

UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação em Arquitetura de Software Distribuído

Werick Silva Rodrigues Cruz

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

Brasília

2019

Werick Silva Rodrigues Cruz

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

Trabalho de Conclusão de Curso em Arquitetura de
Software Distribuído.

Orientador: Tadeu dos Reis Faria

Brasília

2019

RESUMO

O presente projeto tem como objetivo especificar os passos de criação e implementação de uma plataforma de comércio eletrônico na modalidade dropshipping, bem como apresentar soluções funcionais para a mesma. Tendo em vista que o comércio eletrônico vem ocupando gradativamente mais espaço no mercado. Faz-se necessária a busca pelo avanço das tecnologias. Nos últimos anos, o número de lojas virtuais tem crescido expressivamente, gerando assim, enérgica concorrência e notórios desafios para aqueles que buscam promover maiores experiências nos processos de compras. Aspectos como agilidade das entregas, fidelização dos clientes, inovações e comodidade, são pontos importantes para os que buscam a criação e implementação de uma plataforma de comércio eletrônico. Assim como, atender os anseios estabelecidos pelos proprietários, e, sobretudo, buscar estratégias para a diminuição dos custos e aumento dos lucros também se fazem necessárias para o processo. A modalidade dropshipping faculta ao proprietário manter estoque e responsabilidades logísticas de entrega dos produtos. Estas podem ser delegadas ao fornecedor, que estabelecerá relação efetiva com cliente. Portanto, a fim de promover as já citadas melhores experiências nos processos de compras, este projeto abordará 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, aqui considerados mais importantes.

Palavras-chave: arquitetura de software distribuído, comércio eletrônico, dropshipping, sistema, requisitos não funcionais, dispositivos móveis.

SUMÁRIO

1. Objetivos do trabalho	5
2. Descrição geral da solução	6
2.1. Apresentação do problema	6
2.2. Descrição geral do software	7
3. Definição conceitual da solução	8
3.1. Requisitos Funcionais	8
3.2 Requisitos Não-Funcionais.....	13
3.3. Restrições Arquiteturais	17
3.4. Mecanismos Arquiteturais.....	17
4. Modelagem e projeto arquitetural	18
4.1. Modelo de casos de uso	18
4.2. Descrição resumida dos casos de uso	25
4.3. Modelo de componentes.....	32
4.4. Modelo de implantação	36
4.5. Modelo de dados	39
5. Prova de conceito / protótipo arquitetural.....	40
5.1. Implementação e implantação.....	40
5.2. Interfaces/ APIs	43
6. Avaliação da Arquitetura.....	46
6.1. Análise das abordagens arquiteturais	46
6.2. Identificação dos atributos de qualidade.....	46
6.3. Cenários	46
6.4. Avaliação.....	48
6.5. Resultados	67
7. Conclusão.....	69
REFERÊNCIAS	70
APÊNDICES	71

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. Com isto, tem-se como objetivos específicos, fornecer uma plataforma na qual todos os envolvidos no processo de compra e venda tenham suas atividades facilitadas. Proporcionar alta disponibilidade, segurança e desempenho. Estabelecer a semelhante experiência de navegação da mesma plataforma, em diversos dispositivos diferentes com acesso à internet através de um browser (notebooks, desktops, tablets e smartphones).

Os objetivos específicos são:

1. Criar o módulo de loja, permitindo que os usuários no caso clientes e vendedores possam navegar por uma lista de produtos, adicionar a um carrinho e realizar, respectivamente as compras. Os usuários poderão ainda atualizar dados cadastrais, acompanhar entregas e consultar o histórico de pedidos realizados. Neste módulo, apenas a lista de produtos e o carrinho poderão ser visualizados publicamente. O restante das operações irá requerer autenticação segura.
2. Criar o módulo administrativo, responsável por permitir que seja feita a controle de promoções, propagandas, produtos, venda, fornecedores. Será possível também aos administradores acessar um conjunto de relatórios gerados a partir de informações do sistema de business intelligence, para facilitar na tomada de decisões. Este módulo será protegido por autenticação segura.
3. Criar o módulo de serviço de atendimento ao cliente (sac), que permitirá aos usuários conectar-se a um canal diretamente com a loja para obter informação e tirar 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 produzir integrações com os sistemas externos envolvidos no sistema da plataforma (fornecedores, plataforma de pagamentos e organizações governamentais para controle fiscal, etc.). Esse módulo terá todas suas funcionalidades protegidas por autenticação segura

2. Descrição geral da solução

2.1. Apresentação do problema

Nos últimos anos, o comercial tem sofrido constantes modificações em função da inclusão da Era Digital. Com isto, o avanço das tecnologias e a popularização do acesso à internet, faz com que o comércio eletrônico ocupe cada vez mais espaço no mercado de compras e vendas. Devido a estes fatores, faz-se necessária a modificação do uso de tecnologia da informação.

Atualmente, há inúmeros fatores significantes para ao sucesso das empresas estruturadas no modelo de comércio eletrônico. Estes fatores estão relacionados a compreensão de que, o tempo livre da população a cada dia se torna mais escasso, logo, os clientes prezam pela agilidade, facilidade e segurança na realização de suas compras.

Assim, a criação e implementação de uma plataforma de comércio eletrônico, além de se atentar aos anseios da população cada vez mais conectada, precisará também, atender às necessidades dos proprietários, e, sobretudo, buscar estratégias para a diminuição dos custos e aumento dos lucros.

O comércio eletrônico na modalidade dropshipping, é uma estratégia que visa manter ações competitivas em um mercado estatisticamente cada vez mais disputado. Tendo em vista que, diversas 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, a modalidade de comércio dropshipping, apresenta uma alternativa eficaz onde a loja não tem a responsabilidade de manter um estoque e nem de cuidar da logística de entrega dos produtos; estas são delegadas ao fornecedor que estabelecerá relação transparente com cliente.

O dropshipping permite que as lojas concentrem-se nos clientes e na experiência de compra, enquanto que os fornecedores não se preocuparão com detalhes de suporte às vendas, publicidade online ou com a manutenção de um sistema de comércio eletrônico.

Entretanto, comércios eletrônicos com dropshipping, apresentam dificuldades relacionadas às integrações. A comunicação com fornecedores, à disponibilidade de produtos

e dados referentes aos eventos relacionados aos pedidos e às entregas são exemplos de dificuldades encontradas nesta modalidade. Logo, estratégias que proporcionem que os clientes sejam notificados a cada evento ocorrido em tempo real são importantes no processo de elaboração de um dropshipping. O fácil acesso a plataformas e meios de pagamentos diminuem a complexidade do trato dessas questões na plataforma.

Por fim, o cuidado em estabelecer integrações com entidades governamentais para controles fiscais, como a Secretaria da Fazenda também são pontos importantes no processo de criação. Por consequência, gerir e sincronizar todos os aspectos e estratégias até então apresentado é extremamente complexo e trabalhoso.

2.2. Descrição geral do software

A construção desse software tem por objetivo oferecer uma plataforma integrada de comércio eletrônico, na qual todos os envolvidos no processo de compra e 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.

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

A plataforma terá que possuir alta disponibilidade, segurança, desempenho 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

- **Lista produtos**

O sistema deve permitir que os usuários naveguem por uma lista 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 a descrição e as avaliações de quem comprou aquele produto.

- **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;.

- **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 visualizar suas compras.

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 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.

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.

- **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 na lista de produtos 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 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.

- **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 inativos deverão sair do lista até estarem novamente ativos.

3.2 Requisitos Não-Funcionais

- Usabilidade – o sistema deve prover boa usabilidade

Estímulo	Usuário buscando produtos por nome
Fonte do estímulo	Usuário acessando a funcionalidade de lista de produtos, incluindo itens em seu carrinho de compras.
Ambiente	Funcionamento, carga normal
Artefato	Módulo Loja
Resposta	A camada de apresentação apresenta facilidade de navegação, facilitada, simplicidade e objetividade.
Medida da resposta	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

Estímulo	Usuário buscando produtos por nome
Fonte do estímulo	Usuário acessando a funcionalidade de lista de produtos, incluindo itens em seu carrinho de compras, em um smartphone ou tablet.
Ambiente	Funcionamento, carga normal
Artefato	Módulo Loja
Resposta	A camada de apresentação se adaptou à resolução e tamanho da tela, mudando os componentes de tamanho e posição para facilitar a navegação.
Medida da resposta	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.

Estímulo	Acessar uma página privada pela URL sem estar autenticado no sistema
Fonte do estímulo	Usuário não autenticado
Ambiente	Funcionamento, carga normal
Artefato	Módulo Loja, Módulo Serviço de Atendimento ao

	Cliente, Módulo Administrativo
Resposta	O sistema redirecionou o usuário para a respectiva tela de autenticação.
Medida da resposta	O sistema não permite acesso à página solicitada, e obriga a autenticação.

- o Acesso a integração sem estar autenticado

Estímulo	O sistema do fornecedor envia uma requisição de atualização de status de uma entrega sem estar autenticado no sistema
Fonte do estímulo	Usuário não autenticado
Ambiente	Funcionamento, carga normal
Artefato	Módulo Integrações
Resposta	O sistema faz o envio da atualização de status.
Medida da resposta	Sistema permite atualizar o status.

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

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

- Desempenho – o sistema deve ser rápido

Estímulo	Usuário navegando pela lista de produtos
-----------------	--

Fonte do estímulo	Usuário buscando produtos de uma determinada categoria ou nome do produto.
Ambiente	Funcionamento, carga normal
Artefato	Módulo Loja
Resposta	O sistema respondeu com os dados solicitados.
Medida da resposta	O sistema respondeu em menos de 5 segundos.

- Testabilidade – o sistema deve ser simples para testar

Estímulo	Execução de testes no sistema
Fonte do estímulo	Analista desenvolvedor
Ambiente	Ambiente de desenvolvimento
Artefato	Módulo Loja, Módulo Serviço de Atendimento ao Cliente, Módulo Administrativo, Módulo Integrações
Resposta	O sistema testou todas as funcionalidades disponíveis.
Medida da resposta	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

Estímulo	Alteração no SGBD utilizado
Fonte do estímulo	Compra de licença de outro SGBD com maior quantidade de recursos e suporte do fornecedor
Ambiente	Aproximação dos limites de utilização do banco de dados na versão contratada
Artefato	Módulo Loja, Módulo Serviço de Atendimento ao Cliente, Módulo Administrativo, Módulo Integrações
Resposta	As modificações no código são realizadas isoladamente nos componentes da camada de acesso a dados das aplicações.

Medida da resposta	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. Java , Spring Boot (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 Html e Bootstrap
Back-end	Regras de negócio da aplicação	Java e Spring Boot
Integrações 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	Hibernate
Persistência	Banco de dados relacional e plataforma de Big Data	AWS RDS (Relational Database Service) e Microsoft SQL Server
Serviços agendados	Execução de processamentos agendados	AWS Cloudwatch e AWS Lambda

Alta disponibilidade	Balanceamento de carga das aplicações	AWS ElasticBeanstalk
Autenticação e autorização	Verificação das credenciais para execução de ações	OAuth ,AWS Cognito
Exposição de API's	Exposição de Restful API's	AWS API Gateway
Notificações de usuários	Envio de notificações aos usuários por e-mail	Spring boot starter mail, AWS SNS – Simple Notification 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	Maven e Angular-Cli
Automação de testes	Execução de testes automatizados das aplicações	Junit 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.

