

# AWS USER GROUP TRIÂNGULO MINEIRO

CloudCraft: Construindo Infraestrutura com AWS

AWS USER GROUP  
Triângulo Mineiro

# QUEM SOMOS?

## Confidence-building strategies

Este é um grupo para qualquer pessoa interessada em Amazon Web Services, Cloud Computing, Certificações, implementação e manutenção de arquitetura AWS, EC2, S3, High Scalability Computing, usos reais de AWS e servidores em nuvem, SaaS, IAAS, PAAS e muito mais. Todos os níveis de habilidades são bem-vindos. Iniciamos este grupo para conhecer outros entusiastas do Triângulo Mineiro. A região recebeu este nome justamente porque tem a forma de um triângulo e ao menos 1,2 milhão de habitantes vivem nas cidades do Triângulo Mineiro. Nosso objetivo é nos reunirmos regularmente para compartilhar ideias, cases, responder a perguntas e aprender sobre novos serviços e boas práticas da nuvem AWS.

Motivação

Conhecimento

Carreira

Certificação

Insights

**CONHECIMENTO** Compartilhar conhecimento, proporcionar reflexão e instigar os profissionais a procura por novidades na carreira;

**CARREIRA** Desenvolvimento de novas competências e conhecimentos dos serviços e soluções da AWS (SoftSkill);

**MOTIVAÇÃO** Integrar e motivar os profissionais da comunidade do Triângulo Mineiro;

**CERTIFICAÇÃO** Engajar a comunidade do Triângulo Mineiro na obtenção de Certificações AWS;

**INSIGHTS** Gerar novas ideias e oportunidades para expansão de novos negócios na região;

# RESULTADOS ESPERADOS



AWS USER GROUP  
Triângulo Mineiro



**NOSSA “CASA”**  
**ONDE DIGULGAMOS TODOS**  
**NOSSOS EVENTOS.**

<https://www.meetup.com/pt-BR/aws-user-group-triangulo-mineiro>

Olá, meu nome é Rogério Fontes, sou Arquiteto de Software especialista em Sistemas (CES), com uma trajetória que combina tecnologia, liderança e educação. Iniciei minha carreira desenvolvendo aplicações corporativas em Java e, ao longo dos anos, ampliei minha experiência para outras tecnologias, como Go e Cloud Computing (AWS/Azure), além de atuar na formação de novos talentos no mercado de tecnologia em Faculdades como UNA. Trabalhei em várias Empresas no mercado e Fintechs. Na NTTData atuei em projetos no setor de Banking (Bradesco/Itaú) . Saúde (Amil) e Telecom (Telefônica/Vivo). Hobbies:, Músico (Guitarra/Bateria), Games, Séries, Trabalhos Sociais.

## SOBRE MIM





# AMAZON WEB SERVICES (AWS)

A maior plataforma de nuvem do mundo, oferecendo mais de 200 serviços gerenciados.



# AMAZON WEB SERVICES (AWS)

## Pontos-chave:

- Lançada em 2006
- Alta disponibilidade global
- Pagamento sob demanda
- Segurança de nível empresarial
- Escalabilidade automática



# AMAZON WEB SERVICES (AWS)

## Categorias de serviços:

- Compute (EC2, Lambda, ECS, EKS)
- Storage (S3, EBS, EFS)
- Networking (VPC, Route 53, ELB)
- Databases (RDS, DynamoDB)





# AMAZON WEB SERVICES (AWS)

## Região AWS:

Conjunto de datacenters isolados geograficamente (ex: sa-east-1 – São Paulo).



# AMAZON WEB SERVICES (AWS)

## Availability Zones (AZs):

- 2 a 6 por região
- datacenters independentes, conectados em alta velocidade
- base da alta disponibilidade



# AMAZON WEB SERVICES (AWS)

## Edge Locations (CloudFront):

- Pontos locais de CDN
- Melhoram latência e distribuição de conteúdo
-



# AMAZON WEB SERVICES (AWS)

Resumo visual:



Região → várias AZs → cada AZ contém múltiplos datacenters



# AMAZON WEB SERVICES (AWS)

- **Virtual Private Cloud (VPC)**  
É sua rede privada dentro da AWS.

