DESENVOLVIMENTO DE UM ASSISTENTE VIRTUAL PARA AUTOMAÇÃO DE AGENDAMENTOS DE TREINAMENTOS E TAREFAS CORPORATIVAS[1]

Thiago Fernandes Griebler[2]

RESUMO

O presente trabalho apresenta o desenvolvimento de um assistente virtual para o setor de qualidade corporativa de uma empresa atuante do mercado de refrigeração e eletrodomésticos. O assistente virtual utiliza as ferramentas disponibilizadas no *Google Workspace* e *Google Cloud* e é responsável por marcar treinamentos, dar informações diversas sobre treinamentos e executar tarefas administrativas para usuários que possuam liberação para tal. Dessa forma, busca-se uma alternativa para reduzir o tempo utilizado com atividades que não beneficiam o alcance de metas do setor e o desenvolvimento de uma cultura digital, onde entende-se que a máquina pode ser aplicada às mais diversas atividades. Outro objetivo do desenvolvimento da ferramenta é servir de base para implementações digitais mais robustas, onde levará em consideração conceitos de IHC, para aproximar o trabalhador à produção, com acesso mais facilitado a dados e ao mesmo tempo apresentar uma interface mais amigável e intuitiva que permite uma abrangência de usuários maior.

Palavras-chave: IHC; Chatbot; Qualidade; Automação; Indústria.

ABSTRACT

This work presents the development of a virtual assistant for the corporate quality sector of a company operating in the refrigeration and household appliances market. The virtual assistant uses the tools available on Google Workspace and Google Cloud and is responsible for scheduling training, providing various information about training and carrying out administrative tasks for users who are authorized to do so. In this way, an alternative is sought to reduce the time spent on activities that do not benefit the achievement of sector goals and the development of a digital culture, where it is understood that the machine can be applied to the most diverse activities. Another objective of developing the tool is to serve as a basis for more robust digital implementations, which will take into account HCI concepts, to bring the worker closer to production, with easier access to data and at the same time present a more friendly and intuitive interface that allows a wider range of users.

Keywords: IHC; Chatbot; Quality; Automation; Industry.

---

[1] Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do grau de bacharel no Curso de Ciência e Tecnologia, do Centro Tecnológico de Joinville (CTJ), da Universidade Federal de Santa Catarina (UFSC), sob orientação do(a) Dr.(a) Benjamin Grando Moreira.

[2] Graduando como Bacharel em Ciência e Tecnologia. E-mail: thiago.griebler@outlook.com

1. INTRODUÇÃO

O setor de Qualidade Corporativa da companhia onde este trabalho é aplicado possui como uma de suas tarefas o treinamento dos funcionários das plantas globais para a utilização de ferramentas de qualidade e metodologias *Lean Six Sigma*. Esses treinamentos são importantes para capacitar os colaboradores da empresa à desempenhar atividades que envolvam desenvolvimento de projetos e solução de problemas seguindo conceitos globais de qualidade, gerando dessa forma uma uniformidade na coleta, criação e apresentação de dados.

A uniformidade é crucial para garantir que as informações sendo transmitidas entre as plantas à nível global sejam precisas, consistentes e confiáveis. Desse modo, decisões estratégicas, otimização de processos e oportunidades de melhoria podem ser implementadas de maneira mais satisfatória e com margem de erro reduzida. Esses aspectos reforçam a cultura *Lean* dentro da companhia, que visa a eliminação de desperdícios, focando na eficiência e na melhoria contínua. Assim a produtividade é maximizada e os custos reduzidos, garantindo preços competitivos no mercado consumidor.

Entretanto, com a alta demanda de trabalho e a saturação da mão de obra, muitas vezes a organização e gerenciamento dessas atividades acaba se tornando complicado. Desse modo foi desenvolvido um assistente virtual baseado na plataforma *Google Workspace*, que gerenciará os pedidos de treinamentos realizados por líderes de setores, através do *Google Chat*, os salvando em um banco de dados e a partir dele distribuindo notificações aos líderes responsáveis por cada ferramenta de qualidade, da Qualidade Corporativa. Também, os usuários podem poder conferir quais treinamentos estão sendo oferecidos e quais são os instrutores. Além disso, existe a possibilidade, para os usuários definidos como administradores, de realizar diversas atualizações e alterações no banco de dados, através do assistente. O assistente virtual tem seu código hospedado na *Google Cloud* e baseado em *Google Apps Script*, linguagem similar ao *Javascript,* rodando com uma *API* do *Google Chat*.

O principal resultado obtido do desenvolvimento, foi a criação de uma base sólida de automação que pode ser direcionada para diversas atividades desenvolvidas de maneira abrangente, como: Gerenciamento de clientes, consulta de indicadores de produção, coleta de dados e análise de dados.

2. FUNDAMENTAÇÃO TEÓRICA E TÉCNICA

Como elementos para a fundamentação teórica deste trabalho serão apresentados os conceitos de Interface Humano Máquina (IHC), a plataforma Google Workspace, a linguagem de programação Google Apps Script, definições de *chatbots* e APIs.

2.1. IHC – Interface Humano Máquina

Segundo Oliveira (2015) a interface é a maneira que o usuário tem de avaliar o sistema. A ele não interessa a linguagem de programação na qual o sistema foi desenvolvido, o tipo de equipamento utilizado para realizar esse desenvolvimento e nem a metodologia utilizada para conceber o sistema. Esse conceito não se limita apenas a beleza estética da tela, mas também o tempo de resposta do sistema, grau

de dificuldade de uso, corretude das informações apresentadas, nível de erros que podem ser cometidos pelo usuário na utilização, acessibilidade, entre outros.

Oliveira (2015) define para desenvolver sistemas com tais características é necessário entender não somente como o computador funciona, mas também como o ser humano pensa. Desse modo, conclui-se que *IHC* é a ciência que busca a diminuição da barreira entre usuário e máquina, visando uma interação cada vez mais orgânica e acessível ao indivíduo, mas que também não seja livre demais, possibilitando uma menor quantidade de erros e problemas envolvendo dados e a utilização das ferramentas disponíveis.

2.1.2. Critérios para Medir a Qualidade de Interfaces Humano – Computador

Para Shneiderman (1997 apud OLIVEIRA, 2015) existem cinco fatores cruciais para se avaliar a qualidade de uma interface humano – computador, sendo eles: tempo para aprender, performance, taxas de erro, tempo de retenção, satisfação subjetiva.

Oliveira (2015) define tempo de espera como uma medida simples e direta que mede o tempo médio para um usuário típico aprender a utilizar o sistema e executar suas tarefas. Quanto menor for o tempo de espera, menos recursos deverão ser implementados para realizar o treinamento do usuário utilizador do sistema.

Posteriormente, performance, é definida por Oliveira (2015) como o tempo gasto para o usuário médio realizar uma atividade representativa do domínio da aplicação. Ou seja, quanto mais execuções o usuário conseguir realizar no período tradicional de utilização, melhor para a companhia que implementa esse sistema.

O terceiro critério, taxas de erro, é determinado por Oliveira (2015) como um balanço entre a performance e quantidade de erros cometidos pelo usuário na execução, de modo que pequenos erros em troca de performance são aceitáveis, desde que sejam facilmente reversíveis.

Tempo de retenção, por sua vez, estipula-se como a métrica que indica o tempo que o usuário mantém o conhecimento de como utilizar a aplicação (OLIVEIRA, 2015). O tempo de retenção está relacionado com o tempo de aprendizado e a frequência de uso, de modo que um sistema mais simples permite naturalmente que o usuário mantenha o conhecimento técnico da ferramenta utilizada por ele, por mais tempo (OLIVEIRA, 2015).

