PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA E DE ARTES
GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

DESENVOLVIMENTO DE UM *CHATBOT* BASEADO EM INTELIGÊNCIA ARTIFICIAL

MARCO AURELIO AYRES ROCHA DE OLIVEIRA

GOIÂNIA
2025

MARCO AURELIO AYRES ROCHA DE OLIVEIRA

DESENVOLVIMENTO DE UM *CHATBOT* BASEADO EM INTELIGÊNCIA ARTIFICIAL

Trabalho de Conclusão de Curso apresentado à Escola Politécnica e de Artes, da Pontifícia Universidade de Goiás, como parte dos requisitos para obtenção do título de Bacharel em Engenharia da Computação.
Orientador (a):
Prof.ª. Dra. Solange da Silva.
Banca examinadora:
Prof. Me. Gustavo Siqueira Vinhal,
Prof.ª. Dra. Juliana Paula Felix.

GOIÂNIA
2025

MARCO AURÉLIO AYRES ROCHA DE OLIVEIRA

DESENVOLVIMENTO DE UM *CHATBOT* BASEADO EM INTELIGÊNCIA ARTIFICIAL

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Engenharia de Computação, em: __/___/___.

_____

Orientadora: Prof.ª. Dra. Solange da Silva.

_____

Prof. Me. Gustavo Siqueira Vinhal

_____

Profa. Dra. Juliana Paula Felix

GOIÂNIA
2025

DEDICATÓRIA

Dedico este trabalho, em primeiro lugar, a Deus, por me conceder sabedoria, força e resiliência nos momentos em que a caminhada parecia impossível. Sua presença silenciosa foi meu refúgio e meu guia em todas as etapas deste percurso.

À minha mãe, Dalba, dedico com todo o meu coração. Você foi e continua sendo meu alicerce, meu exemplo de coragem e amor incondicional. Cada conquista minha é, antes de tudo, um reflexo do seu esforço, da sua fé em mim e da sua luta diária.

Ao meu pai, Pereira, à minha irmã Marcela e ao meu cunhado Júnio, agradeço pelo apoio constante, pelas palavras de incentivo e pelo carinho inabalável que sempre me acompanharam, mesmo nos dias em que não pude estar tão presente.

Dedico também à memória da minha avó Dalva e do meu padrinho José Pereira, que partiram durante a pandemia, mas permanecem vivos em cada lembrança, conselho e gesto de afeto que deixaram. Este trabalho é, em parte, herança do que aprendi com eles: a importância da honestidade, da dedicação e da esperança.

Este TCC é dedicado a todos vocês, que me impulsionaram mesmo quando eu duvidei de mim. Que ele represente, de forma simples, mas profunda, o meu amor e a minha gratidão eterna.

AGRADECIMENTOS

Agradeço primeiramente à minha família, que foi o alicerce emocional e prático de toda essa jornada. Sem o apoio, a paciência e o incentivo constante de vocês, este trabalho jamais teria se concretizado.

À professora Dra. Solange da Silva, minha orientadora, expresso minha sincera gratidão. Sua confiança no meu potencial, seus conselhos técnicos e humanos e sua dedicação ímpar foram determinantes para a realização deste trabalho. A sua orientação foi além do acadêmico foi um verdadeiro exemplo de compromisso com o ensino e com os alunos.

Aos colegas e amigos que estiveram comigo nos momentos de dificuldade e de comemoração, obrigado pela escuta, pelas trocas de ideias e pelas risadas que tornaram essa etapa mais leve e significativa. Em especial àqueles que participaram de forma direta ou indireta do desenvolvimento do *chatbot* e das discussões sobre IA nosso projeto é também um pouco de cada um.

Aos professores da Escola Politécnica e de Artes da PUC Goiás, que contribuíram ao longo da minha formação, deixo meu reconhecimento por todo o conhecimento transmitido e pelo exemplo profissional.

Por fim, agradeço àqueles que não mencionei nominalmente, mas que fizeram parte dessa jornada com gestos de apoio, palavras de incentivo ou simplesmente acreditando no meu caminho. Cada um de vocês deixou uma marca neste trabalho.

RESUMO

Este trabalho teve o objetivo geral de desenvolver um *chatbot* inteligente com integração ao *WhatsApp*, utilizando as *APIs* da Meta e o modelo *GPT*, para otimizar o atendimento ao cliente em corretoras de seguros. Quanto aos aspectos metodológicos a natureza dessa pesquisa é um resumo de assunto. Quanto aos objetivos é uma pesquisa exploratória e em relação aos procedimentos técnicos, é uma pesquisa bibliográfica e experimental. Foi realizada uma revisão bibliográfica com autores que abordam a temática inteligência artificial, Processamento de Linguagem Natural e desenvolvimento de chatbots, e foi desenvolvido um protótipo funcional utilizando o modelo *GPT* e as *APIs* da Meta. Os resultados obtidos permitiram concluir que o *chatbot* apresentou capacidade de automatizar atendimentos, reduzir o tempo de resposta e manter interações em linguagem natural, contribuindo para a otimização do atendimento em corretoras de seguros. Notou-se que a aplicação de modelos baseados na arquitetura *Transformer*, mesmo em um contexto prático de desenvolvimento individual, proporcionou um direcionamento para a integração das tecnologias, organização das funcionalidades e obtenção de resultados satisfatórios no atendimento automatizado.

Palavras-chave: *Chatbot*. Inteligência Artificial. Processamento de Linguagem Natural. *WhatsApp*. Corretoras de Seguros.

ABSTRACT

This study had the general objective of developing an intelligent chatbot integrated with WhatsApp, using the Meta APIs and the GPT model, in order to optimize customer service in insurance brokerages. Regarding the methodological aspects, the nature of this research is classified as a subject review. With respect to its objectives, it is an exploratory research, and in relation to the technical procedures, it is a bibliographic and experimental research. A bibliographic review was carried out with authors addressing the topics of artificial intelligence, Natural Language Processing, and chatbot development, and a functional prototype was developed using the GPT model and the Meta APIs. The results obtained made it possible to conclude that the chatbot demonstrated the ability to automate customer service, reduce response time, and maintain interactions in natural language, contributing to the optimization of customer service in insurance brokerages. It was noted that the application of models based on the Transformer architecture, even in a practical context of individual development, provided guidance for the integration of technologies, organization of functionalities, and achievement of satisfactory results in automated customer service.

Keywords: Chatbot. Artificial Intelligence. Natural Language Processing. WhatsApp. Insurance Brokerages.

LISTA DE ILUSTRAÇÕES

LISTA DE SIGLAS

API	*Application Programming Interface, Interface de Programação de Aplicações*
BaaS	*Backend as a Service, Backend como Serviço*
CPF	*Cadastro de Pessoa Física*
CPU	*Central Processing Unit, Unidade Central de Processamento*
GPT	*Generative Pre-trained Transformer, Transformador Pré-Treinado Generativo*
GPT-3.5	*Generative Pre-trained Transformer 3.5*
HTTP	*HyperText Transfer Protocol, Protocolo de Transferência de Hipertexto*
HTTPS	*HyperText Transfer Protocol Secure, Protocolo Seguro de Transferência de Hipertexto*
IA	*Inteligência Artificial*
IDE	*Integrated Development Environment, Ambiente Integrado de Desenvolvimento*
JSON	*JavaScript Object Notation, Notação de Objetos JavaScript*
LGPD	*Lei Geral de Proteção de Dados*
LLM	*Large Language Model, Modelo de Linguagem de Grande Escala*
LSTM	*Long Short-Term Memory, Memória de Longo Curto Prazo*
ML	*Machine Learning, Aprendizado de Máquina*
NLP	*Natural Language Processing, Processamento de Linguagem Natural*
Node.js	*Node JavaScript Runtime Environment, Ambiente de Execução JavaScript*
OpenAI	*OpenAI Research Organization, Organização de Pesquisa em Inteligência Artificial*
PLN	*Processamento de Linguagem Natural*
RAM	*Random Access Memory, Memória de Acesso Aleatório*
REST	*Representational State Transfer, Transferência Representacional de Estado*
RNN	*Recurrent Neural Network, Rede Neural Recorrente*
SDK	*Software Development Kit, Kit de Desenvolvimento de Software*
URL	*Uniform Resource Locator, Localizador Uniforme de Recursos*
UX	*User Experience, Experiência do Usuário*
VS Code	*Visual Studio Code, Ambiente de Desenvolvimento*
WABA	*WhatsApp Business API, API do WhatsApp Business*

SUMÁRIO

1 INTRODUÇÃO

A inteligência artificial (IA) começou a ganhar forma ainda na década de 1950, quando pesquisadores passaram a estudar como máquinas poderiam realizar tarefas associadas ao raciocínio humano. Um dos marcos desse período foi a Dartmouth College Conference, realizada em 1956, em que nomes como John McCarthy, Marvin Minsky, Alan Newell e Herbert Simon apresentaram ideias que deram início às primeiras linhas de pesquisa na área.

Desde então, os avanços tecnológicos ampliaram o alcance da IA e permitiram que ela fosse aplicada em diversos setores, entre eles o financeiro e o segurador. O interesse crescente por soluções que são capazes de agilizar atendimentos, reduzir etapas manuais e assim melhorar a experiência do usuário tem impulsionado a adoção de sistemas inteligentes nesses mercados (Silva; Chan; Decoster, 2025).

Nesse cenário, os *chatbots* se destacam por atuar como interfaces conversacionais capazes de interagir com usuários em linguagem natural. Eles automatizam tarefas simples, ajudam a organizar o fluxo de atendimento e oferecem suporte imediato, características essenciais em ambientes de serviços digitais (Ferreira et al., 2021).

Segundo Russell e Norvig (2021, p. 1),

> “A IA é um dos campos mais recentes em ciências e engenharia. [...] Atualmente, a IA abrange uma enorme variedade de subcampos, do geral (aprendizagem e percepção) até tarefas específicas, como jogos de xadrez, demonstração de teoremas matemáticos, criação de poesia, direção de um carro em estrada movimentada e diagnóstico de doenças.”

Conforme Jurafsky e Martin, o Processamento de Linguagem Natural (PLN) utiliza técnicas como análise de sentimentos, classificação de textos e extração de informações para apoiar a interpretação automática de linguagem humana, permitindo que o sistema produza respostas em linguagem natural de acordo com o contexto da interação. (Jurafsky; Martin, 2025, p. 176).

