

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

**NÚCLEO DE EDUCAÇÃO A DISTÂNCIA**

**Pós-graduação *Lato Sensu* em Arquitetura de Software Distribuído**

**Yago Henrique Ferreira**

**PLATAFORMA INTEGRADA DE COMÉRCIO ELETRÔNICO BASEADO EM  
DROPSHIPPING**

Belo Horizonte

2018

**Yago Henrique Ferreira**

**PLATAFORMA INTEGRADA DE COMÉRCIO ELETRÔNICO BASEADO EM  
DROPSHIPPING**

Trabalho de Conclusão de Curso de Especialização  
em Arquitetura de Software Distribuído como  
requisito parcial à obtenção do título de especialista.

Orientador: Tadeu dos Reis Faria

Belo Horizonte

2018

*“O sucesso, para mim, não se mede pela vitória e sim pela entrega, por fazer o seu melhor a cada dia. Na vida pessoal ou profissional, você jamais vai conquistar os resultados sem esforço e dedicação.”*

*(Rogério Ceni)*

## **AGRADECIMENTOS**

Agradeço a Deus por todas as bênçãos que me concede. Por me deixar de pé e não permitir que desistisse no percurso da elaboração desse trabalho. Agradeço aos meus pais e minha irmã, meus companheiros de jornada, meus incentivadores e meu motivo de viver e batalhar. Agradeço aos meus gatos, que me proporcionaram momentos de descontração durante a confecção deste trabalho (fora o fato de que algumas partes tiveram de ser reescritas devido ao sapateado dos gatos sobre o teclado de minha máquina). Agradeço aos amigos que fiz durante toda a jornada na profissão. Pessoas da mais alta qualidade, formadores de profissionais e de caráter. Em especial, agradeço ao amigo Milton César Rodrigues, com quem tenho profunda dúvida de gratidão, por ter sido o primeiro a acreditar em mim como programador, e por ter cometido a gafe irreparável de não citá-lo em meu trabalho de conclusão da graduação. Agradeço também aos professores e à instituição, por proporcionarem a infraestrutura e o conhecimento, me ajudando no crescimento como profissional.

## RESUMO

Este projeto aborda a criação de uma solução para implementação de uma plataforma de comércio eletrônico na modalidade dropshipping. Com o avanço das tecnologias e a popularização do acesso à internet, o comércio eletrônico ocupa cada vez mais espaço no mercado. O número de lojas virtuais tem crescido em ritmo alucinante, fazendo com que a concorrência aumente e os desafios também. Prover grandes experiências nos processos de compras, agilizar as entregas, fidelizar os clientes e, como em todo negócio, minimizar custos para maximizar os lucros. Nesse cenário, a modalidade de comércio dropshipping vem ganhando cada vez mais espaço. Nessa modalidade, a loja não tem a responsabilidade de manter um estoque e nem de cuidar da logística de entrega dos produtos. Essas responsabilidades são delegadas ao fornecedor, de maneira transparente para o cliente. O projeto aborda todos os aspectos da elaboração de uma arquitetura para suprir as necessidades de negócio e a criação de uma prova de conceito para os requisitos considerados mais importantes.

**Palavras-chave:** arquitetura de software, projeto arquitetural, requisitos não funcionais, dropshipping, comércio eletrônico, dispositivos móveis.

## SUMÁRIO

1.	Objetivos do trabalho .....	7
<b>2.</b>	<b>Descrição geral da solução .....</b>	<b>8</b>
2.1.	Apresentação do problema.....	8
2.2.	Descrição geral do software .....	9
<b>3.</b>	<b>Definição conceitual da solução.....</b>	<b>9</b>
3.1.	Requisitos Funcionais.....	9
3.2	Requisitos Não-Funcionais .....	16
3.3.	Restrições Arquiteturais .....	20
3.4.	Mecanismos Arquiteturais .....	20
<b>4.</b>	<b>Modelagem e projeto arquitetural .....</b>	<b>22</b>
4.1.	Modelo de casos de uso .....	22
4.2.	Descrição resumida dos casos de uso.....	28
4.3.	Modelo de componentes .....	36
4.4.	Modelo de implantação .....	40
4.5.	Modelo de dados .....	43
<b>5.</b>	<b>Prova de conceito / protótipo arquitetural.....</b>	<b>44</b>
5.1.	Implementação e implantação .....	44
5.2.	Interfaces/ APIs .....	47
<b>6.</b>	<b>Avaliação da Arquitetura .....</b>	<b>50</b>
6.1.	Análise das abordagens arquiteturais .....	50
6.2.	Identificação dos atributos de qualidade.....	51
6.3.	Cenários .....	51
6.4.	Avaliação.....	53
6.5.	Resultados .....	78
7.	Conclusão.....	80
	<b>REFERÊNCIAS .....</b>	<b>81</b>
	<b>APÊNDICES.....</b>	<b>82</b>

## 1. Objetivos do trabalho

O objetivo geral deste projeto é apresentar uma proposta de arquitetura para construção de uma plataforma integrada de comércio eletrônico baseado em dropshipping. O projeto visa fornecer uma plataforma na qual todos os envolvidos no processo de venda tenham suas atividades facilitadas. A plataforma deverá possuir alta disponibilidade, segurança, performance e poderá ser acessada por diversos dispositivos diferentes com acesso à internet através de um browser (desktops, notebooks, tablets e smartphones).

Os objetivos específicos são:

1. Criar o módulo de loja, possibilitando que os usuários - no caso clientes e vendedores - possam navegar por um catálogo de produtos, adicionar a um carrinho e efetivar, respectivamente, compras e vendas. Os usuários poderão ainda atualizar dados cadastrais, acompanhar entregas e consultar o histórico de pedidos realizados. Neste módulo, apenas o catálogo e o carrinho poderão ser visualizados publicamente. O restante das operações irão requerer autenticação segura.
2. Criar o módulo administrativo, responsável por permitir que seja feita a gestão de propagandas, promoções, vendedores, fornecedores e produtos. Também será permitido aos administradores acessar uma gama de relatórios gerados a partir de informações do sistema de business intelligence, para auxiliar na tomada de decisões. Este módulo será protegido por autenticação segura.
3. Criar o módulo de serviço de atendimento ao cliente, que permitirá aos usuários acessar um canal direto com a loja para o esclarecimento de dúvidas e solução de problemas. Os atendimentos poderão ser avaliados pelos clientes. Esse módulo também será protegido por autenticação segura.
4. Criar um componente para realizar integrações com os sistemas externos envolvidos no processo da plataforma (fornecedores, plataforma de pagamentos e entidades governamentais para controle fiscal). Esse módulo terá todas suas funcionalidades protegidas por autenticação segura.

## 2. Descrição geral da solução

### 2.1. Apresentação do problema

Com o avanço das tecnologias e a popularização do acesso à internet, o comércio eletrônico ocupa cada vez mais espaço no mercado. Em um mundo em que o tempo livre é cada vez mais escasso, clientes prezam pela agilidade, facilidade e segurança na realização de compras. Além disso, buscam cada vez mais participar de processos frictionless, estando no controle das ações com o menor atrito e maior transparência possível.

Por sua vez, as lojas virtuais têm grandes desafios: prover grandes experiências nos processos de compras, agilizar as entregas, fidelizar os clientes e, como em todo negócio, minimizar custos para maximizar os lucros. Tudo isso para que possam se manter competitivas em um mercado cada vez mais disputado. Muitas dessas lojas trabalham com uma vasta gama de produtos, o que torna cada vez mais difícil os processos de manutenção de estoque e gestão da logística de entregas.

Nesse cenário, a modalidade de comércio dropshipping vem ganhando cada vez mais espaço. Nessa modalidade, a loja não tem a responsabilidade de manter um estoque e nem de cuidar da logística de entrega dos produtos. Essas responsabilidades são delegadas ao fornecedor, de maneira transparente para o cliente. O dropshipping permite que as lojas concentrem-se nos clientes e na experiência de compra, enquanto que os fornecedores não têm de se preocupar com detalhes de suporte às vendas, publicidade online ou com a manutenção de um sistema de comércio eletrônico.

Em comércios eletrônicos com dropshipping, as grandes dificuldades estão concentradas nas integrações. Na comunicação com fornecedores, é preciso saber sobre a disponibilidade de produtos e sobre os eventos relacionados aos pedidos e às entregas. Já os clientes, precisam ser notificados a cada evento ocorrido em tempo real. Por vezes, há ainda a integração com plataformas de pagamentos, que facilitam a disponibilização de meios de pagamentos e diminuem a complexidade do trato dessas questões na plataforma. Isso sem esquecer das integrações com entidades governamentais para controles fiscais, como a Secretaria da Fazenda. Gerir e sincronizar todos esses aspectos é extremamente complexo e trabalhoso.

## **2.2. Descrição geral do software**

A elaboração desse software tem por objetivo fornecer uma plataforma integrada de comércio eletrônico, na qual todos os envolvidos no processo de venda tenham suas atividades facilitadas.

Os clientes poderão acessar a loja de qualquer dispositivo com acesso à internet via browser, a qualquer hora do dia e de qualquer lugar. Quaisquer problemas ocorridos, serão resolvidos no menor tempo possível, através de um SAC – Sistema de Atendimento ao Cliente.

Os vendedores terão à mão o catálogo dos produtos a qualquer momento, podendo levar aos clientes uma grande experiência de compra, atingindo assim um maior público.

Gestores terão acesso à dashboards com diversos relatórios gerados pelo sistema de business intelligence, auxiliando na tomada rápida de decisões. Algumas dessas decisões poderão ser executadas diretamente na plataforma, como alterações de preços, realização de promoções ou alterações no relacionamento com fornecedores.

A plataforma deverá possuir alta disponibilidade, segurança, performance e poderá ser acessada por diversos dispositivos diferentes com acesso à internet através de um browser (desktops, notebooks, tablets e smartphones).

## **3. Definição conceitual da solução**

### **3.1. Requisitos Funcionais**

#### **Módulo Loja**

- Catálogo**

O sistema deve permitir que os usuários naveguem por um catálogo de produtos, sendo possível navegar por categorias ou realizar pesquisas por nome;

- Detalhar produtos**

O sistema deve permitir que os usuários vejam os detalhes dos produtos, assim como as avaliações de quem comprou aquele item.

- **Carrinho**

O sistema deve permitir que os usuários adicionem vários produtos a um carrinho de compras;

O sistema deve permitir que os usuários removam produtos de seu carrinho de compras.

O sistema deve permitir que os usuários alterem a quantidade de produtos diretamente no carrinho.

- **Cadastro**

O sistema deve permitir que os clientes se cadastrem, informando seus dados pessoais e um e-mail válido;

O sistema deve permitir que vendedores cadastrem clientes, informando seus dados pessoais e um e-mail válido.

- **Autenticação**

O sistema deve permitir que usuários que possuem cadastro (tanto clientes quanto vendedores) possam se autenticar no sistema, efetuando login;

O sistema deve permitir que usuários autenticados possam alterar seus dados pessoais, como telefones e endereços.

- **Realizar compras/vendas**

O sistema deve permitir que usuários autenticados como clientes possam efetivar a compra dos itens que estão em seu carrinho. Para tal, o cliente deverá informar o endereço de entrega e efetuar o pagamento;

O sistema deve permitir que usuários autenticados como vendedores possam efetivar a venda dos itens que estão em seu carrinho. Para tal, o vendedor deverá

informar os dados de cadastro do cliente e também o código do pagamento realizado presencialmente;

Quando a compra for realizada por um cliente, o processo de pagamento deverá ser feito através de uma plataforma eletrônica de pagamentos, permitindo quaisquer meios de pagamento disponíveis na plataforma escolhida. No caso de compras realizadas através da mediação de vendedores, o pagamento deverá ser realizado via cartão de crédito ou de débito, utilizando a máquina para pagamentos disponibilizada pela plataforma de pagamentos, gerando a necessidade de que no momento da venda seja informado o código do pagamento gerado pela máquina;

O pedido somente deverá ser enviado ao fornecedor uma vez que o pagamento tenha sido confirmado. Ocorrendo qualquer problema com o processo de pagamento, a venda deverá ser cancelada e os envolvidos notificados.

- **Consultar compras/vendas**

O sistema deve permitir que usuários autenticados como clientes possam consultar suas compras;

O sistema deve permitir que usuários autenticados como vendedores possam consultar suas vendas.

- **Cancelar compras**

O sistema deve permitir que usuários autenticados como clientes possam cancelar suas compras. Quando solicitado o cancelamento, o pagamento deverá ser estornado. Se a compra foi realizada através de um vendedor, este deverá ser notificado do cancelamento.

## **Módulo Serviço de Atendimento ao Cliente**

- **Solicitar atendimento**

O sistema deve permitir que usuários autenticados como cliente possam solicitar atendimento, que pode ser uma dúvida, uma solicitação ou até mesmo uma

sugestão. Esse atendimento deverá ser classificado em alguma categoria como “financeiro”, “entrega” ou “devolução”.

- **Direcionar atendimento**

O sistema deve automaticamente direcionar as novas solicitações de atendimento aos atendentes, de acordo com o número de chamados em que estão atuando, de forma a garantir que não haverá sobrecarga dos atendentes.

- **Consultar fila de atendimentos**

O sistema deve permitir que um usuário autenticado como atendente possa consultar a fila de atendimentos a ele atribuídos. Na fila o usuário poderá visualizar seus atendimentos em andamento, que estejam pendentes de realização de ações da sua parte, além de poder iniciar o próximo atendimento atribuído automaticamente.

- **Iniciar atendimento**

O sistema deve permitir que um usuário autenticado como atendente possam iniciar um atendimento a ele atribuído. O cliente deverá ser notificado em tempo real sobre a mudança de status de seu atendimento.

- **Realizar comunicação**

O sistema deve permitir que usuários autenticados como atendentes possam responder aos atendimentos a ele direcionados, estabelecendo a comunicação com o cliente;

O sistema deve permitir que usuários autenticados como clientes possam responder aos atendimentos por ele abertos, estabelecendo a comunicação com o atendente;

Clientes e atendentes deverão ser notificados em tempo real sobre as mensagens recebidas da outra parte.

- **Encerrar atendimento**

O sistema deve permitir que os usuários autenticados como clientes possam encerrar um atendimento. No encerramento, o usuário poderá avaliar o atendimento, o que acarretará em uma métrica para posterior extração relacionado à qualidade dos atendentes e dos atendimentos em geral;

O sistema deverá encerrar automaticamente os atendimentos que não tiveram resposta por parte dos usuários no período de trinta dias. Os clientes deverão ser notificados quando sua solicitação estiver pendente de ação, e lembrados a cada dez dias sobre sua inatividade.

## **Módulo Administrativo**

- **Manter fornecedores**

O sistema deve permitir que os usuários autenticados como administradores possam cadastrar e manter os cadastros de fornecedores. Nesse cadastro será possível de uma só vez bloquear ou habilitar as vendas de todos os produtos relacionados ao fornecedor.

- **Manter categorias de produtos**

O sistema deve permitir que os usuários autenticados como administradores possam cadastrar e manter os cadastros de categorias de produtos.

- **Manter produtos**

O sistema deve permitir que os usuários autenticados como administradores possam cadastrar e manter os cadastros de produtos. Um produto obrigatoriamente deve estar relacionado a um fornecedor e a uma categoria de produtos. A disponibilidade de um produto será atualizada de acordo com a integração realizada com o sistema do fornecedor. Os usuários autenticados como administradores também poderão desabilitar manualmente a venda de determinados produtos.

- **Manter vendedores**

O sistema deve permitir que os usuários autenticados como administradores possam cadastrar e manter os cadastros de vendedores;

O sistema deve permitir que os usuários autenticados como administradores bloqueiem o acesso dos vendedores a plataforma.

- **Gerenciar promoções**

O sistema deve permitir que os usuários autenticados como administradores possam cadastrar e manter os cadastros de promoções;

As promoções deverão possuir uma data de início e poderão possuir uma data de previsão de encerramento. As promoções cujas datas de encerramento forem informadas deverão ser encerradas automaticamente pelo sistema ao atingir-se essa data;

Será possível vincular quaisquer produtos a uma promoção, informando o novo preço. Esse é o preço que será exibido no catálogo e considerado no momento das compras/vendas durante o período da promoção.

- **Gerenciar propagandas**

O sistema deverá permitir que os usuários autenticados como administradores possam cadastrar e manter o cadastro das propagandas que serão exibidas para os clientes. Cada propaganda terá sua ordem de exibição na tela inicial da loja pré-definida e obrigatoriamente deverá possuir um banner. Quando clicada, deverá levar a um endereço específico do site.

- **Visualizar dashboards**

O sistema deverá permitir que os usuários autenticados como administradores possam visualizar relatórios gerados pelo sistema de business intelligence para auxiliar na tomada de decisão, como:

- o Relatório de vendas;
- o Vendedores com melhor desempenho;
- o Produtos mais vendidos;

- o Produtos com maior rentabilidade;
- o Relatório de custos.

## **Módulo Geração de Informações Gerenciais**

- **Exportação de dados de vendas**

O sistema deverá permitir que, em intervalos parametrizáveis de tempo, sejam exportados dados sobre as vendas realizadas para alimentar a base do sistema de business intelligence.

- **Exportação de dados de entregas**

O sistema deverá permitir que, em intervalos parametrizáveis de tempo, sejam exportados dados sobre as entregas para alimentar a base do sistema de business intelligence.

- **Exportação de dados de atendimentos aos clientes**

O sistema deverá permitir que, em intervalos parametrizáveis de tempo, sejam exportados dados sobre os atendimentos realizados aos clientes para alimentar a base do sistema de business intelligence.

## **Módulo Integrações**

- **Notificação de mudança de status da entrega**

O sistema deverá permitir que os fornecedores possam notificar a loja sobre alterações no status de uma entrega. Uma vez que o sistema receba a notificação, deverá imediatamente informar todos os envolvidos no processo daquela venda por e-mail e SMS (se disponível).

- **Notificação de cancelamento da venda**

O sistema deverá permitir que os fornecedores possam notificar a loja sobre o cancelamento de uma venda, por qualquer que seja o motivo. Uma vez que o sistema receba o cancelamento, o pagamento realizado pelo cliente deverá ser

estornado. A nota fiscal emitida relacionada ao pedido deverá ser cancelada. Todos os envolvidos no processo dessa venda deverão ser notificados imediatamente.

- **Notificação de alteração no status do pagamento**

O sistema deverá permitir que a plataforma de pagamentos informe alterações no status do pagamento. Uma vez que um pagamento tenha sido confirmado, o pedido é liberado para o fornecedor. Um pagamento cancelado ou recusado deverá fazer com que o cancelamento da compra ocorra, notificando todos os envolvidos no processo.

- **Atualização automática de disponibilidade**

O sistema deverá, num período parametrizável de tempo (inicialmente definido como uma vez ao dia), integrar-se aos sistemas de fornecedores para consultar a disponibilidade dos produtos cadastrados. Produtos indisponíveis deverão sair do catálogo até estarem novamente disponíveis.

