



Tópicos em Engenharia de Software

**Aplicação Web integrada aos Serviços AWS S3 e
AWS RDS**

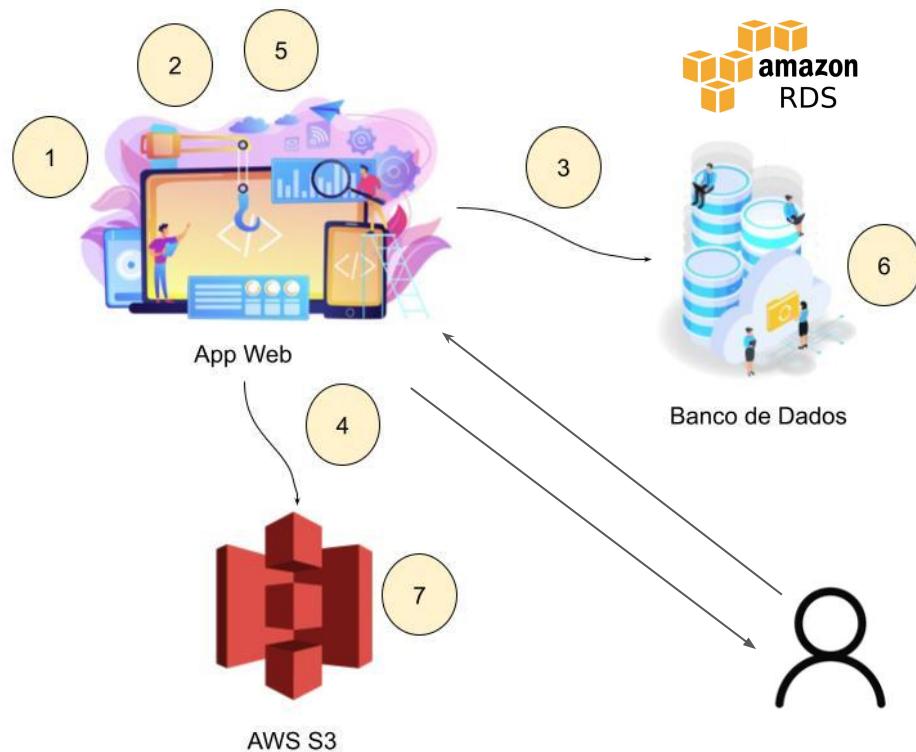
(Prova de conceito)

Armando Soares Sousa

Departamento de Computação - UFPI

armando@ufpi.edu.br

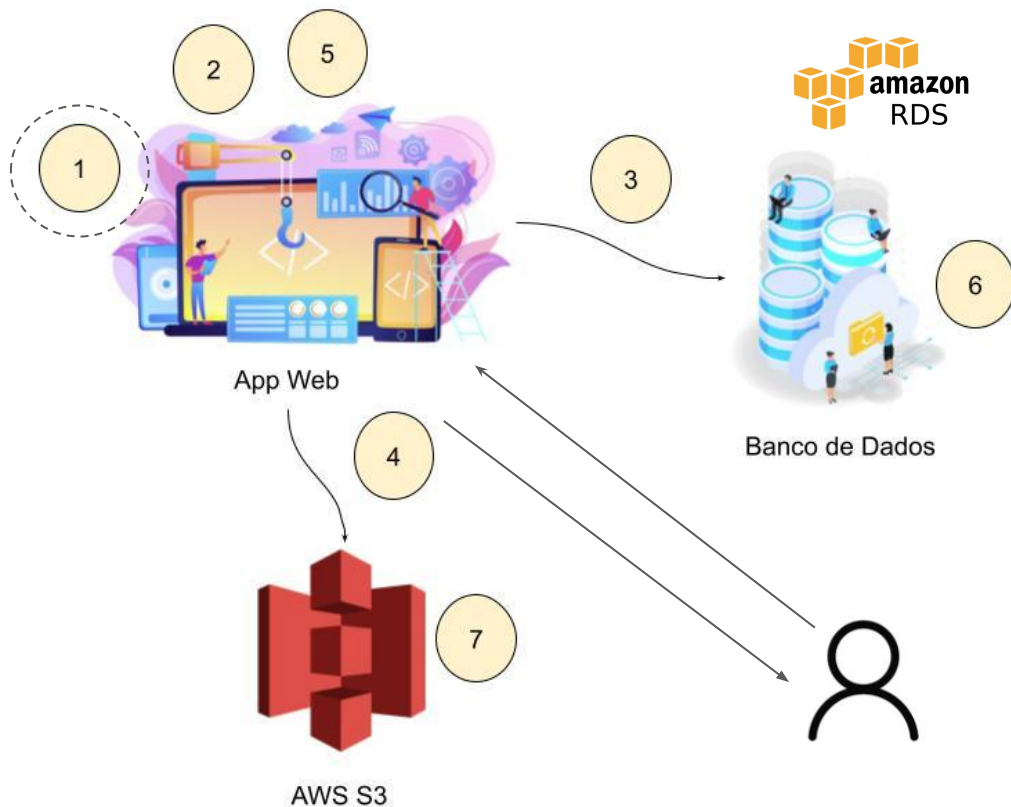
App Web com S3 e RDS



Exemplo de aplicação web que faz upload e download de arquivos.

1. Estrutura da Aplicação Web
2. Credenciais de Acesso AWS
3. Conexão com o Banco de Dados
4. Conexão com o Serviço S3
5. Rotas da Aplicação Web
6. Estrutura do Banco de dados
7. Estrutura do Bucket S3

