Deco AI Sales Assistant: Utilizando LLMs para experiências de e-commerce personalizadas e sob demanda

Itamar de Paiva Rocha Filho

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

João Pessoa, PB
2024

Itamar de Paiva Rocha Filho

Deco AI Sales Assistant: Utilizando LLMS para experiências de e-commerce personalizadas e sob demanda

Relatório Técnico apresentado ao curso Engenharia da Computação
do Centro de Informática, da Universidade Federal da Paraíba,
como requisito para a obtenção do grau de Bacharel em Engenharia da Computação

Orientador: Yuri de Almeida Malheiros Barbosa

Abril de 2024

Catalogação na publicação
Seção de Catalogação e Classificação

R672d Rocha Filho, Itamar de Paiva.
Deco AI sales assistant: utilizando LLMs para experiências de e-commerce personalizadas e sob demanda / Itamar de Paiva Rocha Filho. - João Pessoa, 2024.
63 f. : il.

Orientação: Yuri Malheiros Barbosa.
TCC (Graduação) - UFPB/CI.

1. Assistente de compras. 2. Comércio eletrônico. 3. Aplicação web. 4. Engenharia de software. 5. No-code. I. Barbosa, Yuri Malheiros. II. Título.

UFPB/CI CDU 004.777

Elaborado por Michelle de Kássia Fonseca Barbosa - CRB-738

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

Trabalho de Conclusão de Curso de Engenharia da Computação intitulado **Deco Sales AI Assistant: Utilizando LLMS para experiências de e-commerce personalizadas e sob demanda** de autoria de Itamar de Paiva Rocha Filho, aprovado pela banca examinadora constituída pelos seguintes professores:

Prof. Dr. Yuri de Almeida Malheiros Barbosa
Universidade Federal da Paraíba

Prof. Dr. Thaís Gaudencio do Rêgo
Universidade Federal da Paraíba

Prof. Dr. Matheus Gaudencio do Rêgo
deco.cx

RESUMO

A Inteligência Artificial (IA) tem se tornado cada vez mais acessível e suas aplicações podem ser vistas em praticamente todos os setores da sociedade. Com o desenvolvimento dos grandes modelos de linguagem, a automatização de tarefas conversacionais se tornou possível com um desempenho bem mais próximo do nível humano. O objetivo deste trabalho é criar uma experiência de compras única e personalizada para os clientes. Por meio de recomendações precisas e sob medida, a utilização do AI Sales Assistant atua aumentando a conversão das lojas de forma significativa. Nesse relatório, apresentaremos as decisões técnicas e as funcionalidades do assistente. Esse projeto se utiliza de LLMs (*Large Language Models*) e tem como características multilinguagem, compreensão de texto, imagem e áudio e acesso a todo o catálogo de produtos disponíveis nas lojas implementadas.

Palavras-chave: Assistente de compras, Comércio eletrônico, Aplicação Web, Engenharia de Software, No-code.

ABSTRACT

Artificial Intelligence (AI) has become increasingly accessible, and its applications can be seen in virtually every sector of society. With the development of large language models, the automation of conversational tasks has become possible with performance much closer to the human level. This work aims to create a unique and personalized shopping experience for customers. Through precise and tailored recommendations, the AI Sales Assistant significantly increases store conversion rates. In this report, we will present the technical decisions and functionalities of the assistant. This project utilizes LLMs (Large Language Models) and features multilingual capabilities, text, image, audio comprehension, and access to the entire product catalog available in the implemented stores.

Keywords: Shopping assistant, E-commerce, Web application, Software engineering, No-code.

AGRADECIMENTOS

Queria começar esse trabalho agradecendo à minha família. A minha mãe, Maria Margareth Rocha, que me deu todo suporte e sempre esteve comigo durante a minha graduação, além de ser um grande exemplo de determinação, coragem e altruísmo. Obrigado por tanta dedicação desde meu nascimento, por todo o amor e todos os conselhos e lições.

À minha irmã, Maria Helena Rocha Pedrosa, que foi responsável por me ensinar tantas coisas nessa vida e foi uma conselheira e amiga nos momentos de paz e de caos, sempre trazendo novas ideias, me dando paz e me ajudando a enxergar novos horizontes.

Ao meu Pai, Itamar Rocha, que me apoiou nas decisões e me deu o suporte necessário para que eu conseguisse ter a liberdade de me aventurar e ter diversas experiências profissionais durante o curso.

À Maricélia Quirino, minha segunda mãe, que foi fonte de carinho, tranquilidade e que me ensinou a arte da paciência desde minha infância.

À minha avó Laurita Martins, que sempre foi um exemplo de curiosidade e de bom humor e ao meu avô Izidório Nogueira, que faleceu ano passado, e que foi um homem íntegro e um grande amante da música e da poesia.

Ao meu cunhado, Daniel Pedrosa, a quem eu tenho como um irmão. A minha Tia Marlede Rocha, que sempre esteve presente. A minha prima Maria Isabel Rocha, a quem também tenho como uma irmã e que faz muita falta aqui.

À Giulia Carvalhal, que possui um lugar no meu coração, sempre acreditou em mim (muito mais do que eu mesmo) e que me ensinou muito e foi uma grande fonte de inspiração.

Ao meu grande amigo Davi Leone, que além de ter feito um papel de irmão, também foi uma das primeiras pessoas a despertar em mim a faísca de ter um impacto global e ampliar meus horizontes geograficamente através dos *summer jobs*.

Aos meus amigos João Pedro Vasconcelos e João Wallace. Entrar neste mundo de computação foi um verdadeiro desafio para mim, e sem a ajuda de vocês, teria sido impossível. Vocês são mais que amigos, são irmãos, e sou extremamente grato por ter tido a sorte de conhecê-los. Com engenheiros e pessoas como vocês, não tenho dúvida que, como Marcelo Yuri afirma: "Dessa sala vai sair pelo menos um unicórnio" e o nosso será em poucos anos!

Ter nascido nessa família, o que eu não tive nenhuma influência, foi a minha maior conquista. E aos que não são da família de sangue, mas de coração, vocês foram uma escolha mais que certeira. Sou extremamente grato pela vida que eu tive e pelo convívio e amizade de todos vocês.

Aos meus amigos Lara Pontes e Felipe Honorato, que mesmo entrando um ano depois de mim na graduação, foram das pessoas que eu mais convivi e que me proporcionaram momentos ímpares, além de uma amizade que sei que levarei comigo para vida.

Aos meus amigos Matheus Melo, Guilherme Jácome, Humberto Navarro agradeço a companhia, as conversas e as parcerias nas matérias e na TAIL. Aos queridos Tales Nobre e Nicholas Rodrigues agradeço pelo processo de revisão do TCC e pelo apoio dado nos últimos meses.

A professora Thaís Gaudencio por todo suporte dado desde o primeiro período da graduação, por me ensinar que computação, literatura e arte não são coisas excludentes e que ter uma vida equilibrada é algo que nunca devemos abrir mão (só raramente). Obrigado pela paciência, pela dedicação ao aluno, não só como aluno, mas como pessoa e por ser tão humana.

Ao professor Yuri Malheiros pela paciência, pela orientação detalhada e pelas oportunidades concedidas. Obrigado pela dedicação incessante, pelo bom humor constante e por sempre estar disponível para ajudar.

Ao professor Telmo Filho, juntamente com os professores Thaís e Yuri por terem acreditado em mim e no projeto da TAIL e por terem nos apoiado desde sempre na criação dessa liga acadêmica quatro anos atrás. Sem vocês e as suas orientações e apoio esse projeto não teria saído do papel. E obrigado por terem feito parte de forma tão constante dessa minha experiência ímpar na UFPB.

Tive muitos professores excelentes durante minha graduação e muitos que viraram amigos e a quem sou extremamente grato. Gostaria de deixar um agradecimento especial à Anand Subramanian, Tiago Maritan e Lincoln Silva, que além de professores excelentes foram essenciais nessa jornada. Também gostaria de registrar meus agradecimentos aos professores Hugo Cavalcante, Mardson Amorim, Bruno Bruck, Teobaldo Bulhões, Alisson Brito, Marcelo Iury, Raoni Kulesza, Yuska Aguiar e Ewerton Salvador. Obrigado por terem feito essa experiência tão boa e por sua dedicação ao Centro de Informática, vocês fazem nosso corpo docente ser tão bom como é e me fizeram enxergar computação por muito bons olhos.

Durante a graduação, tive a oportunidade de trabalhar em diversas empresas e participar de projetos extracurriculares. Deixo meu agradecimento ao Google, na pessoas dos meus gerentes (Felipe Schargorosdky e Mengzhen Pan), à Meta (Jan Zikes e Kunal Chitkara), à Praso (Fernando Bilfinger), à Atoptima (Guillaume Marques e Vitor Nesello). Nessas experiências pude conhecer a indústria, conhecer engenheiros incríveis, diferentes culturas e países. Além disso, agradeço imensamenta à Fundação Estudar, por ter sido selecionado como

um dos bolsistas no ano de 2021, o que me deu acesso à oportunidades e à pessoas incríveis, o que, sem dúvidas, alavancou minha carreira de forma exponencial, além de ter aberto meus horizontes e mudado minha perspectiva em relação aos meus planos futuros. Agradeço também aos amigos que fiz através da Fundação e que foram fonte de acolhimento e inspiração durante os últimos anos.

Por fim, em Novembro de 2023, comecei meu primeiro emprego como engenheiro em tempo integral na deco.cx. Desde então, tive oportunidade de trabalhar com pessoas incríveis com bagagens extraordinárias. Pude aprender bastante desde que entrei e sou muito feliz com a oportunidade. Deixo meu agradecimento a todo o time da deco, mas sobretudo a Guilherme, Luciano, Candeia, Matheus e Caroline, que me acompanharam de perto desde minha entrada na empresa e tiveram um papel fundamental na concepção do projeto do AI Sales Assistant.

Apesar das probabilidades independentes, acredito fortemente que as pessoas tem um limite de sorte durante a vida e tenho certeza que nunca vou ganhar na loteria, por ter tido o privilégio de ter crescido e convivido com todos vocês nos últimos anos. Deixo aqui também meus agradecimentos aos inúmeros colegas e amigos que fiz nos últimos anos dentro e fora do curso, além de meus outros familiares. Fui ensinado tantas coisas que não conseguiria sequer enumerá-las. E por isso, dedico meu trabalho de conclusão final do curso a vocês. Obrigado por terem feito minha vida tão feliz e tão única dentro e fora da UFPB.

"First, notice that a decision in-itself accomplishes nothing. The decision is a milestone that unblocks and lights up the path for execution. Excellent execution, in fact, can even turn imperfect technical decisions into successful products."

Luiz André Barroso

LISTA DE FIGURAS

LISTA DE CÓDIGOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

LLM - *Large Language Model* (Grande Modelo de Linguagem)

AI - *Artificial Intelligence* (Inteligência Artificial)

API - *Application Programming Interface* (Interfaces de Programação de Aplicações)

SEO - *Search Engine Optimization* (Otimizador de Motores de Busca)

CSS - *Cascading Style Sheets*

HTML - *Hypertext Markup Language*

AGI - *Artificial General Intelligence*

GA - *Google Analytics*

NRF - *National Retail Federation*

CMS - *Content Management System* (Sistema de Gerenciamento de Conteúdo)

AWS - *Amazon Web Services*

RAG - *Retrieval Augmented Generation* (Recuperação de Geração Aumentada)

TAIL - *Technology and Artificial Intelligence League* (Liga de Tecnologia e Inteligência Artificial)

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos, a evolução da Inteligência Artificial (IA) tem revolucionado diversos setores, desde carros autônomos virando parte da realidade, por empresas como Waymo [11] e Cruise [2], até soluções mais simples, como a geração de texto baseado em *prompts* por modelos de linguagem, como o ChatGPT [6], LLama [5], Claude [1], entre outros. Particularmente, o avanço da pesquisa em modelos de linguagem, que tem como um dos marcos o artigo "Attention Is All You Need" [10], trouxe um novo patamar de possibilidades para criação de produtos e inovação. Com o lançamento do ChatGPT, em Novembro de 2022, o mundo teve acesso ao que antes era tido como confidencial/fora da realidade/exclusivo de quem trabalhava na área e com isso, pudemos ver o potencial dessas ferramentas aplicadas às mais diversas esferas da sociedade, o que levou a OpenAI a ter **57 milhões de usuários ativos** no primeiro mês de lançamento de sua plataforma de *chat*, um novo recorde mundial em termos de aquisição de usuários por uma empresa[1].

Atualmente, empresas como Microsoft, Slack, Duolingo, Grammarly e *startups* brasileiras como Teachy[2] e Sindria[3] já estão utilizando grandes modelos de linguagem (do inglês, *Large Language Models*, LLMs) em seus produtos, em áreas como busca, educação e comunicação. Já no *e-commerce*, empresas como Amazon e Shopify já vem integrando sistemas de aprendizagem de máquina em seus setores e explorando novas possibilidades há anos. Esses avanços estão tornando cada vez mais possível uma das maiores demandas do setor de comércio eletrônico (*e-commerce*), um atendimento personalizado, dedicado e instantâneo, algo que antes só uma pessoa poderia fazer. A utilização de técnicas de comércio conversacionais é responsável por uma quebra de paradigma no setor. Contudo, o processo de criação de um assistente de vendas para um negócio é um trabalho desafiador, que ainda exige conhecimento específico sobre os atuais modelos de LLM, suas interfaces de programação de aplicações (do inglês, *Application Programming Interface*, APIs), como manipular os dados e engenharia de *prompt*, o que pode ser inacessível para a maioria das empresas que não atuam no ramo de tecnologia.

Nesse contexto de inovação e democratização, o projeto **Deco AI Sales Assistant** emerge como uma resposta direta às demandas por soluções mais eficazes e personalizadas no *e-commerce*. Nesse relatório abordaremos o desenvolvimento da primeira versão do

[1] Curry, David. "ChatGPT Revenue and Usage Statistics (2023).". Disponível em: https://www.businessofapps.com/data/chatgpt-statistics/. Acesso em: 7 Abr. 2024.
[2] Teachy Educação. Disponível em: https://www.teachy.com.br/. Acesso em: 23 Abr. 2024.
[3] Sindria. Diseponível em: https://www.sindria.ai/. Acesso em: 23 Abr. 2024.

produto, que tem como objetivo proporcionar aos clientes do *e-commerce* uma experiência de compras personalizada, por meio de recomendações precisas e sob medida. Com isso, espera-se aumentar a conversão das lojas de forma considerável. Essa implementação representa um passo significativo em direção à adoção de soluções de IA no varejo. Com a promessa de uma integração simples e eficaz, com poucos cliques, este projeto não apenas reflete o potencial dos modelos de linguagem, mas o impacto que a pesquisa aplicada, de forma acessível, pode gerar na experiência do usuário e na conversão das lojas.

Este trabalho está dividido em cinco capítulos. No primeiro, há a introdução do tema, junto com a motivação por trás da solução. No segundo, temos conceitos gerais da aplicação, tecnologias utilizadas e a identificação dos envolvidos (clientes, usuários e colaboradores). No terceiro capítulo, será apresentada a narrativa do processo de desenvolvimento e a apresentação da proposta, com os respectivos requisitos, casos de uso e arquitetura do sistema. No quarto capítulo, passaremos pelo plano de testes e métricas. Por fim, no Capítulo 5, são abordadas as conclusões obtidas e listados os possíveis trabalhos futuros a serem realizados.

