

# Catálogo de Filmes – Mackenzie Edition

Projeto Integrador – Cloud Developing – 2025/1

Turma 5H – Universidade Presbiteriana Mackenzie

## Integrantes do Grupo

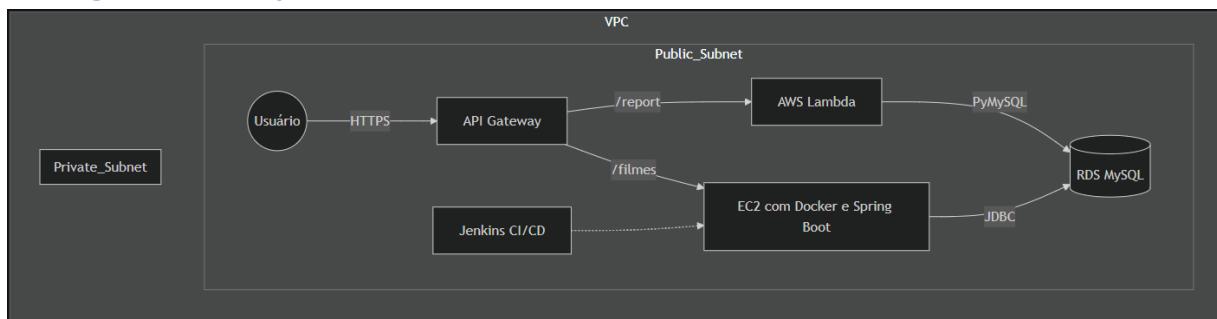
RA	Nome
10426930	Leonardo De Castro Tonon
10417578	João Pedro Fernandes Milhomens
10417821	Marcel Nobrega Zamboni

## 1. Visão Geral do Projeto

O Catálogo de Filmes – Mackenzie Edition é uma aplicação em nuvem que permite cadastrar, listar, editar e excluir filmes, de forma simples e intuitiva. A aplicação foi desenvolvida em Java com Spring Boot, integrada a um banco de dados MySQL no RDS, e implantada em uma instância EC2 com Docker.

Além disso, uma função AWS Lambda foi criada e integrada ao API Gateway, permitindo a geração de um relatório estatístico (/report) com dados consumidos diretamente da API principal.

## 2. Diagrama da Arquitetura



Descrição das Camadas:

Frontend – HTML, CSS e JS: Interface web que consome a API de filmes

Backend – Spring Boot (EC2 + Docker): API REST com as rotas CRUD de filmes

Banco de Dados – Amazon RDS (MySQL): Armazena de forma persistente os filmes cadastrados

Gateway – Amazon API Gateway: Encaminha rotas para EC2 e Lambda

Função Lambda – AWS Lambda (Python): Gera relatório estatístico de filmes (/report)

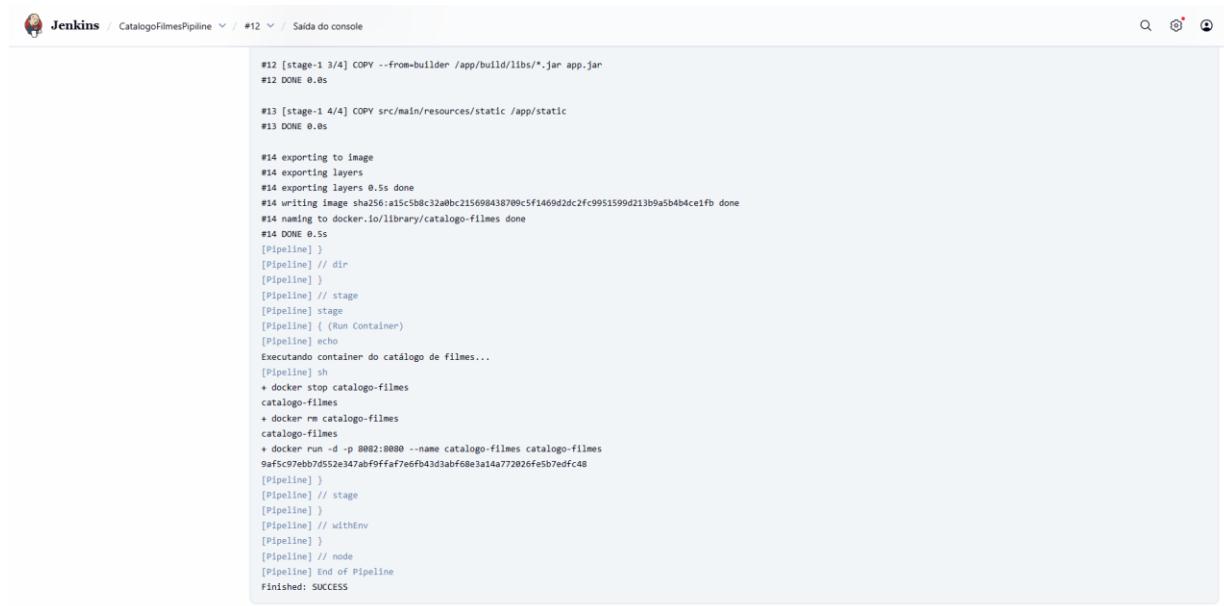
CI/CD – Jenkins: Automatiza build, criação de imagem e deploy no EC2