O último critério, satisfação subjetiva, diz respeito à experiência que o usuário tem ao utilizar um aplicativo para realizar determinado tipo de tarefa. Mede-se a insatisfação do usuário originada a partir da falta de suporte do sistema para realizar determinadas atividades, por não encontrar as opções que ele procura e também por sentir que não possui controle da ferramenta, por exemplo (OLIVEIRA, 2015). Para medir essa informação, utiliza-se pesquisas de satisfação e de concordância (OLIVEIRA, 2015).

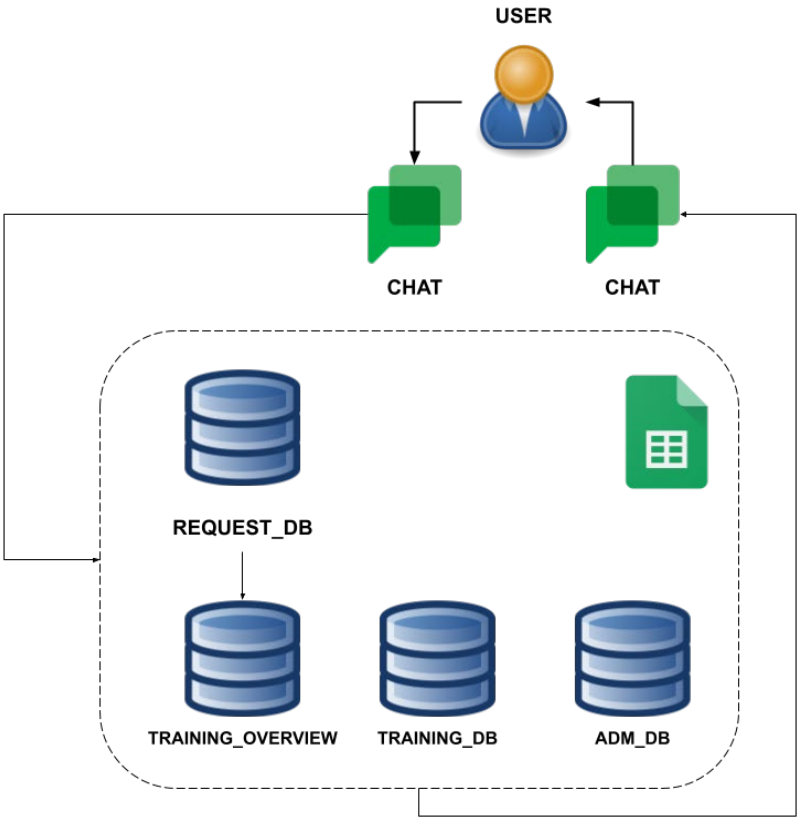
2.2. Google Workspace

Iyer e Jeevaganambi definem (2022) o *Google Workspace* como um conjunto de ferramentas que visa permitir os usuários manterem a produtividade através da conclusão de tarefas de maneira eficiente enquanto se mantêm conectados aos seus colegas de trabalho. Esses serviços permitem, também, que o usuário acesse as informações de qualquer lugar e de qualquer dispositivo, desde

que possua internet. Para a criação do assistente virtual, utilizaram-se três aplicativos do *Google Workspace*, sendo eles: *Google Cloud, Google Sheets e Google Chat.*

O *Google Chat* é a plataforma onde ocorre a interação do usuário com o *chatbot*. É possível definir, dessa maneira, o *chat* como um intermediador entre o banco de dados e o usuário, tanto para consultas de informações como para registros no mesmo. Esse banco de dados é uma planilha do *Google Sheets* com múltiplas abas, cada uma com uma função. A Figura 1 apresenta um diagrama representando o banco de dados.

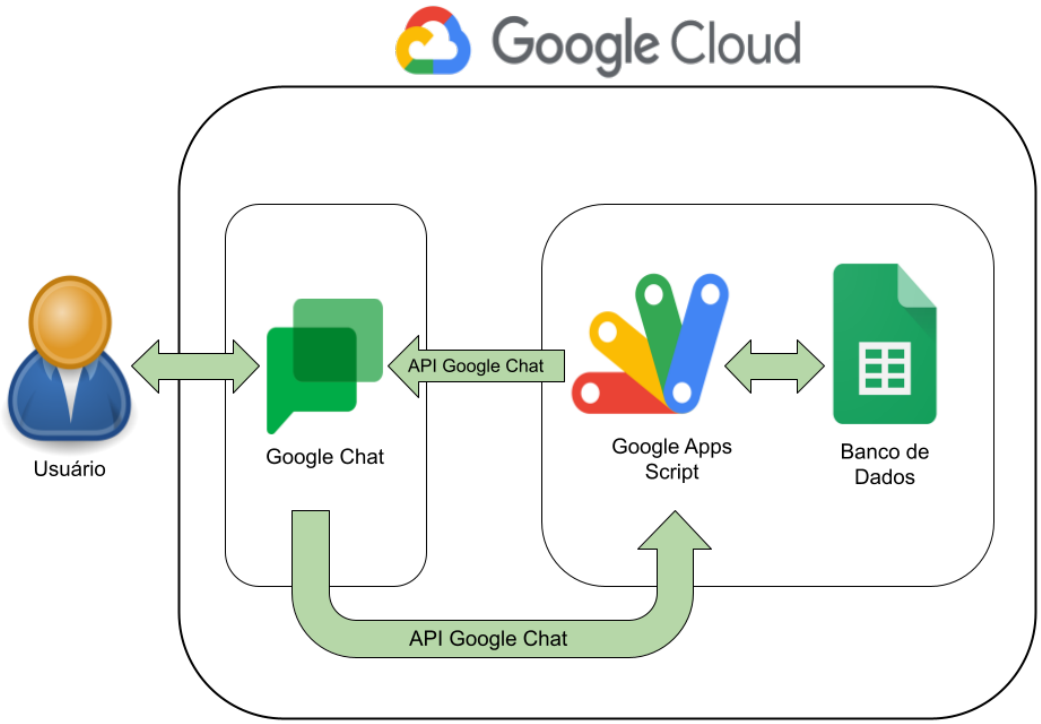
Figura 1 - Digrama de Banco de Dados



Fonte: O Autor (2023).

Entretanto, essa interação entre *chat* e banco de dados é inviável de maneira completamente direta, desse modo é necessário hospedar um projeto dentro do *Google Clouds* que una o *script* ao *Google Chats* através de uma *API*. O diagrama da Figura 2 demonstra como é a relação das ferramentas e como é feita a interação entre elas:

Figura 2 - Processo de Comunicação e Troca de Dados Entre as Ferramentas que Compõem o Assistente Virtual



Fonte: O Autor (2023).

2.3. Google Apps Script

Google Apps Script é uma plataforma de aplicação e desenvolvimento dentro do *Google Apps*, permitindo o usuário a adicionar funcionalidades à planilhas, *Gmail, Google Sites* e outros serviços digitais prestados pela Google (FERREIRA, 2014).

Para Ferreira (2014) existem algumas vantagens em possuir o *script* armazenado na plataforma *Google Apps*. A primeira vantagem é a segurança digital já estar integrada, removendo a necessidade de criação de *patches* de atualização de segurança para servidores legado e a constante necessidade de monitoramento para prevenir ataques maliciosos. Outra vantagem é a plataforma ser *web based* e não requere transferência de arquivos, múltiplos *backups*, controles de revisão, *uploads* para servidores de produção, atualização de *softwares* de desenvolvimento e diversos outros aspectos relacionados à programação que devem ser tratados e gerenciados durante o processo de criação de uma ferramenta digital.

