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

Felipe Torres da Silva

SISTEMA DE GESTÃO DA QUALIDADE PARA TRABALHOS REMOTOS

Belo Horizonte
2020

Felipe Torres da Silva

SISTEMA DE GESTÃO DA QUALIDADE PARA TRABALHOS REMOTOS

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

Orientador(a): Prof. Dr. Pedro A. Oliveira

Belo Horizonte
2020

Dedico este trabalho a Gabriela que, com muito amor e apoio, não mediu esforços para que eu concluísse mais uma etapa da minha vida.

AGRADECIMENTOS

A elaboração deste trabalho não teria sido possível sem a colaboração, estímulo e empenho de diversas pessoas. Por essa razão gostaria de expressar toda a minha gratidão e apreço a todos aqueles que, direta ou indiretamente, contribuíram para que esta tarefa se tornasse uma realidade. A todos quero manifestar os meus sinceros agradecimentos.

Em primeiro lugar, a Gabriela pela amizade, companheirismo, compreensão, apoio e amor incondicional.

A todos os professores e funcionários da Pontifícia Universidade Católica de Minas Gerais, que, com ensinamentos e orientações, me ajudaram ativa ou passivamente neste projeto.

Desejo igualmente agradecer a todos os meus amigos e familiares cujo apoio e amizade estiveram presentes em todos os momentos.

RESUMO

Naturalmente a descentralização do trabalho como forma de minimizar custos de manutenção de espaços físicos e grandes deslocamentos é uma tendência, muitas vezes tentadora, para grandes e médias empresas que possuem um alto custo operacional. Em vista disso, o sistema proposto contém ferramentas para diminuição dos impactos da descentralização como ferramentas de agendamento de reuniões, videoconferência, *deadline* de eventos importantes e módulos tradicionais para gestão da qualidade como a gestão de documentações e de pessoas, indicadores de desempenho e planejamentos de ações e riscos. Além disso, a arquitetura foi realizada em nuvem e em camadas para facilitar o processo de desenvolvimento e implantação, tendo como forma de validação de requisitos as evidências arquiteturais relacionadas à segurança, manutenção e usabilidade. Este trabalho visa apresentar o projeto arquitetural de um Sistema de Gestão da Qualidade (SGQ) para empresas de pequeno e médio porte com foco na melhoria da comunicação entre seus funcionários de forma descentralizada.

Palavras-chave: arquitetura em nuvem, SGQ, descentralização, arquitetura em camadas.

SUMÁRIO

1. Objetivos do trabalho

Este projeto objetiva apresentar a solução arquitetural de um software para gestão da qualidade para uma empresa de médio ou de pequeno porte. O projeto visa fornecer uma plataforma para facilitar a comunicação para regimes de trabalho descentralizados.

Os objetivos específicos são:

1. Projetar o módulo de colaboradores, melhorando o fluxo de atividades de gestão de documentos de auditoria e relatórios de não conformidades, gestão de pessoas, planejamento de riscos e ações.
2. Projetar o módulo de gestão, permitindo aos gestores realizarem o controle de acesso às informações da empresa por grupos, avaliarem os indicadores obtidos e aprovarem planos, publicação de documentações e contratações;
3. Projetar o módulo de comunicação possibilitando a interação dos membros da empresa por meio de mensagens para avisos de agenda de auditorias ou reuniões, atualização de itens dos outros módulos e reunião por videoconferência;
4. Projetar uma API (*Application Programming Interface*) de integração para exposição de indicadores, com intuito de prover os indicadores obtidos para alimentar outros sistemas, como o ERP (*Enterprise Resource Planning*).

2. Descrição geral da solução

2.1. Apresentação do problema

Em meio a tantas informações e competitividade presentes nos dias atuais, uma empresa que se destaca no mercado é aquela que se mantém focada em aperfeiçoar produtos e processos, como também, na satisfação do cliente.

A necessidade do cumprimento de determinados requisitos para que as características e o processo de produção garantam a satisfação do consumidor, deu origem a ISO9001 que é um Sistema de Gestão da Qualidade (SGQ). O SGQ é uma ferramenta de gestão organizacional que fornece meios e formas de controlar e gerenciar processos. Ela introduz padrões de qualidade na produção tornando as tarefas operacionais mais simples e

contribuindo para o aperfeiçoamento e a qualidade destas. Ela também aumenta a produtividade, uma vez que os profissionais conseguem reduzir os custos no tempo de produção e diminuir as falhas que podem acabar prejudicando a sua lucratividade. Além de melhorar o desempenho da equipe, diferente dos processos manuais o sistema de gestão ajuda a empresa a alcançar os objetivos mais rápidos, garante maior segurança e confiabilidade das informações além de assegurar aos profissionais a direção certa dos processos, a padronização dos dados e também a integração dos setores. Como também aperfeiçoa a tomada de decisões, posto que a tecnologia possibilita gerar relatórios mais eficientes, gerenciar dados e organizar o grande volume de informações. Entre outras características.

Os processos manuais oferecem pouca garantia em relação ao estabelecimento do padrão da qualidade do produto, uma vez que sem o auxílio da tecnologia, as tarefas operacionais se tornam mais complicadas e o controle das informações e resultados se perdem. Em vista disto, o desenvolvimento de um software de gestão da qualidade com base nos requisitos da norma ABNT NBR ISO 9001:2015 proporciona para a empresa maior capacidade para prover produtos e serviços que atendam aos requisitos do cliente e aos requisitos estatutários e regulamentares aplicáveis, como também, proporciona uma maior facilidade para aproveitar oportunidades e garantir o aumento da satisfação do cliente, além da capacidade para abordar riscos e oportunidades associados com o seu contexto e objetivos.

2.2. Descrição geral do software (Escopo)

A elaboração desse software de gestão da qualidade visa fornecer uma plataforma para estender e facilitar a comunicação entre os colaboradores de uma empresa por meio de módulos de auxílio da manutenção da norma ISO9001:2015. O sistema deverá prover ferramentas para a comunicação e segurança de acesso para computadores pessoais.

Os colaboradores poderão acessar a plataforma utilizando dispositivo desktop com acesso à internet via browser de navegação, a qualquer hora do dia, por meio de seu código de identificação da empresa.

Os gestores poderão consultar e administrar os dados de planejamento, gerenciamento de indicadores, controlar acesso a módulos e aprovar novas documentações, contratações e planejamentos.

3. Definição conceitual da solução

Esta seção apresenta uma definição conceitual da solução a ser desenvolvida: requisitos funcionais e não funcionais restrições e mecanismos arquiteturais considerados.

3.1. Requisitos Funcionais

Módulo Colaborador

- Autenticação

  O sistema deve liberar acesso aos módulos e a itens de módulos mediante autenticação segura;

  O sistema deve permitir encerrar a sessão por meio de opção presente na aplicação.

- Gestão de documentos

  O sistema deve possibilitar criar um documento;

  O sistema deve possibilitar buscar um documento;

  O sistema deve possibilitar salvar um documento;

  O sistema deve possibilitar publicar um documento;

  O sistema deve possibilitar excluir um documento (Mantendo histórico);

  O sistema deve possibilitar editar um documento (Mantendo histórico).

- Gestão de pessoas

  O sistema deve possibilitar criar uma contratação;

  O sistema deve possibilitar buscar contratação;

  O sistema deve possibilitar salvar uma contratação;

O sistema deve possibilitar publicar uma contratação;

O sistema deve possibilitar excluir uma contratação (Mantendo histórico);

O sistema deve possibilitar editar uma contratação (Mantendo histórico).

- Plano de ação

  O sistema deve possibilitar criar um Plano de Ação;

  O sistema deve possibilitar editar um Plano de Ação (Manter histórico);

  O sistema deve possibilitar excluir um Plano de Ação (Manter histórico);

  O sistema deve possibilitar salvar um Plano de Ação;

  O sistema deve possibilitar publicar um Plano de Ação;

  O sistema deve possibilitar buscar um Plano de Ação.

- Plano de riscos

  O sistema deve possibilitar criar um Plano de Riscos;

  O sistema deve possibilitar editar um Plano de Riscos (Manter histórico);

  O sistema deve possibilitar excluir um Plano de Riscos (Manter histórico);

  O sistema deve possibilitar salvar um Plano de Riscos;

  O sistema deve possibilitar publicar um Plano de Riscos;

  O sistema deve possibilitar buscar um Plano de Riscos.

- Comunicação

  O sistema deve permitir o agendamento de uma reunião por vídeo;

  O sistema deve permitir o agendamento de uma auditoria;

O sistema deve permitir o cancelamento de uma reunião por vídeo;

O sistema deve permitir o cancelamento de auditoria;

O sistema deve enviar alerta para as pessoas anexadas como necessárias à reunião/auditoria.

Módulo Gerencial

- Autenticação

  O sistema deve liberar acesso aos módulos e a itens de módulos mediante autenticação segura;

  O sistema deve permitir encerrar a sessão por meio de opção presente na aplicação.

- Indicadores

  O sistema deve possibilitar a inserção dos seguintes parâmetros:

  1. Resultados totais obtidos;
  2. Resultados totais pretendidos;
  3. Recursos totais empregados;
  4. Capacidade de produção total;
  5. Trabalho realizado total;
  6. Vendas totais;
  7. Investimentos totais;
  8. Cota de mercado;
  9. Lucro Total.

  O sistema deve possibilitar a inserção de comentários para os parâmetros inseridos;

O sistema deve possibilitar o cálculo dos seguintes indicadores:

1. Indicador de Eficácia (Resultados totais obtidos/Resultados totais pretendidos);
2. Indicador de Eficiência (Resultados totais obtidos/Recursos totais empregados);
3. Indicador de Capacidade (Capacidade de produção total/Tempo (mês));
4. Indicador de Qualidade (Trabalho realizado total/Tempo (mês));
5. Indicador de Lucratividade (Vendas totais/Lucro total);
6. Indicador de Rentabilidade (Investimentos totais/Lucro total);
7. Indicador de Competitividade (Cota de mercado).

- Controle do sistema

O sistema deve permitir controle sobre os seguintes parâmetros:

1. Acesso aos módulos por grupos;
2. Criar um grupo de acesso.

- Aprovação

O sistema deve prover lista para aprovações/reprovações de atividades diversas como:

1. Publicar Contratação;
2. Publicar Documento;
3. Publicar Plano de Riscos;
4. Publicar Plano de Ação.

O sistema deve prover um campo de comentário para a aprovação/reprovação.

- Comunicação

O sistema deve permitir o agendamento de uma reunião por vídeo;

O sistema deve permitir o agendamento de uma auditoria;

O sistema deve permitir o cancelamento de uma reunião por vídeo;

O sistema deve permitir o cancelamento de auditoria;

O sistema deve enviar alerta para as pessoas anexadas como necessárias à reunião/auditoria.

Console de administrador

- Controle de custos

  O sistema deve possibilitar uma estimativa mensal para o custo de manutenção/operação do sistema.

- Usuários

  O sistema deve possibilitar *upload* de tabela de novos usuários;

  O sistema deve possibilitar a exclusão de usuários.

3.2 Requisitos Não-Funcionais

Segurança - O sistema deve manter um padrão rigoroso de segurança de acesso.

- Acesso ao sistema sem ter as credenciais de acesso.

Estímulo	Acessar um módulo sem realizar a autenticação.
Fonte de estímulo	Usuário com credenciais incorretas as insere na interface de login.
Ambiente	Funcionamento, carga normal.
Artefato	Sistema SGQ (Qualquer Módulo)
Resposta	O sistema deve informar que as credenciais não conferem.
Medida da resposta	Sistema não permite acesso.

- Acesso ao sistema com credenciais de acesso corretas.

Estímulo	Acessar um módulo utilizando credenciais corretas.
Fonte de estímulo	Usuário com credenciais de outro módulo as insere na interface de login.
Ambiente	Funcionamento, carga normal.
Artefato	Sistema SGQ (Qualquer Módulo)
Resposta	O sistema deve informar a necessidade de um fator adicional de autenticação.
Medida da resposta	Sistema não permite acesso imediato ao módulo.

- Criptografia ponta-a-ponta.

Estímulo	Publicação de um Plano de Ação.
Fonte de estímulo	Usuário com acesso ao módulo envia um plano de ação para publicação.
Ambiente	Funcionamento, carga normal.
Artefato	Sistema SGQ (Seção de documentos)
Resposta	O sistema criptografa os dados de publicação antes de enviá-la.
Medida da resposta	Utilizando um software de análise de protocolo de rede, não é possível visualizar o conteúdo da mensagem.

Disponibilidade – O sistema deve estar disponível em qualquer período (Regime 24/7), com indisponibilidade máxima de 30 minutos semanais.

Estímulo	Indisponibilidade de um servidor.
Fonte de estímulo	Usuário não autenticado inicia um ataque DoS ao servidor.
Ambiente	Funcionamento, carga normal.
Artefato	Gerenciador de servidores.
Resposta	O sistema fica disponível.

Medida da resposta	Sistema volta a responder dentro de 30 minutos.

Desempenho – O sistema deve ter respostas rápidas.

Estímulo	Usuário navegando pelo módulo Colaborador.
Fonte de estímulo	Usuário inicia cadastro de um plano de ação.
Ambiente	Funcionamento, carga normal.
Artefato	Sistema SGQ
Resposta	O sistema criou um novo plano de ação.
Medida da resposta	O sistema criou um novo plano de ação em menos de 5 segundos.

Interoperabilidade – O sistema deve prover indicadores e dados para sistemas ERP.