App Web com S3 e RDS

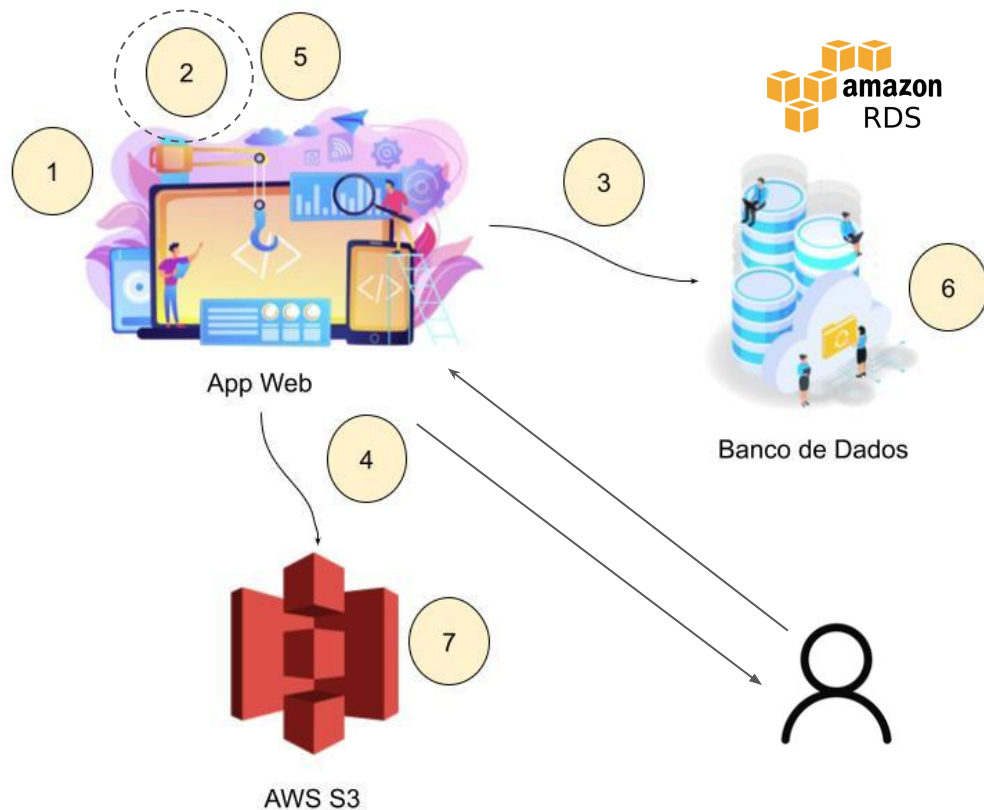


1. Estrutura da Aplicação Web

Aplicação [Flask](#)

- principal.py
- banco.py
- s3_handle.py
- utilidades.py
- templates
 - home.html
 - upload.html
 - downloads.html
 - image_form.html

App Web com S3 e RDS



2. Credenciais de Acesso AWS

Crie um usuário via [AWS IAM](#) com as políticas de segurança de acesso ao EC2, S3 e RDS.

Exemplo: my_user_app

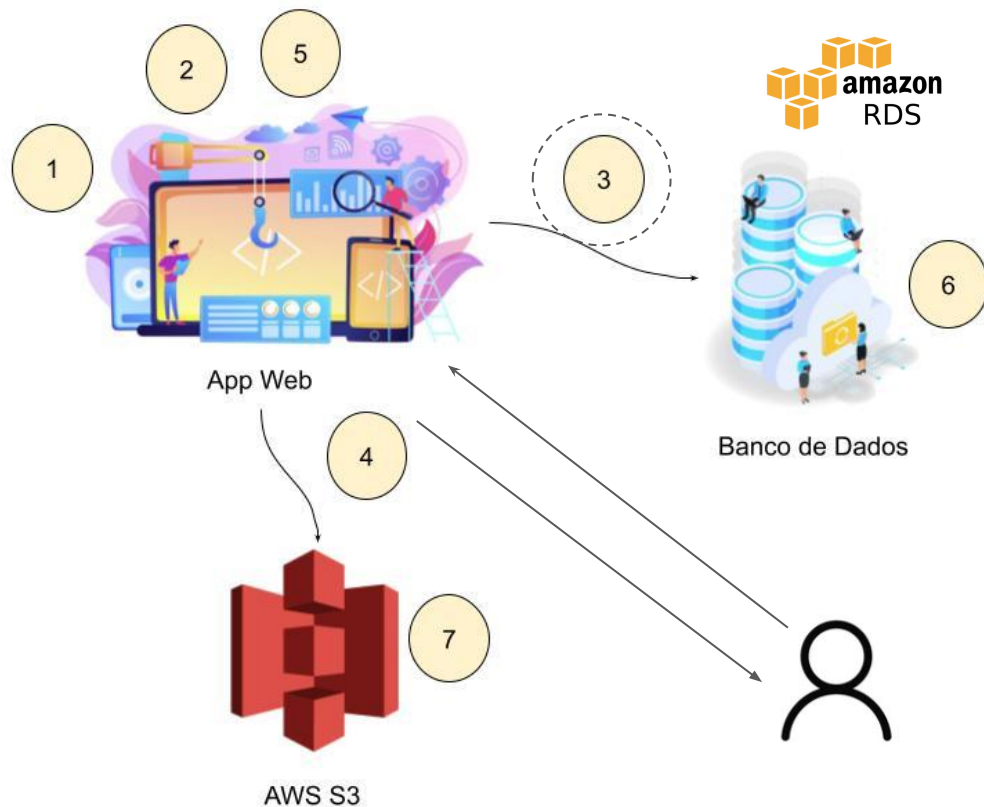
ACCESS_KEY_ID=?

SECRET_ACCESS_KEY=?

Liberar as [portas tcp](#) e recursos http/https

Exemplo: portas: 80, 443, 8080, 5000, 3306

App Web com S3 e RDS



3. Conexão com o Banco de Dados

Defina a string de conexão com o banco de dados escolhido.

Database Type: mysql

User Credentials: username/password

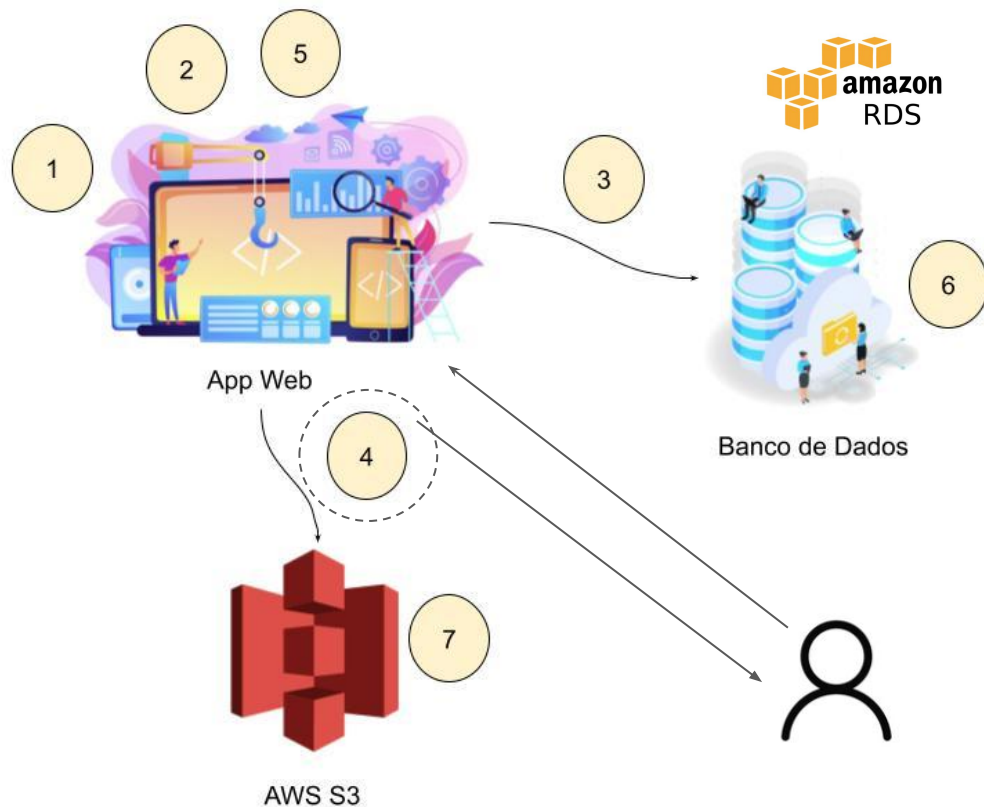
Host Name: nome DNS da localização da instância AWS RDS