### **3.2 Requisitos Não-Funcionais**

- Usabilidade – o sistema deve prover boa usabilidade

<b>Estímulo</b>	Usuário buscando produtos
<b>Fonte do estímulo</b>	Usuário acessando a funcionalidade de catálogo, incluindo itens em seu carrinho de compras.
<b>Ambiente</b>	Funcionamento, carga normal
<b>Artefato</b>	Módulo Loja
<b>Resposta</b>	A camada de apresentação apresenta facilidade de navegação, simplicidade e objetividade.
<b>Medida da resposta</b>	Usuário conseguiu identificar os itens que deseja, adicionou ao carrinho e está pronto para finalizar o processo de compra em no máximo 5 minutos.

- Acessibilidade – o sistema deve ser suportar ambientes web responsivos e ambientes móveis

<b>Estímulo</b>	Usuário buscando produtos
<b>Fonte do estímulo</b>	Usuário acessando a funcionalidade de catálogo, incluindo itens em seu carrinho de compras, em um smartphone.
<b>Ambiente</b>	Funcionamento, carga normal
<b>Artefato</b>	Módulo Loja
<b>Resposta</b>	A camada de apresentação se adaptou à resolução e tamanho da tela, mudando os componentes de tamanho e posição para facilitar a navegação.
<b>Medida da resposta</b>	A identidade visual se mantém parecida em todas as telas e resoluções e não há perda de funcionalidades.

- Segurança – o sistema deve apresentar altos padrões de segurança
  - o Acesso a página privada sem estar autenticado

<b>Estímulo</b>	Acessar uma página privada pela URL sem estar autenticado no sistema
<b>Fonte do estímulo</b>	Usuário não autenticado
<b>Ambiente</b>	Funcionamento, carga normal
<b>Artefato</b>	Módulo Loja, Módulo Serviço de Atendimento ao Cliente, Módulo Administrativo
<b>Resposta</b>	O sistema redirecionou o usuário para a respectiva tela de autenticação.
<b>Medida da resposta</b>	O sistema não permite acesso à página solicitada, e obriga a autenticação.

- o Acesso a integração sem estar autenticado

<b>Estímulo</b>	O sistema do fornecedor envia uma requisição de atualização de status de uma entrega sem estar autenticado no sistema
-----------------	---

<b>Fonte do estímulo</b>	Usuário não autenticado
<b>Ambiente</b>	Funcionamento, carga normal
<b>Artefato</b>	Módulo Interações
<b>Resposta</b>	O sistema recusou o envio da atualização de status.
<b>Medida da resposta</b>	Sistema não permite atualizar o status.

- Interoperabilidade – o sistema deve se comunicar com vários sistemas, com tecnologias heterogêneas

<b>Estímulo</b>	Teste de comunicação
<b>Fonte do estímulo</b>	Sistema do fornecedor – consulta ao estoque de um produto
<b>Ambiente</b>	Funcionamento, carga normal
<b>Artefato</b>	Módulo Interações
<b>Resposta</b>	O serviço do fornecedor respondeu com sucesso a solicitação, informando o status de disponibilidade do produto.
<b>Medida da resposta</b>	Comunicação efetuada.

- Desempenho – o sistema deve ser rápido

<b>Estímulo</b>	Usuário navegando pelo catálogo
<b>Fonte do estímulo</b>	Usuário buscando produtos de uma determinada categoria
<b>Ambiente</b>	Funcionamento, carga normal
<b>Artefato</b>	Módulo Loja
<b>Resposta</b>	O sistema respondeu com os dados solicitados.
<b>Medida da resposta</b>	O sistema respondeu em menos de 5 segundos.

- Disponibilidade – o sistema deve estar disponível em qualquer período, do dia e da noite em regime 24/7

<b>Estímulo</b>	Desligamento forçado de um dos nós do <i>cluster</i> dos
-----------------	--

	servidores de aplicação
<b>Fonte do estímulo</b>	Administrador do <i>cluster</i> de servidores de aplicação
<b>Ambiente</b>	Funcionamento, carga normal
<b>Artefato</b>	Módulo Loja
<b>Resposta</b>	Todos os usuários continuaram usando o sistema sem notar que houve uma queda de um dos nós.
<b>Medida da resposta</b>	Todas as solicitações de usuários foram atendidas, podendo haver um atraso de 2 segundos devido à queda de um dos nós.

- Testabilidade – o sistema deve ser simples para testar

<b>Estímulo</b>	Execução de testes no sistema
<b>Fonte do estímulo</b>	Analista desenvolvedor
<b>Ambiente</b>	Ambiente de desenvolvimento
<b>Artefato</b>	Módulo Loja, Módulo Serviço de Atendimento ao Cliente, Módulo Administrativo, Módulo Integrações
<b>Resposta</b>	O sistema testou todas as funcionalidades disponíveis.
<b>Medida da resposta</b>	O sistema deve possibilitar efetuar os testes com <i>scripts</i> automatizados, que podem ser executados por máquinas.

- Manutenibilidade – o sistema deve apresentar manutenção facilitada

<b>Estímulo</b>	Alteração no SGBD utilizado
<b>Fonte do estímulo</b>	Compra de licença de outro SGBD com maior quantidade de recursos e suporte do fornecedor
<b>Ambiente</b>	Aproximação dos limites de utilização do banco de dados na versão contratada
<b>Artefato</b>	Módulo Loja, Módulo Serviço de Atendimento ao Cliente, Módulo Administrativo, Módulo Integrações

<b>Resposta</b>	As modificações no código são realizadas isoladamente nos componentes da camada de acesso a dados das aplicações.
<b>Medida da resposta</b>	Foram modificados apenas os componentes responsáveis por realizar a comunicação com o banco, não impactando demais componentes.

### 3.3. Restrições Arquiteturais

- A plataforma deve ser desenvolvida em .NET Core (backend) e Angular (frontend);
- A plataforma deve ser modular para facilitar a implantação;
- A plataforma deverá utilizar uma plataforma de pagamentos para disponibilizar a maior quantidade possível de meios de pagamento aos clientes;
- O sistema deve funcionar de forma responsiva em aparelhos menores, como smartphones e tablets;
- O sistema deve ser implantável na plataforma de nuvem da Amazon (AWS).

### 3.4. Mecanismos Arquiteturais

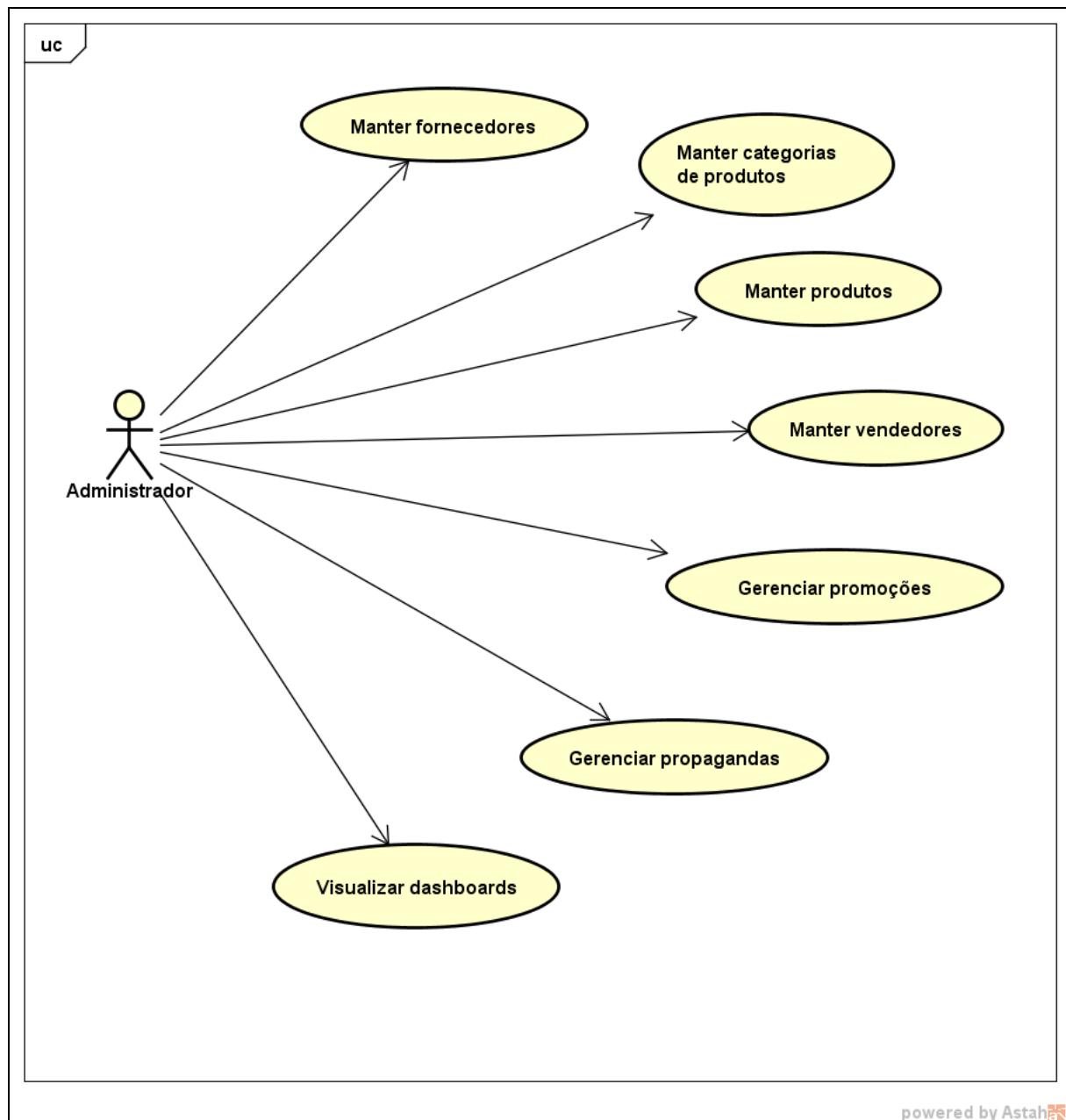
Mecanismo de análise	Mecanismo de design	Mecanismo de implementação
Front-end	Interface de comunicação com o usuário do sistema	Angular e Bootstrap
Back-end	Regras de negócio da aplicação	.NET Core
Interações com outros módulos e sistemas	Interfaces utilizando XML e/ou JSON	WebServices WS-* e Restful API's
Persistência	ORM e tecnologias de acesso a dados	EFCore e ADO.Net
Persistência	Banco de dados relacional e plataforma de Big Data	AWS RDS (Relational Database Service) e

		Microsoft SQL Server
Interações com plataforma de BigData	Tarefas agendadas para extração de dados	Sqoop
Serviços agendados	Execução de processamentos agendados	AWS Cloudwatch e AWS Lambda
Log	Framework de Log	Serilog e AWS Cloudwatch
Alta disponibilidade	Balanceamento de carga das aplicações	AWS ElasticBeanstalk
Autenticação e autorização	Verificação das credenciais para execução de ações	AWS Cognito
Exposição de API's	Exposição de Restful API's	AWS API Gateway
Notificações de usuários	Envio de notificações aos usuários por SMS e e-mail	AWS SNS – Simple Notification Service e AWS SES – Simple Email Service
Disponibilização de conteúdo estático	Aplicação para servir conteúdo estático, como HTML, Javascript, CSS, fontes e imagens	AWS S3 – Simple Storage Service
CI/CD	Ferramenta para pipeline de integração e entrega contínua	Jenkins
Build	Geração de artefatos para publicação nos servidores de aplicação	MSBuild e Angular-Cli
Automação de testes	Execução de testes automatizados das aplicações	MSTest e Jasmine
Deploy	Deploy de artefatos para os servidores de aplicação	AWS Code Deploy
Versionamento	Controle de código-fonte	Git e Github

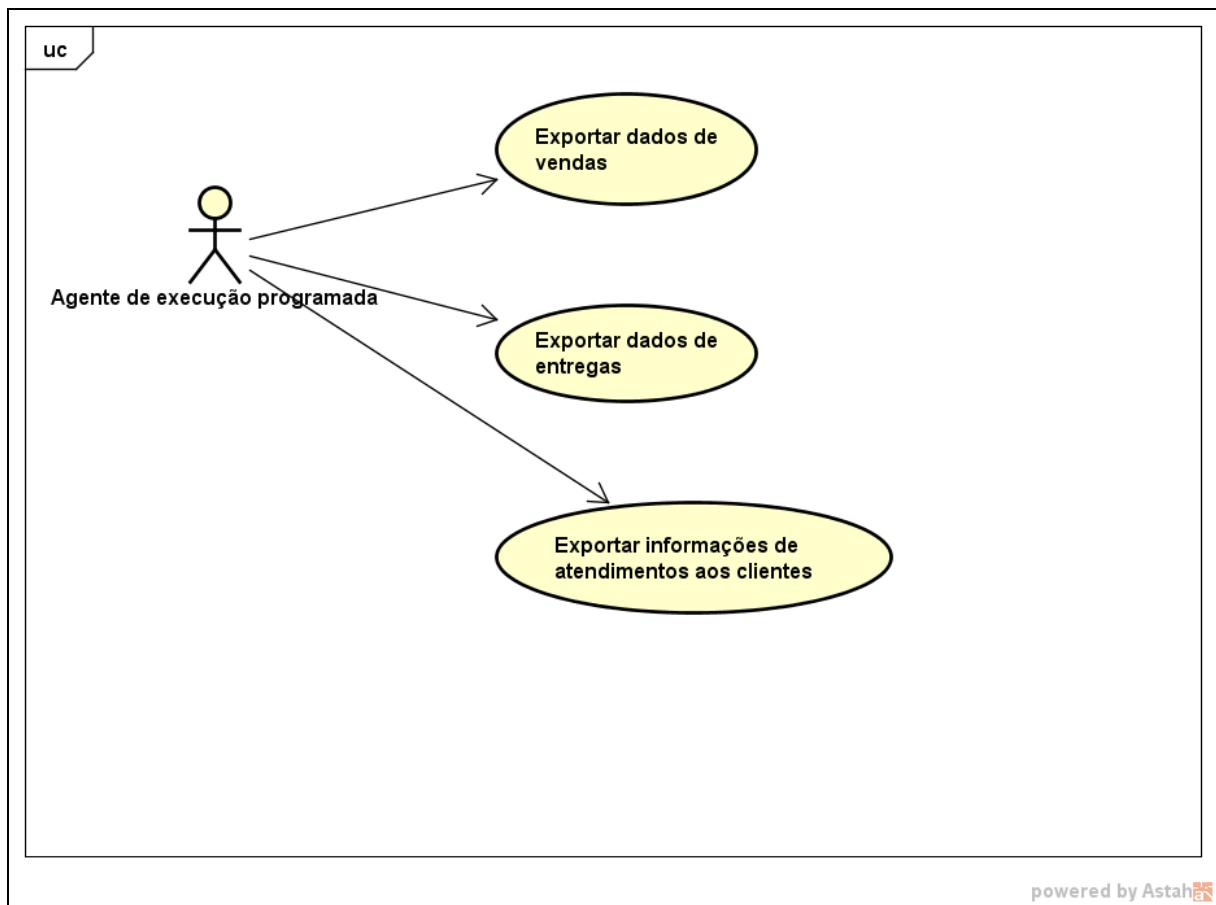
## 4. Modelagem e projeto arquitetural

### 4.1. Modelo de casos de uso

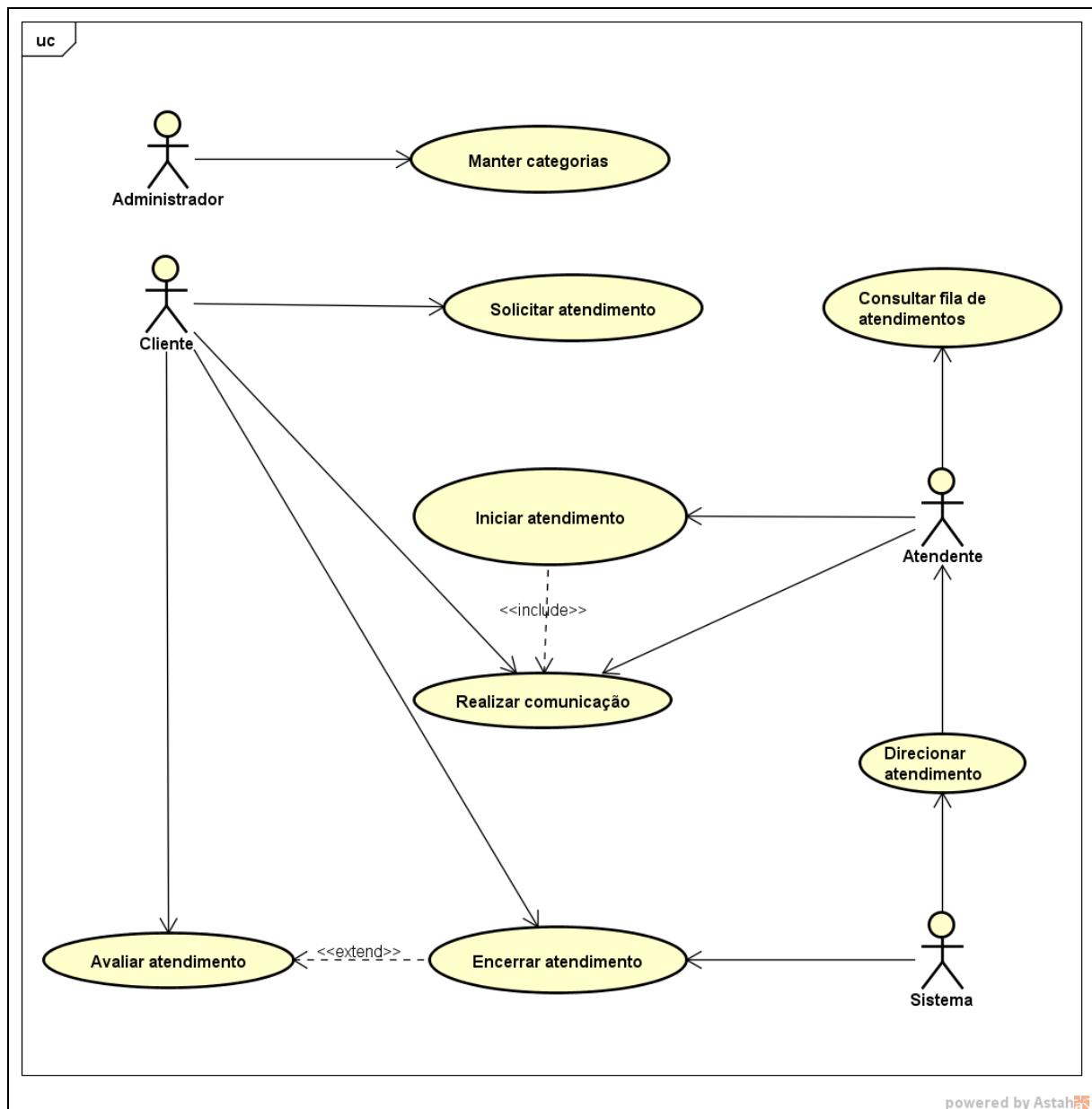
O diagrama de casos de uso oferece uma visão global dos casos de uso e dos atores que dele participam. Para uma melhor análise arquitetural do projeto, os casos de uso foram organizados por módulos, de acordo com os requisitos informados anteriormente.