Estímulo	Requisição de indicadores.
Fonte de estímulo	Sistema ERP solicita dados dos indicadores.
Ambiente	Funcionamento, carga normal.
Artefato	API de integração.
Resposta	Prover os indicadores solicitados.
Medida da resposta	Indicadores são retornados em estrutura de dados.

Manutenção – O sistema deve prover uma estimativa dos custos de operação mensal.

Estímulo	Uso dos módulos por alguns dias.
Fonte de estímulo	Administrador acessa a conta de gerenciamento do sistema.
Ambiente	Console Aws, funcionamento normal.
Artefato	Seção de faturamento mensal.
Resposta	Indicação dos valores de custos.
Medida da resposta	Gráfico apresentando o valor mensal previsto.

3.3. Restrições Arquiteturais

- O sistema deve ser desenvolvido em Javascript utilizando a plataforma Node.js (Back-end) e HTML + Javascript + CSS (Front-end);
- O sistema deve ser desenvolvido para plataforma em nuvem para permitir escalabilidade independente da aquisição de novos equipamentos de infraestrutura;
- O sistema deve ser responsivo, focado em equipamentos maiores como notebooks e desktops.

3.4. Mecanismos Arquiteturais

Análise	**Projeto**	**Implementação**
Front-end	Interface do usuário do sistema.	HTML + CSS + Javascript
Back-end	Regras de negócio	Javascript
Persistência	Banco de dados noSQL.	AWS DynamoDB
EndPoint de API	Métodos de API	AWS APIGateway
Autenticação	Inserção em pool de usuários e emissão de token de acesso.	AWS Cognito
Autorização	Autorização de execução de funções.	AWS IAM
Armazenamento de conteúdo estático	Armazenador de conteúdo web estático.	AWS S3
Endpoint com usuário	Interface e roteamento para acesso do usuário.	AWS S3 + AWS Route 53
Construção	Geração de artefatos.	Visual Studio + Node.js
Implantação	Adicionar e testar os artefatos no servidor.	AWS CLI ou outra ferramenta provida pela AWS.
Versionamento	Controle de código-fonte.	Git + Tortoise
Rastreabilidade	Acompanhamento e rastreabilidade de modificações no projeto.	Redmine

Tabela 1 - Mecanismos Arquiteturais.

4. Modelagem e projeto arquitetural

Nesta seção estão contidos os diagramas que permitem entender a arquitetura da aplicação.

4.1. Modelo de componentes

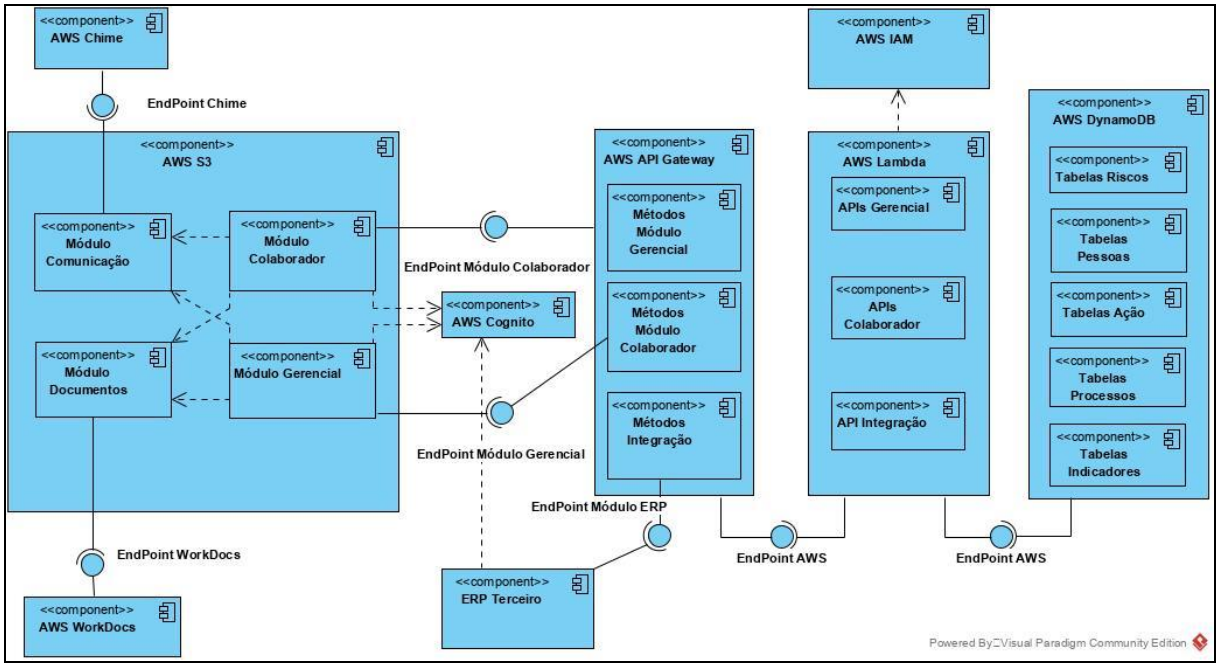

Figura 1 - Diagrama de componentes SGQ.

Visando uma boa prática arquitetural, os componentes apresentados no diagrama foram dispostos de forma a criarem um sistema em camadas, proporcionando a paralelização do desenvolvimento do sistema proposto. Outro ponto importante a mencionar é que o uso do padrão em camadas provê uma maior facilidade de testes e manutenções, uma vez que você tem entregáveis menores para testes contínuos e separação clara de papéis, tornando manutenções corretivas menos complexas e mais assertivas.

Os acessos as APIs são dados por métodos de requisição https diretamente ao EndPoint da API Gateway, o qual é configurado para liberar a requisição mediante ao *token* recebido do grupo de usuários criados no AWS Cognito como utilizadores daquela aplicação. Os serviços Aws Cognito e Aws API Gateway funcionam como uma camada de segurança, prevenindo assim o uso de recursos indevidamente por usuários não autenticados.

Uma vez acessado os serviços de APIs da aplicação, o AWS IAM é utilizado para autorização das funções do AWS Lambda, funções estas criadas para suas rotinas diversas de acesso as tabelas de dados e eventuais gatilhos de outras ações escaladas.

Foram pensados em componentes não só em profundidade de número de camadas, mas também, em granularidade de acordo com as responsabilidades dos módulos, havendo uma separação de Interfaces, EndPoints, APIs e tabela de dados para cada grupo de usuário, facilitando assim o desenvolvimento, manutenção e visualização de métricas de uso do sistema.

Os componentes Módulo Documentos e Módulo Comunicação foram evidenciados como dependências dos módulos Gerencial e Colaborador por serem equivalentes para os dois e por serem serviços oferecidos “as-is” com possiblidades de customização por meio de SDK oferecido pela AWS.

4.2. Modelo de implantação

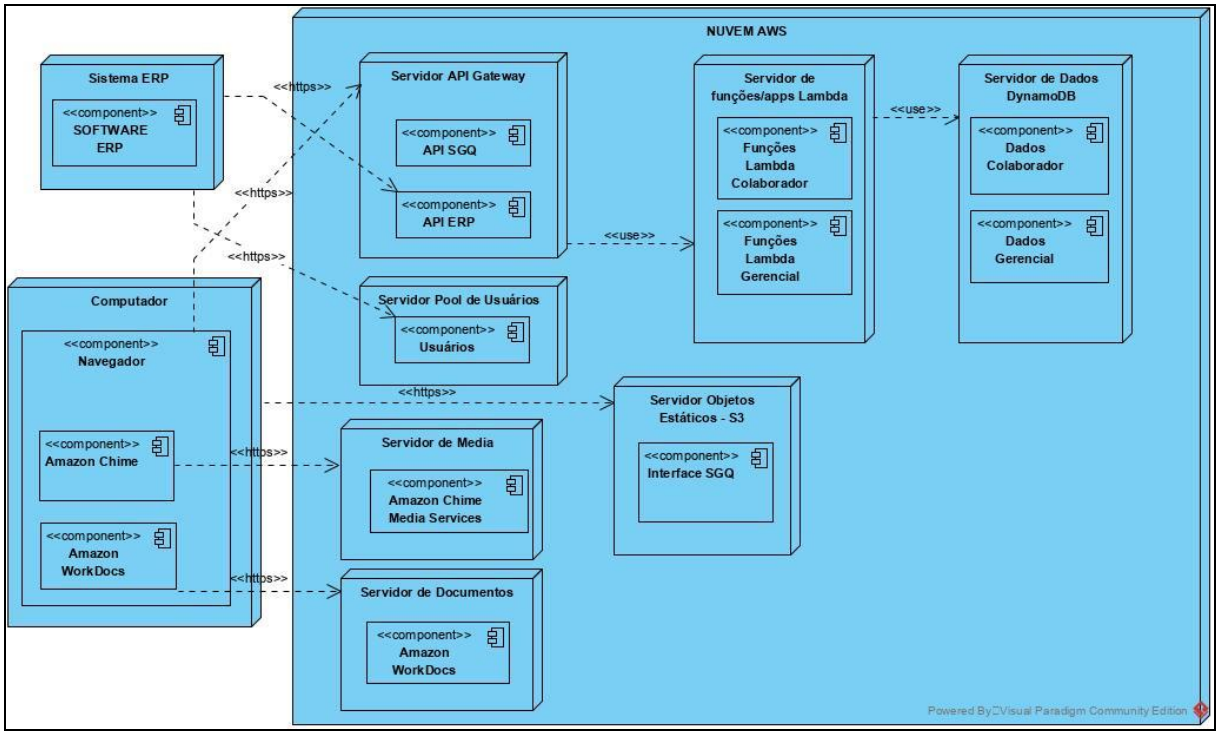

Figura 2 - Diagrama de implantação.

Para a implantação, como mencionado no capítulo de restrições arquiteturais (3.3. Restrições Arquiteturais), uma das restrições é a utilização da nuvem como meio de escalabilidade independente de instalações de infraestrutura. Sendo assim, toda a solução foi inserida em Serviços Web da Amazon (AWS). Os acessos à aplicação e suas funcionalidades se dão por meio de EndPoints de acesso utilizando requisições *https* e utilizando serviços de controle de usuários e gateways.

Uma das motivações para a escolha da nuvem *AWS* para implantação, além da comunidade consolidada e suporte da *Amazon*, é o fato dela possibilitar uma fácil escalabilidade do sistema, provendo serviços e métricas de uso que auxiliam na tomada de decisão por novos serviços que atendam a demanda do sistema.

Por se tratar de um sistema pensado para pequenas e médias empresas, onde a demanda é de certa forma constante, o uso somente de Funções Lambda para implementação das APIs proporciona ao custo de manutenção do sistema um determinado ganho proporcional ao uso, uma vez que, o custo está diretamente relacionado ao número de requisições e tempo de execução da mesma. Dessa forma, o custo de manutenção será proporcional ao uso que esta empresa empregará ao sistema e terá níveis baixos quando não estiver utilizando-o, como por exemplo, em período de férias coletivas.

Tendo como base que a utilização de sistemas de gestão para engajamento dos colaboradores em um cenário de trabalho remoto foi utilizado o serviço de comunicação AWS Chime, o qual prove alerta de agendamento de reuniões e suporte a videoconferências. A princípio o AWS Chime é um serviço de comunicação que oferece um serviço pronto para uso, porém, também é oferecido suporte por meio de SDK para integração com as regras de negócio do sistema proposto.

Componente Nó	**Descrição**
Computador	Máquina do colaborador destinada para execução do navegador web escolhido. Está destinada também a instalação de SDKs necessários para comunicação dos módulos de comunicação e de colaboração de documentos.

Servidor de Objetos Estáticos	Servidor em nuvem denominado S3 destinado ao armazenamento dos objetos estáticos da aplicação desenvolvida.
Servidor Pool de Usuários	Servidor em nuvem denominado Cognito para gerenciamento de grupos de usuários e identidades.
Servidor de API	Servidor em nuvem denominado API Gateway para criação de EndPoints e gatilho de execução de funções e aplicativos.
Servidor de Funções/Aplicativos	Servidor em nuvem denominado Lambda para execução de lógicas de negócios da aplicação.
Servidor de Dados	Servidor em nuvem denominado DynamoDB para armazenamento de dados da aplicação.
Servidor de Media	Servidor em nuvem denominado Chime para comunicação por videoconferências e agendamento de reuniões com alertas.
Servidor de Documentos	Servidor denominado WorkDocs para criação colaborativa de Documentos.

4.3. Diagramas casos de uso (UC)

Os diagramas de casos de uso da Figura 3, Figura 4, Figura 5 e Figura 6 a seguir, são ações comuns realizadas pelo colaborador, como Criar, Editar, Buscar, Salvar e Excluir algum documento, pessoas ou planos.