O PLN utiliza técnicas como análise de sentimentos, classificação textual, extração de informações e geração de linguagem natural, permitindo que máquinas participem de interações cada vez mais sofisticadas com seres humanos (Silva et al., 2025).

Os primeiros sistemas conversacionais como o ELIZA, criado por Weizenbaum em 1966, e PARRY, desenvolvido por Colby em 1972 utilizavam padrões fixos e regras pré-programadas, o que limitava sua flexibilidade. Avanços em redes neurais e em técnicas de *deep learning*, especialmente após a proposta da arquitetura *Transformer* em 2017, ampliaram a capacidade dos modelos de linguagem em considerar o contexto e produzir respostas mais consistentes ao longo da interação. Como afirma Dale (2020, p. 359),

> "Os avanços em aprendizado profundo transformaram os sistemas conversacionais, tornando possível a criação de modelos capazes de compreender melhor o contexto e produzir respostas mais naturais, ampliando significativamente a eficiência e a qualidade das interações automatizadas."

No setor segurador, a necessidade de respostas rápidas, disponibilidade contínua e padronização da comunicação tornou-se ainda mais evidente. Clientes frequentemente solicitam segunda via de documentos, consulta de status de sinistro ou esclarecimento de dúvidas simples, o que sobrecarrega os colaboradores e reduzem a eficiência operacional. Relatórios do setor apontam que o uso de *chatbots* tem reduzido o tempo de atendimento e ampliado a disponibilidade do serviço, o que se reflete em melhor avaliação por parte dos usuários (Oracle, 2023).

Justifica-se o estudo deste tema, pois ocorreu uma alta demanda por soluções que otimizem o atendimento ao cliente, especialmente no setor de seguros, impulsionada ainda mais pela digitalização dos serviços e pela necessidade de respostas rápidas aos clientes.

Diante desse contexto, este Trabalho de Conclusão de Curso propõe investigar a seguinte questão: **um *chatbot* com IA integrada, baseado no modelo *GPT* e conectado ao *WhatsApp* pode otimizar o atendimento ao cliente em corretoras de seguros?**

Este trabalho tem o objetivo geral de desenvolver um *chatbot* inteligente com integração ao *WhatsApp*, utilizando as *APIs* da Meta e o modelo *GPT*, para otimizar o atendimento ao cliente em corretoras de seguros.

E os objetivos específicos são:

- Aprofundar a compreensão sobre Inteligência Artificial aplicada ao desenvolvimento de sistemas conversacionais, com foco em Processamento de Linguagem Natural e na integração de *APIs* de mensageria.
- Identificar as principais necessidades e dificuldades do atendimento em corretoras de seguros, reunindo os requisitos essenciais para o desenvolvimento do sistema.
- Desenvolver, testar e validar um *chatbot* baseado no modelo *GPT* integrado ao *WhatsApp*, analisando seu desempenho e sua adequação ao atendimento automatizado.

Espera-se que os resultados deste trabalho possam contribuir:

- Reduzir o tempo de espera no atendimento ao cliente, tornando o processo mais ágil e acessível.
- Aprimorar a experiência do usuário, oferecendo respostas automatizadas mais naturais, precisas e alinhadas ao contexto da interação.
- Disponibilizar um *chatbot* funcional que automatize tarefas frequentes do atendimento e contribua para aumentar a eficiência das equipes das corretoras.

Em relação aos aspectos metodológicos esta pesquisa quanto a natureza é um resumo de assunto e quanto aos objetivos é uma pesquisa exploratória, e em relação aos procedimentos técnicos, adota-se uma abordagem bibliográfica e experimental.

Esta monografia está estruturada em mais 6 capítulos. O Capítulo 2 traz o referencial teórico, com conceitos e definições e trabalhos relacionados com o tema. O Capítulo 3 descreve os procedimentos metodológicos adotados na pesquisa. O Capítulo 4 apresenta o desenvolvimento do projeto, a definição dos requisitos, a arquitetura e os principais componentes implementados. O Capítulo 5 reúne os testes realizados, os resultados obtidos e a avaliação do desempenho do chatbot. O Capítulo 6 analisa os resultados do estudo e dos conceitos apresentados no referencial teórico. Por fim, o Capítulo 7 apresenta as conclusões do trabalho e as sugestões para trabalhos futuros.

2 REFERENCIAL TEÓRICO

Este capítulo aborda a base teórica que deu suporte ao desenvolvimento do *chatbot*. Também os principais conceitos sobre *chatbots*, Inteligência Artificial e Processamento de Linguagem Natural e como funciona a arquitetura *Transformer* e os modelos textuais. Também apresenta aspectos relevantes do setor de seguros, além de alguns estudos que ajudam a situar esta pesquisa dentro do contexto atual.

2.1 Conceito e Funcionamento de *Chatbots* Inteligentes

O termo chatbot vem da junção de duas palavras em inglês, *chat* (conversa) e *bot* (robô). Além desse termo, os chatbots também podem ser vistos com nomes diferentes, como assistentes virtuais ou agentes virtuais.

McTear (2020) descreve esses agentes como programas capazes de interpretar comandos, executar tarefas e produzir respostas em linguagem natural, apoiando-se em técnicas de *PLN* e aprendizado de máquina.

Modelos tradicionais baseados apenas em regras tendem a apresentar limitações, já que dependem de padrões fixos e não lidam bem com ambiguidades ou variações naturais da linguagem humana.

Com a evolução das redes neurais e dos modelos estatísticos, surgiram os chamados *chatbots* inteligentes, que conseguem identificar intenções, interpretar o conteúdo das mensagens e adaptar o diálogo ao contexto inserido pelo usuário (Almalki; Aziz; Wang, 2021).

Jurafsky e Martin (2025, p. 176–179) destacam que modelos contemporâneos de *PLN* são pré-treinados em grandes coleções textuais, tornando possível aplicar a mesma arquitetura a diferentes tarefas sem necessidade de ajustes extensivos. Esse princípio fundamenta o desempenho de sistemas conversacionais baseados em modelos de linguagem de grande escala.

A estrutura de um *chatbot* inteligente pode ser dividida em quatro componentes principais: interface de comunicação, mecanismos de compreensão da linguagem PLN, motor de decisão e camada de integração com

sistemas externos (Colace et al., 2022). O bom funcionamento do sistema exige que esses elementos operem de forma integrada, condição especialmente importante em setores que demandam agilidade e precisão, como o mercado segurador.

No presente trabalho, o *chatbot* foi iimplementado como um sistema híbrido que combina fluxos estruturados de atendimento com respostas geradas por um modelo de linguagem. O *Backend* é responsável pelas regras de negócio, pelas validações e pela condução dos fluxos, enquanto o modelo *GPT* interpreta o conteúdo textual e produz respostas coerentes com o contexto da conversa. A solução integra também o *WhatsApp* como canal de entrada, a *API* da Meta para transporte das mensagens e o *Firebase* como camada de registro e persistência das interações.

A solução combina diferentes tecnologias, como o *WhatsApp* para envio das mensagens, a *API* da Meta para transporte dos dados, o *Backend* em *Node.js* para tratamento das requisições, o modelo *GPT* para geração das respostas e o *Firebase* para registro das interações.

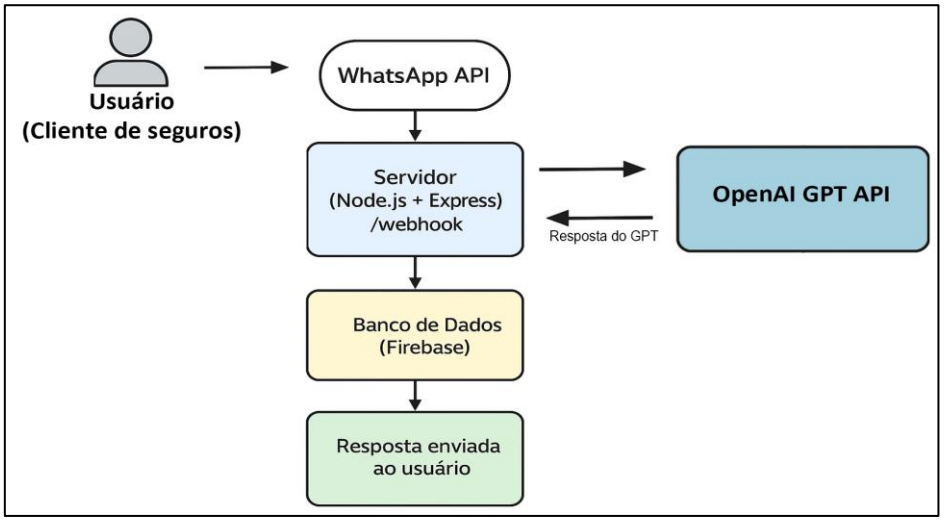
O ciclo de atendimento segue um conjunto de etapas definidas: a mensagem enviada pelo usuário é recebida pelo *webhook.* O *webhook* é basicamente o ponto de entrada do sistema. É ele que recebe, em tempo real, aas mensagens enviadas pelos usuários.

Assim que a Meta detecta uma nova mensagem enviada para o número da corretora, ele encaminha essa informação direto para o endereço que foi configurado no servidor. Dessa forma o webhook atua como o ponto inicial de recepção das mensagens, o sistema pega a mensagem bruta, identifica quem enviou e repassa isso para o restante do fluxo de atendimento continuar.

A mensagem és processada pelo *Backend*, encaminhada ao modelo *GPT* quando necessário e, após a geração da resposta, o conteúdo é formatado, enviado ao usuário e registrado no banco de dados. Esse fluxo permite manter a continuidade do diálogo, organizar as informações trocadas e garantir consistência nas respostas enviadas ao usuário.

A Figura 1 representa o fluxo completo de comunicação entre os componentes da solução.

Figura 1 - Fluxo de Comunicação



Fonte: Elaborado pelo autor (2025).

2.2 A Arquitetura *Transformer*

Como a arquitetura *Transformer* teve impacto direto na forma como o Processamento de Linguagem Natural (PLN) é realizado?

