Detection	Action group	Global
ASP-webprojectcloudcomputing-a2d7	App Service plan	West Europe
backend-webproject-cloudcomp202506131720	Application Insights	West Europe

Criação de um Resource Group para acoplar todos os serviços necessários para hospedar a web application.

2. Criação do SQL Server

The screenshot shows the Azure Portal interface for an SQL server named 'projeto cardetail'. The 'Overview' tab is active, displaying a table of available resources. The table below represents the data visible in the screenshot:

Name	Type	Status	Pricing tier
cardetail-projetofinal	SQL database	Online	General Purpose: Standard-ser

Criação do SQL Server, habilitando a String Connection.

3. Configuração SQL Server

SQL databases

SQL elastic pools

Properties

Locks

Data management

Security

Networking

Microsoft Defender for Cloud

Transparent data encryption

Identity

Auditing

Intelligent performance

Rule

Virtual network

Subnet

Address range

Endpoint status

Resource group

Subscription

State

Firewall rules

Allow certain public internet IP addresses to access your resource. [Learn more](#)

+ Add your client IPv4 address (176.223.10.249) + Add a firewall rule

Rule name

Start IPv4 address

End IPv4 address

Exceptions

☒ Allow Azure services and resources to access this server

Save Discard

Configuração do SQL Server para aceitar String Connections.

4. Criação da Function App

Home >

Function App

Diretório Predefinido (rapaguel43gmail.onmicrosoft.com)

+ Create Manage view Refresh Export to CSV Open query Assign tags Start Restart Stop Delete

You are viewing a new version of Browse experience. Some features may be missing. [Click here to access the old experience.](#)

Filter for any field...

Subscription equals all Resource Group equals all Location equals all + Add filter

Name ↑	Status	Location	Pricing Tier	App Service Plan	Subscription	App Type
backend-webproject-cloudcomputing	Running	West Europe	Dynamic	ASP-webprojectcloudc...	Azure for Students	Function App

Criação da Function App.

5. Upload das Function no Azure Function App

Home > Function App >

Function App

Diretório Predefinido (rapaguel43gmail.onmicr...

+ Create ... Group by none

You are viewing a new version of Browse experience. Some features may be missing. [Click here to access the old experience.](#)

Name ↑

backend-webproject-cloudcomputing

Access control (IAM)

Tags

Diagnose and solve problems

Microsoft Defender for Cloud

Events (preview)

Recommended services (preview)

Log stream

Resource visualizer

Functions

Deployment

Settings

Performance

App Service plan

Development Tools

API

Monitoring

Automation

Support + troubleshooting

Location (move)

West Europe

Subscription (move)

Azure for Students

Subscription ID

630583c7-55f8-4766-ad17-a0bb2351a25b

Tags (edit)

Add tags

App Service Plan

ASP-webprojectcloudcomputing-a2d7f10

Runtime version

4.1040.200.25278

Functions Metrics Properties Notifications (0)