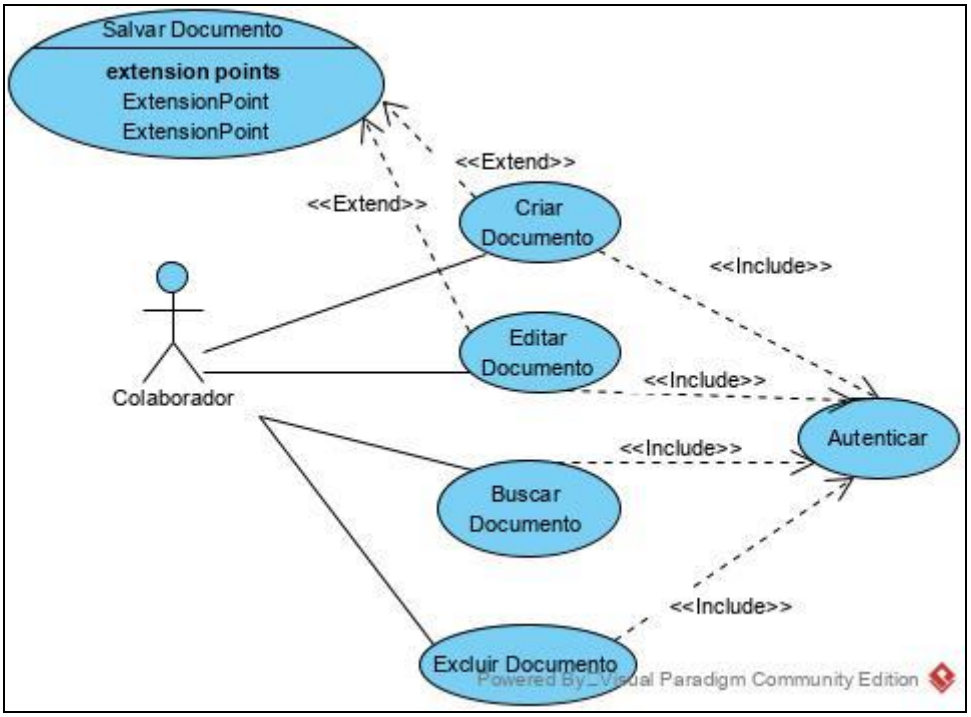

Figura 3 - UC componentes documentos.

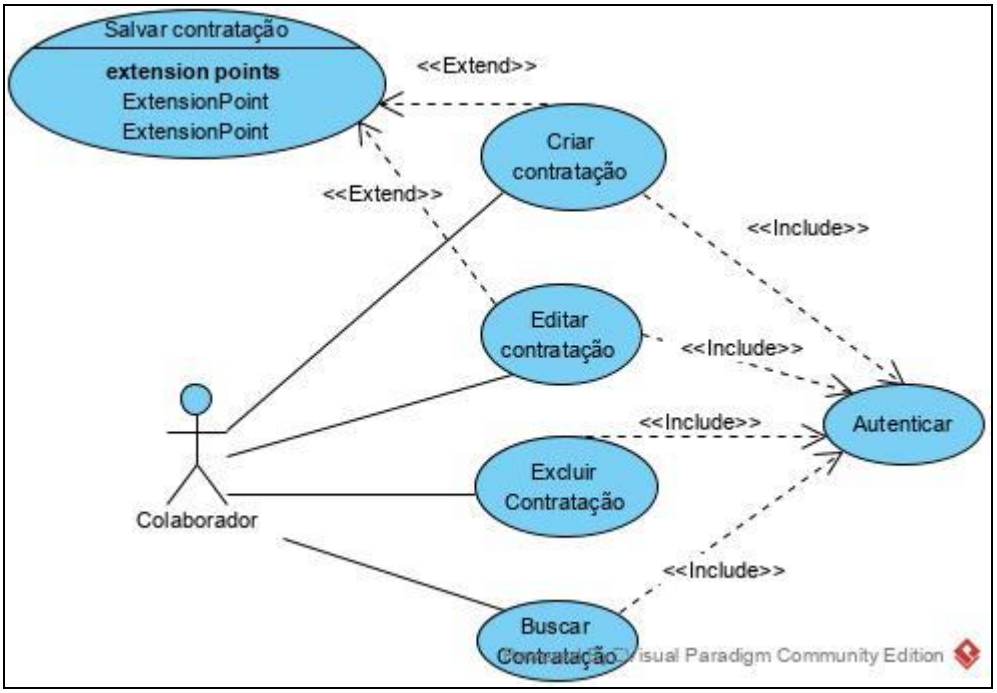

Figura 4 - UC componentes Gestão de Pessoas.

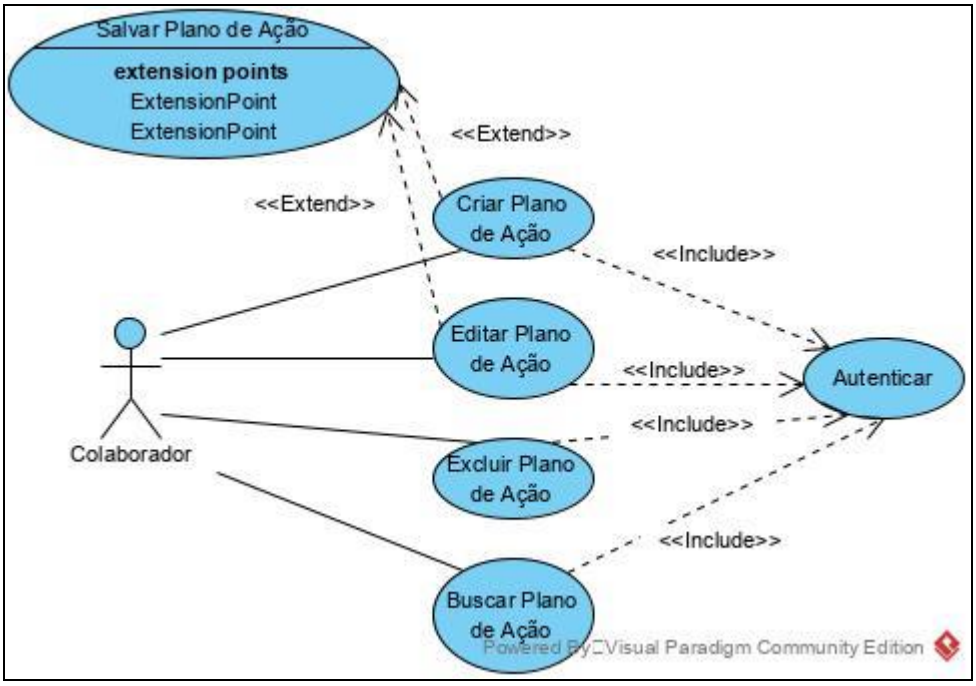

Figura 5 - UC componentes Plano de Ação.

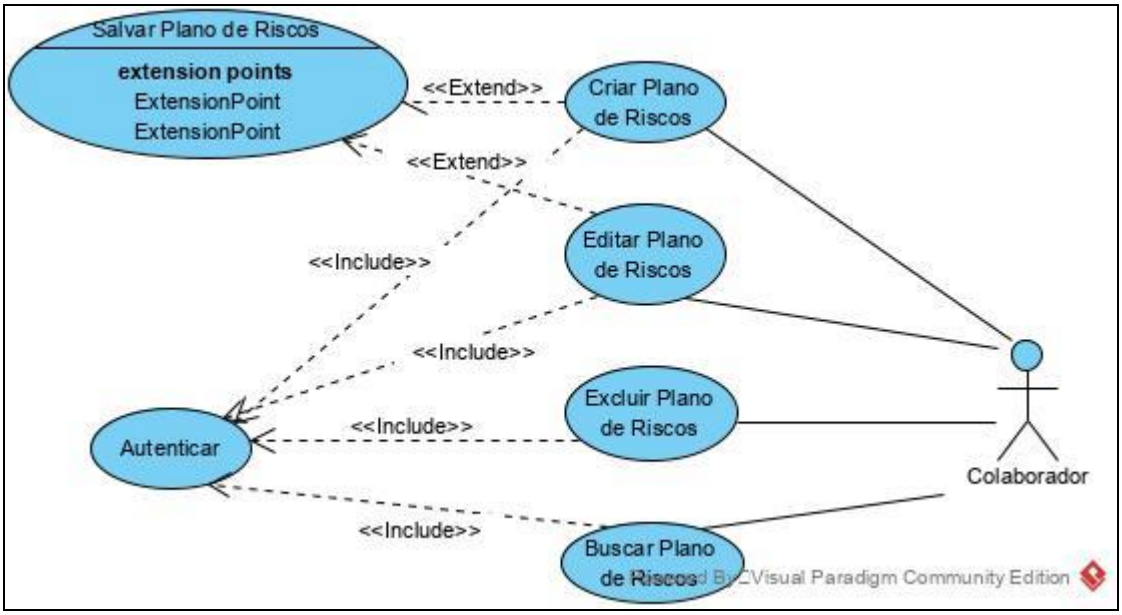

Figura 6 - UC componentes Plano de Riscos.

Na Figura 7, os componentes evidenciados são os de cancelar ou agendar uma reunião/auditoria, gerando um alerta para os colaboradores envolvidos e/ ou essenciais a reunião/auditoria agendada.

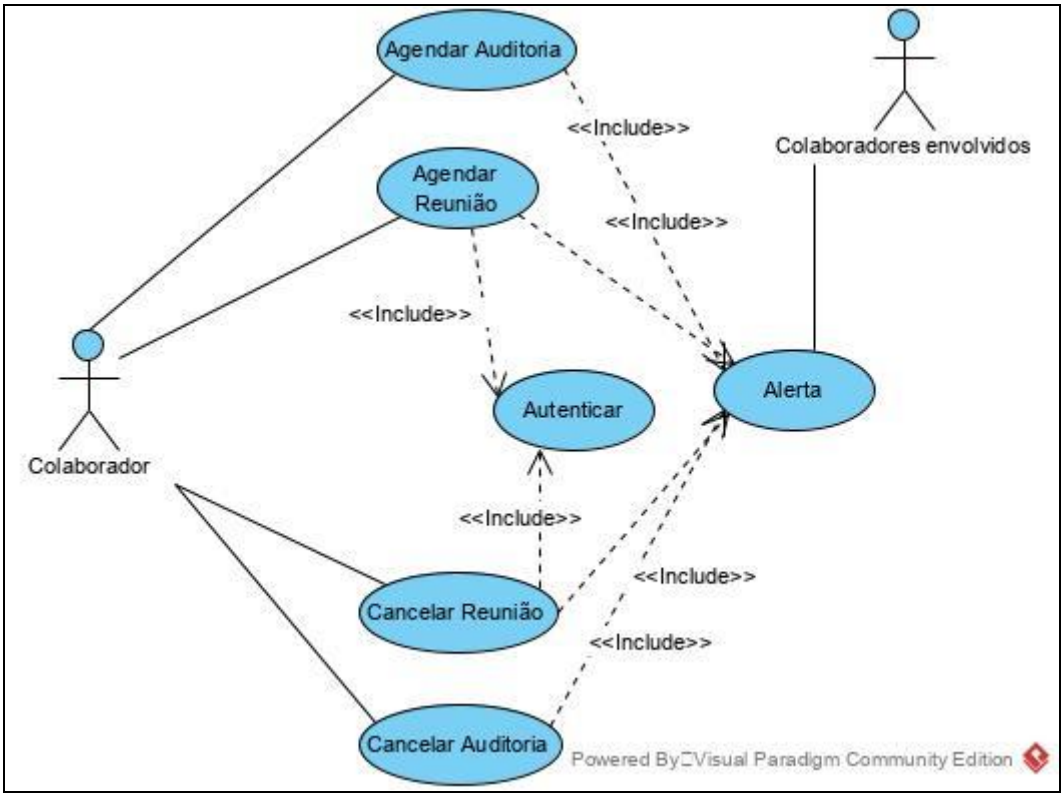

Figura 7 - UC componentes Comunicação.

Na Figura 8, estão evidenciadas as relações das funções de Publicação, as quais podem ser realizadas por colaboradores quando desejam aprovação Gerencial para novas contratações, documentos e Planos publicados no sistema.

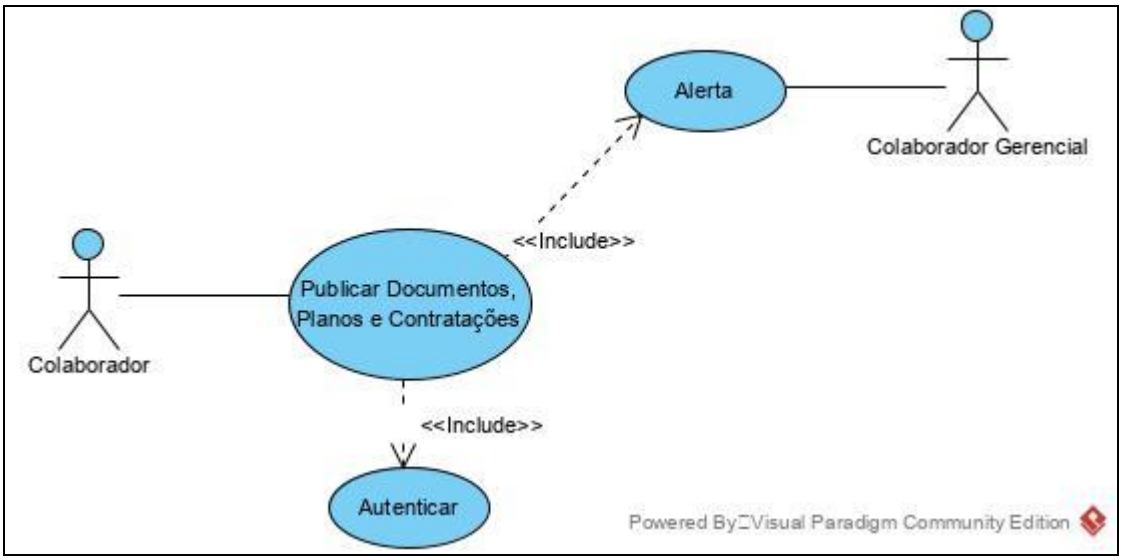

Figura 8 - UC componentes Publicar.

Na Figura 9, está evidenciada a relação da API da aplicação com Módulos ERP, sendo possível mediante autenticação, iniciar uma requisição de indicadores para a API destinada para tal.

