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

Karina Martins Martinez

LOGÍSTICA BASEADA EM MICROSSERVIÇOS

Rio de Janeiro
2021

Karina Martins Martinez

LOGÍSTICA BASEADA EM MICROSSERVIÇOS

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

Orientador(a): Pedro A. Oliveira

Rio de Janeiro

2021

AGRADECIMENTOS

Primeiramente agradeço a todos os professores e funcionários da PUC Minas que fizeram com que essa pós-graduação fosse possível à distância. Não me esquecerei também de todos os amigos, colegas de trabalho e superiores que me orientaram a seguir este caminho. Aos meus familiares que me apoiaram e me deram suporte para que eu pudesse estudar e trabalhar mesmo durante uma pandemia que teve seus inúmeros desafios. A essas e muitas outras pessoas serei sempre grata por me ajudarem e incentivarem a chegar até aqui.

RESUMO

O crescimento atual de empresas na área de logística e transporte criou um cenário competitivo. Muitas delas utilizam ferramentas de software e criam seus próprios sistemas para aprimorar seus processos. Visando auxiliar o cumprimento de suas metas, a empresa fictícia Boa Entrega procura uma solução para o desenvolvimento de seus sistemas. Neste trabalho, uma solução de arquitetura de microsserviços foi proposta, levando em consideração requisitos relevantes para o seu sucesso. A solução possui quatro módulos com diferentes propósitos. O padrão API Gateway foi escolhido para realizar a integração entre os componentes da arquitetura, tendo como pontos fortes a padronização do acesso e como pontos fracos as possíveis falhas e efeitos no desempenho. Tecnologias mais atuais de desenvolvimento para aplicações web foram utilizadas para equilibrar a solução e otimizar as requisições feitas aos serviços. A evolução da solução proposta deverá ser acompanhada de regras e mecanismos que garantam que os requisitos exigidos sejam atingidos.

Palavras-chave: arquitetura de software, microsserviços, API gateway

SUMÁRIO

1. Apresentação

Devido ao contexto da pandemia do COVID-19, serviços de e-commerce tem crescido em todo o Brasil. De acordo com o relatório de um indicador de vendas no varejo, “o e-commerce brasileiro apresentou um crescimento de 75% em 2020 se comparado ao ano anterior” (NOVAREJO). Esse crescimento fez com que muitas empresas tenham se especializado na área de logística que inclui o recolhimento, o transporte e a entrega de mercadorias de diversos tipos.

Muitos são os desafios encontrados no processo de transporte. Empresas como o Frete Barato procuraram implementar soluções tecnológicas para se manterem competitivas no mercado, reduzindo os custos das suas operações e tornando os seus preços mais acessíveis aos clientes (Economia SC). Os investimentos na área de TI e na automatização de processos podem aprimorar o funcionamento e facilitar a execução de atividades. Visando explorar esse tema, para o escopo deste trabalho será considerada a empresa fictícia Boa Entrega que atua no mercado brasileiro como uma transportadora de grande porte.

1.1 Problema

A empresa Boa Entrega criou várias metas gerais que estabelecem critérios para o tempo médio de entrega dos produtos, expansão da atuação no país, criação de parcerias com outras transportadoras e o aumento global do faturamento. A gestão de estoques e o registro de todas as etapas das suas operações são fatores determinantes para o sucesso da empresa e para o alcance de suas metas. Para realizar muitas de suas operações são utilizadas soluções de software desenvolvidas internamente pela equipe de TI.

Muitas empresas se empenharam em realizar a transformação digital durante a pandemia, mudando os processos que antes eram feitos de forma manual para serem feitos com o auxílio de ferramentas de software. Isto acirrou a concorrência em questão de qualidade e rapidez na área de transporte na qual a Boa Entrega está inserida. Agora, a empresa também busca melhorar o desempenho, a qualidade e a disponibilidade de seus sistemas para atender melhor aos seus clientes e tomar melhores decisões estratégicas. Considerando a expectativa de aumento da demanda é necessário que as novas soluções estejam sempre disponíveis para os usuários e possam ser acessadas de várias formas, aumentando a resiliência da empresa ao lidar com situações inesperadas.

1.2 Objetivo do trabalho

O objetivo geral deste trabalho é apresentar uma proposta de arquitetura para incluir novas soluções integradas aos sistemas já existentes na empresa Boa Entrega. Tal proposta será denominada de gestão de serviços de logística e visará a criação de quatro novos módulos que irão auxiliar principalmente nas áreas de gestão da empresa. Os módulos a serem definidos na arquitetura são descritos a seguir.

1. Módulo de informações cadastrais, que irá possibilitar a obtenção e a manutenção de informações de clientes, fornecedores, depósitos e mercadorias. Esse módulo deverá possuir tipos diferentes de perfis para atender à diversos tipos de usuários.
2. Módulo de serviços ao cliente, para que seja possível realizar um acompanhamento mais detalhado de todos os processos de atendimento ao cliente que a empresa Boa Entrega possui. Esse módulo deverá ser baseado em uma solução de workflow com o uso de Bussiness Process Management.
3. Módulo de gestão e estratégia, para que seja possível gerir todas as atividades da empresa apresentando indicadores relevantes para o sucesso das operações. Será escolhida uma ferramenta de gestão corporativa já existente no mercado para contemplar essa solução.
4. Módulo de ciência de dados, que deverá ser composto por um data warehouse e por uma solução de business intelligence. O data warehouse irá armazenar todos os dados corporativos da Boa Entrega reunindo todas as informações presentes nos mais diversos formatos. Já a solução de business intelligence, deverá tratar o grande volume de dados para dar apoio às tomadas de decisões.

Este trabalho tem como objetivos mais específicos:

- apresentar com detalhamento a solução criada para o módulo de informações cadastrais, seus requisitos, arquitetura e implantação a fim de validar a proposta de arquitetura.
- oferecer soluções de integração que possam ser utilizadas com diversos tipos de tecnologia. A empresa Boa Entrega possui sistemas legados importantes aos quais não se têm um conhecimento muito amplo sobre como são feitas as suas integrações e quais são as tecnologias utilizadas. Por conta disto, este trabalho também focará neste ponto.
- tornar a solução algo simples e viável de ser utilizada pelos usuários. Para que possa ocorrer a automatização de todos os processos conforme o estipulado no plano de metas da empresa é importante que todos os funcionários possuam fácil acesso e tenham uma curva de aprendizado favorável ao prazo determinado pela empresa para a utilização dos novos sistemas.

1,3 Definições e Abreviaturas

Algumas das siglas utilizadas neste trabalho são definidas a seguir.

1. MIC - módulo de informações cadastrais.
2. MSC - módulo de serviços ao cliente.
3. MGE - módulo de gestão e estratégia.
4. MCD - módulo de ciência de dados.

2. Especificação da Solução

Está seção irá descrever os requisitos presentes nesse projeto arquitetural, sendo divididos em dois grupos: funcionais e não funcionais.

2.1 Requisitos Funcionais

ID	Descrição Resumida	Dificuldade (B/M/A)*	Prioridade (B/M/A)*
RF01	O sistema deve atender a diferentes perfis de usuário.	A	A
RF02	O sistema deve permitir manter informações de cadastro de depósitos, mercadorias e registros de mercadorias em depósitos através da web.	M	A
RF03	Apenas usuários autorizados deverão poder acessar o sistema, mantendo a segurança das informações.	M	A
RF04	O sistema deve possibilitar a atualização de informações cadastrais pelos clientes.	M	M
RF05	O sistema deve permitir alterar a localização de registros de mercadorias, mantendo o seu histórico.	B	M
RF06	O sistema deve permitir a obtenção do histórico de alteração da localização dos registros de mercadorias em depósitos.	B	M

*B=Baixa, M=Média, A=Alta.

2.2 Requisitos Não Funcionais

ID	Descrição	Prioridade B/M/A
RNF01	O sistema deve ser acessível nas plataformas *web* e móvel.	A
RNF02	O sistema deve possuir boa interoperabilidade, sendo possível acessar e enviar informações de forma transparente.	A
RNF03	O sistema deve apresentar bom desempenho, com tempo de resposta de até 4 segundos para cada requisição.	A

RNF04	O sistema deve possuir interface responsiva	M
RNF05	O sistema deverá ter boa usabilidade, sendo necessário no máximo até 5 etapas para atingir o objetivo final do usuário partindo da tela inicial.	M
RNF06	O sistema deverá prover alta escalabilidade, mantendo o limite de 5 segundos de resposta quando ocorrer um aumento significativo de usuários.	M
RNF07	O sistema deve ser seguro, permitindo o acesso as informações apenas de forma autenticada.	M
RNF08	O sistema deve possuir boa manutenibilidade, sendo possível corrigir e alterar o funcionamento dos componentes de forma isolada.	M
RNF09	O sistema deve ter alta disponibilidade, recuperando-se de falhas em até 2 minutos.	B
RNF10	O sistema deve ser testável em todas as suas funcionalidades, possuído cobertura de testes de unidade em pelo menos 70% do código.	B

2.3 Restrições Arquiteturais

As restrições arquiteturais para a solução foram expressas pelo setor de TI da Boa Entrega dentre os quais se destacam:

- possuir baixo custo;
- ter a possibilidade de integração com os sistemas legados com baixo acoplamento;
- possuir características de aplicação distribuída;
- ser modular e componentizado, utilizando orientação a serviços;
- ser de fácil implantação, desenvolvido utilizando recurso de gestão de configuração, com integração contínua;
- ser hospedado em nuvem híbrida, sendo a forma de hospedagem documentada;
- utilizar APIs ou outros recursos adequados para consumo de serviços.

2.4 Mecanismos Arquiteturais

Análise	*Design*	Implementação
Persistência	Anotações nas classes Java	Hibernate
Front end	Single Page Application	React
Back end	SpringBoot API	Java

Integração	API Gateway	AWS API Gateway
Log do sistema	Log de operações	Log4j
Teste de Software	Testes unitários	Junit
Deploy	Deploy das aplicações em servidores na nuvem	Conexão com Github e arquivos de configuração do tipo YAML

3. Modelagem Arquitetural

Esta seção apresenta a modelagem da arquitetura proposta para os novos sistemas da empresa Boa Entrega. A subseção 3.1 contém o diagrama da macroarquitetura com uma visão geral da solução. A subseção 3.2 inclui alguns casos de uso que serão testados e validados na prova de conceito e a subseção 3.3 exibe os diagramas que complementam a definição da arquitetura.

