

# Documentação Técnica do projeto

## SimplePythonCRUD

### Introdução

Projeto desenvolvido por **Enricco Gemha** para a disciplina de **Computação em Nuvem** do curso de **Engenharia de Computação** do **Insper Instituto de Ensino e Pesquisa**.

Dado a especificação do projeto, foi desenvolvido um CRUD simples em Python utilizando os frameworks FastAPI e SQLAlchemy, e o banco de dados MySQL.

A aplicação foi hospedada na Amazon Web Services (AWS), sendo possível criar toda a infraestrutura necessária para a aplicação utilizando somente o script Terraform disponível neste repositório, bem como destruí-la completamente.

### Subindo a infraestrutura com Terraform

Para começar, é necessário ter uma conta na AWS e obter seu `access_key_id` e `secret_access_key`.

O próximo passo é instalar o AWS CLI, e configurar as credenciais de acesso no seu computador, para isso siga as instruções disponíveis [aqui](#).

Em seguida, é necessário instalar o Terraform, para isso siga as instruções disponíveis [aqui](#).

E configure o arquivo `terraform.tfvars` no `<path_to_this_project>/terraform` com as informações necessárias para a criação da infraestrutura. O arquivo deve ter o seguinte formato:

```
db_username = "<usuário>"  
db_password = "<senha>"
```

Assumindo que você esteja na raíz do repositório, execute os seguintes comandos:

```
cd terraform/bucket  
terraform init  
terraform validate  
terraform plan -out="tfplan"  
terraform apply "tfplan"
```

```
cd ..  
terraform init  
terraform validate  
terraform plan -out="tfplan"  
terraform apply "tfplan"
```

## Testando a aplicação

Ao final da execução do Terraform, serão exibidos no terminal, respectivamente, o ALB DNS name e o Locust Public IPv4 DNS.

Para testar a aplicação FastAPI, é necessário acessar o endereço da ALB no navegador. Para testar a aplicação Locust, é necessário acessar o endereço do Locust no navegador.

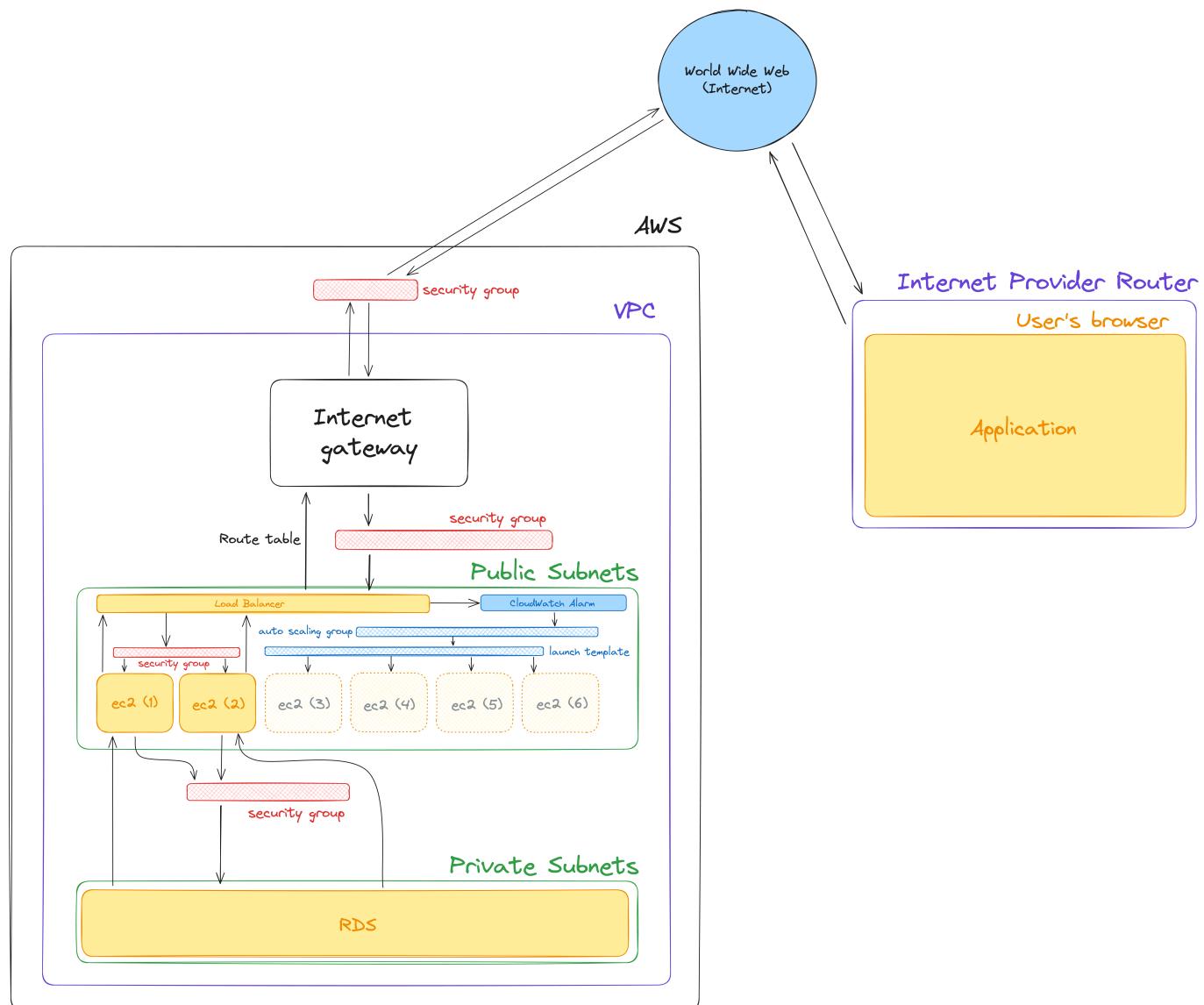
## Destruindo a infraestrutura com Terraform