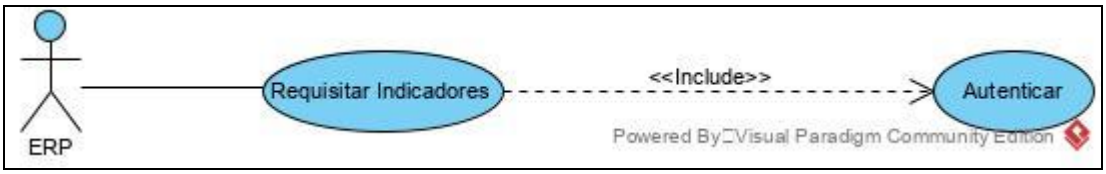

Figura 9 - UC API módulo ERP.

Na Figura 10, está evidenciada a relação do colaborador Gerencial no Sistema, primeiro em relação aos KPIs, onde o mesmo irá inserir os parâmetros para cálculos de KPIs, armazenar os indicadores calculados caso requerido e requisitar indicadores quando também requerido.

Para o colaborador gerencial também estão destacadas as relações com algumas funcionalidades do sistema, podendo o mesmo gerenciar os grupos de acessos aos módulos do sistema.

Outra interação do Colaborador Gerencial é a aprovação de itens diversos, como documentações, contratações e planos.

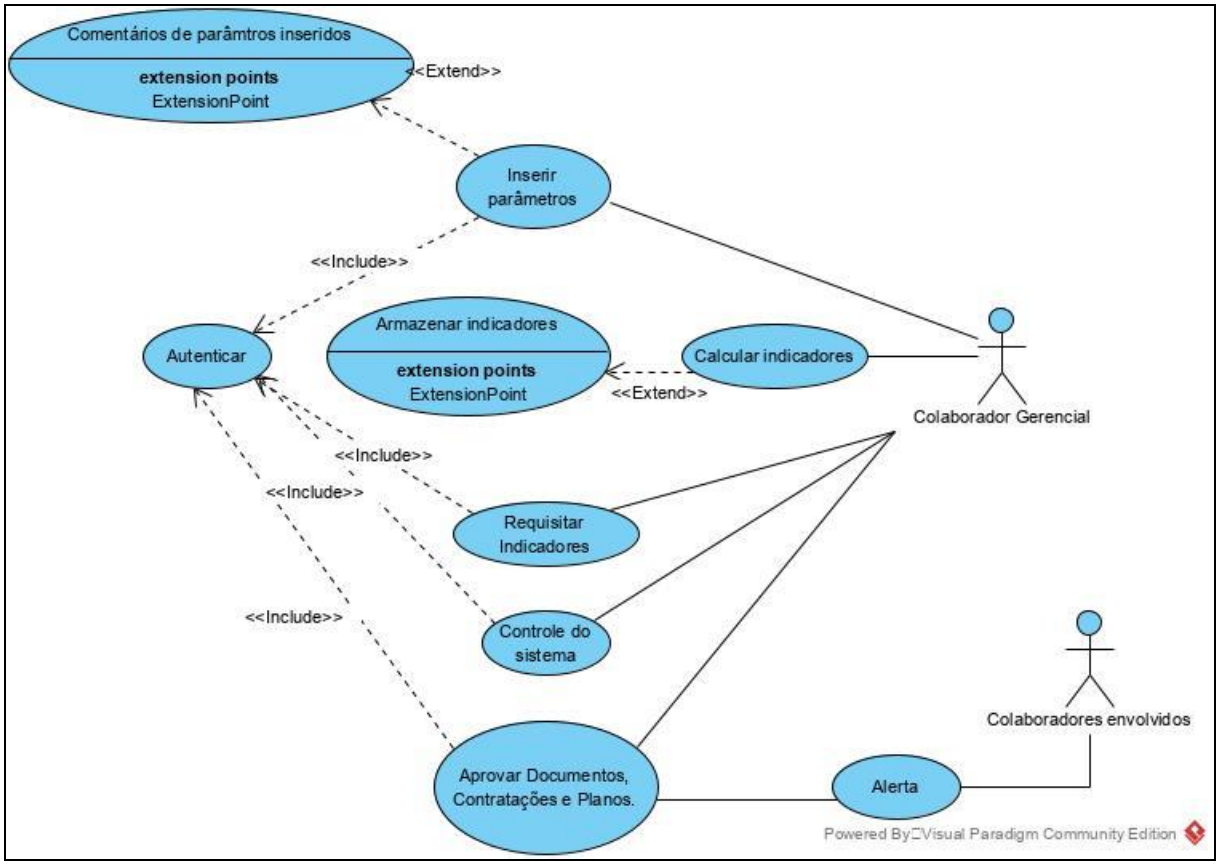

Figura 10 - UC componentes Módulo Gerencial.

Na Figura 11, estão as relações administrativas no sistema, como a visualização das estimativas de custos de operação e a função de inserção de novos usuários.

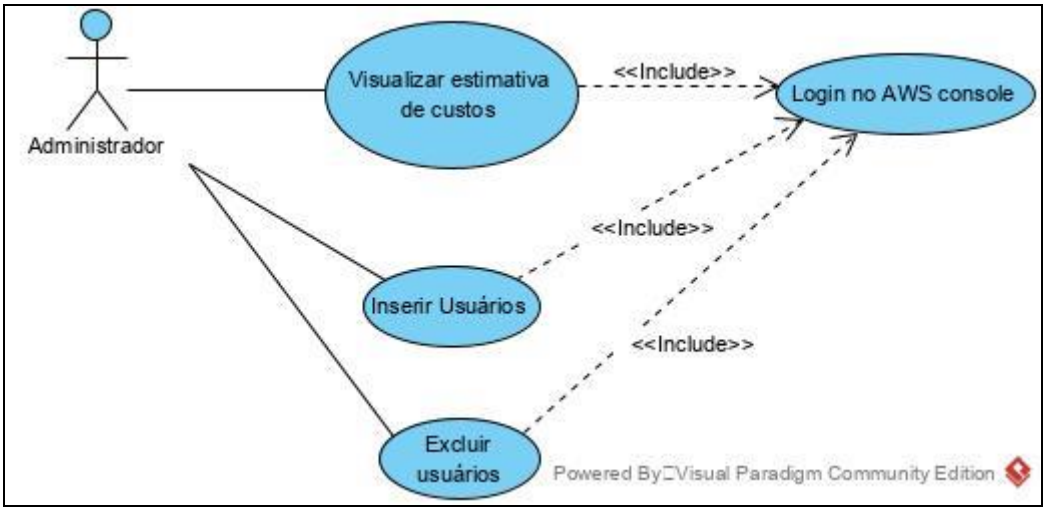

Figura 11 - UC componentes administrador.

5. Prova de Conceito (POC) / protótipo arquitetural

Nesta seção, serão apresentados os requisitos para a prova de conceito, assim como as tecnologias utilizadas, implantação, códigos e API.

5.1. Implementação e Implantação

5.1.1. Requisitos não funcionais

Os requisitos funcionais validados para esta prova de conceito e seus critérios de aceite, estão listados da seguinte forma:

Módulo	Casos de uso (UC)	Requisitos não funcionais
Gerencial	Autenticar	Segurança
	Calcular Indicadores	Usabilidade
	Armazenar indicadores	Segurança
Administrador	Estimativa de custos mensais	Manutenção
API ERP	Requisitar indicadores	Interoperabilidade

Tabela 2 - Tabela Casos de uso x Requisitos não-Funcionais.

- **Segurança** – Autenticar

O uso de um aplicativo corporativo na nuvem expõe informações sensíveis em servidores não fisicamente gerenciáveis e isto muitas vezes desencoraja a prática da infraestrutura. Por esta razão, o primeiro requisito não funcional é a segurança, seus critérios de aceite são:

1. Não permitir a autenticação de usuários não cadastrados no AWS Cognito;
2. Redirecionar para autenticação quando o caminho da aplicação for inserido no buscador utilizado;
3. Não permitir navegação em nenhuma página caso não autenticado;
4. Existir uma opção para sair da aplicação.

- **Segurança** – Armazenar Indicadores

Devido à alta modularização e descentralização proporcionada pelo ambiente em nuvem, o tráfego de requisições ocorre mais frequentemente em domínios que não se pode controlar diretamente, como em um ambiente corporativo de acesso mais restrito. Devido a isso o segundo requisito não funcional também é a segurança, tendo os seguintes critérios de aceite:

1. Ao "*sniffar*" a mensagem do protocolo, não deve ser possível identificar o valor dos parâmetros enviados;
2. Para a requisição, deve ser utilizado o protocolo *https*;
3. Para a resposta, deve ser utilizado o protocolo *https*.

- **Usabilidade** – Calcular Indicadores

1. O tempo de resposta de uma requisição não deve ser maior que 1000 milissegundos para 8 parâmetros enviados;
2. O tempo total de solicitação e de resposta não deve ser maior que 1000 milissegundos.

- **Manutenção** – Estimativa de custos mensais

1. A estimativa dos custos mensais deve ser provida graficamente;
2. O valor deve estar em dólar;
3. Deve ser possível verificar o custo de cada recurso utilizado.

- **Interoperabilidade** - Requisitar Indicadores

1. O sistema deve prover informações de indicadores mediante as requisições de softwares de terceiros.

As tecnologias utilizadas nesta seção são evidenciadas abaixo:

Casos de Uso (UC)	Tecnologias utilizadas
Autenticar	Html, Javascript, CSS, AWS Cognito, AWS IAM, Microsoft Visual Studio 2019.
Calcular Indicadores	Html, Javascript, CSS, AWS Cognito, AWS Lambda, AWS IAM.
Armazenar indicadores	Html, Javascript, CSS, AWS Cognito, AWS Lambda, AWS IAM, DynamoDB.
Estimativas de custos mensais	AWS Console
Requisitar indicadores	AWS Cognito, AWS Lambda, AWS IAM, DynamoDB.

Tabela 3 - Casos de Uso x Tecnologias.

5.1.2. Códigos

No código a seguir, são apresentadas as funções utilizadas para a implementação da API de requisição e cálculo de KPIs. Para melhor entendimento da lógica empregada foram adicionados alguns comentários.

```
// SDK utilizado.
const AWS = require('aws-sdk');

// Instância de objeto para gravação e leitura da tabela.
const ddb = new AWS.DynamoDB.DocumentClient();

// Data utilizada como identificador.
var date = new Date().toISOString();

// ID para leitura da tabela por data.
var ID = generateID(date);

// Parâmetros para scan da tabela.
var params = {
  ExpressionAttributeValues: {
    ':record': ID
  },
  ProjectionExpression: 'RecordID, KPI1, KPI2, KPI3, KPI4, KPI5, KPI6, KPI7, User-
Name',
  FilterExpression: 'contains (RecordID, :record)',
  TableName: 'KPIs'
};

// Estrutura para regra de negócio
var paramCollection = [];

// Verificação da chave de autorização.
exports.handler = (event, context, callback) => {
    if (!event.requestContext.authorizer) {
      errorResponse('Autorização não configurada', context.awsRequestId, callback);
      return;
    }

    //Timestamp para identificador da requisição.
    const recordId = new Date().toISOString();

    // Usuário da requisição
    const username = event.requestContext.authorizer.claims['cognito:username'];

    // Como ja é conhecido o content-type, utilizando parser JSON (O correto seria
verificar o tipo do conteúdo)
    const requestBody = JSON.parse(event.body);

    // Estrutura com os KPIs
    var KPI = requestBody.KPI;

    // Chamada da função da regra de negócio que envolve o cálculo dos KPIs
    var unidade = calcKPI(KPI);

    // Chamada da função para gravação das informações em tabela.
    if (KPI.p10 === 'P') // Grava em tabela calculado
    {
     recordKPI(recordId, username, unidade).then(() => {
        // Callback de resposta para a requisição da API gateway
        callback(null, {
            statusCode: 201,
            body: JSON.stringify({
                RecordID: recordId,
                KPI1: unidade[0],
                KPI2: unidade[1],
                KPI3: unidade[2],
                KPI4: unidade[3],
                KPI5: unidade[4],
                KPI6: unidade[5],
                KPI7: unidade[6],
                UserName: username,
```

```
            }),
            headers: {
                'Access-Control-Allow-Origin': '*',
            },
        });
     }).catch((err) => {
        //console.error(err);
        // Chamada da callback com a mensagem de erro
        errorResponse(err.message, context.awsRequestId, callback);
     });
    }
    else if(KPI.p10 === 'U') // Scaneia tabela para KPIs do dia.
    {
        var kpi1 = 0, kpi2 = 0, kpi3 = 0, kpi4 = 0, kpi5 = 0, kpi6 = 0,
        kpi7 = 0;

        ddb.scan(params, function(err, data){
        if(err)
        {
         return errorResponse(err.message, context.awsRequestId, callback);
        }
        else
        {
        data.Items.forEach(function(element, index, array)
          {
            kpi1 += Number(element.KPI1);
            kpi2 += Number(element.KPI2);
            kpi3 += Number(element.KPI3);
            kpi4 += Number(element.KPI4);
            kpi5 += Number(element.KPI5);
            kpi6 += Number(element.KPI6);
            kpi7 += Number(element.KPI7);
          });
          kpi1 /= Number(data.Count);
          kpi2 /= Number(data.Count);
          kpi3 /= Number(data.Count);
          kpi4 /= Number(data.Count);
          kpi5 /= Number(data.Count);
          kpi6 /= Number(data.Count);
          kpi7 /= Number(data.Count);

         return callback(null, {
            statusCode: 201,
            body: JSON.stringify({
                RecordID: recordId,
                KPI1: kpi1,
                KPI2: kpi2,
                KPI3: kpi3,
                KPI4: kpi4,
                KPI5: kpi5,
                KPI6: kpi6,
                KPI7: kpi7,
                UserName: username,
            }),
            headers: {
                'Access-Control-Allow-Origin': '*',
            },
        });
        }
    });
    }
    else if (KPI.p10 === 'C') //Calcula os KPIs e responde.
    {
        // Callback de resposta para a requisição da API gateway
        return callback(null, { //Somente responde calculo
            statusCode: 201,
```

```
            statusCode: 201,
            body: JSON.stringify({
                RecordID: recordId,
                KPI1: unidade[0],
                KPI2: unidade[1],
                KPI3: unidade[2],
                KPI4: unidade[3],
                KPI5: unidade[4],
                KPI6: unidade[5],
                KPI7: unidade[6],
                UserName: username,
            }),
            headers: {
                'Access-Control-Allow-Origin': '*',
            },
        });
    }
    else errorResponse('Parâmetro não definido na API!', context.awsRequestId, call-
back);
};

// Função que retira os IDs do dia para busca.
function generateID(date){
   var i = date.indexOf('T');
    if (i !== -1)
     return date.substr(0, i);
    else
     return -1;
}

// Função com lógica das regras de negócios envolvendo os KPIs recebidos.
function calcKPI(KPI) {
    //console.log('Returning response: ', KPI.kpium, ', ', KPI.kpidois);
    paramCollection[0] = Number(KPI.p1)/Number(KPI.p2);
    paramCollection[1] = Number(KPI.p1)/Number(KPI.p3);
    paramCollection[2] = Number(KPI.p4)/31;
    paramCollection[3] = Number(KPI.p5)/31;
    paramCollection[4] = Number(KPI.p6)/Number(KPI.p9);
    paramCollection[5] = Number(KPI.p7)/Number(KPI.p9);
    paramCollection[6] = Number(KPI.p8);
    paramCollection[7] = Number(KPI.p10);
    return paramCollection;
}

//Função para gravação dos KPIs
function recordKPI(recordId, username, unidade) {
    return ddb.put({
        TableName: 'KPIs',
        Item: {
            RecordID: recordId,
            KPI1: unidade[0],
            KPI2: unidade[1],
            KPI3: unidade[2],
            KPI4: unidade[3],
            KPI5: unidade[4],
            KPI6: unidade[5],
            KPI7: unidade[6],
            UserName: username,
            //Origin: unidade[7],
        },
    }).promise();
}

//Função para formatação da mensagem de erro.
function errorResponse(errorMessage, awsRequestId, callback) {
```

```
  callback(null, {
    statusCode: 500,
    body: JSON.stringify({
      Error: errorMessage,
      Reference: awsRequestId,
    }),
    headers: {
      'Access-Control-Allow-Origin': '*',
    },
  });
}
```