**Figura 1 – Diagrama de Casos de Uso – Módulo Administrativo**



**Figura 2 – Diagrama de Casos de Uso – Módulo Geração de Informações Gerenciais**



**Figura 3 – Diagrama de Casos de Uso – Módulo Serviço de Atendimento ao Cliente**

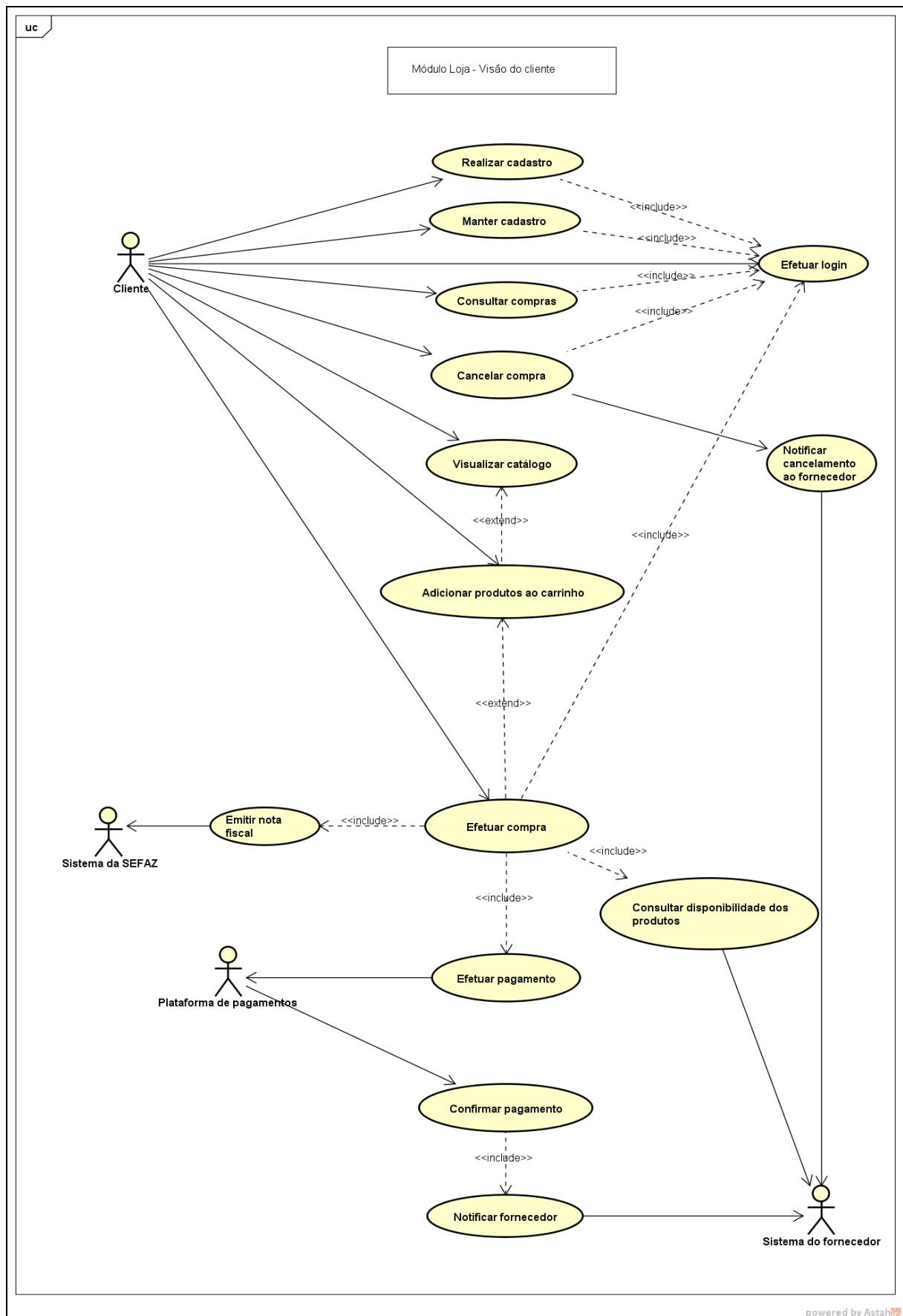


Figura 4 – Diagrama de Casos de Uso – Módulo Loja (visão do cliente)

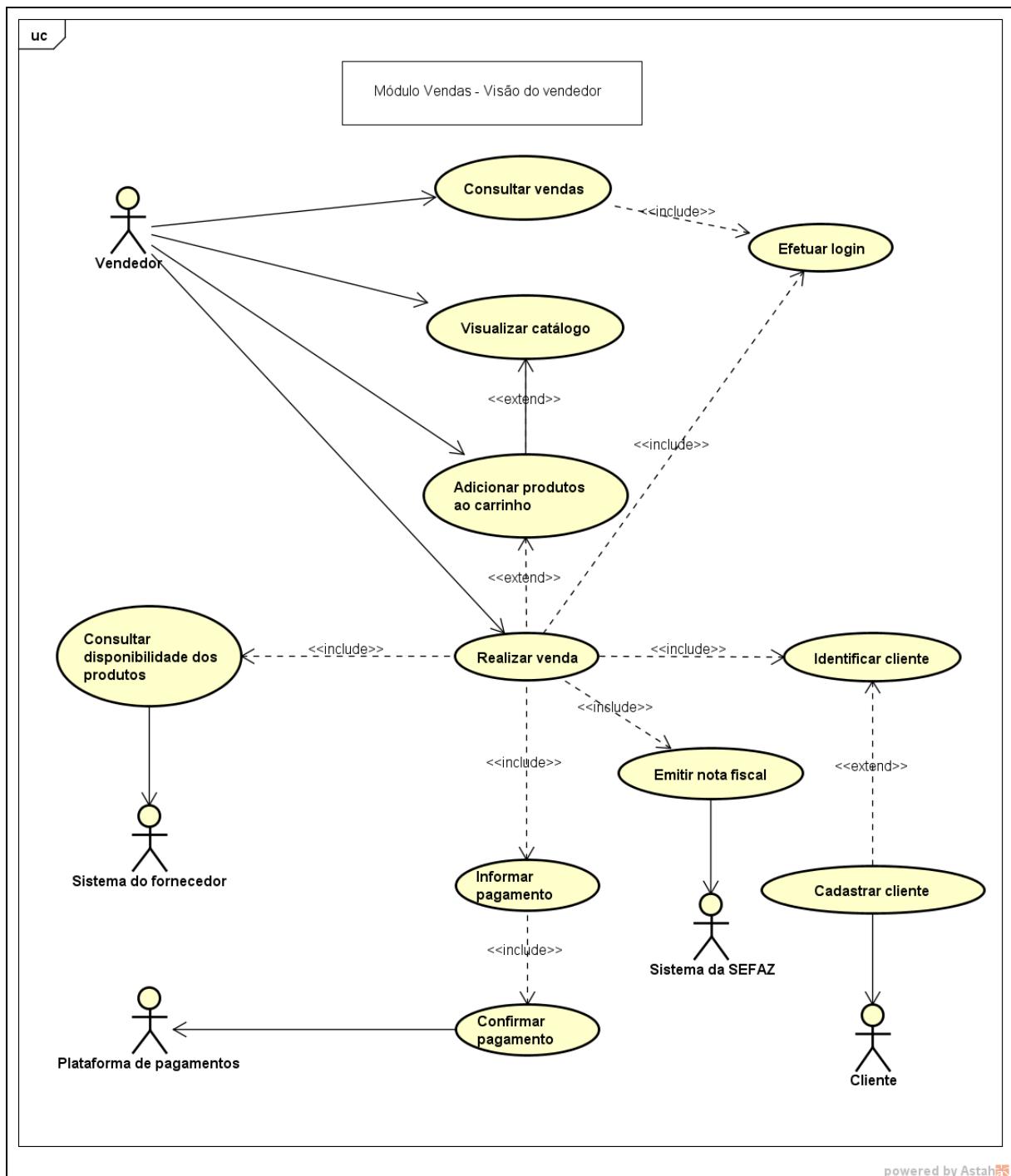
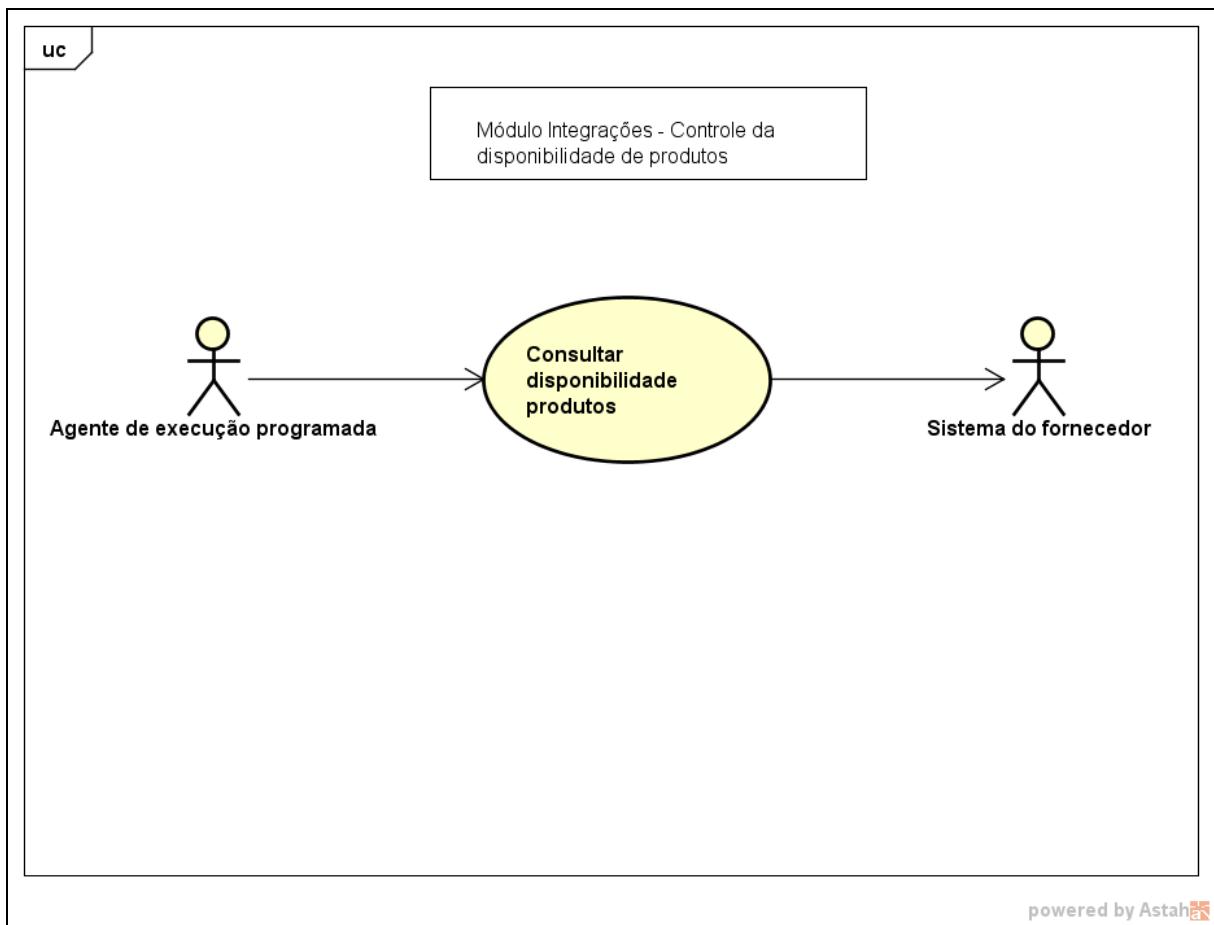
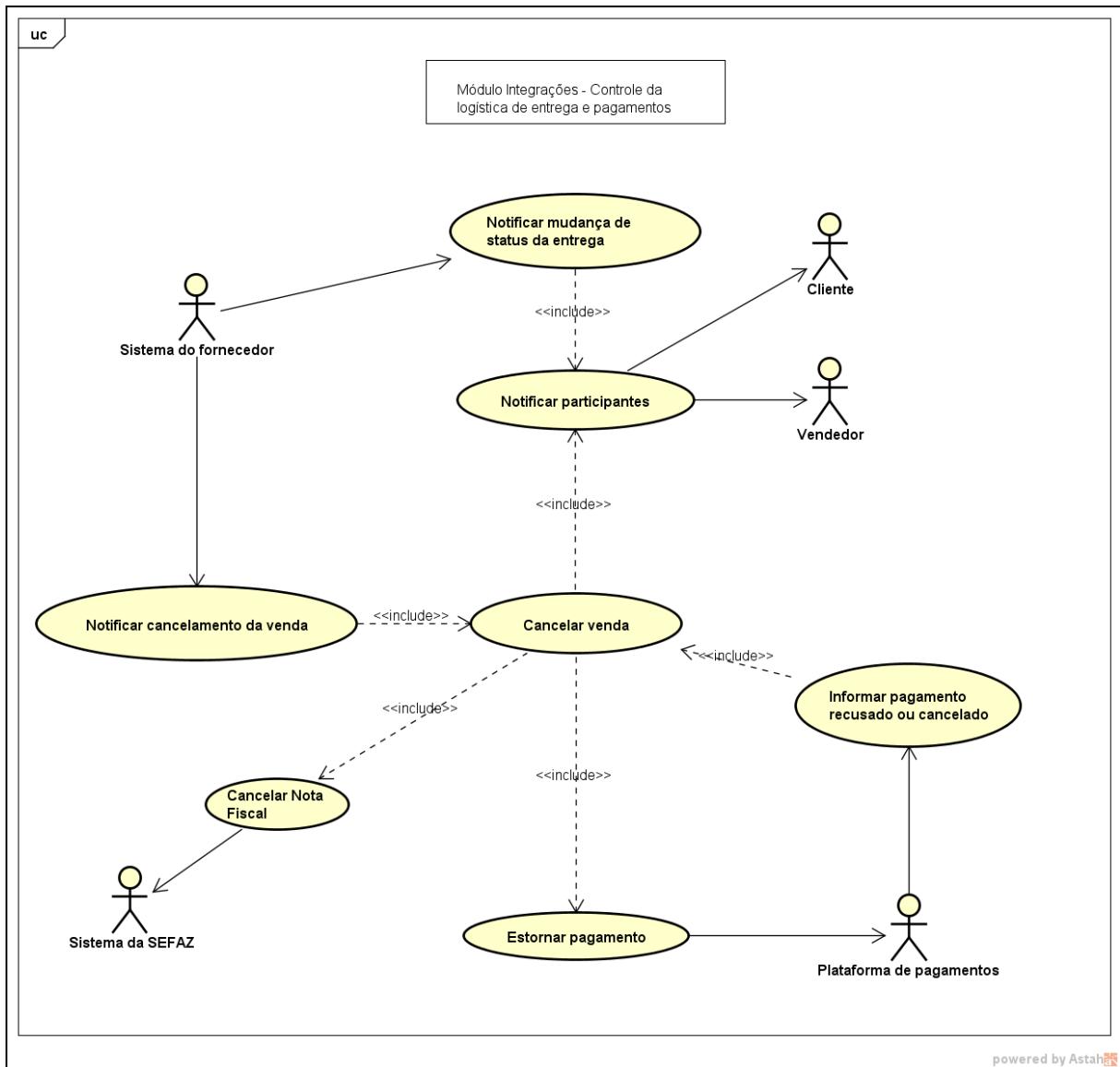


Figura 5 – Diagrama de Casos de Uso – Módulo Loja (visão do vendedor)



**Figura 5 – Diagrama de Casos de Uso – Módulo Integrações (consultar disponibilidade de produtos no fornecedor)**



**Figura 6 – Diagrama de Casos de Uso – Módulo Integrações (controle da logística de entrega e de pagamentos)**

#### 4.2. Descrição resumida dos casos de uso

- **Módulo Administrativo**

##### Caso de uso: Manter fornecedores

**Descrição:** Este caso de uso deve permitir aos administradores listar, criar, atualizar e excluir fornecedores. Também será permitido inativar fornecedores, para que seus

produtos deixem de aparecer no catálogo da loja e deixem de ter sua disponibilidade atualizada automaticamente. Deverão existir filtros para auxiliar o usuário a encontrar o fornecedor desejado.

#### **Caso de uso: Manter categorias de produtos**

**Descrição:** Este caso de uso deve permitir aos administradores listar, criar, atualizar e excluir categorias de produtos. Deverão existir filtros para auxiliar o usuário a encontrar a categoria desejada. Categorias que possuem produtos cadastrados não poderão ser excluídas.

#### **Caso de uso: Manter produtos**

**Descrição:** Este caso de uso deve permitir aos administradores listar, criar, atualizar e excluir produtos. Deverão existir filtros para auxiliar o usuário a encontrar o produto desejado. Todo produto deve estar vinculado a uma categoria. Também será permitido ao usuário inativar um produto, impedindo assim que este apareça no catálogo da loja. A disponibilidade dos produtos será atualizada automaticamente em períodos regulares de tempo.

#### **Caso de uso: Manter vendedores**

**Descrição:** Este caso de uso deve permitir aos administradores listar, criar, atualizar e excluir vendedores. Deverão existir filtros para auxiliar o usuário a encontrar o vendedor desejado. Deverá ser possível inativar os vendedores, impedindo assim o acesso a plataforma da loja.

#### **Caso de uso: Gerenciar promoções**

**Descrição:** Este caso de uso deve permitir aos administradores listar, criar, atualizar e excluir promoções. Deverão existir filtros para auxiliar o usuário a encontrar a promoção desejada. Uma promoção tem obrigatoriamente uma data para iniciar e pode ter uma data prevista de término. Uma promoção com data prevista de término será automaticamente encerrada por um processamento automático. Também será permitido interromper promoções, inativando seu status. Toda promoção deverá possuir um banner, sendo uma

imagem em tamanho pré-definido. Cada promoção deverá possuir ao menos um produto a ela vinculado.

### **Caso de uso: Gerenciar propagandas**

**Descrição:** Este caso de uso deve permitir aos administradores listar, criar, atualizar e excluir propagandas. Deverão existir filtros para auxiliar o usuário a encontrar a propaganda desejada. Cada propaganda deverá possuir uma posição, que indica a ordem em que ela aparecerá na tela inicial da loja. Toda propaganda deverá ter uma URL associada, para a qual o usuário da loja será redirecionado ao clicar sobre o banner na tela inicial. Toda propaganda deverá possuir um banner, sendo uma imagem em tamanho pré-definido. Não poderão existir duas propagandas ativas ao mesmo tempo ocupando uma mesma posição.