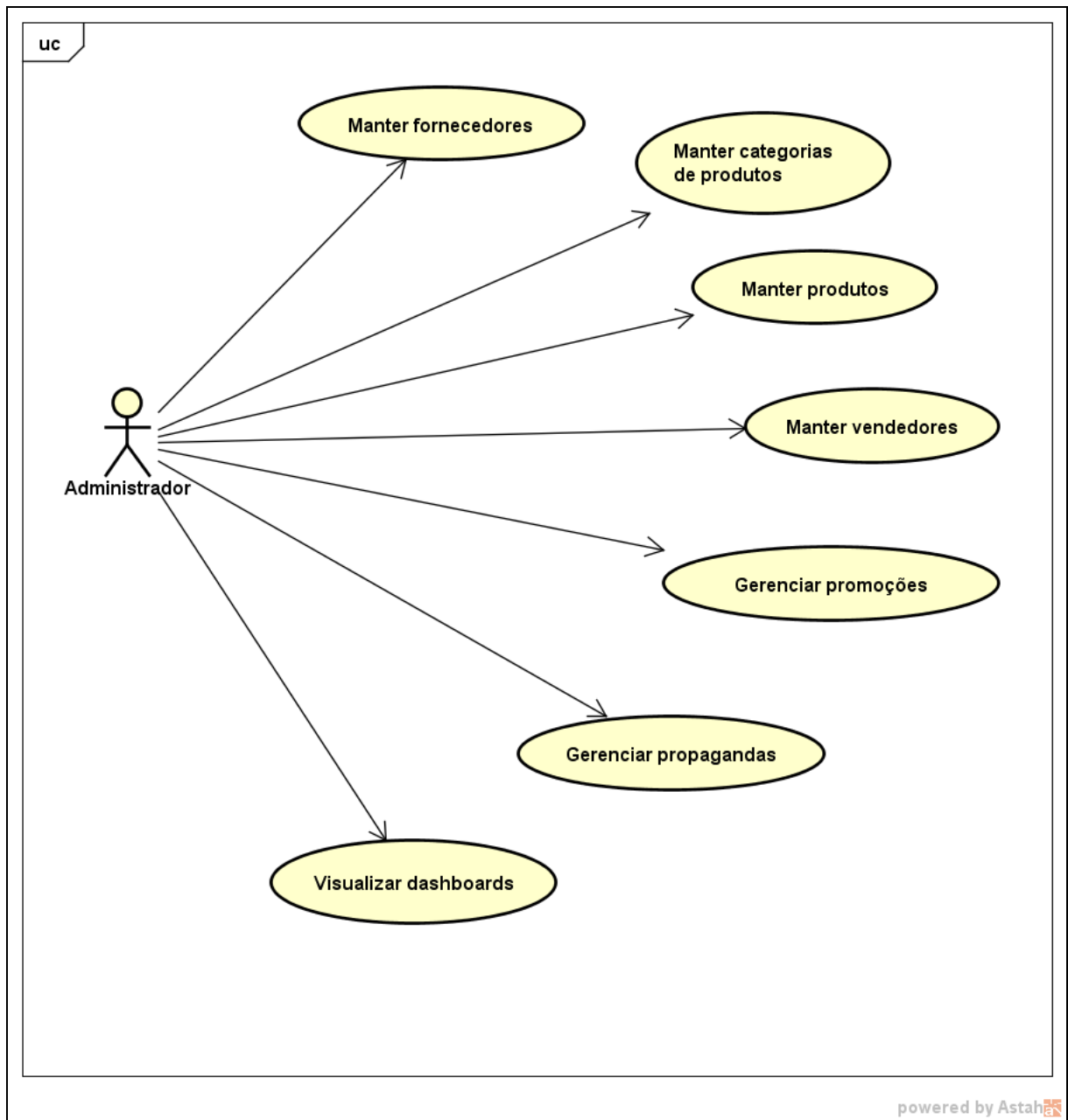


Figura 1 – Diagrama de Casos de Uso – Módulo Administrativo

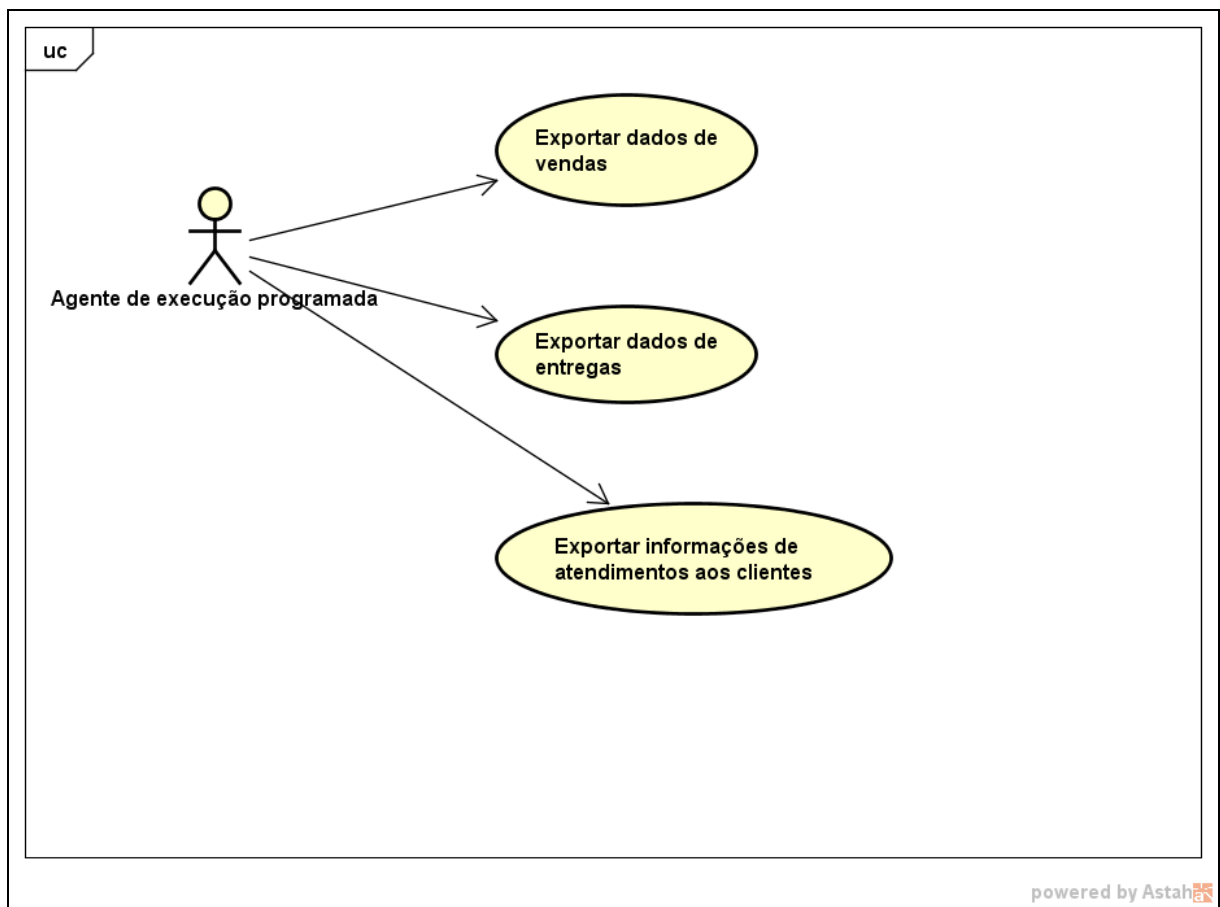


Figura 2 – Diagrama de Casos de Uso – Módulo Geração de Informações Gerenciais

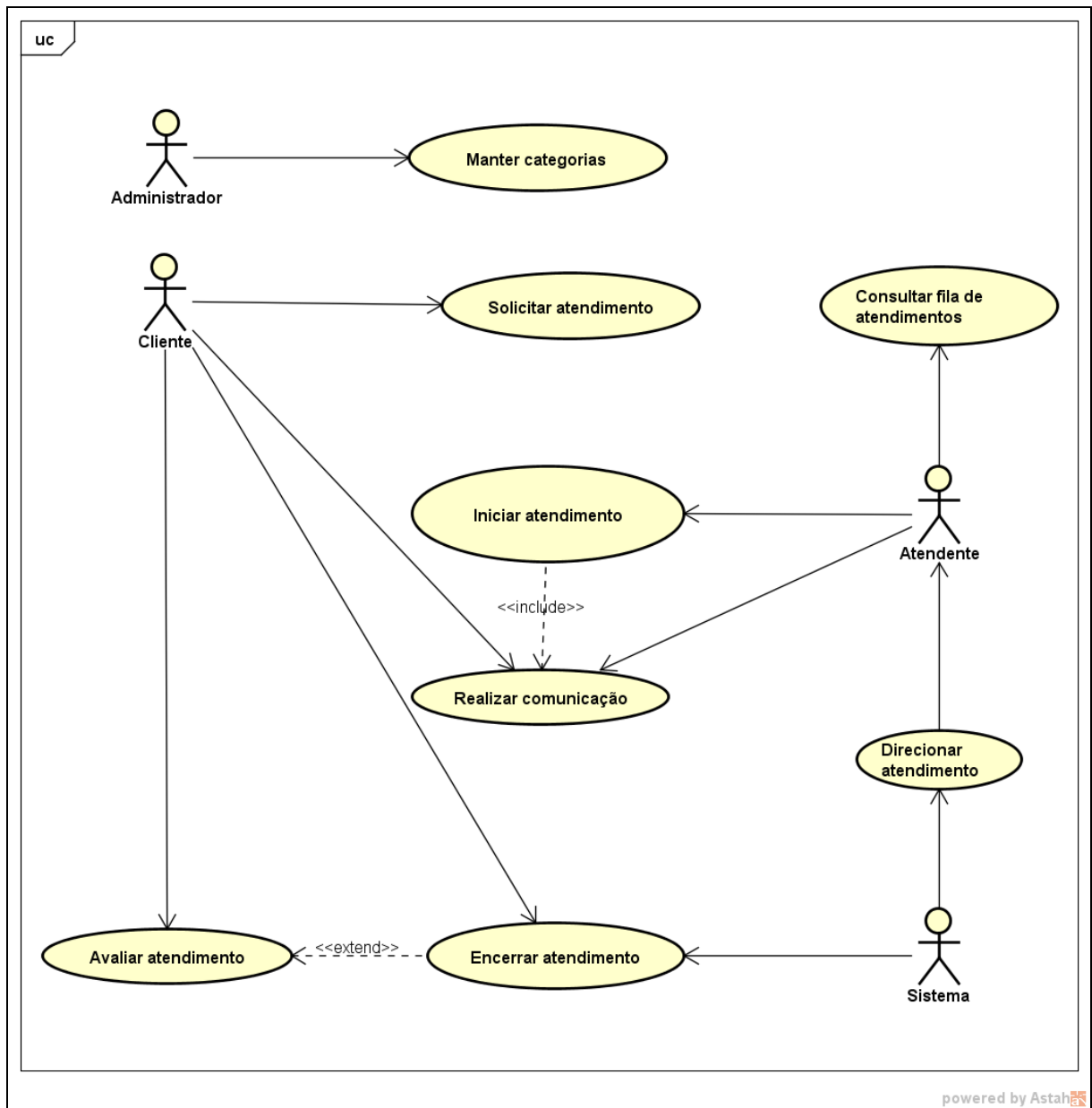


Figura 3 – Diagrama de Casos de Uso – Módulo Serviço de Atendimento ao Cliente

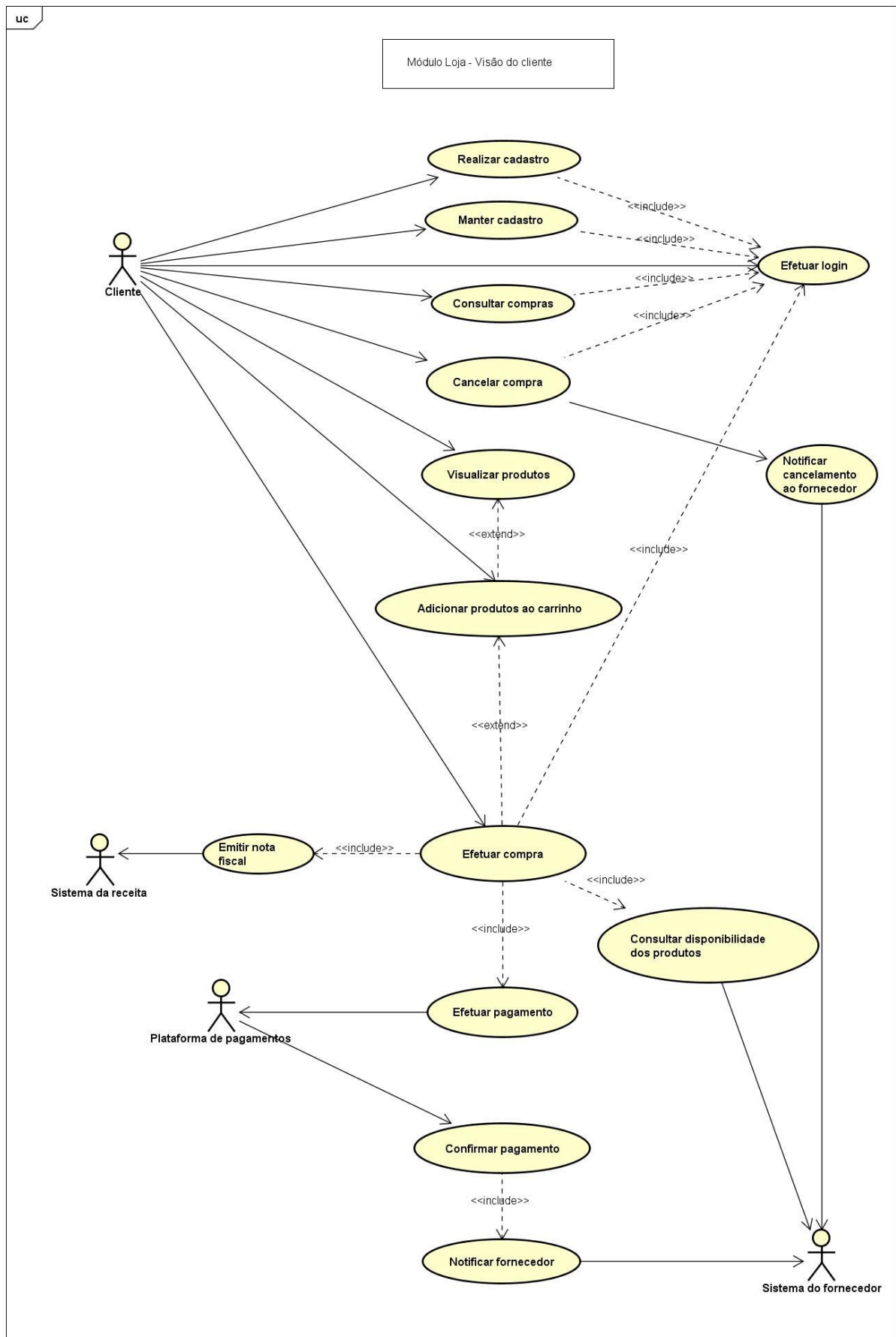


Figura 4 – Diagrama de Casos de Uso – Módulo Loja (visão do cliente)

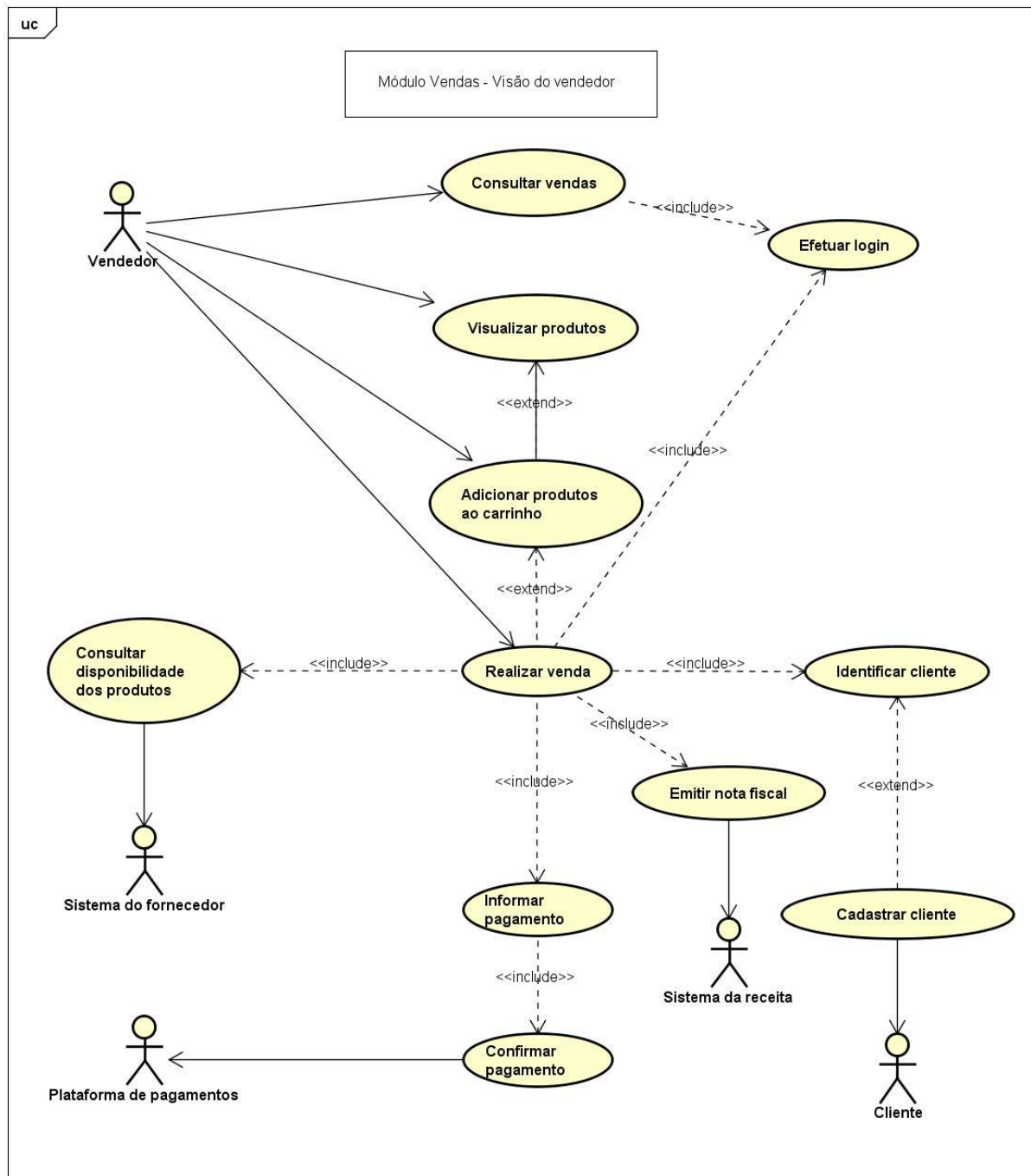


Figura 5 – Diagrama de Casos de Uso – Módulo Loja (visão do vendedor)