As funções a seguir apresentam os mecanismos de autenticação do usuário e de comunicação da aplicação de interface com a API de indicadores.

```
/*_config AmazonCognitoIdentity AWSCognito*/

var SGQ6 = window.SGQ6 || {};

(function scopeWrapper($) {
    var signinUrl = 'signin.html';

    var poolData = {
        UserPoolId: _config.cognito.userPoolId,
        ClientId: _config.cognito.userPoolClientId
    };

    var userPool;

    if (!(_config.cognito.userPoolId &&
          _config.cognito.userPoolClientId &&
          _config.cognito.region)) {
        $('#noCognitoMessage').show();
        return;
    }

    userPool = new AmazonCognitoIdentity.CognitoUserPool(poolData);

    if (typeof AWSCognito !== 'undefined') {
        AWSCognito.config.region = _config.cognito.region;
    }

    SGQ6.signOut = function signOut() {
        userPool.getCurrentUser().signOut();
    };

    SGQ6.authToken = new Promise(function fetchCurrentAuthToken(resolve, reject)
    {
        var cognitoUser = userPool.getCurrentUser();

        if (cognitoUser) {
            cognitoUser.getSession(function sessionCallback(err, session) {
                if (err) {
                    reject(err);
                } else if (!session.isValid()) {
                    resolve(null);
                } else {
                    resolve(session.getIdToken().getJwtToken());
                }
            });
        } else {
            resolve(null);
        }
    });

    function signin(email, password, onSuccess, onFailure) {
        var authenticationDetails = new AmazonCognitoIdenti-
ty.AuthenticationDetails({
            Username: email,
            Password: password
        });

        var cognitoUser = createCognitoUser(email);
        cognitoUser.authenticateUser(authenticationDetails, {
```

```
function createCognitoUser(email) {
        return new AmazonCognitoIdentity.CognitoUser({
            Username: email,
            Pool: userPool
        });
    }

    /*
     *  Event Handlers
     */

    $(function onDocReady() {
        $('#signinForm').submit(handleSignin);
    });

    function handleSignin(event) {
        var email = $('#emailInputSignin').val();
        var password = $('#passwordInputSignin').val();
        event.preventDefault();
        signin(email, password,
            function signinSuccess() {
                console.log('Successfully Logged In');
                window.location.href = 'sgq.html';
            },
            function signinError(err) {
                alert(err);
            }
        );
    }
}(jQuery));
```

```
/*global _config*/


var SGQ6 = window.SGQ6 || {};

(function rideScopeWrapper($) {
    var authToken;
    SGQ6.authToken.then(function setAuthToken(token) {
        if (token) {
            authToken = token;
        } else {
            window.location.href = '/signin.html';
        }
    }).catch(function handleTokenError(error) {
        alert(error);
        window.location.href = '/signin.html';
    });

    function sendKPI() {
        $.ajax({
            method: 'POST',
            url: _config.api.invokeUrl + '/kpi-post',
            headers: {
                Authorization: authToken
            },
            data: JSON.stringify({
                KPI: {
                    p1: $('#param1').val(),
                    p2: $('#param2').val(),
                    p3: $('#param3').val(),
                    p4: $('#param4').val(),
                    p5: $('#param5').val(),
                    p6: $('#param6').val(),
                    p7: $('#param7').val(),
                    p8: $('#param8').val(),
                    p9: $('#param9').val(),
                    p10: $('#param10').val(),
                }
            }),
            contentType: 'application/json',
            success: completeRequest,
            error: function ajaxError(jqXHR, textStatus, errorThrown) {
                alert('Um erro ocorreu na requisição:\n' + jqXHR.responseText);
            }
        });
    }

    function completeRequest(result) {
        displayUpdate('KPI 1: ' + result.KPI1 + ' | KPI 2: ' + result.KPI2 + ' |
KPI 3: ' + result.KPI3 + ' | KPI 4: ' + result.KPI4 + ' | KPI 5: ' + result.KPI5 +
' | KPI 6: ' + result.KPI6 + ' | KPI 7: ' + result.KPI7);
    }

    // Register click handler for #request button
    $(function onDocReady() {
        $('#request').click(handleRequestClick);
        $('#signOut').click(function() {
            SGQ6.signOut();
            alert("Voce foi deslogado.");
            window.location = "signin.html";
        });
```

```
        if (!_config.api.invokeUrl) {
            $('#noApiMessage').show();
        }

        $('#auth').click(function () {
            displayUpdate(authToken);
        });
    });

    function handleRequestClick(event) {
        event.preventDefault();
        sendKPI();
    }

    function displayUpdate(text) {
        $('#updates').append($('<li>' + text + '</li>'));
    }
}(jQuery));
```

5.2. Interfaces/APIs

Na Tabela 4 são apresentados os parâmetros para requisição, gravação e cálculos dos indicadores de desempenho (KPIs). Na última linha da tabela estão evidenciados os parâmetros que são:

P – Utilizado para cálculo e armazenamento dos indicadores calculados;

U – Utilizado para requisição dos indicadores do dia;

C – Utilizado para cálculo e retorno dos indicadores calculados.

Entrada - JSON			
Endereço da API	https://3gu96zgtlf.execute-api.us-east-1.amazonaws.com/prod/kpi-post		
Parâmetro	**Tipo**	**Obrigatório**	**Descrição**
Authorization	String	sim	Token de autorização provido pelo AWS Cognito mediante login e senha.
p1	Integer	Somente para requisições P e C.	Resultados totais obtidos
p2	Integer	Somente para requisições P e C.	Resultados totais pretendidos
p3	Integer	Somente para requisições P e C.	Recursos totais empregados
p4	Integer	Somente para requisições P e C.	Capacidade de produção total
p5	Integer	Somente para requisições P e C.	Trabalho realizado total
p6	Integer	Somente para requisições P e C.	Vendas totais
p7	Integer	Somente para requisições P e C.	Investimentos totais
p8	Integer	Somente para requisições P e C.	Cota de mercado
p9	Integer	Somente para requisições	Lucro Total

		P e C.	
p10	String	sim	Parâmetro para requisição dos dados de KPIs armazenados no dia da requisição. Podendo ser P – Para armazenar os indicadores, U – Para requisitar indicadores ou C – Para calcular indicadores.

Tabela 4 - Tabela do input da API.

Na Tabela 5é mostado os indicadores enviados para a fonte da requisição, sendo informações como o ID gerado pela API, os indicadores KPI e o usuário que iniciou a requisição.

Saída - JSON		
Parâmetro	**Tipo**	**Descrição**
RecordID	String	ID da requisição gerada na API.
KPI1	String	Indicador de Eficácia (Resultados totais obtidos/Resultados totais pretendidos)
KPI2	String	Indicador de Eficiência (Resultados totais obtidos/Recursos totais empregados)
KPI3	String	Indicador de Capacidade (Capacidade de produção total/Tempo (mês))
KPI4	String	Indicador de Qualidade (Trabalho realizado total/Tempo (mês))
KPI5	String	Indicador de Lucratividade (Vendas totais/Lucro total)
KPI6	String	Indicador de

		Rentabilidade (Investimentos totais/Lucro total)
KPI7	String	Indicador de Competitividade (Cota de mercado)
UserName	String	Usuário autenticado que iniciou a requisição na API.

Tabela 5 - Tabela do output da API.

6. Avaliação da Arquitetura

A avaliação da arquitetura desenvolvida neste trabalho é abordada nesta seção, visando avaliar se atende ao que foi proposto.

6.1. Análise das abordagens arquiteturais

A arquitetura utilizada para o SGQ foi desenhada de forma a criar uma aplicação inteiramente em nuvem, ganhando agilidade no desenvolvimento paralelizado e independência de local de infraestrutura, podendo ser tanto desenvolvida quanto utilizada de forma remota.

O uso de serviços proprietários também foi uma escolha pensada no rápido desenvolvimento e na fácil integração, podendo ser a priori utilizados "as-is" para garantir uma entrega inicial mais rápida e posteriormente integrado utilizando os SDKs fornecidos, como nos casos do AWS Chime e AWS Workdocs.

6.2. Cenários

Cenário "Autenticar": Quando for inserida uma URL diretamente para acesso da aplicação sem antes estar autenticada, a página deve ser redirecionada automaticamente para a interface de login. O sistema deve obrigar o usuário a preencher todos os campos antes de enviar uma solicitação. Nenhum conteúdo da aplicação deve ser acessado sem antes estar autenticado. Após uso da aplicação a sessão deverá ser encerrada por meio de opção.

Cenário "Calcular Indicadores": Para solicitações de cálculo de indicadores o usuário deve preencher os parâmetros requeridos e submetê-los para cálculo. O sistema deve

apresentar os indicadores de forma rápida, sem que o usuário tenha que esperar mais que 1 segundo para obtê-los.

Cenário "Armazenar Indicadores": Ao utilizar funcionalidades diversas da aplicação que comunicam com outras camadas e que expõem informações empresariais sigilosas, as mesmas devem estar encapsuladas dentro de camadas de segurança.

Cenário "Estimativas de custos mensais": Ao acessar o console da AWS com usuário e senha root o administrador do sistema deverá obter facilmente as estimativas e custos mensais para toda a operação envolvendo o sistema utilizado.

Cenário "Requisitar indicadores": O sistema de gestão deve ser capaz de se comunicar com outros sistemas por meio de uma API RESTful, na qual quando solicitado um recurso de KPIs, o sistema deve retornar os KPIs do dia em que foi realizada a solicitação.

A tabela a seguir evidencia a complexidade pela importância de cada cenário.

		Complexidade		
	Classificação	**Baixo**	**Médio**	**Alto**
Importância	**Baixo**			
	Médio		**Calcular Indicadores**	**Estimativas de custos mensais**
	Alto		**Requisitar indicadores**	**Autenticar / Armazenar Indicadores**

Tabela 6 - Tabela Complexidade x Importância.

6.3. Avaliação

Cenário "Autenticar": As evidências dos critérios de aceitação para este cenário são:

- Ao inserir a URL diretamente de uma página da aplicação o usuário deve ser redirecionado para a página de *login*.
- Não deve existir nenhuma navegação pela aplicação sem o usuário estar devidamente autenticado.

- Deve existir um botão para que o usuário encerre a sessão que iniciou sendo necessário realizar o *login* novamente.