Set up local environment Refresh

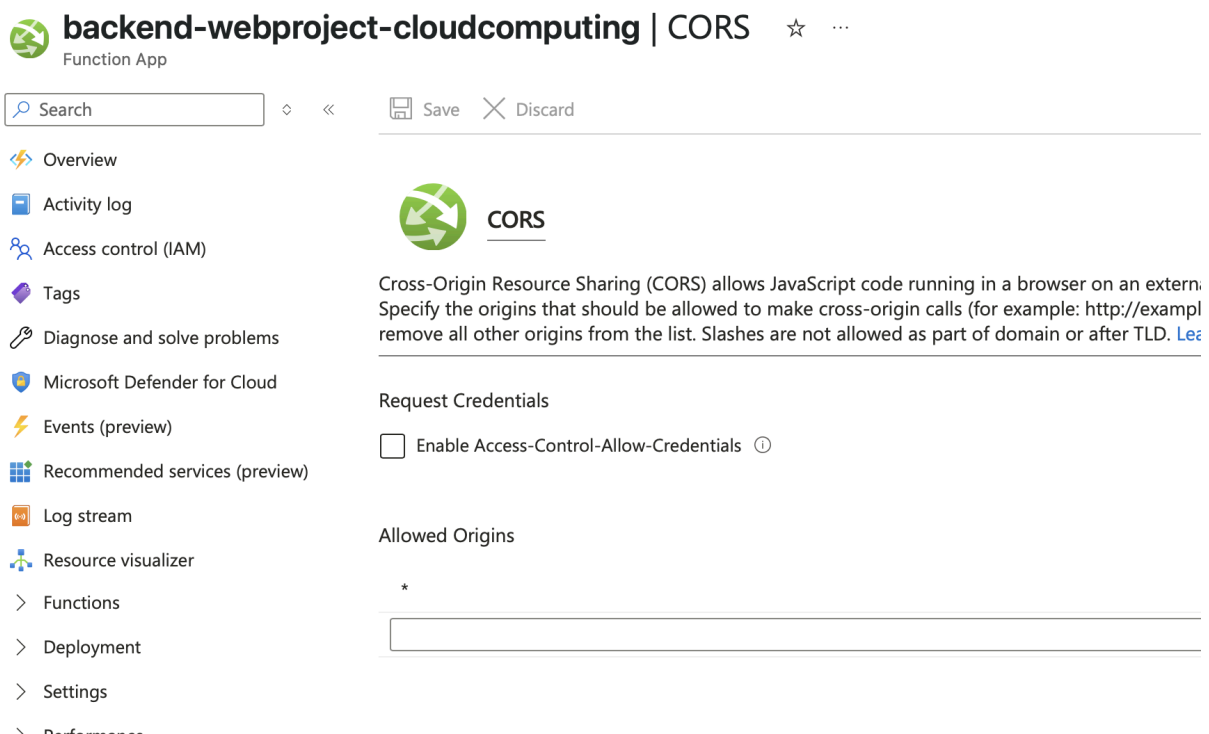
Filter by name...

Name	Trigger	Status ↓	Monitor
CreateCar	HTTP	Enabled	Invocations and more
DeleteCar	HTTP	Enabled	Invocations and more
GetCar	HTTP	Enabled	Invocations and more
UpdateCar	HTTP	Enabled	Invocations and more

As Function foram adicionadas no Azure Function App, a partir da linha de comandos, cmd. Segue então o código utilizado.

```
> brew update && brew install azure-cli //Install packages Azure CLI
> az --version //Azure CLI version
> az login //Login Azure
> func azure functionapp publish backend-webproject-cloudcomputing //Publish the Functions on project
> func azure functionapp logstream backend-webproject-cloudcomputing //Show logs in real time
```