3.1 Macroarquitetura

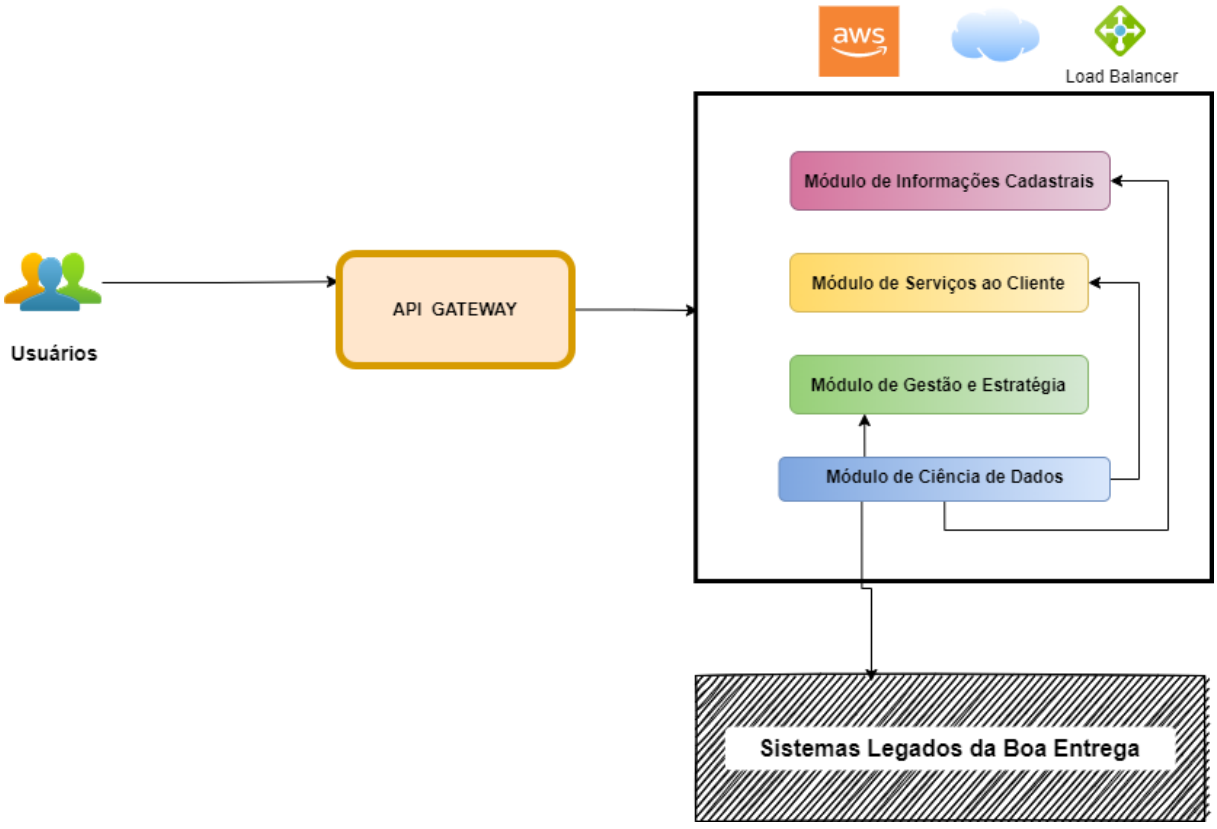

Figura 1 - Visão Geral da Solução.

A figura 1 mostra o diagrama da solução proposta. Todos os módulos serão hospedados em nuvens privadas ou públicas, sendo a plataforma da AWS um dos serviços utilizados. O módulo de ciência de dados se comunica com os demais

módulos da solução e com os sistemas legados para recolher e armazenar os mais variados tipos de dados. O acesso aos serviços disponíveis nos módulos será feito utilizando do padrão API gateway. Este padrão visa garantir mais segurança e simplicidade ao disponibilizar uma interface única e padronizada para os microsserviços internos. Também torna mais simples a integração dos módulos entre si, com os sistemas legados e com outros sistemas que possam vir a existir futuramente.

Os módulos serão implantados utilizando tecnologias que permitam a implantação de mecanismos de balanceamento de carga para distribuir o tráfego de requisições entre uma ou mais instâncias de microsserviços. Por ser esperado um aumento na demanda, é importante que esses recursos estejam disponíveis na solução para que possam ser garantidas a escalabilidade e o bom desempenho das aplicações em momentos futuros da utilização destes sistemas.

3.2 Descrição Resumida dos Casos de Uso

Esta seção apresenta os casos de uso a seguir que se relacionam com os requisitos descritos na seção dois deste trabalho. Os mesmos casos de uso serão testados na prova de conceito e na validação da arquitetura.

UC01 – Atualizar informações cadastrais	
Descrição	O usuário com o perfil de cliente deve poder realizar a atualização das informações cadastrais através de um dispositivo mobile sem grandes dificuldades nas operações.
Atores	Usuário logado como cliente em dispositivo mobile
Prioridade	M
Requisitos associados	RF04, RNF01 e RNF04
Fluxo Principal	O usuário com o perfil de cliente realiza o login no sistema através de um celular e acessa a tela de informações cadastrais. O usuário consegue visualizar e editar as suas informações com facilidade.

UC02 – Movimentar registro de mercadoria	
Descrição	O usuário deve poder movimentar registros de mercadoria em depósitos utilizando serviços de geolocalização para verificar a distância entre os depósitos e o endereço de destino do registro.
Atores	Usuário logado com perfil de administrador
Prioridade	A
Requisitos	RF05, RNF03 e RNF05.

associados	
Fluxo Principal	O usuário com o perfil de administrador realiza o login no sistema e acessa a tela de registros de mercadorias. Após localizar o registro cadastrado o usuário seleciona a opção de mover o registro e é redirecionado para a tela de alterar a localização. O usuário seleciona o depósito desejado e clica na opção de verificar distâncias. O sistema retorna as informações de acordo com os serviços de geolocalização sem demorar para processar as requisições. O usuário confirma a seleção de depósito e salva as alterações.

UC03 – Obter histórico de localização	
Descrição	O sistema deve permitir obter o histórico de localização de registros de mercadorias em depósito apenas através de usuários autenticados.
Atores	Usuário autenticado
Prioridade	M
Requisitos associados	RF06, RF03, RNF02 e RNF07.
Fluxo Principal	O usuário utiliza uma aplicação externa e realiza a autenticação no sistema obtendo o token JWT. O token é utilizado para ter acesso ao histórico de localização de um registro de mercadorias em depósito.

3.3 Visão Lógica

Esta seção apresentará diagramas da solução proposta com seus componentes e as relações entre eles. O diagrama de classes, o diagrama de implantação e o modelo de dados apresentarão os componentes criados para o desenvolvimento da prova de conceito focada no módulo de informações cadastrais. Já o diagrama de componentes apresentará a descrição completa da proposta arquitetural.

3.3.1 Diagrama de Classes

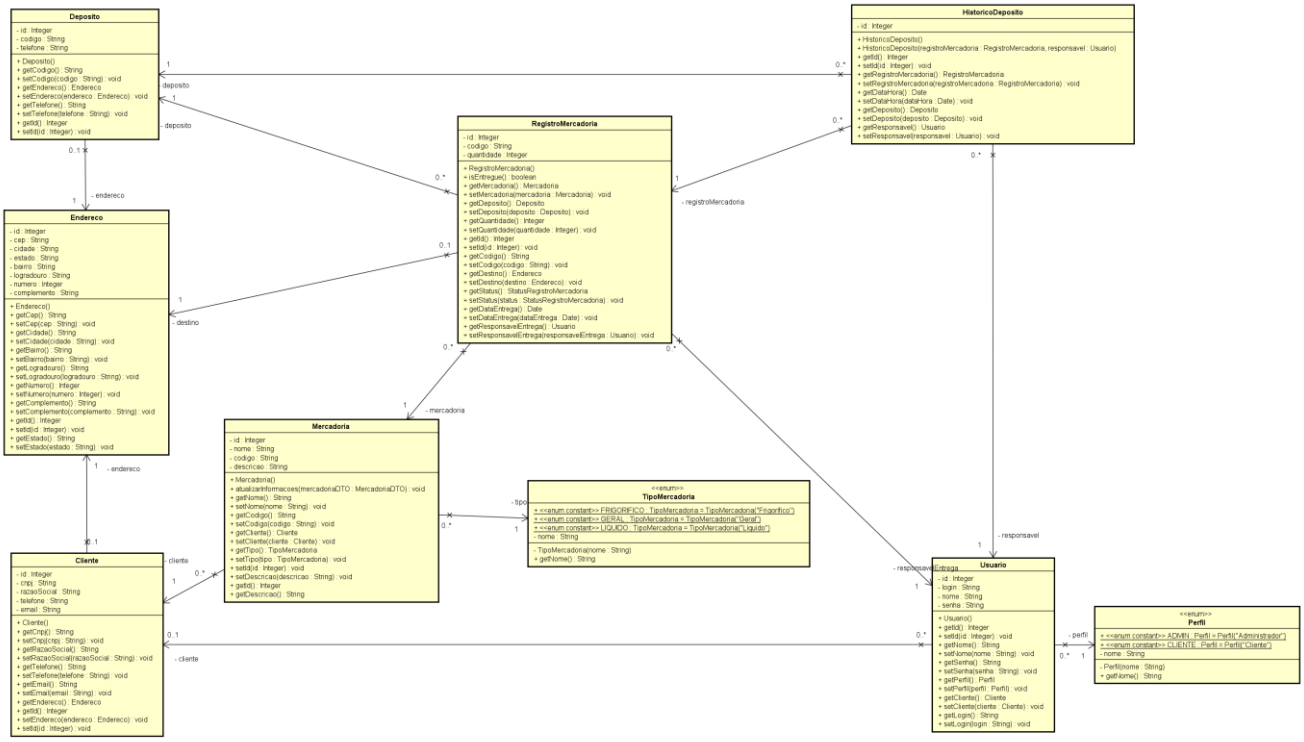

Figura 2 – Diagrama de classes.

