**FACULDADE DE TECNOLOGIA "SHUNJI NISHIMURA" - POMPEIA**
Curso Superior de Tecnologia em Big Data no Agronegócio

**JONAH NISHIMURA KUNIHIRO**
**LUIS OTAVIO JASSI RODRIGUES**
**MATEUS ROBERS AMARAL**

# MARKET MANAGER

Sistema de Gerenciamento de Estoque com QR Code
Infraestrutura de Rede em Nuvem AWS

Pompeia - SP
2025

**JONAH NISHIMURA KUNIHIRO**
**LUIS OTAVIO JASSI RODRIGUES**
**MATEUS ROBERS AMARAL**

# MARKET MANAGER

Sistema de Gerenciamento de Estoque com QR Code

Infraestrutura de Rede em Nuvem AWS

Relatório Técnico apresentado à disciplina de Redes de Computadores do Curso Superior de Tecnologia em Big Data no Agronegócio da Faculdade de Tecnologia "Shunji Nishimura", como requisito parcial para aprovação.

Pompeia - SP

2025

# RESUMO

O presente relatório técnico descreve o desenvolvimento e implantação do sistema Market Manager, uma aplicação web para gerenciamento de estoque comercial, com ênfase na infraestrutura de redes de computadores utilizada para sua hospedagem na Amazon Web Services (AWS). O projeto foi desenvolvido utilizando tecnologias modernas de desenvolvimento web, incluindo React.js com TypeScript no front-end e PostgreSQL como banco de dados, sendo implantado em uma arquitetura de nuvem que utiliza diversos serviços AWS integrados. A infraestrutura de rede implementada contempla conceitos fundamentais de redes de computadores como endereçamento IP, sub-redes (VPC), balanceamento de carga, DNS, protocolos HTTP/HTTPS e segurança através de Security Groups e certificados SSL/TLS. O sistema oferece funcionalidades como cadastro de produtos com geração automática de QR Code, controle de movimentações de estoque, scanner de QR Code em tempo real e dashboard com gráficos interativos. A implantação na AWS demonstra na prática a aplicação de conceitos de redes em um ambiente de computação em nuvem, incluindo configuração de Virtual Private Cloud (VPC), roteamento de tráfego, alta disponibilidade através de múltiplas zonas de disponibilidade e integração de serviços através de endpoints de rede.

**Palavras-chave:** Redes de Computadores. AWS. Computação em Nuvem. VPC. Gerenciamento de Estoque. React.js.

# SUMÁRIO

# 1 INTRODUÇÃO

A evolução das redes de computadores e da computação em nuvem revolucionou a forma como sistemas de informação são desenvolvidos e implantados. Atualmente, aplicações web modernas dependem de uma infraestrutura de rede robusta e escalável para garantir disponibilidade, segurança e desempenho adequados aos usuários.

O gerenciamento de estoque comercial é uma atividade crítica para estabelecimentos de todos os portes. A falta de controle adequado pode resultar em perdas financeiras por excesso de mercadorias, ruptura de estoque ou vencimento de produtos. Neste contexto, sistemas informatizados hospedados em nuvem oferecem vantagens significativas como acesso remoto, escalabilidade automática e redução de custos com infraestrutura local.

O projeto Market Manager foi desenvolvido como uma solução completa de gerenciamento de estoque, utilizando tecnologias modernas de desenvolvimento web e implantado na infraestrutura de nuvem da Amazon Web Services (AWS). Este relatório apresenta não apenas as funcionalidades do sistema, mas principalmente a arquitetura de rede implementada, demonstrando na prática conceitos fundamentais de redes de computadores.

A hospedagem na AWS permite a aplicação de conceitos como Virtual Private Cloud (VPC), sub-redes públicas e privadas, grupos de segurança, balanceamento de carga, DNS e certificados SSL/TLS, oferecendo um ambiente didático para compreensão de como redes de computadores operam em ambientes de produção modernos.

# 2 OBJETIVOS

## 2.1 Objetivo Geral

Desenvolver e implantar um sistema web de gerenciamento de estoque na infraestrutura de nuvem AWS, aplicando conceitos de redes de computadores na configuração de uma arquitetura segura, escalável e de alta disponibilidade.