2 CONCEITOS GERAIS

Neste capítulo, serão apresentadas as tecnologias utilizadas, os principais conceitos de tecnologia por trás da aplicação, e será feita a identificação dos envolvidos (clientes, usuários e colaboradores).

2.1 Tecnologias Utilizadas

Nesta subseção, estão descritas as tecnologias utilizadas, com um breve resumo de sua utilização no projeto.

2.1.1 Deco.cx

De acordo com o site especializado em mercados, investimentos e negócios do Brasil, InfoMoney: "A Deco.cx é uma plataforma de código aberto para Deno, JSX e Tailwind. Ela permite criar sites de *e-commerce*, em uma infraestrutura global de ponta, com altos níveis de customização e eficiência, mas de forma relativamente simples, incorporando ferramentas de arrastar e soltar (*drag-and-drop*) e linguagem natural"[4]. O projeto do Deco Sales Assistant foi desenvolvido como uma nova funcionalidade para os clientes da deco e, portanto, faz uso de todo o sistema Deco. Nas próximas seções, estão descritos alguns dos principais conceitos por dentro da plataforma.

2.1.1.1 Deco *admin*

O *admin* é uma ferramenta que possibilita a edição sem código (*no-cod*e), que junta todos os dados e configurações do site em um único lugar. Nele, é possível ter uma visualização completa das diferentes áreas relacionadas ao site desenvolvido com a Deco, como: pré-visualização, otimizadores de motores de busca (do inglês, *Search Engine Optimization*, SEO) / *Metatags*, acesso ao repositório do GitHub com o código fonte, domínios disponíveis, análise de dados (*analytics*), entre outros. Na Figura 01 pode ser observado a tela inicial do *admin*. Nela, tem-se uma pré-visualização do site desenvolvido, os dados referentes a

[4] Info Money. **Plataforma de criação de site de e-commerce Deco.cx atrai US$ 2,2 milhões em investimentos**. 2024. Disponível em: https://www.infomoney.com.br/consumo/plataforma-de-criacao-de-site-de-e-commerce-deco-cx-atrai-us-22-milhoes-em-investimentos/. Acesso em: 7 Abr. 2024.

velocidade do site, um botão direcionando para a parte de *SEO*, o *link* para o repositório no GitHub, os domínios associados, entre outras opções de menu e configuração.

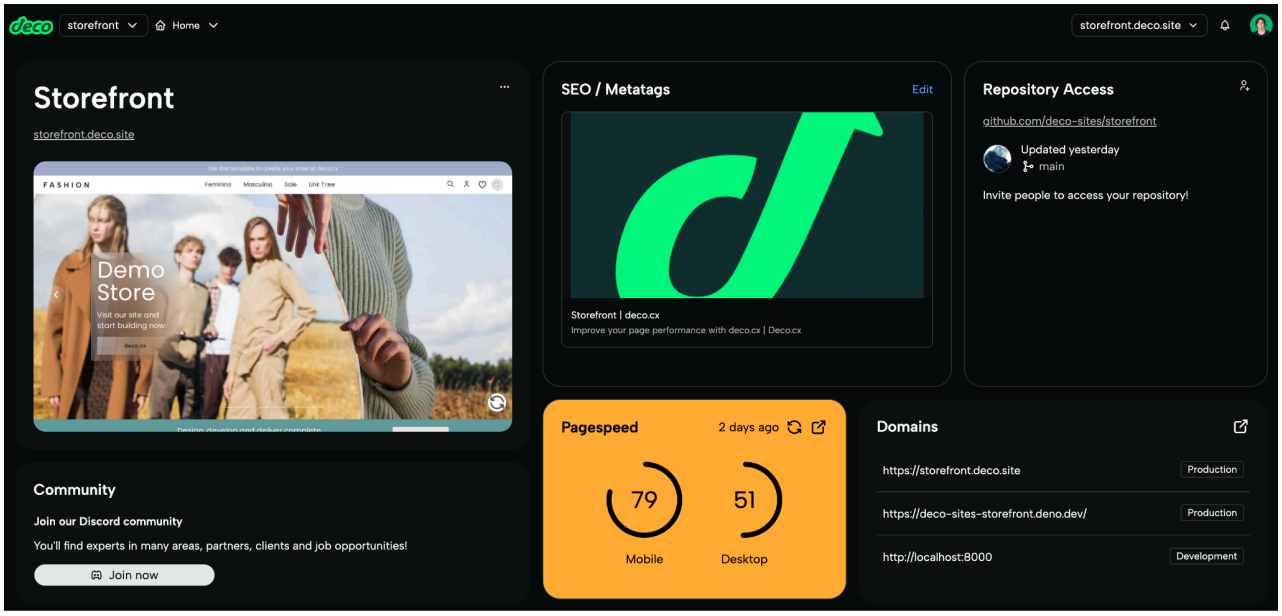

Figura 1: Página inicial admin - Storefront (página template)

2.1.2 Typescript

Dentre as tecnologias utilizadas, a principal linguagem de programação do projeto é o TypeScript. TypeScript [8] é uma linguagem de programação código aberto desenvolvida pela Microsoft. Ele é fortemente tipado e construído a partir do JavaScript. O TypeScript é um superconjunto do JavaScript, o que significa que os programas desenvolvidos em JavaScript também tem suporte no TypeScript. Além disso, o TypeScript permite que sejam definidos os tipos de variáveis, parâmetros de função e retornos, o que torna mais fácil a detecção de erros, melhora a leitura do código e potencializa o uso de ferramentas de autocompletar (*autocomplete*) e refatoração. Por fim, é importante citar que o TypeScript é transpilado para JavaScript, o que permite que ele seja executado em qualquer navegador ou ambiente que suporte JavaScript.

A linguagem TypeScript é utilizada no projeto para o desenvolvimento da aplicação do lado do cliente e do lado do servidor.

2.1.3 Tailwind

Com relação à estilização, Tailwind é o framework escolhido como padrão para os projetos da deco. Tailwind CSS [7] é um *framework* de CSS (*Cascading Style Sheets*), que permite construir interfaces de usuários customizadas sem sair do HTML (*Hypertext Markup Language*). Tailwind fornece classes de baixo nível que facilitam a criação de designs próprios, personalizados e responsivos diretamente no HTML.

2.1.4 Deno

Deno [3] é um *runtime* de JavaScript e TypeScript. Um *runtime* é um ambiente de execução que permite que os programas ou *scripts* sejam executados em um computador. Ele atua como intermediário entre o sistema operacional e o programa executado. Deno tem algumas diferenças positivas em relação ao Node.js (outro *runtime* bastante popular). Entre elas, Deno suporta TypeScript nativamente, sem necessidade de transpilação, como é feito no Node, além de ser seguro por padrão, como limitações ao acesso aos arquivos do sistema, à rede e ao ambiente de execução.

2.1.5 Fresh

Fresh [4] foi escolhido como *framework* web para o Deno. Esse *framework* enfatiza a renderização no servidor, onde a maior parte do site é processada, delegando ao cliente a tarefa de re-renderizar apenas pequenas "ilhas" de interatividade. Esta estratégia reduz a necessidade de JavaScript no lado do cliente, melhorando a velocidade e a eficiência. Fresh oferece suporte completo a TypeScript e foca na eficiência e na simplicidade para aplicações web dinâmicas e reativas.

2.1.6 OpenAI API

A OpenAI é uma empresa de pesquisa e desenvolvimento na área de IA. Segundo seu próprio site institucional[5], sua missão é ter certeza que a inteligência geral artificial (do inglês, *Artificial General Intelligence*, AGI) beneficie toda a humanidade.

[5] OpenAI. "*About*". Disponível em: openai.com/about. Acesso em: 7 Abr. 2024.

O produto mais utilizado da OpenAI, atualmente, é sua API, dentro dela, temos dezenas de modelos variados: LLMs, *Embeddings*[6], *Transcribers*[7], *Image Descriptors*[8], etc. Para a criação do Sales Assistant, foram utilizados três modelos diferentes e quatro APIs distintas, descritas a seguir.

2.1.6.1 Assistants API

Segundo a própria OpenAI, a Assistants API foi criada para facilitar a criação de assistentes baseados em IA capazes de executar uma variedade de tarefas. Essa API ainda está em Beta[9], o que significa que está com suas funcionalidades completas, mas que pode conter problemas (*bugs*) não percebidos, comportamentos inesperados e falhas no serviço.

A API de assistentes funciona de forma similar ao *chat* padrão da OpenAI com a diferença de que:

- Podem ser utilizados diferentes modelos.
- Uma chamada da Assistants API pode configurar instruções novas e específicas para o assistente.
- Os assistentes podem chamar outras ferramentas, como interpretador de código, recuperador de conteúdo ou funções definidas pelo próprio desenvolvedor.
- Podem ser definidas funções novas que podem ser chamadas através do Function Calling, o que potencializa a adaptação da ferramenta para variados contextos.
- O assistente pode acessar arquivos em diversos formatos: Imagens, PDFs, Planilhas, etc.

A Figura 02 descreve a arquitetura de funcionamento dessa API. De forma simplificada, a API funciona primeiro com a criação ou instanciação de um assistente a partir de um ID de um assistente pré-existente. Essa criação ou instanciação pode ser acompanhada de novas instruções. Depois disso, é criada uma *thread*, onde ocorre a troca de mensagens entre o usuário e o assistente. A *thread* equivale a uma sessão de interação entre o usuário e o assistente. Nela, as mensagens serão enviadas e haverá um histórico da conversa que pode ser utilizada para criar um contexto a partir do qual o modelo gerará suas respostas. Após a

[6] OpenAI. "*Embeddings*". Disponível em: platform.openai.com/docs/guides/embeddings. Acesso em: 23 Abr. 2024.
[7] OpenAI. "*Speech to Text*". Disponível em: platform.openai.com/docs/guides/speech-to-text. Acesso em: 23 Abr. 2024.
[8] OpenAI. "*Vision*". Disponível em: platform.openai.com/docs/guides/vision. Acesso em: 23 Abr. 2024.
[9] OpenAI. "How Assistants Work". Disponível em: https://platform.openai.com/docs/assistants/how-it-works/agents. Acesso em: 23 Abr. 2024.

criação da *thread*, as mensagens podem ser enviadas. Para fazer o envio é necessário criar uma entidade Run, que representa uma execução assíncrona e que pode utilizar outras funcionalidades do assistente como interpretação de código, chamada de funções e criação de mensagens.

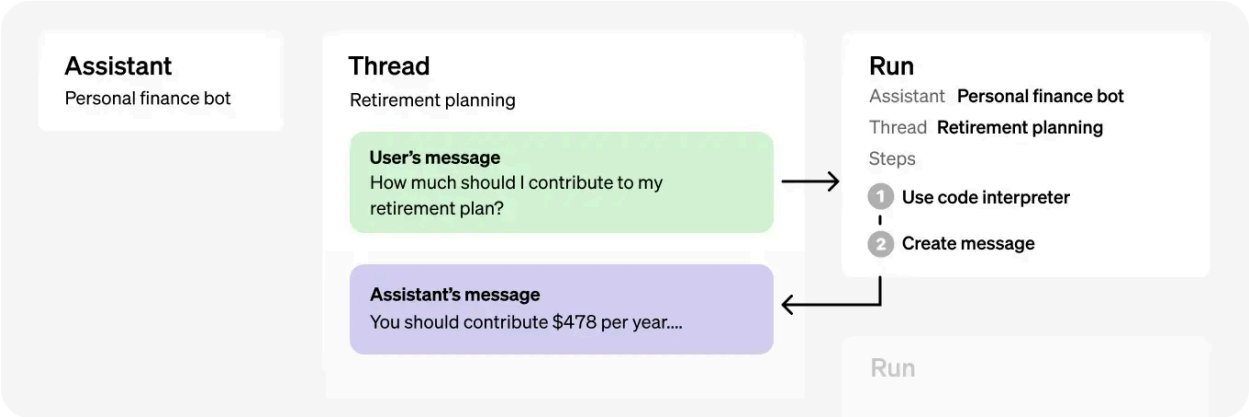

Figura 02: Diagrama de funcionamento - OpenAI Assistants API

O resultado final de uma chamada na API de assistentes da OpenAI pode contar diferentes status como descrito na Figura 03. Quando uma mensagem é enviada, ela automaticamente vai entrar no estado queued, onde ele quase de imediato passará para o estado em progresso (in_progress). Nesse estado, o assistente poderá passar a execução (run) para o estado necessita ação (requires_action), onde haverá atuação da Chamada de Funções (Function Calling) para escolher os parâmetros e a função a se utilizar ou para o estado de cancelamento (cancelling), completude (completed) ou falha (failed).

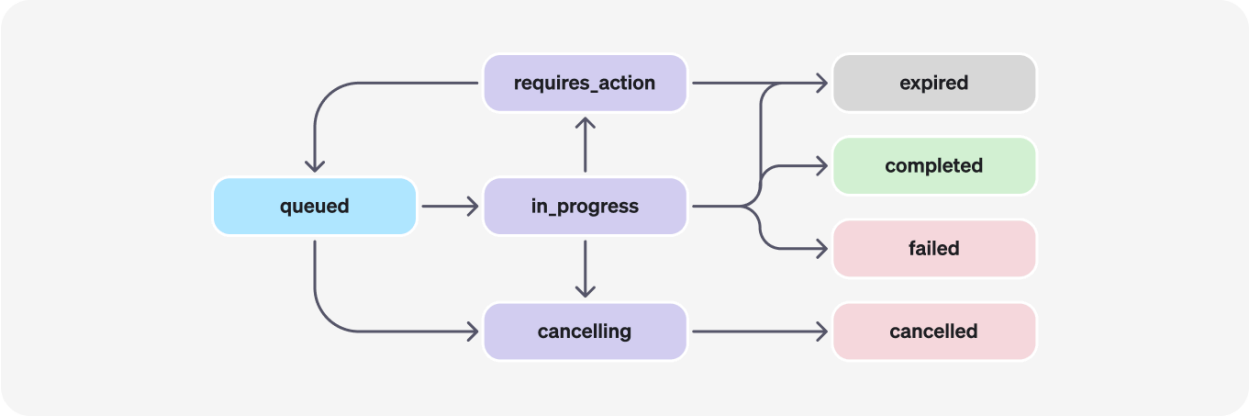

Figura 03: Diagrama contendo os possíveis status retornados

2.1.6.2 GPT4[10]

Como mencionado na seção de Assistants API, podem ser utilizados diferentes modelos[11] com a API. Inicialmente, no projeto do Deco AI Sales Assistant, optou-se por utilizar como modelo padrão o *gpt-4-1106-preview*, que é uma versão do GPT4 com melhorias no mecanismo de seguir instruções. Esse modelo ainda está em *preview*, o que significa que seu uso não é recomendado para produção. O modelo recebe até 128.000 *tokens* na sua janela de contexto e pode produzir até 4096 *tokens* como saída.

2.1.6.3 GPT4 - Vision[12]

Além do modelo GPT4 para texto, também é importante introduzir o modelo GPT4 de visão. O *gpt-4-1106-vision-preview* é um modelo GPT-4 com a habilidade de compreender imagens e responder perguntas a partir delas. Sua compreensão varia desde texto até contexto de elementos visuais dispostos na imagem.