A figura 2 apresenta as classes do módulo de informações cadastrais. Nele um cliente cadastrado pode possuir um ou mais usuários vinculados para ter acesso ao sistema. Também é possível que um usuário seja interno da empresa Boa Entrega, tendo perfil de administrador do sistema. São armazenadas informações de mercadorias, depósitos e registros de mercadorias em depósitos, assim como o histórico de localização desses registros. A aplicação também possui o endereço de clientes, de depósitos e o endereço de destino dos registros de mercadorias. O tipo da mercadoria é uma informação relevante para o negócio da empresa que pode permitir a análise interna do melhor meio de transporte entre os depósitos.

3.3.2 Diagrama de Componentes

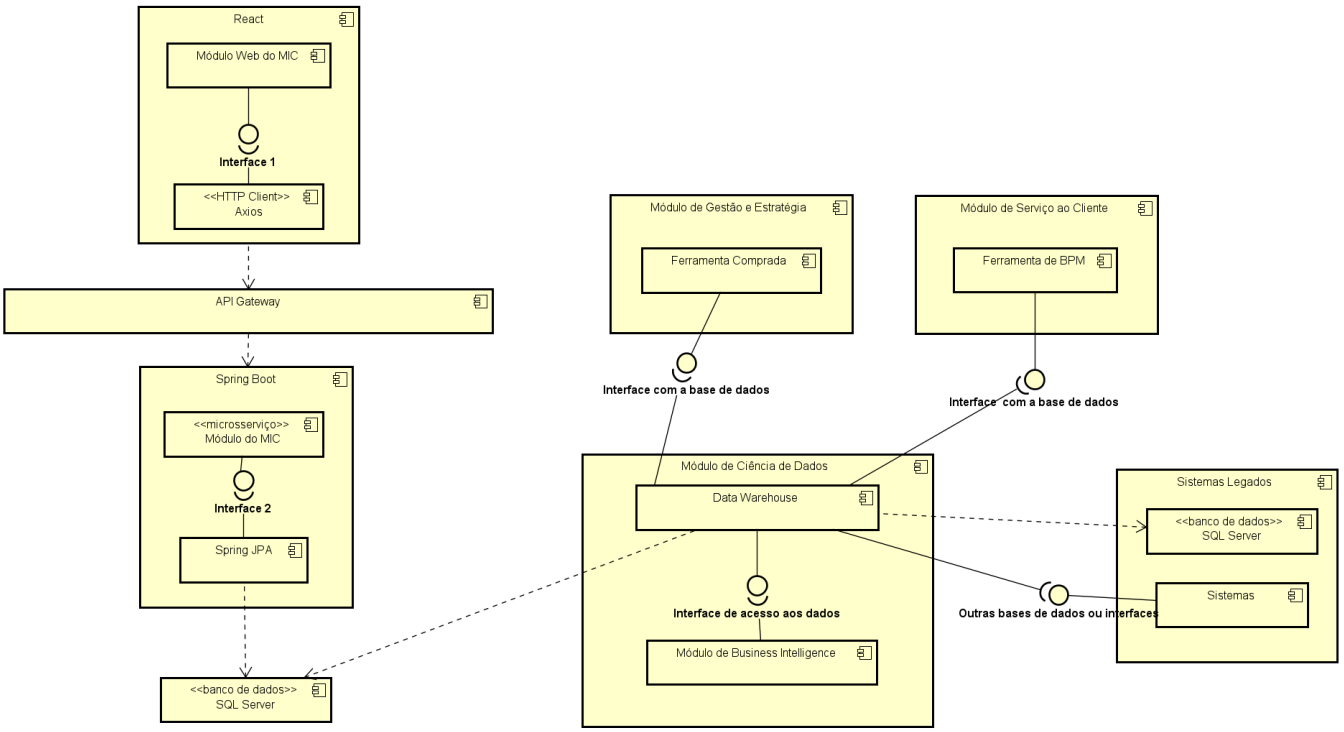

Figura 2 – Diagrama de Componentes.

De acordo com o diagrama apresentado na Figura 3, as entidades participantes da solução são:

- React – o fronted do módulo de informações cadastrais será desenvolvido na tecnologia React. Esta tecnologia possui uma biblioteca chamada Axios, que disponibiliza um cliente HTTP que será utilizado para realizar as requisições da aplicação. Essa tecnologia foi escolhida pois utiliza a linguagem Javascript, também muito utilizada nos sistemas legados e que permite implementar com facilidade o padrão single-page application para as páginas web. Este padrão possui características que contribuem para a performance da aplicação, carregando as páginas web de forma assíncrona e exibindo as informações aos poucos sem que seja preciso esperar que tudo seja carregado para começar a visualizá-las.

- API Gateway – configura os endpoints para as chamadas das APIs disponíveis nos microsserviços do backend em um padrão único de acesso. Este padrão foi escolhido por organizar o acesso ao backend através de uma interface, diminuindo o acoplamento entre as camadas. Deste modo, não é necessário ter conhecimento prévio das tecnologias utilizadas nas outras partes da arquitetura e a utilização de diferentes linguagens de programação é facilitada. Também é possível através da ferramenta configurar mecanismos de segurança, monitorar o acesso e armazenar métricas das transmissões de dados.

- Spring Boot – o banckend do módulo de informações cadastrais será composto por vários microserviços na linguagem Java com o framework Spring Boot. A tecnologia foi escolhida por ser robusta e com as mais diversas bibliotecas e frameworks que facilitam a implementação de funcionalidades como a persistência de dados (Spring JPA/Hibernate), a segurança e a autenticação de usuários (Spring Security).

- SQL Server – banco de dados escolhido para o módulo de informações cadastrais. Foi identificado que os sistemas legados da Boa Entrega também possuem uma base de dados em SQL Server e a escolha do mesmo tipo de banco de dados pode ser uma característica a facilitar a troca de informações entre os módulos. Outros bancos de dados possuem tipos diferentes de dados e de sintaxes de SQL que podem gerar trabalho ao ser necessário realizar a conversão entre os dados.

- Módulo de serviço ao cliente – este módulo deverá ser composto de uma solução de BPM. Por já existirem ferramentas conhecidas e disponíveis no mercado de modelagem de processos foi definido que este modulo também será composto de uma ferramenta adquirida de terceiros no mercado. Esta proposta de arquitetura sugere a utilização da ferramenta Bizagi que é uma das mais conhecidas e utilizadas. Ela apresenta vários recursos de BPM e de integração com outros sistemas, como conexão com serviços web ou Rest, conexão com bancos de dados e integração com sistemas de SAP e ECM. Estes recursos se tornam pontos importantes a serem levados em consideração na escolha da ferramenta ao analisarmos a arquitetura como um todo e pensarmos na integração deste módulo com os demais.

- Módulo de gestão e estratégia – como já foi predefinido, este módulo será composto de uma ferramenta de gestão corporativa adquirida no mercado. Para esta solução foi escolhida a ferramenta Zoho Analytics que tem como foco a criação de painéis de indicadores de desempenho e tem ferramentas de integração com bancos de dados, arquivos e aplicativos. Outro ponto interessante são os aplicativos mobile para iOS e Android que a plataforma disponibiliza de forma gratuita.

- Sistemas legados – além de uma base de dados SQL Server esses sistemas podem possuir outros tipos de dados disponíveis que deverão ser integrados no data warehouse do módulo de ciência de dados.

- Módulo de ciência de dados – é composto por um data warehouse e um módulo de bussiness intelligence. O data warehouse será configurado por uma ferramenta que possibilite a coleta de diferentes tipos e formatos de dados. Esta solução propõe a utilização da ferramenta Amazon Redshift que consegue integrar com os outros módulos e que já possui aplicativos de BI e análise de dados integrados.

3.3.3 Diagrama de Implantação

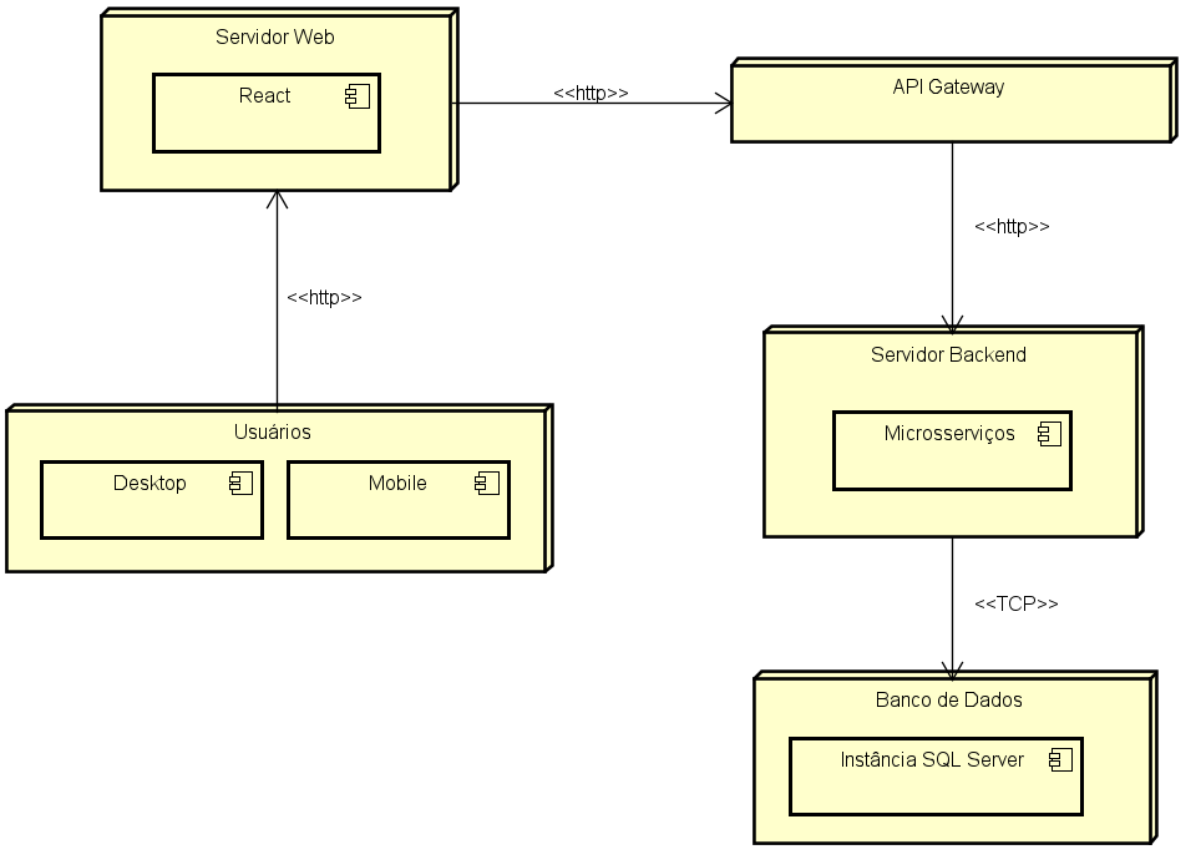

Figura 4 – Diagrama de Implantação.