Para destruir a infraestrutura, execute os seguintes comandos:

```
cd terraform/bucket  
aws s3 rm s3://enricco-terraform-state --recursive  
terraform destroy
```

```
cd ..  
terraform destroy
```

## Diagrama de infraestrutura



Neste diagrama, cada cor representa uma camada de abstração da infraestrutura, sendo **preto** a cor referente a camada de serviços AWS, ou seja, a camada de "cloud", em **roxo** a camada de rede, como o IPRouter e a VPC, em **vermelho** as camadas de segurança, representadas pelos Security Groups, em **verde** as camadas de subnets, pública e privada, em **azul** a camada de monitoramento, representada pelo CloudWatch, e em **laranja** a camada de aplicação, representada pelas instâncias EC2, o ALB e o banco de dados RDS, bem como a própria aplicação CRUD que é visível ao usuário. A World Wide Web é representada por um círculo em **azul**. As setas representam a comunicação entre os serviços.

## Documentação da aplicação

Serviços utilizados:

### VPC (Virtual Private Cloud)

A Virtual Private Cloud (VPC) foi criada para isolar a infraestrutura. A VPC possui CIDR 172.31.0.0/16. Assim, geramos 2 subnets privadas e 2 subnets públicas, sendo:

- Subnet privada 1:
  - CIDR: 172.31.0.0/26
  - Availability Zone: **eu-west-1a**;

- Subnet privada 2:
  - CIDR: 172.31.0.64/26
  - Availability Zone: **eu-west-1b**;
- Subnet pública 1:
  - CIDR: 172.31.0.128/26
  - Availability Zone: **eu-west-1a**;
- Subnet pública 2:
  - CIDR: 172.31.0.192/26
  - Availability Zone: **eu-west-1b**;

## Security Groups

Para garantir a segurança da aplicação, foram criados quatro Security Groups, permitindo somente os serviços necessários para o funcionamento da aplicação. São eles:

- ALB Security Group:
  - Entrada: permite tráfego HTTP (80) e SSH (22) de qualquer origem;
  - Saída: permite tráfego de qualquer protocolo para qualquer destino;
- EC2 Security Group:
  - Entrada: permite tráfego HTTP (80) proveniente do ALB Security Group e SSH (22) de qualquer origem;
  - Saída: permite tráfego de qualquer protocolo para qualquer destino;
- RDS Security Group:
  - Entrada: permite tráfego MySQL (3306) proveniente do EC2 Security Group;
  - Saída: permite tráfego de qualquer protocolo para qualquer destino;
- Locust Security Group:
  - Entrada: permite tráfego de qualquer protocolo para qualquer origem;
  - Saída: permite tráfego de qualquer protocolo para qualquer destino;

## IAM

Para garantir que as permissões adequadas fossem assinaladas para as instâncias EC2, foi criado um IAM Role, com permissões de escrita de logs, principalmente.