Database Name: nome do banco no AWS RDS

String de conexão:

```
mysql+pymysql://{DB_USER}:{PASSWORD_DB_USER}@{INSTAN  
CIA_DB_AWS_RDS}/{BANCO_AWS_RDS}
```

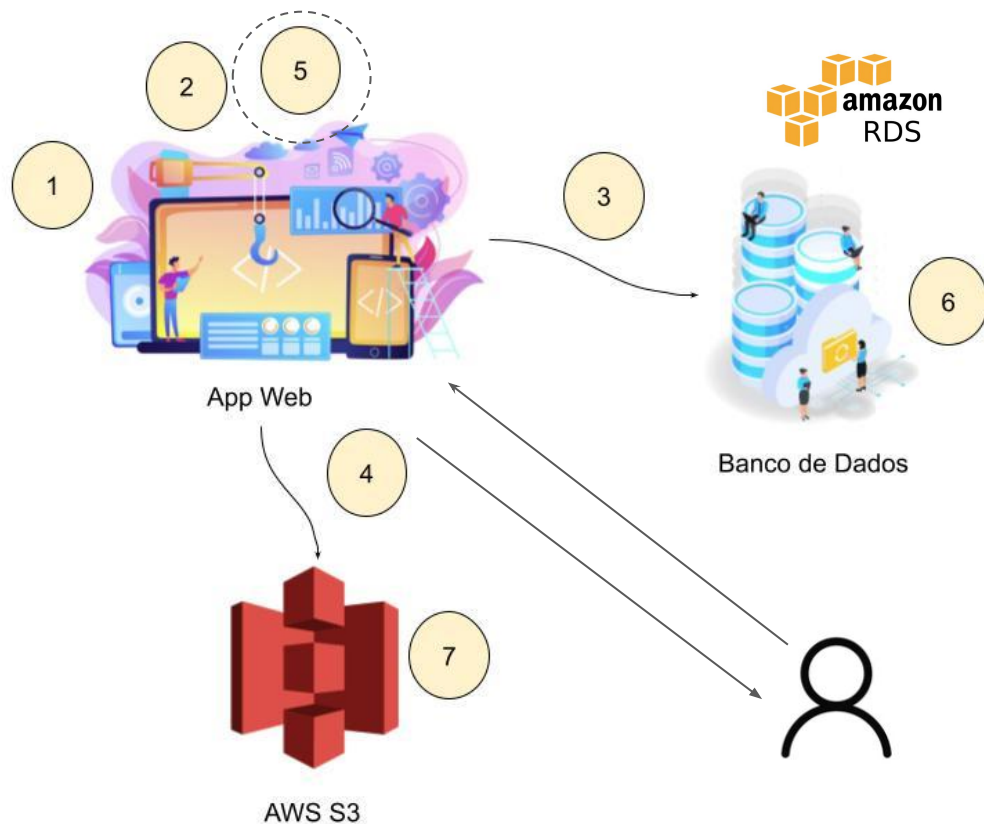
App Web com S3 e RDS



4. Conexão com o Serviço S3

- Configure o [SDK AWS](#) de acordo com sua linguagem escolhida.
- Informe as credenciais de acesso do usuário AWS.
- Crie um handle para manipular a comunicação com serviço AWS S3.

App Web com S3 e RDS



5. Rotas da Aplicação Web

Defina as rotas da aplicação

Por exemplo:

home: página principal

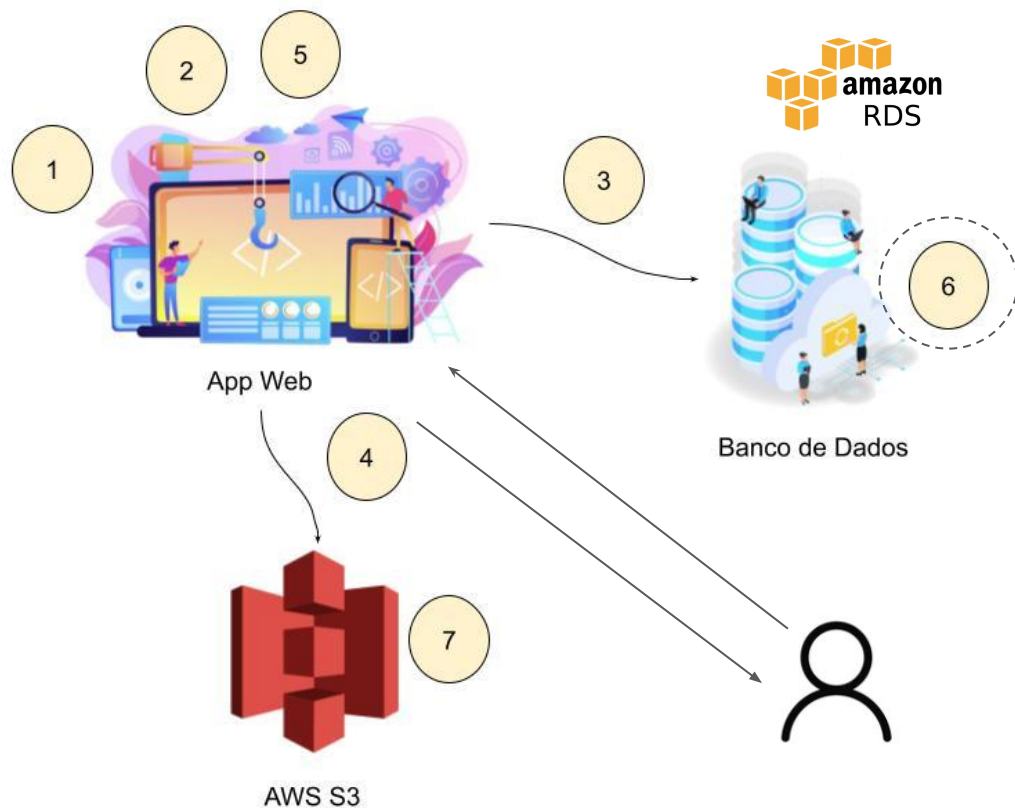
uploads: página de upload de arquivos

downloads: página de downloads de arquivos

my_image: página de exibição de imagem

selecionada

App Web com S3 e RDS



6. Estrutura do Banco de dados

Defina as tabelas do banco de dados.