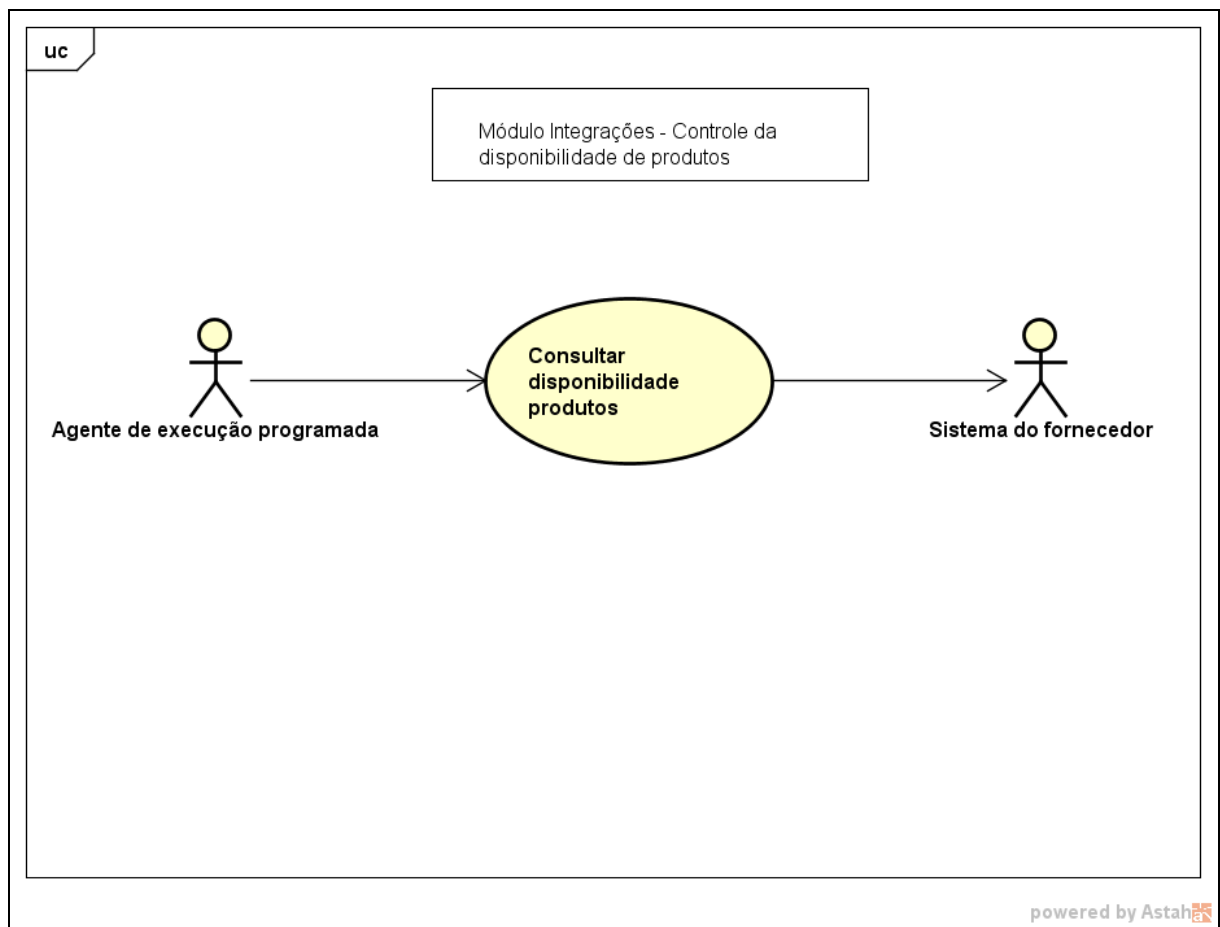


Figura 5 – Diagrama de Casos de Uso – Módulo Integrações (consultar disponibilidade de produtos no fornecedor)

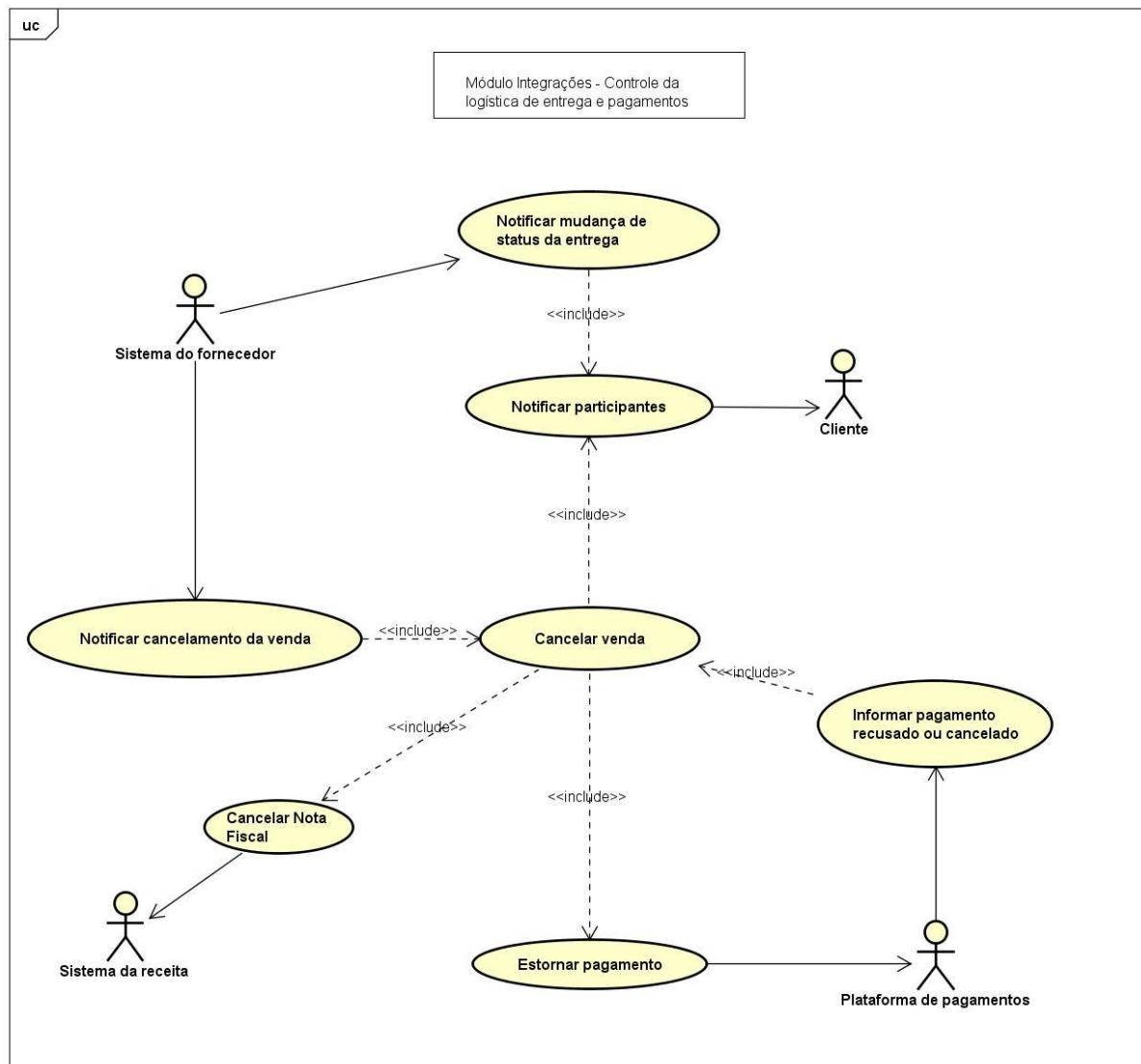


Figura 6 – Diagrama de Casos de Uso – Módulo Integrações (controle da logística de entrega e de pagamentos)

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 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 e fornecedor. Também será permitido ao usuário inativar um produto, impedindo assim que este apareça na 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 e compras efetuadas.

Caso de uso: Manter cadastro

Descrição: Este caso de uso permitirá que usuários autenticados possam visualizar e atualizar seus dados cadastrais, como e-mail, senhas, endereços e documento etc.

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 produtos

Descrição: Este caso de uso permitirá que usuários anônimos ou autenticados possam navegar por uma lista 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.

- **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 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

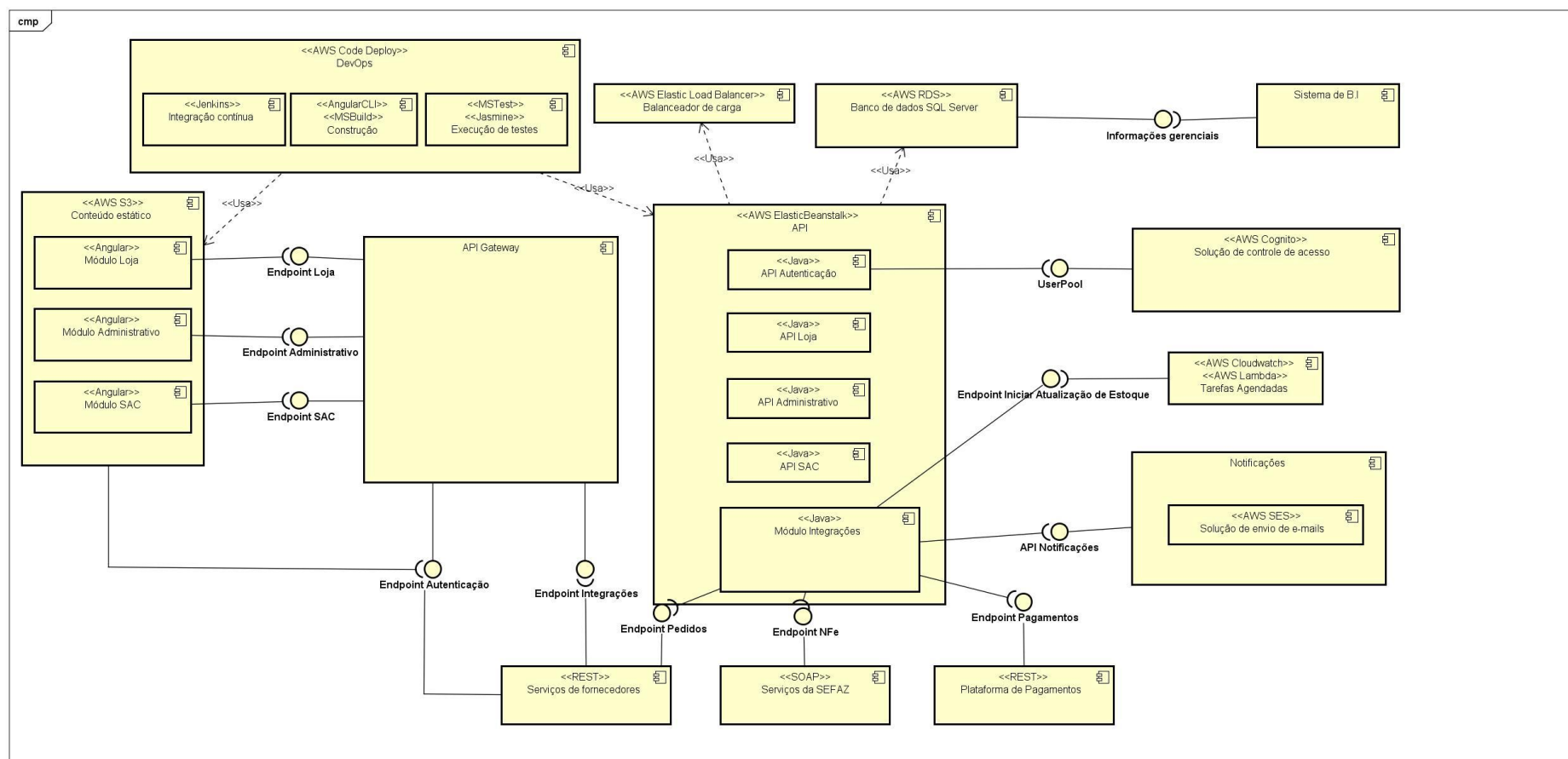
Descrição: Este caso de uso permitirá que o sistema envie requisições de cancelamento dos pedidos aos fornecedores através de integração, após solicitação dos usuários ou informação de pagamento recusado por parte da plataforma de pagamentos.

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). 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, Html 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 oauth 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 receita 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.

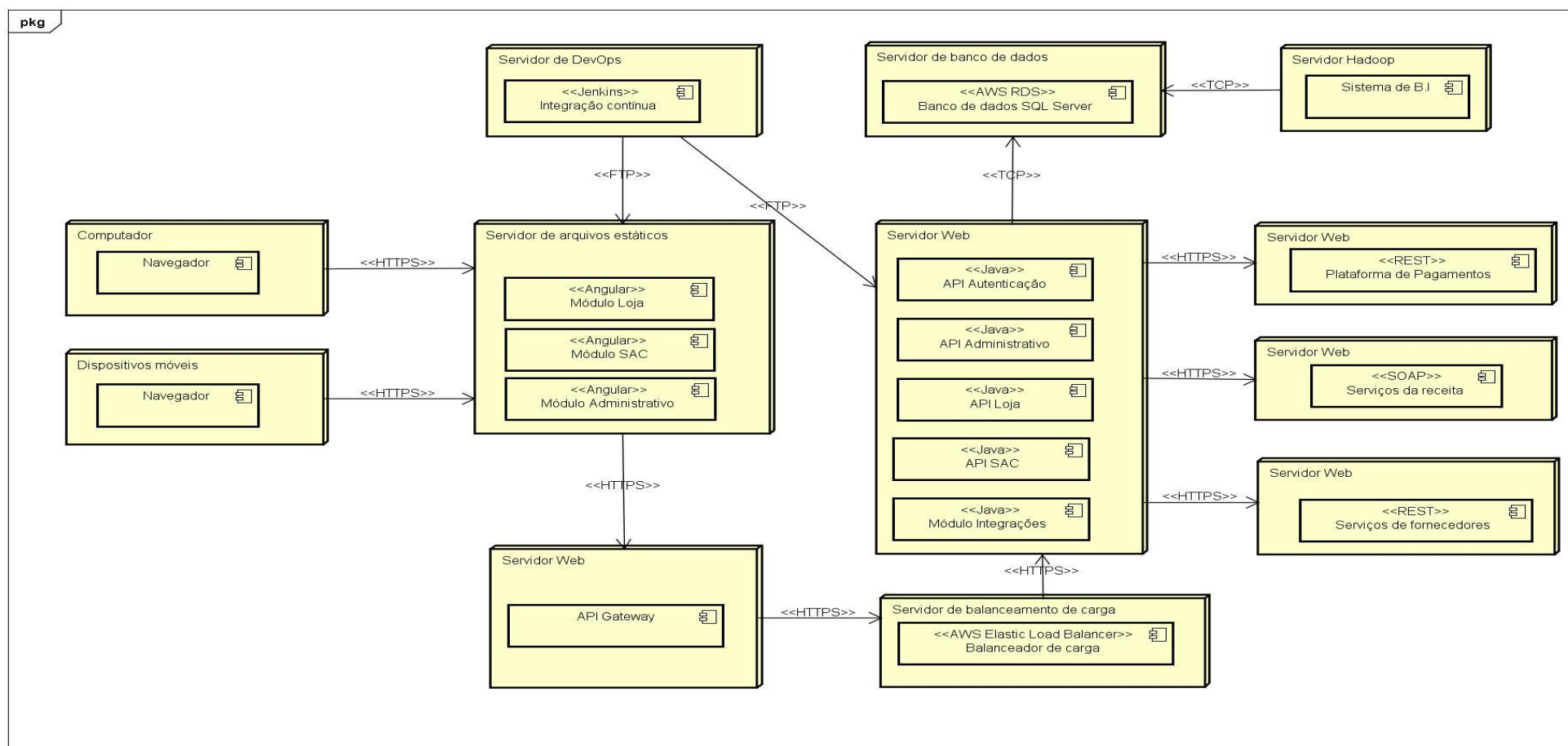
Há ainda a integração em sentido oposto entre loja e fornecedores, na qual a loja expõe através do API Gateway um endpoint de atualização de status dos pedidos. Os fornecedores precisarão seguir um fluxo que consiste em autenticar-se na API de Autenticação, obtendo um oauth e depois chamar a API de Integração, passando esse oauth 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.