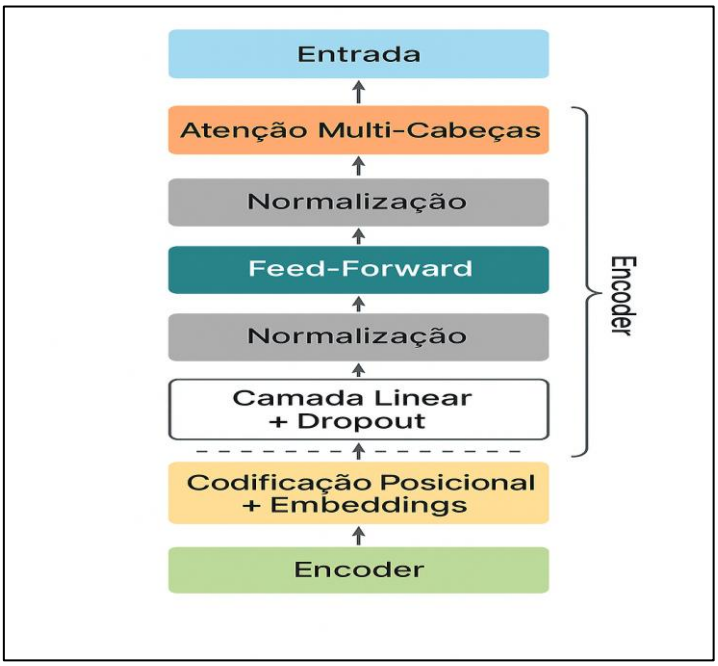
Proposta por Vaswani et al. (2017), essa abordagem rompeu com as estruturas sequenciais tradicionais ao substituir completamente modelos recorrentes, como as *Recurrent Neural Networks* (RNNs) e as *Long Short-Term Memory* (LSTM), por um mecanismo baseado exclusivamente em *self-attention*. Esse mecanismo permite que o modelo avalie, simultaneamente, a relevância de cada palavra da sequência em relação às demais, atribuindo pesos dinâmicos conforme o contexto global, em vez de depender da ordem estrita da entrada, como ocorria nos modelos recorrentes (Vaswani et al., 2017).

Conforme explicam Jurafsky e Martin (2025, p. 179–180), o mecanismo de *self-attention* possibilita que o modelo integre informações provenientes de diferentes posições da frase de maneira paralela, capturando dependências de longo alcance com maior eficiência. Por essa razão, a arquitetura *Transformer* tem sido aplicada em tarefas como tradução automática, sumarização, classificação de textos e geração de conteúdo, que dependem de boa interpretação do contexto.

A Figura 2 representa o núcleo do *encoder* que é composto pelo mecanismo de atenção multi-cabeças (*multi-head attention*), responsável por identificar, em paralelo, os diferentes tipos de relações semânticas presentes no

texto. Após essa etapa, o modelo aplica uma normalização seguida de uma rede *feed-forward*, que refina as representações intermediárias. Uma segunda normalização conclui o bloco, que é replicado ao longo de várias camadas empilhadas para aprimorar progressivamente a capacidade de interpretação do modelo.

Figura 2 - Estrutura interna de um *encoder*



Fonte: Adaptado de Vaswani et al. (2017).

A organização em blocos sucessivos contribui para que modelos baseados em *Transformer* alcancem boa escalabilidade e desempenho. Como observam Vaswani et al. (2017), o paralelismo inerente ao mecanismo de atenção reduz o tempo de treinamento, ao mesmo tempo em que amplia a capacidade de capturar estruturas linguísticas complexas. Por esse conjunto de características, a arquitetura *Transformer* passou a ser amplamente adotada em modelos de linguagem atuais, entre eles o utilizado neste trabalho.

Neste trabalho, compreender a arquitetura *Transformer* é importante, já que ela serve de base para o modelo *GPT* utilizado no *chatbot*. A capacidade de analisar dependências linguísticas em larga escala e gerar representações contextuais precisas permite que o sistema produza respostas coerentes, adaptadas ao fluxo do diálogo e alinhadas às necessidades específicas do atendimento no setor de seguros.

2.3 O Modelo *GPT-3.5* e Suas Aplicações

O modelo *Generative Pre-trained Transformer* 3.5 (*GPT-3.5*) faz parte da família de modelos de linguagem em larga escala desenvolvidos pela *OpenAI* e amplia as possibilidades de compreensão e geração de texto em diferentes contextos. Treinando em extensos conjuntos de dados por meio de aprendizagem autorregressiva, o modelo prevê *tokens* sucessivos e produz respostas coesas e alinhadas ao estilo comunicativo desejado. Essa capacidade possibilita seu uso em tarefas variadas, entre elas redação de textos, síntese de informações, classificação de mensagens e suporte a interações conversacionais.

De acordo com Jurafsky e Martin (2025, p. 178–179), modelos baseados na arquitetura *Transformer* utilizam mecanismos de *self-attention* para capturar dependências linguísticas complexas, o que os torna adequados para múltiplas tarefas de *Processamento de Linguagem Natural* (*PLN*) sem a necessidade de ajustes extensivos. Essa capacidade explica o desempenho de modelos como o *GPT-3.5* em ambientes práticos que exigem flexibilidade e adaptação a diferentes estilos de interação.

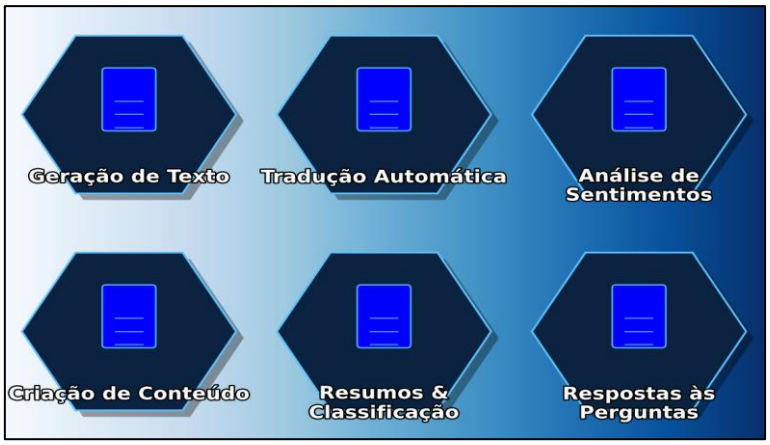
Radford, Narasimhan e Sutskever (2020, p. 03) destacam que modelos de larga escala generalizam com eficiência quando expostos a instruções mínimas, realizando tarefas complexas a partir do próprio aprendizado adquirido durante o pré-treinamento. Esse comportamento fundamenta a utilização de modelos generativos em aplicações corporativas, nas quais a interpretação precisa do contexto e a produção de respostas adequadas são fatores essenciais para a experiência do usuário.

Em ambientes corporativos, o *GPT-3.5* tem sido utilizado em tarefas como elaboração de textos, análise de mensagens, automação de conversas e tradução de conteúdos (OpenAI, 2024). Sua capacidade de adaptação ao contexto o torna particularmente adequado para sistemas de atendimento que precisam lidar com demandas variadas e linguagem informal, como ocorre no setor de seguros.

A Figura 3 representa algumas das principais aplicações de *PLN* incorporadas por modelos baseados em *Transformer*, incluindo geração de texto, análise de sentimentos, classificação de informações, sumarização

automática, resposta a perguntas e tradução. Essas funcionalidades constituem o núcleo das operações do *chatbot* desenvolvido neste trabalho.

Figura 3 - Aplicações do PLN



Fonte: Elaborado pelo autor (2025).

No contexto específico desta pesquisa, o *GPT-3.5* desempenha o papel de motor conversacional do sistema. As mensagens recebidas pelo *Backend* são organizadas em um *prompt* estruturado, que inclui o conteúdo do usuário e informações contextuais relevantes, permitindo que o modelo produza respostas adequadas e coerentes ao fluxo de atendimento. Essa integração possibilita que o *chatbot* possa interpretar variações linguísticas, mantenha continuidade entre interações e ofereça respostas alinhadas às necessidades dos usuários.

Ao combinar ferramentas de atendimento com a flexibilidade do modelo *GPT-3.5*, o sistema se torna essencial para lidar com situações reais em corretoras de seguros, nas quais os usuários podem formular perguntas de formas diversas e utilizar diferentes registros de linguagem.

2.4 Atendimento ao Cliente no Setor de Seguros

No setor de seguros, a oferta de serviços em meios digitais aumentou a procura por soluções que proporcionem atendimento rápido, funcionamento contínuo e comunicação uniforme com o cliente.

Nesse cenário, tecnologias baseadas em Inteligência Artificial vêm desempenhando papel cada vez mais relevante, sobretudo em atividades que

envolvem grande volume de solicitações repetitivas e necessidade de respostas rápidas. Segundo Silva et al. (2025, p. 3), a IA tem se consolidado como uma ferramenta estratégica para a modernização do mercado segurador, com aplicações que incluem precificação, análise de sinistros e atendimento ao cliente.

Nesse setor, *chatbots* têm sido adotados para apoiar tarefas como triagem de solicitações, fornecimento de informações sobre apólices e orientação inicial em casos de sinistro. Ao automatizar essas etapas, as empresas conseguem reduzir o tempo médio de resposta, aumentar a disponibilidade do serviço e diminuir a sobrecarga operacional das equipes humanas. Silva et al. (2025, p. 7) observam que tais soluções contribuem para elevar a satisfação do cliente, ao mesmo tempo em que otimizam processos internos e reduzem custos operacionais.

Um estudo conduzido por Lopes da Silva et al. (2025) analisou relatórios financeiros de trinta seguradoras brasileiras e identificou que empresas que divulgam o uso de tecnologias de IA relatam melhorias consistentes na experiência do cliente e na eficiência operacional. De acordo com os autores, mesmo que a adoção dessas tecnologias ainda seja limitada em parte do setor, há uma tendência crescente de investimentos voltados à automação de processos e ao desenvolvimento de soluções digitais personalizadas.

Nesse contexto, o uso de *chatbots* inteligentes apresenta potencial elevado para aprimorar o relacionamento entre clientes e seguradoras, especialmente quando integrado a plataformas amplamente utilizadas, como o *WhatsApp*. A possibilidade de oferecer atendimento contínuo, linguagem clara e respostas personalizadas contribui para suprir a expectativa dos usuários por interações rápidas e precisas. Além disso, a automação oferece vantagens adicionais, como padronização das informações repassadas, registro sistemático das interações e maior capacidade de escala em períodos de alta demanda.

Ao incorporar modelos de linguagem avançados, como o *GPT-3.5*, os *chatbots* utilizados no setor de seguros podem superar limitações típicas de sistemas baseados em regras, interpretando variações de escrita, dúvidas complexas e mensagens formuladas de modo informal.