### **Caso de uso: Visualizar dashboards**

**Descrição:** Este caso de uso deve permitir aos administradores acessar uma gama de relatórios gerenciais e seus dashboards. São exemplos de relatórios: relatórios de vendas, produtos mais vendidos, rentabilidade, custos e vendedores mais produtivos.

- **Módulo geração de informações gerenciais**

### **Caso de uso: Exportar dados de vendas**

**Descrição:** Este caso de uso deve permitir que um agente de execução automática do sistema de business intelligence se conecte à aplicação e obtenha os dados das vendas realizadas dentro de um período pré-definido. Os dados que poderão ser exportados serão definidos e expostos através de uma visão controlada do banco de dados.

### **Caso de uso: Exportar dados de entregas**

**Descrição:** Este caso de uso deve permitir que um agente de execução automática do sistema de business intelligence se conecte à aplicação e obtenha os dados das entregas realizadas dentro de um período pré-definido. Os dados que poderão ser exportados serão definidos e expostos através de uma visão controlada do banco de dados.

### **Caso de uso: Exportar informações de atendimentos aos clientes**

**Descrição:** Este caso de uso deve permitir que um agente de execução automática do sistema de business intelligence se conecte à aplicação e obtenha os dados relacionados aos atendimentos a clientes realizados dentro de um período pré-definido. Os dados que poderão ser exportados serão definidos e expostos através de uma visão controlada do banco de dados.

- **Módulo SAC – Serviço de Atendimento ao Cliente**

#### Caso de uso: Manter categorias

**Descrição:** Este caso de uso deve permitir aos administradores listar, criar, atualizar e excluir categorias de atendimento. Deverão existir filtros para auxiliar o usuário a encontrar a categoria desejada. Exemplos de categorias são: dúvidas e reclamações. Cada categoria terá uma prioridade definida, o que fará com que atendimentos abertos desse tipo sejam priorizados pelo sistema para inclusão na fila dos atendentes.

#### Caso de uso: Solicitar atendimento

**Descrição:** Este caso de uso deve permitir aos clientes solicitar atendimento, informando sua categoria, a qual item de um pedido se refere, um título e uma descrição para o problema. Será possível anexar evidências às solicitações. Os atendimentos deverão ser enviados para uma fila, na qual o sistema irá atuar para priorizar e direcionar às filas dos atendentes.

#### Caso de uso: Consultar fila de atendimentos

**Descrição:** Este caso de uso deve permitir aos atendentes listar os atendimentos em que está atuando no momento e alternar entre eles. A listagem deve exibir os atendimentos em ordem cronológica, sempre priorizando os que foram abertos há mais tempo.

#### Caso de uso: Direcionar atendimento

**Descrição:** Este caso de uso deve que o sistema encaminhe automaticamente os atendimentos abertos por clientes para os atendentes, de acordo com sua prioridade.

#### Caso de uso: Iniciar atendimento

**Descrição:** Este caso de uso deve permitir aos atendentes iniciar um atendimento que está em sua fila. O cliente deverá receber uma notificação de que seu atendimento foi iniciado.

### **Caso de uso: Realizar comunicação**

**Descrição:** Este caso de uso deve permitir aos atendentes e clientes uma troca de mensagens e anexos em busca da solução do problema. As conversas deverão ser armazenadas para futuras auditorias. A cada nova mensagem, os envolvidos deverão ser notificados de um progresso.

### **Caso de uso: Encerrar atendimento**

**Descrição:** Este caso de uso deverá ser acessível por clientes e pelo próprio sistema para encerrar solicitações. Uma solicitação em aberto há 30 dias (período parametrizável), com pendências por parte dos usuários, serão encerradas automaticamente pelo sistema. Quando as solicitações são encerradas, os clientes deverão ser notificados e receberão uma indicação para que possam avaliar o atendimento. Os atendentes deverão ser notificados para também avaliar os clientes, de forma que sejam obtidos posteriormente insumos sobre os perfis de usuários da plataforma.

### **Caso de uso: Avaliar atendimento**

**Descrição:** Quando uma solicitação é encerrada, o cliente deverá poder avaliar um atendimento com uma nota (variando entre 0 e 10) e uma observação.

- **Módulo Loja**

### **Caso de uso: Realizar cadastro**

**Descrição:** Este caso de uso permitirá que usuários anônimos possam se cadastrar na loja, informando um nome de usuário, um e-mail e uma senha. Também poderão ser informados dados complementares, como telefone, documento e endereços. Os dados complementares não serão necessários no momento do cadastro, mas no momento do fechamento de uma venda devem estar preenchidos. No momento do cadastro, um e-mail deverá ser enviado ao usuário para confirmação. Enquanto a confirmação não for realizada, não será possível efetuar login.

### **Caso de uso: Efetuar login**

**Descrição:** Este caso de uso permitirá que usuários com conta criada e confirmada possam acessar a aplicação. Usuários autenticados poderão acessar uma área segura, na qual será possível atualizar seus dados pessoais e acessar sua lista de pedidos.

### **Caso de uso: Manter cadastro**

**Descrição:** Este caso de uso permitirá que usuários autenticados possam visualizar e atualizar seus dados cadastrais, como telefone, endereços e documento.

### **Caso de uso: Consultar pedidos**

**Descrição:** Este caso de uso permitirá que usuários autenticados possam visualizar suas listas de pedidos (compras - no caso de clientes – ou vendas – no caso de vendedores).

### **Caso de uso: Cancelar compra**

**Descrição:** Este caso de uso permitirá que usuários autenticados como clientes possam cancelar suas compras, dentro de um prazo parametrizável de tempo. Quando uma compra é cancelada, o fornecedor deve ser avisado e o pagamento deverá ser estornado.

### **Caso de uso: Visualizar catálogo**

**Descrição:** Este caso de uso permitirá que usuários anônimos ou autenticados possam navegar por um catálogo de produtos, listando por categoria ou informando critérios para pesquisa. Os usuários poderão a partir daqui adicionar produtos a um carrinho. No carrinho, os usuários poderão atualizar a quantidade de produtos ou removê-los.

### **Caso de uso: Efetuar compra**

**Descrição:** Este caso de uso permitirá que clientes realizem suas compras, após selecionar seus produtos, quantidades e avançar a partir do carrinho. Antes de prosseguir, deverão ser verificadas as disponibilidades dos produtos nos fornecedores. Usuários que não estão autenticados deverão se autenticar ou se cadastrar nesse momento. Um endereço de entrega deverá ser selecionado ou informado. Deverá ser realizado o pagamento através da integração com uma plataforma de pagamentos. Uma vez confirmado o pagamento, uma

nota fiscal deverá ser emitida e enviada ao cliente por e-mail. Os fornecedores dos produtos comprados deverão ser informados para que iniciem o procedimento de entrega.

### **Caso de uso: Efetuar venda**

**Descrição:** Este caso de uso permitirá que vendedores realizem vendas, após selecionar os produtos, quantidades e avançar a partir do carrinho. Antes de prosseguir, deverão ser verificadas as disponibilidades dos produtos nos fornecedores. No momento da venda, o cliente deverá ser informado. Caso o cliente ainda não possua cadastro, o cadastro deverá ser realizado nesse momento. No momento do pagamento o vendedor deverá informar a chave gerada no cupom emitido pela máquina de cartões da plataforma de pagamentos. Essa chave será validada via integração com a plataforma de pagamentos antes de prosseguir com a emissão da nota fiscal e o envio do pedido aos fornecedores.

- **Módulo Integrações**

### **Caso de uso: Consultar disponibilidade de produtos**

**Descrição:** Este caso de uso permitirá que um agente de execução automática dispare (a cada intervalo parametrizável de tempo) um processamento de atualização da disponibilidade de produtos, realizando a integração com os sistemas dos fornecedores. Somente deverão ser atualizados produtos ativos de fornecedores também ativos.

### **Caso de uso: Realizar pedido**

**Descrição:** Este caso de uso permitirá que o sistema envie pedidos aos fornecedores através de integração. Os pedidos deverão ser realizados no momento em que o pagamento é confirmado.

### **Caso de uso: Cancelar pedido**

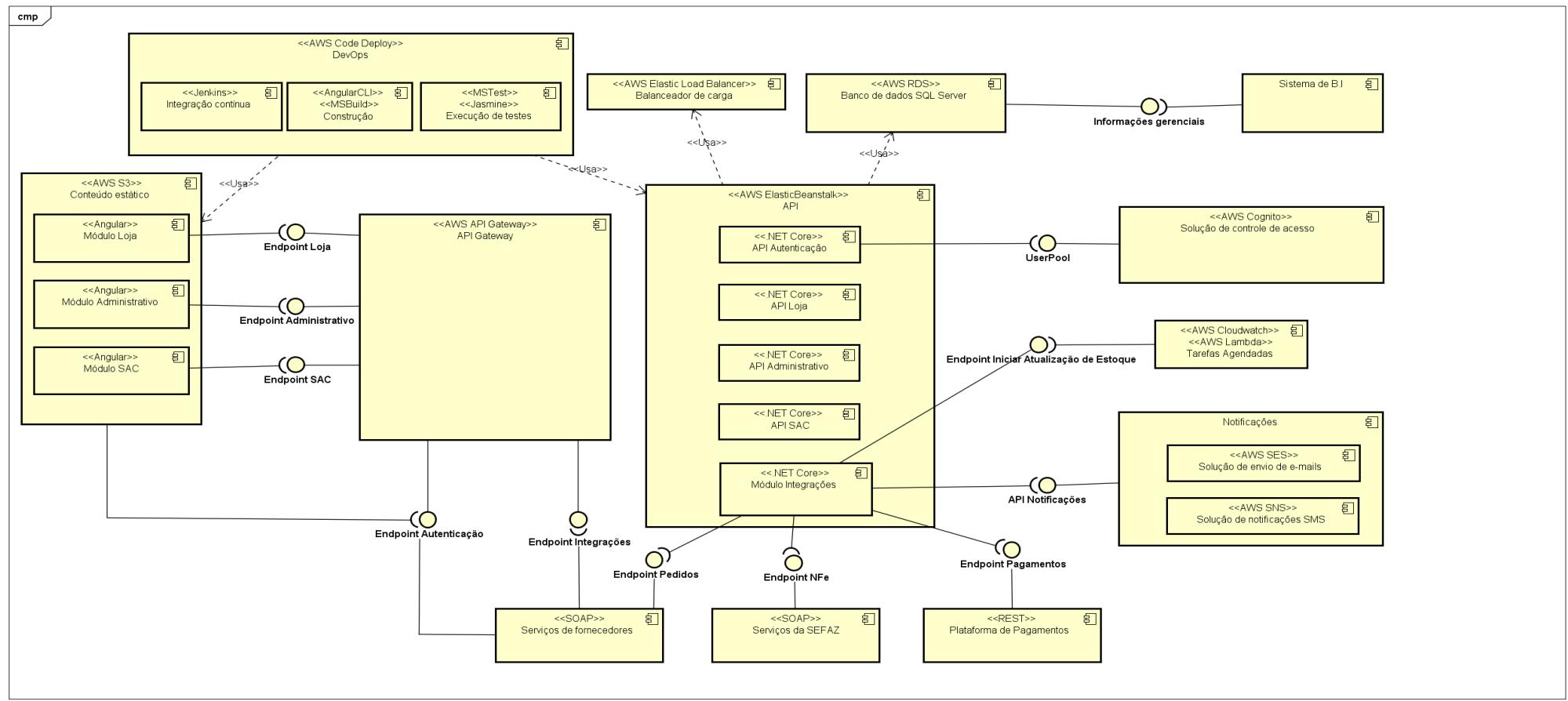
**Descrição:** Este caso de uso permitirá que o sistema envie requisições de cancelamento dos pedidos aos fornecedores através de integração, após solicitação dos usuários ou informação de pagamento recusado por parte da plataforma de pagamentos.

### **Caso de uso: Notificar mudança no status dos pedidos**

**Descrição:** Este caso de uso permitirá que fornecedores autenticados possam informar atualizações no status das entregas de seus pedidos. Toda atualização de status deverá gerar um histórico no sistema, e os usuários envolvidos no processo (clientes e vendedores) deverão ser notificados em tempo real (por e-mail e SMS, quando disponível). No caso de atualizações informando que o pedido foi cancelado por parte do fornecedor, a compra deverá ser cancelada e o estorno do pagamento deverá ser providenciado. A nota fiscal relacionada a essa compra também deverá ser cancelada.

### **4.3. Modelo de componentes**

O diagrama de componentes a seguir detalha a comunicação entre os componentes da arquitetura e suas respectivas tecnologias. Os componentes foram organizados para serem reutilizáveis e fornecem interfaces bem definidas de acordo com suas responsabilidades.



Nessa arquitetura devemos considerar a divisão do sistema em módulos implantáveis separadamente. A separação entre as aplicações de front-end e suas API's auxilia na independência entre componentes. Nas aplicações de front-end não há tratativas de negócio, deixando-as responsáveis somente pela parte comportamental e pela interação com usuários. A utilização da combinação entre Angular e Bootstrap garante uma experiência rápida e intuitiva aos usuários, além de prover a responsividade para exibição de recursos em dispositivos com quaisquer tamanhos de tela. Isso garante o atendimento aos requisitos não funcionais de usabilidade e acessibilidade.

Todas as chamadas para as API's (sejam originadas pelas aplicações de front-end ou por sistemas de fornecedores para realização de integrações) passam por um API gateway. As API's não são acessíveis externamente, a não ser através gateway. Isso facilita na auditoria e na extração de métricas de requisições por recurso, pois o ponto de entrada é único. As rotas protegidas requerem um JWT obtido através da API de autenticação, e são tratadas diretamente pelo gateway. Dessa forma, requisições sem token ou contendo tokens inválidos sequer são propagadas para a camada de API's. Os tokens são validados em um componente de controle de acessos chamado AWS Cognito, garantindo a conformidade com o requisito não funcional de segurança. O AWS Cognito consiste em uma plataforma para controle de acessos baseado em pools de usuários e grupos. A plataforma fornece uma API acessível programaticamente para criação e manutenção de usuários. O registro de usuários na loja, bem como o cadastro de fornecedores e vendedores (feitos por administradores em seu respectivo módulo) se integram a essa plataforma através da API de Autenticação. Todas as API's são provisionadas numa estrutura de balanceamento elástico de carga. Isso garante que recursos constantemente utilizados sejam escalados dinamicamente para que as requisições sejam divididas e a latência diminua. Isso garante a conformidade com o requisito não funcional de desempenho.

A aplicação conta com um único banco de dados relacional, utilizando o serviço de provisionamento de bancos de dados relacionais da AWS, o AWS RDS - Relational Database Service. A infraestrutura provida pelo serviço permite configurar replicação e backup automático dos dados, auxiliando na garantia da disponibilidade e na confiabilidade dos dados. Uma das integrações previstas para a plataforma, é com o sistema de B.I, para geração de informações gerenciais. Uma integração via mensageria, dado o volume de informações, seria inviável. Portanto, a estratégia adotada foi expor os dados necessários de maneira

controlada em estrutura pré-definida através de views no banco de dados. Apenas usuários de serviço específicos para a aplicação de BI enxergariam essas views e poderiam consumi-las em determinado período. Os dados seriam extraídos via Sqoop e descarregados na base do Hadoop do sistema de B.I.

Outras integrações previstas são com a plataforma de pagamentos online (via REST) e com os sistemas da SEFAZ para emissão e cancelamento de NFe's (via SOAP). Há ainda integrações com sistemas de fornecedores para realização e cancelamento de pedidos. Alguns desses sistemas são antigos, desenvolvidos em tecnologias que não comportam alguns padrões de comunicações modernos. Porém, mesmo aplicações COBOL podem fornecer endpoints SOAP, que são consumidos no módulo de integrações através de protocolo WS-\*, com autenticação por usuário e senha. Assume-se aqui que todos os fornecedores irão expor serviços para a plataforma em um mesmo modelo pré-definido. Isso garante a implementação do requisito não funcional de interoperabilidade. No caso da consulta de estoque, um processamento agendado será executado a cada período de tempo pré-definido, disparando uma integração com os serviços do fornecedor para atualização da disponibilidade de produtos. Esse tipo de integração não requer uma infraestrutura alocada em regime 24/7, pois o processamento é esporádico. Foi adotada então uma solução serverless (provisionamento de infraestrutura on-demand), utilizando a solução AWS Lambda. No momento em que esse processamento é disparado, aloca-se uma estrutura para realizar a comunicação com os fornecedores, atualiza-se as disponibilidades e encerra-se o ambiente, gerando custo apenas pelo processamento executado.