É comum o *Google Apps Script* (GAS) ser comparado ao *Visual Basic for Applications* (VBA), que é a solução para desenvolvimento de macros e automações para o pacote *Office* da *Microsoft*. Há duas principais características que são consideradas favoráveis à utilização do GAS ao invés do VBA. O VBA é dependente do versionamento do software do pacote office que está sendo utilizado. Desse modo, quando há o lançamento de um *major patch* de atualização do *Office*, há o risco de o macro ou script que está rodando no software perder o suporte parcialmente ou totalmente, acarretando em erros nos processos que eles

executam. Outra grande questão é a independência de plataformas. Quando *scripts* VBA são criados na plataforma *Windows*, há grande chance deles não funcionarem, por exemplo, na plataforma MAC e vice e versa. Já os *macros* desenvolvidos com GAS funcionam totalmente independente da plataforma na qual foram produzidos (FERREIRA, 2014, tradução nossa).

Desse modo, analisando as vantagens descritas acima e a empresa na qual o assistente virtual está sendo implementado, possuir toda plataforma do *Google Workspace* amplamente disponível dentro da companhia, optou-se pela utilização do GAS como plataforma principal de desenvolvimento.

2.3.3. Limitações do Google Apps Script (W.I.P)

O GAS roda nos servidores da *Google*, e dessa maneira não pode rodar por tempo indeterminado. O tempo limite para retorno das funções é de seis minutos(RAMMALINGAN, 2016). Assim, estratégias devem ser empregadas para reduzir o tempo de atividade do script. Para evitar esse problema, para o assistente virtual, foi utilizada uma estratégia que utiliza estados de usuário para minimizar as execuções do *script.* A definição dessa solução pode ser observada na Figura 3

Figura 3 - Função *onMessage*

```
/**
 * Responds to Google Chat.
 *
 * @param {Object} event the event object from Google Chat
 */
function onMessage(event){
  var userProperty = PropertiesService.getUserProperties();
  estadoUsuario = userProperty.getProperty('estadoUsuario');
  let mensagem = "";
```

Fonte: O Autor (2023).

A função *onMessage* é a função principal da aplicação e roda a partir de um evento na janela de bate papo do *Google*. A cada ciclo o *script* capta qual o valor do atributo *estadoUsuario* e a partir dele roda determinados comandos específicos para aquele estado.

2.4. API

API é a sigla em inglês para *Application Programming Interface*, ou Interface de Programação de Aplicações. As interfaces de programação de aplicativos (APIs) são conjuntos de ferramentas, definições e protocolos para a criação de aplicações de software. APIs conectam soluções e serviços, sem a necessidade de saber como esses elementos foram implementados (RED HAT, 2022).

De forma mais simples, as APIs podem ser definidas, de maneira figurada, como uma ponte entre aplicações, integrando as mais distintas funcionalidades. As APIs representam uma forma ágil e segura de integrar sistemas com linguagens totalmente distintas. Ao contrário de outras abordagens, que exigem instalações complexas para existir compatibilidade, as APIs permitem uma conexão direta e eficiente entre sistemas, acelerando os processos de negócios e a produtividade das

companhias. Essa flexibilidade oferece aos desenvolvedores de softwares e aplicativos a capacidade de unir tecnologias variadas, como diferentes bancos de dados, por exemplo. Além disso, possibilita a utilização de funcionalidades específicas de um aplicativo dentro de outros, simplificando a integração e ampliando as possibilidades de desenvolvimento sem obstáculos (VERTIGO, 2015).

2.5 Chatbots

Segundo Junior e Carvalho (2018) *Bot*, abreviação de *robot*, são softwares desenvolvidos para imitar ações humanas, repetidas vezes, e simular uma interação humano-computador. Os primeiros *bots* eram mais simples e com pouca interação. Atualmente, eles fazem uma análise prévia de necessidades do usuário para auxiliá-lo com uma interação de forma mais natural possível.

A principal característica de um *bot* é o seu código desenvolvido especialmente para automatizar algumas funções do cotidiano, em especial, a interação com humanos, podendo, inclusive, desempenhar ações inerentemente humanas e, assim, passar-se por pessoas durante a realização destas atividades (JUNIOR; CARVALHO, 2018).

2.5.1. Diferença entre Inteligência artificial e o Assistente Virtual

Apesar de simular uma abordagem humana em uma conversa, o assistente virtual não possui capacidade de tomada de decisões. Ele funciona como uma ferramenta guiada pelo usuário que automatiza processos que previamente seriam desenvolvidos por um humano. Inteligências artificiais, segundo Norving e Russel (NORVING; RUSSEL apud JUNIOR;CARVALHO, 2018) desafiam o Teste de Turing, que define quais características um sistema inteligente, de fato, deve apresentar. Esse sistema deve ser capaz de processar a linguagem de forma natural e responder às interações da mesma forma, representar o conhecimento que para ele é passado, responder perguntas e obter novas conclusões a partir dos dados armazenados e ser capaz de se adaptar e extrapolar os limites para achar novas soluções para os problemas que enfrentar.

Quando uma análise do assistente virtual é realizada tendo em vista essas características, percebe-se a distância da ferramenta para uma verdadeira inteligência artificial. Os principais fatores que limitam o *chatbot* é incapacidade de obter conclusões diferentes, a partir dos dados armazenados, e a impossibilidade de extrapolar a sua área de atuação. O assistente apenas retorna as informações que foram previamente determinadas em seu código e não consegue cruzar dados para obter outra informação, se ele não foi previamente desenvolvido para isso. Ele, também, é incapaz de otimizar sua própria atividade de forma direta. Sempre seguirá o mesmo caminho para realizar uma tarefa a não ser que o código fonte seja alterado para isso.

Desse modo observa-se claramente que não há inteligência, de maneira figurada, nas atividades executadas pelo assistente. Ele apenas segue um padrão lógico preestabelecido que é repetido um número infinito de vezes trazendo informações de um ambiente não acessível ao usuário, para a tela.

