

Relatório de Entrega Técnica

Documento de Serviços

Nome do(a) Aluno(a): Miguel Cabral de Carvalho (mcc2@cesar.school) e Nathália Vasconcelos Couto (nvc@cesar.school)

Título do Projeto/Demanda: Fórmula Combustível

Data: 03/10/2024

Sumário

Sumário	2
Descrição geral da demanda técnica	3
Premissas	3
Considerações limitantes	3
Problemas atuais do contexto	5
Proposta de solução preliminar	6
Diagrama de Arquitetura	11
Anexos e conteúdos relacionados	12

Descrição geral da demanda técnica

O projeto é voltado para a modernização e otimização dos processos de gestão do posto de gasolina **Fórmula Combustível**. Recentemente, o posto enfrentou problemas significativos relacionados ao abastecimento e à gestão de estoque, resultando em excessos de demanda por gasolina e perda de eficiência operacional.

Com o aumento da demanda e a complexidade das operações, tornou-se evidente que a infraestrutura atual não atende às necessidades do posto. Para resolver esses problemas, o projeto propõe a implementação de uma solução em nuvem utilizando Amazon Web Services (AWS), que permitirá o monitoramento em tempo real dos níveis de combustível, gestão automatizada de estoques e relatórios detalhados sobre o consumo de gasolina.

Premissas

Disponibilidade da Conexão de Internet:

Assume-se que o Posto de Gasolina Fórmula Combustível terá uma conexão de internet estável e de alta velocidade, necessária para a operação eficiente da solução em nuvem da AWS e para a comunicação dos dispositivos IoT.

Capacidade de Armazenamento e Processamento:

O sistema deve ser dimensionado para suportar o volume atual e futuro de dados gerados pelos sensores IoT e pelos registros de estoque, com capacidade de armazenamento escalável através do Amazon S3 e do DynamoDB.

Manutenção de Hardware e Software:

A equipe do posto se compromete a realizar a manutenção regular dos sensores e do hardware associado para garantir que os dados enviados para a nuvem sejam precisos e confiáveis.

Treinamento da Equipe:

Assume-se que a equipe do Posto de Gasolina Fórmula receberá treinamento adequado para utilizar a nova solução, garantindo uma transição suave e a maximização dos benefícios do sistema.

Suporte Contínuo da AWS:

A solução se baseia no suporte contínuo da AWS para atualizações e manutenções da infraestrutura de nuvem.

Suporte Contínuo da AWS:

A solução precisa utilizar de uso de imagens virtuais Docker

Considerações limitantes

Configurações de Capacidade:

O dimensionamento da solução deve considerar picos de demanda, como feriados ou eventos locais, garantindo que a infraestrutura da AWS possa escalar adequadamente para suportar esses momentos sem degradação de desempenho.

Versões de Aplicações:

As versões das aplicações e das bibliotecas utilizadas devem ser compatíveis com as diretrizes e as práticas recomendadas da AWS, evitando conflitos que possam resultar em falhas operacionais ou vulnerabilidades de segurança.

Configurações Gerais de Segurança:

A solução deve implementar medidas de segurança robustas, incluindo criptografia de dados em trânsito e em repouso, além de políticas rigorosas de controle de acesso utilizando o AWS IAM, para proteger dados sensíveis contra acessos não autorizados.

Regulamentações e Conformidade:

O projeto deve estar em conformidade com as regulamentações locais e nacionais relacionadas ao armazenamento e manuseio de combustíveis, além de respeitar as diretrizes de proteção de dados, como a LGPD (Lei Geral de Proteção de Dados) no Brasil.

Interoperabilidade de Sistemas:

O sistema desenvolvido deve ser capaz de se integrar com soluções existentes no Posto de Gasolina Fórmula, como sistemas de gestão de vendas e contabilidade, garantindo que todos os dados relevantes sejam capturados e utilizados de maneira eficaz.

Limitações Orçamentárias:

O projeto deve ser realizado dentro de um orçamento pré definido, limitando a escolha de tecnologias e serviços dentro do escopo financeiro disponível.

Problemas atuais do contexto

O **Posto de Gasolina Fórmula** está enfrentando sérios problemas relacionados ao armazenamento excessivo de gasolina, resultado da falta de monitoramento em tempo real. Essa situação leva a decisões imprecisas sobre as quantidades a serem adquiridas e armazenadas, frequentemente resultando em falta de produto em estoque. Os gerentes não têm acesso imediato a dados sobre níveis de combustível, vendas e consumo, o que dificulta a tomada de decisões rápidas, especialmente durante picos de demanda.