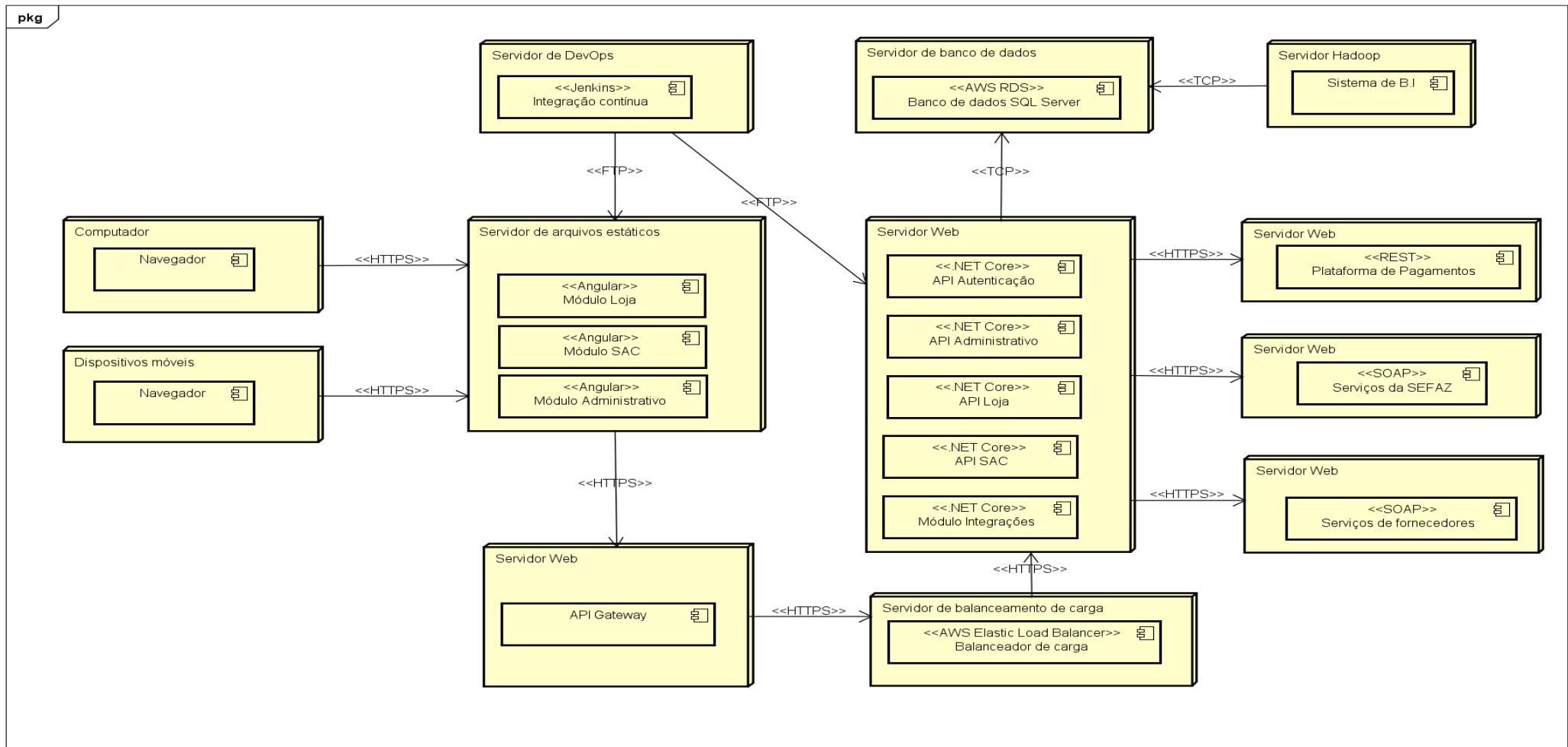
Há ainda a integração em sentido oposto entre loja e fornecedores, na qual a loja expõe através do API Gateway um endpoint de atualização de status dos pedidos. Os fornecedores precisarão seguir um fluxo que consiste em autenticar-se na API de Autenticação, obtendo um JWT e depois chamar a API de Integração, passando esse JWT no header da requisição. No momento em que uma atualização de status de uma entrega ocorre, todos os envolvidos no processo da venda devem ser notificados em tempo real. Para tal, foram utilizados serviços para envio de SMS e e-mail da AWS, que fornecem API's para consumo de suas funcionalidades.

Todos os artefatos executáveis serão construídos, testados e publicados automaticamente por um pipeline DevOps baseado na ferramenta Jenkins, também utilizando-

se de uma infraestrutura da AWS conhecida como AWS Code Deploy. O Jenkins se responsabiliza por realizar checkout dos arquivos no GIT, buildar, executar a suíte de testes automatizados e, em caso de sucesso, publicar os artefatos na plataforma da AWS, munindo-se da infraestrutura do Code Deploy. O pipeline garante os requisitos não funcionais de testabilidade e manutenibilidade (uma vez que quaisquer alterações de código-fonte são construídas, validadas e testadas automaticamente, chegando até o ambiente produtivo em questão de minutos).

#### 4.4. Modelo de implantação

O modelo de implantação auxilia no entendimento de como os componentes de software estarão disponíveis fisicamente e como a comunicação entre eles irá ocorrer. O modelo de implantação da arquitetura está documentado e detalhado abaixo.



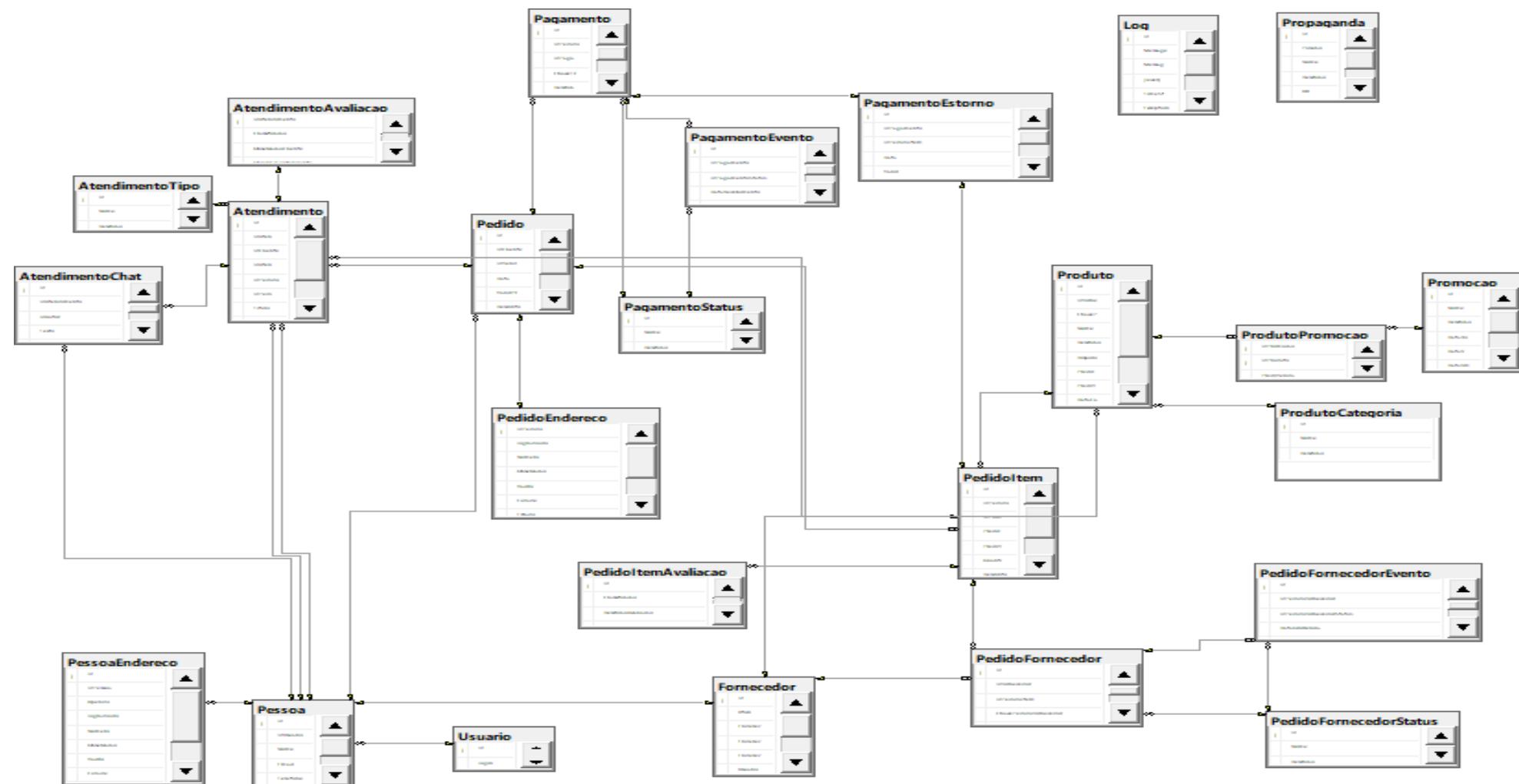
Os módulos do sistema poderiam ser implantados em infraestruturas on-premises também.

<b>Componente</b>	<b>Descrição</b>
Navegador	Representa os browsers utilizados para acesso das aplicações de front-end. Realizam a exibição do HTML para os usuários.
Servidor de arquivos estáticos	Representa servidores web que disponibilizam arquivos estáticos, como imagens, HTML, CSS e Javascript. Todos os artefatos gerados de uma construção em Angular são hospedados em servidores desse tipo. As imagens utilizadas pela aplicação também. Na prova de conceito foi utilizado um serviço da AWS chamado S3 – Simple Storage Service, que provê grande disponibilidade e cache para recursos estáticos, reduzindo também a latência.
Servidor Web	<p>São servidores que conseguem prover conteúdo estático e dinâmico através do protocolo HTTP. No caso das API's o servidor escolhido deverá ter suporte para o pipeline do .NET Core. A infraestrutura utilizada na POC é composta de servidores IIS, rodando em máquinas com S.O Windows em um ambiente clusterizado, para fornecer maior disponibilidade. Há a possibilidade de realizar a containerização desses artefatos.</p> <p>Os servidores externos (plataforma de pagamentos, SEFAZ e fornecedores) não são de nosso controle. A premissa aqui é que consigam comunicar-se via HTTPS.</p>
Servidor de balanceamento de carga	Um servidor para平衡ear as requisições que chegam entre nós do cluster de servidores Web, diminuindo a latência. Na POC foi utilizado um recurso da AWS chamado AWS Elastic Load Balancer. Porém, nada impede a utilização de outros produtos de mercado, como o NGINX. No caso da utilização de containers, poderia ser utilizado o próprio serviço de balanceamento do Docker, o Doc-

	ker Swarm.
Servidor de banco de dados	Servidor Windows que possa fornecer instância de banco de dados SQL Server. Na POC foi considerada a solução da AWS, o AWS RDS. Essa solução já provê replicação e backup automáticos.
Servidor Hadoop	Embora não seja de controle de nossa aplicação, assume-se que o sistema de Business Intelligence está em uma instância de um Hadoop. Essa instância deverá ter permissão de acesso via TCP ao banco de dados SQL Server da aplicação.

## 4.5. Modelo de dados

Como a aplicação utiliza um único banco de dados relacional, apenas um modelo de dados foi gerado, para auxiliar no entendimento da solução.



Um banco de dados relacional foi escolhido devido a sensibilidade de informações contidas na plataforma (como dados financeiros), exigindo uma consistência estrita dos dados e a atomicidade de transações realizadas. Um ponto importante a se destacar é que todos os dados de usuários são mantidos no serviço de gerenciamento de acesso, isolado do banco de dados da aplicação. No banco de dados consta apenas um identificador, para que seja possível associar um usuário a suas entidades relacionadas, como dados pessoais, endereços e pedidos.

## 5. Prova de conceito / protótipo arquitetural

### 5.1. Implementação e implantação

#### 5.1.1. Requisitos não funcionais

A prova de conceito desse projeto visa validar aspectos importantes da arquitetura que dizem respeito aos seguintes requisitos não funcionais:

- **Segurança**

Esse requisito não funcional foi escolhido devido à criticidade e a preocupação em manter os dados sensíveis seguros.

Os critérios de aceite são:

- Não permitir que usuários possam acessar páginas privadas sem estar autenticados no sistema.
- Ao identificar que um acesso a área segura está sendo feito sem autenticação, sistema deverá redirecionar para tela de autenticação.
- O sistema deverá permitir que usuários naveguem em telas públicas sem estar autenticado.

- **Usabilidade**

Esse requisito não funcional foi escolhido pois a facilidade de navegação e utilização é um ponto crucial em lojas eletrônicas para fidelização de clientes. Além disso, nos demais módulos é importantíssimo que usuários possam acessar funcionalidades de maneira simples e objetiva, auxiliando no processo de tomada de decisões.

Os critérios de aceite são:

- A tela do sistema deve apresentar facilidade de navegação.
- O usuário deve ser capaz de encontrar o produto desejado e adicioná-lo ao carrinho em no máximo cinco minutos.
- O acesso às funcionalidades devem apresentar objetividade e não serem confusos.
- **Acessibilidade**

Esse requisito não funcional foi escolhido devido a necessidade de a plataforma funcionar em ambientes responsivos, como celulares e tablets.

Os critérios de aceite são:

- Os componentes das interfaces devem se adaptar de forma que nenhuma funcionalidade seja perdida.
- A navegação deve continuar sendo simples e direta.
- A identidade visual da aplicação deve ser a mesma (fontes e cores).
- **Interoperabilidade**

Esse requisito não funcional foi escolhido pois a comunicação com sistemas de terceiros é uma das peças-chave do sistema, uma vez que dependemos de integrações com fornecedores (além de outros agentes) para que o cliente final seja atendido.

Os critérios de aceite são:

- O sistema deve conseguir se comunicar com tecnologias heterogêneas dos sistemas de fornecedores.
- O sistema deve prover uma interface de comunicação com seus fornecedores nos padrões atuais de tecnologia.

### **5.1.2. Casos de uso**

Para a realização da prova de conceito desse projeto, vários casos de uso foram implementados visando fornecer insumos para validação dos requisitos não funcionais priorizados. São eles:

Módulo	Caso de uso	Requisito não funcional
Loja	Efetuar login	Segurança
	Consultar compras	Segurança, Usabilidade e Acessibilidade
	Visualizar catálogo	Usabilidade e Acessibilidade
	Adicionar produtos ao carrinho	Usabilidade e Acessibilidade
Integrações	Consultar disponibilidade de produtos	Interoperabilidade
	Notificar mudança de status da entrega	Interoperabilidade

Para uma descrição dos casos de uso, consultar a seção [4.2](#) deste documento.

### 5.1.3. Tecnologias utilizadas

As seguintes tecnologias foram utilizadas na implementação da prova de conceito:

Caso de uso	Tecnologias
Efetuar login	Angular, Bootstrap, .NET Core, AWS API Gateway e AWS Cognito
Consultar compras	Angular, Bootstrap, .NET Core, EF Core, AWS RDS (SQL Server), AWS API Gateway e AWS Cognito
Visualizar catálogo	Angular e Bootstrap
Adicionar produtos ao carrinho	Angular e Bootstrap
Consultar disponibilidade de produtos	.NET Core e AWS Lambda
Notificar mudança de status da entrega	.NET Core, ADO.NET, AWS RDS (SQL Server), AWS API Gateway, AWS Cognito, AWS SNS e AWS SES

### 5.1.4. Implantação

Toda a prova de conceito desenvolvida foi implantada na nuvem, aproveitando-se da infraestrutura provida pela Amazon em seu produto AWS – Amazon Web Services (inclusive um serviço escrito em WCF para simular um fornecedor). A tabela a seguir detalha cada implementação e seu recurso de implantação:

Implementação	Recurso de implantação
Loja (front-end)	AWS S3
API Loja	AWS API Gateway / AWS ElasticBeanstalk
API Autenticação	AWS API Gateway / AWS ElasticBeanstalk
API Integrações	AWS API Gateway / AWS ElasticBeanstalk
Função Serverless para consulta de estoque	AWS Lambda
Mock do serviço de consulta de estoque dos fornecedores	AWS API Gateway / AWS ElasticBeanstalk
Banco de dados SQL Server	AWS RDS

Os links para acesso a plataforma encontram-se no apêndice desse documento.

## 5.2. Interfaces/ APIs

### 5.2.1. Consumidas

Para a implementação da prova de conceito foi feita a simulação de consumo de uma API dos fornecedores baseada no protocolo SOAP. A API tem por objetivo simular uma consulta à disponibilidade de um produto na base do fornecedor. Essa API foi desenvolvida em .NET Standart, utilizando a plataforma de serviços WCF – Windows Communication Foundation. Sua interface é a seguinte:

Entrada			
Parâmetro	Tipo	Obrigatoriedade	Descrição
Login	Texto	Sim	Usuário de serviço provido pelo fornecedor. Deve ser enviado no header do envelope SOAP. Dado registrado na tabela <b>Fornecedor</b> da base de dados.
Senha	Texto	Sim	Senha do usuário de serviço. Deve ser enviado no header do envelope SOAP. Dado registrado na tabela <b>Fornecedor</b> da base de dados.
ChaveProduto	Texto	Sim	Chave do produto na base de dados do fornecedor. Dado registrado na tabela <b>Produto</b> da base de dados.

Saída
-------

Parâmetro	Tipo	Descrição
Status	Booleano	VERDADEIRO indica que o produto está disponível.

Dados para teste	
Parâmetro	Tipo
URL	<a href="http://dropshippingfornecedor-env.pr3jhpmmpd.us-east-1.elasticbeanstalk.com/EstoqueService.svc">http://dropshippingfornecedor-env.pr3jhpmmpd.us-east-1.elasticbeanstalk.com/EstoqueService.svc</a>
Método	ConsultarDisponibilidade
Usuário	lojapucminas
Senha	123mudar
ChaveProduto	PROD0001

**Observação:** possivelmente no momento da validação dos testes a aplicação de mock dos fornecedores esteja desprotegida, sem autenticação. Isso pode ocorrer por conta da necessidade de renovação de um certificado digital para assinatura das mensagens SOAP na infraestrutura da AWS. Caso ocorra uma recusa da chamada por motivos de segurança, gentileza efetuar a chamada novamente sem enviar parâmetros de autenticação no cabeçalho da mensagem.

### 5.2.2. Fornecidas

Na implementação da prova de conceito, foram criadas algumas API's Restful para consumo de fornecedores, as quais estão documentadas a seguir:

- **Autenticação (POST)**

Entrada			
Parâmetro	Tipo	Obrigatoriedade	Descrição
Usuario	Texto	Sim	Usuário que deseja se autenticar.
Senha	Texto	Sim	Senha do usuário.

Saída	

Parâmetro	Tipo	Descrição
Status	Numérico	<ul style="list-style-type: none"> <li>1. Indica que a autenticação foi bem-sucedida.</li> <li>2. Indica que a autenticação falhou por falha nas credenciais.</li> <li>3. Indica que a autenticação falhou por um erro na API.</li> </ul>
Token	Texto	JWT com as permissões e informações do usuário autenticado. Só é preenchido quando o campo Status é retornado como 1.

Dados para teste	
Parâmetro	Tipo
URL	<a href="https://x4e23klt10.execute-api.us-east-1.amazonaws.com/aut-dev/api/v1/usuarios/autenticar">https://x4e23klt10.execute-api.us-east-1.amazonaws.com/aut-dev/api/v1/usuarios/autenticar</a>
Método	POST
Login	integracoes.sapatop
Senha	Puc123@Teste

- **Atualizar status da entrega (POST)**

Entrada			
Parâmetro	Tipo	Obrigatoriedade	Descrição
ChavePedidoFornecedor	Texto	Sim	Chave do pedido que se deseja informar uma mudança de status.
Status	Numérico	Sim	Status atualizado do pedido: <ul style="list-style-type: none"> <li>0. Pedido recebido</li> <li>1. Pedido recusado</li> <li>2. Pedido confirmado</li> <li>3. Pedido embalado</li> <li>4. Pedido expedido</li> <li>5. Pedido em transporte</li> <li>6. Pedido entregue</li> </ul>
Informações adicionais	Texto	Não	Informações adicionais sobre a mudança de status que ficarão registradas no

			sistema
Token	Texto	Sim	Token gerado por integração com a API de autenticação. Deve ser enviado no header “Authorization” da requisição HTTP no formato “Bearer + espaço + TOKEN”.

<b>Saída</b>		
<b>Parâmetro</b>	<b>Tipo</b>	<b>Descrição</b>
IdEventoPedidoRegistrado	Numérico	Identificador do evento registrado, caso o fornecedor queira armazenar para fins de auditoria.
ChavePedidoFornecedor	Texto	A chave que foi enviada na entrada.