## Define:

- Faixa de IP (CIDR)
- Sub-redes (públicas/privadas)
- Rotas
- Gateways (IGW/NAT)
- Firewall (SG e NACL)



# AMAZON WEB SERVICES (AWS)

## Por que importa?

Controle total da arquitetura de rede

Segurança 100% configurável

Base para EC2, RDS, Lambda privada, EKS, MSK etc.

## Exemplo:

VPC 10.0.0.0/16

Subnet pública: 10.0.1.0/24

Subnet privada: 10.0.2.0/24

# AMAZON WEB SERVICES (AWS)

**O que é EC2? Amazon EC2 — Elastic Compute Cloud**  
Serviço de máquinas virtuais escaláveis.

## **Você escolhe:**

- Sistema operacional
- Tamanho (vCPU/RAM)
- Storage (EBS)
- Rede/Subnet
- IAM Role
- Security Group



# AMAZON WEB SERVICES (AWS)

O que é EC2?

## Casos de uso:

- Aplicações web
- Banco de dados
- APIs backend
- Processamento de dados
- Servidores de jogo
- Labs / provas de conceito





# AMAZON WEB SERVICES (AWS)

**VPC + EC2: Como funciona**

**EC2 em Subnet Pública:**

- Possui IP público
- Internet Gateway (IGW)
- Usada como Bastion, Web Server, API pública



# AMAZON WEB SERVICES (AWS)

## VPC + EC2: Como funciona

- **Fluxo típico:**  
Usuário → Internet → IGW → EC2 Pública → EC2 Privada  
EC2 Privada → NAT Gateway → Internet



# AMAZON WEB SERVICES (AWS)

**CIDR = Classless Inter-Domain Routing**

É a forma moderna de representar:

- um bloco de endereços IP
- o tamanho daquela rede
- quantos hosts ela suporta

**A notação é:** A.B.C.D /PREFIXO.

**Exemplo:** 10.0.0.0/16



# AMAZON WEB SERVICES (AWS)

**CIDR = Classless Inter-Domain Routing**

A parte /16 é a chave de tudo

🧠 A Regra de Ouro (a que muda tudo)

O número depois da barra indica quantos bits são reservados para a REDE.

👉 Quanto maior o número /XX → MENOR a rede

👉 Quanto menor o número /XX → MAIOR a rede

É contraintuitivo no começo, mas é isso que resolve TUDO..

# AMAZON WEB SERVICES (AWS)

**CIDR = Classless Inter-Domain Routing**

Imagine o CIDR como um condomínio:

O prefixo /16 diz:

“16 casas (endereços) são reservadas para a parte fixa do condomínio (rede)”

O resto (32 - prefixo) são as casas disponíveis para pessoas morarem (hosts).

# AMAZON WEB SERVICES (AWS)

**CIDR = Classless Inter-Domain Routing**

**Fica assim:**

Prefixo    Parte fixa (rede)    Parte variável (hosts)

/16          16 bits    16 bits

/24          24 bits    8 bits

/20          20 bits    12 bits



Fórmula para saber quantos IPs existem na rede

$2^{(32 - \text{prefixo})}$



# AMAZON WEB SERVICES (AWS)

**CIDR = Classless Inter-Domain Routing**

**Exemplos:**

$/16 \rightarrow 2^{(32-16)} = 65536 \text{ IPs}$

$/24 \rightarrow 2^{(32-24)} = 256 \text{ IPs}$

$/20 \rightarrow 2^{(32-20)} = 4096 \text{ IPs}$

👉 Nosso cérebro só precisa decorar os prefixos mais usados.



# AMAZON WEB SERVICES (AWS)

**CIDR = Classless Inter-Domain Routing**

**Exemplos:**

$/16 \rightarrow 2^{(32-16)} = 65536 \text{ IPs}$

$/24 \rightarrow 2^{(32-24)} = 256 \text{ IPs}$

$/20 \rightarrow 2^{(32-20)} = 4096 \text{ IPs}$

👉 Nosso cérebro só precisa decorar os prefixos mais usados.