A figura 4 apresenta o diagrama de implantação da parte desenvolvida para a prova de conceito do módulo de informações cadastrais. A implantação foi feita utilizando ferramentas gratuitas da AWS. O diagrama é constituído dos seguintes componentes:

- Banco de dados – foi criada uma instância do banco de dados SQL Server com a ferramenta Amazon Relational Database Service (Amazon RDS).
- Servidor backend – todos os microsserviços desenvolvidos foram disponibilizados utilizando o serviço AWS Elastic Beanstalk que cria e mantêm em execução instâncias dos serviços. Esta ferramenta apresenta opções que permitem habilitar o balanceamento de carga e a criação de várias instâncias para uma única aplicação. Entretanto, essas opções não estão disponíveis de forma gratuita, estando a prova de conceito limitada a utilizar apenas uma instância.
- API Gateway – todos os endpoints necessários para o funcionamento da aplicação foram configurados em uma API Gateway.

- Servidor Web – a aplicação frontend desenvolvida em React foi implantada através do serviço AWS Amplify voltado para a hospedagem rápida de aplicativos web.

- Usuários – usuários que acessam o sistema por dispositivos de tipos diferentes, como o mobile e o desktop.

Para este trabalho a implantação foi feita de forma simples e rápida anexando arquivos em formato jar para o backend e em zip para o frontend. Entretanto, todas as ferramentas também têm funcionalidades para integração contínua que devem ser configuradas com a evolução do projeto. É possível realizar integrações com o GitHub e montar arquivos de configuração tipo YAML para o deploy das aplicações.

3.3.4 Modelo de Dados

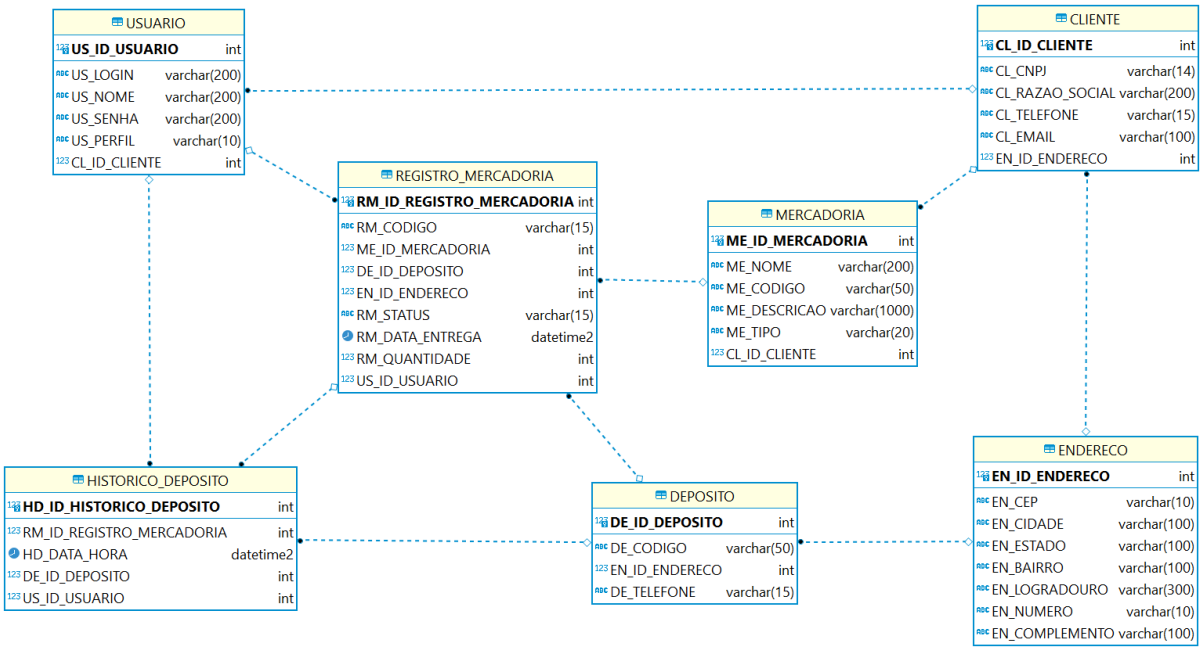

Figura 5 – Tabelas do banco de dados SQL Server.

A figura 5 mostra as tabelas presentes no banco de dados SQL Server utilizado na aplicação. No diagrama as chaves primárias são as primeiras a aparecerem em cada tabela e estão escritas em negrito. Já as chaves estrangeiras, são representadas pelos relacionamentos e pelos atributos que possuem o mesmo nome das chaves primárias das tabelas relacionadas.

4. Prova de Conceito (POC) e Protótipo Arquitetural

Esta seção do trabalho apresenta a POC desenvolvida com foco no módulo de informações cadastrais, visando atender aos requisitos especificados nas seções anteriores deste documento. O código fonte completo da aplicação está disponível

no repositório do GitHub através do link presente na seção de apêndices no final deste documento.

4.1. Implementação

Para a implementação da POC foram desenvolvidas uma aplicação para o backend e outra para o frontend. O backend foi desenvolvido utilizando Java 8 com Spring Boot e a sua estrutura pode ser observada na figura a seguir.

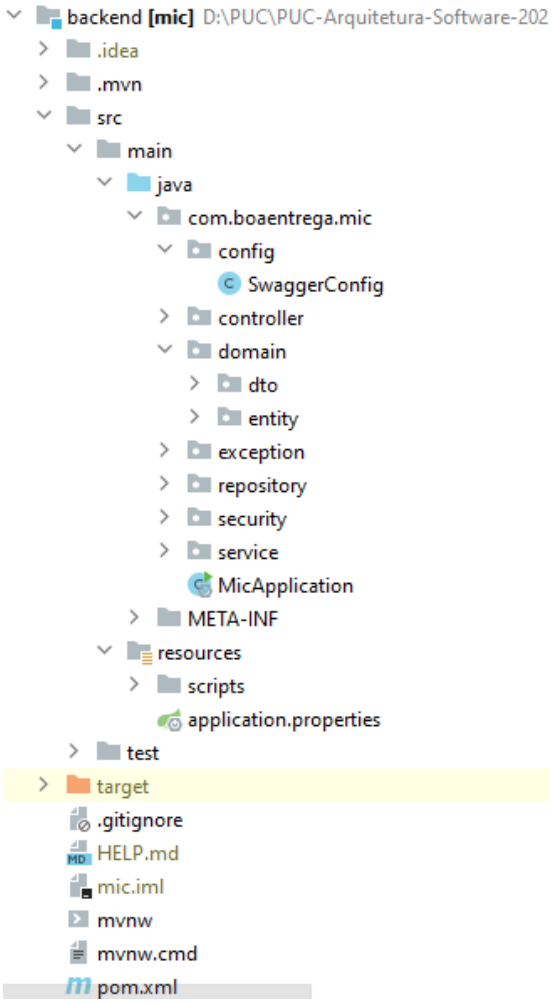

Figura 6 – Estrutura do código do serviço backend.

Os diretórios contém os seguintes conteúdos:

- config – classes de configuração de frameworks, como o Swagger UI.
- controller – classes com as definições das chamadas rest utilizando anotações do Spring.
- domain – classes do domínio da aplicação. A pasta “dto” apresenta os objetos utilizados para realizar as transferências de dados nas APIs e a pasta “entity” contém as classes que mapeiam as tabelas do banco de dados.
- exception – classes de exceções específicas da aplicação.

- repository – classes de interfaces de acesso ao banco de dados para consultas e para a persistência de dados.
- security – classes de configuração de autenticação usando Spring Security.
- service – implementação de classes de serviços da aplicação.
- scripts – contém os scripts de criação das tabelas do banco de dados e de inserção de uma massa inicial de dados.
- test – classes de testes unitários da aplicação utilizando o framework JUnit.

A aplicação do frontend, foi desenvolvida utilizando a tecnologia React e o projeto foi organizado de acordo com a estrutura a seguir.

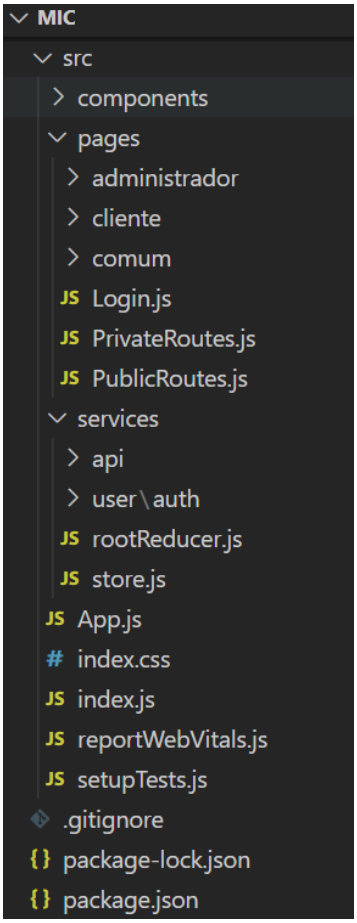

Figura 7 – Estrutura do código da aplicação frontend.