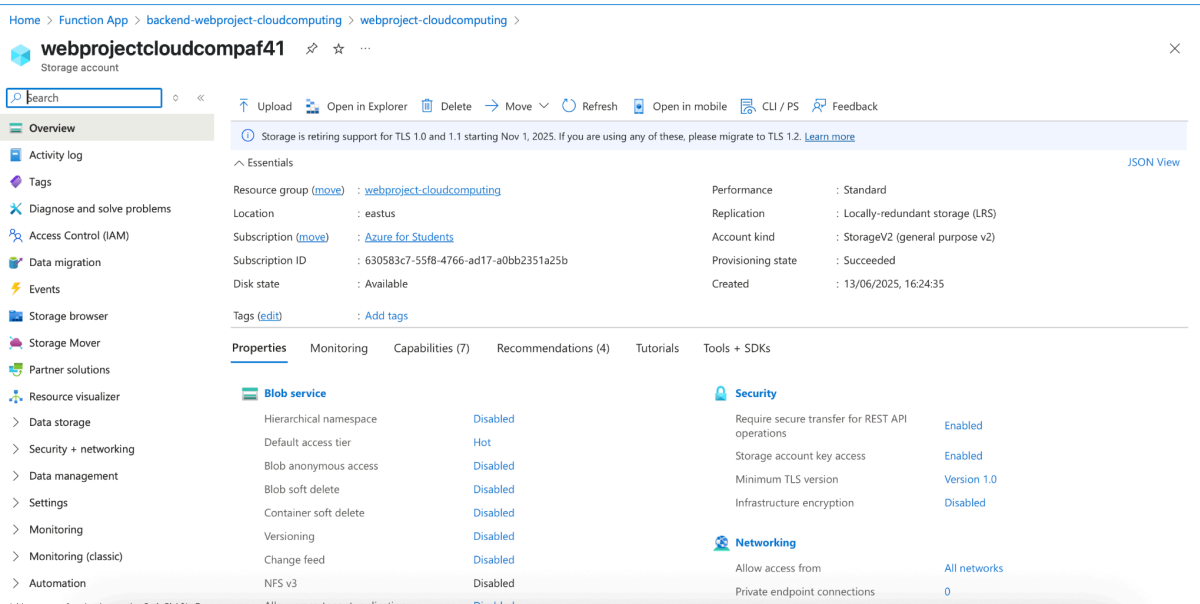
6. Configuração Function App



The screenshot displays the Azure Portal interface for configuring CORS on a Function App. The breadcrumb navigation shows 'backend-webproject-cloudcomputing | CORS'. The left-hand navigation pane includes links to Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Microsoft Defender for Cloud, Events (preview), Recommended services (preview), Log stream, Resource visualizer, Functions, Deployment, Settings, and Performance. The main content area is titled 'CORS' and includes a description: 'Cross-Origin Resource Sharing (CORS) allows JavaScript code running in a browser on an external domain to make requests to resources on another domain. Specify the origins that should be allowed to make cross-origin calls (for example: http://example.com). Remove all other origins from the list. Slashes are not allowed as part of domain or after TLD. [Learn more](#)'. Below this, there is a 'Request Credentials' section with a checkbox for 'Enable Access-Control-Allow-Credentials' which is currently unchecked. The 'Allowed Origins' section shows a single entry, '*', indicating that all origins are permitted. A text input field is visible below the list of origins.

Configuração do CORS, para ajudar nos pedidos do frontend, demos permissão * (all).

7. Criação Storage Account



Criação da Storage Account, foi realizada a partir da linha de comandos cmd. Segue então o código utilizado.

```
> az storage account create --name webprojectcloudcompaf41 --location westeurope
--resource-group webproject-cloudcomputing --sku Standard_LRS //Create Storage account Azure
> az storage blob service-properties update --account-name webprojectcloudcompaf41
--static-website --index-document index.html --404-document index.html //Enable static website
Storage account
```

Upload de o frontend no Storage Account.

```
> ng build --configuration production //Build project
> npm install -g @azure/static-web-apps-cli //Install Azure Static Web Apps CLI
> cd dist/frontend/browser //Go to directory browser
> mv index.csr.html index.html //Change name of file
```

First time upload:

```
> az storage blob upload-batch --account-name webprojectcloudcompaf41 --destination '$web'
--source . //Upload the files to container $web
```

Upgrade files in Storage Account:

```
> az storage blob upload-batch --account-name webprojectcloudcompaf41 --destination '$web'
--source . --overwrite //Upload files overwrite
> az storage account show --name webprojectcloudcompaf41 --query "primaryEndpoints.web" -o tsv
//Get the website url
```


Backend

C# Azure Functions.

- Functions:

/create/car
/get/car
/delete/car
/update/car

- Models:

CarModel DTO.

- Package:

> Microsoft.ApplicationInsights.WorkerService
> Microsoft.Azure.Functions.Worker
> Microsoft.Azure.Functions.Worker.ApplicationInsights
> Microsoft.Azure.Functions.Worker.Extensions.Http.AspNetCore
> Microsoft.Azure.Functions.Worker.Sdk
> Microsoft.EntityFrameworkCore
> Microsoft.EntityFrameworkCore.Design
> Microsoft.EntityFrameworkCore.SqlServer

Frontend

Typescript, Angular19, Css.

- Components:

CarDetailsComponent.ts

- Services:

CarDetailsService.ts

- Environment:

```
export const environment = {  
  production: true,  
  apiUrl: 'https://backend-webproject-cloudcomputing.azurewebsites.net/api/'  
}
```

- Routes:

```
export const routes: Routes = [  
  { path: '', redirectTo: 'cars', pathMatch: 'full' },  
  { path: 'cars', component: CardetailsComponent, pathMatch: 'full' },  
];
```

SQL Database

1. Criação da SQL Database

SQL

cardetail-projetofinal (projeto cardetail / cardetail-projetofinal)

SQL database

Search

Copy

Restore

Export

Set server firewall

Delete

Connect with...