2.1.6.4 Whisper

Além de texto e imagem, a OpenAI também possui modelos de áudio. O Whisper[13] é um modelo de reconhecimento de fala (*speech recognition*), que pode ser utilizado para tarefas como transcrição de voz, tradução e identificação de língua. Ele é um modelo de código aberto,

[10] OpenAI. "GPT4". Disponível em: https://openai.com/research/gpt-4. Acesso em: 23 Abr. 2024.
[11] OpenAI. "Models". Disponível em: https://platform.openai.com/docs/models. Acesso em: 23 Abr. 2024.
[12] OpenAI. "Vision". Disponível em: https://platform.openai.com/docs/guides/vision. Acesso em 25 Abr. 2024.
[13] OpenAI. "Whisper". Disponível em: https://github.com/openai/whisper. Acesso em 25 Abr. 2024.

que é oferecido pela OpenAI também por meio da API, com a vantagem de ser mais rápido pelo ambiente de inferência otimizado que eles oferecem.

2.1.7 HyperDx

No projeto, foi utilizado o HyperDx como ferramenta para *tracing* e *logging*. O HyperDx é uma ferramenta de *tracing* para aplicações que permite o uso de *logs*, *traces* e métricas[14]. Ele pode ser utilizado para verificar o comportamento de uma funcionalidade em produção por meio da disponibilização de erros, gráficos com as métricas e *traces* que permitem identificar, de forma mais clara, os problemas que podem vir a ocorrer ou se a aplicação está funcionando como esperado. Na Figura 04, é visto uma captura de tela com alguns dos possíveis *logs* gerados pelo Deco AI Sales Assistant, além de um gráfico de barras que mostra o número de requisições disposto em uma linha do tempo.

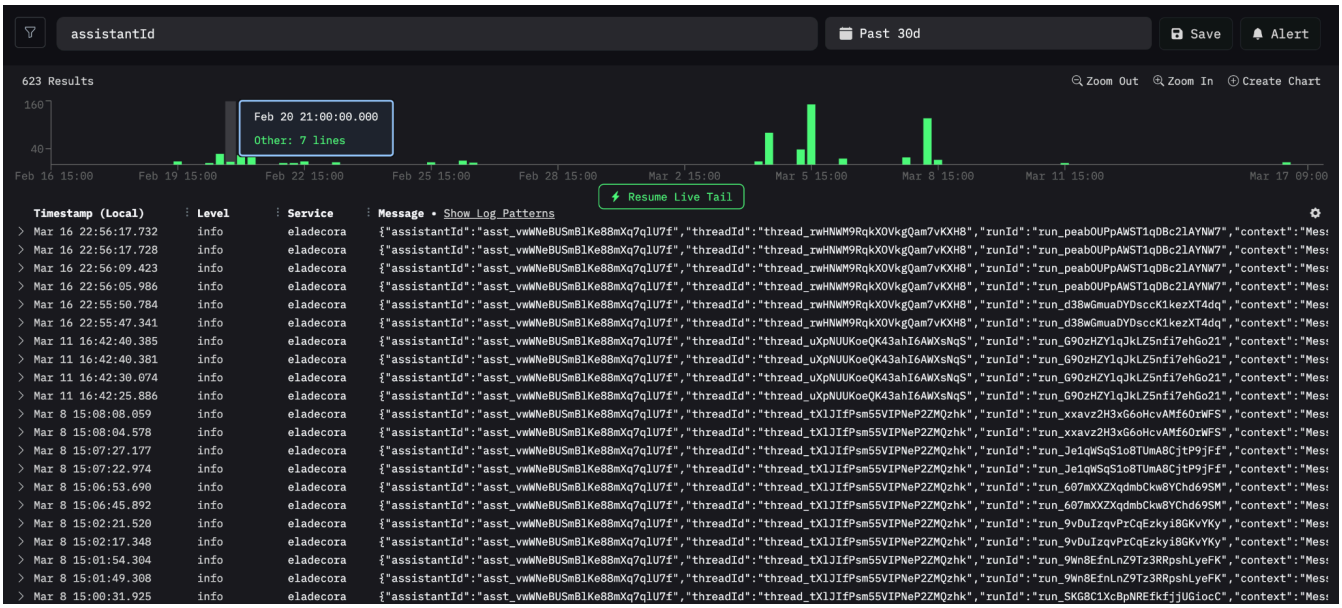

Figura 04: Exemplo de visualização de logs no HyperDX

[14] HyperDx Docs. Disponível em: https://www.hyperdx.io/docs. Acesso em: 24 Abr. 2024.

2.1.8 Plausible e Google Analytics

Por fim, com relação ao envio de eventos do assistente, o Plausible[15] e Google Analytics (GA)[16] são as duas ferramentas utilizadas. O Plausible é uma alternativa código aberto ao GA e está diretamente incorporado no *admin* da deco, o que facilita a visualização dos eventos.

Alguns exemplos de eventos para *e-commerce* são a visualização de itens, a adição ao carrinho e a visualização de páginas.

2.2 Envolvidos

Esta seção descreve as partes interessadas que desempenharam um papel crucial no desenvolvimento e implementação do Deco AI Sales Assistant. Isso inclui a equipe de engenharia que projetou e construiu o sistema, os parceiros e clientes que contribuíram com insights vitais para o aprimoramento do produto e os usuários finais, cujas interações e feedbacks moldam a direção futura do assistente de vendas.

2.2.1 Colaboradores

O projeto do Deco AI Sales Assistant foi desenvolvido por mim em conjunto com outros engenheiros da Deco, em principal, Marcos Candeia (ex-VTEX e ex-Microsoft) e Caroline Uchôa (ex-IBM). Marcos foi responsável por desenvolver a arquitetura do sistema e o projeto inicial, enquanto Caroline fez grande parte do refinamento e adaptação do projeto para produção com a minha coautoria. Além disso, é importante citar outros funcionários que atuaram na orientação, correção de Pull Requests (PRs), divulgação do produto e definição do *roadmap*: Guilherme Rodrigues (CEO), Luciano Júnior (CTO), Rafael Crespo (CRO), Matheus Gaudencio (Software Engineer) e Lucas Ribeiro (Head of Brand).

2.2.2 Clientes

Durante o período de criação do projeto, foram feitas demonstrações do produto em eventos e para potenciais clientes. O projeto inicial foi implementado em um site feito com a Deco para a loja americana ALS. Esse projeto foi utilizado como exemplo durante a National Retail Federation (NRF), o maior evento de varejo mundial, na qual a Deco, apresentou o produto a diversos potenciais clientes e conseguiram ter um *feedback* de novas necessidades e

[15] Plausible. Disponível em: https://plausible.io/. Acesso em 25 Abr. 2024.
[16] Google Analytics. Disponível em: https://developers.google.com/analytics. Acesso em 25 Abr. 2024.

melhorias. O pré-lançamento/divulgação do assistente de vendas (*sales assistant)* foi feito na NRF e no site deco.cx/ai-assistant, através das redes sociais.

Além disso, o segundo teste em produção foi feito com a loja ElaDecora, onde colocamos o assistente habilitado na loja. Inicialmente, deixamos livre para interação dos gerenciadores da loja por meio de uma página não listada. A partir do *feedback* deles fizemos alterações e, em seguida, adicionamos nas páginas da loja para que os usuários tivessem acesso por um período curto de tempo, uma noite. Isso foi feito através de um *matcher* de horário (que ativa ou desativa uma funcionalidade/componente de acordo com a satisfação ou não de um requerimento, nesse caso, intervalor de horário). O horário escolhido foi para que o assistente suprisse a falta de um canal de comunicação da loja no período noturno. Esse teste foi realizado em três oportunidades distintas, após progressão e atualização da ferramenta com os *feedbacks* recebidos.

2.2.3 Usuários

Os usuários finais do *sales assistant*, que de fato terão interação com a aplicação são os clientes das lojas interessadas e também os envolvidos no processo de criação, como os funcionários da Deco e os lojistas.

Durante o período de testes, coletamos *feedbacks* por meio de sessões de *bug bash*, conversas e testes ao vivo com os envolvidos e coleta de logs e métricas no HyperDX.

3. DESENVOLVIMENTO

3.1 Visão Geral

O Deco AI Sales Assistant é uma solução que tem como objetivo trazer o que há de mais novo em tecnologia, tanto para as lojas virtuais, quanto para seus clientes. Isso acontece com o uso de modelos de LLM para criação de uma jornada de compra personalizada e eficiente, que se alavanca do modelo conversacional similar ao disponível nas interações humanas nas lojas físicas. Porém, a solução proposta não tem limite de pessoas que podem ser atendidas simultaneamente, ou seja, de forma escalável, rápida e com recomendações precisas.

Além de beneficiar o cliente, há também um ganho considerável para as lojas. É previsto que essa inovação possa ter como efeitos: maior satisfação e fidelidade à loja, aumento das conversões, redução do custo de atendimento ao cliente, obtenção de *insights* valiosos sobre o comportamento do consumidor, etc.

A aplicação foi desenvolvida a partir da plataforma Deco, utilizando como tecnologias: Fresh, deno, React, Tailwind e AWS S3, Open AI APIs, etc. React e Tailwind foram utilizados para a criação tanto do lado do servido*r* (toda a infraestrutura que se utiliza), quanto do lado do cliente (chat disponível nas lojas). O design do produto foi desenvolvido utilizando Figma, enquanto as funcionalidades que estariam presentes nessa primeira versão foram decididas em conjunto pelo resto do time.

Nas Figuras 05 e 06 podem ser observados algumas imagens do design inicial do *chat.* Na Figura 05, o usuário manda uma mensagem em inglês perguntando quais os melhores tênis de trilha que a loja possui. Em seguida, o assistente pergunta qual o tamanho e qual o destino, para que possa fazer recomendações mais precisas. Na Figura 06, por sua vez, após a resposta do usuário com relação ao lugar e tamanho, o assistente de compras pontua o clima na região e recomenda opções de tênis que seriam mais adequados. As opções podem ser vistas dentro do *chat*, já com uma pequena descrição dos produtos, imagem e botão de adicionar ao carrinho.

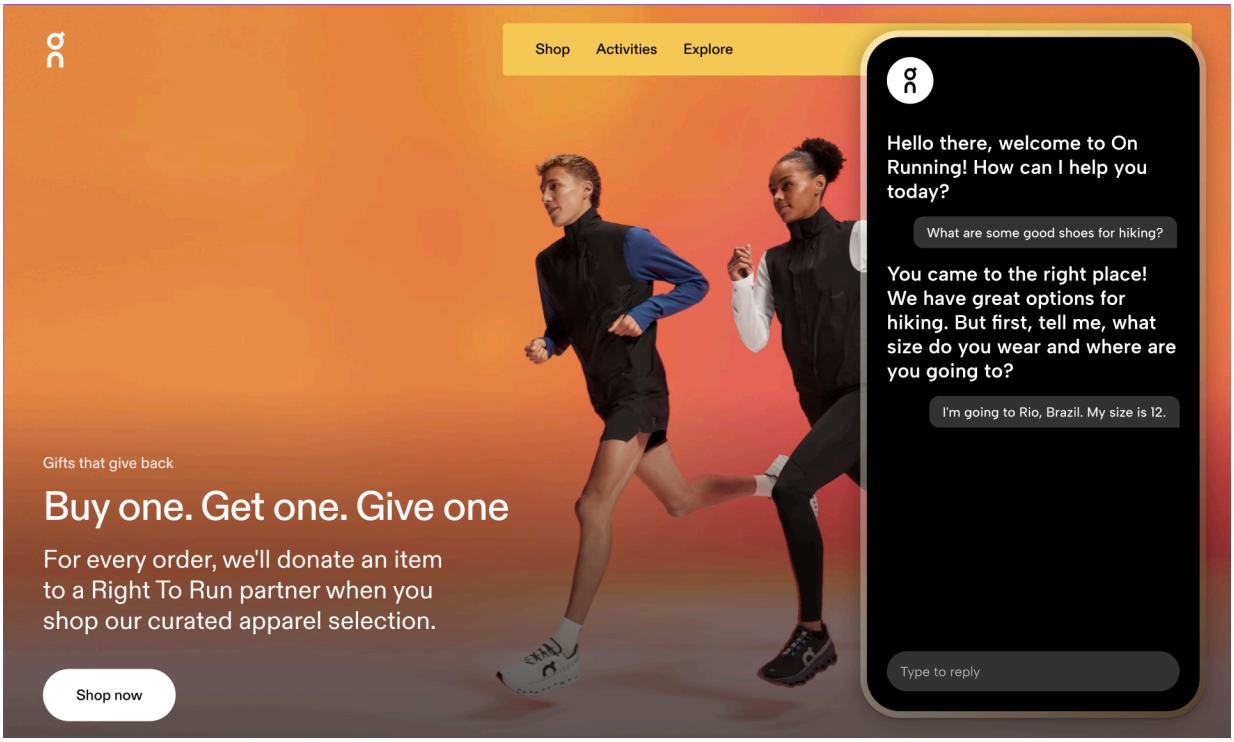

Figura 05: Interação inicial - Chat

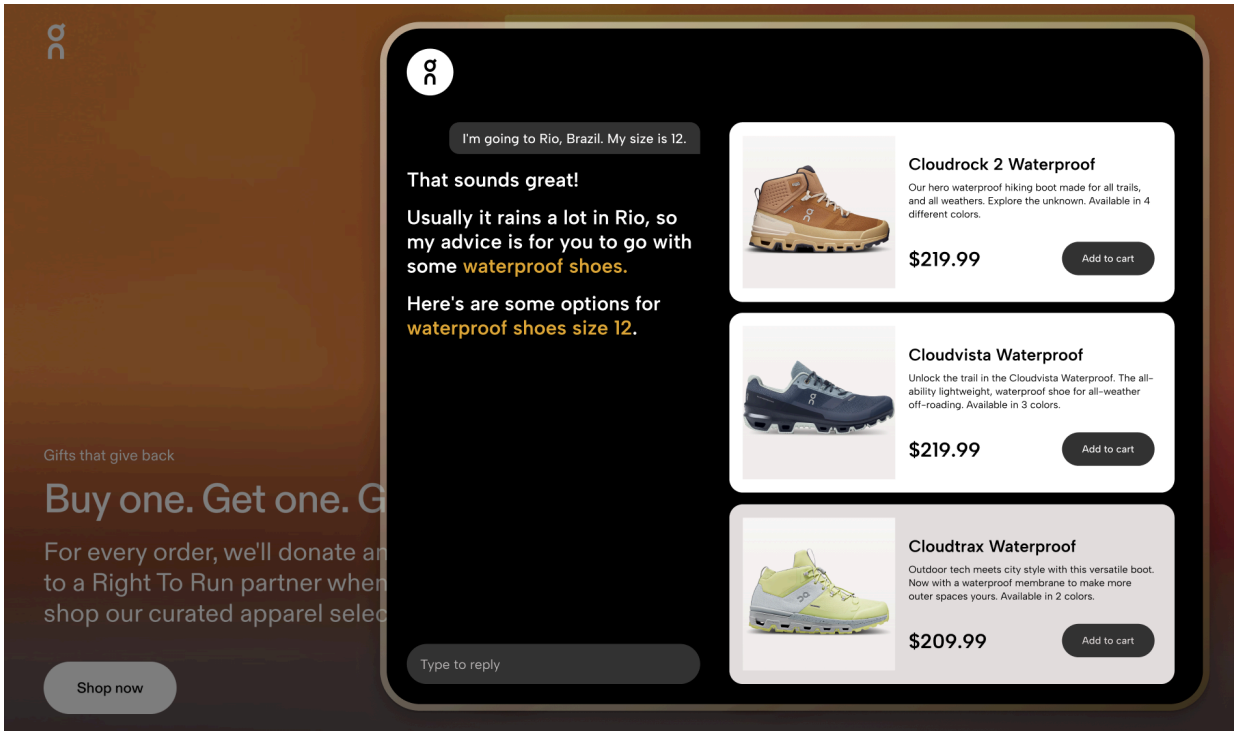

Figura 06: Recomendação de produtos a partir de conversa

Este assistente de vendas inteligente é projetado para funcionar 24 horas por dia nos 7 dias da semana, utilizando algoritmos avançados para garantir que cada cliente receba uma experiência de compra personalizada e interativa. Com a implementação do Deco AI Sales Assistant, espera-se não apenas melhorar a experiência de compra dos clientes, mas também otimizar as operações de vendas das lojas virtuais, aumentando a eficiência e reduzindo os custos operacionais.