- componentes – componentes React desenvolvidos para serem exibidos em páginas da aplicação.
- pages – páginas da aplicação separadas pelos perfis de administrador e de cliente, contendo uma pasta de páginas em comum e a definição de rotas da aplicação utilizando o framework React Router.
- services – arquivos que acessam os serviços backend através do framework Axios para realizar as operações necessárias.

O frontend foi implantado com a ferramenta AWS Amplify e pode ser acessado pelo endereço da aplicação disponível na seção de apêndices. Através da URL é possível acessar o módulo de informações cadastrais onde serão testados e avaliados os casos de uso a seguir que procuram atender a alguns requisitos não funcionais.

- **Caso de uso UC01 – Atualizar informações cadastrais**

  Requisitos não funcionais: este caso de uso procura testar os requisitos RNF01, ser acessível em plataformas web e mobile, e o RNF04, que trata sobre a responsividade do sistema.

  Critério de aceitação: para atender esses requisitos o sistema deve poder ser acessado de um dispositivo móvel e possuir comportamento responsivo. Todas as informações devem ser visíveis e todas as ações devem estar disponíveis para o usuário.

  Descrição da execução do teste: para o teste desse caso de uso foi utilizada a ferramenta do desenvolvedor disponível no navegador Edge que permite simular um disposivo Android Samsung Galaxy S8+. Primeiramente é realizado o login do usuário com o perfil de cliente com as seguintes credenciais:

  Login: joao

  Senha: 4321

  Após o login é acessada a tela de informações do usuário pelo link de “minha conta” disponível no menu superior. Conforme a Figura 8, todos os elementos da tela são organizados de forma responsiva. É possível ler todas as informações do cliente logado e clicar no botão de editar. Ao clicar no botão o sistema habilita os campos que podem ser modificados e salvos com facilidade pelo usuário.

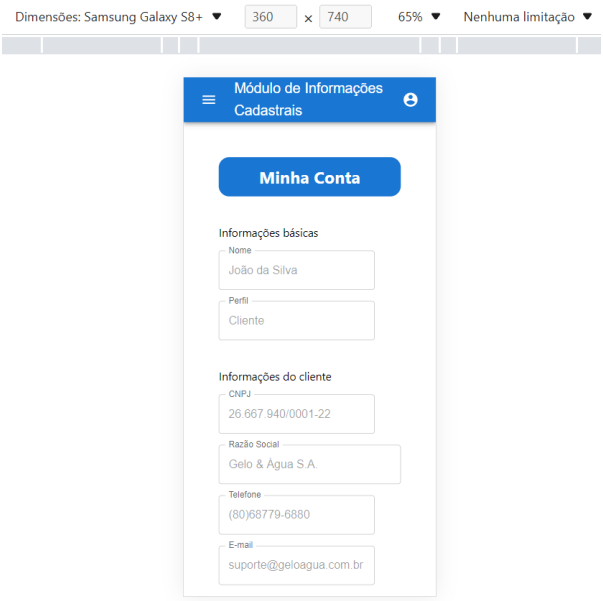

Figura 8 – Tela de informações do usuário logado em um dispositivo mobile.

- **Caso de uso UC02 – Movimentar registro de mercadoria**

  Requisitos não funcionais: este caso de uso tem como objetivo testar o requisito RNF03 que diz respeito ao desempenho do sistema e o RN05 de usabilidade.

  Critério de aceitação: para atender este requisito o sistema deve completar todas as suas requisições no tempo de até 4 segundos e o usuário deve conseguir completar a operação em até 5 etapas de distância da tela inicial.

  Descrição da execução do teste: o primeiro passo é fazer o login com o usuário administrador do sistema que possui as seguintes credenciais:

  Login: admin

  Senha: 1234

  Após o login, é acessada a tela de registro de mercadorias através do menu superior. A tela exibe a lista de registros com os seus códigos identificadores, localização atual, endereço de destino e o status. O status do registro indica se ele acabou de ser registrado, se está em trânsito a caminho do destino ou se a mercadoria já foi entregue. Ao clicar no ícone de “mover mercadoria” em um registro da tabela o usuário é redirecionado para a tela de atualização de localização do registro. A tela apresenta todas as informações do registro, incluindo a mercadoria, o seu tipo e a quantidade. Todos os depósitos cadastrados são exibidos em uma lista para serem selecionados. Através da ferramenta de desenvolvedor do navegador na aba “Rede” é observado o tempo de resposta da aplicação. As requisições feitas de forma assíncrona

respeitam o limite de 4 segundos e são exibidas aos poucos na tela do usuário.

O usuário seleciona um dos depósitos e clica no botão de verificar distâncias. A API do backend consulta um serviço de geolocalização e retorna de forma organizada as distâncias do endereço atual do registro até o depósito escolhido e do depósito escolhido até o endereço de destino. O usuário verifica as distâncias selecionando os demais depósitos e clicando no botão. O sistema cumpre o requisito de desempenho retornando a resposta em menos de 4 segundos, mesmo simulando uma conexão do tipo 3G lenta. Por fim, o usuário clica no botão atualizar que retorna para tela de listagem e atualiza o endereço do registro. Todo o fluxo é completado em menos de 5 etapas, atingindo o requisito de usabilidade.

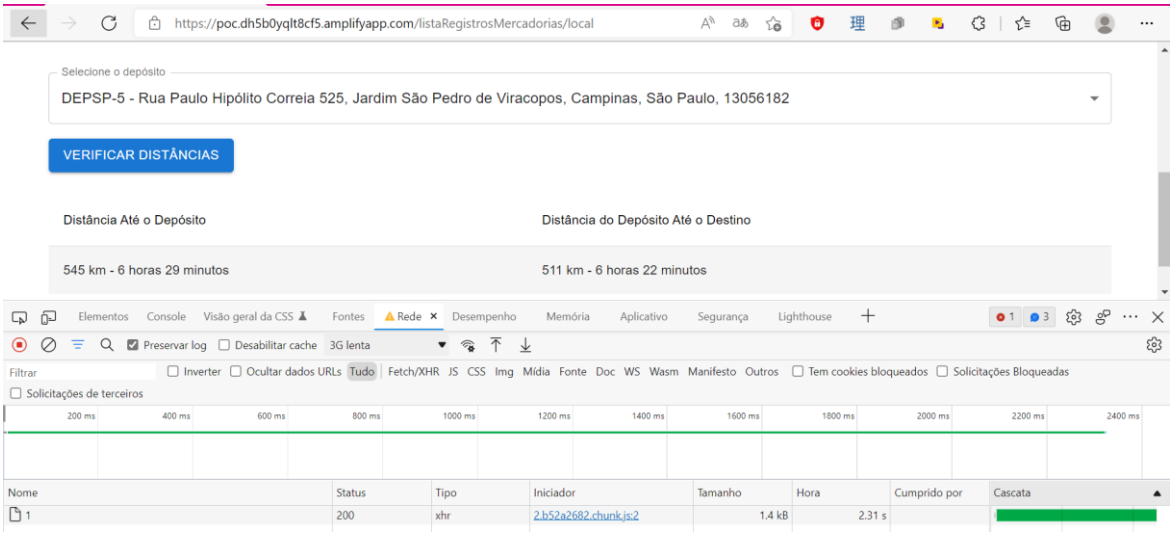

Figura 9 – O sistema retorna a resposta em menos de 4 segundos simulando uma conexão lenta.

- **Caso de uso UC03 – Obter histórico de localização**

Requisitos não funcionais: este caso de uso testa os requisitos RNF02 de interoperabilidade, o RNF07 de segurança.

Critério de aceitação: o sistema deve possibilitar o acesso as informações de forma simples e transparente, utilizando mecanismos que garantam que só pessoas autorizadas no sistema possam realizar as operações.

Descrição da execução do teste: para o teste primeiramente foi utilizada a ferramenta postman para fazer as chamadas para os endereços disponíveis na API Gateway. Primeiramente é necessário chamar o endpoint de autenticação do usuário do sistema através do protocolo HTTP POST que retorna o token JWT conforme a figura abaixo.

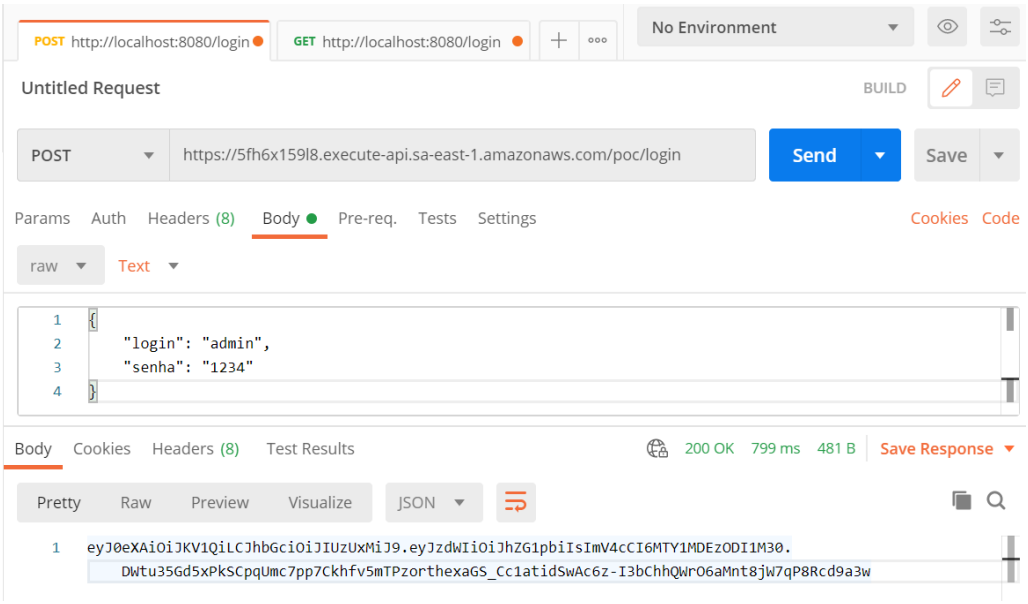

Figura 10 – Autenticação do usuário através do protocolo HTTP POST.