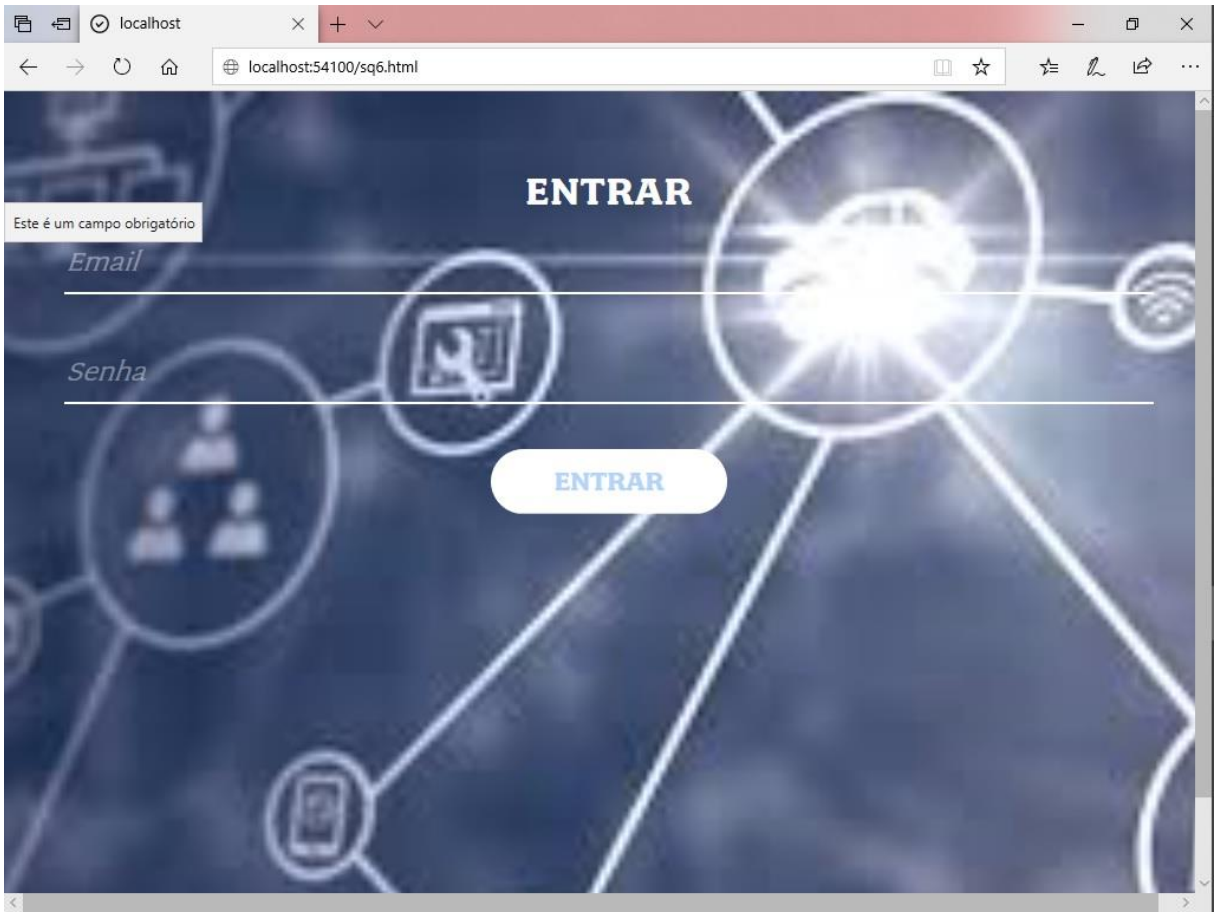

Figura 12. - Interface de Login

Na Figura 12 é apresentada a interface de *login* com campos obrigatórios de senha e usuários previamente cadastrados no AWS Cognito, no grupo denominado SGQUSUARIOS. Não sendo possível navegar pela aplicação, somente após efetuar o *login*.

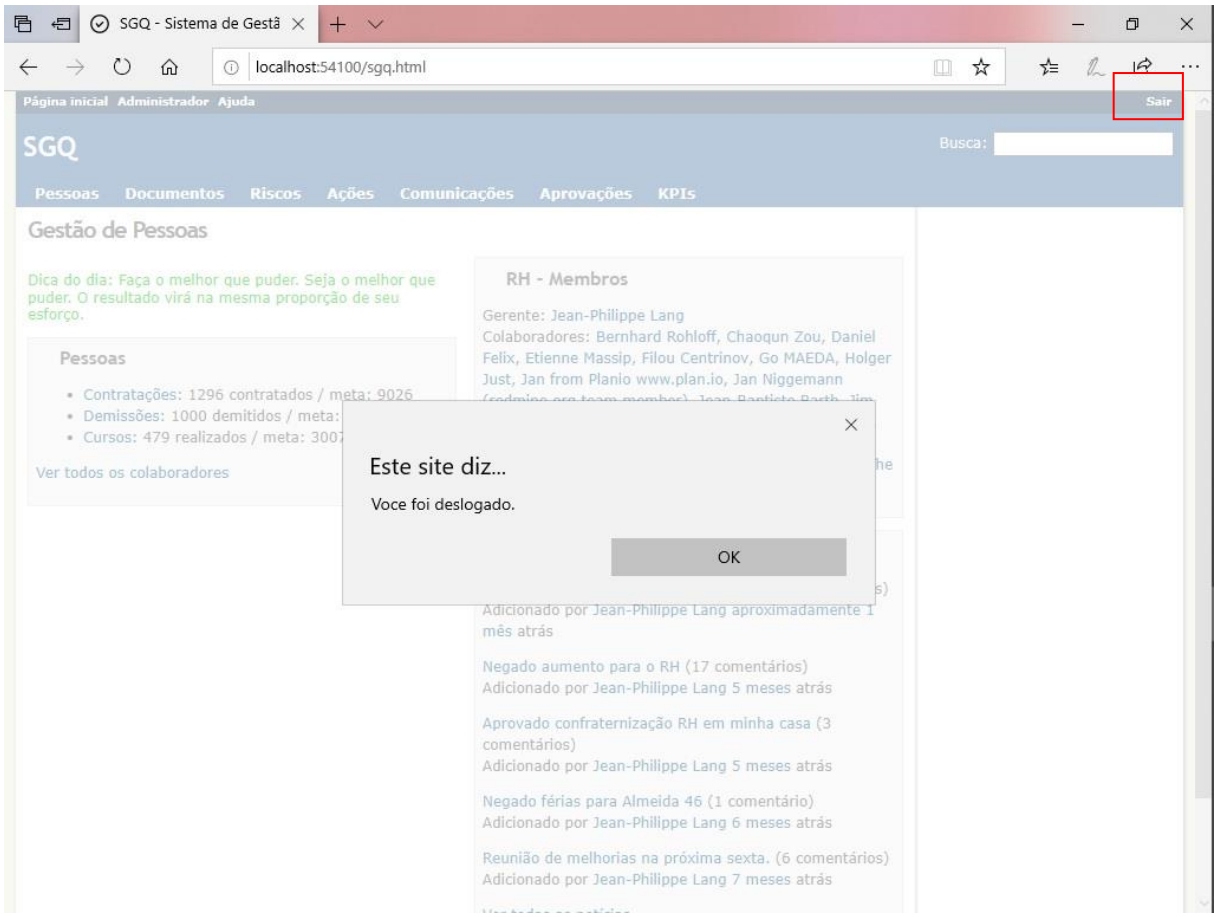

Figura 13 - Botão Sair (Log Out) pressionado.

Na Figura 13 é apresentada a interface após pressionar o botão Sair, indicando ao usuário que o mesmo não está mais na sessão, sendo redirecionado para a Interface de *login* (Figura 12).

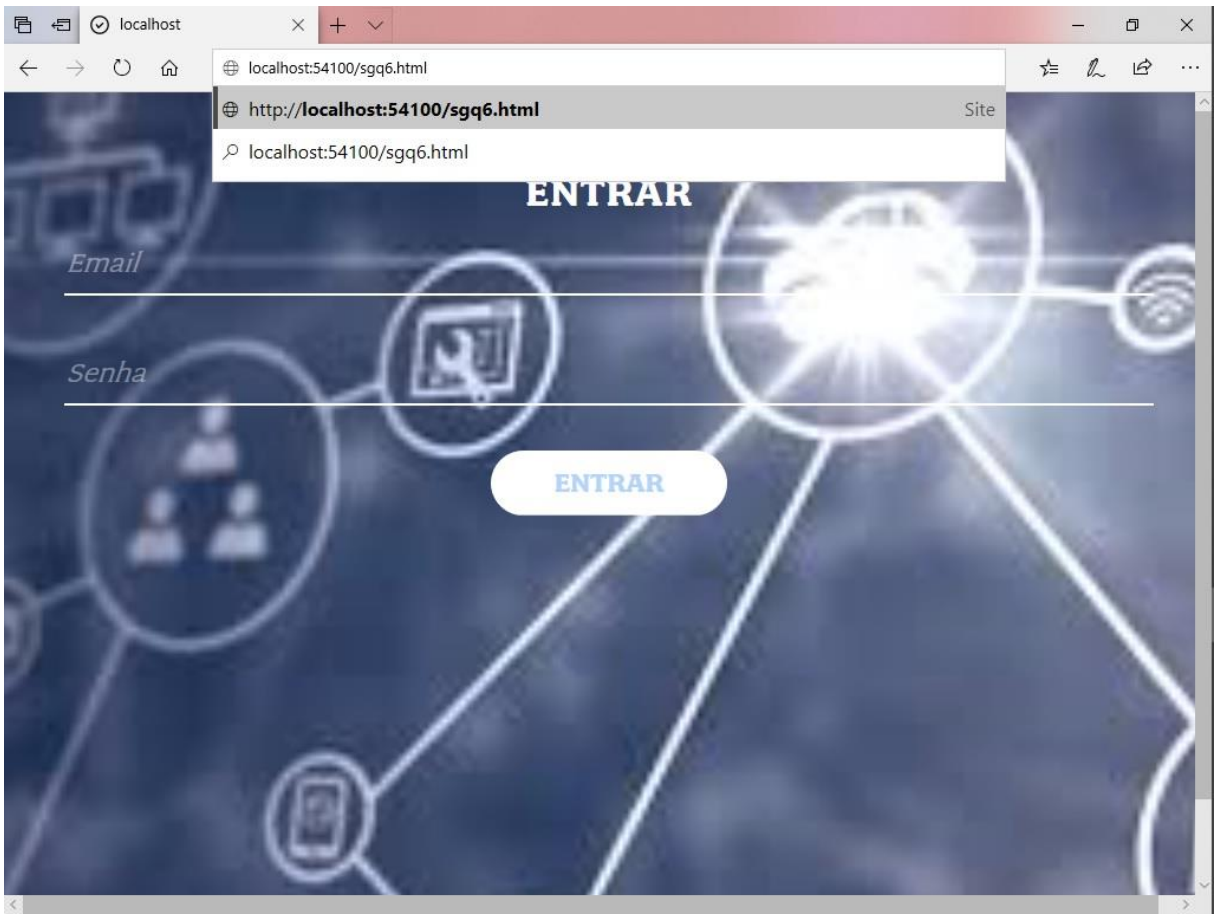

Figura 14 - Tentativa de inserir a URL da aplicação diretamente no navegador.

Na Figura 14 é apresentada a interface de *login* com a tentativa de inserir uma URL diretamente da aplicação, na qual o usuário após pressionar <Enter> é redirecionado para a interface de *login* (Figura 12) novamente.

Cenário "Calcular Indicadores": As evidências de aceitação para este cenário são:

- Tempo de resposta inferior a 1000 milissegundos para uma requisição de cálculo de indicadores;
- Tempo inferior a 1000 milissegundos para a transação total.

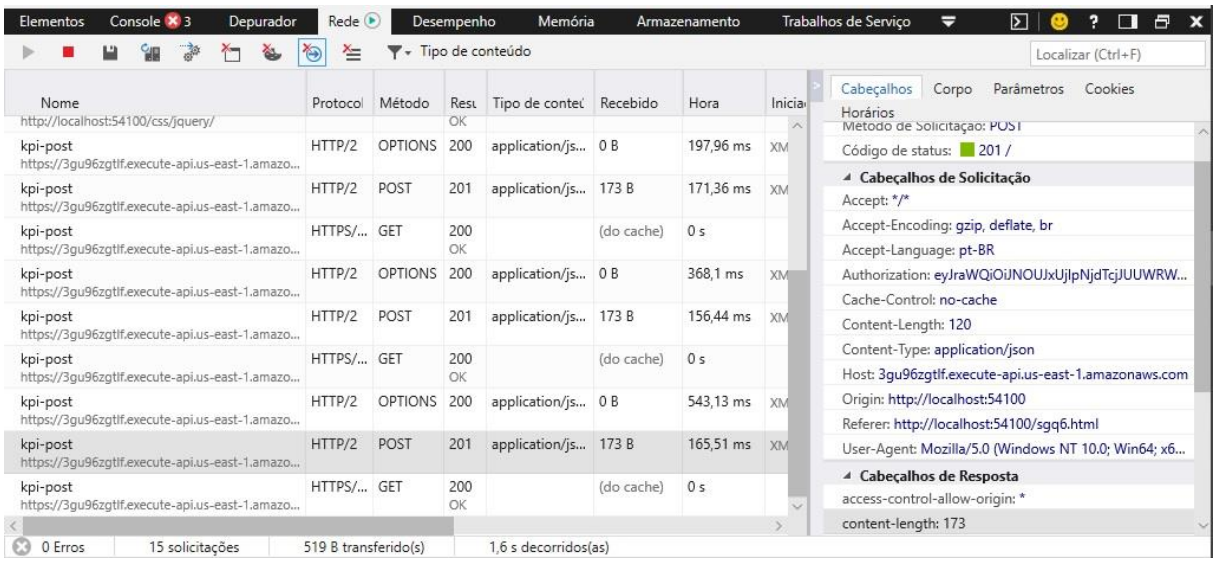

Figura 15 - Interface de "Ferramentas do desenvolvedor - Rede"

A evidência deste cenário apresenta o tempo das requisições para o back-end da aplicação, sendo possível observar requisições inferiores a 1 segundo.