### 3. Explicação do Pipeline (Jenkins)

O pipeline Jenkins foi configurado para automatizar o ciclo de integração e entrega contínua (CI/CD) da aplicação.

Etapas:

1. Checkout: obtém o código-fonte do GitHub.
2. Build: compila o projeto com gradle clean build -x test.
3. Docker Build: cria a imagem com o backend.
4. Deploy: substitui o container antigo e sobe o novo.
5. Verificação: testa a aplicação na porta 8080.



The screenshot shows the Jenkins Pipeline console output for a job named 'CatalogoFilmesPipeline'. The output is divided into five stages, each with its own set of commands and logs:

- Stage 1:** COPY --from=builder /app/build/libs/\*.jar app.jar  
#12 DONE 0.8s
- Stage 2:** COPY src/main/resources/static /app/static  
#13 DONE 0.8s
- Stage 3:** #14 exporting to image  
#14 exporting layers  
#14 exporting layers: 0.5s done  
#14 writing image sha256:a15c5b8c32a0bc215698438709cf1469d2dc2fc9951599d213b9a5b4b4ce1fb done  
#14 naming to docker.io/library/catalogo-filmes done  
#14 DONE 0.5s
- Stage 4:** [Pipeline] // stage  
[Pipeline] stage  
[Pipeline] { (Run Container)  
[Pipeline] echo  
Executando container do catálogo de filmes...  
[Pipeline] sh  
+ docker stop catalogo-filmes  
catalogo-filmes  
+ docker rm catalogo-filmes  
catalogo-filmes  
+ docker run -d -p 8082:8080 --name catalogo-filmes catalogo-filmes  
9af5c97eb7d52e347abf9ffa7ee6fb43d3abf68e3a14a77292fe5b7edfc48
- Stage 5:** [Pipeline] // stage  
[Pipeline] // withEnv  
[Pipeline] // node  
[Pipeline] End of Pipeline

The final message indicates the pipeline finished successfully.

## 4. Capturas de Tela dos Serviços AWS

### Amazon RDS

The screenshot shows the AWS RDS console for the 'catalogofilmes-db' MySQL database. The main summary card displays the database identifier, status (Available), instance type (db.t3.micro), and connection information. Below the summary are tabs for Security & connection, Monitoring, Logs & events, Configuration, Integrations (selected), Maintenance & backups, Migrations & data, Tags, and Recommendations. The Security & connection section details the endpoint (catalogofilmes-db.c762082m6il2.us-east-1.rds.amazonaws.com), port (3306), and VPC (vpc-0a7ba40084f6d5b7b). It also lists the subnet group (banco-catalogofilmes\_sg-08124bc97b6f41e35) and security groups.

Banco de dados MySQL criado em subnet privada.

Conectado via Spring Boot com credenciais seguras no arquivo application.properties.