A gestão manual do estoque é suscetível a erros, como registros incorretos e a falta de dados atualizados, o que pode levar ao desabastecimento em momentos críticos, impactando a satisfação do cliente e a receita do posto. Além disso, a ausência de análise de dados históricos e em tempo real impede que o posto preveja variações na demanda, como aquelas causadas por feriados ou eventos locais, contribuindo para uma gestão ineficiente dos estoques e potenciais perdas financeiras.

Outro desafio é a utilização de sistemas isolados para vendas, estoque e contabilidade, que dificulta uma visão integrada das operações. Essa falta de integração resulta em ineficiências, como a necessidade de realizar processos manuais repetidos e a dificuldade em gerar relatórios consolidados. Durante aumentos repentinos de demanda, como na alta temporada de viagens, a falta de um sistema automatizado impede uma resposta rápida e eficaz, levando a filas longas e clientes insatisfeitos.

Adicionalmente, a infraestrutura atual do posto não permite a escalabilidade necessária para suportar o crescimento do negócio. A capacidade de armazenamento e processamento de dados é limitada, afetando a eficiência operacional à medida que o volume de vendas aumenta. A geração manual e desatualizada de relatórios sobre vendas e consumo também dificulta a análise de desempenho e a identificação de oportunidades de melhoria.

Esses problemas destacam a urgência de uma solução que ofereça visibilidade em tempo real e permita uma gestão eficiente, garantindo a segurança e a satisfação dos clientes do Posto de Gasolina Fórmula.

Proposta de solução preliminar

Pensando em resolver o problema, utilizamos do terraform em conjunto com a AWS e Docker para realizar a codificação necessária.

VPC + Subnets

Primeiro, criamos a vpc com três subnets, uma pública e duas privadas. No código terraform ficou dessa forma

```

resource "aws_vpc" "main_vpc" {
  cidr_block = "172.31.0.0/16"
}

# Criação da Subnet Pública
resource "aws_subnet" "subnet_public" {
  vpc_id      = aws_vpc.main_vpc.id
  cidr_block  = "172.31.1.0/24"
  # sg
  map_public_ip_on_launch = true
}

output "subnet_public_id" {
  value = aws_subnet.subnet_public.id
}

# Criação da Subnet private
resource "aws_subnet" "subnet_private" {
  vpc_id      = aws_vpc.main_vpc.id
  cidr_block  = "172.31.2.0/24"
}

output "subnet_private_id" {
  value = aws_subnet.subnet_private.id
}

# Criação da Subnet private 2
resource "aws_subnet" "subnet_private2" {
  vpc_id      = aws_vpc.main_vpc.id
  cidr_block  = "172.31.3.0/24"
}

# criando o internet gateway
resource "aws_internet_gateway" "main_gw" {
  vpc_id = aws_vpc.main_vpc.id
}

# criando a rota para internet
resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.main_vpc.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.main_gw.id
  }
}

# associando a subnet publica com a rota publica
resource "aws_route_table_association" "public_subnet_assoc" {
  subnet_id      = aws_subnet.subnet_public.id
  route_table_id = aws_route_table.public_rt.id
}

output "aws_vpc_id" {
  value = aws_vpc.main_vpc.id
}

```

Security Groups

Depois provisionamos 4 security groups, um para a ec2 pública com a porta 80 liberada, outra para a instância ec2 privada com a porta 80 liberada para o sg da instância pública, outro que será utilizado pelo RDS com a porta 3306 liberada para o security group da ec2 privada e um para o ALB com a porta 80 liberada. Ficou dessa forma

```

module "vpc" {
  source = "../vpc" # caminho correto para o módulo VPC
}

resource "aws_security_group" "sg_public_ec2" {
  # vpc_id = aws_vpc.vpc.main_vpc.id
  vpc_id = module.vpc.aws_vpc_id

  ingress {
    from_port = 80
    to_port   = 80
    protocol  = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

output "sg_public_ec2_id" {
  value = aws_security_group.sg_public_ec2.id
}

resource "aws_security_group" "sg_private_ec2" {
  vpc_id = module.vpc.aws_vpc_id

  ingress {
    from_port = 80
    to_port   = 80
    protocol  = "tcp"
    security_groups = [aws_security_group.sg_public_ec2.id]
  }
}

output "sg_private_ec2_id" {
  value = aws_security_group.sg_private_ec2.id
}

resource "aws_security_group" "sg_rds" {
  vpc_id = module.vpc.aws_vpc_id

  ingress {
    from_port = 3306
    to_port   = 3306
    protocol  = "tcp"
    security_groups = [aws_security_group.sg_private_ec2.id]
  }
}

output "sg_rds_id" {
  value = aws_security_group.sg_rds.id
}

resource "aws_security_group" "sg_alb" {
  vpc_id = module.vpc.aws_vpc_id

  ingress {
    from_port = 80
    to_port   = 80
    protocol  = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

output "sg_alb_id" {
  value = aws_security_group.sg_alb.id
}