2.5 Usabilidade e Experiência do Usuário (*UX*)

A usabilidade é um dos elementos centrais para o sucesso de sistemas interativos e influencia diretamente a aceitação de tecnologias baseadas em *chatbots*. Segundo Nielsen e Mack (1994, p. 25 a 29), a usabilidade envolve cinco dimensões fundamentais: eficácia, eficiência, satisfação, facilidade de aprendizado e redução de erros. Esses princípios orientam o desenvolvimento de interfaces capazes de atender às necessidades reais dos usuários e de promover interações claras e consistentes.

No contexto de sistemas conversacionais, a experiência do usuário (*User Experience* ou *UX*) depende de fatores como clareza das respostas, tempo de processamento, manutenção do contexto entre mensagens e adequação do tom de comunicação. Shum, He e Li (2018, p. 14 a 18) destacam que um *chatbot* eficiente deve equilibrar capacidades cognitivas (*IQ*), relacionadas à precisão e à relevância das respostas, com capacidades emocionais (*EQ*), que envolvem empatia, adaptação da linguagem e sensibilidade ao estado do usuário. Esse equilíbrio é fundamental para que o sistema seja percebido como útil e confiável.

Com base nesses autores, algumas práticas de *UX* foram consideradas no desenvolvimento do *chatbot* deste trabalho. Entre eles podem ser destacadas:

- Personalização das interações, ajustando respostas com base no histórico e no perfil do usuário (Shum; He; Li, 2018, p. 16).
- Adequação do tom de linguagem, garantindo coerência com o contexto e com o tipo de solicitação apresentada (Nielsen, 1994, p. 27).
- Consistência e clareza das respostas, evitando ambiguidades que possam prejudicar a compreensão (Nielsen, 1994, p. 28).
- Manutenção do contexto conversacional, permitindo continuidade lógica ao longo de diferentes etapas da interação (Shum; He; Li, 2018, p. 15).

No ambiente de atendimento de corretoras de seguros, esses aspectos tornam-se ainda mais relevantes. O usuário frequentemente busca informações específicas sobre apólices, sinistros ou documentos oficiais, o que exige precisão e clareza na comunicação. Por esse motivo, a usabilidade e a *UX* desempenham papel essencial no suporte eficiente ao cliente, na redução do esforço cognitivo e na construção de uma experiência positiva durante todo o atendimento automatizado.

2.6 Ética e Limitações da Inteligência Artificial

O uso crescente de sistemas de Inteligência Artificial em diferentes áreas amplia as possibilidades de automatização, mas traz também questões éticas que não podem ser ignoradas. A utilização de sistemas baseados em IA envolve riscos relacionados a vieses algorítmicos, falta de transparência, limitações de interpretação e potenciais impactos sobre a privacidade e o uso de dados pessoais. Esses aspectos tornam necessária a adoção de práticas que assegurem responsabilidade e confiabilidade no desenvolvimento e na aplicação dessas tecnologias.

No setor público, especialmente em áreas sensíveis como o Poder Judiciário, Locatelli (2023) observa que a implementação de soluções de IA deve ser acompanhada de mecanismos de revisão humana, validação criteriosa de dados e transparência nos processos decisórios.

A ausência desses elementos pode resultar em violações de direitos fundamentais, incluindo a proteção de dados e a imparcialidade das decisões automatizadas.

No setor privado, organizações que adotam sistemas automatizados precisam seguir princípios de ética digital, como equidade, segurança, rastreabilidade e clareza na comunicação com o usuário. As diretrizes internacionais descritas pela OECD (2019) reforçam a importância de informar ao usuário quando ele está interagindo com um sistema automatizado e de garantir a possibilidade de intervenção humana em situações de maior complexidade ou impacto. Isso inclui, por exemplo, permitir que o usuário solicite atendimento humano sempre que necessário.

Outro aspecto relevante é a limitação operacional dos modelos de IA pois embora apresentem desempenho elevado em tarefas estruturadas, essas tecnologias podem gerar respostas inconsistentes em cenários que exigem interpretação subjetiva, análise profunda ou conhecimento especializado. Ferreira et al. (2021) destacam que soluções automatizadas tendem a funcionar melhor quando aplicadas a tarefas simples e bem definidas, apresentando desempenho mais limitado quando dependem de julgamento contextual complexo.

Essas considerações são fundamentais para o desenvolvimento de *chatbots* no setor de seguros, onde as interações frequentemente envolvem dados pessoais, informações contratuais e cenários emocionalmente sensíveis, como situações de sinistro. Utilizar a adoção de práticas responsáveis e o reconhecimento das limitações da IA contribuem para garantir que o sistema forneça suporte eficiente sem comprometer direitos, segurança ou transparência durante o atendimento ao usuário.

2.7 Trabalhos Relacionados

Nos últimos anos, soluções baseadas em *chatbots* têm recebido atenção crescente na literatura, especialmente no contexto empresarial, em que a automação de tarefas repetitivas e a melhoria da experiência do usuário são tratadas como prioridades estratégicas (Rawat; Sharma; Verma, 2021, p. 223). Mesmo antes dos avanços recentes em inteligência artificial, os sistemas conversacionais já eram objeto de pesquisa desde a década de 1960.

Um dos marcos iniciais foi o ELIZA, criado por Joseph Weizenbaum em 1966, que simulava um psicoterapeuta Rogeriano por meio da reformulação das entradas do usuário. Embora rudimentar, o sistema demonstrava a possibilidade de interação homem-máquina baseada em linguagem natural.

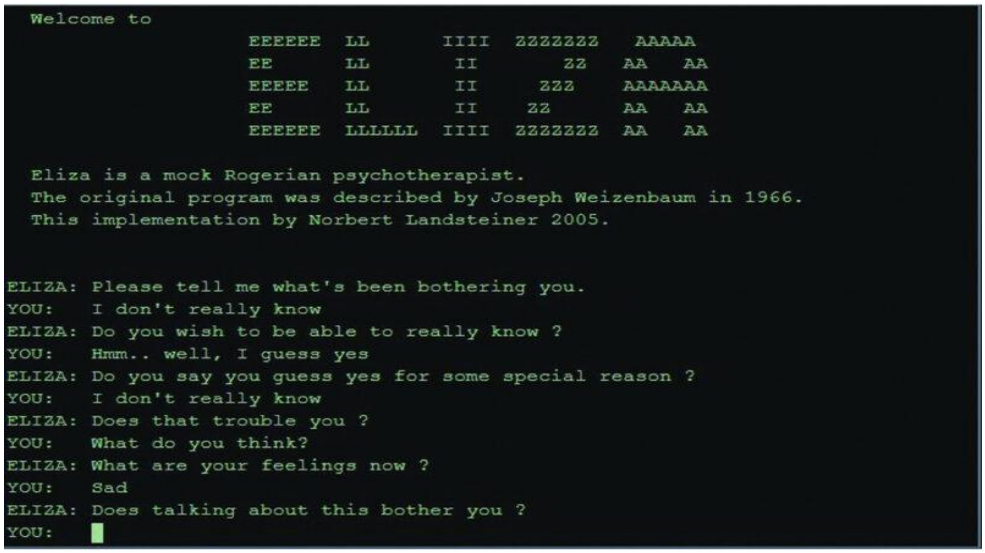
O software utilizou mecanismos como correspondências de padrões (*pattern matching*) e técnicas de substituição de caracteres. Desta forma, após receber uma sequência de caracteres do usuário como parâmetro de entrada, o software primeiramente faz uma busca por palavras-chave, que são termos previamente definidos no código fonte pelo programador e são considerados como sendo importantes para o contexto da conversa.

Se essas palavras forem encontradas no texto de entrada, elas são colocadas em uma pilha com a palavra com o maior ranking no topo desta estrutura de dados, para posterior processamento. Depois, a sentença de entrada é manipulada com o objetivo de encontrar uma regra de transformação apropriada que esteja relacionada com a palavra que possui o maior ranking.

Shum, He e Li (2018, p. 24) destacam que o ELIZA expôs tanto o potencial quanto as limitações de abordagens apoiadas apenas em regras fixas e padrões rígidos.

A Figura 4 representa uma simulação típica desse sistema, ilustrando como suas respostas eram limitadas a variações simples de perguntas previamente definidas.

Figura 4 - Simulação de Diálogo *Chatbot* ELIZA



Fonte: Reprodução da implementação de Norbert Landsteiner (2005).

Sua contribuição histórica permanece essencial por evidenciar a distância entre sistemas baseados em regras e os modelos generativos contemporâneos. Esse marco inicial abriu caminho para agentes conversacionais mais sofisticados, apoiados em modelos estatísticos, redes neurais e avanços no processamento de linguagem natural (Jurafsky; Martin, 2025, p. 90). Com a expansão do uso corporativo de assistentes virtuais, diferentes estudos passaram a investigar aplicações reais, inclusive no setor de seguros foco desta pesquisa.

Um desses estudos foi o de Ferreira et al. (2021), que analisaram a adoção de *chatbots* no atendimento pós-venda de seguros. O objetivo foi compreender a percepção dos consumidores em interações envolvendo triagem, emissão de boletos, segunda via de apólices e alterações cadastrais. A partir de 139 respostas coletadas, observou-se que a automação é bem recebida quando aplicada a tarefas simples e repetitivas, mas encontra resistência em situações que exigem maior detalhamento, como dúvidas sobre coberturas ou regulação de sinistros.

Outro estudo relevante é o de Silva, Marques e Rocha (2024), que documentaram a implementação de um *chatbot* interno em uma cooperativa de crédito. E diferentemente de soluções voltadas ao cliente final, essa proposta buscou otimizar a comunicação interdepartamental. A implantação resultou em melhorias observáveis na produtividade, na agilidade da troca de informações e na organização interna, além de impactos ambientais positivos devido à redução do uso de papel.

Chatbots não substituem equipes, mas ampliam sua atuação ao liberar colaboradores de tarefas repetitivas, permitindo que concentrem esforços em atividades mais estratégicas. Além disso, os estudos apontam que tecnologias conversacionais bem planejadas podem reduzir ruídos, padronizar fluxos para tornar processos operacionais mais claros e humanizados.

No campo específico do mercado segurador, Silva, Chan e Decoster (2025) analisaram relatórios financeiros de trinta seguradoras brasileiras para mapear o uso de tecnologias de IA. Os autores identificaram que apenas cerca de 30% das empresas declaravam utilizar IA em processos internos, incluindo personalização de produtos, automatização de sinistros e melhorias na experiência do cliente. Embora o índice seja relativamente baixo, o estudo revela uma tendência crescente de adoção tecnológica no setor.