## RDS (Relational Database Service)

O RDS foi criado para hospedar o banco de dados MySQL. Portanto, sua engine é **mysql** na versão **8.0.33**, com **Multi-AZ** habilitado. Os backups são retidos por 7 dias, e com uma janela de manutenção semanal, às segundas-feiras, das 03:00 às 04:00, e de backup diário, das 04:00 às 05:00. O RDS possui uma instância **db.t2.micro** com 20GB de armazenamento **gp2**. Para garantir a segurança do acesso ao

banco de dados, os dados de usuário e senha são configurados em um arquivo `terraform.tfvars` e estão disponíveis somente neles.

## ALB (Application Load Balancer)

O ALB foi criado para balancear a carga entre as instâncias EC2. Ele está disponível publicamente na Internet. Ele tem um listener na porta 80 que encaminha o tráfego para o Target Group configurado (Instâncias EC2).

## ASG (Auto Scaling Group)

Mantém a quantidade de instâncias EC2 em 2, com um mínimo de 2 e um máximo de 6. A política de escalonamento é configurada para expandir a quantidade de instâncias quando a utilização de CPU atingir 70%, e reduzir a quantidade de instâncias quando a utilização de CPU atingir 20%. A política de Health Check é configurada para verificar a saúde das instâncias a cada 5 minutos, com um tempo de espera de 1 minuto, e um limite de 1 falha consecutiva.

## Decisões técnicas

- Para a aplicação, configurada em Ubuntu, foi utilizado Elastic Compute Cloud (EC2);
- Para banco de dados, foi hospedado no Relational Database Service (RDS);
- Para a comunicação entre os serviços, foi utilizado o serviço de Virtual Private Cloud (VPC);
- Para monitoramento de utilização, foram implementadas métricas e políticas utilizando o serviço de CloudWatch;
- Para balanceamento de carga, foi utilizado o serviço de Application Load Balancer (ALB);
- Para garantir a proteção da comunicação entre os serviços, foram criados *firewalls*, utilizando o serviço de Security Groups;
- Para garantir o redirecionamento para somente instância saudáveis, foram implementados Health Checks para o ALB;
- Para garantir a alta disponibilidade, foram criadas duas instâncias EC2 padrão, escaláveis até seis, através de uma Auto Scaling Group (ASG);
- Para garantir que não haja concorrência nas operações do Terraform, foi utilizado o serviço de Locking do DynamoDB, com armazenamento de estado no S3, garantindo o versionamento do código;
- Para garantir a segurança das instâncias, foi criado um IAM Role, com somente as permissões necessárias para a execução do script de instalação e configuração da aplicação.

## A escolha de região

Com regiões de disponibilidade em todo o mundo, vem também a necessidade de escolher uma região com requisitos que favoreça o desempenho e custo da aplicação. Esses benefícios são obtidos através de requisitos de:

- **Velocidade de Conexão** (Latência):
  - Por ser uma aplicação web que não será consumida como produto final, a latência é um fator opcional, pois não é um fator que influencia diretamente na experiência do usuário. Por isso **não foi um fator decisivo na escolha** da região;

- **Velocidade de Processamento:**

- Devido aos dados a serem processados serem pequenos e simples, a velocidade de processamento não impacta de forma relevante na aplicação, portanto **não foi um fator decisivo na escolha** da região;

- Disponibilidade de Serviços:

- A AWS possui uma grande variedade de serviços, e a maioria deles está disponível em todas as regiões, contudo existem restrições. Por exemplo, o **t2.micro**, escolhida por ser a opção *low-cost* e *general purpose* da AWS, está **disponível somente** em:
  - **us-east-1** (N. Virginia);
  - **us-west-2** (Oregon);
  - **eu-west-1** (Ireland);
  - **ap-northeast-1** (Tokyo);
  - **ap-southeast-1** (Singapore);
  - **ap-southeast-2** (Sydney);
  - **sa-east-1** (São Paulo);
- Há também a necessidade de se descartar as regiões com outages mais frequentes, e como podemos ver nesta [thread da YCombinator](#) (Aceleradora de Startups de Silicon Valley), a região **us-east-1** (N. Virginia) é explicitamente não recomendada, pois possui um histórico de outages mais frequentes, bem como equipamento ultrapassado. Ainda no mesmo tópico, o site [AWSManiac](#) menciona as seguintes regiões como as de maior quantidade de outages da história da AWS, nesta ordem:
  - **us-east-1** (N. Virginia);
  - **ap-southeast-2** (Sydney);
  - **ap-northeast-1** (Tokyo);
- O site [StatusGator](#) oferece uma lista com as regiões com maior tempo de downtimes parciais em 2022, sendo, nesta ordem, as três piores:
  - **us-east-1** (N. Virginia);
  - **us-west-2** (Oregon);
  - **us-east-2** (Ohio);

- Custo de Serviços:

- O custo de serviços é um fator importante em qualquer aplicação, e como o objetivo deste projeto é criar uma aplicação de baixo custo, é necessário escolher uma região que ofereça os serviços necessários com o menor custo possível. Os dados são o site [ConcurrencyLabs](#), com dados extraídos da AWS PriceList API. Portanto, da lista da **t2.micro** acima, podemos observar os seguintes preços (porcentagem de diferença em relação a **us-east-1**, a mais barata):
  - [0%] **us-east-1** (N. Virginia);
  - [0%] **us-west-2** (Oregon);
  - [11%] **eu-west-1** (Ireland);
  - [22%] **ap-northeast-1** (Tokyo);
  - [14%] **ap-southeast-1** (Singapore);
  - [26%] **ap-southeast-2** (Sydney);
  - [52%] **sa-east-1** (São Paulo);

Baseado nesses requisitos, excluímos todas as regiões que não possuem **t2.micro**. Em seguida, excluímos a região **us-east-1** (N. Virginia) pela imensa quantidade de outages. Excluímos **ap-southeast-2** (Sydney), **ap-northeast-1** (Tokyo) e **sa-east-1** (São Paulo) por possuírem uma grande diferença de custo em relação a **us-east-1** (N. Virginia). E por fim, excluímos **us-west-2** (Oregon) por possuir um histórico de outages, apesar de ser a segunda região mais barata. Com isso, ficamos em um empate entre **eu-west-1** (Ireland) e **ap-southeast-1** (Singapore), e como a região **eu-west-1** (Ireland) possui um custo 11% menor, foi a escolhida para hospedar a aplicação.

## A escolha de monitoramento

O projeto utiliza o CloudWatch para monitorar as instâncias EC2 e o RDS. Métricas essenciais, como Utilização de CPU e Contagem de Requisições da ALB, são monitoradas. Para a utilização de CPU, políticas de escalonamento são definidas em 70% para acionar a expansão e 20% para a redução, garantindo eficiência financeira de recursos. Da mesma forma, a métrica de Contagem de Requisições da ALB é configurada com limites 150 requisições, em um intervalo de espera de 5 minutos para evitar uma redução rápida de escala.

## A escolha de instâncias

O projeto utiliza a **t2.micro** para implantação de instâncias EC2. Por ser uma aplicação CRUD, essa configuração de baixo custo provisona recursos suficientes para lidar com essas operações básicas. Isso contribui para maximizar a eficiência financeira do projeto.

## A escolha do banco de dados

O projeto utiliza a **db.t2.micro** para o RDS, que é ótima para operações CRUD. Optamos pela implantação em Multi-Availability Zone para garantir alta disponibilidade, bem como tolerância a falhas. Por fim, optamos pelo General Purpose SSD (GP2) com capacidade de 20GB que dá uma ótima margem para necessidades de armazenamento do projeto, que cobre uma possível escalada de requisitos do projeto.

## Estimativa de custo de manutenção mensal

Para realizar uma estimativa de custos, foi utilizado o [AWS Pricing Calculator](#). Os custos foram estimados para um período de 1 mês, e os valores foram convertidos para Reais utilizando a cotação do dólar do dia 03/12/2023, de R\$4,92, de acordo com o [Banco Central do Brasil](#).