Ao integrar tecnologias de ponta como LLMs da OpenAI, o assistente se adapta às necessidades e preferências dos usuários, oferecendo suporte e recomendações personalizadas. Além disso, o uso de tecnologias web modernas na construção do assistente garante uma interface ágil e responsiva, capaz de se integrar perfeitamente às diversas plataformas de e-commerce.

Em resumo, o Deco AI Sales Assistant representa uma evolução significativa na maneira como as lojas virtuais interagem com seus clientes, proporcionando um ambiente de vendas mais dinâmico, intuitivo e eficaz. Tudo isso utilizando as tecnologias da Deco, o que torna o produto de fácil uso e instalação.

3.2 Requisitos Funcionais

De acordo com o livro Engenharia de Software Moderna [9], Requisitos definem o que um sistema deve fazer e sob quais restrições. Requisitos relacionados com a primeira parte dessa definição — o que um sistema deve fazer, ou seja, suas funcionalidades — são chamados de Requisitos Funcionais. Abaixo, estão descritos os requisitos funcionais para o Deco AI Sales Assistant. Estes, estarão definidos em dois tipos diferentes: requisitos do cliente (que focam nas funcionalidades do usuário do assistente) e requisitos do gerenciador da loja (que focam na pessoa que de fato vai configurar o assistente e colocar em produção).

3.2.1 Do cliente

3.2.1.1 [RF01] Abertura e fechamento do *chat*

A aplicação deve permitir que o usuário abra e feche o *chat*, ao clicar em um botão específico.

3.2.1.2 [RF02] Comunicação via texto

Deve haver uma caixa de texto onde os usuários podem digitar e enviar mensagens para o assistente. O sistema deve ser capaz de receber e exibir mensagens do usuário e do próprio assistente.

3.2.1.3 [RF03] Envio de áudio

O produto deve permitir que o usuário grave e envie áudios através de clique em botão, que deve deixar claro visualmente, quando o áudio estiver sendo gravado.

3.2.1.4 [RF04] Envio de imagens

A aplicação deve permitir o envio de imagens através de botão disponível no *chat*. Ao clicar no botão, o usuário poderá enviar fotos de sua galeria, quando no desktop ou também tirar fotos, quando no celular.

3.2.1.5 [RF05] Histórico de conversas

O sistema deve reter o histórico de conversas na sessão, de forma que o usuário possa consultar suas últimas mensagens, enquanto estiver com o chat aberto.

3.2.1.6 [RF06] Limpeza de histórico

Deve haver um botão que permita ao usuário limpar o histórico de conversas e reiniciar o *chat*.

3.2.1.7 [RF07] Exibição de itens do catálogo

O *chat* deve ser capaz de exibir uma lista contendo itens do catálogo da loja, com base nos interesses, ou na conversa do usuário. Para isso, deve haver uma integração entre o assistente da loja e o sistema de gerenciamento de catálogo para exibir as informações de forma dinâmica.

3.2.1.8 [RF08] Adição de itens ao carrinho

O sistema deve prover botões nos itens exibidos que permitam a adição destes ao carrinho do usuário na loja.

3.2.1.9 [RF09] Limitação do tamanho do texto

A aplicação deve restringir o tamanho das mensagens de texto para 750 caracteres.

3.2.1.10 [RF10] Limitação da duração do áudio

Mensagens de áudio não devem exceder 1 minuto e 30 segundos de duração. O sistema deve ter uma funcionalidade para controlar a duração da gravação de áudio, interrompendo a gravação automaticamente, quando o limite for alcançado, ou avisando o usuário se o arquivo de áudio enviado exceder esse limite.

3.2.1.11 [RF11] Limitação do tamanho da imagem

O sistema deve permitir o envio de imagens com tamanho máximo de 4 MB. Deve haver uma validação no momento do envio, para garantir que as imagens não ultrapassem o tamanho máximo permitido, com *feedback* adequado ao usuário se a imagem for muito grande.

3.2.1.12 [RF12] *Cooldown* para envio de mensagens

O *chat* deve possuir um mecanismo para evitar que sejam enviadas muitas mensagens consecutivas, visto que é necessário chamar uma API de um LLM para obter a resposta, o que pode demorar alguns segundos e é custoso. Assim, ao adicionar uma limitação pode-se evitar o uso excessivo da API em um curto espaço de tempo e que ela seja utilizada antes do recebimento da resposta das mensagens enviadas anteriormente.

3.2.2 Do gerenciador da loja

3.2.2.1 [RF13] Especificação do identificador (ID) do assistente da OpenAI

O configurador do *chat* deve permitir a especificação do ID de um assistente da OpenAI, para determinar qual modelo será usado nas interações do *chat*.

3.2.2.2 [RF14] Definição de instruções para o assistente

O sistema deve permitir a especificação de instruções detalhadas que o assistente da OpenAI deve seguir, permitindo personalizar a forma como ele responde e interage no *chat*.

3.2.2.3 [RF15] Personalização das cores do assistente

Deve ser possível customizar as cores do *chat assistant*, incluindo a interface do *chat*, para corresponder à identidade visual da marca ou preferências do usuário.

3.2.2.4 [RF16] Nomeação do assistente

O sistema deve permitir que os administradores definam o nome do *chat assistant*, possibilitando uma experiência mais personalizada e identificável para os usuários.

3.2.2.5 [RF17] Seleção do tipo de personalidade do assistente

Deve ser possível especificar o tipo de personalidade que o *chat assistant* deve exibir, permitindo uma interação mais alinhada com o tom e os valores da marca.

3.2.2.6 [RF18] Integração para carregamento de produtos

O configurador deve oferecer uma função para integrar e carregar os produtos do catálogo da loja no *chat*, permitindo uma apresentação dinâmica dos produtos, conforme a interação do usuário.

3.3 Requisitos Não Funcionais

Enquanto os requisitos funcionais focam no que será implementado, os não funcionais especificam as características do sistema que não se relacionam diretamente com as funcionalidades específicas do software. Isto é, ao invés de funcionalidades, ele se concentra em atributos ou características de desempenho, qualidade, segurança, usabilidade, entre outros[17]. Nesta seção estão descritos os requisitos não-funcionais do assistente de compras.

3.3.1 [RNF01] Desempenho

O sistema deve responder às interações dos usuários rapidamente, com latência não superior a alguns segundos. De forma a não prejudicar ou adicionar fricção a experiência de busca do usuário.

[17] Coopersystem. "Requisitos Funcionais E Não Funcionais: O que São E Diferenças?". Disponível: coopersystem.com.br/requisitos-funcionais-e-nao-funcionais-o-que-sao-e-qual-e-a-diferenca/. Acesso em 23 Abr. 2024.

3.3.2 [RNF02] Disponibilidade

O *chat assistant* deve ter um tempo de atividade de 99,9% ou superior (com relação ao tempo que a API da OpenAI esteja funcionando), garantindo que o serviço esteja sempre disponível para os usuários.

3.3.3 [RNF03] Escalabilidade

O sistema deve ser capaz de atender múltiplos usuários, sem interrupção do serviço. Ou seja, ser capaz de processar um grande volume de mensagens e interações simultâneas, sem perda de performance.

3.3.4 [RNF04] Segurança

O sistema tem que ter medidas para evitar que haja ataques contra seu funcionamento normal, como *spam* de mensagens ou *prompt injection* (ataque usando engenharia de prompt através de texto para tentar tirar o modelo de linguagem do comportamento esperado e programado dele).

3.3.5 [RNF05] Confiabilidade

O sistema deve funcionar de forma contínua e correta, com mecanismos de recuperação de erros e redundância para minimizar falhas e perda de dados. Por exemplo, em caso de erro no envio ou recebimento de mensagem, o *chat* deve informar ao usuário o problema, mas continuar com a possibilidade de uso dele.

3.3.6 [RNF06] Usabilidade

O *chat assistant* deve ter uma interface intuitiva e amigável, com design responsivo adaptável a diferentes dispositivos e tamanhos de tela.

3.3.7 [RNF07] Manutenibilidade

O código do sistema deve ser bem documentado, organizado e modular, facilitando a manutenção e a adição ou modificação de funcionalidades, além de ser código aberto.

3.3.8 [RNF08] Testabilidade

O sistema deve ser facilmente testável, com suporte para testes manuais em diferentes níveis através do admin da deco e da utilização em localhost.

3.3.9 [RNF09] Interoperabilidade

O *chat assistant* deve ser capaz de integrar-se eficientemente com outros sistemas e plataformas usados pela loja de *e-commerce*.

3.3.10 [RNF10] Localização e Internacionalização

O sistema deve suportar múltiplos idiomas e culturas para atender uma base de clientes global (focada sobretudo nos idiomas inglês, português e espanhol). Inicialmente, limitado aos idiomas aos quais os modelos da OpenAI oferece suporte, como por exemplo: Chinês, Francês, Alemão, Italiano, Japonês, Português, Russo, Espanhol e Inglês[18].

3.3.11 [RNF11] Portabilidade

A instalação do Sales Assistant em outras lojas/sites deve ser simplificada e bem documentada.

3.4 Casos de Uso

Nessa seção, serão abordados os possíveis casos de uso do Deco AI Sales Asssistant. A critério de comparação, também será apresentada a experiência de compra de um usuário em uma loja que não possui um comprador pessoal (*personal shopper)*.

3.4.1 Experiência sem *Sales Assistant*

Em uma loja virtual sem um *Sales Assistant*, o usuário assume a responsabilidade por toda a jornada de seleção de produtos para compra:

- Navegação: precisa navegar pelas categorias do site, pesquisar por palavras-chave relacionadas ao produto desejado e utilizar filtros para refinar a busca, excluindo produtos, tamanhos e cores que não se encaixam em suas necessidades.

[18] OpenAI Languages supported . Disponível em: https://help.openai.com/en/articles/8357869-how-to-use-chatgpt-in-a-language-other-than-english-alpha. Acesso em: 29 Abr. 2024.

- Pesquisa: a pesquisa exige tempo e esforço, pois o usuário precisa ler descrições de produtos, comparar preços e características, e verificar a disponibilidade de estoque.
- Dúvidas: caso tenha dúvidas sobre produtos, ou o processo de compra, o usuário precisa buscar informações por conta própria, seja através de FAQs, *chat online* ou contato com o suporte da loja.
- Recomendações: as recomendações de produtos geralmente se baseiam em histórico de compras ou popularidade, nem sempre se adequando às necessidades específicas do cliente. Além disso, a experiência de recomendações personalizadas direcionadas às necessidades específicas ou aos questionamentos do usuário é relativamente raras em lojas de *e-commerce*. Além disso, as recomendações personalizadas atuais, que são relativamente comuns, são limitadas às interações do usuário com a plataforma (histórico de compras, visualizações ou buscas passadas).

3.4.2 Experiência com Deco AI Sales Assistant

Com o Deco AI Sales Assistant, há uma mudança em relação ao paradigma anterior de experiência de compra. Antes, o usuário era responsável por "encontrar suas próprias respostas", passando de forma padrão pelos tópicos da seção anterior.

A partir do uso de um assistente pessoal para compras que funciona através de uma *chat*, todo esse processo de filtragem, navegação e busca fica encapsulado em um processo conversacional que tem conhecimento/acesso ao catálogo inteiro da loja e pode fazer recomendações que se adequem melhor ao que o usuário procura. O processo então passa a consistir em:

- Conversar com o assistente expressando o interesse ou como ele pode ajudar.
- Refinar a intenção através de perguntas feitas pelo assistente.
- Receber recomendações de produtos baseadas nas necessidades específicas, podendo aceitar-las ou prosseguir com mais refinamento ou em outra direção.

Isso torna a experiência mais personalizada e mais eficiente. Alguns pontos de melhoria a se destacar são:

- Interação natural: o usuário pode interagir com o assistente de forma natural, como se estivesse conversando com um vendedor em uma loja física, utilizando linguagem simples e perguntas diretas.
- Compreensão profunda: o Deco AI Sales Assistant é capaz de compreender o contexto da conversa, identificar as necessidades do cliente e oferecer soluções personalizadas.

- Recomendações precisas: o assistente utiliza IA para gerar a melhor forma de pesquisar produtos no catálogo da loja, com o intuito de atender as necessidades do cliente, levando em consideração suas preferências e estilo.
- Agilidade e conveniência: o processo de compra se torna mais rápido e prático, pois o assistente pode realizar tarefas como busca e filtragem de forma automática
- Suporte e orientação: o Deco AI Sales Assistant oferece suporte e orientação durante todo o processo de compra, respondendo dúvidas e auxiliando o cliente na escolha dos produtos.

Exemplos de Uso:

- Encontrar o presente ideal: o usuário pode informar ao Deco AI Sales Assistant o perfil da pessoa que deseja presentear, como idade, hobbies e estilo, e o assistente irá recomendar produtos personalizados.
- Montar um *look* completo: o usuário pode pedir ajuda ao Deco AI Sales Assistant para montar um *look* completo, desde a roupa até os acessórios, de acordo com seu estilo e ocasião.
- Obter ajuda com um produto: o usuário pode tirar dúvidas sobre um produto, como forma de uso, cuidados e compatibilidade com outros produtos.

3.5 Arquitetura do Sistema

A arquitetura do sistema do Deco AI Sales Assistant é fundamental para entender como a solução é estruturada e opera de forma eficaz. Esta seção detalha a construção e o funcionamento dos componentes principais que constituem o assistente de vendas, divididos em lado do servidor (*server-side*), lado do cliente (*client-side*) e o gerenciador de conteúdo (*content management system*). Cada subseção abordará as tecnologias envolvidas, as interações entre os componentes do sistema e como eles se integram para criar uma experiência de usuário coesa e eficiente.

Como ilustrado na Figura 07, a arquitetura é dividida em várias camadas que funcionam em conjunto para proporcionar uma experiência fluida ao usuário. Nela, temos uma divisão em três partes principais que serão abordadas com mais detalhe nas próximas sessões: o Deco *Admin* (CMS), o lado do cliente e o lado do servidor. De forma resumida, o *admin* é responsável por gerenciar o conteúdo do Sales Assistant, além de ser o ponto de partida para configurar

tanto o lado do cliente quanto o lado do servidor. Durante o funcionamento do assistente, o lado do cliente e do servidor tem uma comunicação contínua através de conexões *websocket* e via *loaders*. Isso será abordado em mais detalhes nas próximas subseções.