3. METODOLOGIA DE DESENVOLVIMENTO

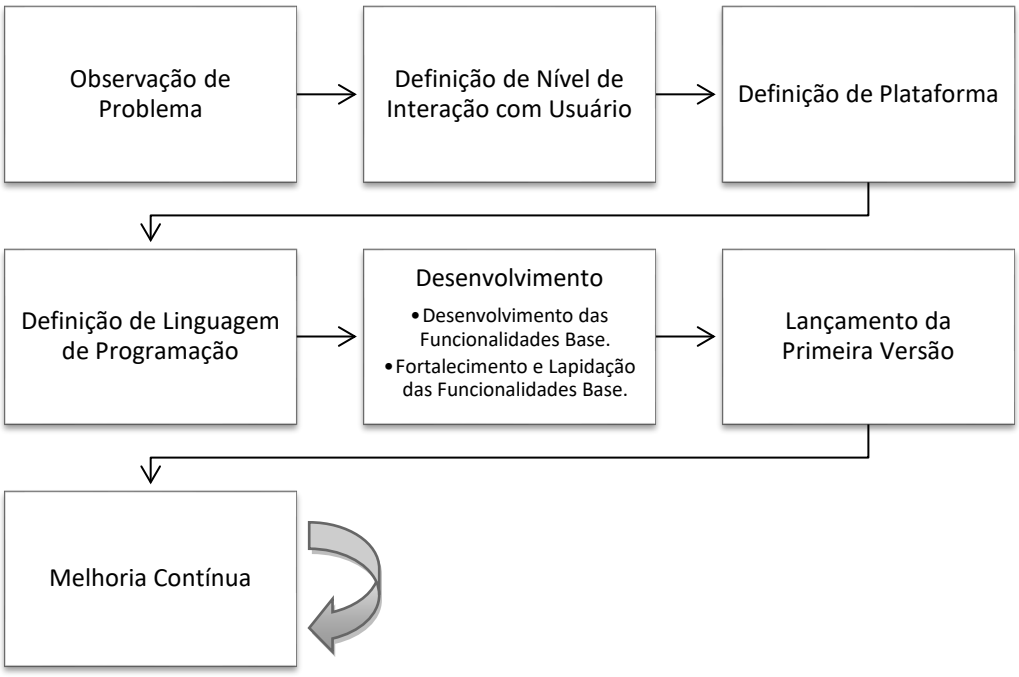
O desenvolvimento do assistente virtual se inicia a com a observação de um problema que poderia ser gerenciado através da aplicação de uma plataforma digital. A partir dessa definição, se dá início a segunda fase do processo de desenvolvimento, que é a definição de nível de interação com o usuário. Diversas opções estavam disponíveis, como: criação de websites, formulários, aplicativos independentes, entre outros. Optou-se pela utilização de um *chatbot*, pois foi levado em consideração, em nível de companhia, que haveria a possibilidade de indivíduos de diversos níveis de conhecimento tecnológico terem contato com a ferramenta. O *chatbot* oferece uma interface mais amigável e possui o nível de interação entre humano e computador mais customizável, podendo assim permitir uma troca de informações menos propensa a erros de utilização pelo usuário e na captura de informações por meio da máquina.

Com as duas primeiras etapas definidas, surge a necessidade de definição da plataforma na qual a ferramenta será baseada. Com a possibilidade de utilização de contas de desenvolvedor do *Google Workspace* e todo sistema de aplicativos da *Google* previamente estabelecido em nível global dentro da companhia, optou-se pela utilização do *Google Workspace* e *Google Apps* como principal plataforma de desenvolvimento e de funcionamento da ferramenta desenvolvida. Visto que a plataforma fora definida, surge a necessidade da definição da linguagem ou linguagens que servirão de base para o desenvolvimento. Como, na etapa anterior a plataforma fornecida pela *Google* foi estabelecida como principal interface, naturalmente a linguagem *Google Apps Script*, ou GAS, foi a determinada como a principal, e única, linguagem para desenvolvimento do assistente. Essa definição é natural devido ao fato de ser uma linguagem que conversa naturalmente e universalmente com os diversos aplicativos disponíveis.

A partir dessa definição da base, começa-se a programação do processo da etapa de desenvolvimento. Ela foi dividida em duas etapas: Criação do assistente e lapidação da base. A primeira fase foca em desenvolver as funcionalidades que solucionariam os problemas observados no primeiro estágio, previamente descrito. Essas funcionalidades seriam a possibilidade de agendamento de treinamentos, consulta de informações relacionadas aos mesmos e algumas ferramentas de administrador. Posteriormente no segundo estágio de desenvolvimento há a realização do amadurecimento do que foi desenvolvido anteriormente. Com o *feedback* das lideranças do setor se fortalece as implementações sem sair do escopo do lançamento inicial, logo o trabalho deve ser focado apenas para melhorar as questões que envolvem o agendamento de treinamento, consultas e funcionalidades de administrador.

Por último está previsto o estágio de melhoria contínua, que deve ser iniciado após o lançamento oficial da ferramenta. Nele, atualizações periódicas são lançadas para: melhorar a ferramenta, otimizar os tempos de execução, enxugar os *scripts* e lançar novas funcionalidades. Na Figura 4 é possível observar um fluxograma que exemplifica esse processo.

Figura 4 - Fluxo de Etapas do Desenvolvimento da Ferramenta



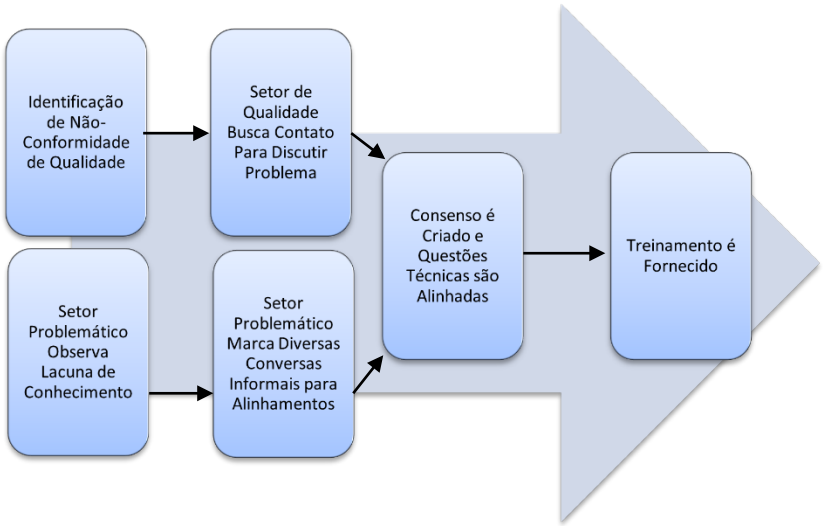
Fonte: O Autor (2023).

3. DESCRIÇÃO DO PROCESSO E SOLUÇÃO

A qualidade corporativa da companhia cujo assistente foi implementado é responsável pelo desenvolvimento, gerenciamento, manutenção e certificação dos processos de qualidade e das ferramentas virtuais e analógicas da qualidade das plantas à nível global. Dentro desse escopo de atuação há a necessidade do fornecimento de treinamentos aos colaboradores para desenvolver uma uniformidade de conhecimento e de capacitação, a fim de diminuir o erro humano em processos de qualidade e melhorar o pensamento crítico.

Atualmente, o processo de treinamento é desenvolvido de maneira informal entre os funcionários, onde a necessidade da aplicação do mesmo é observada após a identificação de uma não conformidade crônica na aplicação das ferramentas de qualidade ou através de inúmeras conversas entre Qualidade Corporativa e os setores que estão apresentando problemas. Os fluxos desse processo podem ser observados na Figura 5.

Figura *5* - Fluxograma do Processo Atual de Desenvolvimento de Treinamentos do Setor de Qualidade Corporativa



Fonte: O Autor (2023).

Os processos apresentados na Figura 5 podem ocorrer de maneira independente ou simultânea. O processo superior, que parte da identificação da não conformidade pelo setor de qualidade, e o inferior, que é gerado pelo setor problemático não possuem o mesmo *Lead Time*. O superior possui o *Lead Time* muito menor, pois a grande maioria das vezes a qualidade sabe como abordar e identificar a necessidade mais rapidamente. Entretanto, ele não é o mais comum, sendo esse, o processo na parte inferior da imagem, que também, é muito mais lento. A lentidão vem do distanciamento digital e físico da área da qualidade e suas áreas clientes. Há desconhecimento, muitas das vezes, dos funcionários que compõem o corpo da qualidade corporativa, quais deles oferecem suporte às ferramentas, quais ferramentas estão sendo suportadas pela qualidade e quais metodologias são implementadas. Desse modo, com o intuito de reduzir o *Lead Time* do processo inferior do fluxo da Figura 5, optou-se pelo desenvolvimento do assistente virtual da qualidade como início de um processo de digitalização de atividades repetitivas e burocráticas.