## 2.2 Objetivos Específicos

- Configurar uma Virtual Private Cloud (VPC) com sub-redes públicas e privadas;
- Implementar regras de Security Groups para controle de tráfego de rede;
- Configurar balanceamento de carga (Load Balancer) para distribuição de requisições;
- Implementar comunicação segura via HTTPS com certificados SSL/TLS;
- Configurar DNS através do Amazon Route 53 para resolução de nomes;
- Desenvolver módulo de cadastro de produtos com geração de QR Code;
- Implementar scanner de QR Code utilizando a câmera do dispositivo;
- Criar dashboard com gráficos para visualização de dados de estoque.

# 3 FUNDAMENTAÇÃO TEÓRICA

## 3.1 Redes de Computadores

Segundo Tanenbaum e Wetherall (2011), uma rede de computadores é um conjunto de computadores autônomos interconectados por uma única tecnologia. Esta interconexão permite o compartilhamento de recursos e a troca de informações entre os dispositivos conectados.

### 3.1.1 Modelo TCP/IP

O modelo TCP/IP (Transmission Control Protocol/Internet Protocol) é a arquitetura de rede utilizada na Internet e em redes corporativas. Organizado em quatro camadas (Aplicação, Transporte, Internet e Acesso à Rede), este modelo define como os dados são transmitidos entre dispositivos em uma rede.

### 3.1.2 Endereçamento IP

O endereço IP (Internet Protocol) é um identificador numérico único atribuído a cada dispositivo em uma rede. O IPv4 utiliza endereços de 32 bits, permitindo aproximadamente 4,3 bilhões de endereços únicos. A notação CIDR (Classless Inter-Domain Routing) permite a divisão de redes em sub-redes menores, fundamental para a organização de redes em ambientes de nuvem.

### 3.1.3 Protocolos HTTP e HTTPS

O HTTP (Hypertext Transfer Protocol) é o protocolo da camada de aplicação utilizado para comunicação na World Wide Web. Opera na porta 80 por padrão e utiliza o modelo requisição-resposta. O HTTPS adiciona uma camada de segurança através do protocolo TLS (Transport Layer Security), criptografando a comunicação e operando na porta 443.

### 3.1.4 DNS (Domain Name System)

O DNS é o sistema responsável pela tradução de nomes de domínio legíveis por humanos (como www.exemplo.com) em endereços IP numéricos. Este serviço é fundamental para a usabilidade da Internet, permitindo que usuários acessem recursos sem necessidade de memorizar endereços IP.

## 3.2 Computação em Nuvem e AWS

A computação em nuvem, segundo Veras (2012), é um modelo de fornecimento de recursos computacionais sob demanda através da Internet. A Amazon Web Services (AWS) é

a plataforma de nuvem líder de mercado, oferecendo mais de 200 serviços para computação, armazenamento, banco de dados e redes.

### 3.2.1 Virtual Private Cloud (VPC)

A VPC é uma rede virtual isolada dentro da infraestrutura AWS onde o usuário pode provisionar recursos. Permite controle total sobre o ambiente de rede, incluindo seleção de faixa de endereços IP, criação de sub-redes, configuração de tabelas de roteamento e gateways de rede.

### 3.2.2 Security Groups

Security Groups funcionam como firewalls virtuais para instâncias EC2, controlando o tráfego de entrada e saída. Operam no nível da instância e são stateful, ou seja, se uma conexão de entrada é permitida, a resposta correspondente é automaticamente permitida.

### 3.2.3 Elastic Load Balancing

O Elastic Load Balancing distribui automaticamente o tráfego de entrada entre múltiplos destinos (instâncias EC2, containers, endereços IP) em uma ou mais zonas de disponibilidade. Aumenta a tolerância a falhas e a disponibilidade da aplicação.

## 3.3 Tecnologias de Desenvolvimento Web

### 3.3.1 React.js e TypeScript

React.js é uma biblioteca JavaScript para construção de interfaces de usuário, desenvolvida pelo Facebook. TypeScript é um superconjunto tipado de JavaScript que adiciona verificação de tipos estáticos, melhorando a manutenibilidade do código.

### 3.3.2 PostgreSQL

PostgreSQL é um sistema de gerenciamento de banco de dados relacional objeto, conhecido por sua robustez e conformidade com padrões SQL. Na AWS, pode ser utilizado através do serviço RDS (Relational Database Service) ou Neon Database para implantação serverless.