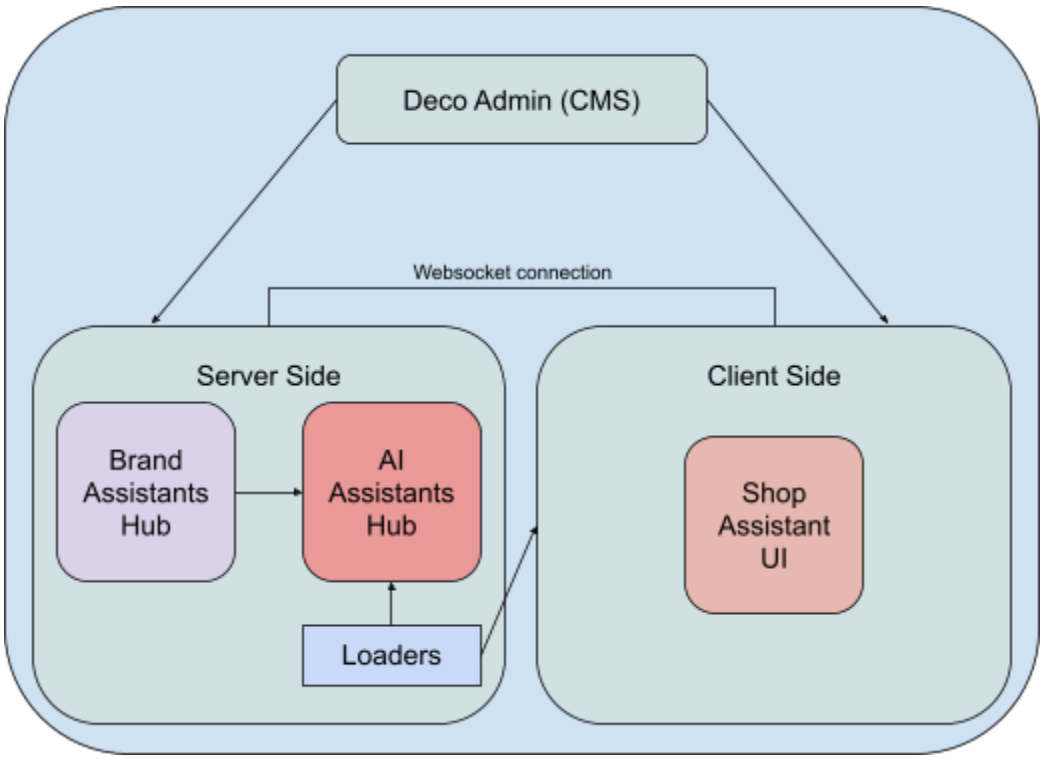

Figura 07: Arquitetura simplificada do sistema

3.5.1 *Server-side*

No lado do servidor, temos a combinação de diversos módulos que trabalham. O 'Brand Assistants Hub' serve como um repositório centralizado para as informações relacionadas às marcas e é essencial para manter a consistência e relevância das interações do assistente.

O 'AI Assistants Hub' é onde os modelos de linguagem são gerenciados e adaptados para as necessidades específicas do *e-commerce*, permitindo que o assistente entenda e responda de forma precisa às questões dos clientes. Este módulo utiliza APIs de terceiros e algoritmos de LLM para processar solicitações, gerenciar dados e integrar com outras funcionalidades do sistema.

Os 'Loaders' são responsáveis por carregar o catálogo de produtos e outras informações necessárias para o funcionamento do assistente, assegurando que a informação apresentada ao cliente seja atualizada e relevante. Nos assistentes implementados até então,

foram utilizados loaders da VTEX para pegar as principais buscas das lojas, listar categorias e procurar produtos.

3.5.2 *Client-side*

No lado do cliente, temos a 'Shop Assistant UI', a interface com a qual os usuários interagem diretamente. Esta seção é projetada para ser intuitiva e fácil de usar, permitindo que os clientes se comuniquem-se com o assistente por texto, voz ou imagens. O design é responsivo, assegurando que a experiência do usuário seja consistente em todos os dispositivos e navegadores. Esta camada lida com a apresentação das informações e com a interatividade do usuário, convertendo as interações em solicitações, que são enviadas ao servidor para processamento Isso é alcançado por meio de uma conexão websocket, que permite uma comunicação bidirecional em tempo real, garantindo que as interações do usuário sejam processadas rapidamente e as respostas sejam entregues sem atrasos perceptíveis. Além do websocket, também são utilizados os loaders para executar outras operações como transcrição de áudio e geração de descrição das imagens.

3.5.3 *Content Management System*

O Deco Admin funciona como o gerenciador do conteúdo do Sales Assistant, onde os gerentes de loja podem configurar e personalizar a experiência do assistente. Aqui, os usuários podem adicionar ou atualizar o conteúdo do catálogo, configurar respostas automatizadas, e monitorar o desempenho do assistente através de métricas e *logs*. A integração com o 'AI Assistants Hub' e 'Brand Assistants Hub' é gerenciada aqui.

A arquitetura do sistema é projetada para ser escalável e manutenível, garantindo que a solução possa evoluir com as demandas do mercado e as expectativas dos clientes. Com a fundação robusta proporcionada por esta arquitetura, o Deco AI Sales Assistant está bem posicionado para oferecer uma experiência de e-commerce revolucionária, alavancando o poder dos modelos de linguagem avançados para melhorar a interação entre as lojas virtuais e seus clientes.

3.6 Engenharia de *Prompt*

Engenharia de *prompt* é uma técnica utilizada em LLMs que envolve a elaboração de instruções que servirão como base para guiar o modelo em relação à geração de respostas ou

ações específicas mais direcionadas ao contexto/objetivo desejado. Com isso, é possível melhorar a eficácia do modelo ao lidar com tarefas específicas, como no caso do Sales Assistante, e-commerce.

No trabalho desenvolvido, foram elaborados dois conjuntos de instruções. O primeiro, disponível no *Brand Assistants Hub*, consiste em um *prompt* que é universalmente relevante para qualquer ambiente de e-commerce. Este prompt foi projetados para engajar os usuários em conversas produtivas, orientando-os na exploração de produtos, esclarecendo dúvidas e auxiliando na tomada de decisões de compra. Também direcionando o assistente a se comportar como um assistente de compras e com instruções que tem como objetivo evitar que ele saia do seu papel programado. O segundo conjunto de instruções consiste nas instruções específicas de cada loja. Este, adiciona mais contexto do segmento além de incluir dados provenientes da loja como: principais buscas, categorias e exemplos de produtos.

Por fim, é importante pontuar que foram tomadas medidas para evitar alterações no comportamento do assistente por injeções de prompts do lado do usuário, mas isso ainda é um risco que pode acontecer. Contudo, por causa da maneira que foi configurado a busca de produtos do assistente com *function calling*, o nome dos produtos, suas informações e seus valores são provenientes diretamente do catálogo da loja e, portanto, não podem sofrer alterações pelo usuário.

3.7 Instalando o Deco AI Sales Assistant

Atualmente, a instalação/habilitação do código referente ao Deco AI Sales Assistant nas lojas da deco ainda não é automática e requer alguns passos[19]. A integração pode ser feita de forma manual ou através da configuração utilizando um arquivo de *Patch* do GitHub. Além disso, também é necessário configurar o assistente no deco admin e adicionar as variáveis de ambiente necessárias para a execução apropriada.

3.7.1 Integração Manual

A integração manual exige uma série de etapas, começando pela cópia dos arquivos necessários do repositório oficial para os diretórios correspondentes na loja. Os arquivos essenciais incluem componentes de chat, sessões, componentes interativos específicos para a

[19] Instruções para instalar o Deco AI Sales Assistant. Disponível em: github.com/deco-sites/storefront/blob/sales-assistant/assistant/ASSISTANT.md. Acesso em 23 Abr. 2024.

funcionalidade do assistente e estilização. É importante personalizar o nome do chat e garantir que todas as configurações, como ícones e cores do chat, estejam corretas no tema Tailwind.

3.7.2 Integração com Patch File

Para uma integração mais rápida, pode-se utilizar um arquivo de *patch* pré-preparado, aplicando-o ao repositório local da loja. Esta opção simplifica o processo de configuração e garante que todos os componentes necessários para a funcionalidade do assistente sejam incluídos corretamente. Após aplicar o *patch*, é essencial verificar a correta implementação das alterações, por meio da etapa de *build* do *manifest* e execução da loja localmente.

3.7.3 Configuração no deco admin

Dentro do painel de administração da deco, os lojistas devem configurar o 'Brand Assistant Hub' e o 'AI Assistant Hub', que devem aparecer como mostrado na Figura 08.

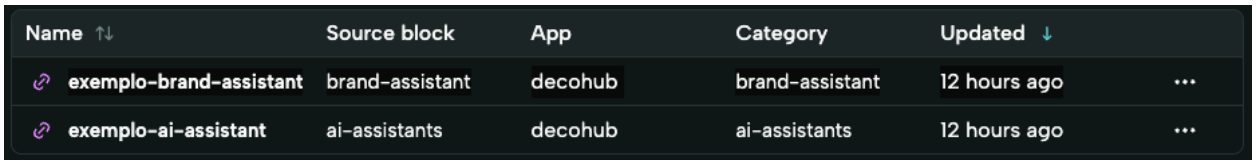

Figura 08: Exemplo de apps instalados

Isso inclui a criação do assistente, o preenchimento de instruções do assistente, adição de um ID correspondente a um assistente na *Assistants Hub da OpenAI* e a adição de nome e mensagens de boas-vindas, se necessário, tem-se como exemplo o formulário da Figura 09. Estas configurações asseguram que o assistente funcione conforme o esperado e esteja alinhado com a identidade da marca.

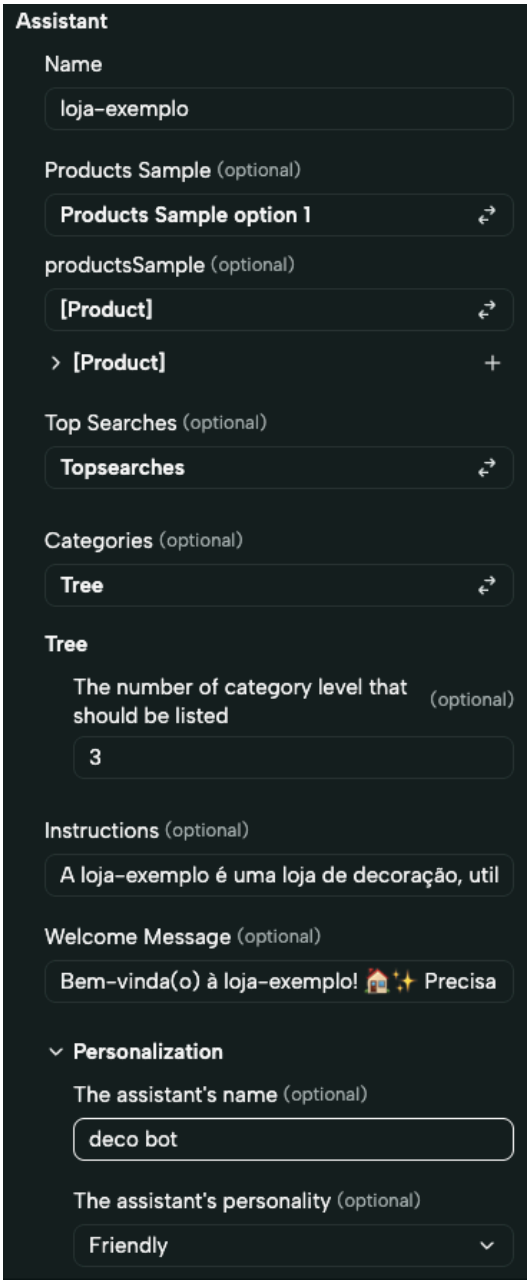

Figura 09: Formulário com informações a serem preenchidas sobre o assistente

Por fim, também é necessário configurar e estilizar a interface do assistente. Essa configuração, também feita no admin, é composta do formulário disponível na Figura 10. Nele, é possível customizar as cores do chat, assim como a logo utilizada e visualizar o chat aberto e fechado. O chat aberto, como disposto, possui botões para fechar o chat, limpar a conversa, escrever texto, mandar fotos e gravar áudios.

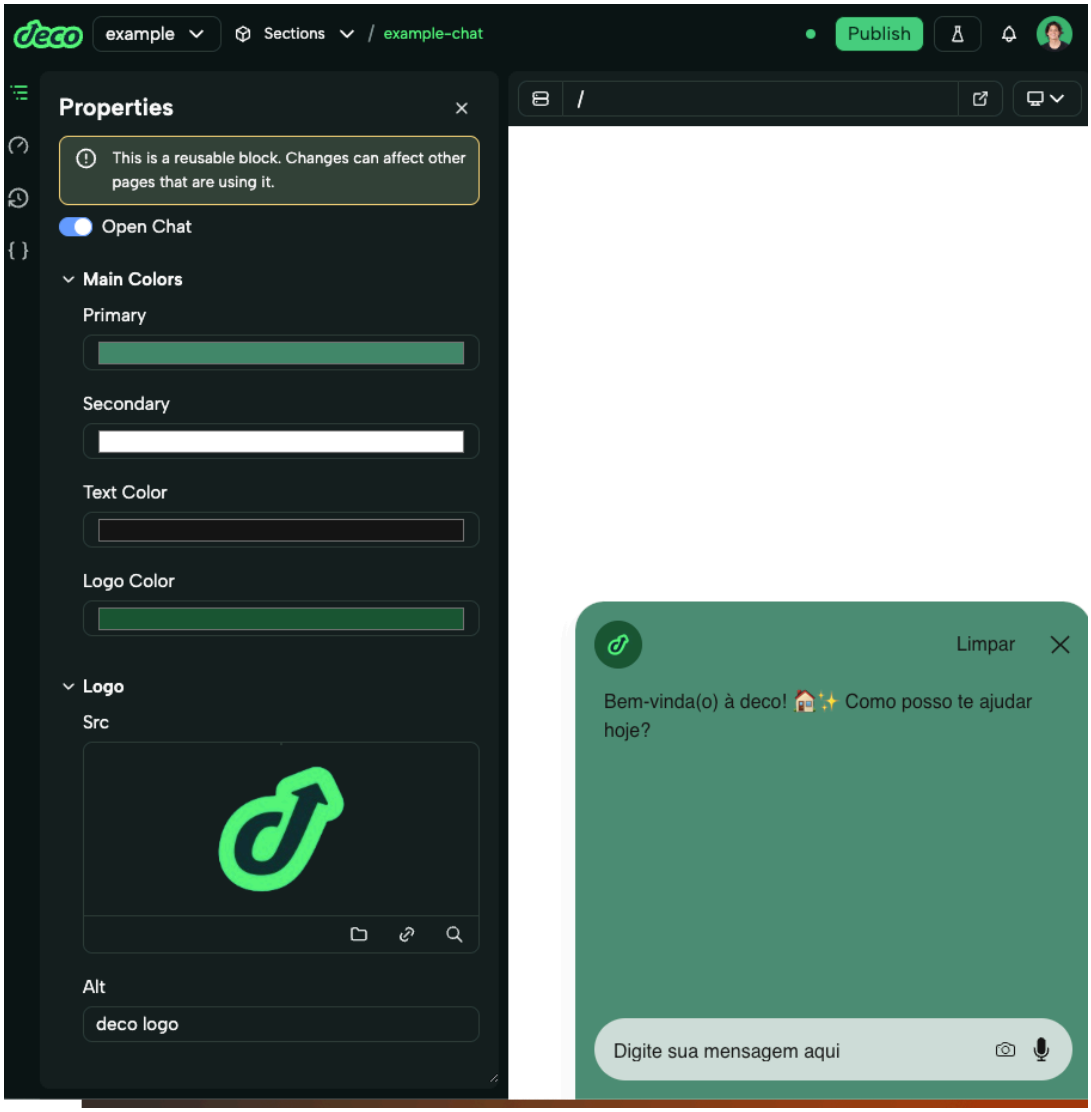

Figura 10: Exemplo de configuração da interface do chat no admin

3.7.4 Variáveis de ambiente

Finalmente, a configuração das variáveis de ambiente é um passo crucial para a execução apropriada do assistente. Isso envolve definir chaves para a API da OpenAI, serviços de armazenamento AWS, e outros parâmetros essenciais que o sistema utiliza para operar. Estas variáveis devem ser estabelecidas tanto localmente (quando o assistente for ser testado) quanto nas variáveis de ambiente no Deno Dash (para que seja utilizado em ambiente de *deploy*/produção).