Por exemplo:

Tabela File

id: chave id do arquivo

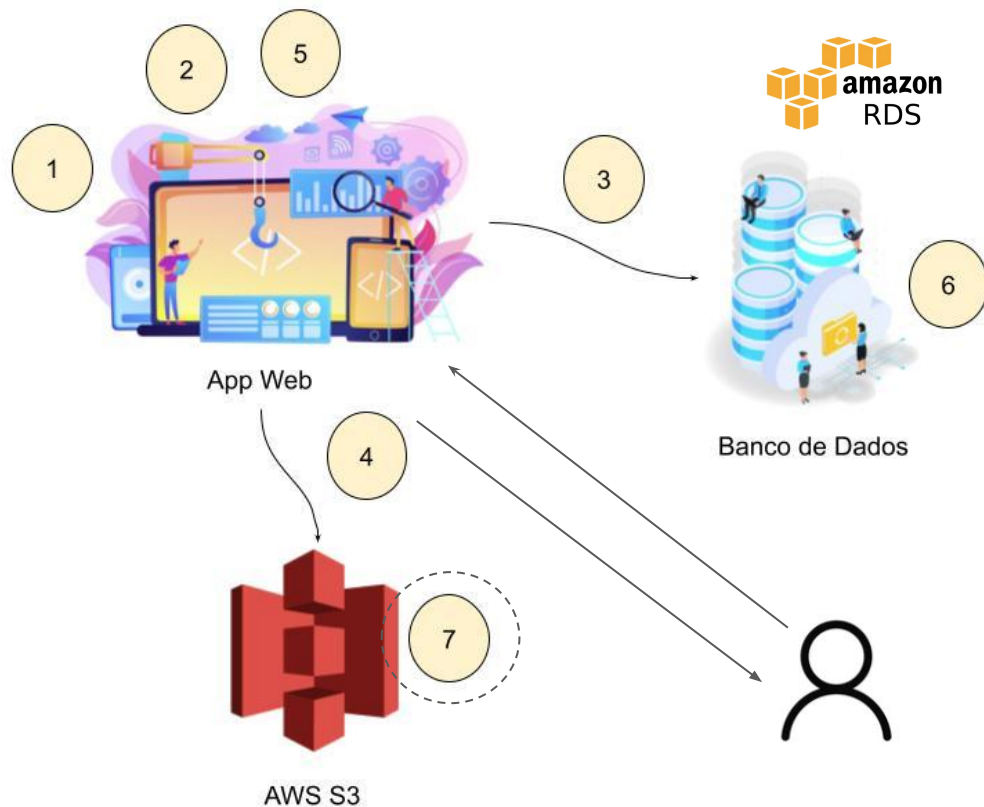
original_filename: nome original do arquivo

filename: nome armazenado no banco e no S3

bucket: nome do bucket de armazenamento

region: id da região do bucket de armazenamento

App Web com S3 e RDS



7. Estrutura do Bucket S3

Configurações do bucket

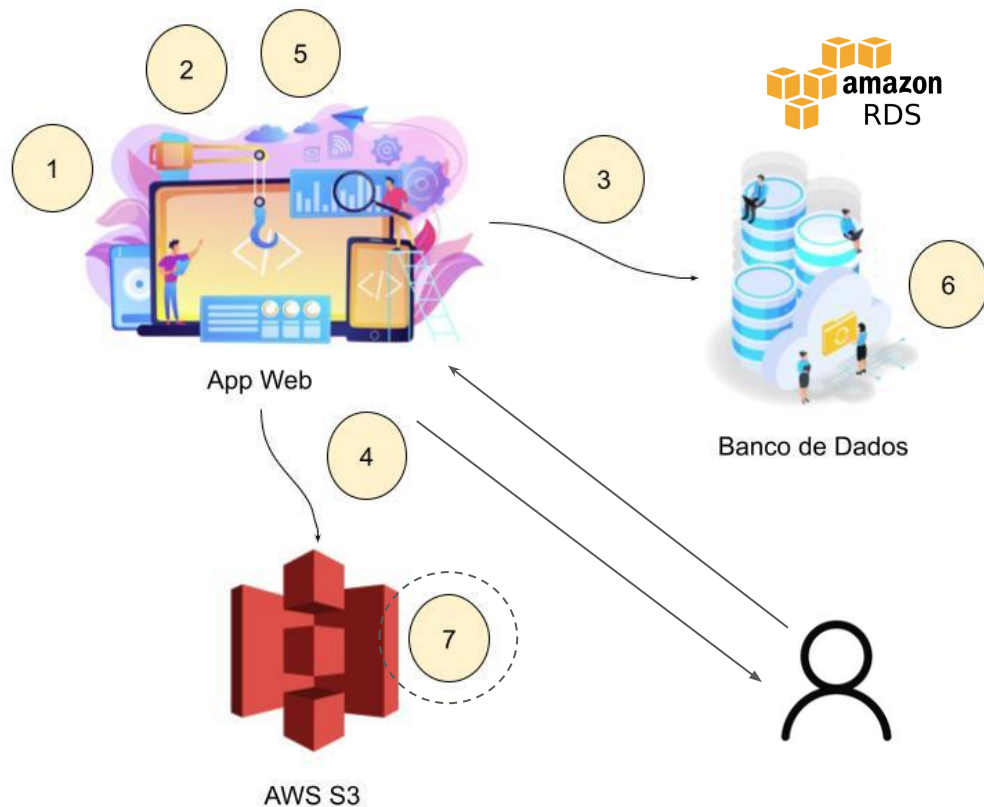
Por exemplo:

Nome: my-app-files-bucket

Região: us-east-1

Política de Acesso: defina um json com as informações de effect, action, e resource

App Web com S3 e RDS



7. Estrutura do Bucket S3

Política de Acesso:

```
{
  "Version": "2012-10-17",
  "Id": "Policy1712415077164",
  "Statement": [
    {
      "Sid": "Stmnt1712415054743",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::my-app-files-bucket/*"
    }
  ]
}
```

Telas da Aplicação Web com S3

← → ↻ ⓘ http://localhost:5000

Bem vindo a página Home

Upload e Downloads de Arquivos do AWS S3

[Upload File](#)

[Download Files](#)

Telas da Aplicação Web com S3

← → ↻ ⓘ http://localhost:5000/upload

Upload to S3

Escolher arquivo

Nenhum arquivo escolhido

Upload

armando.jpeg - [855bebecb3b74d25b0135fe76368fa25.jpeg](#)
network_computer.jpeg - [173277000f0b4074accbe00d75091fab.jpeg](#)
graphics.png - [81031c5518be42d4bd2f3f3e8ee9d347.png](#)
magicforest.png - [44321dd1527d46d09dcdecc8372146c9.png](#)
mypython.png - [904c5624d24d4c97b0c872fabd0ddefa.png](#)
arquitetura.png - [40143cef3f9840c284919d96b0b42cab.png](#)
armando_nyc.jpeg - [86b133a4e6dc44448ec965a1683b2e53.jpeg](#)
use_cases.png - [743dde544411479cbe0a4b04468d7f4d.png](#)

[\[Home\]](#)

Telas da Aplicação Web com S3

← → ↻ ⓘ http://localhost:5000/downloads

Lista de arquivos do S3

armando.jpeg - [download](#) - [View](#)

network_computer.jpeg - [download](#) - [View](#)

graphics.png - [download](#) - [View](#)

magicforest.png - [download](#) - [View](#)

mypythons.png - [download](#) - [View](#)

arquitetura.png - [download](#) - [View](#)

armando_nyc.jpeg - [download](#) - [View](#)

use_cases.png - [download](#) - [View](#)