Para poder utilizar os demais endpoints da API é preciso utilizar o token JWT como forma de autenticação. Caso o token não seja fornecido é retornado erro de código 403, que indica que a operação não foi autorizada. Desta forma, as informações não são concedidas de forma indevida.

Utilizando o token, finalmente é feita a chamada ao endpoint que retorna as informações de histórico de localização de um registro de mercadoria. A API retorna as informações em formato JSON que pode ser lida através de diferentes tecnologias.

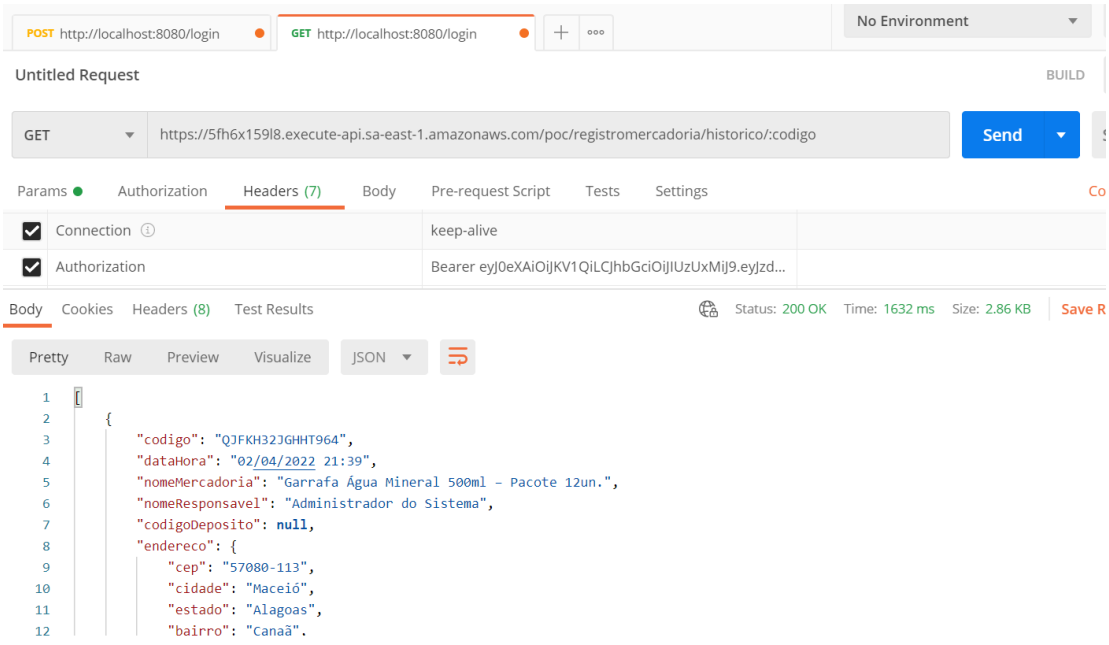

Figura 11 – Chamada a API de histórico de localização de registro de mercadoria.

Um teste similar pode ser feito na aplicação web. Ao acessar uma url interna sem estar autenticado o sistema verifica a ausência do token e redireciona o usuário para a tela de login.

4.2 Interfaces e APIs

A aplicação desenvolvida possui APIs Rest que foram documentadas utilizando Swagger. A figura 13 apresenta alguns dos endpoints disponíveis. A documentação completa com os demais endpoints pode ser acessada através do link http://micbackend-env-1.eba-mermtpj8.sa-east-1.elasticbeanstalk.com/swagger-ui/index.html.

Figura 12 – APIs Rest.

5. Avaliação da Arquitetura

A avaliação da arquitetura desenvolvida e implementada é tratada nesta seção, com o objetivo de analisar se ela atende ao que foi proposto neste trabalho.

5.1. Análise das abordagens arquiteturais

Esta seção apresenta um resumo das características da proposta arquitetural considerando o método Architecture Tradeoff Analysis Method (ATAM), no qual são utilizados cenários para esta análise.

Atributos de Quailidade	Cenários	Importância	Complexidade
Portabilidade	Cenário 1: o sistema deve ser acessível através de dispositivos móveis.	A	M
Interoperabilidade	Cenário 2: o sistema deve possuir boa interoperabilidade.	A	M
Desempenho	Cenário 3: o sistema deve possuir bom desempenho.	A	A
Responsividade	Cenário 4: o sistema deve possuir interface responsiva.	M	B
Usabilidade	Cenário 5: o sistema deve ser de fácil uso pelos usuários.	M	M
Segurança	Cenário 6: o sistema deve possuir mecanismos de autenticação.	A	A

4.2. Cenários

Para o teste da aplicação foram utilizados os seguintes cenários:

- Cenário 1 – Portabilidade: Ao tentar utilizar o sistema através de um dispositivo móvel, ele apresenta o funcionamento adequado.
- Cenário 2 – Interoperabilidade: Ao realizar uma chamada a API via protocolo HTTP GET é obtida uma resposta no formato JSON.
- Cenário 3 – Desempenho: Ao acessar uma tela com várias informações, as chamadas são realizadas de forma assíncrona e os componentes são carregados aos poucos. Todas as requisições respeitam o limite de 4 segundos de resposta à aplicação web.
- Cenário 4 – Responsividade: Ao mudar o tamanho da tela o sistema se adequa continuando a exibir todos os elementos da tela, sendo compreensível e utilizável pelo usuário.
- Cenário 5 – Usabilidade: Ao navegar pelo sistema, o usuário procura acessar a funcionalidade de alterar a localização de registro de mercadoria sem ter conhecimento de como estão dispostas as informações. Ao acessar o menu superior na tela inicial deve ser possível encontrar e utilizar a funcionalidade em até 5 etapas.
- Cenário 6 – Segurança: Ao tentar acessar um endpoint interno do sistema sem estar autenticado, o sistema retorna erro e não permite o acesso as informações.

5.3. Evidências da Avaliação

- Cenário de portabilidade.

Atributo de Qualidade:	Portabilidade
Requisito de Qualidade:	O sistema deve ser acessível nas plataformas web e móvel.
Preocupação:	
O sistema deve poder ser acessado e utilizado através de diferentes tipos de dispositivos.	
Cenário(s):	
Cenário 1	
Ambiente:	
Sistema em operação normal	
Estímulo:	
Usuário acessa o sistema através de um dispositivo web e depois acessa por um dispositivo móvel.	
Mecanismo:	
Disponibilizar o acesso a aplicação através do navegador de dispositivos móveis sem restrições.	
Medida de resposta:	
Retorna a aplicação com todas as suas funcionalidades disponíveis.	
Considerações sobre a arquitetura:	
Riscos:	Não há.
Pontos de Sensibilidade:	Não há.
Tradeoff:	Não há.

- Evidências da avaliação.

  A primeira tela apresenta a aplicação acessada pela web, já a segunda exibe a simulação do acesso através de um dispositivo móvel.

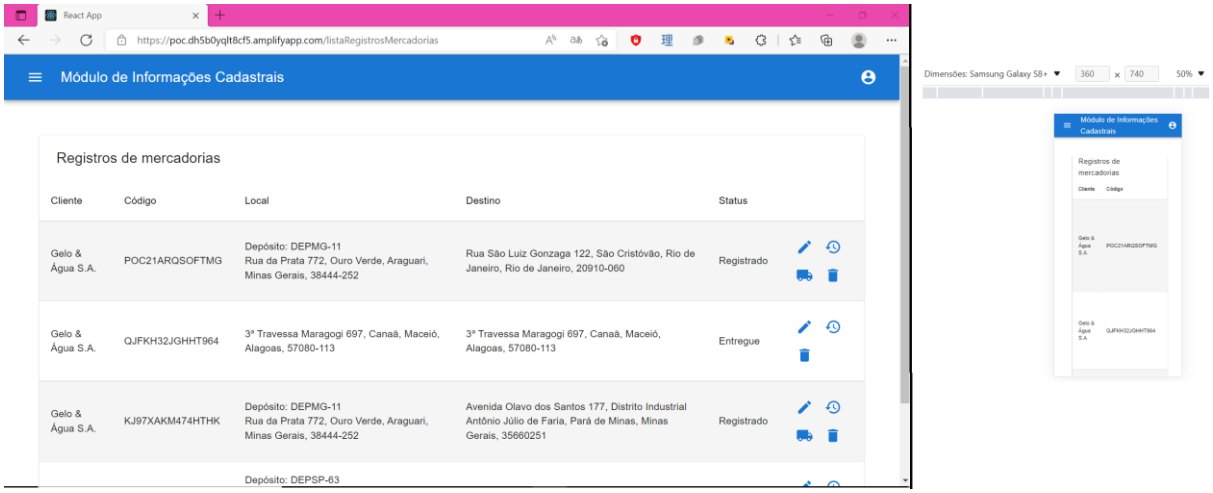

Figura 13 – Aplicação sendo acessada através da web e por dispositivo móvel.

- Cenário de interoperabilidade.

Atributo de Qualidade:	Interoperabilidade
Requisito de Qualidade:	O sistema deve possuir boa interoperabilidade, sendo possível acessar e enviar informações de forma transparente.
Preocupação:	
Disponibilizar a troca de informações de forma padronizada e transparente entre aplicações que utilizem tecnologias variadas.	
Cenário(s):	
Cenário 2	
Ambiente:	
Sistema em operação normal	
Estímulo:	
Acesso a API Gateway através do protocolo HTTP GET.	
Mecanismo:	
Criar uma API Gateway para através de uma interface única atender e redirecionar às requisições aos microsserviços.	
Medida de resposta:	
Retorno da resposta padronizada no formato JSON.	
Considerações sobre a arquitetura:	
Riscos:	Aumento no tempo de resposta da aplicação.
Pontos de Sensibilidade:	A alta quantidade de chamadas simultâneas e

	a sua má orquestração podem vir a sobrecarregar a API Gateway criando um ponto de gargalo.
Tradeoff:	A utilização da API Gateway pode ocasionar aumento no tempo de resposta da aplicação afetando o requisito de desempenho. Foi decidido que a solução proposta apresentaria este risco já que o requisito de interoperabilidade é um dos mais importantes considerando a necessidade de a arquitetura ser de baixo acoplamento como o especificado nas restrições arquiteturais.