# 4 METODOLOGIA

## 4.1 Metodologia de Desenvolvimento

O projeto foi desenvolvido utilizando metodologia ágil com sprints semanais. A equipe organizou o trabalho em três frentes principais: infraestrutura de rede AWS, desenvolvimento back-end e desenvolvimento front-end.

## 4.2 Divisão de Tarefas

**Jonah Nishimura Kunihiro** - Responsável pela infraestrutura de rede e AWS:

- Configuração da VPC com sub-redes públicas e privadas;
- Implementação de Security Groups e regras de firewall;
- Configuração do Application Load Balancer;
- Setup do Route 53 e certificados SSL via ACM.

**Luis Otavio Jassi Rodrigues** - Responsável pelo desenvolvimento front-end:

- Desenvolvimento dos componentes React e páginas da aplicação;
- Implementação da interface responsiva com Tailwind CSS;
- Desenvolvimento do Dashboard com gráficos Recharts;
- Implementação do scanner QR Code com html5-qrcode.

**Mateus Robers Amaral** - Responsável pelo back-end e banco de dados:

- Modelagem e implementação do schema PostgreSQL;
- Desenvolvimento da API REST para comunicação;
- Implementação de triggers e funções SQL;
- Configuração da conexão segura com o banco de dados via SSL.

# 5 INFRAESTRUTURA DE REDE NA AWS

## 5.1 Arquitetura de Rede

A arquitetura de rede do Market Manager foi projetada seguindo as melhores práticas da AWS para aplicações web, com separação clara entre camadas públicas e privadas, garantindo segurança e alta disponibilidade.

O diagrama conceitual da arquitetura inclui:

- **Internet Gateway:** Ponto de entrada para tráfego da Internet;
- **Application Load Balancer:** Distribuição de tráfego entre instâncias;
- **Sub-redes públicas:** Hospedagem do Load Balancer e NAT Gateway;
- **Sub-redes privadas:** Hospedagem das instâncias EC2 e banco de dados;
- **NAT Gateway:** Permite acesso à Internet para recursos em sub-redes privadas.

## 5.2 Virtual Private Cloud (VPC)

Foi configurada uma VPC dedicada para o projeto com as seguintes especificações:

### 5.2.1 Configuração da VPC

```
Nome: market-manager-vpc
CIDR Block: 10.0.0.0/16
Região: sa-east-1 (São Paulo)
Tenancy: Default
```

O bloco CIDR 10.0.0.0/16 permite até 65.536 endereços IP, suficientes para expansão futura do projeto.

### 5.2.2 Sub-redes

Foram criadas quatro sub-redes distribuídas em duas zonas de disponibilidade para garantir alta disponibilidade:

**Sub-redes Públicas:**

```
public-subnet-1a: 10.0.1.0/24 (AZ: sa-east-1a)
public-subnet-1b: 10.0.2.0/24 (AZ: sa-east-1b)
```

**Sub-redes Privadas:**

```
private-subnet-1a: 10.0.10.0/24 (AZ: sa-east-1a)
private-subnet-1b: 10.0.20.0/24 (AZ: sa-east-1b)
```

Cada sub-rede /24 permite 254 endereços IP utilizáveis (256 - 2 reservados para rede e broadcast).

### 5.2.3 Tabelas de Roteamento

Foram configuradas duas tabelas de roteamento:

**Tabela de Roteamento Pública:**

```
0.0.0.0/0 -> Internet Gateway (igw-xxxxx)
10.0.0.0/16 -> Local
```

**Tabela de Roteamento Privada:**

```
0.0.0.0/0 -> NAT Gateway (nat-xxxxx)
10.0.0.0/16 -> Local
```

## 5.3 Serviços AWS Utilizados

### 5.3.1 Amazon EC2 (Elastic Compute Cloud)

As instâncias EC2 hospedam a aplicação Node.js. Configuração utilizada:

```
Tipo de instância: t3.small
vCPUs: 2 | Memória: 2 GB
Sistema Operacional: Amazon Linux 2023
Armazenamento: EBS gp3 20GB
```

### 5.3.2 Application Load Balancer (ALB)

O ALB opera na camada 7 (aplicação) do modelo OSI, permitindo roteamento baseado em conteúdo:

- **Listener HTTP (porta 80):** Redireciona para HTTPS;
- **Listener HTTPS (porta 443):** Encaminha para Target Group;
- **Health Check:** Verifica /health a cada 30 segundos;
- **Cross-Zone Load Balancing:** Habilitado para distribuição uniforme.