```

Auto Scaling group

Depois provisionamos o autoscaling group com no mínimo 1 e no máximo 2 instâncias t2.micro. Modificamos para que executasse um script que instala o docker e utiliza da imagem que criamos e disponibilizamos no docker hub. Ficando da seguinte maneira:


```

module "sg" {
  source = "../sg"
}

module "vpc" {
  source = "../vpc"
}

resource "aws_launch_template" "web_server_lt" {
  name_prefix   = "web-server-"
  image_id      = "ami-003932de22c285676" # ubuntu 20.04
  instance_type = "t2.micro"

  user_data = base64encode(<<-EOF
    #!/bin/bash
    yum update -y
    yum install -y docker
    systemctl start docker
    systemctl enable docker
    docker run -d -p 80:80 --name docker-app nathaliavc/getting-started:latest
  EOF
)

  network_interfaces {
    associate_public_ip_address = true
    security_groups             = [module.sg.sg_public_ec2_id]
  }
}

# Auto Scaling Group
resource "aws_autoscaling_group" "web_asg" {
  desired_capacity = 1
  max_size         = 2
  min_size         = 1
  health_check_type = "EC2"
  health_check_grace_period = 300

  vpc_zone_identifier = [module.vpc.subnet_public_id]

  launch_template {
    id      = aws_launch_template.web_server_lt.id
    version = "$Latest"
  }
}

resource "aws_autoscaling_policy" "scale_up" {
  name                = "scale-up"
  scaling_adjustment  = 1
  adjustment_type     = "ChangeInCapacity"
  autoscaling_group_name = aws_autoscaling_group.web_asg.name
}

resource "aws_autoscaling_policy" "scale_down" {
  name                = "scale-down"
  scaling_adjustment  = -1
  adjustment_type     = "ChangeInCapacity"
  autoscaling_group_name = aws_autoscaling_group.web_asg.name
}

```

Application Load Balancer

Depois fizemos um ALB apontando para o ASG

```

module "sg" {
  source = "../sg"
}

module "vpc" {
  source = "../vpc"
}

resource "aws_lb" "load_balancer_mcc2_nvc" {
  name                 = "my-app-lb"
  internal             = false
  load_balancer_type   = "application"
  security_groups      = [module.sg.sg_alb_id]
  subnets             = [module.vpc.subnet_public_id]

  enable_deletion_protection = false

  tags = {
    Name = "my-app-lb"
  }
}

resource "aws_lb_target_group" "target_group" {
  name     = "my-target-group"
  port     = 80
  protocol = "HTTP"
  vpc_id   = module.vpc.aws_vpc_id

  health_check {
    healthy_threshold   = 2
    unhealthy_threshold = 2
    timeout             = 5
    interval            = 30
    path               = "/"
    protocol            = "HTTP"
  }

  tags = {
    Name = "my-target-group"
  }
}

resource "aws_autoscaling_attachment" "asg_attachment" {
  autoscaling_group_id = aws_autoscaling_group.web_asg.id
  target_group_arn     = aws_lb_target_group.target_group.arn
}

resource "aws_lb_listener" "http_listener" {
  load_balancer_arn = aws_lb.load_balancer_mcc2_nvc.arn
  port             = 80
  protocol         = "HTTP"

  default_action {
    type = "forward"
    target_group_arn = aws_lb_target_group.target_group.arn
  }
}

```

Relational Database Service

Provisionamos uma instância RDS

```

module "sg"{
    source = "../sg"
}

resource "aws_db_instance" "mysql_db" {
    allocated_storage    = 20
    engine               = "mysql"
    engine_version       = "8.0"
    instance_class       = "db.t3.micro"
    username             = "admin"
    password             = "password"
    publicly_accessible  = false
    vpc_security_group_ids = [module.sg.sg_rds_id]

    backup_retention_period = 7
}

```

Arquivo Principal

E a main ficou dessa forma:

```

provider "aws" {
    region = "us-east-2"
}

module "vpc" {
    source = "../vpc/"
}
module "sg"{
    source = "../sg/"
}

module "asg" {
    source = "../asg/"
    # sg_public_ec2_id = module.sg.sg_public_ec2_id
    # subnet_public_id = module.vpc.subnet_public_id
}

```

Imagem docker

Usada para substituir a camada de aplicação.