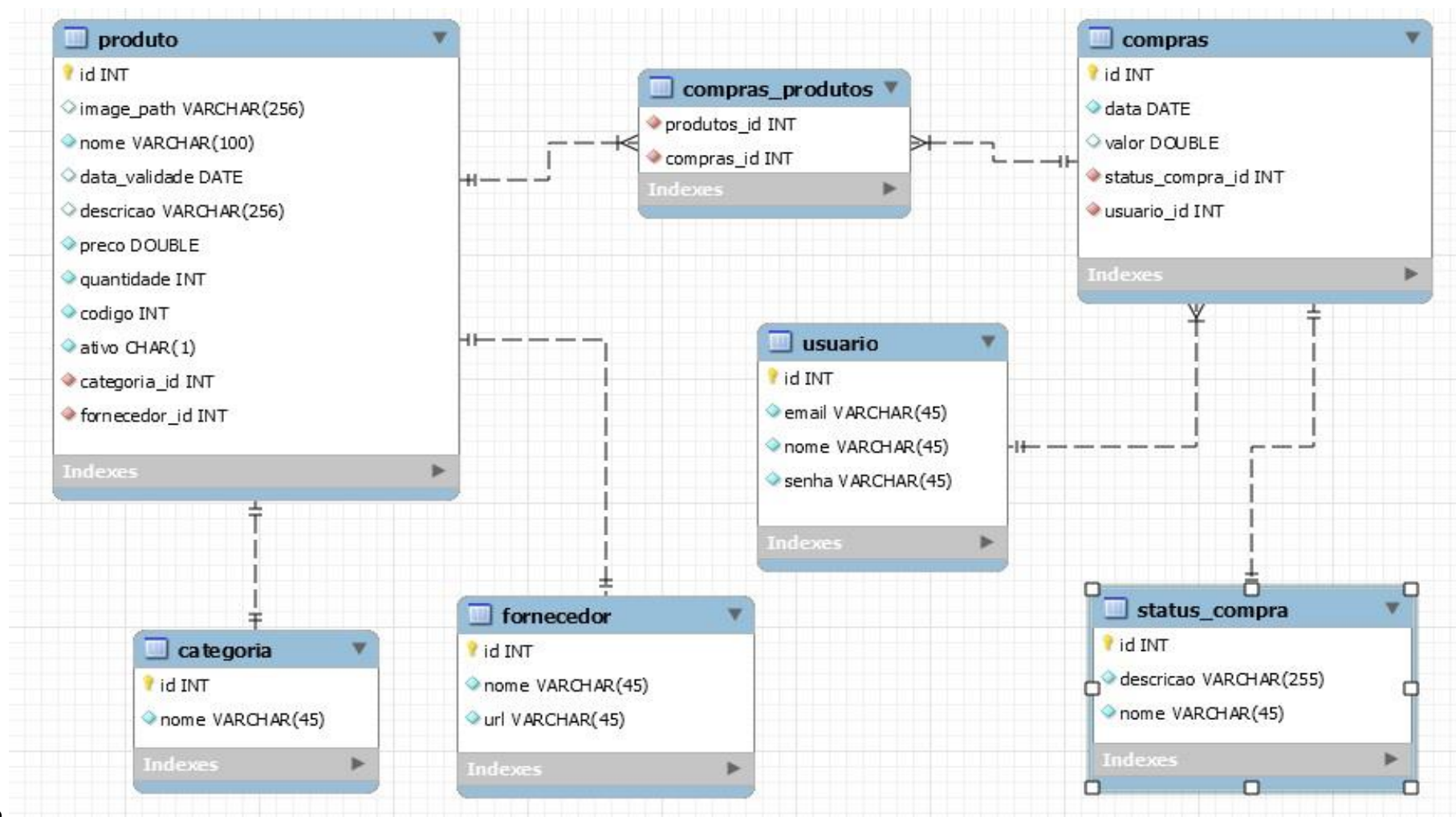
Os módulos do sistema poderiam ser implantados em infraestruturas on-premises também.

Componente	Descrição
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 Java. A infraestrutura utilizada na POC é composta de servidores local, rodando em máquinas com S.O Windows.</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 balancear as requisições que chegam entre nós do cluster de servidores Web, diminuindo a latência. 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 Docker Swarm.
Servidor de banco de dados	Servidor Windows que possa fornecer instância de banco de dados SQL Server. Na POC foi considerada a solução local.

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.
- 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, fácil 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 deve apresentar objetividade e não serem confusos.

- **Acessibilidade**

Esse requisito não funcional foi escolhido devido à 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 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	Consultar compras	Segurança, Usabilidade e Acessibilidade.
	Visualizar produtos	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
Consultar compras	Api rest, postman ,Sql Server, java spring boot
Visualizar produtos	Angular, Html e Bootstrap
Adicionar produtos ao carrinho	Angular, Html e Bootstrap
Consultar disponibilidade de produtos	Java ,Api Rest, spring boot, Sql Server e Postman
Notificar mudança de status da entrega	Java ,Api Rest, spring boot, Sql Server e Postman

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)	Node js
API Loja	Apache
API Autenticação	spring boot security/ spring security oauth2

API Integrações	Api restful, spring boot
Consulta de estoque	Api restful, spring boot
Banco de dados SQL Server	Sql server localhost

Para acesso a plataforma os passo 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 em API's Restful. A API tem por objetivo simular uma consulta à disponibilidade de um produto na base do fornecedor. Essa API foi desenvolvida em .java. 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 enve. Dado registrado na tabela Fornecedor da base de dados.
Senha	Texto	Sim	Senha do usuário de serviço. Deve ser enviado no header . Dado registrado na tabela Fornecedor da base de dados.
ChaveProduto	Texto	Sim	Chave do produto na base de dados do fornecedor. Dado registrado na tabela Produto da base de dados.

Saída		
Parâmetro	Tipo	Descrição
Status	Texto	200 indica que o produto está disponível.

Dados para teste	
Parâmetro	Tipo

URL	http://localhost:8080/produtos
Método	produtos
Usuário	werick
Senha	123
ChaveProduto	1

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	1. 200 Indica que a autenticação foi bem-sucedida. 2. 401 Indica que a autenticação falhou por falha nas credenciais. 3. 400 Indica que a autenticação falhou por um erro na API.
Token	Texto	oauth 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	http://localhost:8080/oauth/token
Método	POST

Login	werick
Senha	123

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

Entrada			
Parâmetro	Tipo	Obrigatoriedade	Descrição
Chave da compra	Texto	Sim	Chave da compra que se deseja informar uma mudança de status.
Status	Numérico	Sim	Status atualizado do pedido: <ul style="list-style-type: none"> 1 Pedido recebido 2 Pedido recusado 3 Pedido expedido 4 Pedido em transporte 5 Pedido entregue
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”.

Saída		
Parâmetro	Tipo	Descrição
Status compra	texto	Identificador do evento registrado, caso o fornecedor queira armazenar para fins de auditoria.

Dados para teste	
Parâmetro	Tipo
URL	http://localhost:8080/compras/
Método	PUT

Token	Obter via API de autenticação
idcompra	1
Status	5

6. Avaliação da Arquitetura

6.1. Análise das abordagens arquiteturais

A arquitetura proposta considera uma série de componentes, todos modulares. Cada componente tem seu próprio grupo de tecnologias e características de implantação. Apesar do uso de vários componentes proprietários e de uma infraestrutura em nuvem, toda a implementação foi feita de forma que seja o mais livre 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 terceiros. 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	A	A

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

6.4. Avaliação

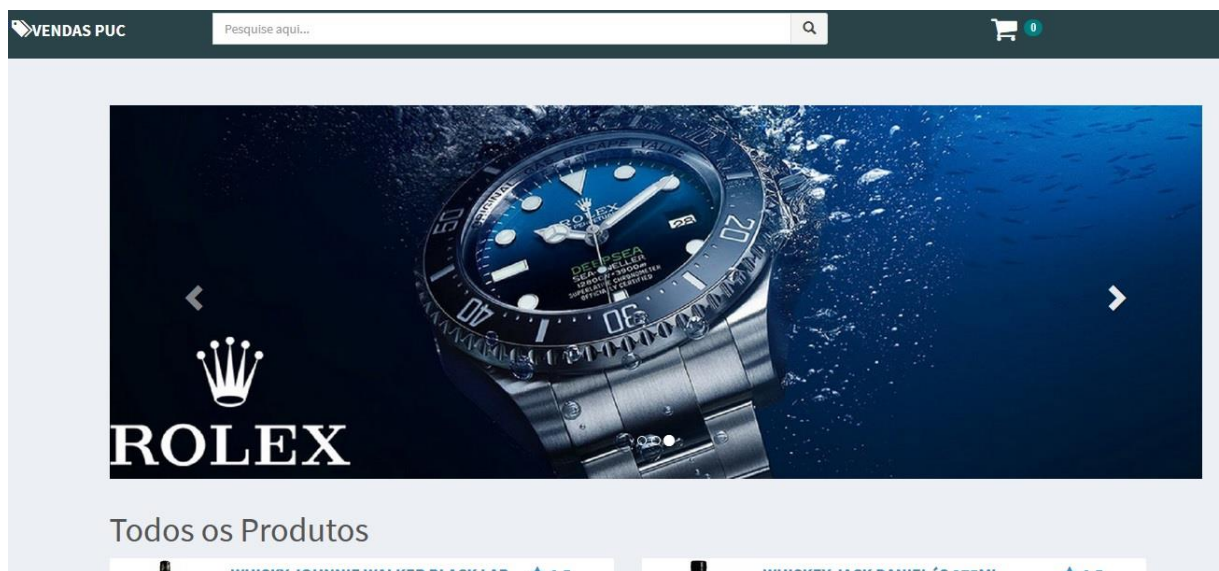
Processo de avaliação dos cenários identificados no item [6.3](#). O objetivo aqui é determinar os riscos, não riscos, pontos de sensibilidade, trade-off's e evidenciar o atendimento aos requisitos de qualidade.

- Cenário 1**

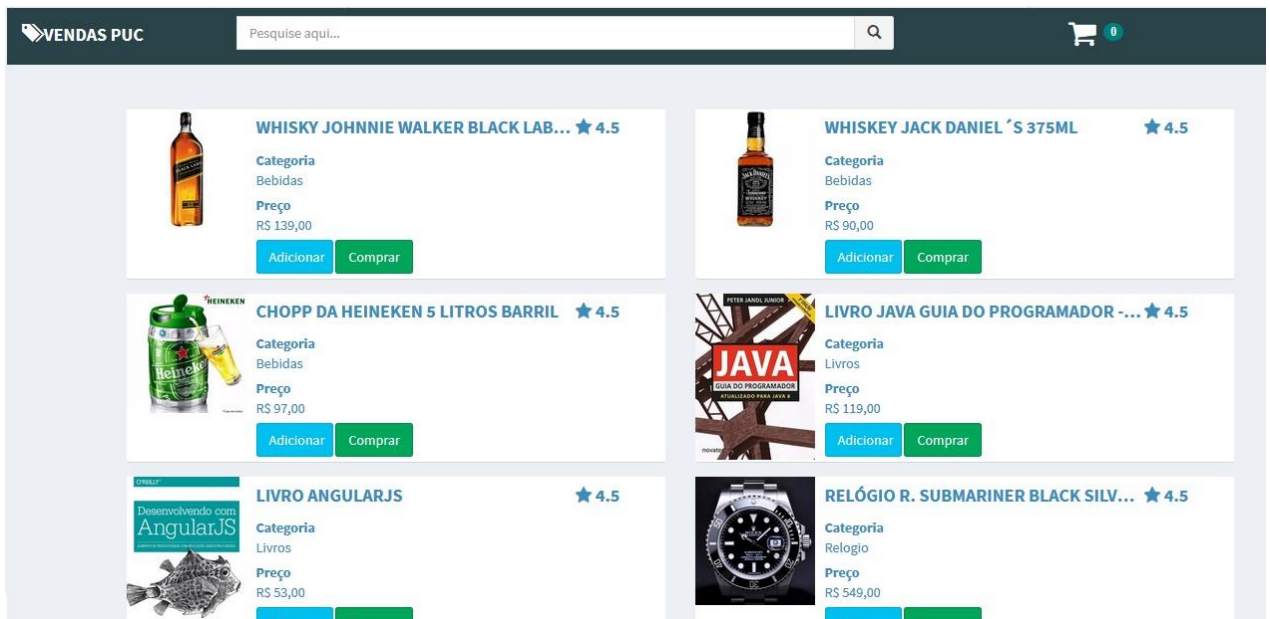
Atributo de qualidade:	Segurança
Requisito de qualidade:	O sistema deve apresentar altos padrões de segurança.
Preocupação	
Impossibilitar o acesso a páginas privadas do sistema sem autenticação no sistema.	
Cenário 1	
Ambiente:	
Sistema em operação normal	
Estímulo:	

Usuário tentando acessar uma página privada sem estar autenticado no sistema.	
Mecanismo:	
Criar um mecanismo de validação de credenciais (tokens, no caso) e suas permissões associadas para acessar recursos protegidos.	
Medida de resposta:	
O usuário deve ser receber mensagem de erro não autorizado	
Considerações sobre a arquitetura:	
Riscos:	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 oauth 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.
Pontos de sensibilidade:	Servidor de aplicação operando em modo HTTPS.
Trade-off:	Não há.