É importante lembrar que existem diversos fatores que influenciam estas requisições como largura de banda de internet, região do servidor de back-end e quantidade de dados transmitidos. Porém, estes servem como base para os demais módulos da aplicação e também para possíveis estudos de escalabilidade do sistema.

Uma observação a ser feita além das evidenciadas na Figura 15 são as características de inserção do teste, como:

- Região de execução da API: us-east-1 (Norte da Virgínia);
- Região de execução da Interface: us-east-1 (Norte da Virgínia);
- Dispositivo de hardware de rede utilizado: Wi-fi Qualcomm Atheros AR956x;
- Banda de internet: 200 MB;
- Origem do ponto de requisição: Brasil, Estado de São Paulo;
- Browser utilizado: Microsoft Edge.

Cenário "Armazenar Indicadores": As evidências de aceitação para este cenário são:

- Ao interceptar as mensagens trafegadas entre a aplicação e o servidor as mesmas devem estar criptografadas.

- O envio e a recepção das mensagens devem ser de forma segura, utilizando a segurança da camada de transporte (TSL) para tal.

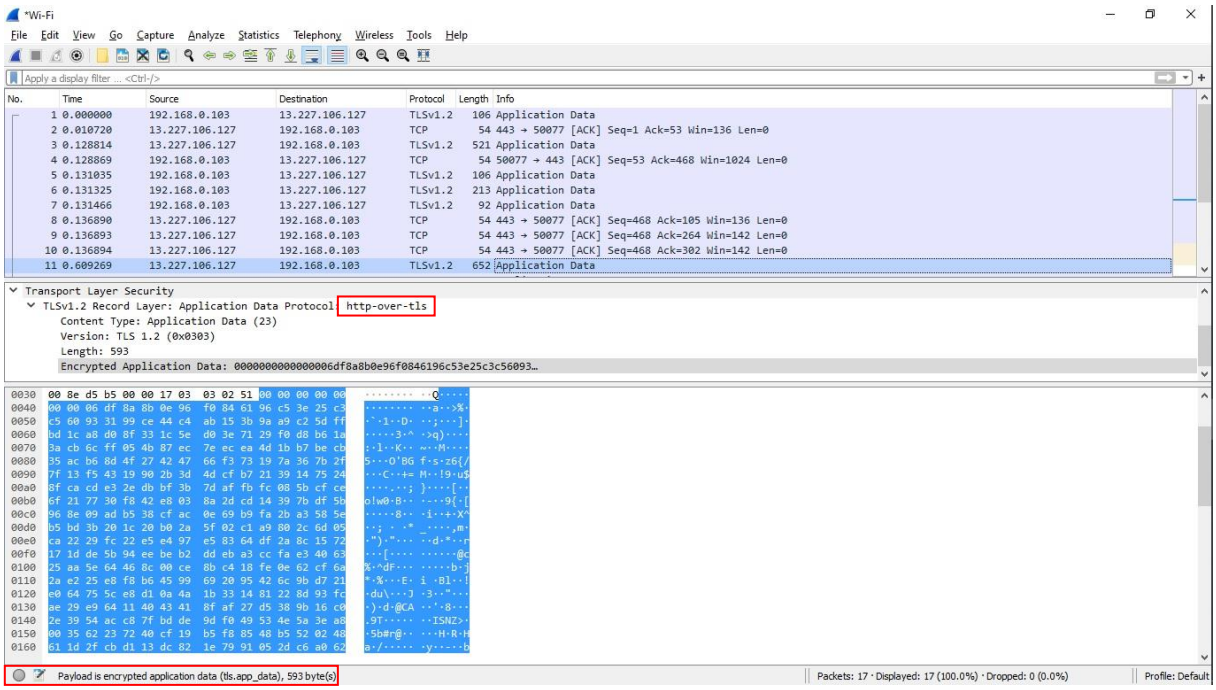

Figura 16 - Pacotes no Wireshark

Para este cenário, a evidência foi obtida por meio de aplicativo de análise de protocolo de rede, na qual é possível observar as etapas de negociação com o servidor para transmissão dos dados pela camada de segurança utilizando o protocolo de transferência http sobre a camada de segurança TLS.

É possível observar também todo o *payload* da mensagem criptografada, sendo possível a obtenção dos dados somente em posse da chave gerada para essa transação e conhecendo o algoritmo acordado entre o servidor e a aplicação durante o *handshake*.

Cenário "Estimativas de custos mensais": As evidências de aceitação para este cenário são:

- Ao entrar no console AWS deve ser possível visualizar graficamente as estimativas mensais;
- O sistema deve possibilitar a visualização dos custos por recurso utilizado;
- Os custos devem ser informados na moeda dólar.

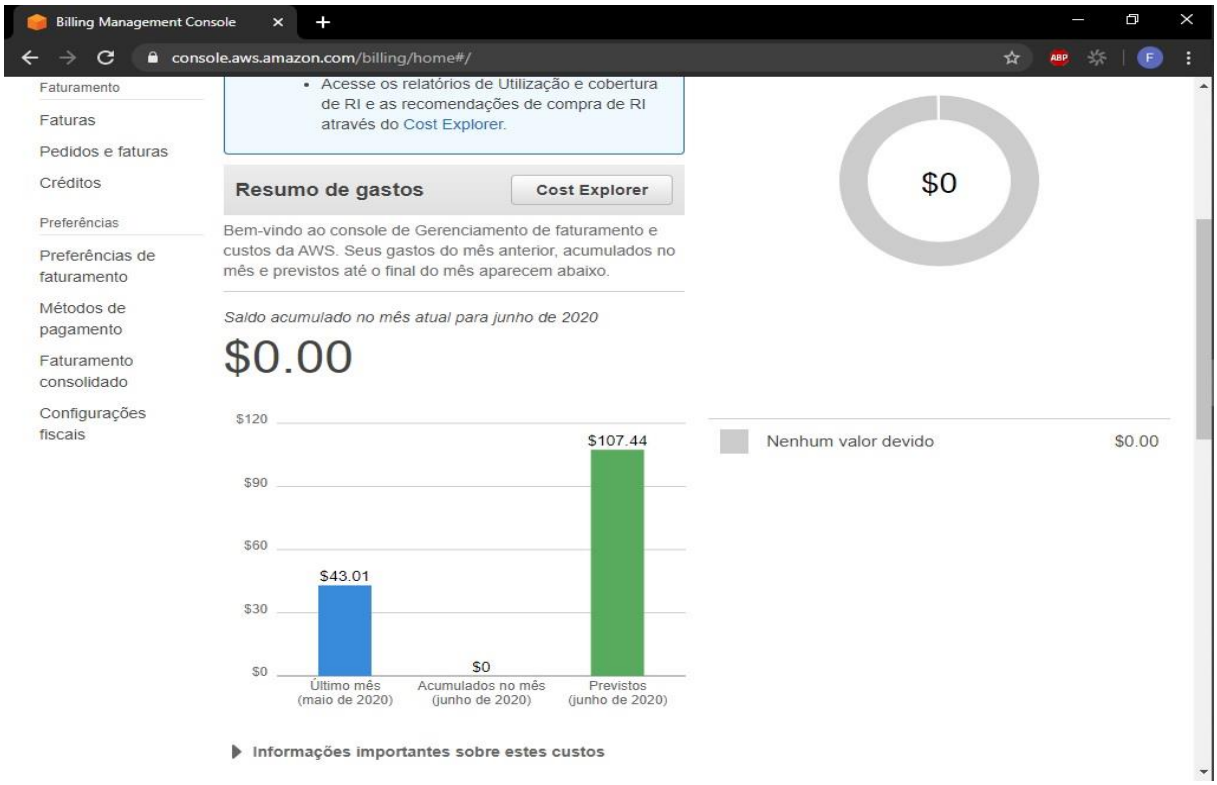

Figura 17 - Explorador de custos no AWS Console

Para evidenciar a possibilidade de previsão de custos a Figura 17 apresenta a área de faturamento dos recursos da aplicação, sendo possível prever qual será o valor para o próximo mês, mediante ao uso do mês corrente e anteriores.

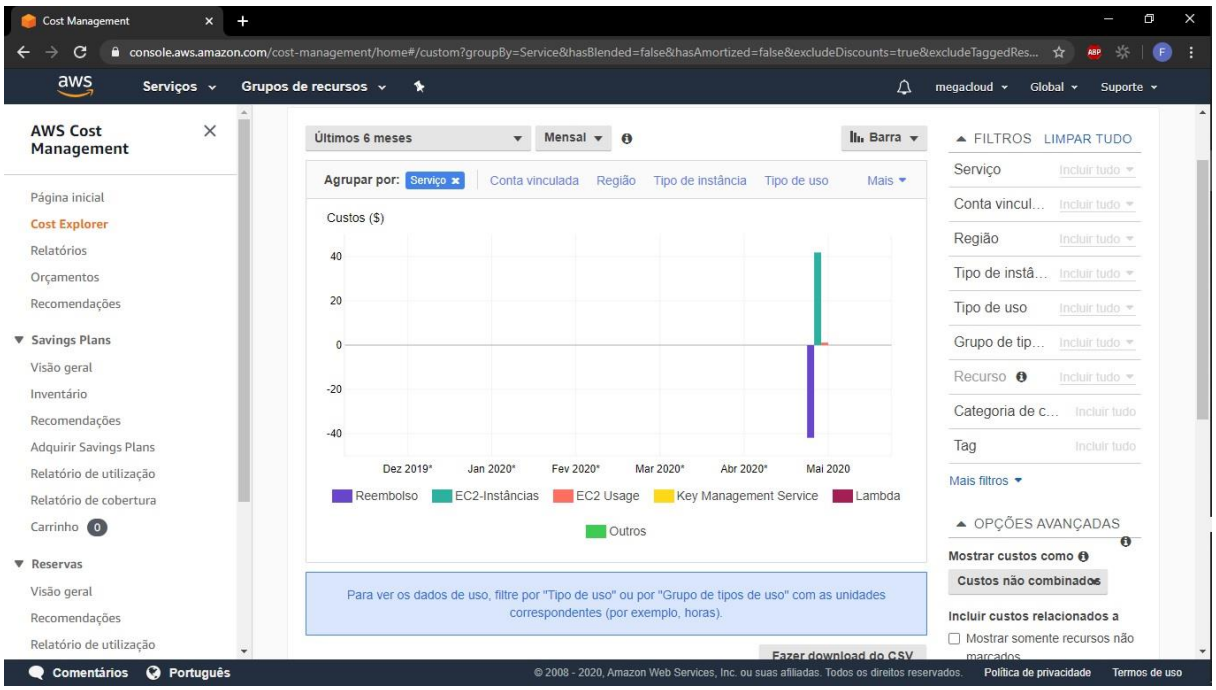

Figura 18 - Custos por recursos.

A importância desta evidência está na possibilidade de visualizar quais os recursos de computação estão sendo mais utilizados, possibilitando tratativas para redução de custos após as implantações iniciais ou até mesmo em criação de novas *releases* da aplicação.

Cenário "Requisitar indicadores": As evidências de aceitação para este cenário são:

- Deve ser possível requisitar as informações de indicadores por meio de requisições à API da aplicação.

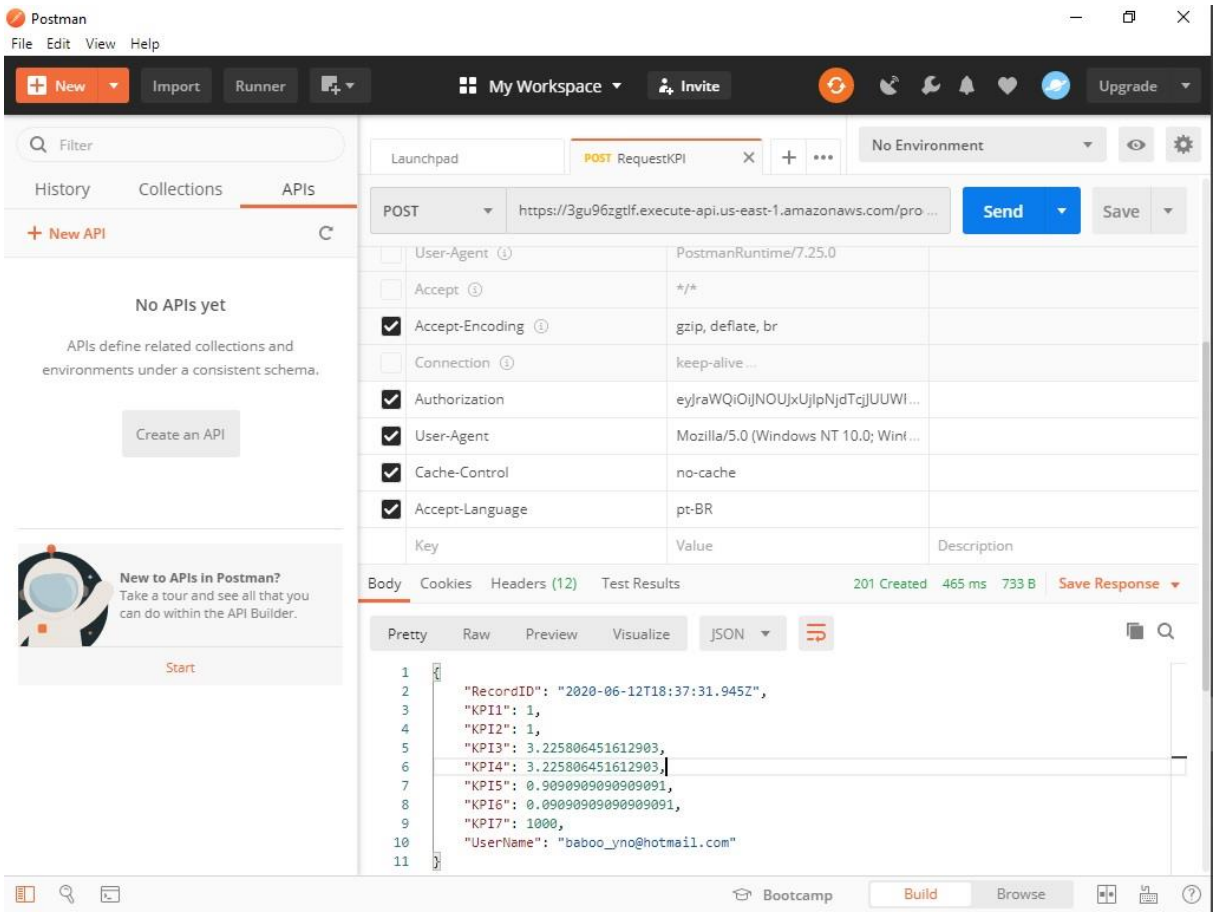

Figura 19 – Retorno da requisição de indicadores no aplicativo Postman.