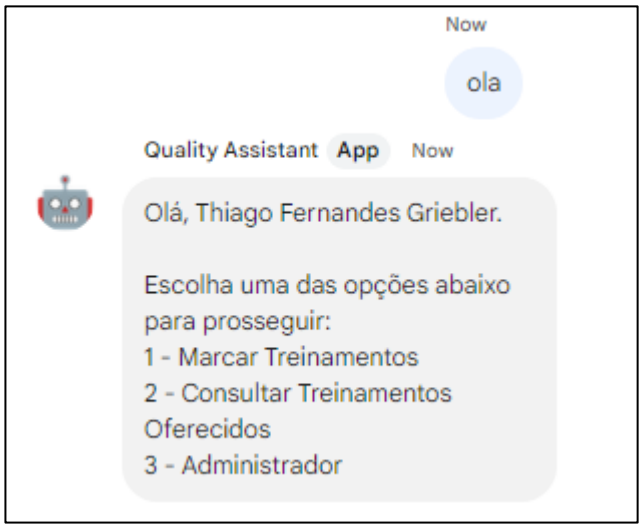
4. FUNCIONALIDADES DO ASSISTENTE VIRTUAL

A seguir são apresentadas as funcionalidades presentes no assistente virtual desenvolvido, como: a possibilidade de marcar treinamento, consultar os treinamentos oferecidos pela qualidade e o poder de adição e remoção de informações do banco de dados através do uso das ferramentas administrativas.

4.1 Marcar Treinamento

Para realizar um agendamento de treinamento, o usuário seleciona a opção *1- Marcar Treinamentos* no menu apresentado pelo assistente virtual, através da escrita do número respectivo da funcionalidade ou o seu título, o último sendo necessário ser uma correspondência exata do *input* do *user*. A Figura 6 representa essa interação.

Figura 6 - Menu Inicial de Interação com o Chatbot



Fonte: O Autor (2023).

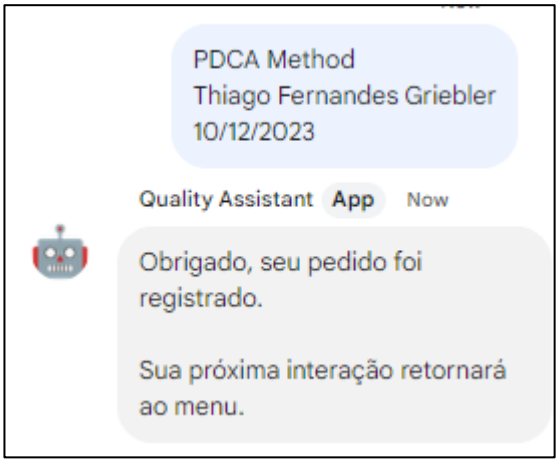
Após começar a comunicação com o chat, o assistente requisitará que o usuário determine o tipo de treinamento, o nome do instrutor e a data na qual ele deseja que o treinamento seja realizado, através da interação na caixa de texto do *Chat*. A Figura 7 exibe a requisição de informações para o registro de um pedido de treinamento realizado pelo usuário.

Figura 7 - Requisição de Informações de Agendamento de Treinamento

Fonte: O Autor (2023).

Com as informações sobre o treinamento, o assistente acessa o banco de dados e registra as informações inseridas, retornando ao usuário, após a finalização dessa tarefa, que o registro foi concluído e que a próxima interação retornará o menu de boas-vindas. A Figura 8 exibe essa notificação de sucesso da operação.

Figura 8 - Notificação Realizada pelo Chatbot Indicando a Finalização do Processo de Registro de Treinamento



Fonte: O Autor (2023).

A segunda etapa do registro ocorre no banco de dados. Os dados que foram lançados pelo usuário são registrados de maneira bruta, na tabela *request_db*.

Na coluna *PEDIDOS DE TREINAMENTOS* é armazenado o valor da variável *resposta* capturada no *chat* pelo assistente. Na coluna *REGISTRO* é inserida a data do dia da interação entre o usuário e o assistente virtual. A última coluna, *USUARIO*, captura o atributo *Name* do usuário que fez a solicitação do treinamento. A Figura 9 exibe a tabela *request_db* e sua estruturação.

Figura 9 - Tabela request_db

PEDIDOS DE TREINAMENTOS	REGISTRO	USUARIO
AQP Thiago Fernandes Griebler 09/04/2024	23/10/23	Thiago Fernandes Griebler
AQP Thiago Fernandes Gri 09/04/1998	23/10/23	Thiago Fernandes Griebler
PDCA Method Thiago Fernandes Gri 10/12/2023	14/11/23	Thiago Fernandes Griebler

Fonte: O Autor (2023).

A tabela *training_overview* captura as informações e realiza um tratamento de dados, alocando as informações em colunas condizentes com cada informação que o usuário forneceu. Na tabela *training_overview*, um *script* realiza uma varredura, identificando os usuários que ainda não foram notificados por e-mail da realização do agendamento do treinamento. A Figura 10 exibe a tabela *training_overview*.