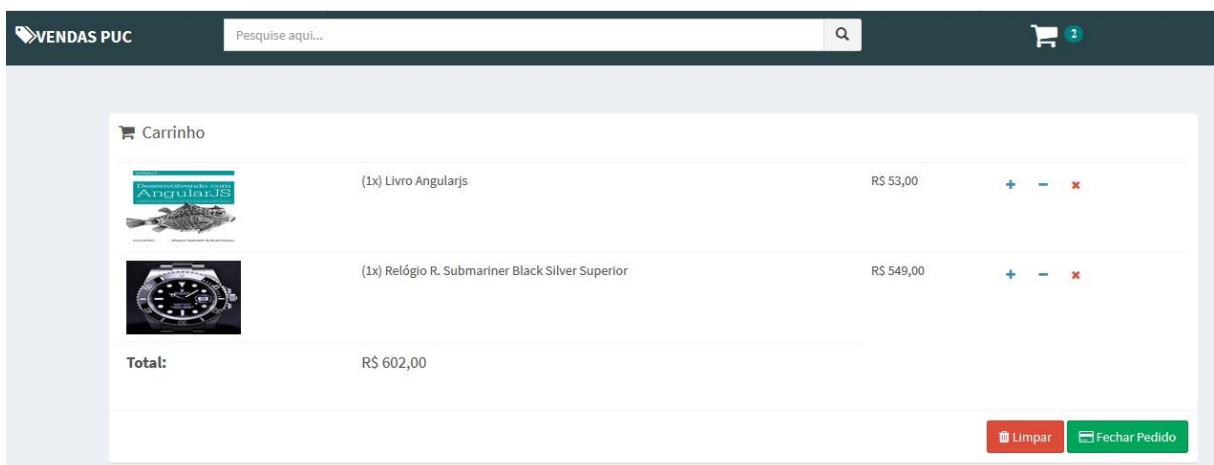
Evidências do cenário 1:



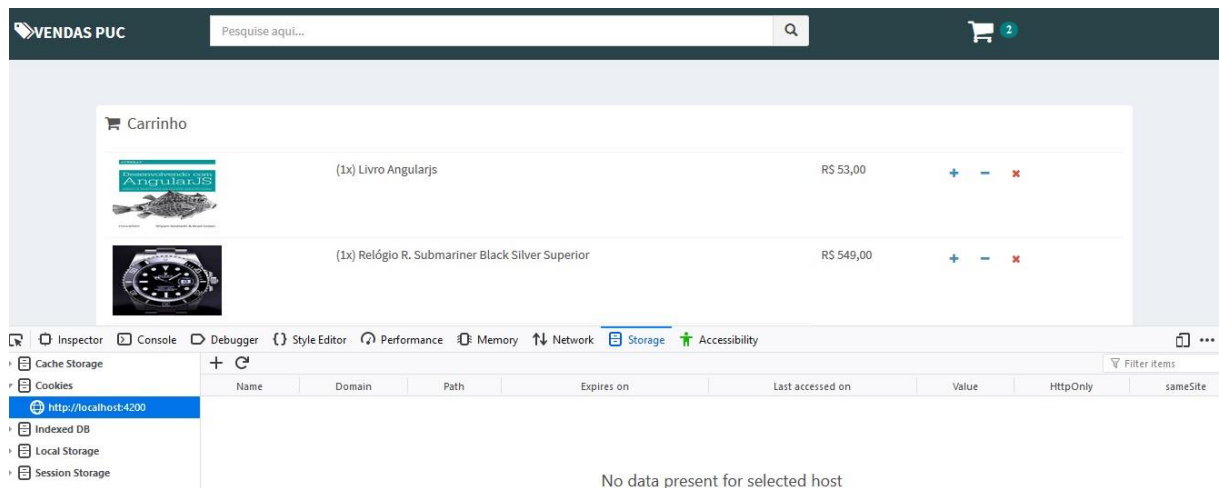
Etapas: Etapa 1: Usuário não identificado acessando página inicial do sistema.



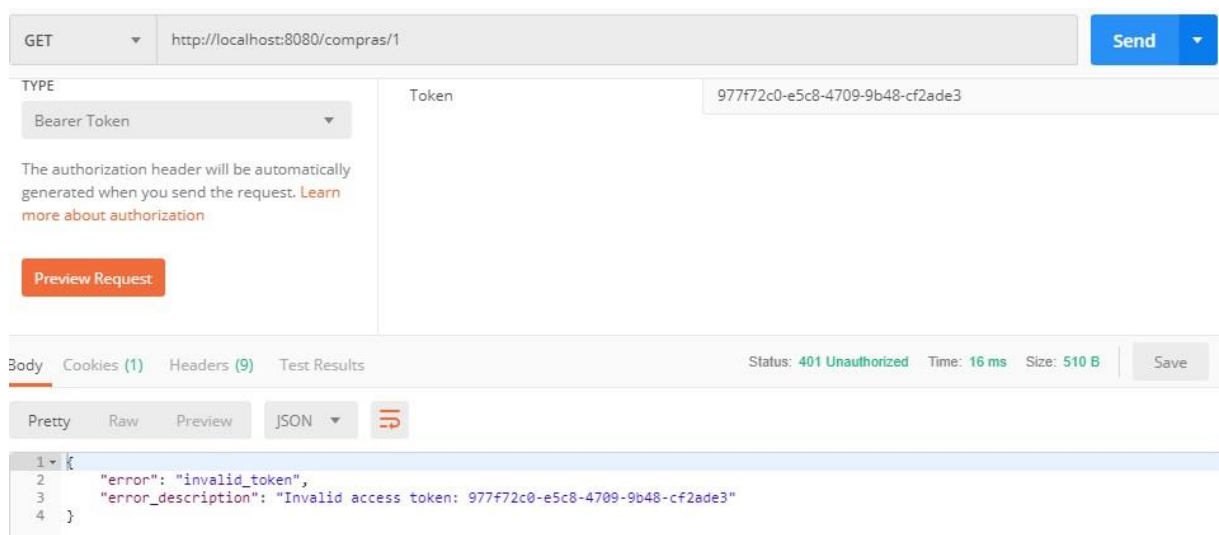
Etapa 2: Usuário não autenticado acessando a lista de produtos.



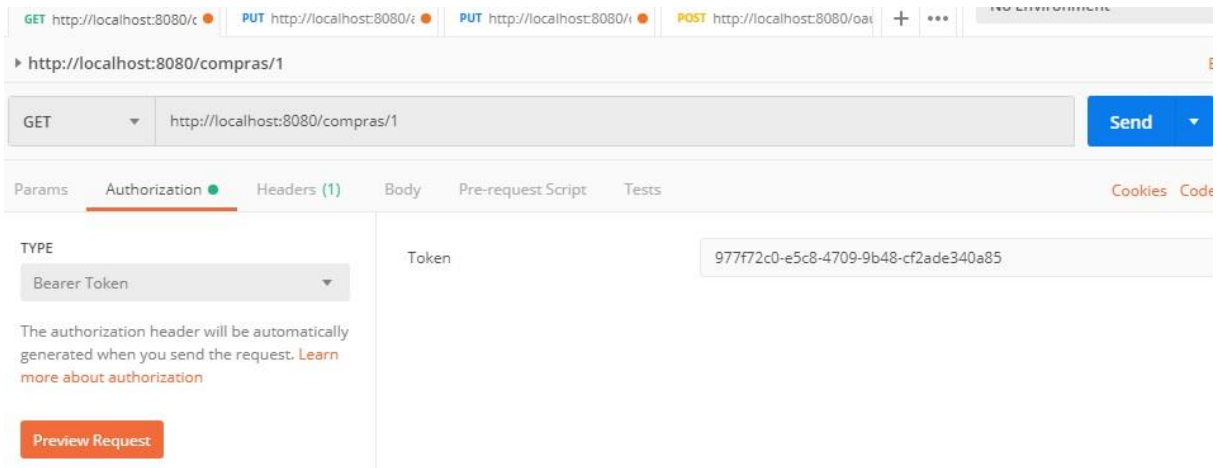
Etapa 3: Usuário não autenticado adicionou produtos ao carrinho e consegue visualizá-los, alterar sua quantidade ou removê-los.



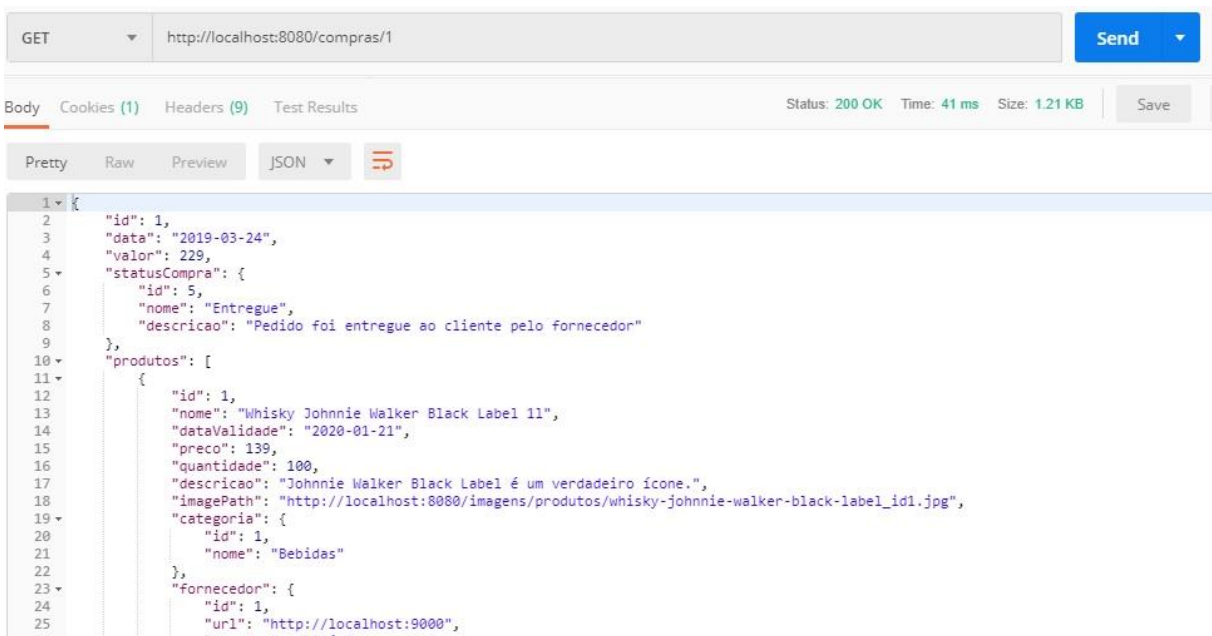
Etapas 4: Evidência de usuário não autenticado (não há token na localstorage do browser, apenas o carrinho).



Etapas 5: Usuário tenta acessar diretamente a URL de pedidos (/compras), sem estar autenticado e recebido um resposta de erro token invalido.



Etapa 6: Usuário informa suas credenciais (token valido).



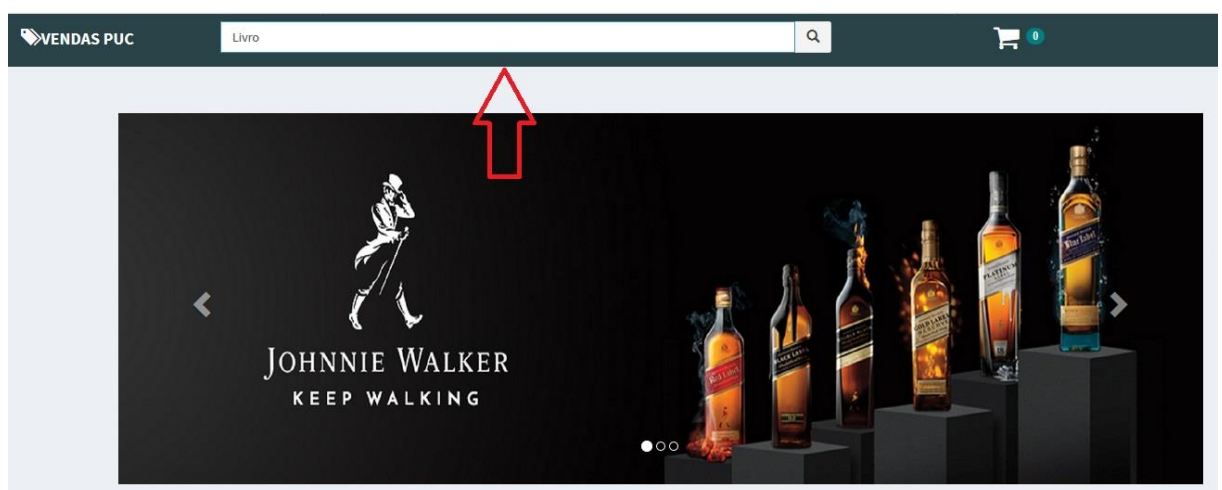
Etapa 7: Usuário recebe da lista de compras.

• Cenário 2

Atributo de qualidade:	Usabilidade
Requisito de qualidade:	O sistema deve prover uma boa usabilidade.
Preocupação	
Fornecer interfaces simples para da rapidez na navegação e tornar a experiência do usuário bem mais fácil e objetiva.	
Cenário 2	

Ambiente:	
Sistema em operação normal	
Estímulo:	
Usuário buscando produtos e adicionando ao carrinho.	
Mecanismo:	
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.	
Medida de resposta:	
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.	
Considerações sobre a arquitetura:	
Riscos:	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.
Pontos de sensibilidade:	Balanceamento de carga ativo.
Trade-off:	Não há.