### 5.3.3 Amazon RDS / Neon Database

O banco de dados PostgreSQL foi configurado com conexão SSL obrigatória:

```
Engine: PostgreSQL 15
Porta: 5432 (padrão PostgreSQL)
SSL Mode: require
Backup: Automático diário
```

### 5.3.4 Amazon Route 53

O Route 53 é o serviço de DNS da AWS utilizado para resolução de nomes:

- **Hosted Zone:** Zona pública para o domínio da aplicação;
- **Registro A (Alias):** Aponta para o Application Load Balancer;
- **TTL:** 300 segundos para propagação rápida de alterações.

### 5.3.5 AWS Certificate Manager (ACM)

Certificado SSL/TLS gratuito provisionado para comunicação HTTPS:

- **Tipo:** Certificado público validado por DNS;
- **Algoritmo:** RSA 2048 bits;
- **Renovação:** Automática pela AWS.

## 5.4 Segurança de Rede

### 5.4.1 Security Groups

Foram configurados três Security Groups com regras específicas:

**SG-ALB (Security Group do Load Balancer):**

```
Inbound:
  - HTTP (80) de 0.0.0.0/0
  - HTTPS (443) de 0.0.0.0/0
Outbound: All traffic
```

**SG-EC2 (Security Group das instâncias):**

```
Inbound:
  - HTTP (3000) de SG-ALB apenas
  - SSH (22) de IP específico (admin)
Outbound: All traffic
```

**SG-RDS (Security Group do banco de dados):**

```
Inbound:
  - PostgreSQL (5432) de SG-EC2 apenas
Outbound: None
```

Esta configuração implementa o princípio do menor privilégio, onde cada recurso só aceita conexões dos recursos que realmente precisam se comunicar com ele.

### 5.4.2 Network ACLs

As Network ACLs (NACLs) foram configuradas como camada adicional de segurança no nível da sub-rede. Diferente dos Security Groups que são stateful, as NACLs são stateless e requerem regras explícitas para tráfego de entrada e saída.

### 5.4.3 Criptografia em Trânsito

Toda comunicação é criptografada:

- **Cliente → ALB:** HTTPS com TLS 1.2/1.3;
- **EC2 → RDS:** SSL obrigatório na conexão PostgreSQL;
- **API externa:** Supabase utiliza HTTPS para todas as operações.

# 6 DESENVOLVIMENTO DO SISTEMA

## 6.1 Estrutura do Banco de Dados

O banco de dados PostgreSQL foi modelado com seis tabelas principais:

- **users:** Usuários com controle de papéis (admin/gestor);
- **categories:** Categorização de produtos;
- **suppliers:** Cadastro de fornecedores;
- **products:** Produtos com QR Code único, preços e estoque;
- **stock_movements:** Registro de entradas, saídas e ajustes;
- **audit_logs:** Logs de auditoria de todas as operações.

## 6.2 Módulos e Funcionalidades

### 6.2.1 Dashboard

Painel principal com gráficos interativos (Recharts) mostrando total de produtos, estoque baixo, produtos próximos ao vencimento, valor total do inventário e movimentações dos últimos 7 dias.

### 6.2.2 Scanner QR Code

Utiliza a biblioteca html5-qrcode para acesso à câmera do dispositivo. Permite consulta rápida de produtos e registro de movimentações diretamente pelo código escaneado.

### 6.2.3 Gestão de Produtos

CRUD completo de produtos com geração automática de QR Code usando react-qr-code. Inclui filtros por categoria, fornecedor e status, além de impressão em lote de QR Codes.

### 6.2.4 Movimentações de Estoque

Registro de entradas, saídas, ajustes e inventário com atualização automática do estoque via triggers SQL.

## 6.3 Comunicação Cliente-Servidor

A comunicação entre o cliente (browser) e o servidor segue o padrão REST (Representational State Transfer), utilizando os métodos HTTP:

- **GET:** Consulta de dados (produtos, categorias, movimentações);
- **POST:** Criação de novos registros;

- **PATCH:** Atualização parcial de registros;
- **DELETE:** Remoção de registros.

Todas as requisições trafegam sobre HTTPS (porta 443), garantindo confidencialidade e integridade dos dados. O formato de troca de dados é JSON (JavaScript Object Notation).