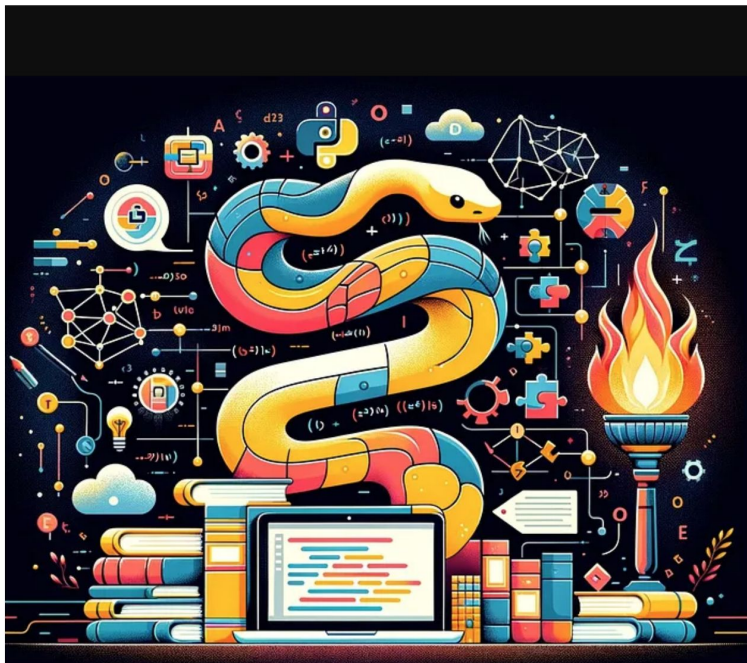
[\[Home\]](#)

Telas da Aplicação Web com S3

← → ↺ ⓘ http://localhost:5000/myimage/904c5624d24d4c97b0c872fabd0ddefa.png

Arquivo do S3

Image from S3 Byte Content



Código da Aplicação Web (backend)

```
11 # True para limpar a instancia banco de dados atual
12 # Obs: o valor deve ser True na 1a execucao da aplicacao
13 # para criar um banco limpo a estrutura limpa das tabelas
14 DROP_DATA_BASE = True
15
16 # Carrega os valores das credenciais de acesso da AWS
17 ACCESS_KEY_ID = os.getenv('ACCESS_KEY_ID')
18 SECRET_ACCESS_KEY = os.getenv('SECRET_ACCESS_KEY')
19 DB_USER = os.getenv('DB_USER')
20 PASSWORD_DB_USER = os.getenv('PASSWORD_DB_USER')
21 INSTANCIA_DB_AWS_RDS = 'mydbfileteste.cdwwkuakqjji.us-east-1.rds.amazonaws.com'
22 BANCO_AWS_RDS = 'mydbfiles'
23 SQLALCHEMY_DATABASE_URI = f'mysql+pymysql://{DB_USER}:{PASSWORD_DB_USER}@{INSTANCIA_DB_AWS_RDS}/{BANCO_AWS_RDS}'
24
25 # Instancia principal da aplicação
26 app = Flask(__name__)
27 app.secret_key = 'thisismysecretkeyfrommywebapplication'
28 app.config['SQLALCHEMY_DATABASE_URI'] = SQLALCHEMY_DATABASE_URI
29
30 # Inicializa a instância do banco de dados
31 banco.db.init_app(app)
32 banco.create_tables(app, DROP_DATA_BASE)
33
34 # Carrega o componente S3
35 print('Carregando as credenciais da AWS')
36 s3 = s3_handle.carrega_s3(ACCESS_KEY_ID, SECRET_ACCESS_KEY)
```

```
principal.py > ...
1 from flask import Flask, render_template
2 from flask import request, redirect, url_for
3 from flask import flash, send_file
4 import os
5 import banco
6 import s3_handle
7 import uuid
8 import utilidades
9 import requests
10 import base64
11 import io
```


Código da Aplicação Web (backend)

```
33 # Rota para a pagina home
34 @app.route("/")
35 > def home_page(): ...
37
38 # Rota para a pagina de uploads
39 @app.route("/upload", methods=["GET", "POST"])
40 > def upload_page(): ...
64
65 # Rota que carrega a pagina de downloads dos arquivos
66 @app.route("/downloads", methods=["GET"])
67 > def downloads_page(): ...
75
76 # Recupera os bytes de uma imagem do bucket S3
77 > def get_image_bytes(image_url): ...
81
82 # Rota que recupera os bytes da imagem e guarda em um formato base64
83 # para exibir o conteudo na pagina image_form
84 @app.route("/myimage/<nome>")
85 > def show_image(nome): ...
```


Código da Aplicação Web (backend)

```
38 # Rota para a pagina de uploads
39 @app.route("/upload", methods=["GET", "POST"])
40 def upload_page():
41     if request.method == "POST":
42         try:
43             uploaded_file = request.files["file-to-save"]
44
45             if not utilidades.allowed_file(uploaded_file.filename):
46                 flash("Tipo de arquivo não permitido!")
47                 return redirect(url_for('upload_page'))
48
49             new_filename = uuid.uuid4().hex + '.' + uploaded_file.filename.rsplit('.', 1)[1].lower()
50             s3.upload_fileobj(uploaded_file, s3_handle.BUCKET_NAME, new_filename)
51             file = banco.File(original_filename=uploaded_file.filename, filename=new_filename,
52                               bucket=s3_handle.BUCKET_NAME, region=s3_handle.AWS_S3_REGION)
53             banco.db.session.add(file)
54             banco.db.session.commit()
55         except Exception as ex:
56             flash(f"Erro no upload! {str(ex)}")
57             return redirect(url_for('upload_page'))
58
59     return redirect(url_for("upload_page"))
60
61 files = banco.File.query.all()
62
63 return render_template("upload.html", files=files)
```

```
33 # Rota para a pagina home
34 @app.route("/")
35 def home_page():
36     return render_template("home.html")
```

Código da Aplicação Web (backend)

```
65 # Rota que carrega a pagina de downloads dos arquivos
66 @app.route("/downloads", methods=["GET"])
67 def downloads_page():
68     files = banco.File.query.all()
69
70     if not files:
71         flash('Nenhum arquivo para download!')
72         return redirect(url_for('download_page'))
73
74     return render_template("downloads.html", files=files)
75
76 # Recupera os bytes de uma imagem do bucket S3
77 def get_image_bytes(image_url):
78     response = requests.get(image_url)
79     response.raise_for_status() # Raise an exception for non-200 status codes
80     return response.content
```

Código da Aplicação Web (backend)

```
82 # Rota que recupera os bytes da imagem e guarda em um formato base64
83 # para exibir o conteudo na pagina image_form
84 @app.route("/myimage/<nome>")
85 def show_image(nome):
86     bucket_path = "https://my-app-files-bucket.s3.amazonaws.com"
87     image_url = bucket_path + "/" + nome
88     image_bytes = get_image_bytes(image_url)
89     extensao = utilidades.get_file_extension(nome)
90
91     encoded_bytes = base64.b64encode(image_bytes).decode('utf-8') # Encode as base64 and decode for URI
92     image_data_uri = f"data:image/{extensao};base64,{encoded_bytes}"
93
94     return render_template("image_form.html", image_data_uri=image_data_uri)
```