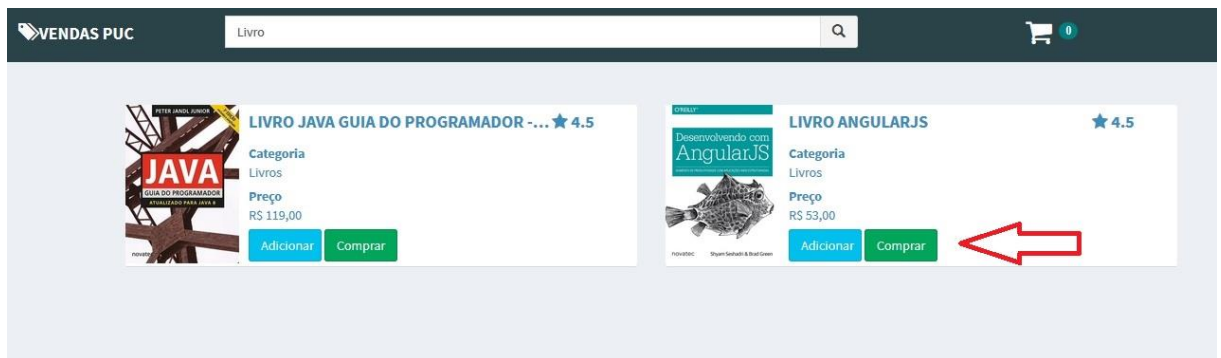
Evidências do cenário 2:



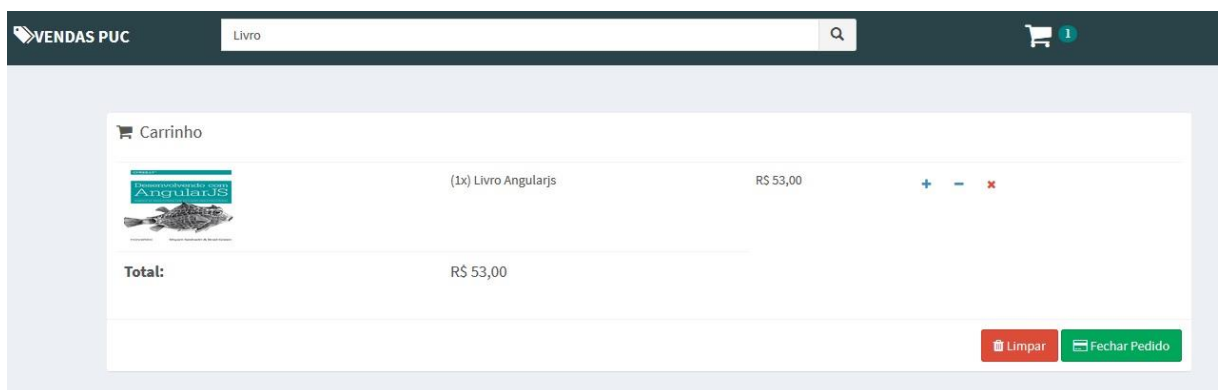
Etapas 1: Usuário acessa a página inicial da loja e já informa o nome do produto desejado (sem sequer precisar scollar). Depois clica no botão de lupa para iniciar a pesquisa.



Etapa 2: O sistema exibi com os resultados da pesquisa efetuada. Usuário pode refinar ainda mais sua pesquisa, e o menu ainda está visível.



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

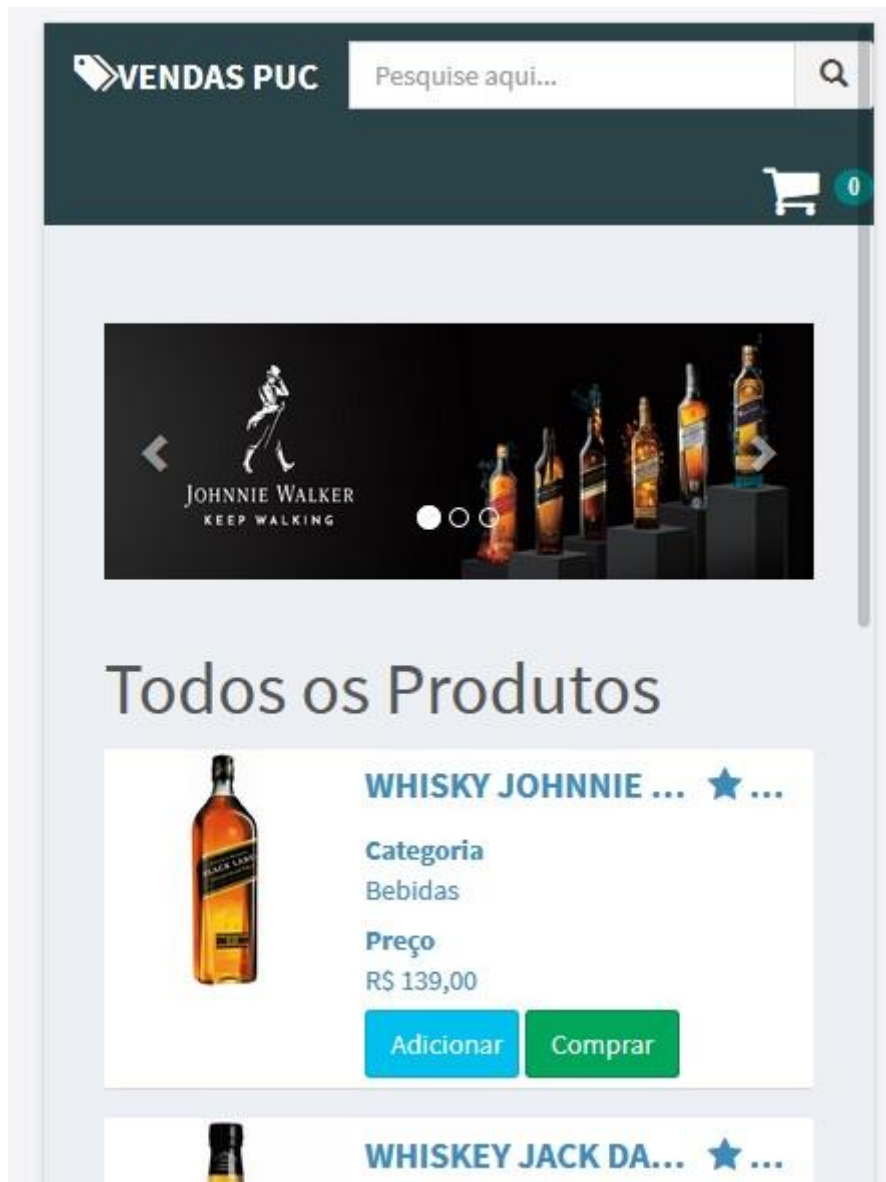


Etapa 4: Carrinho é exibido com o produto adicionado. Para chegar até aqui foram necessários apenas três . O usuário então está pronto para finalizar o processo de compra.

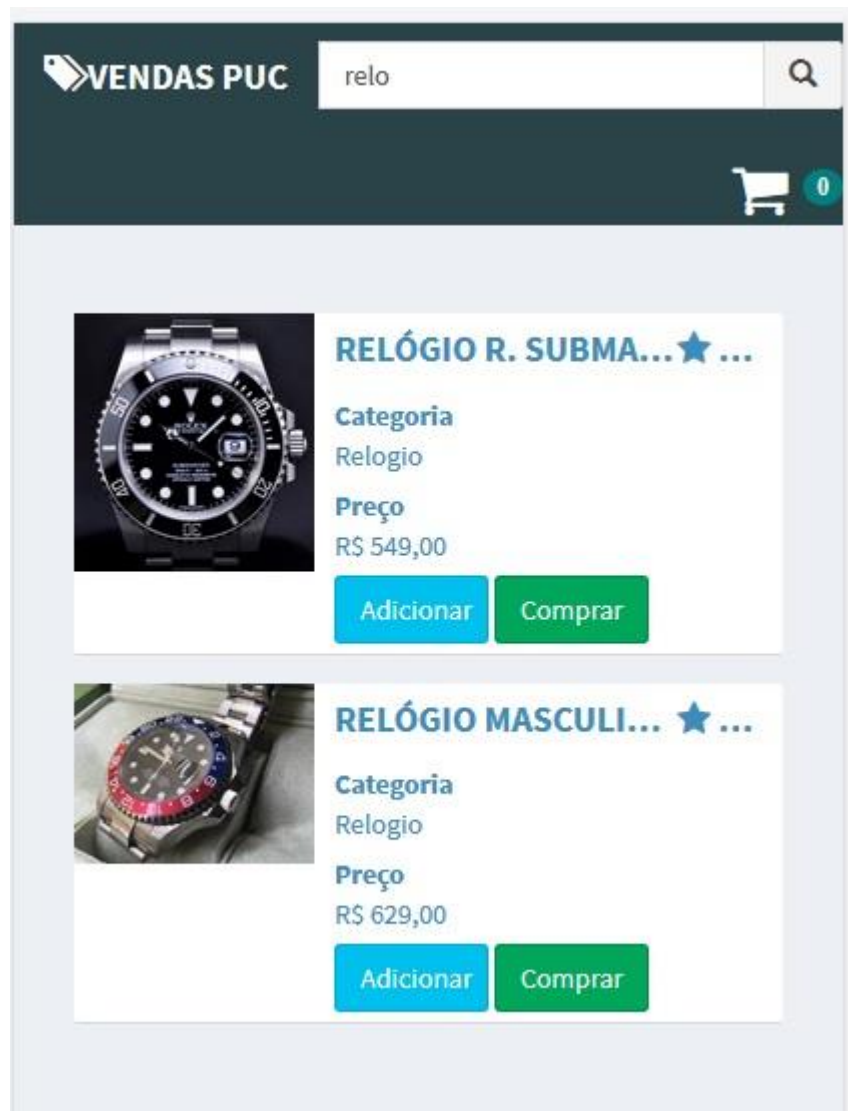
- **Cenário 3**

Atributo de qualidade:	Acessibilidade
Requisito de qualidade:	O sistema deve suportar ambientes web responsivos e ambientes móveis.
Preocupação	
O sistema deve se adaptar a interfaces de diversos tamanhos sem perda de funcionalidade e sem causar impactos na qualidade de navegação.	
Cenário 3	
Ambiente:	
Sistema em operação normal	
Estímulo:	
Usuário buscando produtos e adicionando ao carrinho.	
Mecanismo:	
Criação de telas utilizando mecanismos de design responsivos e ajustáveis, movimentando os componentes para que caibam em dispositivos diferentes.	
Medida de resposta:	
O sistema deve se adaptar a resoluções de tela dos diversos dispositivos, sem perder funcionalidades.	
Considerações sobre a arquitetura:	
Riscos:	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 antigos poderão causar alguns deslocamentos indesejáveis de componentes (porém sem a perda das funcionalidades).
Pontos de sensibilidade:	Não há.
Trade-off:	Não há.

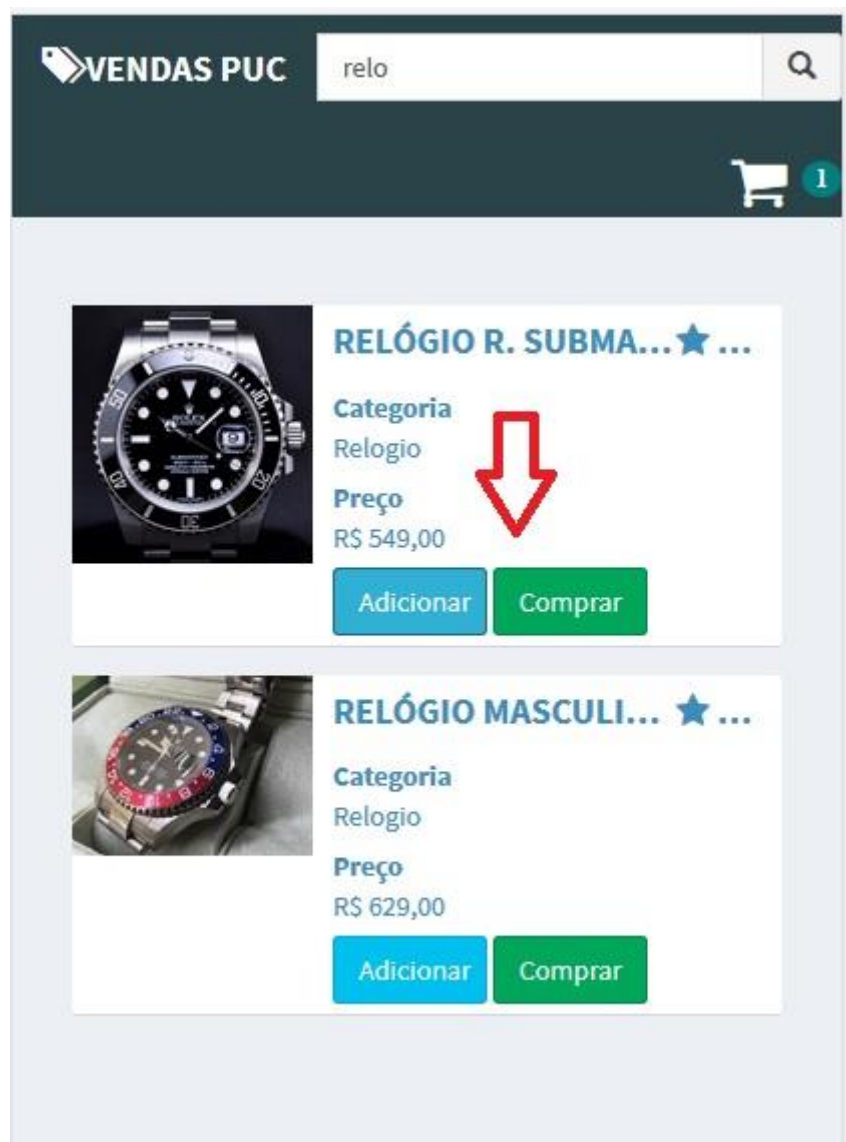
Evidências do cenário 3:



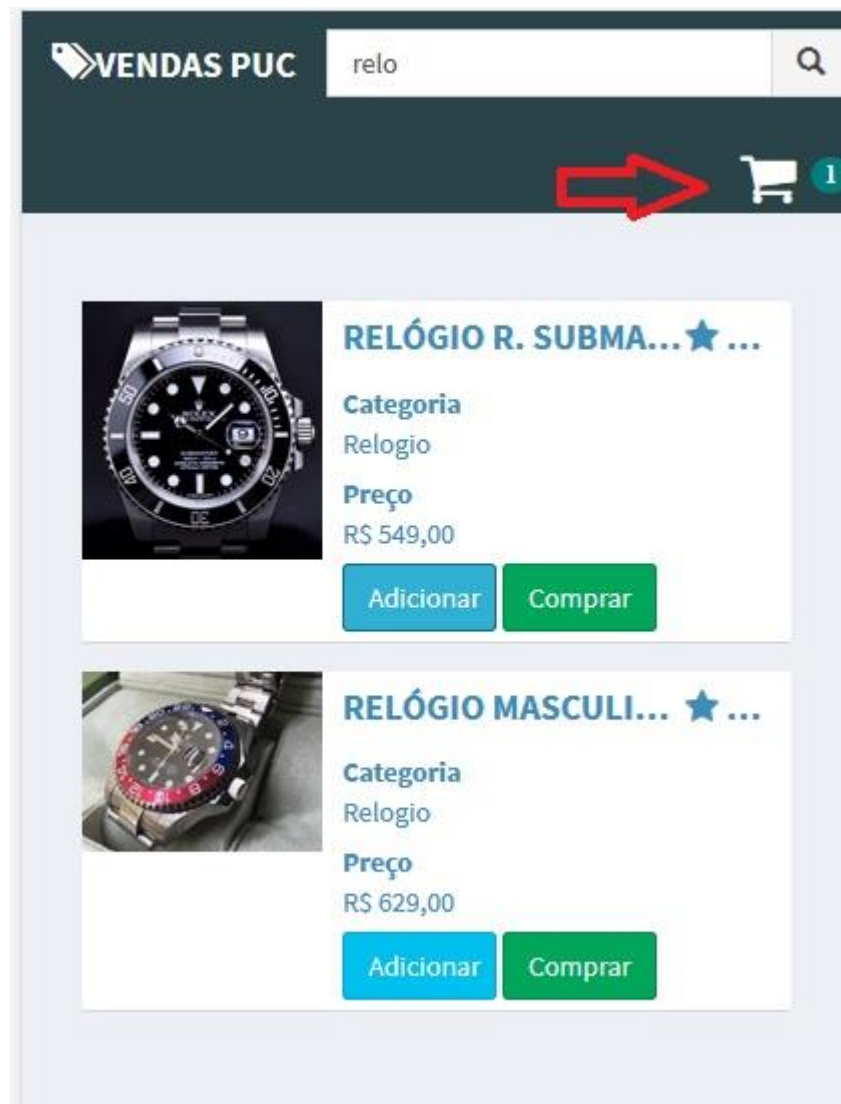
Etapas 1: Usuário acessa a aplicação por um dispositivo móvel.