- Evidências da avaliação.

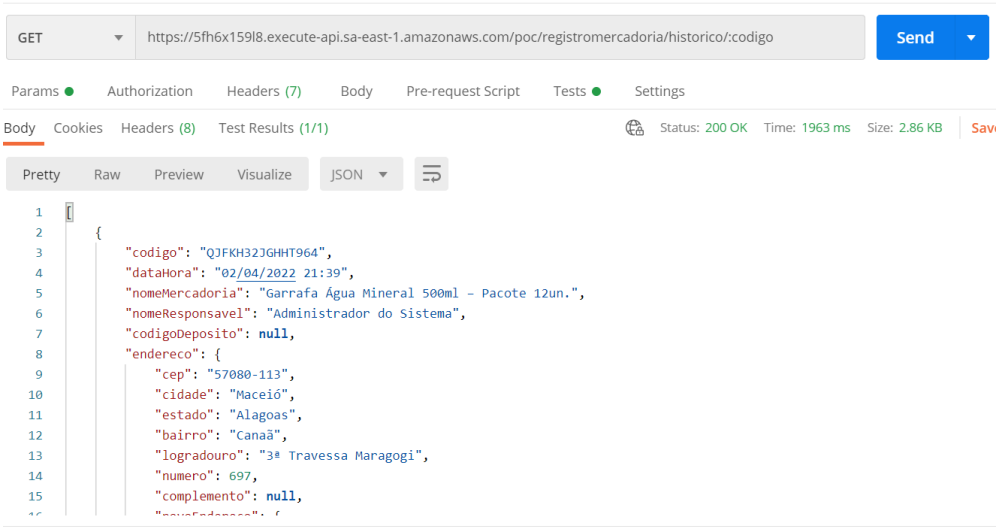

Figura 14 – Resposta em Json ao chamar um endpoint configurado na API Gateway.

- Cenário de desempenho.

Atributo de Qualidade:	Desempenho
Requisito de Qualidade:	O sistema deve apresentar bom desempenho, com tempo de resposta de até 4 segundos para cada requisição.
Preocupação:	
O sistema deve responder as requisições com um tempo adequado de forma que a sua	

navegação seja agradável aos usuários.	
Cenário(s):	
Cenário 3	
Ambiente:	
Sistema em operação normal	
Estímulo:	
Usuário entra em uma tela do sistema web que realiza mais de uma requisição.	
Mecanismo:	
A aplicação web processa as chamadas de forma assíncronas e utiliza o padrão single-page application. Ao passar as informações de um componente a outro, apenas recarregando os componentes alterados na tela, este padrão consegue diminuir a quantidade de requisições feitas pela aplicação.	
Medida de resposta:	
As respostas das requisições são obtidas em até 4 segundos e os componentes da tela são carregados separadamente.	
Considerações sobre a arquitetura:	
Riscos:	Pode existir uma demanda alta nos servidores causando um aumento no tempo de resposta.
Pontos de Sensibilidade:	Falha no funcionamento do balanceamento de carga nos microsserviços.
Tradeoff:	Não há.

- Evidências da avaliação.

  A imagem abaixo contém o tempo de resposta das requisições realizadas na tela da aplicação.

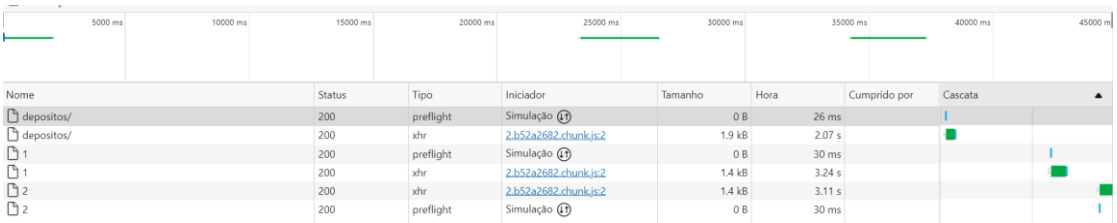

Figura 15 – Tempo das requisições na tela da aplicação web.

- Cenário de responsividade.

Atributo de Qualidade:	Responsividade
Requisito de Qualidade:	O sistema deve possuir interface responsiva.
Preocupação:	
Todas as informações presentes no sistema devem poder ser visualizadas independente do tamanho da tela do dispositivo utilizado para acessar o sistema.	
Cenário(s):	
Cenário 4	
Ambiente:	
Sistema em operação normal	
Estímulo:	
Usuário modifica o tamanho da tela em que ele está acessando o sistema.	
Mecanismo:	
Estilos de CSS aplicados na página web que tratam a disposição dos seus elementos.	
Medida de resposta:	
Tela do sistema com todas as informações visíveis.	
Considerações sobre a arquitetura:	
Riscos:	Não há.
Pontos de Sensibilidade:	Não há.
Tradeoff:	Não há.

- Evidências da avaliação.

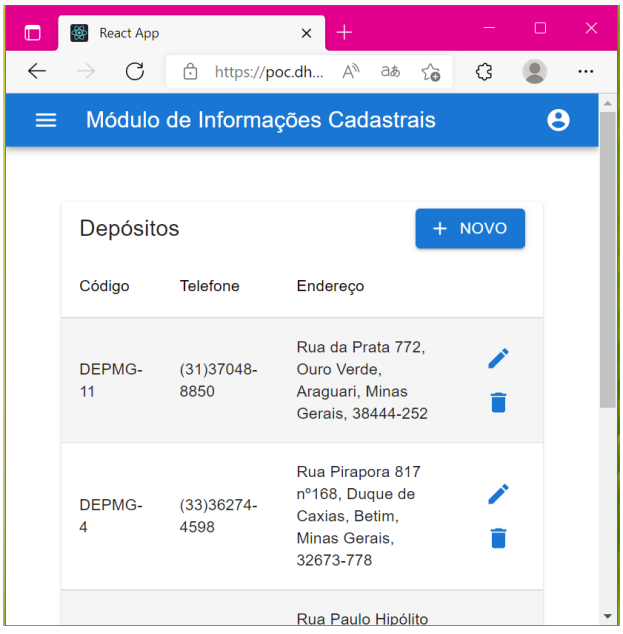

Figura 16 – Tela do sistema minimizada com as informações em forma responsiva.

- Cenário de usabilidade.

Atributo de Qualidade:	Usabilidade
Requisito de Qualidade:	O sistema deverá ter boa usabilidade, sendo necessário no máximo até 5 etapas para atingir o objetivo final do usuário partindo da tela inicial.
Preocupação:	
O sistema deve prover uma navegação clara com facilidade de encontrar e executar as suas operações.	
Cenário(s):	
Cenário 5	
Ambiente:	
Sistema em operação normal	
Estímulo:	
Usuário tenta encontrar e utilizar a funcionalidade de alteração de localização de registro de mercadoria partindo da tela inicial do sistema.	
Mecanismo:	
Utilização de um menu superior com títulos claros.	
Medida de resposta:	
O usuário realiza a operação em quatro etapas no sistema.	
Considerações sobre a arquitetura:	
Riscos:	Não há.
Pontos de Sensibilidade:	Não há.
Tradeoff:	Não há.

- Evidências da avaliação.

  De acordo com a figura a seguir, o usuário conclui a operação desejada sendo necessário realizar as etapas:

  1. Encontrar e clicar no menu de “Registros de Mercadorias”.
  2. Encontrar o registro desejado na tabela e clicar no botão de alterar a localização.
  3. Selecionar o depósito desejado e clicar no botão de “Atualizar”.

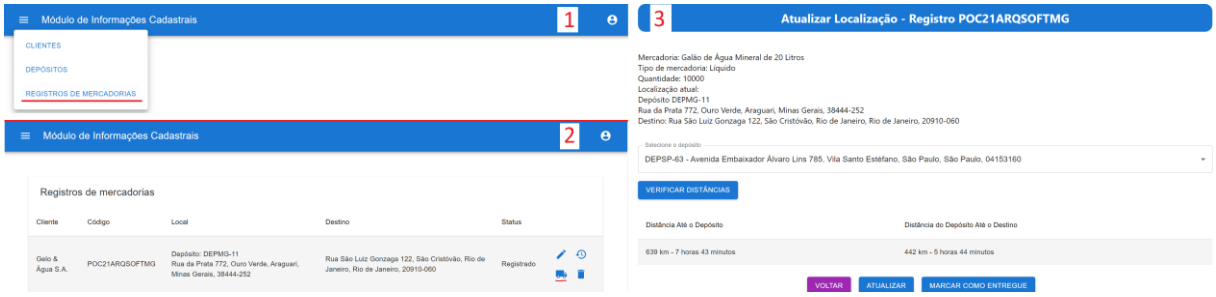

Figura 17 – Usuário realiza a operação de atualizar localização de registro.

- Cenário de segurança

Atributo de Qualidade:	Segurança
Requisito de Qualidade:	O sistema deve ser seguro, permitindo o acesso as informações apenas de forma autenticada.
Preocupação:	
O sistema deve manter a segurança controlando o acesso as informações sensíveis que a empresa armazena.	
Cenário(s):	
Cenário 6	
Ambiente:	
Sistema em operação normal	
Estímulo:	
Usuário tentar realizar chamada ao serviço backend sem ter sido autenticado.	
Mecanismo:	
É requisitado o envio da autenticação do usuário através de um token JWT.	
Medida de resposta:	
A aplicação retorna uma mensagem de erro indicando que a chamada realizada não é permitida.	
Considerações sobre a arquitetura:	
Riscos:	As credenciais de acesso ao sistema devem ser bem protegidas, caso o contrário as informações da empresa podem se tornar vulneráveis. Podem existir vulnerabilidades na criação e transmissão do token.
Pontos de Sensibilidade:	Credenciais de autenticação e geração do

	token JWT.
Tradeoff:	Não há.