### API Gateway

The screenshot shows the AWS API Gateway console for the 'CatalogoFilmesAPI' API. The integrations section displays a single integration for the '/report' endpoint. It uses the 'ANY [proxy+]' method and 'HTTP ANY' as the target. The integration details show the URI as 'ANY http://54.198.176.65:8080/{proxy}' and the description 'v7vsrm'. The integration is currently active. The left sidebar shows the navigation menu for API Gateway, including Develop, Deploy, Monitor, and Protect sections.

The screenshot shows the AWS API Gateway Integrations page. On the left, there's a sidebar with sections like APIs, Develop, Monitor, Protect, and Deploy. The main area is titled 'Integrações' (Integrations) and shows 'Anexar integrações a rotas' (Attach integrations to routes). A table lists a single integration:

Rotas para CatalogoFilmesAPI	Detalhes da integração para a rota
ANY /report	URI HTTP: ANY http://54.198.176.65:8080/[proxy] ID de integração: v7lvsrm Descrição: Tempo limite: 30000 Mapeamento de parâmetros de solicitação: Não configurado Mapeamentos de parâmetros de resposta: Não configurado

Configurado com CORS habilitado.

/filmes → direciona para a API hospedada na EC2.

/report → chama a função Lambda que gera o relatório.

## 🧠 AWS Lambda

The screenshot shows the AWS Lambda Functions page. The top navigation bar includes 'Lambda', 'Funções', and 'reportFilmes'. The main area displays the function details:

- reportFilmes** (Lambda function)
- API Gateway** (Integration target)
- ARN da função**: arn:aws:lambda:us-east-1:337364215776:function:reportFilmes
- Última modificação**: há 12 horas
- Descrição**: -

Below this, the 'Código' tab is selected, showing the code editor with the file 'lambda\_function.py' open:

```

 1 import json
 2 import urllib.request
 3
 4 def lambda_handler(event, context):
 5     try:
 6         # URL da sua API Gateway
 7         API_URL = "https://7itkaw7gz0.execute-api.us-east-1.amazonaws.com/prod/filmes"
 8
 9         # Faz requisição GET para obter os filmes
10         with urllib.request.urlopen(API_URL) as response:
11             data = response.read().decode()
12             filmes = json.loads(data)
13

```

```

{
  "mensagem": "Relatório gerado com sucesso!",
  "relatorio": {
    "ano_mais_recente": 2021,
    "ano_mais_antigo": 1971,
    "media_avaliacao": 8.39
  }
}

```

Implementada em Python.

Faz requisições HTTP para a API /filmes e gera estatísticas JSON (média, total, mais recente).

## Amazon EC2

Name	ID da instância	Estado da instância	Tipo de instância	Verificação de status	Status do alarme	Zona de disponibilidade	DNS IPv4 público	Endereço IP privado	IP elástico
Catalogo_De_Filmes	i-05726f6df4e35e669	Executando	t2.medium	2/2 verificações a	Exibir alarmes	us-east-1d	ec2-54-198-176-65.co...	54.198.176.65	54.198.17...

**i-05726f6df4e35e669 (Catalogo\_De\_Filmes)**

```

sr-0dd53cbe9a9bf141b      80      TCP      0.0.0.0/0      launch-wizard-1
sr-0987b07b02b484aff      22      TCP      0.0.0.0/0      launch-wizard-1
sr-0699c2aef188d6980     8080      TCP      0.0.0.0/0      launch-wizard-1
sr-090ee89d65ea23b58      3306      TCP      sg-08124bc97b6f41e35      banco-catalogofilmes
sr-02faf64d99999284a      9091      TCP      0.0.0.0/0      banco-catalogofilmes

```

Last login: Fri Nov 2 03:17:38 2025 from 13.219.72.236  
[ec2-user@ip-172-31-19-27 ~]\$ docker ps

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
56a7f4646cf4	catalogo-filmes	"java -jar app.jar"	About an hour ago	Up About an hour	0.0.0.0:8080->8080/tcp, ::1:8080->8080/tcp	catalogo-filmes
56af374825e9	jenkins/jenkins:1.5	"/usr/bin/tini -- /usr/bin/jenkins.sh"	4 hours ago	Up 3 hours	0.0.0.0:50000->50000/tcp, ::1:50000->50000/tcp, 0.0.0.0:9091->9091/tcp, ::1:9091->9091/tcp	jenkins-unlocked