4. Análise dos resultados

A análise dos resultados é um componente crítico para avaliar a eficácia do Deco AI Sales Assistant, permitindo a identificação de áreas de melhoria e validando o desempenho do sistema em diferentes cenários. Esta seção detalha o plano de testes, as personas envolvidas, a execução dos testes, as métricas de desempenho utilizadas e os métodos de *debugging* aplicados.

4.1 Plano de testes

Este plano de testes é destinado a avaliar o Deco AI Sales Assistant em todas as suas funcionalidades principais, simulando as interações típicas de usuários durante uma jornada de compra na loja virtual. O objetivo dos testes é identificar e corrigir potenciais problemas antes da implementação final, assegurando que o sistema esteja nas melhores condições de uso possíveis.

Para garantir uma análise abrangente, foram definidas personas representativas dos diferentes segmentos de usuários da loja, incluindo variações de idade, comportamento de compra e familiaridade com a tecnologia. Essas personas foram usadas para simular o processo de interação com o *chat*, desde a descoberta de produtos até a finalização da compra.

Os testes foram estruturados para cobrir o fluxo completo de uso do assistente, incluindo o acesso ao *chat*, a interação por meio de texto, áudio e imagem, a visualização de produtos recomendados, a adição de itens ao carrinho e o processo de *checkout*. Além disso, foram realizados testes para avaliar a capacidade do sistema de responder a consultas complexas e processar solicitações simultâneas em diferentes condições de carga.

A execução dos testes foi realizada pelo autor deste trabalho, com o suporte de ferramentas digitais adequadas. Para a realização dos testes, foram utilizados dispositivos com sistemas operacionais variados, incluindo Windows, Mac OS e Android, e a aplicação foi acessada através do navegador Google Chrome, Safari e Firefox, para garantir a consistência e a replicabilidade dos resultados em diferentes ambientes tecnológicos.

Esta abordagem meticulosa no planejamento e execução dos testes visa não apenas validar a funcionalidade e a performance do Deco AI Sales Assistant, mas também otimizar a experiência do usuário, garantindo uma jornada de compra suave e satisfatória.

4.2 Personas

Estas personas ilustram a diversidade no uso do Deco AI Sales Assistant, mostrando como diferentes grupos de usuários podem interagir com o sistema de maneiras distintas, seja por meio de texto, áudio, imagem ou uma combinação desses métodos.

4.2.1 Maria, 68 anos, aposentada

- Uso do Assistente: Maria prefere usar comandos de áudio devido à sua limitada familiaridade com a tecnologia e dificuldade em digitar em dispositivos móveis.
- Comportamento: Ela usa o assistente para pedir recomendações de produtos, como suplementos de saúde e produtos de jardinagem. Maria frequentemente envia mensagens de áudio perguntando sobre os benefícios e características dos produtos.
- Expectativa: Ela espera que o assistente entenda claramente sua voz e responda com sugestões úteis e informações detalhadas sobre os produtos.

Um exemplo de experiência de interação de Maria com o assistente de compras está disponível na Figura 11. Nela, Maria interage através somente de áudios e consegue receber uma recomendação condizente com sua necessidade.

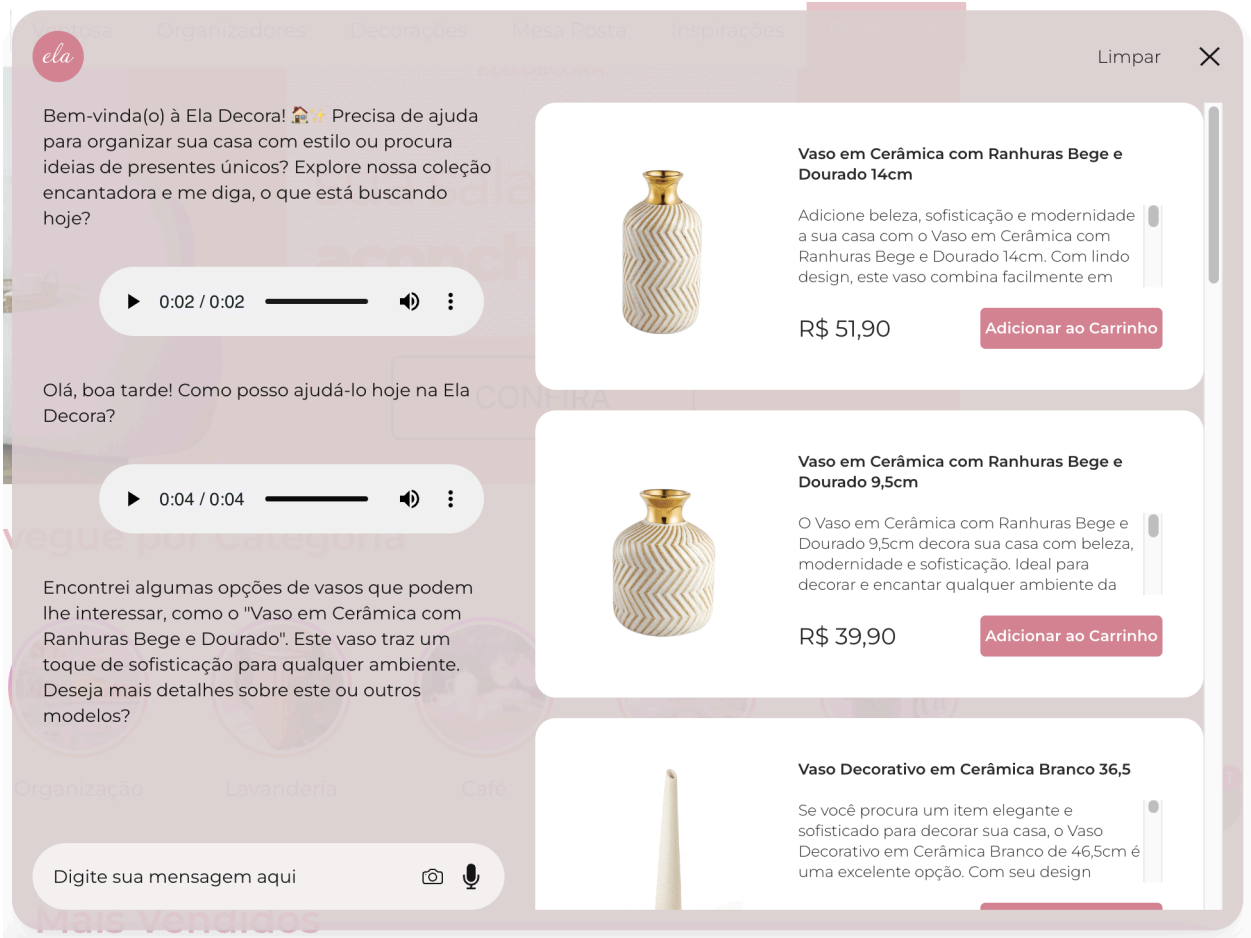

Figura 11: Experiência de compra de Maria com Áudio

4.2.2 Lucas, 22 anos, estudante universitário

- Uso do Assistente: Lucas está sempre em busca das últimas tendências esportivas e utiliza a função de texto e imagem para interagir com o assistente.
- Comportamento: Ele envia fotos de sapatos e acessórios que encontra em blogs e redes sociais, perguntando se a loja possui produtos similares ou melhores. Lucas valoriza respostas rápidas e informações precisas sobre especificações e preços.
- Expectativa: Ele espera que o assistente reconheça rapidamente os produtos nas imagens e forneça opções de compra relevantes.

Na Figura 12, tem-se uma possível interação de Lucas com o assistente através de texto e, na Figura 13, através de uma imagem de produto enviada pelo chat.

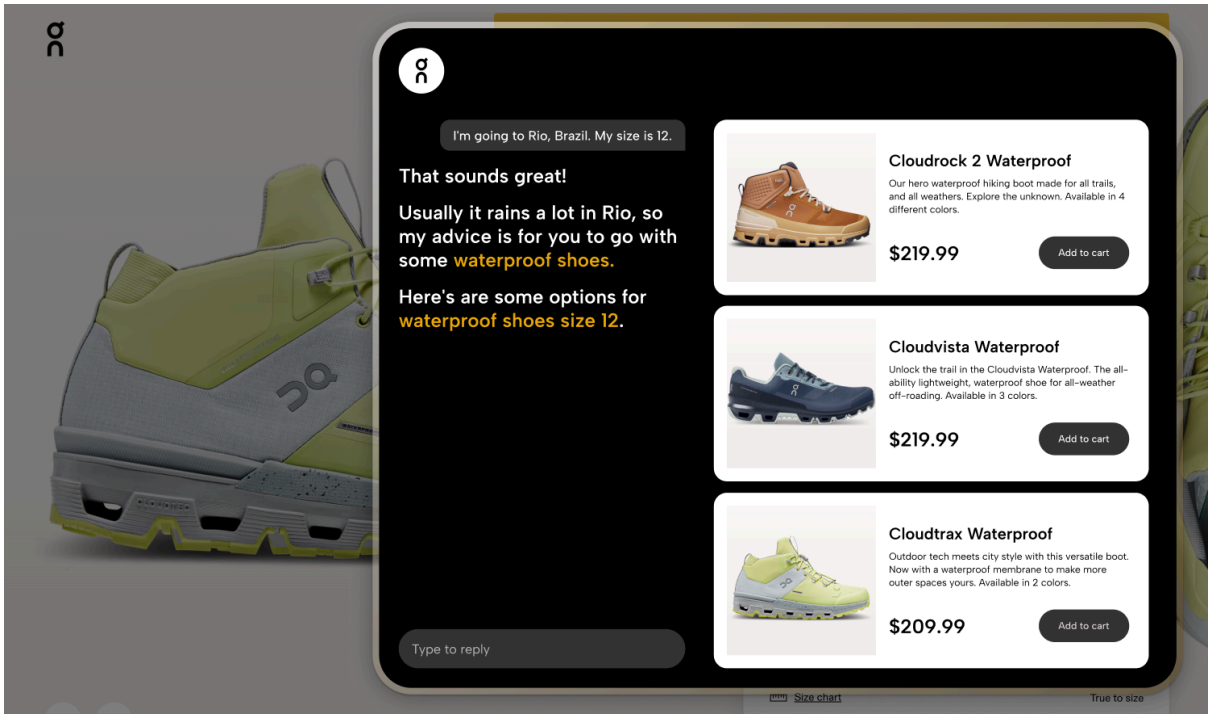

Figura 12: Experiência de compra de Lucas com texto

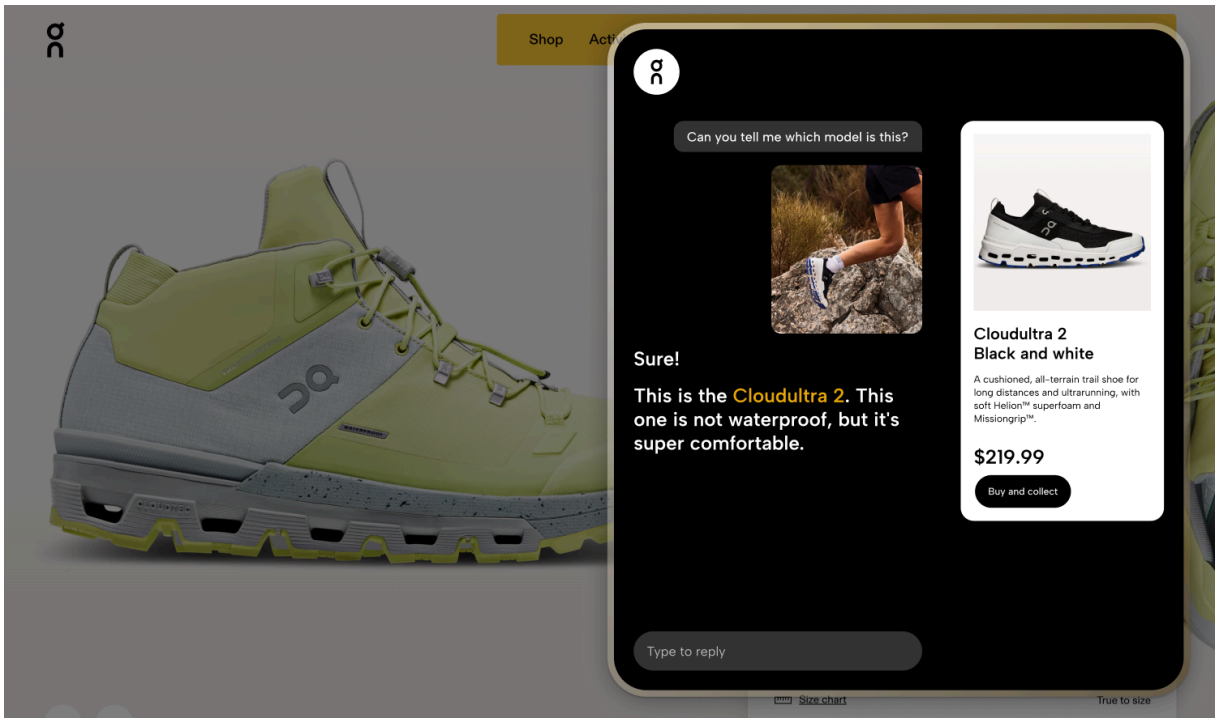

Figura 13: Experiência de compra de Lucas com Imagens

4.3 Testes

Os testes do Deco AI Sales Assistant são essenciais para garantir que o sistema funcione conforme esperado em diferentes ambientes, proporcionando uma experiência de usuário consistente e confiável. A execução desses testes em ambos os ambientes, desenvolvimento e produção, é crucial para identificar e corrigir problemas antes que o assistente seja amplamente adotado pelos usuários finais.