# AMAZON WEB SERVICES (AWS)

**CIDR = Classless Inter-Domain Routing**

 CIDRs que você realmente usa na AWS

Aqui está a tabela definitiva da vida real (dica de arquiteto):

CIDR	Tamanho da rede /Hosts disponíveis	Tamanho	Quando usar
/16	65.536 endereços	enorme	VPC (AWS recomenda)
/20	4.096 endereços	grande	Muitas EC2 e autoscaling
/24	256 endereços	médio	Subnet padrão
/26	64 endereços	pequeno	Subnet para serviços específicos
/28	16 endereços	muito pequeno	NAT, endpoints, infra fixa



# AMAZON WEB SERVICES (AWS)

**CIDR = Classless Inter-Domain Routing**

**Macete de arquiteto:**

VPC → /16

Subnets → /24


Resolve 98% dos projetos.

.



# AMAZON WEB SERVICES (AWS)

**CIDR = Classless Inter-Domain Routing**

 Por que 10.0.0.0/16 é tão usado em VPC?

Porque ele te dá margem para dividir:

10.0.1.0/24

10.0.2.0/24

10.0.3.0/24

...

10.0.255.0/24



# AMAZON WEB SERVICES (AWS)

**Ou seja:**

- 👉 até 256 subnets /24 dentro de uma VPC.
- 👉 escalável, limpo, organizado.



# AMAZON WEB SERVICES (AWS)

 **Explicação Visual — Como “cortar” a VPC em Subnets**

**Vamos supor:**

VPC: 10.0.0.0/16

**A subnet pública:**

10.0.1.0/24

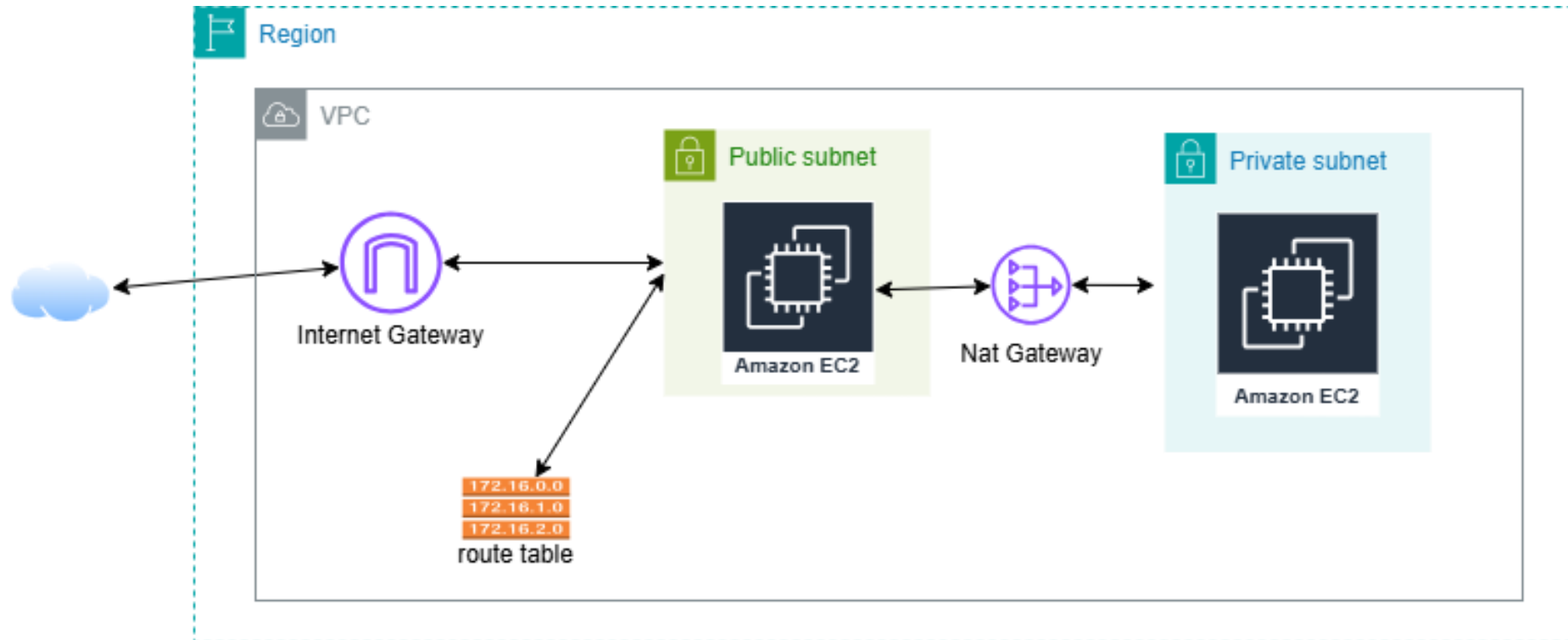
Você está “pegando emprestado bits do host” para criar subnets.