<b>Dados para teste</b>	
<b>Parâmetro</b>	<b>Tipo</b>
URL	<a href="https://wngtbhofq5.execute-api.us-east-1.amazonaws.com/int-dev/api/v1/pedidos/registrarevento">https://wngtbhofq5.execute-api.us-east-1.amazonaws.com/int-dev/api/v1/pedidos/registrarevento</a>
Método	POST
Token	Obter via API de autenticação
ChavePedidoFornecedor	PUCDROPPED001
Status	5

## 6. Avaliação da Arquitetura

### 6.1. Análise das abordagens arquiteturais

A arquitetura proposta contempla uma série de componentes, todos modulares. Cada componente tem seu próprio conjunto de tecnologias e características de implantação. Apesar da utilização de vários componentes proprietários e de uma infraestrutura em nuvem, toda a implementação foi feita de forma que seja o mais independente de plataforma possível. Foi

ainda elaborada uma esteira DevOps para auxiliar no transporte de evoluções para os ambientes produtivos, com maior assertividade e garantia de qualidade através da execução de testes automatizados.

## 6.2. Identificação dos atributos de qualidade

Os atributos identificados estão relacionados aos requisitos listados na seção [5.1.1](#): segurança, usabilidade, acessibilidade e interoperabilidade.

## 6.3. Cenários

**Cenário 1:** Ao realizar o acesso a uma URL ou página, o sistema deve apresentar altos padrões de segurança, garantindo que o usuário possa acessar as páginas seguras apenas se estiver autenticado no sistema. O sistema deve redirecionar o usuário para a tela de autenticação quando forem feitas tentativas de acesso à páginas privadas sem credenciais válidas. O sistema deverá garantir que as páginas públicas possam ser acessadas sem necessidade de autenticação. Esta será a garantia de que o requisito não funcional de segurança foi satisfeito.

**Cenário 2:** Ao navegar na tela, o sistema deve apresentar boa usabilidade. A navegação deve apresentar facilidade e o acesso às funcionalidades deve ser intuitivo e objetivo. O usuário deve conseguir identificar e adicionar um produto a seu carrinho, estando pronto para concluir a compra em no máximo cinco minutos. Esta será a garantia de que o requisito não funcional de usabilidade foi satisfeito.

**Cenário 3:** Ao acessar a aplicação através de um dispositivo móvel ou desktop com resolução reduzida, a tela do usuário deverá se adaptar automaticamente, redimensionando seus componentes visuais de acordo com a resolução, porém sem perder funcionalidades ou complicar a navegação. Esta é a garantia de que o requisito não funcional de acessibilidade foi satisfeito.

**Cenário 4:** A aplicação deve conseguir se comunicar com sistemas de tecnologias heterogêneas. No caso da consulta de estoque automática, o sistema deverá conseguir acessar o serviço dos fornecedores e obter uma resposta válida. Essa comunicação bem-sucedida, juntamente com o cenário 5, são as garantias de que o requisito não-funcional de interoperabilidade foi atendido.

**Cenário 5:** A aplicação deve prover recursos para que a comunicação entre sistemas terceiros e a plataforma ocorra dentro de padrões atuais de tecnologias. Um fornecedor que vai informar uma mudança no status de um pedido para a aplicação deve conseguir se comunicar desde que esteja autenticado e possua uma compatibilidade com API's RESTful. Fornecedores com credenciais inválidas não poderão informar alterações de status em pedidos. Além disso, cada fornecedor só poderá informar atualizações de seus próprios pedidos. A conclusão desse cenário, juntamente com o cenário 4, são as garantias de que o requisito não funcional de interoperabilidade foi satisfeito (além de estarem em conformidade com o requisito não funcional de segurança).

Na priorização foi utilizado o método de árvore de utilidades reduzida e com prioridades. Os atributos foram categorizados de acordo os requisitos a que estão relacionados, e então foram classificados em função de sua importância e complexidade (considerando a percepção de negócio e arquitetura). As duas variáveis de priorização são "Importância" (IMP) e "Complexidade" (COM). As classificações possíveis são "Alta" (A), "Média" (M) e "Baixa" (B).

Categoria	Atributo de qualidade	Cenário	IMP	COM
Confidencialidade	Segurança	1. O sistema deve apresentar altos padrões de segurança.	A	A
Funcionalidade	Usabilidade	2. O sistema deve prover boa usabilidade.	A	M
	Acessibilidade	3. O sistema deve suportar ambientes web responsivos e ambientes móveis.	M	A
Compatibilidade	Interoperabilidade	4. O sistema deve se comunicar com vários sistemas, com tecnologias heterogêneas.	A	A

		5. O sistema deve prover recursos para receber comunicação nos padrões atuais de tecnologias.	M	M
--	--	---	---	---

#### 6.4. Avaliação

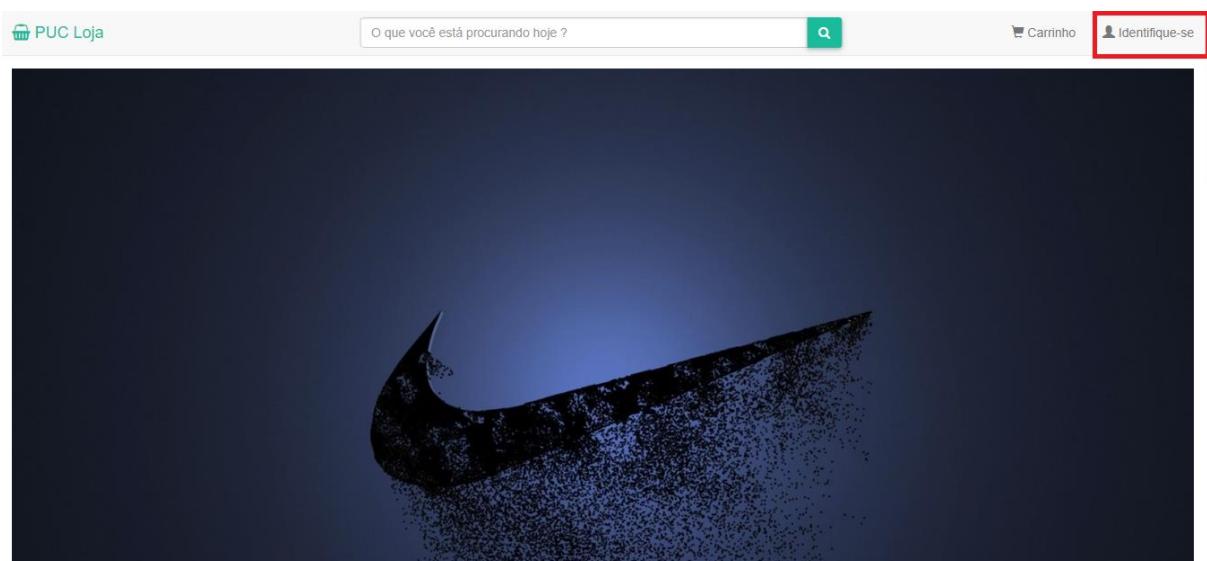
Processo de avaliação dos cenários identificados no item [6.3](#). O objetivo aqui é determinar os riscos, não riscos, pontos de sensibilidade, trade-off's e evidenciar o atendimento aos requisitos de qualidade.

- **Cenário 1**

<b>Atributo de qualidade:</b>	Segurança
<b>Requisito de qualidade:</b>	O sistema deve apresentar altos padrões de segurança.
<b>Preocupação</b>	Impossibilitar o acesso a páginas privadas do sistema sem autenticação no sistema.
<b>Cenário 1</b>	
Ambiente:	
Sistema em operação normal	
<b>Estímulo:</b>	
Usuário tentando acessar uma página privada sem estar autenticado no sistema.	
<b>Mecanismo:</b>	
Criar um mecanismo de validação de credenciais (tokens, no caso) e suas permissões associadas para acessar recursos protegidos.	
<b>Medida de resposta:</b>	
O usuário deve ser redirecionado para tela de autenticação.	
<b>Considerações sobre a arquitetura:</b>	

<b>Riscos:</b>	O gerenciamento de autenticação e autorização são pontos críticos para a segurança na web. Falhas nessa área tipicamente envolvem vazamento de credenciais e informações sensíveis, que em mãos erradas causarão grande impacto. A utilização de JWT's é uma maneira simples e rápida de proteger API's de acessos indevidos. Porém o algoritmo de criptografia e a chave utilizada para assinar o token devem estar bem seguros, garantindo assim que ninguém conseguirá forjar um token manualmente.
<b>Pontos de sensibilidade:</b>	Servidor de aplicação operando em modo HTTPS.
<b>Trade-off:</b>	Não há.

### Evidências do cenário 1:

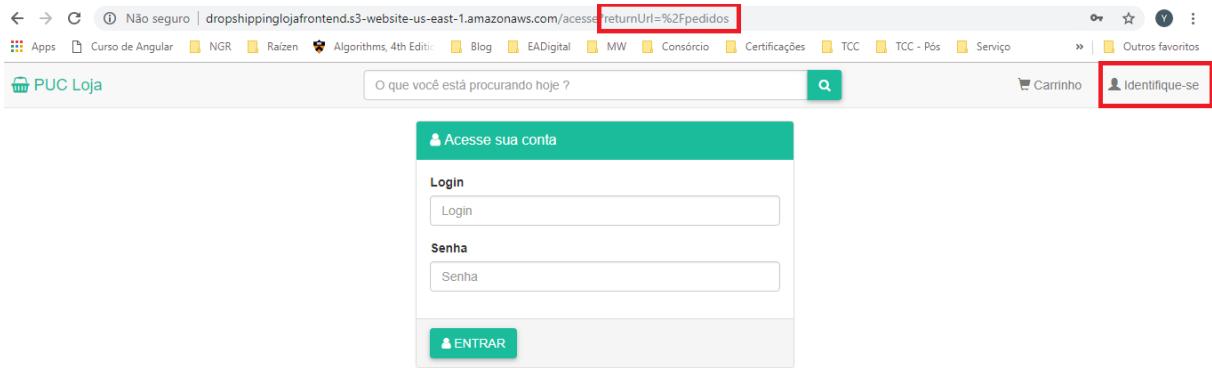


**Etapa 1: Usuário não identificado acessando página inicial do sistema.**

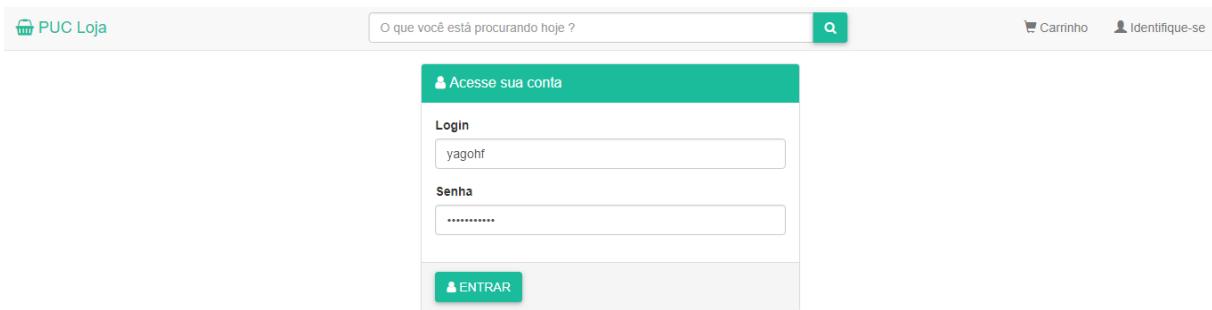
### Etapa 2: Usuário não autenticado acessando catálogo de produtos.

### Etapa 3: Usuário não autenticado adicionou produtos ao carrinho e consegue visualizá-los, alterar sua quantidade ou removê-los.

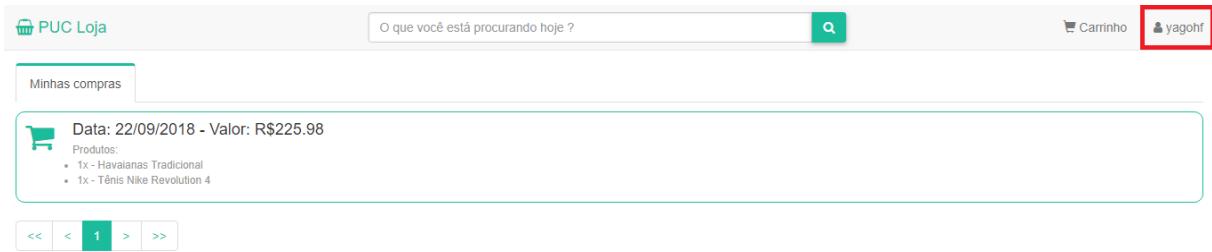
### Etapa 4: Evidência de usuário não autenticado (não há token na localstorage do browser, apenas o carrinho).



**Etapa 5: Usuário tenta acessar diretamente a URL de pedidos (/pedidos), sem estar autenticado e é redirecionado para tela de login.**



**Etapa 6: Usuário informa suas credenciais e clica em “Entrar”**



**Etapa 7: Usuário é redirecionado para a URL que tentava acessar e seu login passa a ser exibido no canto superior direito.**

The screenshot shows a browser interface with a search bar and a shopping cart icon labeled 'PUC Loja'. Below the search bar is a button 'Minhas compras'. Underneath is a summary box for a purchase on 'Data: 22/09/2018 - Valor: R\$225.98' containing two items: '1x - Havaianas Tradicional' and '1x - Tênis Nike Revolution 4'. At the bottom are navigation buttons '<< < 1 > >>'. Below this is the browser's developer tools Application tab, specifically the Local Storage section. A red box highlights the 'carrinho' key, which has a value of a JSON array: '[{"id": 3, "nome": "T\u00e9nis Kappa Impact - Branco e Preto", "preco": 86.99, "quantidade": 1, ...}, {"id": 3, "nome": "T\u00e9nis Kappa Impact - Branco e Preto", "preco": 86.99, "quantidade": 1, ...}, {"id": 4, "nome": "T\u00e9nis Olympikus Spirit 2 Masculino", "preco": 131.99, "quantidade": 1, ...}]'. This indicates that a new entry was added to the local storage.

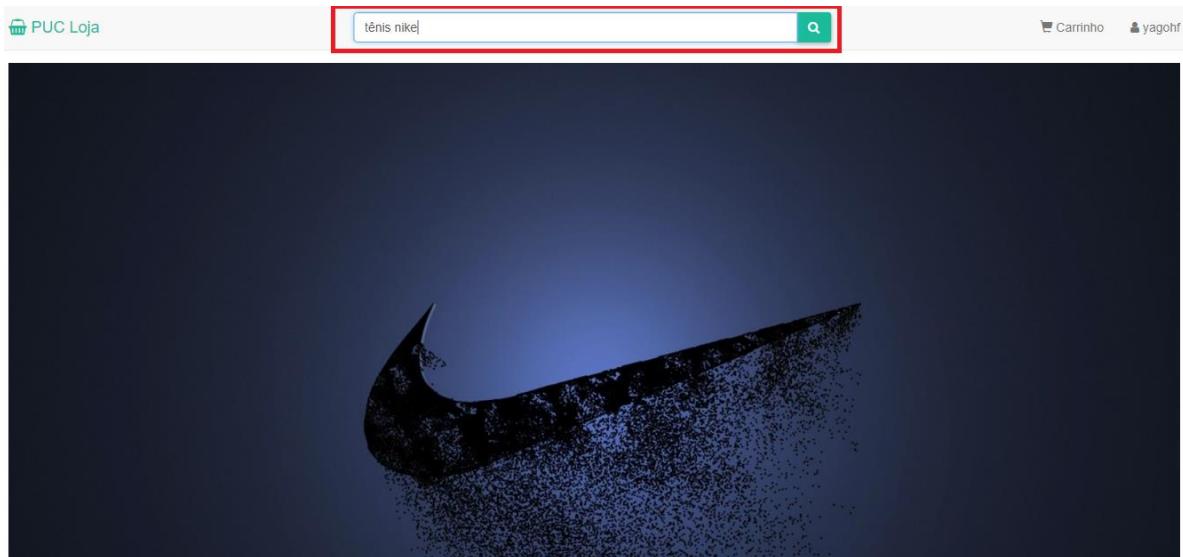
**Etapa 8: Uma nova entrada é gerada na localStorage do browser, correspondente ao token gerado pela API de autenticação. Toda requisição ao back-end de agora em diante envia esse token no header “Authorization”, garantindo assim que credenciais válidas foram informadas.**

- **Cenário 2**

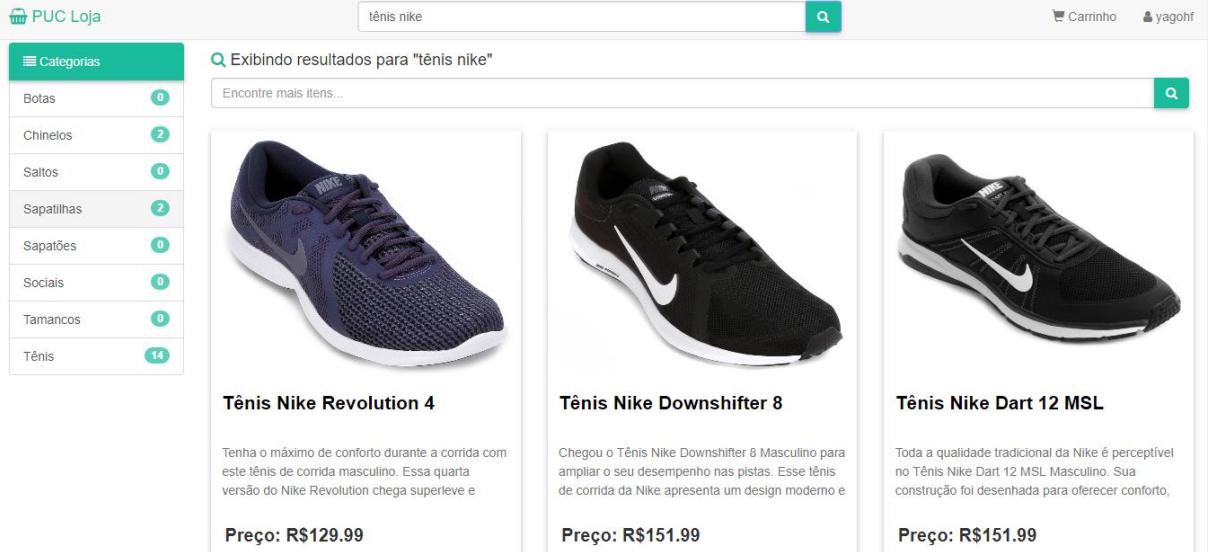
<b>Atributo de qualidade:</b>	Usabilidade
<b>Requisito de qualidade:</b>	O sistema deve prover boa usabilidade.
<b>Preocupação</b>	
Fornecer interfaces simples para agilizar a navegação e tornar a experiência do usuário bem rápida e objetiva.	
<b>Cenário 2</b>	
<b>Ambiente:</b>	
Sistema em operação normal	
<b>Estímulo:</b>	
Usuário buscando produtos e adicionando ao carrinho.	
<b>Mecanismo:</b>	
Criação de telas simples e objetivas. Menus de navegação visíveis durante todo o tempo, possibilitando efetuar uma busca a qualquer momento. Carregamento de todo o conteúdo estático (HTML, CSS e Javascript) do site no momento do início da navegação, fazendo com que as requisições seguintes aos servidores de back-end tragam apenas dados em formato JSON, o que agiliza o carregamento das páginas.	
<b>Medida de resposta:</b>	

<p>O usuário deve encontrar o que busca de maneira rápida, adicionando produtos ao carrinho e estando pronto para concluir o processo de compra em no máximo cinco minutos.</p>	
<b>Considerações sobre a arquitetura:</b>	
<b>Riscos:</b>	Pode ocorrer algum pico de memória no servidor ou um número de usuários muito grande ocasionando sobrecarga no servidor de aplicação, tornando os processamentos para obtenção de dados mais lentos por um período curto de tempo, prejudicando a experiência do usuário.
<b>Pontos de sensibilidade:</b>	Balanceamento de carga ativo.
<b>Trade-off:</b>	Não há.