4.3.1 Teste em ambiente de desenvolvimento

Nesta fase, a aplicação foi executada localmente (*localhost*) e no modo de *preview* do Deno Deploy, utilizando exemplos de lojas como ElaDecora, ALS e Storefront da Deco. O objetivo desses testes foi verificar a funcionalidade básica do assistente, incluindo a interação por meio de texto, áudio e imagem, a precisão das respostas do assistente, a exibição correta dos produtos e a integração com as APIs de *back-end*.

Durante os testes no ambiente de desenvolvimento, foram simuladas diversas interações com o assistente para garantir que todos os componentes estavam funcionando corretamente. Isso incluiu a validação das funcionalidades de pesquisa de produtos, o processamento de linguagem natural dos comandos de texto e áudio, e a capacidade do sistema de lidar com consultas de imagens. Além disso, foi verificada a performance do *chat* em termos de tempo de resposta e precisão das informações fornecidas.

Também foi executada uma sessão de *bug bash*, um evento com o objetivo de descobrir o maior número de *bugs* em uma aplicação em um curto período de tempo. Além de *bugs*, o formulário também englobava possíveis destaques de funcionamento do produto e a indicação de novas funcionalidades. Esse evento teve participação de todos os empregados da deco e a partir dele foi possível descobrir novos erros e possíveis melhorias para a aplicação. Na Figura 14, pode-se observar um gráfico de pizza contendo a classificação de cada tipo de resposta submetida ao formulário.

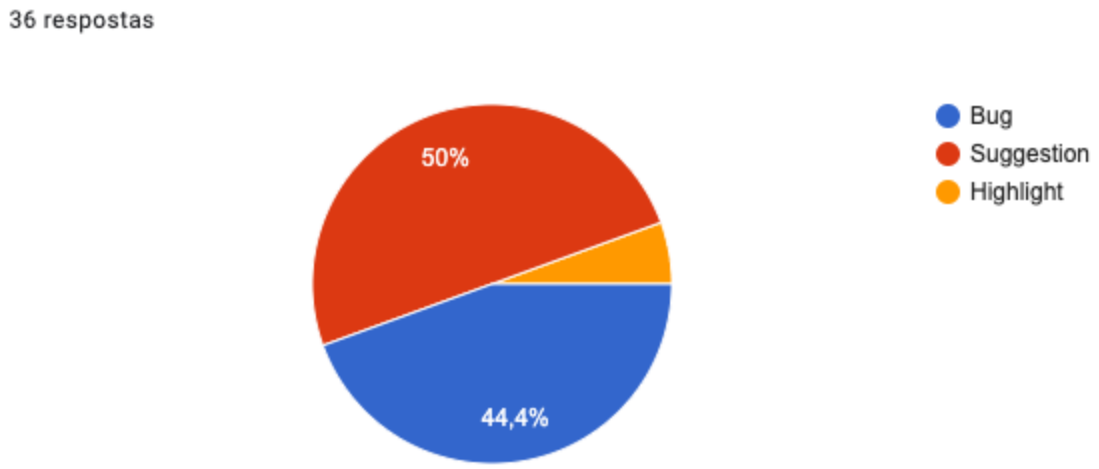

Figura 14: Gráfico de respostas contendo as 36 respostas do bug bash session

Os *bugs* encontrados, em sua maioria, tinham relação com o processo de retornar produtos do assistente (produtos não condizentes com a descrição do usuário, ou nenhum produto retornado), o design do *chat* e as respostas do assistente. Já as sugestões (*suggestions*) continham novas funcionalidades. Tanto os *bugs*, como as sugestões, foram todos mapeados e cadastrados para resolução e possível futura implementação.

4.3.2 Teste em ambiente de produção

Os testes em ambiente de produção foram realizados nas lojas ALS e ElaDecora. Esses testes visavam avaliar o desempenho do assistente em um cenário real de uso, onde a carga de tráfego e as interações dos usuários são mais imprevisíveis e diversificadas. Durante esta fase, foram observadas pouca instabilidade da aplicação, a consistência das respostas do assistente em diferentes tipos de consultas, e a integração com o sistema de *e-commerce* das lojas.

No entanto, esses testes em produção foram realizados de forma rápida e acabaram sendo postergados devido à priorização de outros projetos pela empresa. Isso significa que, embora os testes iniciais em produção tenham fornecido *insights* valiosos sobre o comportamento do assistente em um ambiente ao vivo, uma avaliação mais aprofundada ainda é necessária para entender completamente o desempenho e a usabilidade do sistema na prática.

4.3.3 Exemplos de bugs encontrados

Como mencionado nas sessões 4.3.1 e 4.3.2, foram encontrados alguns comportamentos inesperados durante o processo de desenvolvimento. Abaixo serão citados alguns.

Primeiramente, temos um problema que foi relativamente frequente no começo do desenvolvimento. A recomendação de produtos errados, mas próximos em uso ao que havia sido pedido inicialmente. No exemplo da Figura 15, temos o usuário pedindo recomendações de tênis de trilha e recebendo meias para trilha.

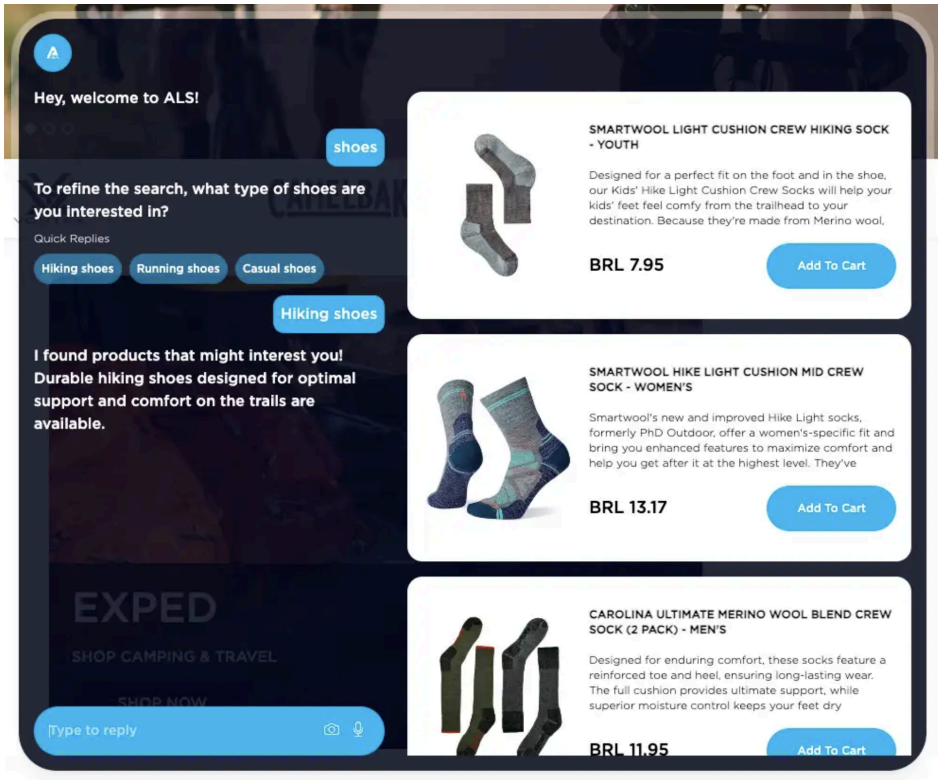

Figura 15: Ao pedir um tênis de trilha o usuário recebe recomendações de meias

Esse bug foi solucionado/aliviado com a adição de instruções extras ao assistente, além de um refinamento no mecanismo de busca utilizado. Além disso, também é importante citar outros bugs que não estão relacionados diretamente ao modelo de linguagem. Um deles é o de sobreposição de texto nos botões de imagem e áudio. Como pode ser visto na Figura 16, isso dificulta a leitura do texto e obviamente foge do funcionamento desejado e do design planejado.

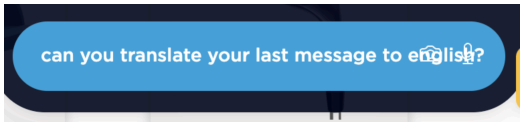

Figura 16: Texto sobrepondo ícones de envio de imagem e áudio

A solução desse erro consistiu na limitação do espaço ocupado pela caixa de texto. UM último erro a ser incluído nessa seção foi encontrado durante os testes em ambiente de produção. Nesse caso, o que ocorreu em uma das lojas foi um problema envolvendo o z-index do botão de finalizar compra presente no sidebar do carrinho e o botão de abertura do sales assistant. Da maneira que estava configurado, o sales assistant ficou sobreposto ao botão de finalizar compra, o que impossibilitou os usuários momentaneamente de concluírem sua compra. Por sua vez, esse problema foi solucionado com uma mudança no z-index do assistente.

4.4 Métricas

Para o monitoramento e aprimoramento do Deco AI Sales Assistant, implementamos várias métricas específicas, que são registradas através de código para rastrear e quantificar o desempenho e a confiabilidade do sistema, e que serão descritas nas próximas subseções.

4.4.1 Erros de *Upload* para a nuvem

A métrica *assistant_aws_upload_error* é utilizada para contar e analisar as ocorrências de erros durante o processo de *upload* de imagens para a nuvem, nesse caso, o Amazon Web Services (AWS). Isso é crucial para a detecção de problemas relacionados à rede ou integração com o serviço de armazenamento.

4.4.2 *Tokens* de *Prompt* de Imagem

O histograma *assistant_image_prompt_tokens* registra a quantidade de *tokens* utilizados nos *prompts* de entrada para a API da OpenAI, quando imagens precisam ser descritas. Este dado permite otimizar os *prompts* para melhor uso dos *tokens* disponíveis.

4.4.3 *Tokens* de Conclusão de Imagem

Da mesma forma, *assistant_image_completion_tokens* acompanha a quantidade de *tokens* produzidos como resposta pelos modelos da OpenAI. A eficiência nesta métrica é indicativa da precisão e da economia no uso do modelo de linguagem.

4.4.4 Erros na Descrição de Imagem

Erros ocorridos durante a descrição de imagens são monitorados pela métrica *assistant_describe_image_error*, oferecendo um meio para avaliar a confiabilidade dessa funcionalidade e a necessidade de aprimoramentos.

4.4.5 Tamanho do Áudio na Transcrição

A métrica *assistant_transcribe_audio_size* mede o tamanho dos arquivos de áudio recebidos, o que ajuda a monitorar a eficiência do sistema na gestão de arquivos de diferentes dimensões e a qualidade da transcrição.

4.4.6 Erros na Transcrição de Áudio

Incrementada por *assistant_transcribe_audio_error*, essa métrica quantifica os erros encontrados na transcrição de áudio, sinalizando áreas que podem precisar de atenção técnica imediata ou melhorias a longo prazo.

4.4.7 Latência de Resposta

A métrica *assistant_latency* é segmentada em diversas subcategorias para analisar a latência desde o momento de recebimento da solicitação, até o envio da resposta pelo sistema, incluindo várias etapas do processamento. Esses valores são fundamentais para entender e otimizar a velocidade de resposta do assistente.

4.5 Eventos

O registro de eventos é uma parte integral da análise de como os usuários interagem com o assistente e ajuda a entender o comportamento do consumidor, dentro do ecossistema de *e-commerce*. Para o registro de eventos, usamos tanto o Plausible, que fica disponível na

página de *analytics* do Deco *admin*, como o Google Analytics, que fica disponível para as empresas. A seguir está um detalhamento de como cada evento disparado pelo assistente:

4.5.1 Evento de Abertura e Fechamento do *Chat*

O evento *select_promotion* é disparado toda vez que o usuário clica para abrir ou fechar o *chat*. Este evento é crucial para rastrear a frequência com que o *chat* é acessado, oferecendo *insights* sobre a visibilidade e a atratividade do assistente. Além disso, a ação armazena o estado do *chat* na sessão do usuário, permitindo manter a consistência da experiência do usuário durante a navegação.

4.5.2 Evento de Visualização de Item

O evento *view_item* é acionado quando um produto é visualizado através do assistente. Este registro é fundamental para entender quais produtos estão atraindo mais atenção e como as recomendações do assistente influenciam a jornada de descoberta do usuário no site.

4.5.3 Evento de Adição ao Carrinho

Quando um usuário adiciona um produto ao carrinho, o evento *add_to_cart* é registrado. Isso não só permite monitorar quais produtos recomendados pelo assistente estão convertendo, mas também oferece uma oportunidade para analisar o valor médio do pedido e a efetividade das sugestões do assistente.

Esses eventos coletam dados valiosos que podem ser usados para otimizar a funcionalidade e a interface do usuário do assistente, a fim de promover uma experiência de usuário mais engajadora e eficiente, resultando em um aumento de conversões para a loja de *e-commerce*.

Na Figura 17, pode-se observar como os dados ficam disponíveis no *Plausible*. Na imagem, pode se observar os diferentes tipos de eventos, além de sua contagem e a frequência de conversão.

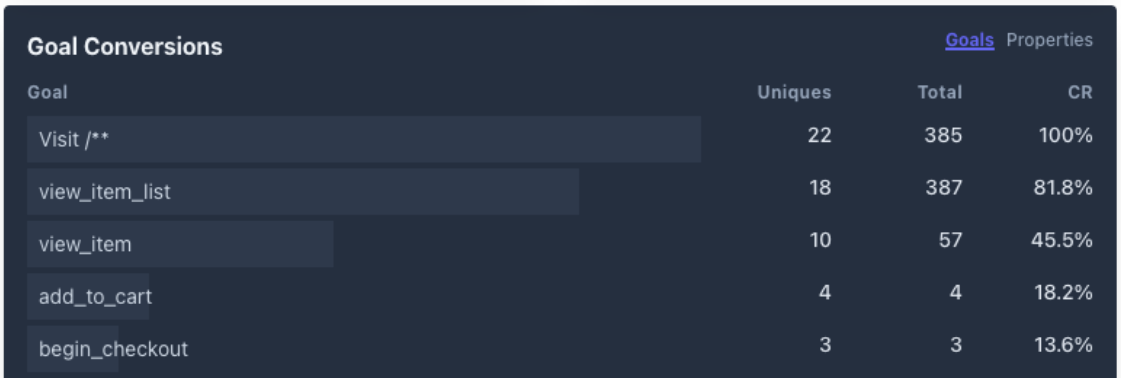

Figura 17: Exemplo de eventos no Plausible

Estes dados, juntamente com as informações de métricas, podem ser utilizados futuramente para medir o impacto do assistente e estimar seu custo de forma mais assertiva.

4.6 *Logging*

Ao implementar o *logging* no Deco AI Sales Assistant, utilizamos o HyperDX como uma solução robusta para capturar e registrar os detalhes de todas as interações. Os *logs* são ricos em informações, capturando cada aspecto da conversa entre o usuário e o assistente. Isso é essencial não apenas para o aprimoramento contínuo do modelo de linguagem que alimenta o assistente, mas também para fins de depuração e análise.

Cada entrada de *log* contém um carimbo de data/hora, um nível de severidade e uma mensagem que normalmente inclui um *'assistantId'* único e um *'threadId'*. O *'threadId'* é particularmente útil, pois permite que rastreamos uma sessão de conversa específica ou "*thread*" de interação do início ao fim. Isso é vital para manter o contexto durante a depuração e para entender a experiência do usuário em uma sequência de mensagens.