Containers

Images

Volumes

Builds

Docker Scout

Extensions

Images / nathaliavc/getting-started:latest

<

nathaliavc/getting-started:latest

c251e61ea14e

CREATED

7 minutes ago

SIZE

344.57 MB

Recommended fixes

Run

Analyzed by docker scout

Give feedback

Layers (14)

Images

Vulnerabilities

Packages

0	ADD file:ee5bb8409915b11413f44cce4c22fed658a...	9.49 MB
1	CMD ["/bin/sh"]	0 B
2	ENV NODE_VERSION=18.20.4	0 B
3	RUN /bin/sh -c addgroup -g 1000 node && adduser ...	119.77 MB
4	ENV YARN_VERSION=1.22.19	0 B
5	RUN /bin/sh -c apk add --no-cache --virtual .build-de...	5.69 MB
6	COPY docker-entrypoint.sh /usr/local/bin/ # buildkit	20.48 KB
7	ENTRYPOINT ["docker-entrypoint.sh"]	0 B
8	CMD ["node"]	0 B
9	WORKDIR /app	8.19 KB
10	COPY . . # buildkit	6.69 MB
11	RUN /bin/sh -c yarn install --production # buildkit	117.08 MB
12	CMD ["node" "src/index.js"]	0 B
13	EXPOSE map[3000/tcp:{}]	0 B

This image has not been analyzed

You can use Docker Scout to analyze local images and list its vulnerabilities.

Start analysis

Enable background indexing in Settings

so your results are always ready.

Engine running

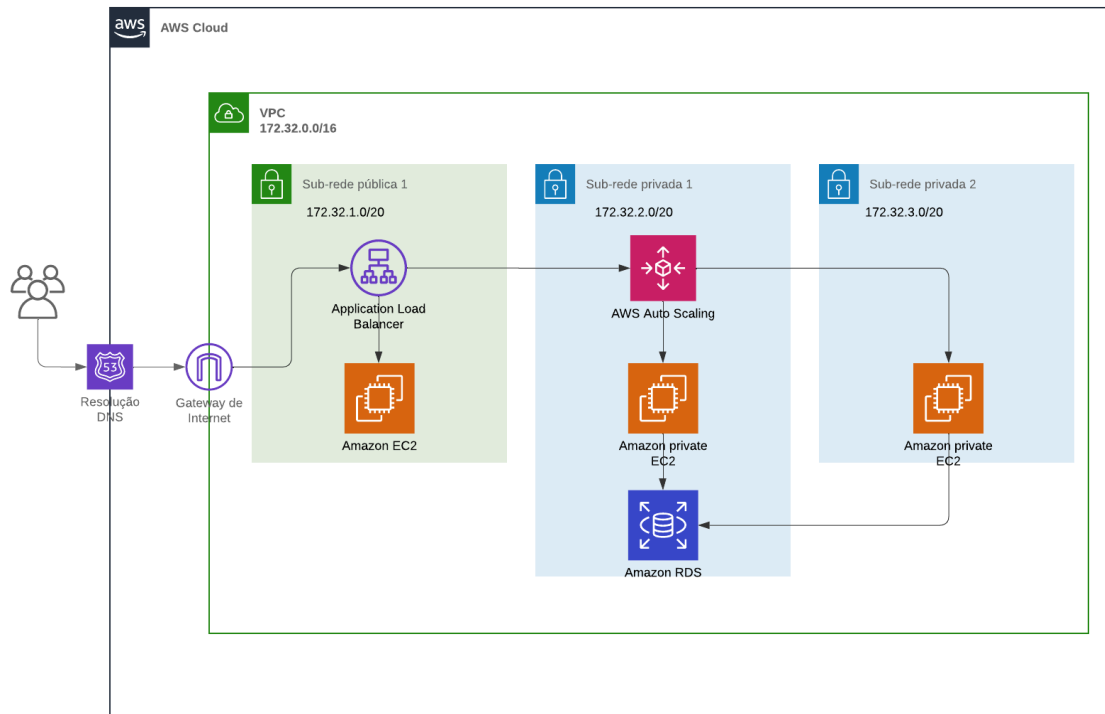
RAM 1.05 GB CPU -- % Disk 57.56 GB avail. of 62.67 GB

BETA

_ Terminal

✓ v4.34.2

Diagrama de Arquitetura



Anexos e conteúdos relacionados

Link do repositório: <https://github.com/Miguel-sdj/terraform-exercicio02>