Código da Aplicação Web (backend)

```
96  # Recupera os bytes da imagem e guarda em memoria
97  @app.route("/myimage2/<nome>")
98  def show_image2(nome):
99      bucket_path = "https://my-app-files-bucket.s3.amazonaws.com"
100      image_url = bucket_path + "/" + nome
101      image_bytes = get_image_bytes(image_url)
102
103      extensao = utilidades.get_file_extension(nome)
104      my_mimetype = "image" + "/" + extensao
105
106      return send_file(io.BytesIO(image_bytes), mimetype=my_mimetype)
```

Código da Aplicação Web (backend)

banco.py > ...

```
1  from flask_sqlalchemy import SQLAlchemy
2
3  # componente de banco de dados
4  db = SQLAlchemy()
5
6  # Classe que representa os dados de um arquivo
7  class File(db.Model):
8      id = db.Column(db.Integer, primary_key=True)
9      original_filename = db.Column(db.String(100))
10     filename = db.Column(db.String(100))
11     bucket = db.Column(db.String(100))
12     region = db.Column(db.String(100))
13
14 # Cria as tabelas do banco
15 def create_tables(app, drop_data_base):
16     try:
17         with app.app_context():
18             print('Carrega as tabelas do banco')
19             if drop_data_base:
20                 db.drop_all()
21                 db.create_all()
22                 db.session.commit()
23             print('Tabelas carregadas com sucesso!')
24     except Exception as ex:
25         print(f'Erro ao carregar o banco! {str(ex)}')
```

Código da Aplicação Web (backend)

s3_handle.py > carrega_s3

```
1  import boto3
2
3  BUCKET_NAME = "my-app-files-bucket"
4  AWS_S3_REGION = "us-east-1"
5
6  def carrega_s3(access_key_id, secret_access_key):
7      try:
8          if access_key_id is None or secret_access_key is None:
9              raise ValueError('Credenciais inválidas!')
10
11          # Componente para acessar o AWS S3
12          my_s3 = boto3.client(
13              's3',
14              aws_access_key_id=access_key_id,
15              aws_secret_access_key=secret_access_key
16          )
17          print("Componente de acesso ao S3 carregado com sucesso!")
18          return my_s3
19      except Exception as ex:
20          print(f"Erro ao carregar o componente do S3: {str(ex)}")
```


Código da Aplicação Web (backend)

utilidades.py >  get_file_extension

```
1  ALLOWED_EXTENSIONS = {'txt', 'pdf', 'png', 'jpeg'}
2
3  def allowed_file(filename):
4      return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS
5
6  def get_file_extension(filename):
7      return filename.rsplit('.', 1)[1].lower()
```

≡ requirements.txt

```
1  Flask
2  Flask-SQLAlchemy
3  boto3
4  requests
5  pymysql
```

Código da Aplicação Web (frontend)

templates > <> home.html >  html

```
1  <html>
2      <head>
3          <title>Home</title>
4      </head>
5      <body>
6          <h1>Bem vindo a página Home</h1>
7          <p>Upload e Downloads de Arquivos do AWS S3</p>
8
9          {% with messages = get_flashed_messages() %}
10         {% if messages %}
11             <ul class="flashes">
12                 {% for message in messages %}
13                     <h2>{{ message }}</h2>
14                 {% endfor %}
15             </ul>
16         {% endif %}
17         {% endwith %}
18
19         <a href="/upload">Upload File</a>
20         <br>
21         <a href="/downloads">Download Files</a>
22     </body>
23 </html>
```


Código da Aplicação Web (frontend)

templates > <> downloads.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Downloads</title>
6  </head>
7  <body>
8      <h3>Lista de arquivos do S3</h3>
9
10     {% with messages = get_flashed_messages() %}
11     {% if messages %}
12         <ul class="flashes">
13             {% for message in messages %}
14                 <h2>{{ message }}</h2>
15             {% endfor %}
16         </ul>
17     {% endif %}
18     {% endwith %}
19
20     {% for file in files %}
21     <div>
22         {{ file.original_filename }} - <a href="https://{{ file.bucket }}.s3.{{ file.region }}.amazonaws.com/
23         - <a href="{{url_for('show_image', nome=file.filename)}}">View</a>
24     </div>
25     {% endfor %}
26     <br>
27     <a href="/">[Home]</a>
28 </body>
29 </html>
```

Código da Aplicação Web (frontend)

templates > <> upload.html >  html

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Upload file</title>
6  </head>
7  <body>
8      <h3>Upload to S3</h3>
9
10     {% with messages = get_flashed_messages() %}
11     {% if messages %}
12         <ul class="flashes">
13             {% for message in messages %}
14                 <h2>{{ message }}</h2>
15             {% endfor %}
16         </ul>
17     {% endif %}
18     {% endwith %}
```

Código da Aplicação Web (frontend)

```
20 <form method="POST" enctype="multipart/form-data" style="margin-bottom: 30px;">
21   <input type="file" name="file-to-save" />
22   <button>Upload</button>
23 </form>
24
25 {% for file in files %}
26 <div>
27   {{ file.original_filename }} -
28   <a href="https://{{ file.bucket }}.s3.{{ file.region }}.amazonaws.com/{{ file.filename }}">
29     {{ file.filename }}</a>
30 </div>
31 {% endfor %}
32 <br>
33 <a href="/">[Home]</a>
34 </body>
35 </html>
```