### Evidências do cenário 2:



**Etapa 1:** Usuário acessa a página inicial da loja e já informa os parâmetros que deseja para encontrar o produto desejado (sem sequer precisar scrollar). Depois clica no botão de lupa para iniciar a pesquisa.



PUC Loja

tênis nike

Exibindo resultados para "tênis nike"

Carrinho yagohf

Categorias	
Botas	0
Chinelos	2
Saltos	0
Sapatinhas	2
Sapatos	0
Sociais	0
Tamancos	0
Tênis	14



**Tênis Nike Revolution 4**

Tenha o máximo de conforto durante a corrida com este tênis de corrida masculino. Essa quarta versão do Nike Revolution chega superleve e

Preço: R\$129.99

[ADICIONAR](#)



**Tênis Nike Downshifter 8**

Chegou o Tênis Nike Downshifter 8 Masculino para ampliar o seu desempenho nas pistas. Esse tênis de corrida da Nike apresenta um design moderno e

Preço: R\$151.99

[ADICIONAR](#)



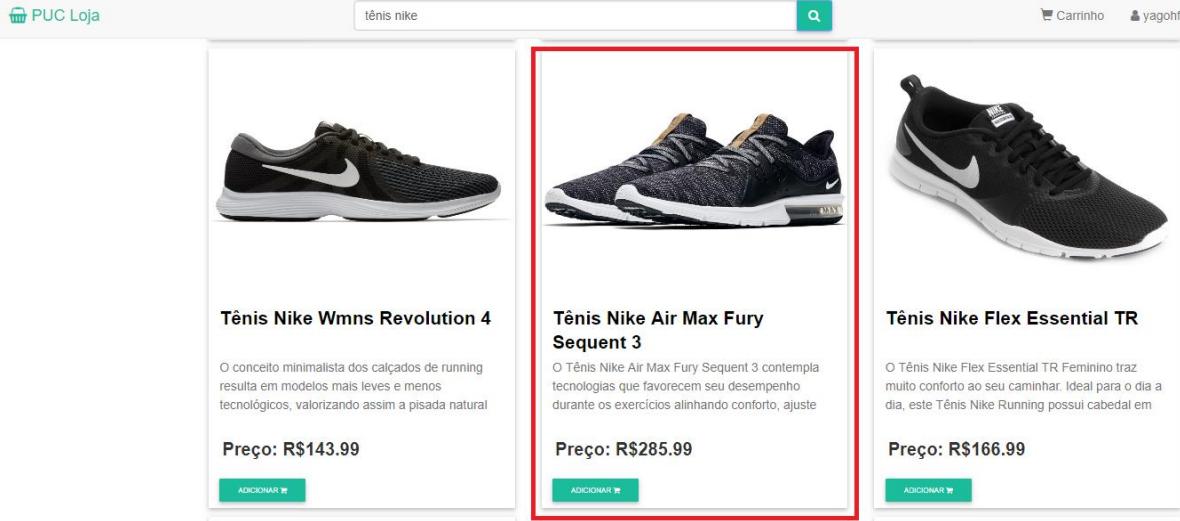
**Tênis Nike Dart 12 MSL**

Toda a qualidade tradicional da Nike é perceptível no Tênis Nike Dart 12 MSL Masculino. Sua construção foi desenhada para oferecer conforto,

Preço: R\$151.99

[ADICIONAR](#)

**Etapa 2: Catálogo é exibido com os resultados da pesquisa efetuada. Usuário pode refinar ainda mais sua pesquisa diretamente no catálogo e o menu ainda está visível.**



PUC Loja

tênis nike

Carrinho yagohf



**Tênis Nike Wmns Revolution 4**

O conceito minimalista dos calçados de running resulta em modelos mais leves e menos tecnológicos, valorizando assim a pisada natural

Preço: R\$143.99

[ADICIONAR](#)



**Tênis Nike Air Max Fury Sequent 3**

O Tênis Nike Air Max Fury Sequent 3 contempla tecnologias que favorecem seu desempenho durante os exercícios alinhando conforto, ajuste

Preço: R\$285.99

[ADICIONAR](#)



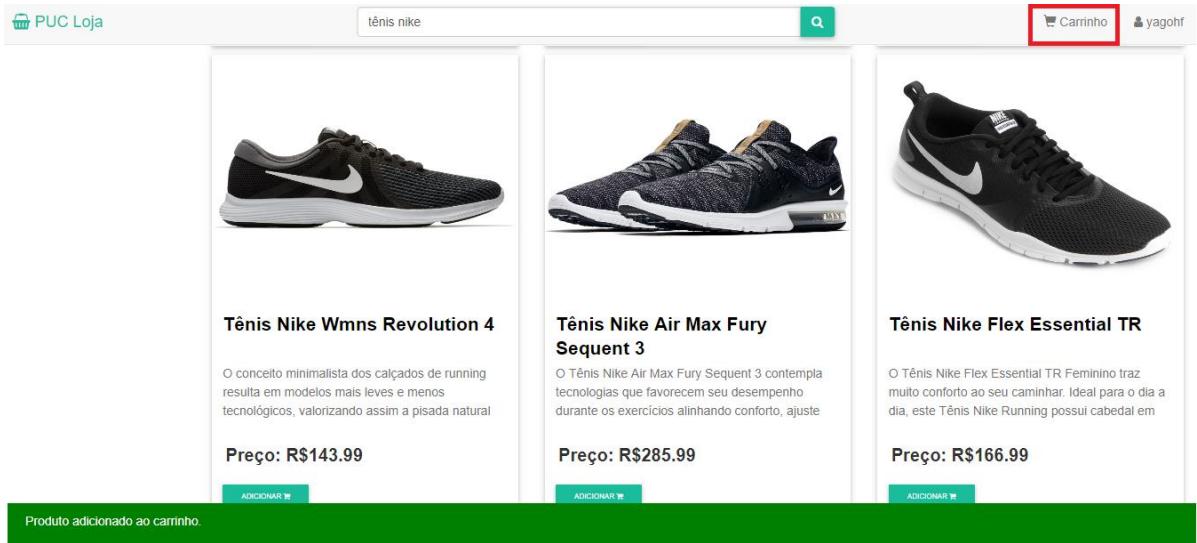
**Tênis Nike Flex Essential TR**

O Tênis Nike Flex Essential TR Feminino traz muito conforto ao seu caminhar. Ideal para o dia a dia, este Tênis Nike Running possui cabedal em

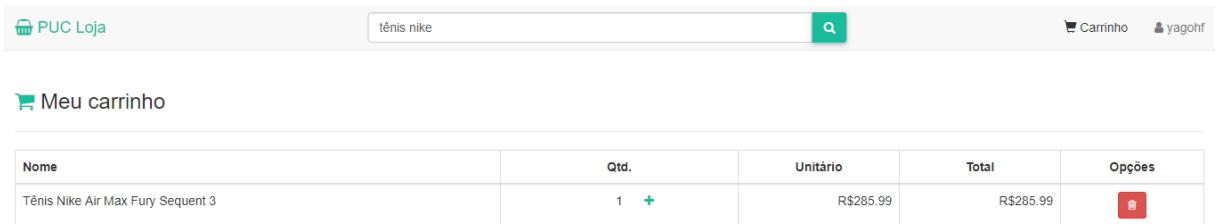
Preço: R\$166.99

[ADICIONAR](#)

**Etapa 3: Usuário scrolla a página e encontra o produto desejado (e o menu segue visível). Feito isso, clica em “Adicionar” para incluí-lo no carrinho.**



**Etapa 4: Sistema exibe mensagem de produto adicionado ao carrinho. Usuário então clica no menu “Carrinho”.**



**Etapa 5: Carrinho é exibido com o produto adicionado. Para chegar até aqui foram necessários apenas três cliques e um scroll. O usuário então está pronto para concluir o processo de compra.**

- **Cenário 3**

<b>Atributo de qualidade:</b>	Acessibilidade
<b>Requisito de qualidade:</b>	O sistema deve suportar ambientes web responsivos e ambientes móveis.
<b>Preocupação</b>	
O sistema deve se adaptar a interfaces de diversos tamanhos sem perda de funcionalidade e sem causar impactos na qualidade de navegação.	
<b>Cenário 3</b>	
Ambiente:	
Sistema em operação normal	
<b>Estímulo:</b>	
Usuário buscando produtos e adicionando ao carrinho.	

<b>Mecanismo:</b>	Criação de telas utilizando mecanismos de design responsivos e ajustáveis, movimentando os componentes para que caibam em dispositivos diferentes.
<b>Medida de resposta:</b>	O sistema deve se adaptar a resoluções de tela dos diversos dispositivos, sem perder funcionalidades.
<b>Considerações sobre a arquitetura:</b>	
<b>Riscos:</b>	A experiência de navegação pode ser altamente prejudicada pela qualidade da rede em que se está tentando o acesso. Além disso, resoluções extremamente pequenas de dispositivos muito抗igos poderão causar alguns deslocamentos indesejáveis de componentes (porém sem a perda das funcionalidades).
<b>Pontos de sensibilidade:</b>	Não há.
<b>Trade-off:</b>	Não há.

### Evidências do cenário 3:

60 ⌂ 11:29

(i) te-us-east-1.amazonaws.com 1

PUC Loja

Categorias	
Botas	0
Chinelos	2
Saltos	0
Sapatilhas	2
Sapatos	0

< O □

**Etapa 1:** Usuário acessa a aplicação por um dispositivo móvel.

PUC Loja

## Categorias

Botas	0
Chinelos	2
Saltos	0
Sapatilhas	2
Sapatões	0
Sociais	0
Tamancos	0
Tênis	14

## Promoções

**Etapa 2:** Usuário seleciona a categoria de produtos desejada (no caso “Tênis”), e é direcionado para o catálogo.

80 11:30

PUC Loja

Exibindo resultados para "Tênis"

Encontre mais itens...

**Tênis Kappa Impact - Branco e Preto**

O Tênis Kappa Impact Masculino feito para homens quem buscam conforto e estilo para suas caminhadas e atividades esportivas,



Etapa 3: Usuário visualiza o catálogo exibindo os resultados para a categoria selecionada.



 PUC Loja



## Tênis Nike Revolution 4

Tenha o máximo de conforto durante a corrida com este tênis de corrida masculino. Essa quarta versão do Nike Revolution chega

**Preço: R\$129.99**

[ADICIONAR !\[\]\(e8bcaa650ebf5bf989653962c35b1062\_img.jpg\)](#)

**Etapa 4:** Usuário desliza até o produto desejado e toca em “Adicionar”.



The screenshot shows a mobile application interface for 'PUC Loja'. At the top, there is a navigation bar with icons for back, forward, and search, along with signal strength and battery status. The time '11:35' is also displayed. The main content area features a large image of a dark blue Nike Revolution 4 running shoe. Below the image, the text 'Tênis Nike Revolution 4' is displayed in bold black font. A descriptive paragraph in gray text follows: 'Tenha o máximo de conforto durante a corrida com este tênis de corrida masculino. Essa quarta versão do Nike Revolution chega'. A green horizontal bar at the bottom contains the price 'Preço: R\$129,99' and a message 'Produto adicionado ao carrinho.' (Product added to cart). The bottom of the screen shows standard Android navigation buttons for back, home, and recent apps.

**Etapa 5:** Sistema exibe mensagem informando que o produto foi adicionado ao carrinho.  
Usuário então clica sobre o menu no canto superior direito.

67 11:35

PUC Loja

Carrinho

yagohf

O que você está procurando hoje ?

## Tênis Nike Revolution 4

Tenha o máximo de conforto durante a corrida com este tênis de corrida masculino. Essa quarta versão do Nike Revolution chega

**Preço: R\$129.99**

ADICIONAR

Etapa 6: No menu, usuário seleciona a opção “Carrinho”.



11:35

PUC Loja



## Meu carrinho

Nome	Qtd.	Total	Opcões
Tênis Nike Revolution 4	1 	R\$129.99	



Etapa 7: O carrinho de compras é exibido com o produto adicionado.

- Cenário 4

<b>Atributo de qualidade:</b>	Interoperabilidade
<b>Requisito de qualidade:</b>	O sistema deve se comunicar com vários sistemas, com tecnologias heterogêneas.
<b>Preocupação</b>	
Possibilitar que o sistema realize integrações com fornecedores, mesmo que estes disponham de tecnologias bastante antigas.	
<b>Cenário 1</b>	
Ambiente:	
Sistema em operação normal	
<b>Estímulo:</b>	
Sistema executando processamento agendado de consulta de estoque dos fornecedores.	
<b>Mecanismo:</b>	
Criar um mecanismo de acesso aos sistemas de fornecedores utilizando o padrão de mensagens SOAP, interoperável com tecnologias legadas.	
<b>Medida de resposta:</b>	
Sistema conseguiu a comunicação e obteve o status atualizado dos produtos.	
<b>Considerações sobre a arquitetura:</b>	
<b>Riscos:</b>	Comunicações via rede estão sujeitas à instabilidade e indisponibilidade do ambiente a que se deseja acessar. Se não houver um mecanismo de retry para chamadas sem sucesso, pode-se acabar um processamento com os dados em um estado inconsistente.
<b>Pontos de sensibilidade:</b>	Não há.
<b>Trade-off:</b>	Não há.

#### Evidências do cenário 4:

The screenshot shows a SQL Server Management Studio interface. The top window is titled 'SQLQuery1.sql - ya...pping (yagohf (71))' and contains the following SQL code:

```

1 | SELECT *
2 |   FROM Produto
3 | WHERE Disponivel = 0

```

The bottom window is titled 'Resultados' and displays a grid of product data. The columns are: Id, IdFornecedor, ChaveProdutoFornecedor, Nome, Descricao, Disponivel, PrecoCusto, PrecoVenda, DataCadastro, IdProdutoCategoria, Ativo. Two rows of data are shown:

Id	IdFornecedor	ChaveProdutoFornecedor	Nome	Descricao	Disponivel	PrecoCusto	PrecoVenda	DataCadastro	IdProdutoCategoria	Ativo
1	5	2	PROD0005	Tênis Nike Revolution 4	0	179.99	215.99	2018-09-24 16:44:34.040	2	1
2	18	2	PROD0018	Sapatilha Pietra Fernandes Biqueira Lacinho	0	39.99	47.99	2018-09-18 16:44:34.040	3	0

**Etapa 1: Evidência de que o produto “Tênis Nike Revolution 4” está indisponível (campo “Disponível” = 0) na base de dados, porém está ativo (o que o torna elegível para busca de disponibilidade no serviço do fornecedor).**

**Etapa 2: Catálogo evidenciando que o produto não foi encontrado. Reparar na quantidade de itens disponíveis para a categoria “Tênis” (13).**

**Etapa 3: Função serverless na plataforma AWS Lambda para disparar processamento de atualização de disponibilidade (ConsultarEstoque). Essa função é disparada em um período configurável através de um serviço de schedule da AWS chamado CloudWatch.**

**Etapa 4: Interface de detalhes da função lambda. Vamos efetuar um teste manual, clicando sobre o botão “Teste”.**

The screenshot shows the AWS Lambda console interface. In the top navigation bar, 'Services' is selected under 'Resource Groups'. The ARN of the function is shown as 'arn:aws:lambda:us-east-1:775424642702:function:ConsultarEstoque'. Below the ARN, there are tabs for 'Controlar', 'Qualificadores', 'Ações', 'EvtTesteConsultarEsto...', 'Teste' (which is currently selected), and 'Salvar'. The main content area is titled 'ConsultarEstoque' and displays the results of a recent execution. A green checkmark indicates success, and the log output shows the single character '1'. Below the log, a summary table provides execution details:

Resumo	
Código SHA-256	caa2994b-c71e-11e8-9bd0-d52bc07fb790
AApUARvMPb6vNac6ZyW3Kzt0lQrU27mE8vrGyqDFsXE=	ID da solicitação
Duração	500 ms
469.65 ms	Período cobrado
Recursos configurados	Memória máxima usada
256 MB	67 MB

**Etapa 5: Resultado do teste indicando que a função foi executada com sucesso.**

The screenshot shows a SQL Server Management Studio (SSMS) window. The query pane contains the following T-SQL code:

```

1 | SELECT *
2 | FROM Produto
3 | WHERE Disponivel = 0

```

The results pane shows a table with one row of data:

	Id	IdFornecedor	ChaveProdutoFornecedor	Nome	Descricao	Disponivel	PrecoCusto	PrecoVenda	DataCadastro	IdProdutoCategoria	Ativo
1	18	2	PROD0018	Sapatilha Pietra Femandes Biqueira Lacinha	O estilo delicado da Sapatilha Pietra Femandes ...	0	39.99	47.99	2018-09-18 16:44:34.040	3	0

**Etapa 6: Consulta ao banco de dados para exibir os produtos inativos. Reparar que o produto “Tênis Nike Revolution 4” sumiu da lista de inativos, pois sua disponibilidade foi atualizada após consulta ao serviço de seu fornecedor.**

PUC Loja

Tênis Nike Revolution 4

Exibindo resultados para "Tênis Nike Revolution 4"

Encontre mais itens...