O valor total estimado para o período de 1 mês foi de **\$73.71**, ou seja, **R\$362,65**. O resultado da calculadora de custo já com os valores de cada serviço configurado está disponível publicamente [nesta link](#), ou no PDF dentro do repositório, no caminho [/docs/AWS Pricing Calculator](#). Abaixo, temos um resumo da configuração utilizada para a estimativa de custos:

### Amazon Virtual Private Cloud (VPC)

Parâmetros:

- Região: **eu-west-1** (Ireland);
- VPC services: **Data Transfer**;
- Number of VPN Connections: **1**;
- Data Transfer Intra-region (GB): **1**;

- Data Transfer All other regions (GB): **1**;
- Data Transfer Out to Internet (GB): **1**;

## Amazon RDS for MySQL

Parâmetros:

- Região: **eu-west-1** (Ireland);
- Quantidade de instâncias: **1**;
- Tipo de instância: **db.t2.micro**;
- Utilização: **On-Demand (100%)**;
- Deployment options: **Multi-AZ**;
- Storage: **General Purpose SSD (gp2)**;
- Storage (GB): **20**.

## Amazon EC2

Parâmetros:

- Região: **eu-west-1** (Ireland);
- Tipo de instância: **t2.micro**;
- Tenancy: **Shared**;
- Operating System: **Linux**;
- Workloads: **Daily spike traffic**;
- Workload (days): **Monday to Friday**;
- Baseline (instances): **2**;
- Peak (instances): **6**;
- Duration of peak (hours): **6**;
- Payment option: **EC2 Instance Savings Plans (1 Year, No Upfront)**.

## Elastic Load Balancing

Parâmetros:

- Região: **eu-west-1** (Ireland);
- Tipo de load balancer: **Application Load Balancer**;
- Features: **Load Balancer on Outposts**;
- Número de ALBs: **1**.

## Amazon Simple Storage Service (S3)

Parâmetros:

- Região: **eu-west-1** (Ireland);
- S3 Storage Class: **Standard**;
- Storage (GB): **0.01**;
- Requests: **10**;
- Data Returned by S3 Select (GB): **0.001**;
- Data Scanned by S3 Select (GB): **0.01**.

## Amazon DynamoDB

Parâmetros:

- Região: eu-west-1 (Ireland);
- Features: DynamoDB Data Import from Amazon S3 feature;
- Source file size (GB): 0.01;

## Amazon API Gateway

Parâmetros:

- Região: eu-west-1 (Ireland);
- API Type: REST API;
- Request units: millions;
- Requests per month: 1;