### 6.3.1 Fluxo de uma Requisição

O fluxo de rede de uma requisição típica:

1. Cliente resolve DNS via Route 53 (domínio → IP do ALB);
2. Requisição HTTPS chega ao Application Load Balancer;
3. ALB verifica certificado SSL e encaminha para instância EC2 saudável;
4. Aplicação Node.js processa a requisição;
5. Se necessário, conecta ao PostgreSQL via SSL (porta 5432);
6. Resposta retorna pelo mesmo caminho até o cliente.

# 7 RESULTADOS

O projeto Market Manager foi implantado com sucesso na infraestrutura AWS, demonstrando a aplicação prática de conceitos de redes de computadores. Os principais resultados alcançados:

- **Arquitetura de Rede Funcional:** VPC configurada com separação adequada entre sub-redes públicas e privadas, garantindo isolamento de recursos sensíveis;
- **Alta Disponibilidade:** Recursos distribuídos em duas zonas de disponibilidade, eliminando ponto único de falha;
- **Segurança em Camadas:** Security Groups e NACLs implementados seguindo o princípio do menor privilégio;
- **Comunicação Criptografada:** 100% do tráfego protegido por TLS/SSL;
- **Balanceamento de Carga:** ALB distribuindo tráfego eficientemente entre instâncias;
- **DNS Configurado:** Resolução de nomes funcionando corretamente via Route 53;
- **Aplicação Funcional:** Todas as funcionalidades do sistema operando corretamente (Dashboard, Scanner, Produtos, Movimentações).

# 8 CONCLUSÃO

O desenvolvimento e implantação do sistema Market Manager na infraestrutura AWS proporcionou uma experiência prática valiosa na aplicação de conceitos de redes de computadores em um ambiente de nuvem real.

A configuração da VPC permitiu compreender na prática o endereçamento IP, criação de sub-redes e tabelas de roteamento. A implementação de Security Groups demonstrou como firewalls virtuais controlam o fluxo de tráfego entre recursos. O uso do Application Load Balancer ilustrou conceitos de balanceamento de carga e operação na camada de aplicação do modelo OSI.

A configuração de DNS via Route 53 e certificados SSL via ACM mostrou como serviços de rede críticos para a Internet funcionam na prática. A comunicação entre os componentes da aplicação (cliente, servidor, banco de dados) demonstrou a aplicação de protocolos como HTTP/HTTPS e conexões TCP.

Como trabalhos futuros, sugere-se a implementação de um ambiente de disaster recovery em outra região AWS, configuração de VPN site-to-site para acesso administrativo seguro, implementação de WAF (Web Application Firewall) para proteção adicional, e configuração de CloudWatch para monitoramento detalhado de métricas de rede.

Conclui-se que o projeto alcançou seus objetivos, resultando em uma aplicação funcional hospedada em uma infraestrutura de rede robusta, segura e escalável, demonstrando a integração entre conhecimentos de desenvolvimento de software e redes de computadores.

# REFERÊNCIAS

AMAZON WEB SERVICES. *Amazon VPC User Guide*. Disponível em: <https://docs.aws.amazon.com/vpc/>. Acesso em: 15 nov. 2024.

AMAZON WEB SERVICES. *Elastic Load Balancing User Guide*. Disponível em: <https://docs.aws.amazon.com/elasticloadbalancing/>. Acesso em: 15 nov. 2024.

AMAZON WEB SERVICES. *AWS Well-Architected Framework*. Disponível em: <https://aws.amazon.com/architecture/well-architected/>. Acesso em: 10 nov. 2024.

KUROSE, James F.; ROSS, Keith W. *Redes de Computadores e a Internet: uma abordagem top-down*. 8. ed. São Paulo: Pearson, 2021.

META PLATFORMS. *React Documentation*. Disponível em: <https://react.dev>. Acesso em: 10 nov. 2024.

NEON. *Neon Documentation: Serverless Postgres*. Disponível em: <https://neon.tech/docs>. Acesso em: 08 nov. 2024.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. *PostgreSQL 16 Documentation*. Disponível em: <https://www.postgresql.org/docs>. Acesso em: 05 nov. 2024.

TANENBAUM, Andrew S.; WETHERALL, David J. *Redes de Computadores*. 5. ed. São Paulo: Pearson, 2011.

VERAS, Manoel. *Cloud Computing: Nova Arquitetura da TI*. Rio de Janeiro: Brasport, 2012.