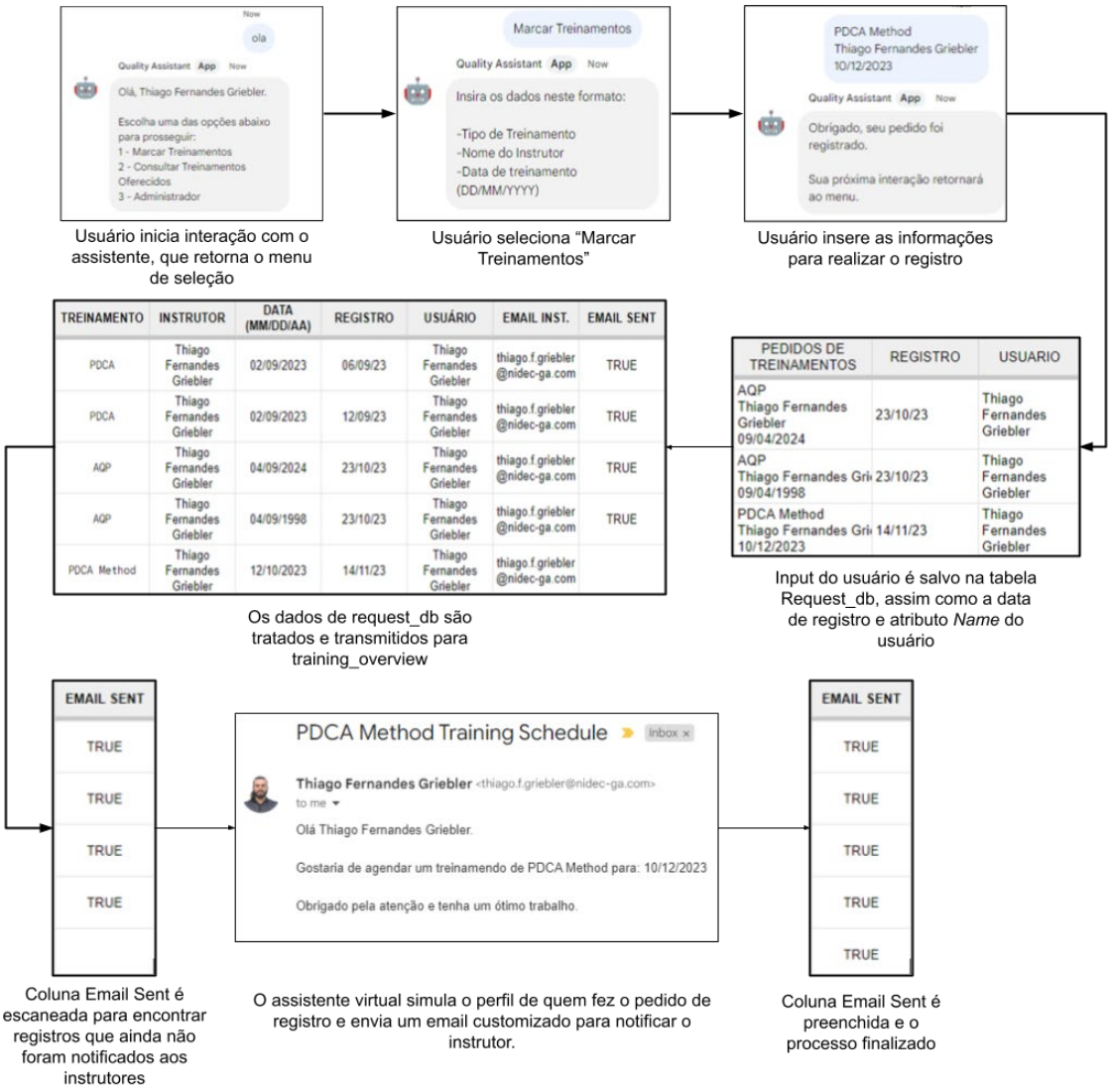
Figura 10 - Tabela training_overview

TREINAMENTO	INSTRUTOR	DATA (MM/DD/AA)	REGISTRO	USUÁRIO	EMAIL INST.	EMAIL SENT
PDCA	Thiago Fernandes Griebler	02/09/2023	06/09/23	Thiago Fernandes Griebler	thiago.f.griebler @nidec-ga.com	TRUE
PDCA	Thiago Fernandes Griebler	02/09/2023	12/09/23	Thiago Fernandes Griebler	thiago.f.griebler @nidec-ga.com	TRUE
AQP	Thiago Fernandes Griebler	04/09/2024	23/10/23	Thiago Fernandes Griebler	thiago.f.griebler @nidec-ga.com	TRUE
AQP	Thiago Fernandes Griebler	04/09/1998	23/10/23	Thiago Fernandes Griebler	thiago.f.griebler @nidec-ga.com	TRUE
PDCA Method	Thiago Fernandes Griebler	12/10/2023	14/11/23	Thiago Fernandes Griebler	thiago.f.griebler @nidec-ga.com	TRUE

Fonte: O Autor (2023).

Quando identificado o usuário, o assistente elabora um e-mail com as informações pertinentes para o instrutor que foi selecionado e o envia para o mesmo. A partir desse momento, é papel do instrutor selecionado entrar em contato com o usuário que agendou o treinamento para alinhamento das expectativas, da confirmação da data e entendimento geral da situação. Na Figura 11, é representado todo fluxo do processo de marcar treinamentos.

Figura 11 - Fluxo Completo do Processo de Marcar Treinamento



Fonte: O Autor (2023).

4.1.1. Criação do Email de Notificação

A função *enviaEmails* é uma das funções auxiliares que rodam a partir do *script* principal que serve de base do assistente virtual. Ela é responsável por analisar as pendências de requisições de treinamentos e também montar os emails que serão enviados aos instrutores de cada ferramenta e metodologias. Ela pode ser observada na Figura 12.

Figura 12 – Função *enviaEmails*

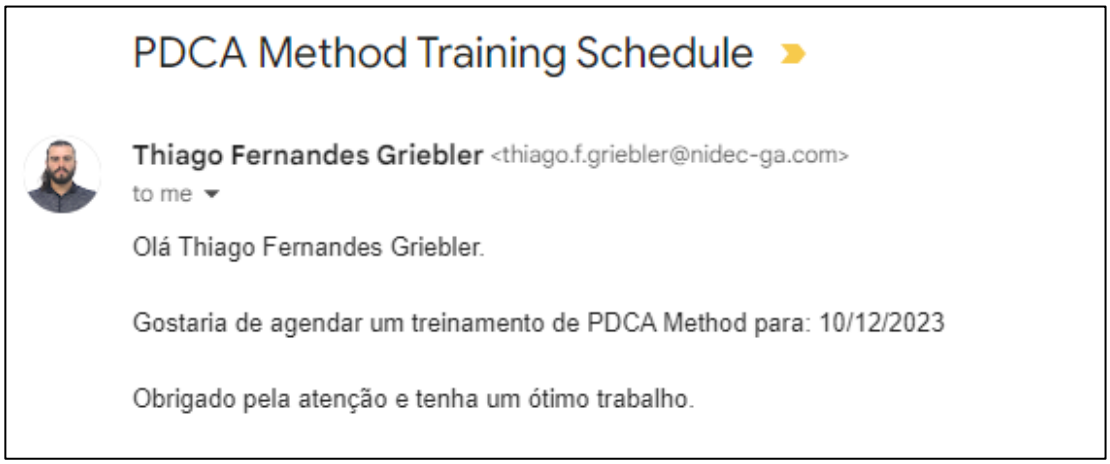
```
114  //Função dos emails
115  function enviaEmails(){
116    const sheet = SpreadsheetApp.openByUrl("https://docs.google.com/
       spreadsheets/d/1ARykXLxSztLjuTWtDjzbOk9LT1uaIlmQfZggKfhhfB8/edit#gid=0");
117    let ss = sheet.getSheetByName("training_overview");
118
119    let lastrow = ss.getRange("J1").getValue();
120    let data = ss.getRange("A2:G"+(lastrow-1)).getValues();
121
122    for(let i = 0;i<data.length;i++){
123      if(data[i][6]==""){
124        MailApp.sendEmail(data[i][5],data[i][0]+" Training Schedule","Olá "
           +data[i][1]+".\n\nVocê tem um pedido para agendar um treinamento de "+
           data[i][0]+" para o dia "+ data[i][2].toDateString()+" realizado por "
           +data[i][4]+".\n\nObrigado pela atenção e tenha um ótimo trabalho.",
           {name:'Thiago Fernandes Griebler'});
125        ss.getRange("G"+(i+2)).setValue("TRUE");
126      }
127    }
128
129    return;
130  }
```

Fonte: O Autor (2023).

As primeiras duas linhas são responsáveis por definir, primeiramente, o documento que será acessado e posteriormente a tabela que será utilizada. Sucedendo isso, há a definição das variáveis *lastrow* e *data*. A primeira é estabelecida para armazenar qual o índice da última linha preenchida da tabela. Isso é importante para evitar laços infinitos ou longos tempos de processamento. Já *data* armazena os dados presentes na tabela em um vetor bidimensional. A criação desse vetor de dados é importante pois reduz os tempos de processamento em comparação com a invocação dos elementos da tabela continuamente.

O laço *for* é responsável pela identificação de quais requisições de treinamento ainda estão pendentes e de preparar o email com as informações e enviá-lo. A Figura 13, exibe um exemplo de email enviado pelo assistente.

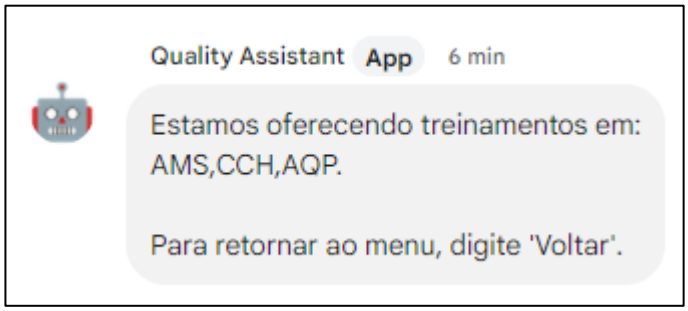
Figura 13 – Exemplo de Email de Notificação Enviado Pelo Assistente Virtual



4.2.Consultar Treinamentos

A consulta dos treinamentos ofertados pode ser realizada utilizando o mesmo método do registro de treinamentos, através do *menu* representado na Figura 6. A partir do comando do usuário o assistente virtual exibirá quais ferramentas e metodologias estão sendo suportadas. Essa interação é exibida na Figura 14.