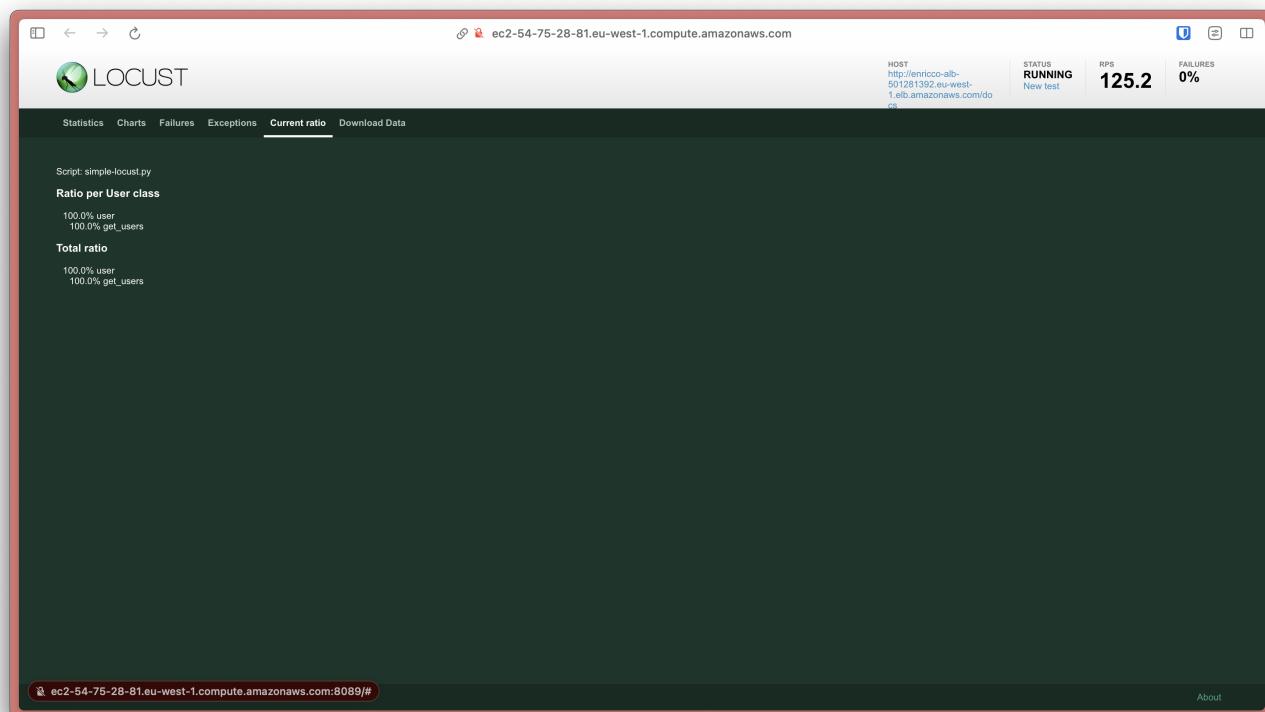
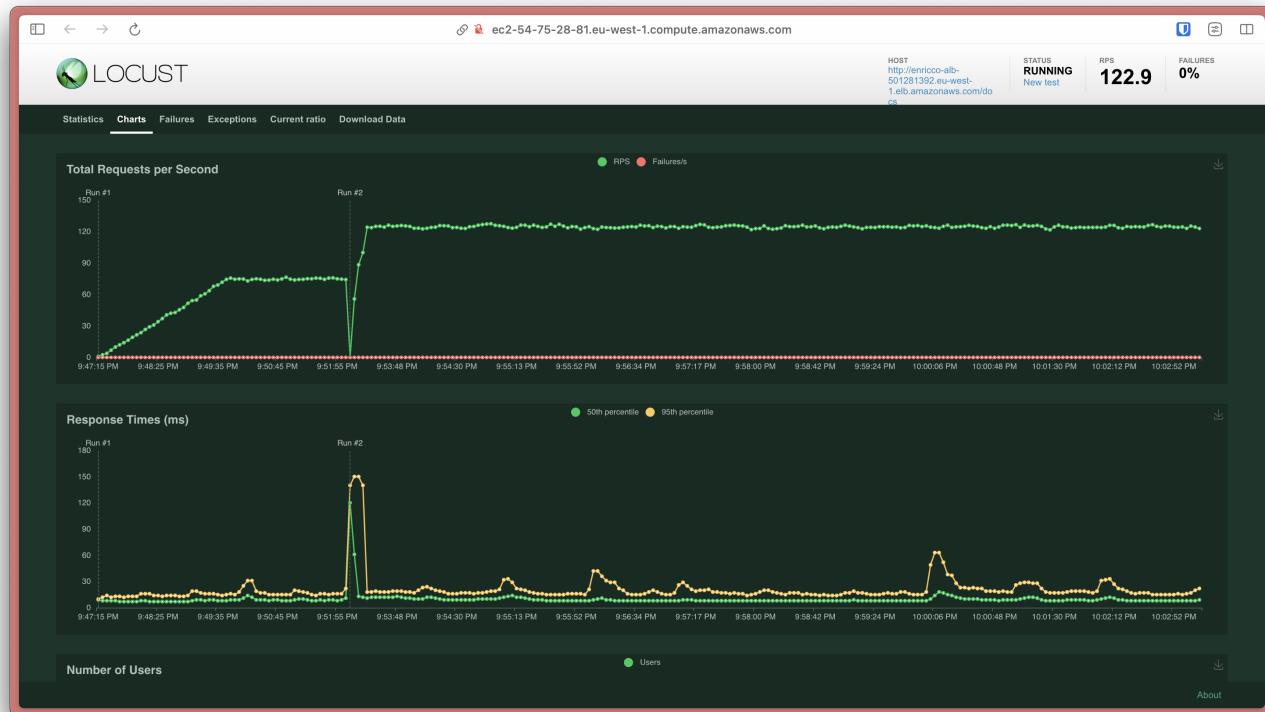
## Amazon CloudWatch