- Evidências da avaliação.

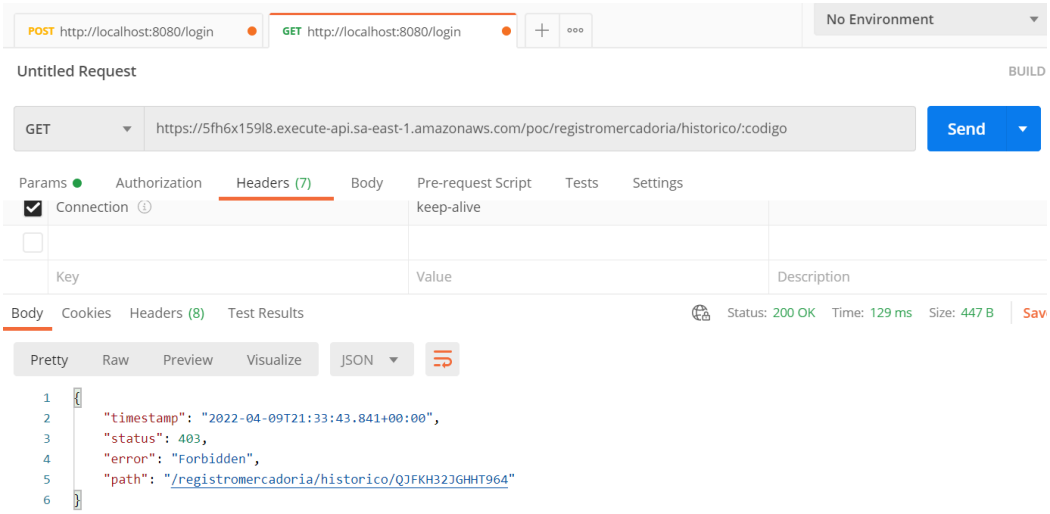

Figura 18 – Resposta da chamada retorna com erro ao não ser fornecido o token JWT.

5.4. Resultados

Conforme os resultados apresentados nas evidências da seção anterior, serão explorados os pontos fortes e as limitações existentes na solução da arquitetura.

Requisitos Não Funcionais	Testado	Homologado
RNF01: O sistema deve ser acessível nas plataformas web e móvel.	SIM	SIM
RNF02: O sistema deve possuir boa interoperabilidade, sendo possível acessar e enviar informações de forma transparente.	SIM	SIM
RNF03: O sistema deve apresentar bom desempenho, com tempo de resposta de até 4 segundos para cada requisição.	SIM	SIM
RNF04: O sistema deve possuir interface responsiva	SIM	SIM
RNF05: O sistema deverá ter boa usabilidade, sendo necessário no máximo até 5 etapas para atingir o objetivo final do usuário partindo da tela inicial.	SIM	SIM
RNF07: O sistema deve ser seguro, permitindo o acesso as in-	SIM	SIM

formações apenas de forma autenticada.		

A tabela contém os requisitos testados na aplicação na sessão anterior. Os resultados dos testes demonstraram que a solução proposta para a arquitetura atende bem aos requisitos de portabilidade e responsividade. Entretanto enfatizo que deverão ser criados padrões de estilos e configurações para garantir que todas as partes da aplicação web atendam a esses requisitos. Ao ter um sistema que é utilizado e modificado por muito tempo por várias pessoas é comum que ocorram problemas na qualidade do desenvolvimento. Portanto, esses padrões deverão ser sólidos e bem definidos pela empresa desde o início. As mesmas observações se aplicam ao requisito de usabilidade.

Para os objetivos definidos para a prova de conceito o requisito de segurança foi atendido. Sendo um requisito complexo, existirão ajustes a serem feitos com a evolução do projeto. Uma melhoria possível de ser feita é a configuração mais restrita de endereços de IPs e tipos de protocolos que podem ser utilizados para acessar os endpoints da aplicação. Este tipo de configuração é simples de ser implementado diretamente na ferramenta de API Gateway.

O padrão API Gateway contribuiu para garantir a interoperabilidade no acesso aos novos serviços, porém a arquitetura ainda pode apresentar limitações ao precisar utilizar partes dos sistemas legados. Este componente também pode se tornar um gargalo na arquitetura se não for bem orquestrado. Sobre o requisito de desempenho, pelos testes realizados o tempo limite de resposta das requisições não foi ultrapassado. Como pontos fortes a solução utiliza tecnologias que possibilitam a sua otimização e que futuramente poderão ser mais explorados.

6. Conclusão

Concluo que os objetivos gerais e específicos deste trabalho foram alcançados visto que a solução da arquitetura foi concluída com foco no módulo de informações cadastrais. A avaliação feita demonstra que ela oferece possibilidades de integrações com outros sistemas e atende a requisitos de usabilidade e portabilidade que farão com que os usuários tenham facilidade de uso.

Durante o desenvolvimento do trabalho encontrei limitações nos testes e consequentemente nos resultados da avaliação devido as limitações existentes na infraestrutura e por poder utilizar apenas a camada gratuita dos serviços da Amazon. Gostaria de ter explorado de forma mais ampla o requisito de desempenho simulando a cria-

ção de várias instâncias do serviço. Entendo que isto pode se tornar um ponto fraco na solução e que poderá ocorrer um gargalo na comunicação dos serviços, na orquestração da API Gateway e no acesso ao banco de dados.

Pelos testes realizados e pela minha experiência como desenvolvedora, em que utilizo tecnologias mais antigas como o JSF, pude perceber na prática como tecnologias mais atuais como o React são mais rápidas e eficientes. Considero um dos pontos fortes da solução.

Por fim, devido aos resultados apresentados acredito que a solução proposta apresenta possibilidades de ser bem utilizada e evoluída com o tempo.

REFERÊNCIAS

NOVAREJO. E-commerce: o setor que cresceu 75% em meio à pandemia, 2021. Disponível em: <https://www.consumidormoderno.com.br/2021/02/19/e-commerce-setor-cresceu-75-crise-coronavirus/>. Acesso em: 10 de abril de 2022.

Economia SC. Empresa de logística cresce mais de 600% durante a pandemia, 2020. Disponível em: <https://economiasc.com/2020/10/12/empresa-de-logistica-cresce-mais-de-600-durante-a-pandemia/>. Acesso em: 10 de abril de 2022.

APÊNDICES

URL do repositório do código fonte: https://github.com/KarinaMz/PUC-Arquitetura-Software-2021

URL do vídeo no youtube: https://youtu.be/vp76M551vrU

URL da aplicação web: https://poc.dh5b0yqlt8cf5.amplifyapp.com/

URL da API Gateway: https://5fh6x159l8.execute-api.sa-east-1.amazonaws.com/poc/

URL da documentação das APIs rest com Swagger: http://micbackend-env-1.eba-mermtpj8.sa-east-1.elasticbeanstalk.com/swagger-ui/index.html

CHECKLIST PARA VALIDAÇÃO DOS ITENS E ARTEFATOS DO TRABALHO

(OPCIONAL)

Nº	Item a ser cumprido	Sim	Não	Não se aplica
Completeza do documento				
1	Todos os elementos iniciais do documento (capa, contracapa, resumo, sumário...) foram definidos?	X		
2	Os objetivos do trabalho (objetivos gerais e pelo menos três específicos) foram especificados?	X		
3	Os requisitos funcionais foram listados e priorizados?	X		
4	Os requisitos não funcionais foram listados e identificados usando o estilo estímulo-resposta?	X		
5	As restrições arquiteturais foram definidas?	X		
6	Os mecanismos arquiteturais foram identificados?	X		
7	Um diagrama de caso de uso foi apresentado junto com uma breve descrição de cada caso de uso?	X		
8	Um modelo de componentes e uma breve descrição de cada componente foi apresentada?	X		
9	Um modelo de implantação e uma breve descrição de cada elemento de hardware foi apresentada?	X		
10	Prova de conceito: uma descrição da implementação foi feita?	X		
11	Prova de conceito: as tecnologias usadas foram listadas?	X		
12	Prova de conceito: os casos de uso e os requisitos não funcionais usados para validar a arquitetura foram listados?	X		
13	Prova de conceito: os detalhes da implementação dos casos de uso (telas, características, etc) foram apresentadas?	X		
14	Prova de conceito: foi feita a implantação da aplicação e indicado como foi feita e onde está disponível?	X		
15	As interfaces e/ou APIs foram descritas de acordo com um modelo padrão?	X		
16	Avaliação da arquitetura: foi feita uma breve descrição das características das abordagens da proposta arquitetural?	X		
17	Avaliação da arquitetura: Os atributos de qualidade e os cenários onde eles seriam validados foram apresentados?	X		
18	Avaliação da arquitetura: uma avaliação com as evidências dos testes foi apresentada?	X		
19	Os resultados e a conclusão foram apresentados?	X		
20	As referências bibliográficas foram listadas?	X		
21	As URLs com os códigos e com o vídeo da apresentação da POC foram listadas?	X		

Nº	Item a ser cumprido	Sim	Não	Não se aplica
Consistência dos itens do documento				
1	Todos os requisitos funcionais foram mapeados para casos de uso?	X		
2	Todos os casos de uso estão contemplados na lista de requisitos funcionais?	X		
3	Os requisitos não funcionais, mecanismos arquiteturais e restrições c arquiteturais estão coerentes com os modelos de componentes e implantação?	X		
4	Os modelos de componentes e implantação estão coerentes com os requisitos não funcionais, mecanismos arquiteturais e restrições arquiteturais?	X		
5	As tecnologias listadas na implementação estão coerentes com os requisitos não funcionais, mecanismos arquiteturais e restrições arquiteturais?	X		
6	Os casos de uso e os requisitos não funcionais listados na implementação estão coerentes com o que foi listado nas seções anteriores?	X		
7	Os atributos de qualidade usados na avaliação estão coerentes com os requisitos não funcionais na sessão 3?	X		
8	Os cenários definidos estão no contexto dos casos de uso implementados?	X		
9	O apresentado no item resultado está coerente com o que foi mostrado no item avaliação?	X		