Código da Aplicação Web

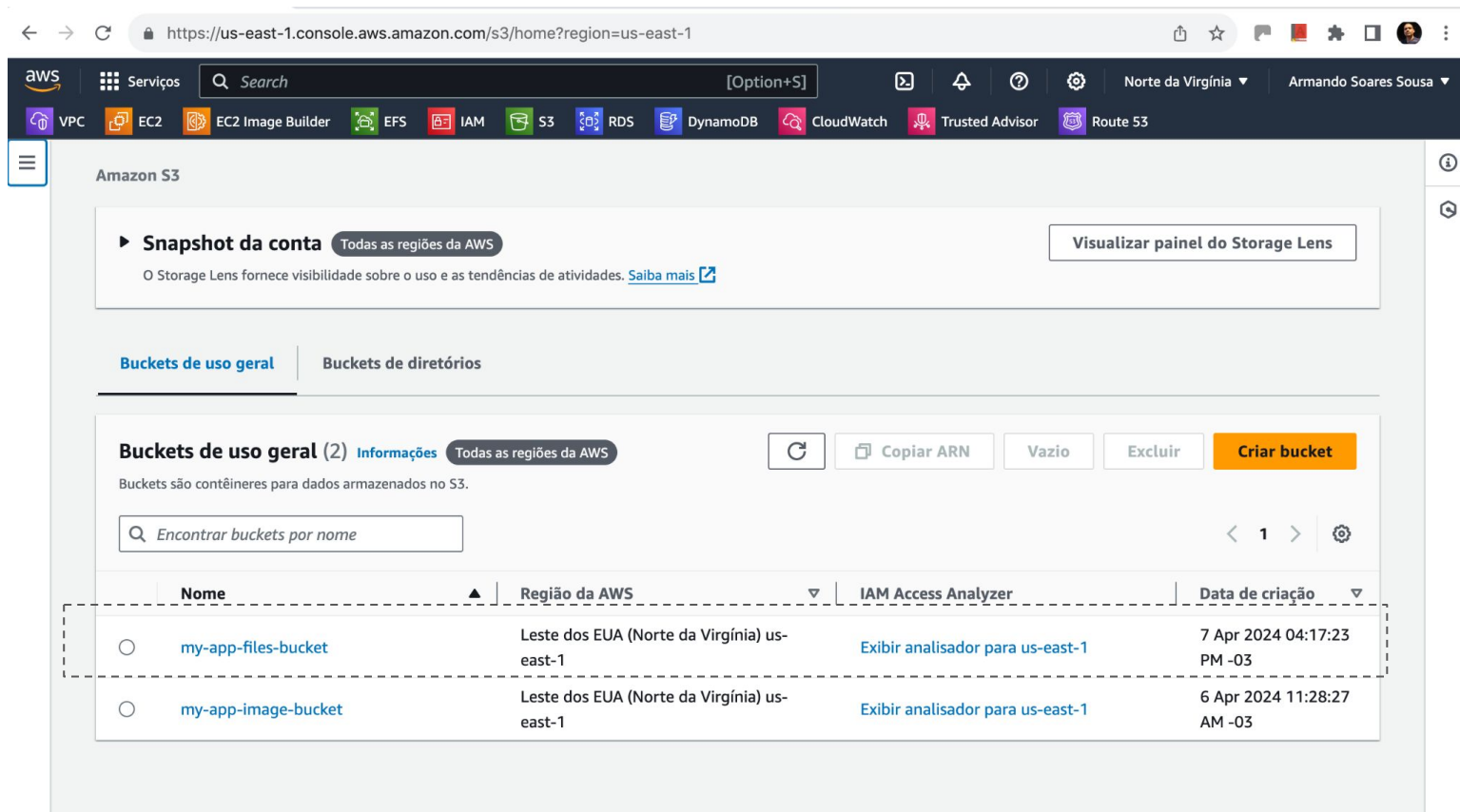
templates > <> image_form.html >  html

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Downloads</title>
6  </head>
7  <body>
8      <h3>Arquivo do S3</h3>
9
10     {% with messages = get_flashed_messages() %}
11     {% if messages %}
12         <ul class="flashes">
13             {% for message in messages %}
14                 <h2>{{ message }}</h2>
15             {% endfor %}
16         </ul>
17     {% endif %}
18     {% endwith %}
19
20     <h1>Image from S3 Byte Content</h1>
21     
22     <br>
23     <a href="/">[Home]</a>
24 </body>
25 </html>
```

Código da Aplicação Web

Disponível em <https://github.com/armandossrecife/mysimpleuploads3rds>

Console AWS S3 (Lista de buckets)



The screenshot shows the AWS S3 console interface. At the top, there's a navigation bar with the AWS logo, a search bar, and various service icons (VPC, EC2, EC2 Image Builder, EFS, IAM, S3, RDS, DynamoDB, CloudWatch, Trusted Advisor, Route 53). The user is logged in as Armando Soares Sousa in the Norte da Virgínia region.

The main content area is titled "Amazon S3". It features a "Snapshot da conta" section with a "Todas as regiões da AWS" filter and a "Visualizar painel do Storage Lens" button. Below this, there are two tabs: "Buckets de uso geral" (selected) and "Buckets de diretórios".

The "Buckets de uso geral" section shows a list of buckets. The header includes a search bar "Encontrar buckets por nome", a refresh button, a "Copiar ARN" button, a "Vazio" button, an "Excluir" button, and a "Criar bucket" button. The list is filtered to show 1 bucket.

	Nome	Região da AWS	IAM Access Analyzer	Data de criação
<input type="radio"/>	my-app-files-bucket	Leste dos EUA (Norte da Virgínia) us-east-1	Exibir analisador para us-east-1	7 Apr 2024 04:17:23 PM -03
<input type="radio"/>	my-app-image-bucket	Leste dos EUA (Norte da Virgínia) us-east-1	Exibir analisador para us-east-1	6 Apr 2024 11:28:27 AM -03

Console AWS S3 (Lista de objetos)

Objetos

Propriedades

Permissões

Métricas

Gerenciamento

Pontos de acesso

Objetos (14) [Informações](#)

Copiar URI do S3

Copiar URL

Fazer download

Abrir

Excluir

Ações ▼

Criar pasta

Carregar

Os objetos são as entidades fundamentais armazenadas no Amazon S3. Você pode usar o [inventário do Amazon S3](#) para obter uma lista de todos os objetos em seu bucket. Para outras pessoas acessarem seus objetos, você precisará conceder permissões explicitamente a eles. [Saiba mais](#)

< 1 >

<input type="checkbox"/>	Nome ▲	Tipo ▼	Última modificação ▼	Tamanho ▼	Classe de armazenamento ▼
<input type="checkbox"/>	173277000f0b4074accbe0Od75091fab.jpeg	jpeg	8 Apr 2024 12:21:34 AM -03	339.9 KB	Padrão
<input type="checkbox"/>	2f7967f49c164d87a2e0bc3bae5f0414.jpeg	jpeg	10 Apr 2024 12:26:27 AM -03	433.7 KB	Padrão
<input type="checkbox"/>	40143cef3f9840c284919d96b0b42cab.png	png	9 Apr 2024 10:48:06 PM -03	102.3 KB	Padrão

Console AWS S3 (Política de acesso de um bucket)

Política do bucket

[Editar](#)[Excluir](#)

A política de bucket, escrita em JSON, fornece acesso aos objetos armazenados no bucket. As políticas de bucket não se aplicam a objetos de propriedade de outras contas. [Saiba mais](#) 