A pesquisa indica que a adoção de soluções baseadas em *chatbots* pode aumentar a competitividade e reduzir custos operacionais, evidenciando o papel estratégico da inteligência artificial no aprimoramento dos serviços. Apesar desses avanços, a literatura ainda apresenta lacunas importantes: poucos estudos exploram soluções que integrem modelos generativos de linguagem, plataformas de mensageria amplamente utilizadas como o *WhatsApp* e bancos de dados em tempo real. E foi justamente diante dessa lacuna que este trabalho buscou contribuir, propondo uma arquitetura prática capaz de suprir essa demanda no setor de seguros.

Estudos recentes mostram que *chatbots* alimentados por modelos de linguagem de grande porte têm impactos positivos em serviços de atendimento ao cliente em diversos setores da indústria, apresentando desempenho superior a sistemas baseados em regras em termos de qualidade de respostas, fluidez na conversa e adaptação ao contexto (Meduri, 2024).

3 PROCEDIMENTOS METODOLÓGICOS

Esta pesquisa, quanto à sua natureza, é classificada como resumo de assunto, que conforme Wazlawick (2014), esse tipo de pesquisa visa à compreensão de determinado tema por meio da coleta e sistematização de conteúdos já existentes, organizando-os de maneira lógica e coerente com a proposta investigativa.

Quanto aos objetivos, caracteriza-se como uma pesquisa exploratória, pois busca proporcionar maior familiaridade com o problema estudado, de forma a permitir uma compreensão aprofundada do tema e a fundamentação de soluções possíveis. De acordo com Gil (2021, p. 27), “As pesquisas exploratórias têm como propósito proporcionar maior familiaridade com o problema, com vistas a torná-lo mais explícito ou a construir hipóteses”.

No que se refere aos procedimentos técnicos, esta pesquisa é do tipo bibliográfica e experimental.

A parte bibliográfica envolveu o levantamento de referências teóricas sobre *chatbots*, Processamento de Linguagem Natural (PLN), modelos generativos como o *GPT* e o uso dessas tecnologias no setor segurador. Foram utilizadas fontes como livros técnicos, artigos científicos, relatórios técnicos, documentação oficial de *APIs* e ferramentas, além de trabalhos acadêmicos relacionados.

As etapas da pesquisa bibliográfica, conforme Gil (2021), foram:

- a) Escolha do tema: uso de *chatbots* com IA no atendimento automatizado de seguradoras;
- b) Levantamento bibliográfico preliminar: busca por artigos, livros e documentos técnicos que abordassem o uso de tecnologias como *GPT*, *WhatsApp Business API*, *Node.js* e *Firebase*;

- c) Formulação do problema: **É possível desenvolver um *chatbot* com *GPT* integrado ao *WhatsApp* capaz de otimizar o atendimento ao cliente em corretoras de seguros?**

- d) Busca das fontes: Fontes relevantes foram consultadas em artigos científicos indexados nas bases CAPES, Scielo e Google *Academy*, além de livros especializados disponíveis em bibliotecas acadêmicas e acervo pessoal. Essas referências foram essenciais para fundamentar teoricamente o trabalho, contribuir para a formulação do problema de pesquisa e esclarecer os processos técnicos e conceituais envolvidos no desenvolvimento da solução proposta.;

- e) Leitura e fichamento: Extração dos conceitos-chave aplicáveis ao projeto;

- f) Redação: escrita do TCC.

A pesquisa experimental consiste em um tipo de investigação científica na qual o pesquisador manipula variáveis controladas para observar os efeitos produzidos em um determinado fenômeno. Seu principal objetivo é verificar relações de causa e efeito entre os elementos estudados, testando hipóteses em um ambiente controlado ou simulado, conforme descreve Gil (2008, p. 14.

A seguir, detalham-se as etapas do método experimental com base em Gil (2021):

a) Formulação do problema: **Avaliar se um *chatbot* baseado em *GPT* pode automatizar o atendimento ao cliente em corretoras de seguros com eficácia.**

b) Definição do plano experimental: consistiu na construção, teste e análise de um protótipo funcional de *chatbot* inteligente, utilizando o modelo de linguagem *GPT* da *OpenAI*. O sistema foi integrado à *API* oficial do *WhatsApp* (Meta), com *Backend* em *Node.js* e banco de dados em *Firebase*. Serão realizadas Identificação das funcionalidades mais demandadas: segunda via de apólice, status de sinistro e dúvidas frequentes; foram feitas as seleções das

tecnologias: *GPT* da *OpenAI*, *WhatsApp API*, *Node.js* e *Firebase*; Construção, execução e testes do protótipo em ambiente simulado.

c) Determinação do ambiente: Foram utilizados o Sistema operacional: *Windows 11*; o Hardware: computador com 32 GB de *Random Access Memory (RAM)* e processador *AMD Ryzen 7*; o Software: IDE *Visual Studio Code*, navegador atualizado e credenciais autorizadas para uso da *API* da Meta e da *OpenAI*.

d) Coleta de dados: Para avaliar o desempenho do sistema, foram realizadas simulação com interações controladas durante um período de 30 dias, utilizando mensagens de teste elaboradas para representar situações comuns de atendimento. Esse procedimento permitiu registrar métricas como tempo de resposta, ocorrência de falhas, clareza das respostas e mensagens não compreendidas.

e) Análise e interpretação dos dados: A análise dos resultados obtidos baseou-se em métricas como:

- Taxa de resolução automática;
- Tempo médio de resposta;
- Clareza das respostas;
- Testes bem-sucedidos com o atendimento automatizado.

f) Redação do relatório: Escrita do TCC.

4 DESENVOLVIMENTO DO PROJETO

Para efetuar o desenvolvimento do sistema, o projeto foi organizado em etapas para garantir que cada parte fosse testada separadamente antes de ser integrada ao todo. O levantamento da arquitetura foi feito com foco na comunicação entre os quatro componentes principais: o *WhatsApp*, que serve como canal de entrada do usuário; o *Backend* em *Node.js*, responsável pela lógica do atendimento; o modelo GPT, que interpreta as mensagens e gera as respostas; e o *Firebase*, que armazena o histórico e o estado das conversas.

O projeto foi dividido em três camadas: entrada, processamento e armazenamento. A camada de entrada é aa responsável por receber as mensagens enviadas pelo usuário no *WhatsApp*. É nesse ponto que o sistema capta o texto bruto e prepara as informações para o backend.

Já na camada de processamento é onde a lógica acontece de fato, nessa etapa, o backend interpreta a mensagem.

, identifica o que o usuário quer, aplica regras internas e se necessário, envia a solicitação para o modelo GPT gerar uma resposta.

Por fim, a camada de armazenamento registra tudo o que é relevante para manter o histórico das conversas e o estado do atendimento. O banco de dados guarda essas informações, permitindo rastreamento e continuidade do diálogo.

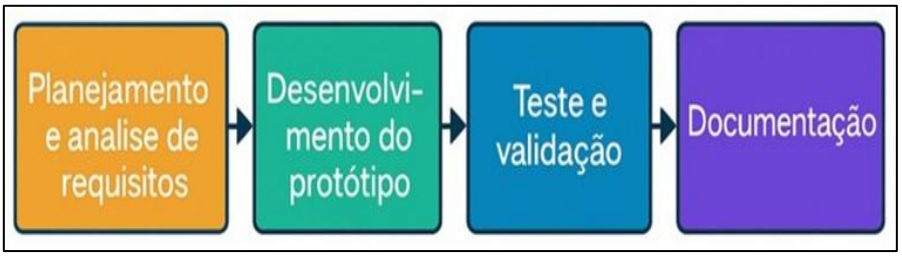
4.1 Prototipação

A prototipação do sistema foi organizada em quatro etapas principais. A primeira etapa envolveu o planejamento e a análise de requisitos, momento em que foram definidos os objetivos do projeto, aas funcionalidades essenciais e as integrações necessárias com *APIs* externas. Em seguida, foi dado início ao desenvolvimento do protótipo, no qual a estrutura do *backend*, a lógica conversacional e a conexão com o *WhatsApp* e o modelo GPT foram implementadas.

Após essa fase, o sistema passou por um período de teste e validação, garantindo que as funcionalidades planejadas funcionassem conforme esperado e assim permitindo ajustes antes da versão final. Por fim, ocorreu a etapa de documentação, ela foi responsável por registrar todas as decisões técnicas,

funcionalidades implementadas e resultados obtidos, e serão discutidas detalhadamente abaixo. A Figura 5 representa o fluxograma das etapas estruturais da prototipação.

Figura 5 - Etapas do Processo de Desenvolvimento do Protótipo



Fonte: Elaborado pelo autor (2025).

4.1.1. Planejamento do Projeto e Análise de Requisitos

A primeira etapa consistiu no planejamento do projeto, no qual foram definidos o objetivo, o escopo e as limitações iniciais do *chatbot*. Também foram escolhidas as tecnologias essenciais do sistema, como *Node.js*, a *API* da Meta, o modelo da OpenAI e o *Firebase*.

Para o próximo passo foi efetuada a análise de requisitos, que identificou as necessidades reais do usuário e os comportamentos esperados do sistema. Essa análise serviu para orientar a definição dos fluxos de atendimento e das ações que o sistema deveria executar em cada etapa, como emissão de segunda via de apólices e consulta de status de sinistros.

4.1.2 Desenvolvimento do Protótipo

A Prototipação marcou a construção da primeira versão funcional do sistema. Nessa fase foi criada a primeira versão do *chatbot*, utilizada para testar as interações básicas e validar a comunicação com o *Backend* e com as *APIs* externas. Foram implementados o esqueleto do *Backend* em *Node.js*, os primeiros fluxos de conversação e os testes iniciais com a *API* do *WhatsApp* e o modelo *GPT*.

Já na etapa de Implementação, o código passou a assumir sua estrutura definitiva. O *Backend* organizou o processamento das mensagens, recebendo os dados do *WhatsApp*, aplicando as regras de atendimento e enviando

solicitações ao modelo *GPT*. Além disso, foram incluídos os fluxos guiados, o tratamento das intenções, o armazenamento no *Firebase* e a integração completa com a *API* da Meta. O *Backend* foi desenvolvido com *Node.js* e o *framework Express.js*.

4.1.3 Teste e Validação

A fase de Testes avaliou o protótipo em diferentes situações para verificar a estabilidade do sistema e o comportamento do fluxo de mensagens. Foram testados desde o funcionamento do *webhook* até o tratamento de respostas inesperadas enviadas pelos usuários.