Figura 14 – Assistente Virtual Exibindo os Treinamentos Oferecidos pela Qualidade



Fonte: O Autor (2023).

Os treinamentos oferecidos estão registrados na tabela *training_db* dentro do banco de dados. Essas informações só podem ser alteradas por usuários que possuam acesso administrativo. A tabela *training_db* é exibida na Figura 15.

Figura 15 – Tabela *training_db*

TREINAMENTO	INSTRUTORES				
AMS	Thiago Fernandes Griebler	Frederico Karklin	Joel Olivério		
CCH	Thiago Fernandes Griebler	Frederico Karklin			
AQP	Thiago Fernandes Griebler	Rodrigo Martins			

Fonte: O Autor (2023).

As informações da tabela são extraídas através da função auxiliar *getTreinos*, que salva as informações da coluna de treinamentos em um vetor e usa o método *toString* para converter tudo em texto, para posteriormente ser chamado pela função principal. A função *getTreinos* está presente na Figura 16.

Figura 16 – Função *getTreinos*

```
//Função para buscar os treinamentos oferecidos pela qualidade corporativa.
(DONE)(TESTED)
function getTreinos(){
  const sheet = SpreadsheetApp.openByUrl("https://docs.google.com/spreadsheets/d/
  1ARykXLxSztLjuTWtDjzbOk9LT1uaIlmQfZggKfhhfB8/edit#gid=0");
  let ss = sheet.getSheetByName("training_db");


  let lastrow = ss.getLastRow();
  let treinos = (ss.getRange("A2:A"+lastrow).getValues()).toString();

  return treinos;
}
```

Fonte: O Autor (2023).

A invocação da função *getTreinos* no código principal do *chatbot* pode ser observada na Figura 17.

Figura 17 – Invocação da Função *getTreinos* no Código Principal

```
case "2":
case "consultar treinamentos oferecidos":
  userProperty.setProperty('estadoUsuario','3');
  return {"text" : "Estamos oferecendo treinamentos em:\n"+
  getTreinos()+".\n\nPara retornar ao menu, digite 'Voltar'."};
```

Fonte: O Autor (2023).

4.3.Ferramentas Administrativas

As ferramentas administrativas são responsáveis pelo gerenciamento do banco de dados do assistente. Elas só podem ser acessadas pelos usuários registrados como administradores. Essa funcionalidade está presente para prevenir o uso indevido da planilha do banco de dados e da realização de alterações feitas diretamente no documento, podendo acarretar em erros nos *scripts* que nela atuam.

O acesso à essas ferramentas é feito através do menu de boas-vindas, conforme as formas previamente citadas, selecionando *3 – Administrador*. O *script* capta o e-mail do usuário que está *logando* e identifica se ele está presente no banco de dados de administrador. Esse banco pode ser observado na Figura 18.

Figura 18 – Banco de Dados de Administradores

	ADMINISTRADORES		
Thiago Fernande thiago.f.griebler(	Thiago Fernandes Griebler	thiago.f.griebler@nidec-ga.com	6
Benjamin Grand benjamin.grandc	Benjamin Grando	benjamin.grando@nidec-ga.com	
Frederico Karklir frederico.t.karlin	Frederico Karklin	frederico.t.karlin@nidec-ga.com	
Polibio de Souza polibio.souza@n	Polibio de Souza	polibio.souza@nidec-ga.com	
Rodrigo Martins rodrigo.martins@	Rodrigo Martins	rodrigo.martins@nidec-ga.com	

Fonte: O Autor (2023).

Sendo verificada a existência do usuário, o acesso às ferramentas torna-se permitido. A Figura 19 exibe o código responsável por essa verificação.

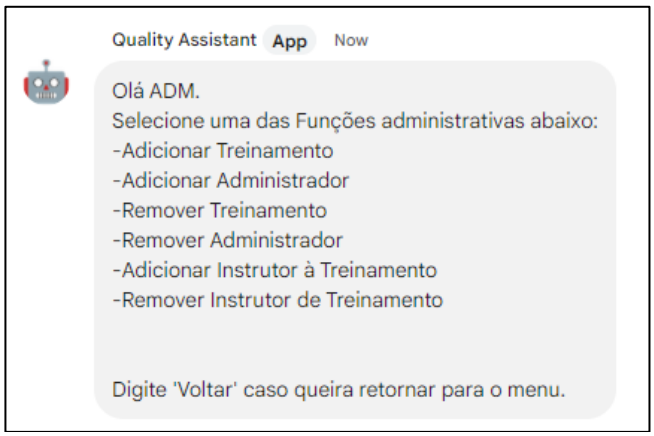
Figura 19 - Função Auxiliar que Realiza a Verificação do Usuário como Administrador

```
//Função para verificar se usuário é administrador. (DONE) (TESTED)
function admVerificar(user){
  const sheet = SpreadsheetApp.openByUrl("https://docs.google.com/spreadsheets/d/
  1ARykXLxSztLjuTWtDjzbOk9LT1uaIlmQfZggKfhhfB8/edit#gid=0");
  let ss = sheet.getSheetByName("adm_db");

  let lastrow = ss.getRange("E2").getValue()+1;
  let adms = ss.getRange("B2:B"+lastrow).getValues();

  for(let i=0;i<adms.length;i++){
    if((adms[0][i]).includes(user)==true){
      return true;
    }
  }

  return false;
}
```

Fonte - O Autor (2023).

As ações que o administrador, atualmente, pode realizar, encontram-se definidas na Figura 20, sendo elas: adicionar treinamento, adicionar administrador, remover treinamento, remover administrador, adicionar instrutor à treinamento e remover instrutor à treinamento.

Figura 20 – Ações Administrativas Disponíveis na Ferramenta.



Fonte - O Autor (2023).

4.3.1. Adicionar Treinamento

A adição de treinamentos é realizada através da utilização da variável *resposta*, que captura a informação inserida pelo usuário no *chat* e a salva no banco de dados, conforme na tabela *training_db*, exibida na Figura 15. O procedimento é realizado pelo *script* que identifica a última linha do banco que está com os campos preenchidos e salva esse índice na variável *lastrow* somando mais um ao mesmo. Posteriormente utiliza-se o método *setValue* para inserir o valor de *resposta* no campo correto da tabela. A função *addTraining*, que é responsável pela registro de novos treinamentos no banco de dados, é representada na Figura 21.

Figura 21 - Função *addTraining*

```
//Função para adicionar treinamento. (DONE) (TESTED)
function addTraining(resposta){
  const sheet = SpreadsheetApp.openByUrl("https://docs.google.com/spreadsheets/d/
  1ARykXLxSztLjuTWtDjzbOk9LT1uaIlmQfZggKfhhfB8/edit#gid=0");
  let ss = sheet.getSheetByName("training_db");

  let lastrow = ss.getLastRow()+1;

  ss.getRange("A"+lastrow).setValue(resposta);

}
```

Fonte - O Autor (2023).

4.3.2. Adicionar Administrador

Para adicionar administradores ao banco de dados é utilizada a função auxiliar *addAdm,* presente na Figura 22.