De /16 → para /24

Você está aumentando o prefixo de 16 para 24, então está diminuindo o número de IPs..



# AMAZON WEB SERVICES (AWS)





# AMAZON WEB SERVICES (AWS)



## Pré-requisitos:

- Ter conta AWS ativa
- Saber em qual **região** vai trabalhar (ex: us-east-1)
- Ter um **par de chaves SSH** ou criar um na hora do EC2



# AMAZON WEB SERVICES (AWS)



## PARTE 1 — Criar a VPC

### 1. Acessar a VPC

No Console AWS, procure por **VPC** no search bar.

Clique em **VPC** (serviço “Amazon VPC”).





# AMAZON WEB SERVICES (AWS)



## PARTE 2 — Criar a VPC

Menu lateral esquerdo → **Your VPCs** (Suas VPCs).

Clique em **Create VPC**.

Em **Resources to create**, escolha:

- ☒ **VPC only** (pra ensinar tudo na mão)

Em **Name tag**:

- demo--ufu-vpc -ec2

Em **IPv4 CIDR block**:

- 172.16.0.0/16



# AMAZON WEB SERVICES (AWS)



## PARTE 2 — Criar a VPC

Mantenha o resto padrão (IPv6 desabilitado se quiser simplificar).

Clique em **Create VPC**.

**Pronto:** você tem uma VPC vazia, sem subnet, sem rota, sem nada.



# AMAZON WEB SERVICES (AWS)



## PARTE 3 — Subnet Pública

No menu esquerdo → **Subnets**.

Clique em **Create subnet**.

Em **VPC ID**, selecione sua VPC: demo-vpc-vpc-ec2.

Em **Subnet name**:

- Demo-ufu-public-a



# AMAZON WEB SERVICES (AWS)



## PARTE 3 — Subnet Pública

Em **Availability Zone**:

- Escolha uma (ex: us-east-1a).

Em **IPv4 CIDR block**:

- 172.16.1.0/24

Clique em **Create subnet**.



# AMAZON WEB SERVICES (AWS)



## PARTE 3 — Subnet Pública

Habilitar IP público automático na subnet pública:

1. Na lista de subnets, selecione demo-public-a.
2. Clique em **Actions** → **Edit subnet settings** (ou “Editar configurações”).
3. Marque **Enable auto-assign public IPv4 address**.
4. Salvar.



# AMAZON WEB SERVICES (AWS)



## PARTE 4 — Subnet Privada

Habilitar IP público automático na subnet pública

De novo em **Create subnet**.

**VPC ID:** demo-ufu-vpc-ec2.

**Subnet name:** demo-ufu-private-b.

**Availability Zone:** outra, se quiser (ex: us-east-1b).



# AMAZON WEB SERVICES (AWS)



## PARTE 3 — Subnet Privada

**Habilitar IP público automático na subnet pública**

**IPv4 CIDR block:**

- 172.16.2.0/24

**Create subnet.**

(Não habilitar IP público nessa.)



# AMAZON WEB SERVICES (AWS)



## PARTE 4 — Criar e anexar o Internet Gateway (IGW)

1. Menu esquerdo → **Internet Gateways**.
2. Clique em **Create internet gateway**.
3. **Name tag**: demo-igw.
4. Clique em **Create internet gateway**.
5. Com ele selecionado, clique em **Actions** → **Attach to VPC**.
6. Selecione a VPC demo-vpc-vpc-ec2 → **Attach internet gateway**.