**Tênis Nike Revolution 4**

Tenha o máximo de conforto durante a corrida com este tênis de corrida masculino. Essa quarta versão do Nike Revolution chega superleve e

Preço: R\$129.99

**Etapa 7: Ao realizar a mesma pesquisa na loja, produto passa a aparecer como disponível.**

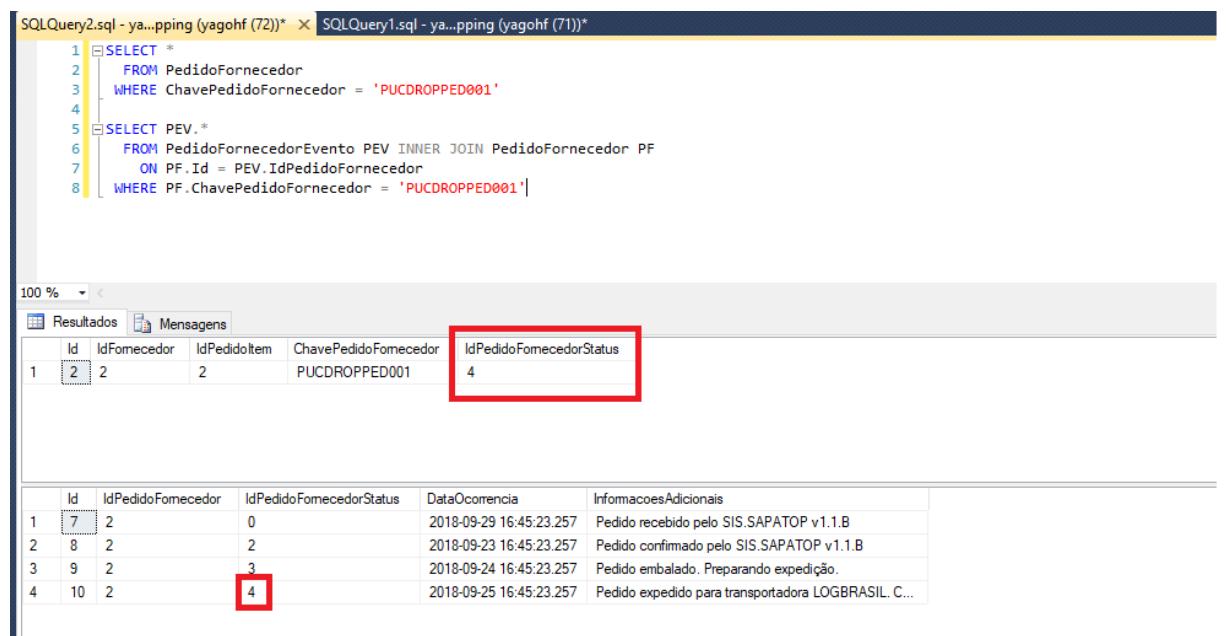
**Reparar que a quantidade de produtos ativos da categoria “Tênis” também mudou.**

- **Cenário 5**

<b>Atributo de qualidade:</b>	Interoperabilidade
<b>Requisito de qualidade:</b>	O sistema deve prover recursos para que a comunicação no sentido “sistema terceiro – loja” ocorra dentro de padrões atuais de tecnologias.
<b>Preocupação</b>	Possibilitar que o sistema receba integrações (de fornecedores e da plataforma de pagamentos), provendo endpoints construídos com conceitos modernos.
<b>Cenário 1</b>	
Ambiente:	
Sistema em operação normal	
<b>Estímulo:</b>	
Sistema recebendo integração de um fornecedor informando que um pedido mudou de status (saiu para entrega).	
<b>Mecanismo:</b>	
Criar uma API Restful exposta através do gateway para que os fornecedores possam consumir e notificar sobre alterações na mudança de status.	
<b>Medida de resposta:</b>	
Fornecedor conseguiu estabelecer a comunicação (depois de obter credenciais válidas na API de Autenticação).	
<b>Considerações sobre a arquitetura:</b>	

<b>Riscos:</b>	Além das considerações de segurança que se aplicam de maneira igual no cenário 1, há um ponto importante a se ressaltar. Forçar a utilização de tecnologias RESTFul pode limitar a quantidade de fornecedores adeptos a utilização da plataforma, baseado no quão antigos seus sistemas são e qual o peso e custo das alterações para se adequarem a esse novo modelo de comunicação.
<b>Pontos de sensibilidade:</b>	Não há.
<b>Trade-off:</b>	Não há.

### Evidências do cenário 5:



The screenshot shows the SQL Server Management Studio interface with two queries and their results.

```

SQLQuery2.sql - ya...pping (yagohf (72))* × SQLQuery1.sql - ya...pping (yagohf (71))*
1 | SELECT *
2 |   FROM PedidoFornecedor
3 | WHERE ChavePedidoFornecedor = 'PUCDROPPED001'
4 |
5 | SELECT PEV.*
6 |   FROM PedidoFornecedorEvento PEV INNER JOIN PedidoFornecedor PF
7 |     ON PF.Id = PEV.IdPedidoFornecedor
8 | WHERE PF.ChavePedidoFornecedor = 'PUCDROPPED001'

```

Results of the first query (SQLQuery1.sql):

Id	IdFornecedor	IdPedidoItem	ChavePedidoFornecedor	IdPedidoFornecedorStatus
1	2	2	PUCDROPPED001	4

Results of the second query (SQLQuery2.sql):

Id	IdPedidoFornecedor	IdPedidoFornecedorStatus	DataOcorancia	InformacoesAdicionais
1	7	0	2018-09-29 16:45:23.257	Pedido recebido pelo SIS.SAPATOP v1.1.B
2	8	2	2018-09-23 16:45:23.257	Pedido confirmado pelo SIS.SAPATOP v1.1.B
3	9	3	2018-09-24 16:45:23.257	Pedido embalado. Preparando expedição.
4	10	4	2018-09-25 16:45:23.257	Pedido expedido para transportadora LOGBRASIL C...

**Etapa 1: Evidência de que o pedido encontra-se na base de dados no status 4 (expedido). O último registro de evento do pedido é também o de status 4.**

The screenshot shows a POST request to <https://xe4e23klt10.execute-api.us-east-1.amazonaws.com/aut-dev/api/v1/usuarios/autenticar>. The request body is JSON:

```

1 [
2   "Usuario": "integracoes.sapatop",
3   "Senha": "Puc123@teste"
4 ]

```

The response status is 200 OK with a response time of 1664 ms. The response body is a large JSON object containing a token:

```

1 {
2   "status": 1,
3   "token": "eyJraWQiOjIySwgybTFBOSs0OG85UDFKYkt0aN09LMFbtbVdySmh4S29DTFLQjFhM1BzPSIsImFsZyI6I1JTMju2In0...  
[redacted]
4 }

```

**Etapa 2: Utilizando o POSTMAN (extensão para Google Chrome que simula chamadas à API's Restful), enviamos as credenciais do fornecedor para a API de autenticação através do gateway e obtemos um JWT.**

The screenshot shows a POST request to <https://wngtbtbhofq5.execute-api.us-east-1.amazonaws.com/int-dev/api/v1/pedidos/registrarevento>. The Headers tab shows the Authorization header set to Bearer followed by a long token.

Key	Value	Description	...	Bulk Edit	Presets
Content-Type	application/json				
Authorization	Bearer [long token]				

**Etapa 3: Configuramos uma nova requisição, dessa vez para o endpoint de atualização de status de entrega (registro de eventos). No cabeçalho da requisição, incluímos o token gerado na etapa 2.**

The screenshot shows the Postman interface. At the top, there are tabs for 'Autenticar fornecedor' and 'Registrar evento de pedido'. The main area shows a 'POST' request to 'https://wnngtbhofq5.execute-api.us-east-1.amazonaws.com/int-dev/api/v1/pedidos/registraevento'. The 'Body' tab is selected, showing a JSON payload:

```

1 {
2   "ChavePedidoFornecedor": "PUCDROPPED001",
3   "Status": "5",
4   "InformacoesAdicionais": "Teste de informações adicionais"
5 }

```

The response status is '200 OK' and the time taken is '1195 ms'. The response body is:

```

1 {
2   "idEventoPedidoRegistrado": 17,
3   "chavePedidoFornecedor": "PUCDROPPED001"
4 }

```

**Etapa 4: Enviamos a requisição para o servidor (adicionando os parâmetros ao corpo da mensagem para identificar o pedido que estamos atualizando e seu novo status). O servidor responde com status 200 (OK) e devolve um objeto JSON contendo o ID do evento gerado na base do nosso sistema.**

The screenshot shows a SQL Server Management Studio window with two queries. The top query is:

```

1 SELECT *
2   FROM PedidoFornecedor
3  WHERE ChavePedidoFornecedor = 'PUCDROPPED001'
4
5 SELECT PEV.*
6   FROM PedidoFornecedorEvento PEV INNER JOIN PedidoFornecedor PF
7     ON PF.Id = PEV.IdPedidoFornecedor
8  WHERE PF.ChavePedidoFornecedor = 'PUCDROPPED001'

```

The results grid shows a single row with the following data:

Id	IdFornecedor	IdPedidoItem	ChavePedidoFornecedor	IdPedidoFornecedorStatus
1	2	2	PUCDROPPED001	5

The bottom part of the screenshot shows a larger results grid with multiple rows, where the last row is highlighted with a red box:

Id	IdPedidoFornecedor	IdPedidoFornecedorStatus	DataOcorrencia	InformacoesAdicionais
1	2	0	2018-09-29 16:45:23.257	Pedido recebido pelo SIS.SAPATOP v1.1.B
2	2	2	2018-09-23 16:45:23.257	Pedido confirmado pelo SIS.SAPATOP v1.1.B
3	2	3	2018-09-24 16:45:23.257	Pedido embalado. Preparando expedição.
4	2	4	2018-09-25 16:45:23.257	Pedido expedido para transportadora LOGBRASIL. C...
5	2	5	2018-10-03 19:46:07.733	Teste de informações adicionais

**Etapa 5: Realizamos novamente a consulta ao banco de dados e podemos verificar que o status do pedido foi modificado para o que foi informado no corpo da requisição da etapa 4.**

Um novo registro foi criado na tabela de eventos, e seu ID é o mesmo retornado pela API na etapa 4.

The screenshot shows a SQL Server Management Studio window with two tabs: 'SQLQuery2.sql - ya...pping (yago...f (72))' and 'SQLQuery1.sql - ya...pping (yago...f (71))'. The 'SQLQuery1.sql' tab is active and contains the following SQL code:

```
1 | SELECT p.*  
2 | FROM Pessoa P INNER JOIN Pedido PP ON PP.IdCliente = P.Id  
3 | INNER JOIN PedidoItem PIT ON PIT.IdPedido = PP.Id  
4 | INNER JOIN PedidoFornecedor PF ON PF.IdPedidoItem = PIT.Id  
5 | WHERE PF.ChavePedidoFornecedor = 'PUCDROPPED001'
```

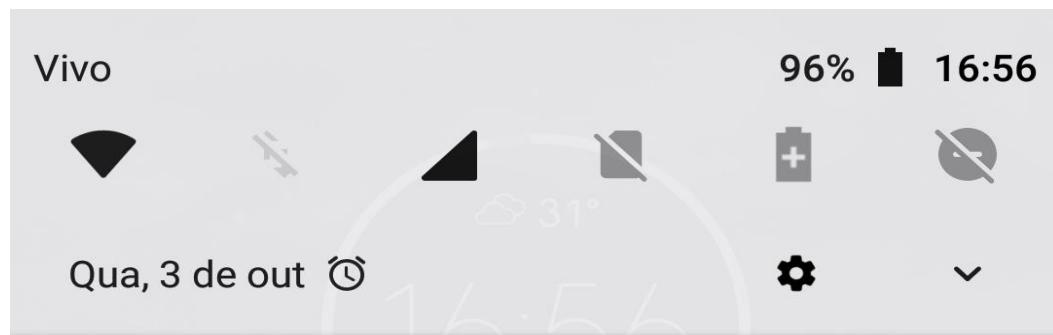
Below the code, the results of the query are displayed in a table:

	Id	IdUsuario	Nome	Email	Telefone	Documento
1	3	3	Yago Henrique Ferreira	yagoferreira21@gmail.com	5516997304540	57544205010

Etapa 6: Identificamos o cliente relacionado ao pedido atualizado.



Etapa 7: Verificamos que o cliente recebeu em seu e-mail instantaneamente a informação de que seu pedido mudou de status.



Mensagens • 9 min ^

27199

PUC Loja Informa - Pedido 00000002:  
Pedido esta sendo transportado pelo  
fornecedor



**MARCAR COMO LIDA**    **RESPONDER**

Telefone

1 Correio de voz

Discar \*555

Google • 37° em Boa Esperança do Sul • 5 h ▾

Moto Configurar Google Contatos Play Store LIMPAR TUDO



VIVO SP 16



**Etapa 8: Verificamos que o cliente recebeu em seu celular via SMS a atualização do status de seu pedido.**

## 6.5. Resultados

Dados os atributos de qualidade e realizada a validação arquitetural, nota-se que as necessidades propostas foram atendidas, porém há margem para melhora. A validação permitiu executar cenários para definir os pontos fortes e fracos da proposta. Os seguintes requisitos de qualidade foram validados:

<b>Requisitos não funcionais</b>	<b>Testado</b>	<b>Homologado</b>
Segurança – o sistema deve apresentar altos padrões de segurança	Sim	Sim
Usabilidade – o sistema deve prover boa usabilidade	Sim	Sim
Acessibilidade - o sistema deve ser suportar ambientes web responsivos e ambientes móveis	Sim	Sim
Interoperabilidade - o sistema deve se comunicar com vários sistemas, com tecnologias heterogêneas	Sim	Sim

Do ponto de vista da construção de código, todos os componentes foram divididos em camadas. O front-end em Angular divide-se naturalmente em um padrão de componentes e serviços do próprio framework que favorece o isolamento e a testabilidade. A utilização do Bootstrap como framework CSS agiliza o desenvolvimento de interfaces amigáveis responsivas, o que poupa bastante tempo em prototipações e interfaces mais simples. Porém, para recursos visuais mais complexos, dá-se a necessidade de uma equipe especializada em UX/UI para atingir melhores resultados.

As API's em .NET Core são divididas em camadas lógicas, separando o que é exposição de dados, regras de negócio e acesso a repositórios de dados (sejam bancos de dados ou serviços de terceiros). Todo o código gerado favorece a testabilidade e a componentização utilizando-se de princípios de programação baseados em SOLID. Cada camada do código pode ser substituída por outra sem afetar seus consumidores, desde que o contrato seja mantido. É fácil também realizar mocks, favorecendo a testabilidade. Em casos que a velocidade no desenvolvimento e a facilidade de mudança é necessária, o ORM EF Core supre bem as ne-

cessidades. Porém, no que tange a performance, há de se preferir a utilização de ADO.NET para acesso a banco. Casos de uso críticos devem se valer desses artifícios para entregarem o resultado esperado, como na integração com os fornecedores, em que o tempo é recurso precioso.

Do ponto de vista das integrações, as comunicações realizadas com sistemas terceiros são bem sucedidas, mesmo com a variação de tecnologias (REST e SOAP). Porém, a comunicação é muito dependente da implementação realizada em .NET. Há uma margem para melhora incluindo um ESB – Enterprise Service Bus - para gerir essas comunicações, removendo da camada .NET a responsabilidade por gerir essa miscelânea de tecnologias de comunicação.

No que diz respeito a utilização de componentes prontos, fornecidos pela AWS, existem também os prós e os contras. A principal vantagem é utilizar componentes amplamente testados e validados, e no caso da implantação na nuvem da própria empresa há um ganho na facilidade da realização de integrações com outros serviços. Há também um ganho no tempo de desenvolvimento por se utilizar soluções integráveis de maneira simples. Porém há também a desvantagem de estar preso a um fornecedor, visto que as chamadas às API's de notificação por e-mail e SMS são feitas por um SDK proprietário da Amazon. A mudança para outro produto implicaria na reescrita de uma parte do código (impacto reduzido pelo isolamento de componentes e pelo desenvolvimento guiado a interfaces, com injeção de dependência).

Do ponto de vista da implantação, utilizar a solução em nuvem da Amazon para prover os serviços traz uma facilidade de configuração, escalabilidade e integração entre componentes que é bastante interessante. A maioria das configurações podem ser feitas por interface gráfica, sendo bem intuitivas e com uma documentação bem rica, provendo exemplos para várias linguagens de programação diferentes. Além disso, não há a necessidade de contar com uma infraestrutura robusta, adquirir servidores, links de internet, firewalls e todas as equipes para cuidar desse tipo de coisa. O contra nesse caso é o custo. Embora o mantra de provedores de serviços em nuvem seja “pague pelo que usar”, as tarifas não chegam a ser das mais atrativas para alguns serviços. É o preço a se pagar pela comodidade, confiabilidade e SLA's bem robustos. Para uma empresa que não tem restrições financeiras, é uma excelente proposta. Há ainda uma margem para evolução com a utilização de contêineres para implantação das API's.

Provisionar um ambiente de DevOps nessa arquitetura é bem interessante, pois agiliza e facilita o processo de validação de novas implementações, auxiliando no cumprimento dos requisitos não funcionais de testabilidade e manutenibilidade.

## 7. Conclusão

Este trabalho apresentou um projeto arquitetural para uma plataforma de loja virtual baseada em dropshipping. Foram exploradas diversas tecnologias, algumas extremamente atuais (como o modelo de programação serverless), tecnologias emergentes (como .NET Core e Angular, em constante evolução), porém foi mantido o suporte às comunicações com sistemas legados (COBOL). Foram apresentadas sugestões de implantações em nuvem, tema cada vez mais presente no cotidiano e tendência global, porém não há qualquer dependência, podendo implantar os componentes propostos em ambientes on-premises (com infraestrutura própria). Por utilizar-se de vários padrões emergentes e de tecnologias muito recentes, há sempre a dificuldade de mudança de mentalidade e quebra de paradigmas, que são necessários para a evolução.

Conclui-se assim que os objetivos foram atingidos, embora haja margem para melhora, que pode ocorrer em uma próxima versão.

## REFERÊNCIAS

AWS Documentation. Amazon. Disponível em: <[https://docs.aws.amazon.com/index.html#lang/pt\\_br](https://docs.aws.amazon.com/index.html#lang/pt_br)>. Acesso em: 02 de outubro de 2018.

AWS .NET SDK. Amazon. Disponível em: <[https://docs.aws.amazon.com/pt\\_br/sdk-for-net/v3/developer-guide/aws-sdk-net-v3-dg.pdf](https://docs.aws.amazon.com/pt_br/sdk-for-net/v3/developer-guide/aws-sdk-net-v3-dg.pdf)>. Acesso em: 02 de outubro de 2018.

Guia do .NET Core. Microsoft. Disponível em: <<https://docs.microsoft.com/pt-br/dotnet/core>>. Acesso em: 02 de outubro de 2018.

Angular Docs. Google. Disponível em: <<https://angular.io/docs>>. Acesso em: 02 de outubro de 2018.

NGX-Bootstrap. Valor Software. Disponível em: <<https://valor-software.com/ngx-bootstrap/#/getting-started>>. Acesso em: 02 de outubro de 2018.

## APÊNDICES

URL da loja virtual implantada na AWS: <http://dropshippinglojafrontend.s3-website-us-east-1.amazonaws.com>

URL da API de para registro de pedidos de fornecedores implantada na AWS:  
<https://wngtbhofq5.execute-api.us-east-1.amazonaws.com/int-dev/api/v1/pedidos/registrarevento>

URL da apresentação da POC no Youtube: <https://youtu.be/grdRAZIdgdc>