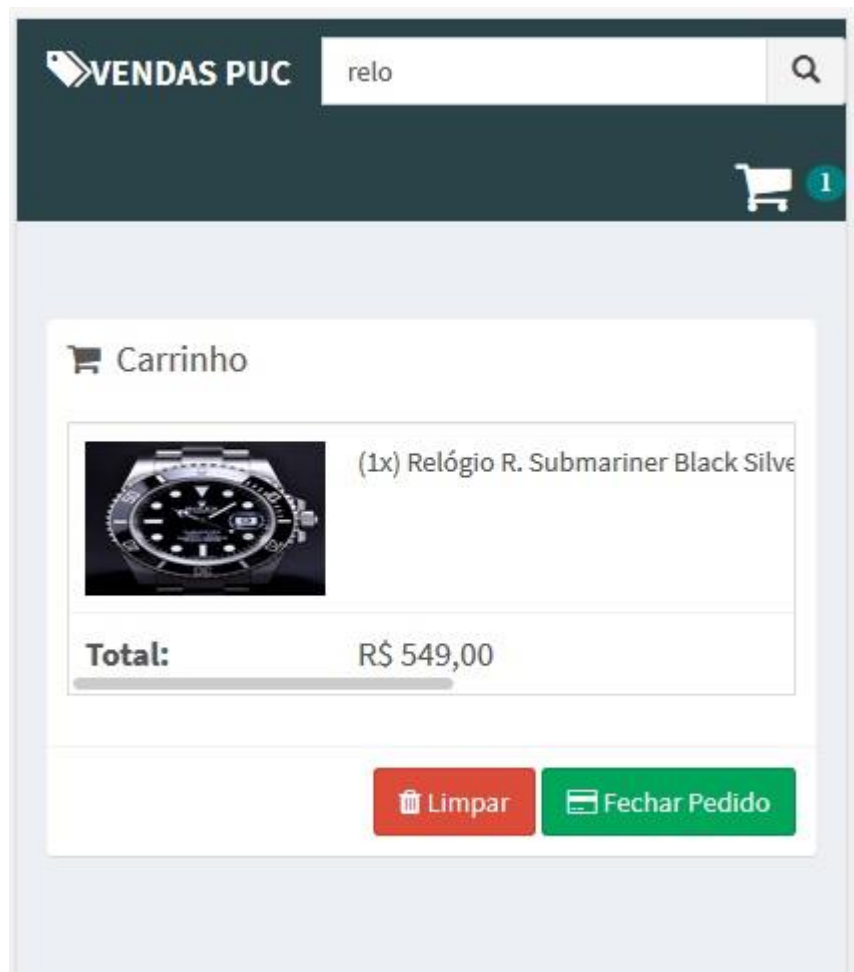
Etapa 2: Usuário pesquisa o nome do produto desejado e é direcionado para a lista de produtos encontrado com o critério de pesquisa.



Etapa 3: Usuário desliza até o produto desejado e toca em “Adicionar”.



Etapa 4: No menu, usuário seleciona a opção “Carrinho”.



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

- Cenário 4**

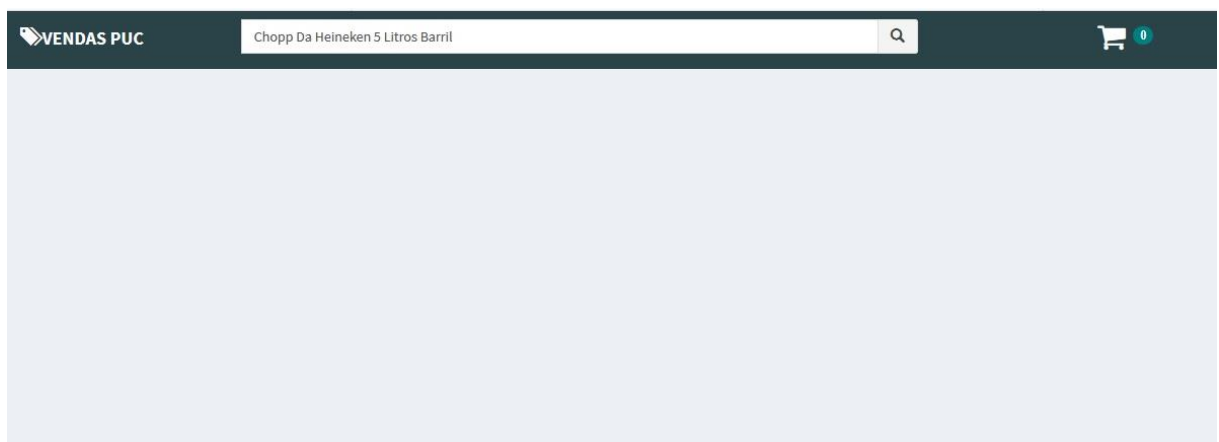
Atributo de qualidade:	Interoperabilidade
Requisito de qualidade:	O sistema deve se comunicar com vários sistemas
Preocupação	
Possibilitar que o sistema realize integrações com fornecedores	
Cenário 1	
Ambiente:	
Sistema em operação normal	
Estímulo:	
Sistema executando processamento de consulta de estoque dos fornecedores.	
Mecanismo:	

Criar um mecanismo de acesso aos sistemas de fornecedores	
Medida de resposta:	
Sistema conseguiu a comunicação e obteve o status atualizado dos produtos.	
Considerações sobre a arquitetura:	
Riscos:	Comunicações via rede estão sujeitas à instabilidade e indisponibilidade do ambiente a que se deseja acessar..
Pontos de sensibilidade:	Não há.
Trade-off:	Não há.

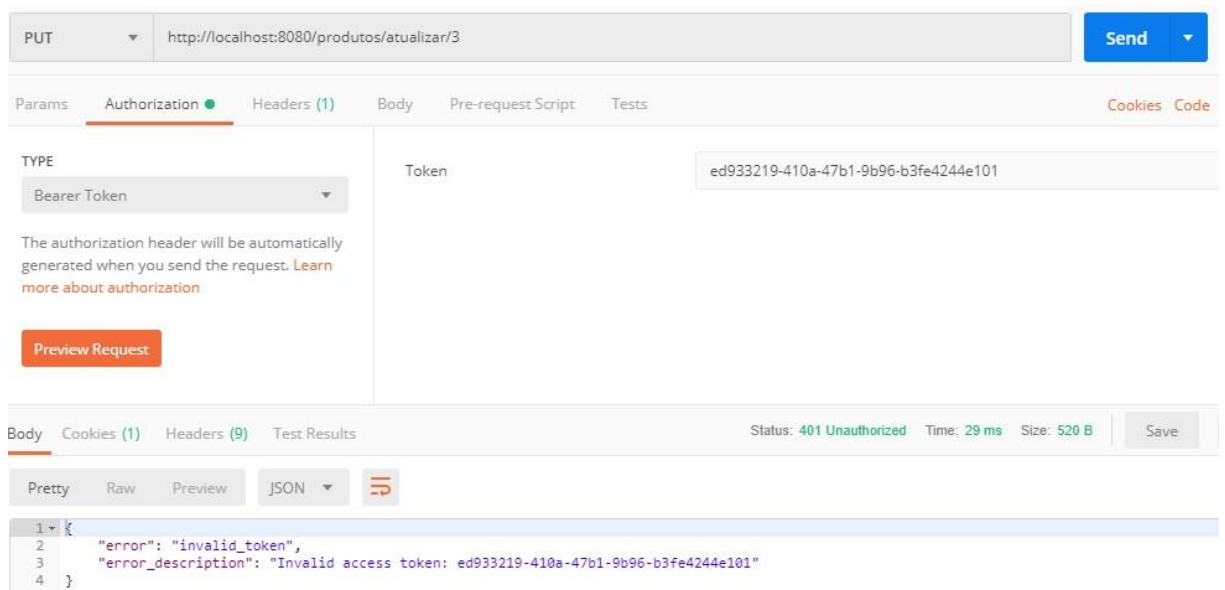
Evidências do cenário 4:

nome	categoria_id	fornecedor_id	data_validade	descricao	preco	quantidade	codigo	ativo
-5-litros-barril.jpg Chopp Da Heineken 5 Litros Barril	1	1	2020-12-31	Heineken é uma cerveja super premium puro ma...	97	100	3	0

Etapa 1: Evidência de que o produto “Chopp Da Heineken 5 Litros Barril” está indisponível (campo “Ativo” = 0) na base de dados, o que o torna elegível para busca de disponibilidade no serviço do fornecedor.



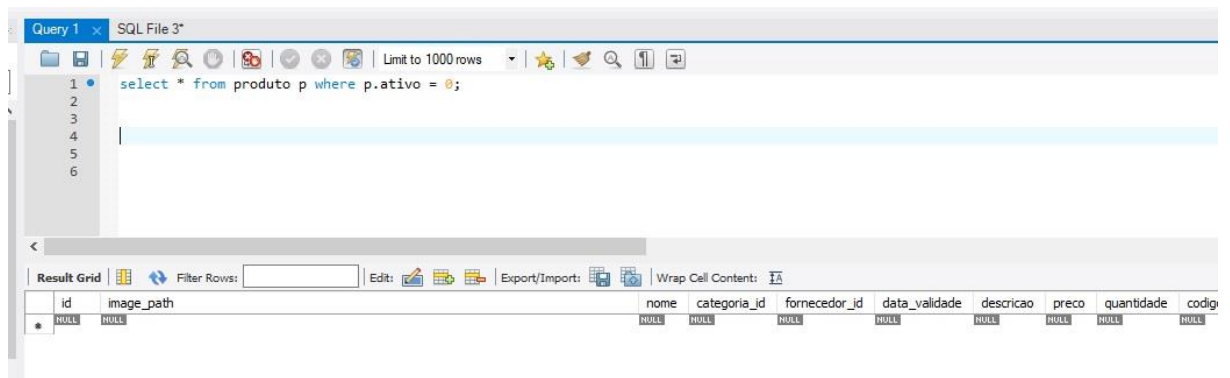
Etapa 2: Sistema evidenciando que o produto não foi encontrado.



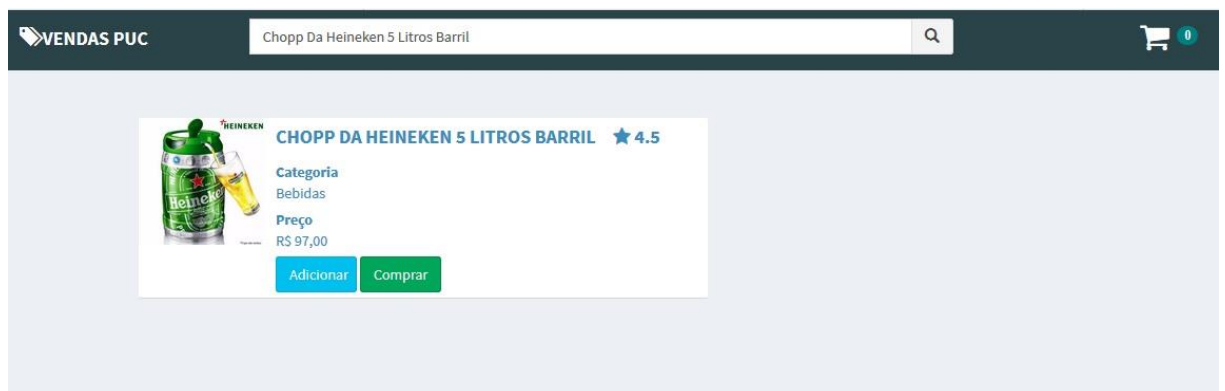
Etapa 3: Utilizando o POSTMAN (extensão para Google Chrome que simula chamadas à API's Restful), como se trata de um recurso que necessita de um token para acessar o mesmo e adicionado uma mensagem de erro.



Etapa 4 Utilizando o POSTMAN agora com um token valido o produto e atualizado na base de dados e torna ativo.



Etapa 6: Consulta ao banco de dados para exibir os produtos inativos. Reparar que o produto “Chopp Da Heineken 5 Litros Barril” sumiu da lista de inativos, pois sua disponibilidade foi atualizada após consulta ao serviço de seu fornecedor.



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

- Cenário 5**

Atributo de qualidade:	Interoperabilidade
Requisito de qualidade:	O sistema deve prover recursos para que a comunicação no sentido “sistema terceiro – loja” ocorra dentro de padrões atuais de tecnologias.
Preocupação	
Possibilitar que o sistema receba integrações (de fornecedores e da plataforma de pagamentos), provendo endpoints construídos com conceitos modernos.	
Cenário 1	
Ambiente:	
Sistema em operação normal	
Estímulo:	
Sistema recebendo integração de um fornecedor informando que uma compra mudou de status (saiu para entrega).	
Mecanismo:	

Criar uma API Restful exposta através do gateway para que os fornecedores possam consumir e notificar sobre alterações na mudança de status.

Medida de resposta:

Fornecedor conseguiu estabelecer a comunicação (depois de obter credenciais válidas na API de Autenticação).

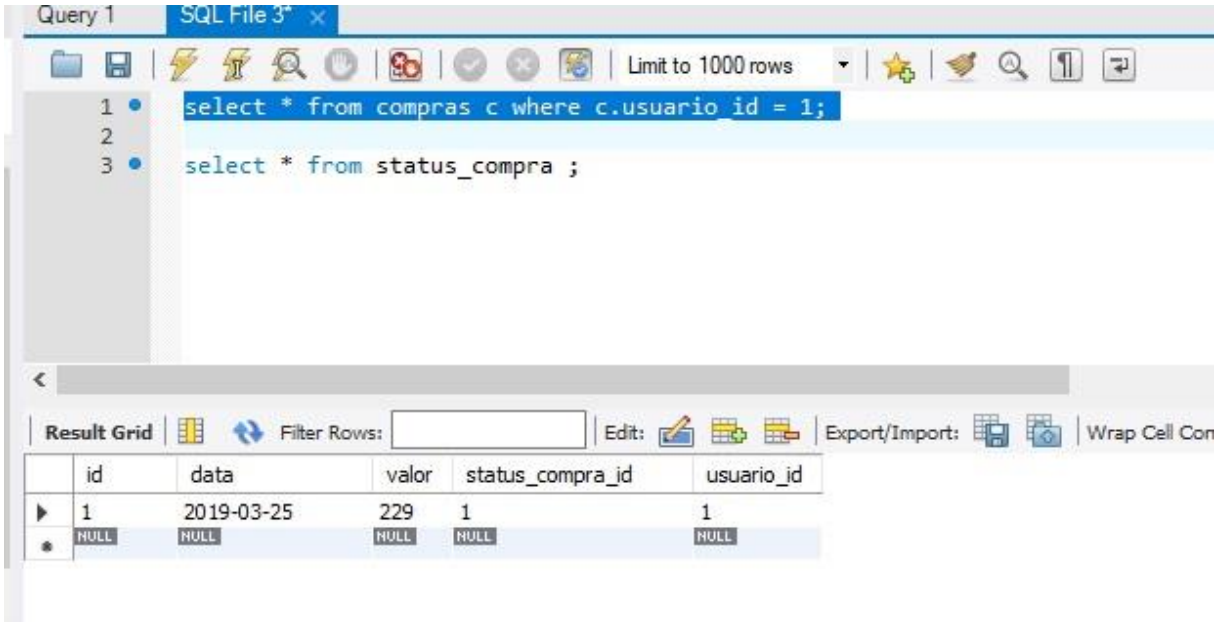
Considerações sobre a arquitetura:

Riscos: 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.