Parâmetros:

- Região: eu-west-1 (Ireland);
- Number of metrics: 2;
- Number of Standard Resolution Alarm Metrics: 2;

## Custo real utilizando o Locust para testes de carga

Para realizar os testes de carga, foi utilizado o Locust, uma ferramenta de código aberto para testes de carga. Sua utilização foi tão somente acessar o endereço no navegador e configurar um teste de carga de 250 usuários, com 50 usuários por segundo, e um tempo de execução de 10 minutos. O resultado do teste pode ser visto nas imagens abaixo:



Isso resulta na execução da Policy estabelecida para o ALB, que pode ser vista no dashboard depois de pouco tempo de execução do teste:

The screenshot shows the AWS EC2 Instances page with 7 instances listed:

| Name              | Instance ID         | Instance state | Instance type | Status check      | Alarm status | Availability Zone | Public IPv4 DNS          | Public IPv6 DNS |
|-------------------|---------------------|----------------|---------------|-------------------|--------------|-------------------|--------------------------|-----------------|
| enrico-launch-... | i-0ff191507b86e113d | Running        | t2.micro      | Initializing      | No alarms    | eu-west-1b        | ec2-54-154-115-17.eu...  | 54.154.115.17   |
| enrico-launch-... | i-0ec7feebf75fa1f8c | Running        | t2.micro      | Initializing      | No alarms    | eu-west-1b        | ec2-34-242-110-177.eu... | 34.242.110.177  |
| enrico-launch-... | i-08867f23b7f97ec39 | Running        | t2.micro      | 2/2 checks passed | No alarms    | eu-west-1b        | ec2-54-154-66-154.eu...  | 54.154.66.154   |
| enrico-launch-... | i-0b067ae9f2444b5a7 | Running        | t2.micro      | 2/2 checks passed | No alarms    | eu-west-1a        | ec2-3-250-215-185.eu...  | 3.250.215.185   |
| enrico-locust     | i-08ff6e920d9ca0229 | Running        | t2.micro      | 2/2 checks passed | No alarms    | eu-west-1a        | ec2-54-75-28-81.eu.we... | 54.75.28.81     |
| enrico-launch-... | i-0b0919199f7ab705  | Running        | t2.micro      | Initializing      | No alarms    | eu-west-1a        | ec2-34-240-216-145.eu... | 34.240.216.145  |
| enrico-launch-... | i-05d16e19ee5e6733  | Running        | t2.micro      | Initializing      | No alarms    | eu-west-1a        | ec2-3-252-167-14.eu.w... | 3.252.167.14    |

Depois do fim do teste carga, podemos ver que foi bem sucedido o downscaling das instâncias EC2:

The screenshot shows the AWS EC2 Instances page with 3 instances listed, all filtered to be 'running':

| Name                   | Instance ID         | Instance state | Instance type | Status check      | Alarm status | Availability Zone | Public IPv4 DNS          |
|------------------------|---------------------|----------------|---------------|-------------------|--------------|-------------------|--------------------------|
| enrico-launch-template | i-08867f23b7f97ec39 | Running        | t2.micro      | 2/2 checks passed | No alarms    | eu-west-1b        | ec2-54-154-66-154.eu...  |
| enrico-launch-template | i-0b46888f22049af15 | Running        | t2.micro      | 2/2 checks passed | No alarms    | eu-west-1a        | ec2-34-250-29-107.eu...  |
| enrico-locust          | i-08ff6e920d9ca0229 | Running        | t2.micro      | 2/2 checks passed | No alarms    | eu-west-1a        | ec2-54-75-28-81.eu.we... |

Para calcular o custo real de manutenção mensal, foi utilizado o [AWS Billing and Cost Management](#).