Catalogo de Filmes - Mackenzie Edition								<a href="#">Adicionar Filme</a>
<b>Shrek</b> Diretor: Andrew Adamson, Vicky Jenson Gênero: Animação, Comédia Ano: 2001 Avaliação: 9.1 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Interestelar</b> Diretor: Christopher Nolan Gênero: Ficção Científica Ano: 2014 Avaliação: 8.6 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Coringa</b> Diretor: Todd Phillips Gênero: Drama Ano: 2019 Avaliação: 8.4 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Forrest Gump</b> Diretor: Robert Zemeckis Gênero: Drama Ano: 1994 Avaliação: 8.3 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Matrix</b> Diretor: Lana e Lilly Wachowski Gênero: Ficção Científica Ano: 1999 Avaliação: 8.7 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>A Origem</b> Diretor: Christopher Nolan Gênero: Ficção Científica Ano: 2010 Avaliação: 8.8 <a href="#">Editar</a> <a href="#">Excluir</a>			
<b>O Poderoso Chefão</b> Diretor: Francis Ford Coppola Gênero: Drama Ano: 1972 Avaliação: 9.2 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Homem-Aranha: Sem Volta Para Casa</b> Diretor: Jon Watts Gênero: Ação Ano: 2021 Avaliação: 8.3 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Vingadores: Ultimato</b> Diretor: Anthony e Joe Russo Gênero: Ação Ano: 2019 Avaliação: 8.4 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>A Lista de Schindler</b> Diretor: Steven Spielberg Gênero: Drama Histórico Ano: 1993 Avaliação: 8.9 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Batman: O Cavaleiro das Trevas</b> Diretor: Christopher Nolan Gênero: Ação Ano: 2008 Avaliação: 9 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>O Senhor dos Anéis: O Retorno do Rei</b> Diretor: Peter Jackson Gênero: Fantasia Ano: 2003 Avaliação: 8.9 <a href="#">Editar</a> <a href="#">Excluir</a>			
<b>La La Land</b> Diretor: Damien Chazelle Gênero: Musical, Romance Ano: 2016 Avaliação: 8 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Titanic</b> Diretor: James Cameron Gênero: Drama, Romance Ano: 1997 Avaliação: 7.9 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Pantera Negra</b> Diretor: Ryan Coogler Gênero: Aventura Ano: 2018 Avaliação: 8.5 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Dragon Ball Super Broly</b> Diretor: Tatsuya Nagamine Gênero: Ação, Ficção científica Ano: 2019 Avaliação: 7.9 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>O Predador</b> Diretor: Shane Black Gênero: Ação, Ficção científica Ano: 2018 Avaliação: 5.3 <a href="#">Editar</a> <a href="#">Excluir</a>	<b>Alien, o 8.º Passageiro</b> Diretor: Andrew Adamson, Vicky Jenson Gênero: Terror, Ação Ano: 1997 Avaliação: 8.9 <a href="#">Editar</a> <a href="#">Excluir</a>			
<b>PROJETO DE NUVENS</b> Leonardo De Castro Tonon João Pedro Fernandes Milhomens Marcel Nobrega Zamboni								

Instância EC2 configurada com Docker.

Executa o backend com acesso público ao catálogo de filmes.

## 5. Atividades de Cada Membro

Leonardo De Castro Tonon – Desenvolvimento do backend, Dockerfile, deploy no EC2, pipeline Jenkins e video

João Pedro Fernandes Milhomens – Função AWS Lambda, configuração do API Gateway e Readme

Marcel Nobrega Zamboni – Criação e configuração do banco MySQL no RDS, regras de rede, permissões Documentação

## 6. Demonstração em Vídeo

Link do vídeo no YouTube:

<https://youtu.be/CrwvpVQhhBE>

## 7. Considerações Finais

O projeto Catálogo de Filmes – Mackenzie Edition integra diversos serviços AWS em uma solução completa e automatizada. Com o uso de EC2, RDS, API Gateway, Lambda e Jenkins, o grupo demonstrou domínio sobre o ciclo completo de desenvolvimento e implantação em nuvem.