Pontos de sensibilidade: Não há.

Trade-off: Não há.

Evidências do cenário 5:

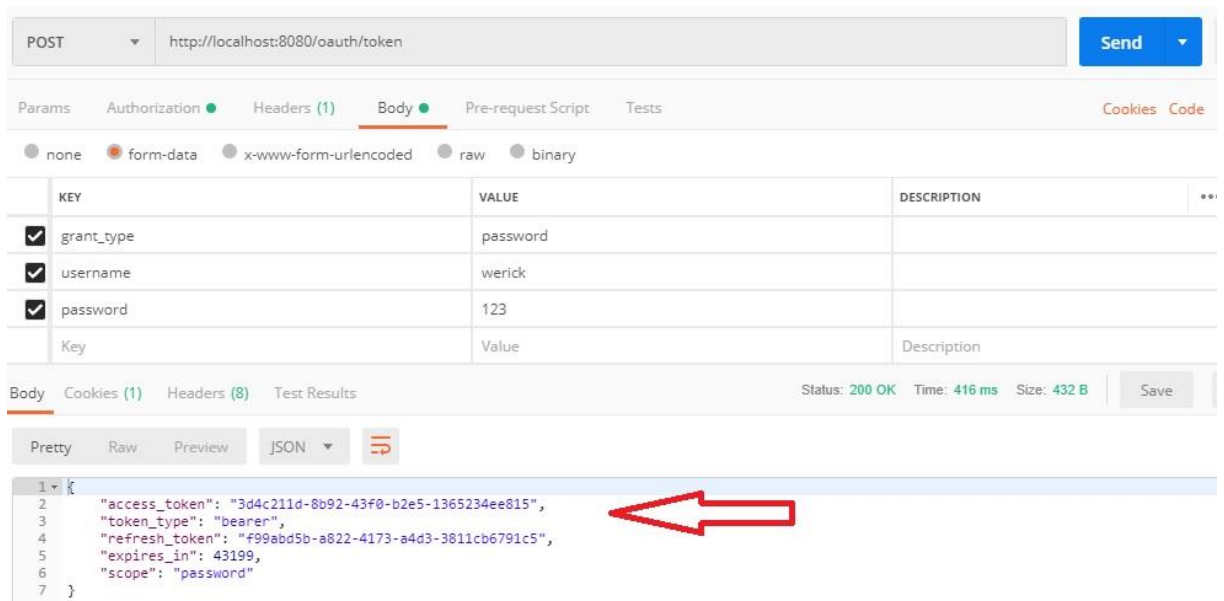


The screenshot shows a SQL query tool interface. The query editor displays two queries: `select * from compras c where c.usuario_id = 1;` and `select * from status_compra ;`. The results grid shows a single row with the following data:

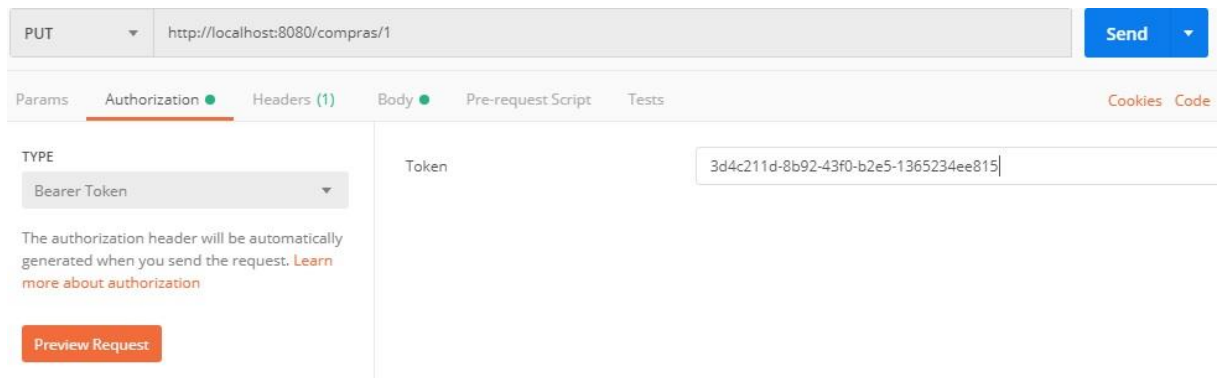
id	data	valor	status_compra_id	usuario_id
1	2019-03-25	229	1	1

The interface also includes a toolbar with various icons and a 'Limit to 1000 rows' dropdown.

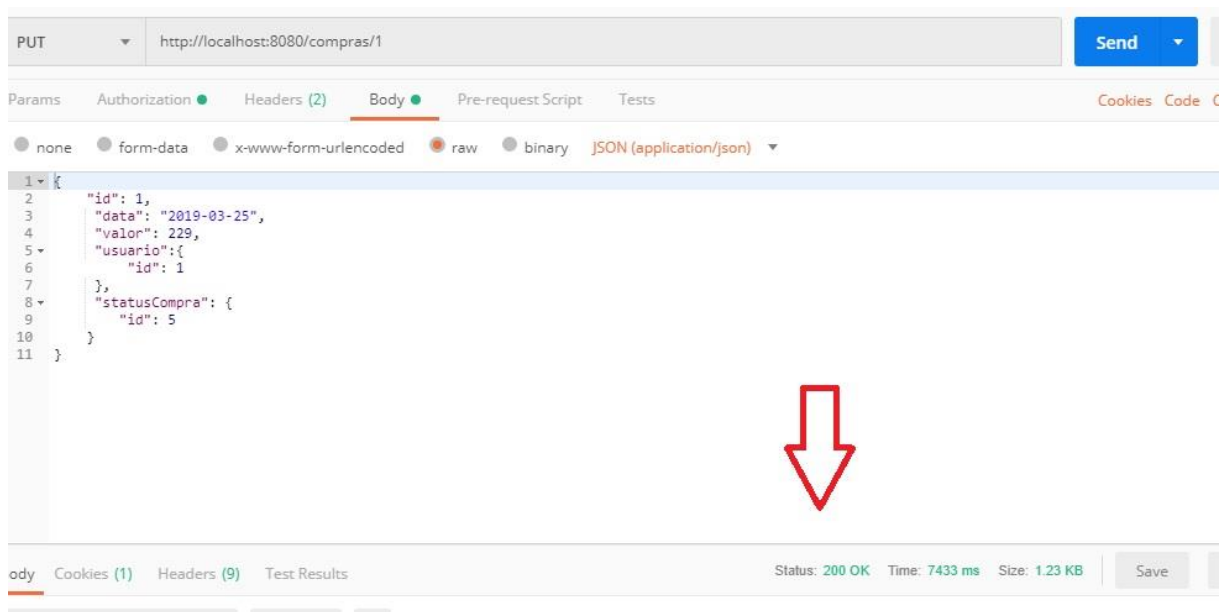
Etapa 1: Evidência de que a compra encontra-se na base de dados no status 1 (Pedido foi recebido pelo fornecedor).



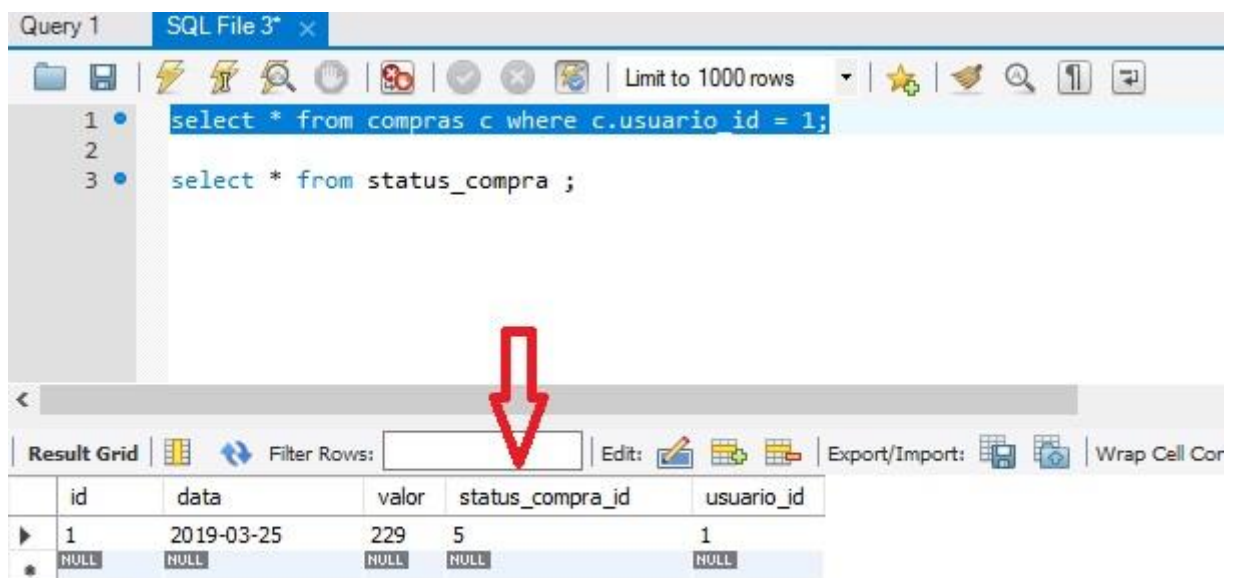
Etapa 2: Utilizando o POSTMAN (extensão para Google Chrome que simula chamadas à API's Restful), enviamos as credenciais do fornecedor para a API de autenticação através do gateway e obtemos um oauth.



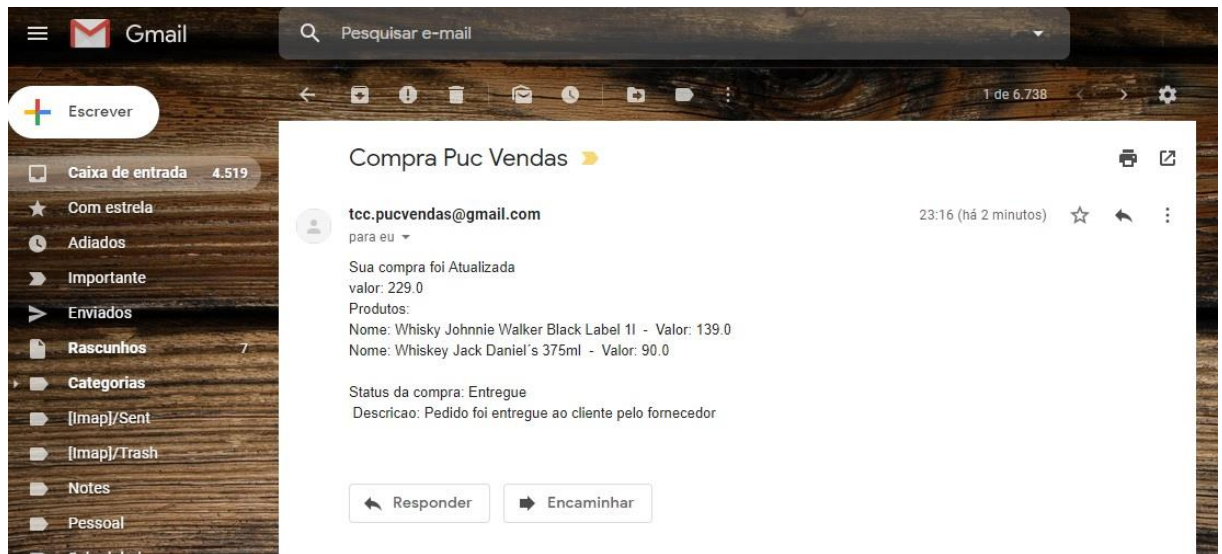
Etapa 3: Configuramos uma nova requisição, dessa vez para o endpoint de atualização de status de entrega (registro de eventos). No authorization da requisição, incluímos o token gerado na etapa 2.



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.



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.



Etapas 6: Verificamos que o cliente recebeu em seu e-mail instantaneamente a informação de que seu pedido mudou de status.

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:

Requisitos não funcionais	Testado	Homologado
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,	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 .Java com spring boot 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. 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 Hibernate supre bem as necessidades. 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). Porém, a comunicação é muito dependente da implementação realizada em .Java.

Do ponto de vista da implantação localhos, podemos 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 atraentes 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

O trabalho apresentou um projeto arquitetural para uma plataforma de loja virtual baseada em dropshipping. No qual, foram apresentadas diversas tecnologias, que se complementaram entre si, algumas extremamente atuais como (apis rest com spring boot) correlacionarem-se entre si com tecnologias emergentes como (.java, spring e Angular, em constante evolução). Optou-se por manter o suporte às comunicações com sistemas legados (COBOL).

Foram apresentadas sugestões de implantações em nuvem, devido ao tema está cada vez mais presente no cotidiano e ser uma tendência global, entretanto, não há qualquer dependência, podendo então, implantar os componentes propostos em ambientes on-premises (com infraestrutura própria).

Conclui-se assim, que os objetivos foram atingidos. Porém cabe ressaltar, que há margem para melhora. Esta pode ocorrer em uma próxima versão.

REFERÊNCIAS

Spring boot : < <https://spring.io/guides>> . >. Acesso em: 20 de abril de 2019.

AWS Documentation. Amazon. Disponível em:
<https://docs.aws.amazon.com/index.html#lang/pt_br>. Acesso em: 20 de abril de 2019.

Angular Docs. Google. Disponível em: <<https://angular.io/docs>>. Acesso em: 20 de abril de 2019.

Bootstrap.: < <https://getbootstrap.com/docs/4.3/getting-started/introduction/>>. Acesso em: 20 de abril de 2019.

APÊNDICES

URL da apresentação da POC no Youtube:

<<https://www.youtube.com/watch?v=7z3XzijUIyQ&t=11s>>

URL do código fonte. No arquivo README.md de cada pasta/projeto estar descrito o passo a passo para executar o código fonte localhost como foi apresentado na poc.

<https://github.com/wericksilva/tcc-puc-minas>