Para evidenciar este cenário foi utilizado um aplicativo para desenvolvimento de APIs, o PostMan, para criar a solicitação dos KPIs à aplicação.

Na Figura 19 é possível observar a estrutura JSON retornada pela aplicação contendo os dados de KPI, o ID gerado da requisição e a informação de usuário relacionado à autorização enviada na requisição.

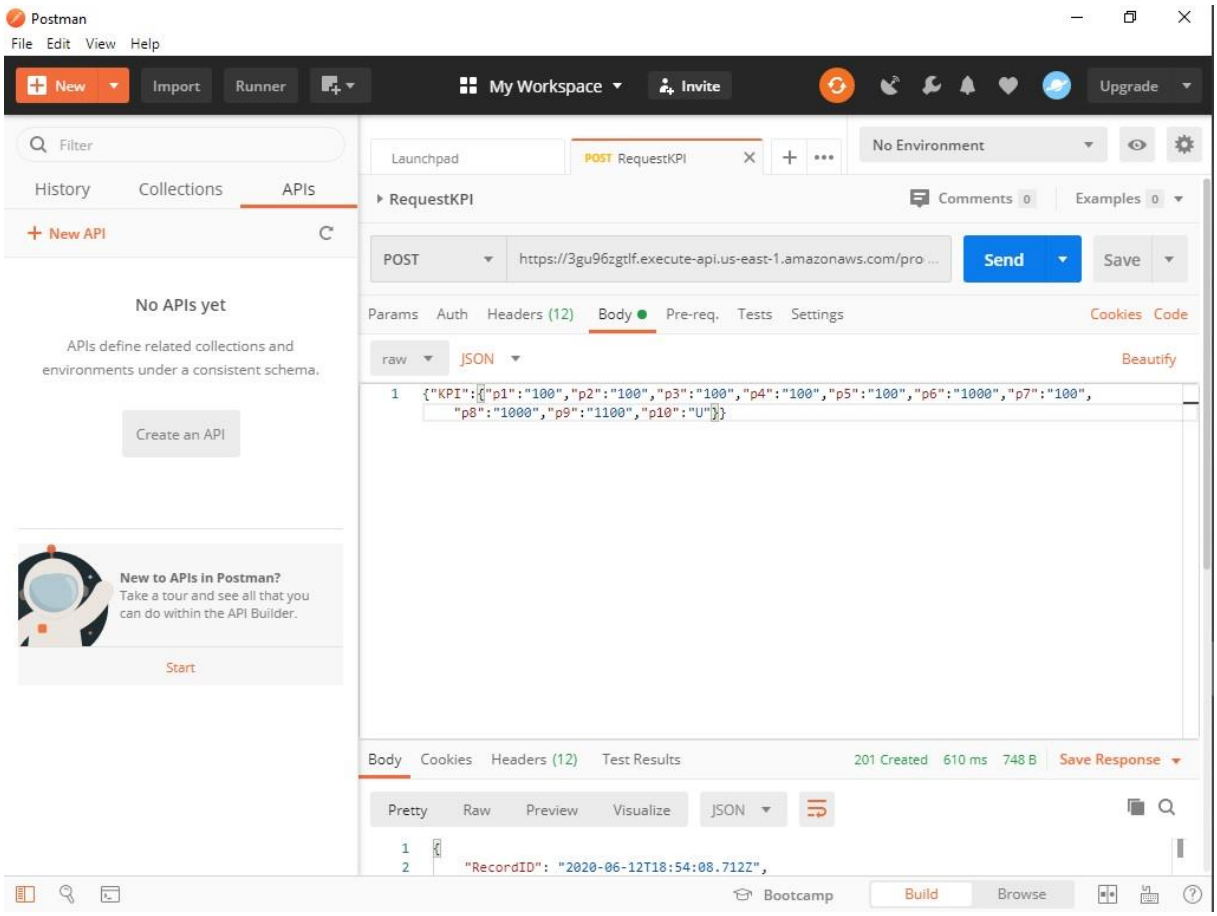

Figura 20 - Mensagem de requisição de indicadores para a aplicação.

Na Figura 20, no campo *Body* está a estrutura de requisição contendo o parâmetro p10, o qual determina a requisição dos indicadores do dia, sendo retornados os dados da tabela de indicadores da aplicação.

O uso de um aplicativo genérico evidencia também que a aplicação é genérica e funciona nos padrões da tecnologia de uso de APIs.

6.4. Resultado

A arquitetura proposta neste trabalho e os requisitos validados visam à criação de um sistema como ferramenta para práticas da norma de gestão da qualidade. Desde o módulo de comunicação, que pode ser utilizado para facilitar o engajamento das pessoas nas atividades rotineiras ou criações de ações. Até a gestão de documentos, onde é possível centralizar relatórios e procedimentos de auditoria e utilizá-los para manter um ambiente que facilita a qualidade em processos diários.

A arquitetura proposta, também, foi dirigida para facilitar o trabalho remoto, tendo o módulo de comunicação, como meio para agendar reuniões por vídeo conferência e facilitar o fluxo de comunicação, que diversas vezes é realizado por ferramentas externas, não controladas e descentralizadas, como e-mail e ferramentas de videoconferência de provedores diferentes.

A evidenciação dos componentes não foi quantificada em termos de números a serem seguidos fielmente para o desenvolvimento, porém foram minimamente elucidados nos diagramas de implantação e de componentes, podendo se desdobrar em mais componentes, caso necessário.

Os requisitos não funcionais aplicados escolhidos tiveram a preocupação em termos de segurança, a qual pode ser comprovada por apresentação breve da criptografia dos protocolos utilizados nas trocas de mensagens, dos mecanismos de autenticação e de encerramento de aplicação. Porém, existem outros mecanismos de segurança que não foram investigados na prova de conceito, como autenticações do tipo MFA (Multi-factor authentication), encerramento de sessão por inatividade e desenvolvimento de criptografia própria.

Para o requisito não funcional de tempo de requisições não foi aprofundado o estudo do tema de recursos de roteamento da AWS para remanejo de localização de aplicações, o qual se faz mais necessário caso exista um grande fluxo e necessidade de computação mais próxima do usuário.

Para o requisito de manutenção foram explorados recursos inerentes ao modelo provido pela AWS. A escolha do custo de manutenção para a prova de conceito foi realizada por ser um recurso que facilita na tomada de decisões em relação aos *stakeholders*, uma vez que se trata muitas vezes de uma restrição arquitetural imposta pelos mesmos, o orçamento.

Outra motivação para a avaliação dos custos mensais é a promoção da aplicação, uma vez que o cliente pode se beneficiar da projeção destes custos.

O requisito de interoperabilidade foi demonstrado de forma genérica, possibilitando a inserção manual pelo sistema SGQ ou até mesmo por outros módulos de obtenção dos parâmetros utilizados na construção de indicadores, como sistemas financeiros independentes, sistemas empregados em métricas de produção e até mesmo módulos ERP. Uma melhoria clara é a possibilidade de requisição por data definida por parâmetro de *input* da API.

A elaboração da POC deste trabalho possibilitou a avaliação da arquitetura em profundidade de camadas, validando as etapas propostas de construção do sistema.

Alguns dos mecanismos inseridos na arquitetura, como o uso de publicações e alertas, foram pensados mediante a vivência em indústrias que estão migrando ou ainda se utilizam de métodos manuais para informar e/ou veicular as informações de realizações de processos diários, o que faz com que a informação dos processos não tenha uma rápida fluidez e gera uma possibilita de falha na comunicação.

7. Conclusão

Assim como na arquitetura de casas, edifícios, áreas urbanas e afins, o arquiteto deve ter a preocupação dos detalhes como forma de unir requisitos, sejam eles do cliente que contratou os serviços, dos usuários dessas áreas ou imóveis e das restrições impostas pelo ambiente em que este projeto esta inserido, de forma que todos os envolvidos estejam satisfeitos com o resultado final do que a arquitetura propõe.

Em um sistema de software não é muito diferente, os requisitos funcionais tentam englobar o que o usuário necessita ou o mercado demanda para competitividade. Os não funcionais são aqueles que o usuário muitas vezes não tem conhecimento direto, mas impactam diretamente na operação do sistema. E as restrições, como o próprio nome diz, são os limitantes da arquitetura que vão desde orçamentos limitados até códigos legados que necessitam ser mantidos em novos sistemas propostos.

Este trabalho atendeu a proposta de criar um projeto arquitetural, uma vez que usou de recursos de imagens, textos e diagramas para propor uma solução global e comum visando evidenciar alguns dos principais problemas encontrados durante o desenvolvimento, os quais

podem impactar de maneira desastrosa caso não sejam antevistos em documentação arquitetural.

REFERÊNCIAS

AWS Documentatiom. Amazon. Disponível em: <https://docs.aws.amazon.com/>. Acesso em: 10 de janeiro de 2020.

ABNT NBR ISSO 9001:2015. ABNT. Disponível em: <https://www.abntcatalogo.com.br/sebrae/norma.aspx?ID=345041> Acesso em: 01 de dezembro de 2019.

APÊNDICES

Dados da POC	
URL da POC	http://meubaldevirginia.s3-website-us-east-1.amazonaws.com/
Usuário	puc@puc
Senha	puc123
URL dos arquivos + Vídeos	https://github.com/Babooyno/pocpuc.git

CHECKLIST PARA VALIDAÇÃO DOS ITENS E ARTEFATOS DO TRABALHO

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?			
2	Os objetivos do trabalho (objetivos gerais e pelo menos três específicos) foram especificados?			
3	Os requisitos funcionais foram listados e priorizados?			
4	Os requisitos não funcionais foram listados identificados usando o estilo estímulo-resposta?			
5	As restrições arquiteturais foram definidas?			
6	Os mecanismos arquiteturais foram identificados?			
7	O diagrama de caso de uso foi apresentado junto com uma breve descrição de cada caso de uso?			
8	O modelo de componente e uma breve descrição de cada componente foi apresentada?			
9	O modelo de implantação e uma breve descrição de cada elemento de hardware foi apresentada?			
10	Prova de conceito: uma descrição da implementação foi feita?			
11	Prova de conceito: as tecnologias usadas foram listadas?			
12	Prova de conceito: os casos de uso e os requisitos não funcionais usados para validar a arquitetura foram listados?			
13	Prova de conceito: os detalhes da implementação dos casos de uso (telas, características, etc) foram apresentadas?			
14	Prova de conceito: foi feita a implantação da aplicação e indicado como foi feita e onde está disponível?			
15	As interfaces e/ou APIs foram descritas de acordo com um modelo padrão?			
16	Avaliação da arquitetura: foi feita uma breve descrição das características das abordagens da proposta arquitetural?			
17	Avaliação da arquitetura: Os atributos de qualidade e os cenários onde eles seriam validados foram apresentados?			
18	Avaliação da arquitetura: a avaliação com as evidências dos testes foi apresentada?			
19	Os resultados e a conclusão foi apresentada?			
20	As referências bibliográficas foram listadas?			
21	As URLs com os códigos e com o vídeo da apresentação da POC foram listadas?			

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?			
2	Todos os casos de uso estão contemplados na lista de requisitos funcionais?			
3	Os requisitos não funcionais, mecanismos arquiteturais e restrições c arquiteturais estão coerentes com os modelos de componentes e implantação?			
4	Os modelos de componentes e implantação estão coerentes com os requisitos não funcionais, mecanismos arquiteturais e restrições arquiteturais?			
5	As tecnologias listadas na implementação estão coerentes com os requisitos não funcionais, mecanismos arquiteturais e restrições arquiteturais?			
6	Os casos de uso e os requisitos não funcionais listados na implementação estão coerentes com o que foi listado nas seções anteriores?			
7	Os atributos de qualidade usados na avaliação estão coerentes com os requisitos não funcionais na cessão três?			
8	Os cenários definidos estão no contexto dos casos de uso implementados?			
9	O apresentado no item resultado está coerente com o que foi mostrado no item avaliação?			