Na etapa de Validação, verificou-se se o sistema atendia aos requisitos definidos na fase inicial do projeto, resultando em ajustes nos fluxos e correções no código e no armazenamento no *Firebase*.

4.1.4 Documentação

A etapa final consistiu na Documentação. Nessa fase foram registradas as decisões técnicas, o funcionamento do sistema e a arquitetura utilizada, garantindo que o projeto pudesse ser reproduzido ou expandido futuramente.

O objetivo dessa estrutura metodológica foi demonstrar que o desenvolvimento seguiu uma abordagem prática com avanços progressivos e revisões frequentes do código, garantindo a organização e o bom funcionamento do protótipo.

4.2 Contexto Organizacional

Para desenvolver o *chatbot* voltado ao atendimento de uma corretora de seguros, os primeiros passos envolveram a análise da rotina real de trabalho desse tipo de empresa. Para isso, foi feito o levantamento das atividades junto aos responsáveis e colaboradores de uma empresa corretora de seguros em Goiânia-GO. Nesse processo, inicialmente foram filtradas as solicitações que apresentam volume elevado e caráter repetitivo, e quais delas representam uma parte significativa do tempo gasto pelos atendentes.

Como essas demandas acontecem frequentemente e seguem praticamente o mesmo passo a passo, elas serviram de motivação e base para decidir quais partes poderiam ser automatizadas pelo *chatbot*. Somente a partir dessa análise prática, é que foi possível compreender quais as funcionalidades deveriam estar no sistema, além dos requisitos básicos iniciais.

4.3 Requisitos do sistema

No sentido de que o *chatbot* pudesse atender tanto as necessidades simples do dia a dia quanto as demandas mais frequentes da corretora, a definição dos requisitos funcionais e não funcionais foi essencial. Eles serviram como um guia na construção do sistema e ajudaram a manter o foco no que realmente precisava ser entregue.

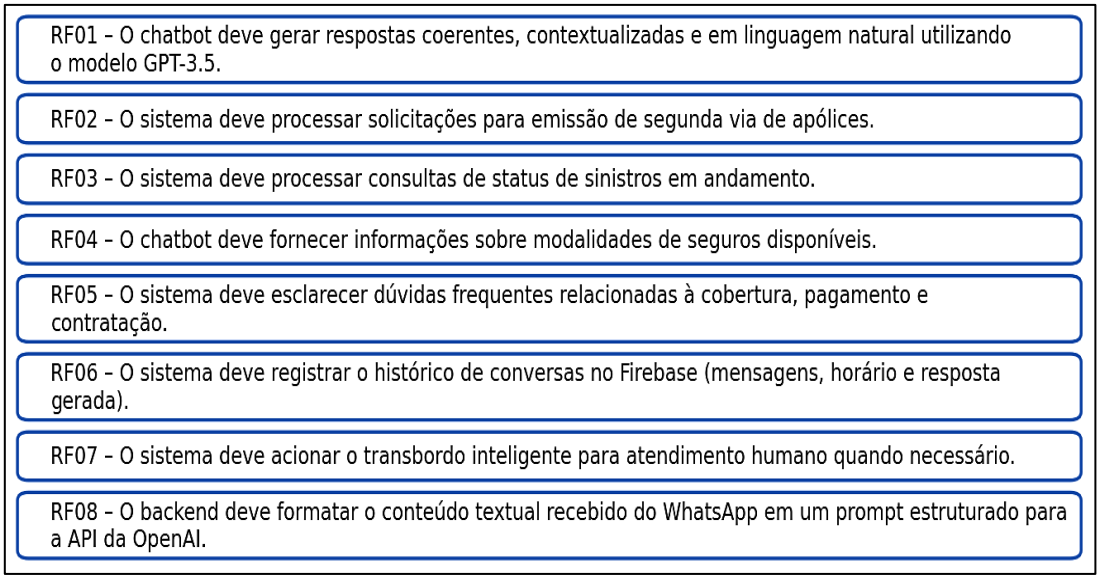
Os requisitos funcionais foram úteis pois permitiram identificar o que o sistema deveria fazer, como emitir segunda via de documentos, consultar boletos, além de fornecer informações sobre coberturas e registrar solicitações de sinistro.

Já os requisitos não funcionais mostraram as qualidades que o *chatbot* precisava demonstrar depois de pronto, como ele se comportará, se tem r*api*dez nas respostas, o que foi feito para a segurança no tratamento dos dados, disponibilidade, facilidade de uso e integração a com os sistemas.

4.3.1 Requisitos Funcionais (RF)

Na parte dos requisitos funcionais, foi detalhado o que o sistema tinha que fazer para executar as tarefas do atendimento automatizado. São as ações, operações e respostas que o *chatbot* deve executar. Nessa etapa foram definidas por exemplo, quais consultas o usuário vai poder fazer, quais informações o sistema deve devolver e quais processos ele consegue iniciar sozinho durante a conversa. A Figura 6 representa a definição de todos os requisitos funcionais:

Figura 6 - Requisitos Funcionais do Sistema

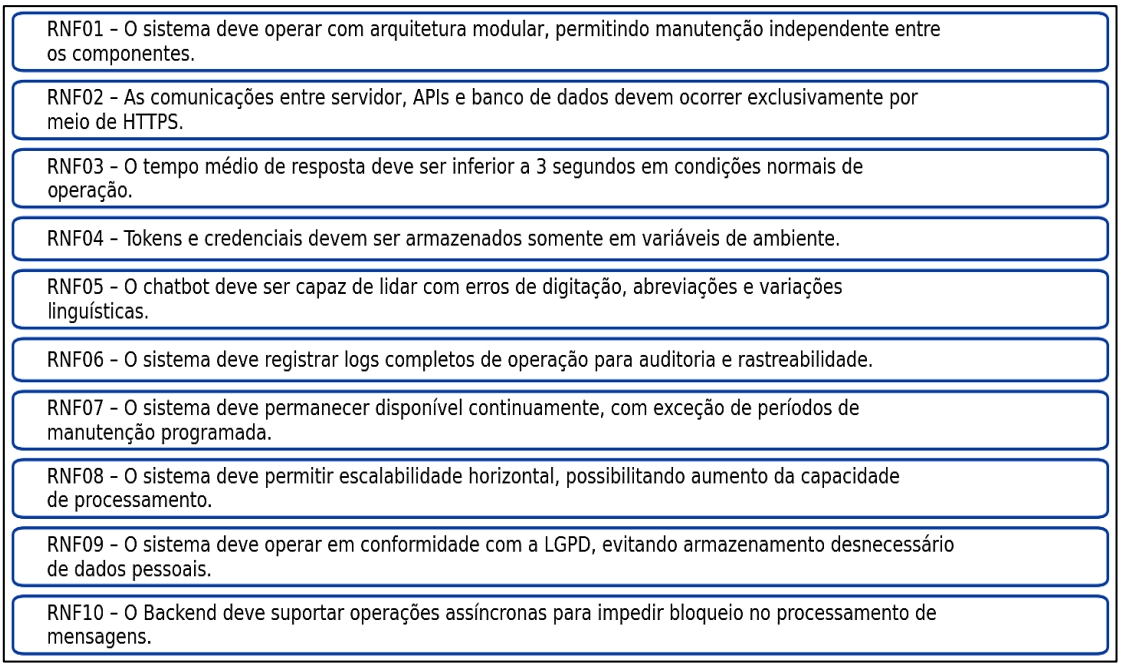


Fonte: Elaborado pelo autor (2025).

4.3.2 Requisitos Não Funcionais (RNF)

Já os requisitos não funcionais definiram características de qualidade importantes para o desempenho, mostrados na Figura 7.

Figura 7 - Requisitos Não Funcionais do Sistema.



Fonte: Elaborado pelo autor (2025).

4.4 Arquitetura do Sistema

Para começar a definição da arquitetura, o sistema foi inicialmente organizado de forma modular, isso permitiu um entendimento facilitado e alterar as partes separadamente. Essa divisão ficou estruturada em três camadas principais: entrada, processamento e armazenamento. Essa escolha acabou ajudando bastante na manutenção do código e permitiu ajustar algumas coisas no caminho sem precisar refazer tudo.

Uma vantagem da modularização é que além de aprimorar correções e mudanças pontuais, o sistema permite integrar novos serviços no futuro sem alterar o funcionamento central do *chatbot*.

A camada de entrada é a responsável por integrar a *API* do Meta e é por esse canal que as mensagens enviadas pelos usuários chegam ao servidor, onde realmente começarão a ser processadas.

Depois disso, veem a camada de processamento, ou núcleo lógico. Desenvolvida em *Node.js* usando o Express, é nessa parte que ficam as regras de atendimento, as validações e todo o controle do fluxo das conversas. Nessa camada o *Backend* chama outras *APIs* e faz a ponte entre os módulos do sistema.

O entendimento das mensagens em si fica por conta da *API* da OpenAI. Para interpretar o que o usuário escreve e gerar uma resposta coerente com o contexto da conversa foi utilizada a *API* do modelo *gpt-3.5-turbo*.

Já a camada de armazenamento e banco de dados desenvolvida com o uso do *Firebase*. Ele guarda o histórico das conversas, registra o estado de cada atendimento e mantém outras informações de logging, estatísticas e falhas, facilitando a utilização do sistema, que precisa funcionar de forma contínua e organizada.

Por fim, foi implementado um mecanismo de transbordo responsável por encaminhar a conversa para um colaborador sempre que o *chatbot* identifica que a solicitação ultrapassa o escopo automatizado. Esse recurso evita que o usuário permaneça sem atendimento e assegura a continuidade do processo mesmo em situações em que o sistema não é capaz de concluir a interação de forma autônoma.

4.5 Tecnologias de *Backend*: *Node.js*, *JavaScript*, *Express.js* e *Axios*

O *Backend* do sistema foi desenvolvido em *Node.js*, pensando na facilidade de integração com as *APIs*. Ele permite rodar *JavaScript* no servidor e oferece suporte nativo para todas as operações desejadas no projeto. A adoção é facilitada para esse tipo de sistema, pois o *Node.js* é uma solução projetada para trabalhar com um volume grande de mensagens chegando o tempo todo. Nesse ambiente, o servidor consegue lidar com várias requisições de forma contínua, sem travar, ao mesmo tempo em que espera respostas e sincroniza com os serviços externos.

A camada do servidor foi implementada em *JavaScript*. A escolha dessa linguagem também simplificou o desenvolvimento por utilizar sintaxe já familiar devido ao contato prévio com linguagens da mesma família, como C e C++.