# AMAZON WEB SERVICES (AWS)



## PARTE 5 — Route Table da Subnet Pública

1. Menu esquerdo → **Route tables**.
2. Clique em **Create route table**.
3. **Name**: demo-public-rt.
4. **VPC**: demo-vpc-vpc-ec2.
5. **Create route table**.



# AMAZON WEB SERVICES (AWS)



## PARTE 6 — Adicionar rota para a internet

1. Selecione demo-ufu-public-rt.
2. Aba **Routes** → **Edit routes** → **Add route**.
3. Em **Destination**: 0.0.0.0/0.
4. Em **Target**: selecione **Internet Gateway** → demo-ufu-igw.
5. Clique em **Save changes**.



# AMAZON WEB SERVICES (AWS)



## PARTE 7 — Associar a subnet pública a essa route table

1. Ainda na demo-ufu-public-rt, vá na aba **Subnet associations**.
2. Clique em **Edit subnet associations**.
3. Marque a subnet-ufu-demo-public-a.
4. **Save associations**.
5. Agora: tudo na demo-ufu-public-a tem rota de saída para a internet.



# AMAZON WEB SERVICES (AWS)



## PARTE 8 — (Opcional) NAT Gateway para subnet privada

Se quiser ensinar **subnet privada com saída para internet**:

1. Menu → **NAT Gateways**.
2. **Create NAT gateway**.
3. **Name**: demo-ufu-nat-gw.
4. **Subnet**: escolha demo-ufu-public-a.
5. **Connectivity type**: Public.
6. Clique em **Allocate Elastic IP**.
7. Clique em **Create NAT gateway**.



# AMAZON WEB SERVICES (AWS)



## PARTE 8 — (Opcional) NAT Gateway para subnet privada

Depois, crie uma **route table** privada:

1. **Route tables** → **Create route table**.
2. Demo-ufu-private-rt → VPC demo-ufu-vpc-vpc-ec2.
3. Em **Routes** → Add route:
  1. Destination: 0.0.0.0/0
  2. Target: NAT Gateway → demo-nat-gw
4. Em **Subnet associations** → associe demo-ufu-private-b.



# AMAZON WEB SERVICES (AWS)

## PARTE 8 — (Opcional) NAT Gateway para subnet privada

Depois, crie uma **route table** privada:

1. **Route tables** → **Create route table**.
2. Demo-ufu-private-rt → VPC demo-ufu-vpc-vpc-ec2.
3. Em **Routes** → Add route:
  1. Destination: 0.0.0.0/0
  2. Target: NAT Gateway → demo-nat-gw
4. Em **Subnet associations** → associe demo-ufu-private-b.



# AMAZON WEB SERVICES (AWS)

Criando as VM's (EC2)



# AMAZON WEB SERVICES (AWS)



## PARTE 9 — Criar a EC2 na Subnet Pública

### 1. Acessar EC2

No Console AWS, procure por **EC2** e clique no serviço.

Clique em **Instances** → **Launch instances**.





# AMAZON WEB SERVICES (AWS)



## PARTE 9 — Criar a EC2 na Subnet Pública

### 2. Configurar Instância EC2

#### **Name and tags:**

- Name: demo-ec2-public.

#### **Application and OS Images (AMI):**

- Selecione **Amazon Linux 2023** (ou Amazon Linux 2).

#### **Instance type:**

- t2.micro (Free Tier, se estiver disponível).

#### **Key pair (login):**

- Escolha uma key existente ou crie uma (ex: demo-key).



# AMAZON WEB SERVICES (AWS)



## PARTE 9 — Criar a EC2 na Subnet Pública

### 3. Configurar Instância EC2

#### Network settings:

- **VPC:** selecione demo-vpc-vpc-ec2.
- **Subnet:** selecione demo-public-a.
- **Auto-assign public IP:** **Enable** (se não estiver automático).