Figura 22 - Função *addAdm*

```
//Função para adicionar usuário como administrador. (DONE) (TESTED)
function addAdm(resposta){
  const sheet = SpreadsheetApp.openByUrl("https://docs.google.com/spreadsheets/d/
  1ARykXLxSztLjuTWtDjzbOk9LT1uaIlmQfZggKfhhfB8/edit#gid=0");
  let ss = sheet.getSheetByName("adm_db");

  let lastrow = ss.getRange("E2").getValue()+1;

  ss.getRange("A"+lastrow).setValue(resposta);
  return {"text": "Registro realizado com sucesso."};
}
```

Fonte - O Autor (2023).

Similar à função *addTraining*, ela também utiliza a variável *resposta* para armazenar o conteúdo inserido pelo usuário e segue a mesma lógica para gravar a informação no banco de dados. A diferença é que a tabela utilizada para o registro das informações é a *adm_db*, que foi previamente exibida na Figura 18.

4.3.3. Remover Treinamento

A função *trainingRMV* é responsável pela remoção de treinamentos dentro do banco de dados. Ela captura os dados da tabela através de um vetor bidimensional e compara-os com a resposta dada pelo usuário. Quando houver semelhança o *script* limpa todos os campos de informação da linha na qual a informação está presente e realiza uma ordenação por ascendente em ordem alfabética. A função *trainingRMV* pode ser observada na Figura 23.

Figura 23 - Função *trainingRMB*

```
//Função para remover treinamentos. (DONE) (TESTED)
function trainingRMV(resposta){
  const sheet = SpreadsheetApp.openByUrl("https://docs.google.com/spreadsheets/d/
  1ARykXLxSztLjuTWtDjzbOk9LT1uaIlmQfZggKfhhfB8/edit#gid=0");
  let ss = sheet.getSheetByName("training_db");

  let lastrow = ss.getLastRow();
  let data = ss.getRange("A2:A"+lastrow).getValues();

  for(let i = 0; i<data.length;i++){
    if (resposta == data[i][0]){
      ss.getRange(i+2,1).setValue("");
      ss.getRange(i+2,2).setValue("");
      ss.getRange(i+2,3).setValue("");
      ss.getRange(i+2,4).setValue("");
      ss.getRange(i+2,5).setValue("");
      ss.getRange(i+2,6).setValue("");
      ss.getRange('A2:F'+lastrow).activate().sort({column: 1, ascending: true});
      return 1;
    }
  }
  return 0;
}
```

Fonte - O Autor (2023).

4.3.4. Remover Administrador

A função responsável pela remoção de um administrador é a *removeADM*. É uma função simples que compara apenas a resposta do usuário com o conjunto de dados salvo do banco de dados em um vetor bidimensional. A Figura 24 exibe a função *remove ADM.*

Figura 24 - Função *removeADM*

```
//Função para remover ADM. (DONE) (TESTED)
function removeADM (resposta){
  const sheet = SpreadsheetApp.openByUrl("https://docs.google.com/spreadsheets/d/
  1ARykXLxSztLjuTWtDjzbOk9LT1uaIlmQfZggKfhhfB8/edit#gid=0");
  let ss = sheet.getSheetByName("adm_db");

  let indicador = ss.getRange("E2").getValue();
  let data = ss.getRange("B2:B"+indicador).getValues();

  for(let i = 0; i<data.length;i++){
    if(resposta == data[i][0]){
      ss.getRange("A"+(i+2)).setValue("");
      return 1;
    }
  }
  return 0;
}
```

Fonte - O Autor (2023).

O *loop for* compara os dados e caso encontre a sentença que seja igual, esvazia o campo correspondente na tabela e retorna o valor numérico um, para indicar ao usuário que houve a remoção do administrador. Caso não seja encontrada nenhuma sentença que seja igual à fornecida pelo usuário, o *script*

retorna o valor numérico zero e notifica o usuário de que não foi encontrada nenhuma similaridade com a informação por ele fornecida, no banco de dados.

4.3.5. Adicionar Instrutor à Treinamento e Remover Instrutor de Treinamento

As funcionalidades de adicionar e remover instrutores do banco de dados funcionam de maneira similar à função de remoção treinamentos. O *script* para identificar o treinamento no banco de dados é o mesmo. Entretanto há a inserção de um *loop for* extra, que trabalha nas colunas da linha do treinamento que deseja-se realizar a inserção ou remoção de um instrutor. Para a inserção, o nome fornecido pelo usuário é inserido no primeiro campo vazio que for encontrado nas colunas. Já para a remoção, percorre-se todas as colunas em busca da informação fornecida pelo usuário. Caso haja a similirade total entre as informações, o *script* esvazia a informação do campo.

5. CONSIDERAÇÕES FINAIS

Nas fases finais de desenvolvimento, observou-se a capacidade de expansão da ferramenta, tanto para uso dentro do setor como para outros setores, devido à facilidade de trabalhar com o *Google Workspace.* Possíveis melhorias planejadas são: criação e gerenciamento de turmas de treinamento, conexão com a matriz de conhecimento para gerar indicadores e conexão com as planilhas de performance de qualidade para exibição de indicadores de primeiro nível, como custo de sucata e índice de qualidade.

Uma área de atuação externa interessante, seria em questões de relação entre companhia e cliente, principalmente em processos que envolvem reclamações. Desenvolver um processo mais intuitivo através do *chatbot* com conexões à planilhas e formulários do *Google* seria efetivo na redução da insatisfação da interface atual do sistema. Entretanto esse fluxo possui uma complexidade muito maior, pois envolve múltiplos setores, principalmente o de Tecnologia da Informação, devido aos riscos de segurança digital que o lançamento de uma ferramenta à clientes externos apresenta.

Quando apresentado aos gestores, deu-se um bom *feedback* em relação à iniciativa e as bases de conhecimento e técnica que foram desenvolvidas, por haver um grande atraso na digitalização dos processos e também nos conhecimentos em relação a ferramentas de sistemas da informação. Foi sugerida à criação de uma ferramenta nova, chamada de *Quality Skill Matrix* (QSM), que controlaria toda base de conhecimento de qualidade das plantas e utilizaria o assistente virtual como ferramenta para realizar lançamentos e consultas diretamente nela, além de outras automações de processos que à envolveriam. A QSM seria uma união entre *dashboards*, indicadores e controle de rotinas, utilizando a base do que já foi desenvolvido com o assistente virtual da qualidade.

REFERÊNCIAS

CARVALHO JÚNIOR, C. F. DE; CARVALHO, K. R. S. DOS A. DE. Chatbot: uma visão geral sobre aplicações inteligentes. **Revista Sítio Novo**, v. 2, n. 2, p. 68, 20 dez. 2018.

FERREIRA, J. **Google Apps Script**. [s.l.] “O’Reilly Media, Inc.”, 2014.

GANAPATHY, R. **Learning Google Apps Script**. [s.l.] Packt Publishing, 2016.

IYER, B.; ABHI JEEVAGANAMBI. **Google Workspace User Guide**. [s.l.] Packt Publishing Ltd, 2022.

OLIVEIRA, F. **Interação Humano Computador**. 2. ed. Fortaleza - Ceará: EdUECE, 2015.

RED HAT.**What is an API?** Disponível em: <https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces#overview>. Acesso em: 2 dez. 2023.

VERTIGO. **Entenda o que é uma API! Aprenda com quem é especialista no assunto.** Disponível em: <https://vertigo.com.br/entenda-o-que-e-uma-api/#:~:text=API%20(Application%20Program%20Interface)%20%C3%A9>. Acesso em: 2 dez. 2023.