Para organizar o *Backend*, foi adotado o *framework Express.js*, que oferece uma estrutura simples para criação das rotas, aplicação de *middlewares* e o tratamento de requisições. Essa ferramenta foi também utilizada para construir a *API* responsável por receber as mensagens do *WhatsApp* e encaminhá-las aos módulos de processamento.

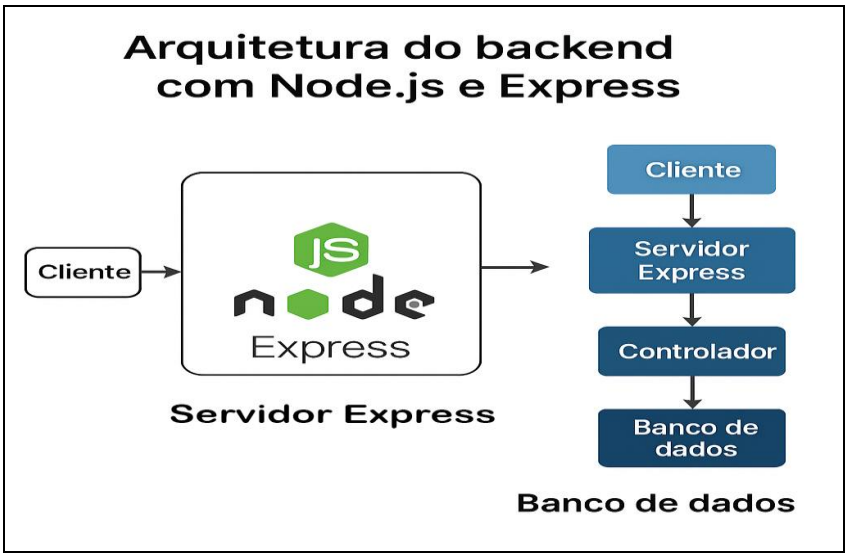
Na comunicação com serviços externos, como a *API* da Meta (*WhatsApp Business*) e a *API* da *OpenAI*, foi utilizada a biblioteca *Axios*. A biblioteca serviu para enviar e receber requisições *HTTP*, montar e enviar *payloads* em *JSON*, receber as respostas do modelo *gpt-3.5-turbo* e tratar erros de rede quando necessário.

A Figura 8 representa a organização geral do *Backend*. As requisições do usuário que são enviadas pelo *WhatsApp* chegam ao servidor que foi implementado com *Express.js*, ele recebe cada chamada e direciona para o controlador correspondente.

Os controladores são partes do *Backend* responsáveis por organizar o que acontece quando uma mensagem ou requisição chega ao servidor. Nesse caso cada controlador tem uma função específica. Por exemplo: um dos controladores és responsável por consultar boletos, outro por emitir documentos, outro por lidar com o histórico de atendimento.

Quando o Express recebe uma requisição, ele não executa nada diretamente; ele apenas encaminha para o controlador correto, que será responsável por processar essa requisição.

Figura 8 - Arquitetura básica backend.



Fonte: Elaborado pelo autor (2025).

4.6 Integração com a *API* do *WhatsApp* (*Meta API*) e *Webhook*

A comunicação entre os usuários e o *chatbot* foi implementada utilizando a *WhatsApp Business Platform*, disponibilizada pela Meta. É por meio dessa *API* que o sistema consegue enviar e receber mensagens da plataforma.

Para habilitar o uso da *API*, foi necessário cadastrar um CNPJ no *Meta Business Manager*, realizar a verificação comercial e vincular um número telefônico que foi destinado ao atendimento automatizado. A aprovação cadastral e comercial do grupo Meta levou em média 40 dias, e após todas as verificações o sistema pôde então gerar um *token* de acesso e permitir configurar o *webhook* responsável por receber as mensagens enviadas ao *WhatsApp*.

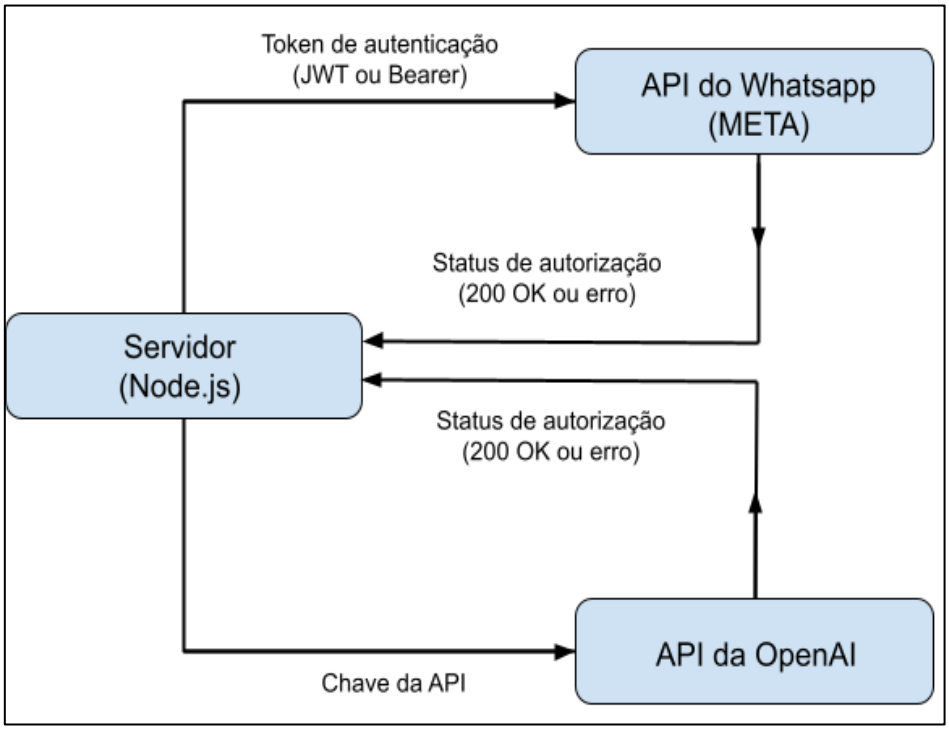
O servidor é o responsável por receber cada requisição enviada pelo *webhook*. Ele interpreta o conteúdo da mensagem e executa o fluxo correspondente, como por exemplo uma consulta de apólice.

Ele também é responsável por encaminhar a mensagem ao modelo da OpenAI e utiliza a resposta retornada para compor o atendimento.

No que diz respeito a cobrança da Meta para uso do serviço, cada tipo de mensagem enviada ou recebida tem um valor específico. As mensagens de marketing são as mais caras, enquanto as mensagens de utilidade são as mais baratas. No caso do projeto, a maior parte das interações acabou entrando justamente nessa categoria de utilidade. Esse tipo de mensagem é usado para enviar informações operacionais e orientações ao usuário, e por isso tem um custo bem menor dentro da plataforma.

Já na parte de autenticação com a Meta funciona da seguinte maneira: toda requisição enviada para o servidor vem acompanhada de um *token*, que é como uma “chave” usada para confirmar que aquela chamada realmente veio da Meta. O servidor só aceita então a mensagem que tiver a chave correta. Caso contrário ele retorna um status de erro. A Figura 9 apresenta o fluxo básico de autenticação descritas acima entre o *Backend*, a Meta e a OpenAI.

Figura 9 - Modelo de Autenticação



Fonte: Elaborado pelo autor (2025).

4.7 Integração com a *API* da *OpenAI* e Processamento de Respostas

O processamento realizado pela *API* da *OpenAI*, modelo *gpt-3.5-turbo* constitui a parte central da lógica inteligente do *chatbot*, auxiliando o servidor a manter o contexto das conversas e a organizar o atendimento de acordo com as demandas da corretora.

A inteligência conversacional do *chatbot* foi implementada por meio da *API* da OpenAI, utilizando o modelo *gpt-3.5-turbo*. Esse modelo foi adotado devido a facilidade de uso, baixo custo operacional e ao seu tempo de resposta adequado para aplicações em tempo real.

O modelo utilizado pertence à família *Generative Pre-trained Transformer* (*GPT*), projetada para interpretar texto, identificar intenções e gerar respostas com consistência. A cobrança é realizada com base na quantidade de *tokens* enviados e recebidos, o que permite ajustar os custos de acordo com o fluxo real de mensagens.

No *Backend*, as mensagens recebidas do usuário são convertidas em um *prompt* que reúne o texto enviado e, quando necessário, informações adicionais armazenadas no *Firebase*. Esse *prompt* é enviado à *API* da OpenAI por meio de uma requisição *HTTP* assíncrona no formato *JSON*. O modelo retorna uma resposta em linguagem natural, que é tratada pelo *Backend* para garantir que o conteúdo esteja de acordo com o padrão de atendimento definido para a corretora antes de ser encaminhado ao usuário.

Para este projeto, a *API* do *chatbot* foi programada para responder somente perguntas relacionadas à corretora de seguros, essa limitação de conteúdo foi feita para garantir que ele não respondesse dúvidas genéricas ou coisas que não têm relação com o contexto de seguros. O sistema também utiliza instruções internas no *prompt* para orientar o comportamento do modelo, permitindo controlar aspectos como o tom da resposta, o direcionamento aos menus apropriados e a recomendação de encaminhamento para atendimento humano quando necessário.

Além disso, o *Backend* realiza verificações adicionais para identificar respostas inadequadas, com gírias, malformadas, genéricas ou incompatíveis com o fluxo esperado. Quando esse tipo de situação ocorre, o sistema aplica regras de correção ou se necessário aciona o procedimento de transbordo.

4.8 Armazenamento e Segurança da Informação com *Firebase*

O armazenamento das interações entre os usuários e o *chatbot* foi feito usando o *Firebase*, ele por sua vez é um banco de dados *NoSQL* com suporte a operações em tempo real. A adoção dessa ferramenta se deu porque ela se integra muito bem com aplicações em *JavaScript* e já trabalha nativamente com operações que combinam com o volume de mensagens que o sistema precisa processar.

O *Firebase* registra as conversas de forma organizada, separando por usuário e por sessão de atendimento. Além das mensagens em si, ele também guarda informações importantes, como horário das interações (*timestamps*), em que estado o atendimento está, e se houve transbordo para um atendente humano, além de outros eventos usados para auditoria.

As regras de acesso do banco de dados foram configuradas para permitir operações somente dos serviços que realmente são autorizados. O próprio sistema de segurança do *Firebase* permite definir permissões baseadas em autenticação e em algumas condições específicas. Essa configuração foi adotada para que apenas componentes autorizados possam realizar operações no banco.