# AMAZON WEB SERVICES (AWS)



## PARTE 9 — Criar a EC2 na Subnet Pública

### 4. Configurar Instância EC2

#### Firewall (security group):

- Escolha **Create security group**:
  - Name: demo-sg-ssh-http
  - Inbound rules:
    - **SSH** → Port 22 → Source: **My IP** (ou seu IP fixo)
    - **HTTP** → Port 80 → Source: 0.0.0.0/0 (pra demo de web).
- Outbound: deixe All traffic (padrão).

Clique em **Launch instance**.



# AMAZON WEB SERVICES (AWS)



## PARTE 10 — Conectar na EC2 (SSH)

Depois da instância ficar em **running**:

Clique na instância.

Copie o **Public IPv4 address**.

.



# AMAZON WEB SERVICES (AWS)



## PARTE 10 — Conectar na EC2 (SSH)

**Em Linux/macOS:**

No terminal:

```
chmod 400 demo-key.pem
```

```
ssh -i demo-key.pem ec2-user@SEU_IP_PUBLICO
```

Exemplo:

```
ssh -i demo-key.pem ec2-user@54.123.45.67
```

.



# AMAZON WEB SERVICES (AWS)



## PARTE 10 — Conectar na EC2 (SSH)

**Em Windows:**

Use o **Windows Terminal** com OpenSSH ou o **PuTTY** (convertendo a chave).

Com PowerShell (se tiver OpenSSH):

```
ssh -i .\demo-key.pem ec2-user@SEU_IP_PUBLICO
```

.



# AMAZON WEB SERVICES (AWS)



## PARTE 11 — Configurar o user\_data

### 4. Instalar NGINX pra testar HTTP

Já logado na EC2 (Amazon Linux):

```
sudo yum update -y          # opcional, mas bom pra aula
```

```
sudo yum install -y nginx
```

```
sudo systemctl enable nginx
```

```
sudo systemctl start nginx
```



# AMAZON WEB SERVICES (AWS)



## PARTE 11 — Configurar o user\_data

**Agora, no navegador:**

`http://SEU_IP_PUBLICO`

Se tudo estiver certo, vai aparecer a tela padrão do NGINX.





# AMAZON WEB SERVICES (AWS)



## PARTE 12 — (Opcional) EC2 na Subnet Privada + Bastion

**Se quiser fazer “modo arquiteto” completo na aula:**

1 - Criar uma segunda EC2 (demo-ec2-private) usando:

- VPC: demo-ufu-vpc-ec2
- Subnet: demo-ufu-private-b
- Sem IP público
- SG permitindo SSH **apenas** a partir do SG do bastion (demo-ufu-sg-ssh-http, por exemplo).

# AMAZON WEB SERVICES (AWS)

## PARTE 12 — (Opcional) EC2 na Subnet Privada + Bastion

Se quiser fazer “modo arquiteto” completo na aula:

2 - Usar a EC2 pública como **bastion host**:

- Conectar na pública via SSH.
- De dentro da pública, conectar na privada via IP privado.

Exemplo de comando:

```
ssh -i demo-key.pem ec2-user@10.0.2.10 # IP privado da EC2 privada
```



# AMAZON WEB SERVICES (AWS)



## Boas Práticas que você deve ensinar

- Nunca abrir SSH para 0.0.0.0/0
- Usar EC2 Connect
- Sempre separar subnets públicas e privadas
- EC2 backend sempre na privada
- Usar Load Balancer + Autoscaling
- Usar IAM Roles (sem Access Key na EC2!)
- Habilitar logs de VPC Flow Logs



# AMAZON WEB SERVICES (AWS)



## Boas Práticas que você deve ensinar

- Nunca abrir SSH para 0.0.0.0/0
- Usar EC2 Connect
- Sempre separar subnets públicas e privadas
- EC2 backend sempre na privada
- Usar Load Balancer + Autoscaling
- Usar IAM Roles (sem Access Key na EC2!)
- Habilitar logs de VPC Flow Logs



# AMAZON WEB SERVICES (AWS)

GitHub



# OBRIGADO



**Se interessou em nos ajudar, entre em contato no e-mail?**

`awsugtriangulomineiro@gmail.com`