```
{
  "Version": "2012-10-17",
  "Id": "Policy1712415077164",
  "Statement": [
    {
      "Sid": "Stmt1712415054743",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::my-app-files-bucket/*"
    }
  ]
}
```

[Copiar](#)

Console AWS S3 (Detalhes de um objeto)

ca2ec5697d0744699ea729fb51b1ffd7.png [Informações](#)

 Copiar URI do S3

 Fazer download

 Abrir

Ações de objeto ▼

Propriedades

Permissões

Versões

Visão geral do objeto

Proprietário

armando.sousa

Região da AWS

Leste dos EUA (Norte da Virgínia) us-east-1

Última modificação

10 Apr 2024 12:17:51 AM -03


Tamanho

4.5 MB

Tipo

png

Chave

 ca2ec5697d0744699ea729fb51b1ffd7.png


URI do S3

 s3://my-app-files-bucket/ca2ec5697d0744699ea729fb51b1ffd7.png

Nome de recurso da Amazon (ARN)

 arn:aws:s3:::my-app-files-bucket/ca2ec5697d0744699ea729fb51b1ffd7.png

Tag da entidade (Etag)


 570ad5c57c674732937c16fd53e4f555


URL de objeto


 <https://my-app-files-bucket.s3.amazonaws.com/ca2ec5697d0744699ea729fb51b1ffd7.png>

Console AWS S3 (Permissões de um objeto)

ca2ec5697d0744699ea729fb51b1ffd7.png [Informações](#)

 Copiar URI do S3

 Fazer download

 Abrir

Ações de objeto ▼

Propriedades


Permissões

Versões

Lista de controle de acesso (ACL)




[Editar](#)

Conceda permissões básicas de leitura/gravação a contas da AWS. [Saiba mais](#)



Esse bucket tem a configuração imposto pelo proprietário do bucket aplicada à propriedade do objeto

Quando a configuração **imposto pelo proprietário do bucket** for aplicado, use políticas de bucket para controlar o acesso. [Saiba mais](#)

Beneficiário	Objeto	ACL do objeto
Proprietário do objeto (sua conta da AWS) ID canônico:  da018f3776d306f41c7bb92ab6c465c8451759385bccb70c38082a8ab5860443	Leitura	Leitura, gravação
Todos (acesso público) Grupo:  http://acs.amazonaws.com/groups/global/AllUsers	-	-
Grupo de usuários autenticados (qualquer pessoa com uma conta da AWS) Grupo:  http://acs.amazonaws.com/groups/global/AuthenticatedUsers	-	-

Console AWS S3 (Download de um objeto)

<https://my-app-files-bucket.s3.amazonaws.com/ca2ec5697d0744699ea729fb51b1ffd7.png>



Console AWS RDS (Instância criada)

✓ Banco de dados criado com êxito mydbfilesteste

Visualizar detalhes da conexão

Você pode usar as configurações do mydbfilesteste para simplificar a configuração de complementos do banco de dados sugeridos enquanto concluímos a criação do banco de dados para você.

RDS > Bancos de dados



Considere criar uma implantação azul/verde para minimizar o tempo de inatividade durante as atualizações

Você pode considerar o uso de implantações azuis/verdes do Amazon RDS e minimizar seu tempo de inatividade durante as atualizações. Uma implantação azul/verde fornece um ambiente de preparação para alterações nos bancos de dados de produção. [Guia do usuário do RDS](#) [Guia do usuário do Aurora](#)

Bancos de dados (1)

☒ Recursos do grupo



Modificar

Ações ▼

Restaurar do S3

Criar banco de dados

🔍 Filtrar por bancos de dados


< 1 > ⚙

<input type="checkbox"/> Identificador de banco de dados ▲	Status ▼	Função ▼	Mecanismo ▼	Região e AZ ▼	Tamanho ▼	Recomendações ▼
<input checked="" type="radio"/> mydbfilesteste	✓ Disponível	Instância	MariaDB	us-east-1d	db.t3.micro	




Console AWS RDS (Segurança e conexão)

[RDS](#) > [Bancos de dados](#) > mydbfileteste

mydbfileteste

 [Modificar](#) [Ações ▼](#)


Resumo

Identificador de banco de dados mydbfileteste	Status  Disponível	Função Instância	Mecanismo MariaDB	Recomendações
CPU  3.35%	Classe db.t3.micro	Atividade atual  0 Conexões	Região e AZ us-east-1d	

[Segurança e conexão](#) | [Monitoramento](#) | [Logs e eventos](#) | [Configuração](#) | [Manutenção e backups](#) | [Tags](#) | [Recomendações](#)

Segurança e conexão

Endpoint e porta

Endpoint
 mydbfileteste.cdwkkuakqpi.us-east-1.rds.amazonaws.com

Porta
3306

Redes


Zona de disponibilidade
us-east-1d

VPC
[vpc-0e1426c149056cec2](#)

Grupo de sub-redes
[default-vpc-0e1426c149056cec2](#)

Sub-redes
[subnet-04dbb803db26da23](#)
[subnet-04b41e74b7f36a19](#)
[subnet-02d79321c9d25cdc3](#)
[subnet-0861dd1952bdfdc90](#)
[subnet-0c163a828b14fba3d](#)
[subnet-06fb92bf4c5bbe16e](#)

Segurança

Grupos de segurança da VPC
[default \(sg-00750ce86a5dd273e\)](#)
 **Ativo**

Publicamente acessível
Sim

Autoridade de certificação [Informações](#)
rds-ca-rsa2048-g1

Data da autoridade de certificado
May 25, 2061, 20:34 (UTC-03:00)

Data de expiração do certificado da instância de banco de dados
April 13, 2025, 13:25 (UTC-03:00)

Console AWS RDS (Regras de segurança e replicação)

Regras de grupos de segurança (3)

Filtrar por Regras de grupos de segurança

<

1

>

Grupo de segurança	Tipo	Regra
default (sg-00750ce86a5dd273e)	EC2 Security Group - Inbound	sg-00750ce86a5dd273e
default (sg-00750ce86a5dd273e)	CIDR/IP - Inbound	0.0.0.0/0
default (sg-00750ce86a5dd273e)	CIDR/IP - Outbound	0.0.0.0/0

Replicação (1)

Filtrar por Replicação

<

1

>

Identificador de banco de dados	Função	Região e AZ	Origem da replicação	Estado da replicação	Atraso
mydbfileteste	Instância	us-east-1d	-	-	-

Console AWS RDS (Configuração)

mydbfilesteste

Modificar

Ações ▾

Resumo

Identificador de banco de dados mydbfilesteste	Status Disponível	Função Instância	Mecanismo MariaDB	Recomendações
CPU <div>3.35%</div>	Classe db.t3.micro	Atividade atual <div>0 Conexões</div>	Região e AZ us-east-1d	

Segurança e conexão

Monitoramento

Logs e eventos

Configuração

Manutenção e backups

Tags

Recomendações

Instância

Configuração

ID da instância de banco de dados
mydbfilesteste

Versão do mecanismo
10.11.6

Nome do banco de dados
mydbfiles

Modelo de licença
General Public License

Grupos de opções
default:mariadb-10-11 Em sincronia

Nome de recurso da Amazon (ARN)

arn:aws:rds:us-east-1:229020406899:db:mydbfilesteste

ID do recurso
db-WDO53HKPB3AI4KBRAL553K3I2I

Classe de instância

Classe de instância
db.t3.micro

vCPU
2

RAM
1 GB

Disponibilidade

Nome do usuário principal
admin

Senha principal

Autenticação do banco de dados do IAM
Não habilitado

Multi-AZ

Armazenamento

Criptografia
Não habilitado

Tipo de armazenamento
SSD de uso geral (gp2)

Armazenamento
20 GiB

IOPS provisionadas
-

Taxa de throughput
-

Escalabilidade automática do armazenamento
Desabilitado

Configuração do sistema de arquivos de armazenamento

Performance Insights

Performance Insights habilitado
Desativado