Além disso, detalhes como o *'context'* e *'subcontext'* são registrados, fornecendo uma visão mais aprofundada do que foi solicitado e como o assistente respondeu. Isso nos permite não só identificar possíveis falhas e gargalos, mas também analisar como o assistente está realizando em termos de compreensão e resposta às necessidades dos usuários. No Código 01, temos a exemplificação do código que loga a informação proveniente, por exemplo, da ação que transcreve o aúdio do assistente de vendas.

```javascript
const response = await ctx.openAI.audio.transcriptions.create({
    model: "whisper-1",
    file: file,
  });

  logger.info({
    assistantId: assistantId,
    threadId: threadId,
    context: "transcribeAudio",
    subcontext: "response",
    response: JSON.stringify(response),
  });
```

Código 01: Logging da resposta da transcrição de áudio feita com Whisper

O módulo responsável por gerar as descrições das imagens e fazer o *upload* das imagens para a AWS registram as informações de maneira similar, já a parte de texto é registrada a cada mensagem enviada e recebida como disposto no Código 02 e um exemplo da informação registrada na Figura 18.

JavaScript

```
    onMessageReceived: (logInfo: Log) => {
      logger.info({
        assistantId: logInfo.assistantId,
        threadId: logInfo.threadId,
        runId: logInfo.runId,
        context: "Message received",
        model: logInfo.model,
        message: JSON.stringify(logInfo.message),
      });
    },
    onMessageSent: (logInfo: Log) => {
      logger.info({
        assistantId: logInfo.assistantId,
        threadId: logInfo.threadId,
        runId: logInfo.runId,
        context: "Message sent",
        model: logInfo.model,
        message: JSON.stringify(logInfo.message),
      });
    },
```

Código 02: Código responsável por logar mensagem

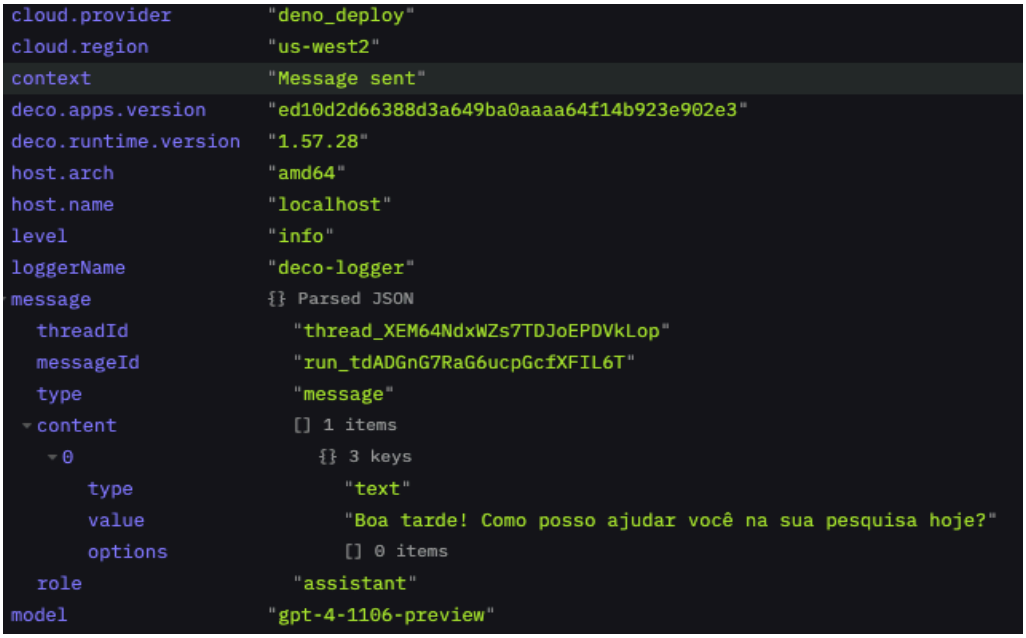

Figura 18: Exemplo de mensagem de *log*

Por meio desses registros detalhados, podemos realizar uma variedade de análises, desde avaliar o desempenho do assistente, até identificar padrões de uso dos usuários. Esses *insights* são cruciais para dirigir a evolução do produto e garantir que o assistente continue a atender e exceder as expectativas dos usuários. Com esses dados em mãos, a equipe pode iterar sobre o modelo do assistente, fazendo ajustes finos e refinando as capacidades de interação do sistema para fornecer um serviço mais preciso, útil e personalizado.

Um outro exemplo é na própria checagem do funcionamento do assistente, já que as mensagens trocadas entre o usuário e o assistente ficam disponíveis nos logs, é possível verificar se as interações estão indo como necessário além de poder mapear junto aos eventos as conversas que foram bem sucedidas e culminaram na adição dos produtos ao carrinho e eventual conversão.

4.7 Cálculo de Custo

Após a construção do mínimo produto viável, foi feita uma pequena análise de custo sobre o assistente e de viabilidade de cálculo de custo para cada loja que o implementasse. Os resultados estão disponíveis na Tabela 01. Nela, temos os diferentes modelos utilizados em cada fileira, seguidos de:

- Preço da entrada, baseado na página de custos da OpenAI[20].
- Máximo tamanho da entrada (que foi estabelecido através do código da aplicação, de forma a limitar o uso excessivo sem fins direcionados ao assistente ao mesmo tempo que não prejudique a experiência do usuário).
- Preço da saída, também baseado na página de custos da OpenAI.
- Máximo tamanho da saída, baseada também na página de custos da OpenAI.
- Tamanho médio da saída, uma medida empírica e aproximada baseada nos *logs* e métricas.

Modelo	Precificação da Entrada	Máximo Tamanho da Entrada	Precificação da Saída	Máximo Tamanho da Saída	Tamanho Médio da Saída
gpt-4-1106-preview	$0.01/1k *tokens*	750 palavras (aprox 1000 *tokens*)	$0.03/1k *tokens*	4096 *tokens*	Menos de 750 palavras (aprox 1000 tokens)
gpt-4-1106-vision-preview	$0.01/1k *tokens*	4 MB (aprox 760 tokens)	$0.03/1k *tokens*	4096 *tokens*	Menos de 750 palavras (aprox 1000 tokens)
Whisper	$0.006/minuto	No máximo 1:29 minutos	-	-	-

Tabela 01: Análise de custos do assistente

Como visto, foi feita uma limitação do tamanho máximo de forma a atender dos Requisitos Funcionais 09, 10 e 11. Essa limitação juntamente ao *cooldown* (Figura 19) estabelecido para envio de mensagens tem como objetivo manter os custos em uma média viável para o funcionamento efetivo da aplicação.

[20] OpenAI Pricing. Disponível em: https://openai.com/pricing. Acesso em 29 Abr. 2024.

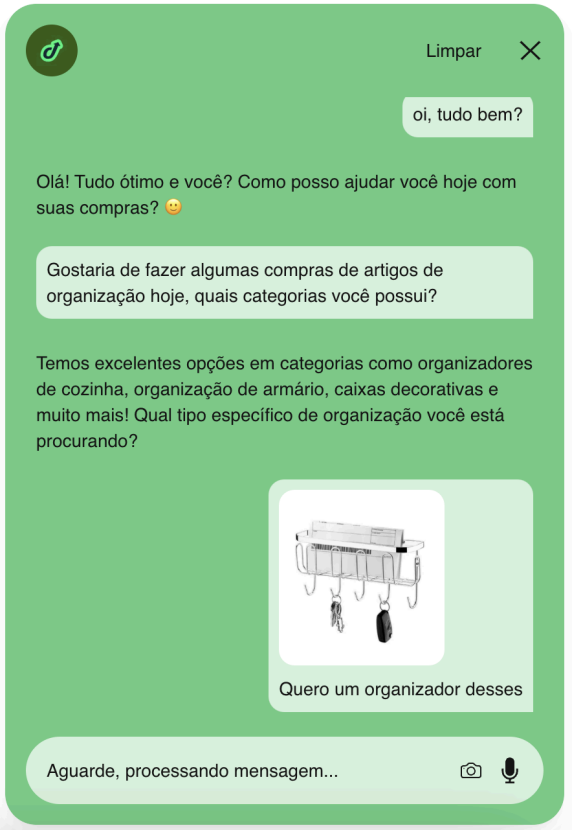

Figura 19: Mensagem de *cooldown* disponibilizada

5. Conclusão

O Deco AI Sales Assistant é uma solução inovadora que transforma a experiência de *e-commerce*, oferecendo um serviço de atendimento ao cliente personalizado e eficiente através do uso de LLMs. Este projeto representa um avanço significativo na maneira como as lojas virtuais podem se comunicar e entender as necessidades dos seus clientes, criando um ambiente de vendas mais intuitivo e engajador. A integração dessas tecnologias avançadas, especialmente desenvolvidas para a plataforma Deco, evidencia como a aplicação prática de IA pode revolucionar o setor de varejo online.

A aplicação não apenas demonstrou a capacidade de melhorar a conversão das lojas, mas também se mostrou uma ferramenta valiosa para ganhar *insights* sobre o comportamento do consumidor. A implementação da primeira versão do produto, apesar de desafios e limitações, confirmou a viabilidade e o potencial do projeto em proporcionar uma experiência de compra excepcional para os usuários finais.

5.1 Projetos Futuros

A seção de projetos futuros aborda inovações e melhorias que podem ser implementadas para continuar a evoluir com o assistente de compras. Esses passos futuros têm como objetivo expandir as capacidades do sistema, introduzir novas funcionalidades e aprimorar a experiência do usuário se alavancando dos avanços tecnológicos.

5.1.1. Recuperação de Geração Aumentada (*Retrieval Augmented Generation* - RAG)

Integrar recursos de RAG para enriquecer as respostas do assistente com informações recuperadas de um banco de dados mais amplo, proporcionando respostas ainda mais precisas e detalhadas. Isso transformaria a atual busca contextual em uma busca semântica e proporcionaria o avanço em termos de subjetividade.

5.1.2. *Caching*

Implementar estratégias de cache para melhorar o tempo de resposta e a eficiência do sistema, reduzindo a carga nas APIs e acelerando a entrega de conteúdo recorrente.

5.1.3. *Streaming*

Explorar capacidades de *streaming* para melhorar a entrega das mensagens de texto da Open AI, o que poderia melhorar a interação do usuário e tornar o *chat* mais rápido e eficiente.

5.1.4. Modelo próprio

Desenvolver um modelo de LLM próprio, adaptado especificamente às necessidades e ao contexto do *e-commerce*, otimizando ainda mais as recomendações e assistência fornecidas aos clientes.

5.1.5. *Reranking*

Aplicar técnicas de *reranking* para aprimorar a ordenação dos resultados fornecidos pelo assistente, assegurando que as melhores e mais relevantes recomendações sejam apresentadas primeiro.

5.1.6. FAQ

Construir e integrar um sistema avançado de FAQ que permita ao assistente responder a perguntas frequentes de forma mais eficiente e automatizada.

5.1.7. Redirecionamento para suporte

Estabelecer um sistema de encaminhamento para suporte humano quando o assistente não conseguir resolver uma consulta, garantindo que o usuário sempre tenha acesso à ajuda necessária.

5.1.8. Criação de perfil do usuário a partir de conversas do sistema

O armazenamento de informações do usuário e a criação de um padrão de comportamento tornará mais fácil para o sistema recomendar produtos no futuro. Isso pode ser feito através de detalhes nas perguntas/solicitações, como: preferência de cores, esportes praticados, tipo de imóvel que está reformando, etc (dependendo do segmento da loja).

5.1.9. Aba de histórico de conversas

Atualmente, o projeto não salva o histórico de conversas. Manter o *thread id* e uma aba com as conversas passadas facilitaria para o usuário revisitar conversas e continuar de onde

parou, sem a necessidade de começar do zero a criar um contexto em direção similar ao que já tinha feito.

5.1.10 Adequação aos demais requisitos não funcionais

O projeto do Deco AI Sales Assistante é um MVP que foi desenvolvido de forma célere e, portanto, ainda necessita melhorias em alguns fatores. Dentre eles, é importante pontuar alguns requisitos não funcionais que não foram comprovados durante o trabalho. Dentre eles, o RNF01, o RNF02 e o RNF03 precisam de mais estudo e de melhorias dependentes inclusive da OpenAI. Primeiramente, o RNF01 (Desempenho), não foi apontado como um problema durante os testes, mas também não foi testado de forma assídua e, em alguns momentos, a API tinha um tempo maior que 5 segundos, o que está longe do desejável, mas que não depende da infraestrutura da deco. Ademais, por não terem sidos realizados testes contínuos, também não foi possível fazer uma mensuração mais precisa do RNF02 (Disponibilidade). Por fim, pelo mesmo motivo, o RNF03 (Escalabilidade), também necessita de testes de carga e de mais tempo em produção para averiguação, o que não foi possível de ser conduzido no tempo restrito disponível para implementação desse trabalho.

5.1.11. *Fine-tuning*

Com o uso de *logs* para salvar as diversas conversas e interações do assistente, seria possível criar uma base de dados com bons exemplos para servir como base para fazer o fine-tuning de um modelo no futuro. Essa funcionalidade, inclusive, está disponível[21] na OpenAI por preços acessíveis e promessas de ganho de desempenho.

5.1.12 Instalação simples

É necessário diminuir a fricção no processo de instalação do Sales Assistant, para isso, é preciso melhorar a experiência atual e possibilitar o funcionamento do assistente com menos configurações obrigatórias.

O desenvolvimento futuro deve continuar a focar na usabilidade, acessibilidade e na personalização da experiência de compra, alinhando tecnologia de ponta com as expectativas

[21] OpenAI Fine-tuning. Disponível em: https://platform.openai.com/docs/guides/fine-tuning. Acesso em: 30 Abr. 2024.

em constante evolução dos consumidores. O compromisso com a inovação contínua e a melhoria da tecnologia subjacente garantirá que o Deco AI Sales Assistant permaneça na vanguarda da transformação digital no e-commerce.

Referências

[1] Claude Documentation. Disponível em: https://docs.anthropic.com/claude/docs/intro-to-claude. Acesso em: 24 Abr. 2024.

[2] Cruise Technology. Disponível em: https://www.getcruise.com/technology/. Acesso em 7 Abr. 2024.

[3] Deno Documentation. Disponível em: https://docs.deno.com/. Acesso em: 7 Abr. 2024.

[4] Fresh Framework. Disponível em: https://fresh.deno.dev/. Acesso em: 10 Abr. 2024.

[5] LLaMa Documentation. Disponível em: https://llama.meta.com/docs/get-started/. Acesso em: 24 Abr. 2024.

[6] OpenAI. **GPT-4 Technical Report**. Disponível em: https://cdn.openai.com/papers/gpt-4.pdf. Acesso em: 7 Abr. 2024.

[7] Tailwind CSS. Disponível em: https://tailwindcss.com/. Acesso em: 10 Abr. 2024.

[8] TypeScript Documentation. Disponível em: https://www.typescriptlang.org/docs/ . Acesso em: 7 Abr. 2024.

[9] VALENTE, Marco Tulio. "Cap.3: Requisitos - Engenharia de Software Moderna". Disponível em: engsoftmoderna.info/cap3.html. Acesso em: 23 Abr. 2024.

[10] VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N.; KAI, Łukasz; POLOSUKHIN, Illia. **Attention Is All You Need**. 2017. Disponível em: https://arxiv.org/abs/1706.03762. Acesso em: 7 Abr. 2024.

[11] Waymo one San Francisco. Disponível em: https://waymo.com/waymo-one-san-francisco/. Acesso em: 7 Abr. 2024.