Como forma de validar a estimativa da Calculadora de Custos AWS, levamos em conta o tempo em que o teste carga rodou (10 minutos), com uma taxa de swarm do Locust de aproximadamente 122 RPS. Assim, observamos o painel de custos da região **eu-west-1** (Ireland), que pode ser visto abaixo:

The screenshot shows the AWS Billing and Cost Management console with a search filter applied for "Region name = EU (Ireland)". The results table lists the following items:

| Description                 | Usage Quantity | Amount in USD   |
|-----------------------------|----------------|-----------------|
| Elastic Compute Cloud       |                | USD 15.86       |
| EU (Ireland)                |                | USD 1.96        |
| Relational Database Service |                | USD 5.88        |
| EU (Ireland)                |                | USD 0.64        |
| Elastic Load Balancing      |                | USD 3.53        |
| EU (Ireland)                |                | USD 0.40        |
| CloudWatch                  |                | USD 0.06        |
| EU (Ireland)                |                | USD 0.00        |
| Data Transfer               |                | USD 0.01        |
| EU (Ireland)                |                | USD 0.00        |
| Simple Storage Service      |                | USD 0.00        |
| EU (Ireland)                |                | USD 0.00        |
| DynamoDB                    |                | USD 0.00        |
| EU (Ireland)                |                | USD 0.00        |
| Virtual Private Cloud       |                | USD 0.00        |
| EU (Ireland)                |                | USD 0.00        |
| <b>Total tax</b>            |                | <b>USD 3.51</b> |

Contudo é necessário deduzir os custos não relacionados ao swarm do Locust, neste caso os valores contornados em vermelho na imagem abaixo:

The screenshot shows the AWS Billing and Cost Management console with a search filter applied for "Region name = EU (Ireland)". The results table lists the following items, with several specific cost items highlighted with red boxes:

| Description                                                                         | Usage Quantity | Amount in USD   |
|-------------------------------------------------------------------------------------|----------------|-----------------|
| Elastic Compute Cloud                                                               |                | USD 15.86       |
| EU (Ireland)                                                                        |                | USD 1.96        |
| Amazon Elastic Compute Cloud NatGateway                                             | 0 GB           | USD 0.00        |
| \$0.048 per GB Data Processed by NAT Gateways                                       | 32 Hrs         | <b>USD 1.54</b> |
| Amazon Elastic Compute Cloud running Linux/UNIX                                     | 30.615 Hrs     | <b>USD 0.39</b> |
| \$0.0126 per On Demand Linux t2.micro Instance Hour                                 |                | <b>USD 0.39</b> |
| EBS                                                                                 |                | USD 0.04        |
| \$0.05 per GB-Month of snapshot data stored - EU (Ireland)                          | 0.045 GB-Mo    | USD 0.00        |
| \$0.11 per GB-month of General Purpose SSD (gp2) provisioned storage - EU (Ireland) | 0.34 GB-Mo     | USD 0.04        |
| Relational Database Service                                                         |                | USD 5.88        |
| EU (Ireland)                                                                        |                | USD 0.64        |
| Amazon Relational Database Service for MySQL Community Edition                      |                | USD 0.54        |
| \$0.018 per RDS db.t2.micro instance hour (or partial hour) running MySQL           | 0.025 Hrs      | USD 0.00        |
| \$0.036 per RDS db.t2.micro Multi-AZ instance hour (or partial hour) running MySQL  | 14.975 Hrs     | <b>USD 0.54</b> |
| Amazon Relational Database Service Provisioned Storage                              |                | USD 0.10        |
| Elastic Load Balancing                                                              |                | USD 3.53        |
| EU (Ireland)                                                                        |                | USD 0.40        |

Portanto, do total de **\$3.51**, devem ser descontados:

- 10 minutos representam aproximadamente 1.11% de 15 horas, portanto 11% sobre \$0.54, resulta em desconto de **\$0,48**;

- 10 minutos representam aproximadamente 0.52% de 32 horas, portanto 0.52% sobre \$1.54, resulta em desconto de **\$1.53**;
- 10 minutos representam aproximadamente 0.56% de 30 horas, portanto 0.56% sobre \$0.39, resulta em desconto de **\$0.389**.

Assim, o custo real de manutenção em 10 minutos de teste de carga é de **\$1.111**. Vale ressaltar que esse é um fluxo muito atípico e que representa o equivalente ao fluxo de 1 dia da aplicação no mundo real. Assim, podemos extrapolar isso para a aplicação rodando todos os dias da semana, o custo real de manutenção mensal é de **\$33,33**, ou seja, **R\$163,99**.