Overview

Activity log

Tags

Diagnose and solve problems

Query editor (preview)

Mirror database in Fabric (preview)

Resource visualizer

Settings

Data management

Integrations

Power Platform

Security

Intelligent performance

Status

Online

Location

West US

Subscription (move)

Azure for Students

Subscription ID

630583c7-55f8-4766-ad17-a0bb2351a25b

Tags (edit)

Add tags

Elastic pool

No elastic pool

Connection strings

Show database connection strings

Pricing tier

General Purpose: Gen5, 2 vCores

Earliest restore point

2025-06-11 18:03 UTC

Getting started

Monitoring

Properties

Features

Notifications (1)

Integration

Start working with your database

Connect to your database and start working with data with a few simple steps.

Criação da SQL Database.

A Criação de Tabelas na Database foi realizada a partir de código C# e linha de comandos cmd.

Código utilizado em C#.

```
MasterDbContext.cs
1 using System.ComponentModel.DataAnnotations;
2 using System.ComponentModel.DataAnnotations.Schema;
3 using Microsoft.EntityFrameworkCore;
4 using Microsoft.EntityFrameworkCore.Metadata.Internal;
5
6 namespace backend_api.Context
7 {
8     10 references
9     public class MasterDbContext : DbContext
10     {
11         0 references
12         public MasterDbContext(DbContextOptions<MasterDbContext> options) : base(options) { }
13         5 references
14         public DbSet<Cars> Cars { get; set; }
15
16         [Table("Cars")]
17         2 references
18         public class Cars
19         {
20             [Key]
21             3 references
22             public int? Id { get; set; }
23             4 references
24             public string? Mark { get; set; }
25             4 references
26             public string? Model { get; set; }
27             4 references
28             public string? Color { get; set; }
29             4 references
30             public string? Km { get; set; }
31             4 references
32             public string? Price { get; set; }
33             4 references
34             public string? Year { get; set; }
35         }
36     }
37 }
```

Configuração da String Connection SQL Database.

```
6      },
7      "ConnectionStrings": {
8        "Master": "Server=tcp:projeto-cardetail.database.windows.net,1433;Initial Catalog=cardetail-projetofinal;Persist Security Info=True;User ID=projeto-cardetail;Password=@ProjetoCardetail123;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;
9      },
10     "Host": {
11       "CORS": "*"
12     }
13   }
```

Código utilizado para implementar e criar as Tabelas na database.

- > dotnet clean **//Clean project**
- > dotnet build **//Build project**
- > dotnet ef migration add TableCarsDetails **//Create a migrations upload on database**
- > dotnet ef database update **//Update the database**

Imagens

Projeto final Web Application.

<https://webprojectcloudcompaf41.z13.web.core.windows.net/>

The image shows a web form titled "Cadastro de Carros" (Car Registration). The form is set against a light blue background. It contains several input fields arranged in two columns. The first column has fields for "Marca" (Brand) and "Preço" (Price). The second column has fields for "Modelo" (Model), "Km", and "Ano" (Year). Each field has a placeholder text matching its label. Below the input fields are two large, dark green buttons: "Cadastrar Carro" (Register Car) and "Ver Carros" (View Cars).

The image shows a web page titled "Lista de Carros" (Car List). It features a table with the following columns: ID, Marca, Modelo, Cor, Km, Preço, Ano, and Ações. The first row of data shows a car with ID 18, Marca Toyota, Modelo Supra, Cor Azul, Km 123456, Preço 12345, and Ano 2001. The "Ações" column for this row contains two buttons: "Editar" (Edit) and "Apagar" (Delete). Above the table, there is a form with input fields for "Cor" (Color), "Km", "Preço", and "Ano", and two large green buttons: "Cadastrar Carro" and "Ver Carros".

ID	Marca	Modelo	Cor	Km	Preço	Ano	Ações
18	Toyota	Supra	Azul	123456	12345	2001	<button>Editar</button> <button>Apagar</button>

Conclusão

A realização deste projeto, permitiu adquirir conhecimentos e aprender novas formas de desenvolvimento e implementação de web application no portal Azure.

Foi possível aplicar neste projeto algumas linguagens de programação como Angular, Typescript, C#, SQL Querys, como também foi possível aplicar métodos de implementação de web applications no portal do Azure, onde foi possível utilizar Function App, Storage Account e SQL Server.

Concluindo, com a realização deste projeto foi possível, aprender, desenvolver e colocar em produção (online).