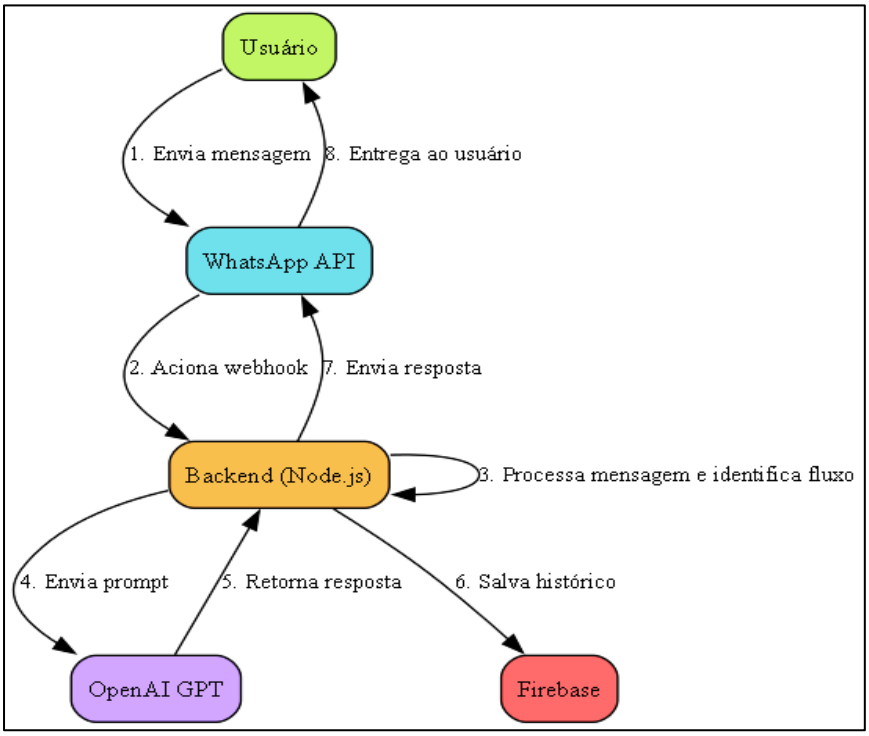
As credenciais sensíveis, como as chaves do *Firebase*, ficaram guardadas em variáveis de ambiente para não aparecerem no código-fonte e para não ficarem expostas durante o tráfego. De modo que o sistema também foi configurado para armazenar apenas o que realmente é necessário para que o atendimento funcione.

O *Firebase* funciona como a principal camada de persistência da solução. Ele registra todas as interações, além de manter o contexto das conversas ao longo das sessões e fornecer dados que poderão ser usados posteriormente para análises de comportamento do sistema.

4.9 Fluxo de Comunicação do *Chatbot*

A Figura 10 representa o fluxo de comunicação e mostra, na prática, como cada parte do sistema participa do atendimento. Ele começa quando o usuário envia uma mensagem pelo *WhatsApp*. Assim que a Meta recebe essa mensagem, automaticamente, encaminha para o *Backend* por meio do *webhook* configurado.

Figura 10 - Fluxo completo de processamento da mensagem no *chatbot*.



Fonte: Elaborado pelo autor (2025).

Quando o servidor recebe essa requisição, ele identifica o usuário pelo número de telefone e consulta no *Firebase* qual é o estado daquela sessão, verificando se já existe uma conversa ativa ou se é um atendimento novo. Com essas informações, o sistema já sabe qual fluxo deve seguir.

No caso de a mensagem precisar de análise de texto, o *Backend* monta um *prompt* com o que o usuário escreveu e com trechos relevantes do histórico. Esse *prompt* é enviado para a *API* da OpenAI, que gera a resposta. Assim que ela retorna, o *Backend* revisa, formata e enviam novamente para o *WhatsApp*.

Em paralelo, cada interação é registrada no *Firebase*, onde ficam salvas as mensagens, os horários, o estado do atendimento e até as situações em que houve transbordo.

4.10 Objetivo da implementação

O objetivo da implementação é desenvolver um ambiente funcional para a construção, integração e validação de um chatbot inteligente voltado ao atendimento automatizado em corretoras de seguros. A implementação tem como finalidade criar um cenário controlado que permita testar a comunicação entre a plataforma WhatsApp, o backend da aplicação e o modelo de linguagem baseado em GPT, avaliando o comportamento do sistema em interações reais de atendimento.

O resultado esperado é demonstrar como a integração entre inteligência artificial e plataformas de mensageria pode ser aplicada de forma eficiente no suporte ao cliente, automatizando atendimentos iniciais, reduzindo o tempo de resposta e auxiliando na organização do fluxo de atendimento, sem substituir o atendimento humano em casos que exigem maior complexidade.

4.11 Transbordo e Contingência

Embora o sistema tenha sido feito para trabalhar de forma totalmente autônoma foi preciso incluir um mecanismo de transbordo humano para atender os casos em que o c*hatbtot* não consegue resolver sozinho. Isso evita que o usuário permaneça em um fluxo de atendimento sem progresso e garante que alguém da corretora consiga assumir quando for necessário.

O transbordo é ativado com base em regras definidas no *Backend*. Por exemplo: quando o usuário repete várias vezes a mesma dúvida, quando aparece alguma expressão que indica que a pessoa precisa de uma orientação mais técnica, ou quando ela pede claramente para falar com um atendente. Essas verificações são feitas pelo próprio servidor, independentemente da resposta do GPT.

Quando o sistema identifica que precisa acionar o transbordo, o *Backend* registra essa situação no *Firebase* e envia uma notificação para o time de

suporte. Assim, o atendente consegue assumir a conversa já com todo o histórico registrado, sem precisar pedir informações novamente.

Além disso, foram implementadas algumas rotinas de contingência para lidar com falhas externas, como instabilidade na *API* da Meta ou demora nas respostas da OpenAI. Quando isso acontece, o *Backend* tenta reenviar a requisição e registra tudo nos *logs* para análise posterior. O sistema também conta com redundância de provedores de *API*, permitindo recorrer a outra opção caso uma delas fique indisponível.

Também foi efetuada a implementação de um mecanismo de *fallback* para as situações em que o sistema não consegue finalizar o processamento da mensagem. Quando isso ocorre o *chatbot* envia uma resposta padrão orientando o usuário a tentar novamente, aguardar um momento ou pedir para falar com um colaborador.

4.12 Ambiente de Desenvolvimento

O desenvolvimento do projeto foi realizado utilizando a *IDE Visual Studio Code*, escolhida pelas ferramentas integradas que oferece, como o terminal embutido e o suporte a extensões para projetos em *JavaScript* e *Node.js*. Os testes e execuções do *Backend* foram feitos localmente em uma máquina com Windows 11 Professional, utilizando o *Node.js* na versão 21.7.1.

Para iniciar a construção do projeto, foi necessário instalar as dependências e ferramentas listadas a seguir:

• *Node.js*: plataforma de execução e desenvolvimento em JavaScript no lado do servidor;

• *Node Package Manager* (NPM): gerenciador de pacotes do Node.js;

• *Express.js: framework* para *Node.js* utilizado na construção da *API* do *backend*;

• Axios: biblioteca utilizada para realizar requisições *HTTP* às *APIs* externas;

• *dotenv*: biblioteca para carregamento de variáveis de ambiente a partir de um arquivo. env;

• *Firebase (Firestore):* banco de dados *NoSQL* utilizado para registrar interações, armazenar estados de conversa e dados relacionados ao atendimento;

• APIs externas: *API oficial do WhatsApp Business Platform* (Meta) para envio e recebimento de mensagens e *API* da *OpenAI* para processamento e geração de respostas com o modelo *GPT*.

5 TESTES, RESULTADOS E AVALIAÇÃO DO SISTEMA

Este capítulo apresenta os testes realizados para avaliar o funcionamento do *chatbot*, verificando tanto a estabilidade do sistema quanto a forma com que ele interpreta as mensagens e atende aos requisitos definidos no projeto. Também são descritos os procedimentos adotados durante a execução dos testes e os resultados obtidos em cenários que simulam o uso real da aplicação.

5.1 Arquitetura Escalável e Considerações de Segurança

A arquitetura modular utilizada no desenvolvimento permitiu realizar ajustes no *Backend* sem afetar outras partes do sistema. A separação entre as camadas de entrada, processamento e armazenamento contribuiu para manter o funcionamento estável durante os testes.

No que diz respeito à segurança, credenciais e chaves de acesso foram armazenadas em variáveis de ambiente, evitando exposição direta no código. As conexões foram realizadas por meio de *HTTPS*, e validações de entrada foram aplicadas para reduzir riscos de chamadas incorretas ou não autorizadas. O registro de *logs* auxiliou na identificação de falhas e no monitoramento do comportamento do sistema durante as interações.

5.2 Metodologia de Testes

A metodologia de testes foi organizada para verificar se o *chatbot* atendia aos requisitos funcionais e não funcionais definidos no projeto. Os testes avaliaram a consistência das respostas, a estabilidade da comunicação e a integridade dos dados armazenados.

Foram adotadas as seguintes abordagens:

- Testes exploratórios: simulações de atendimentos reais, incluindo dúvidas sobre apólices, solicitações de segunda via, perguntas sobre coberturas e pedidos diretos de atendimento humano.
- Testes baseados em cenários: validação de fluxos como abertura de sinistro, consulta de status e triagem de solicitações, considerando caminhos alternativos e mensagens inesperadas.

- Testes de robustez linguística: envio de mensagens com abreviações, erros de digitação, gírias e linguagem informal para avaliar a interpretação do modelo *GPT*.
- Testes de carga moderada: envio de várias mensagens em sequência por usuários simulados para observar o comportamento do *Backend* sob maior volume de requisições.
- Testes de falhas e exceções: simulação de indisponibilidade da *Meta API* ou da OpenAI para verificar o funcionamento dos mecanismos de *fallback* e transbordo humano.

Além dos testes manuais, foram criados *scripts* em *Node.js* e *Python* conectados ao *Backend* para envio automático de lotes de mensagens em intervalos regulares. Esses *scripts* permitiram simular picos moderados de carga e repetição rápida de comandos.

Durante a execução dos testes, o *Backend* registrou automaticamente:

- horários de envio e recebimento
- tempo de processamento de cada interação
- falhas na comunicação com serviços externos
- mensagens não interpretadas
- ocorrências de *timeout* ou indisponibilidade

5.3 Resultados Funcionais

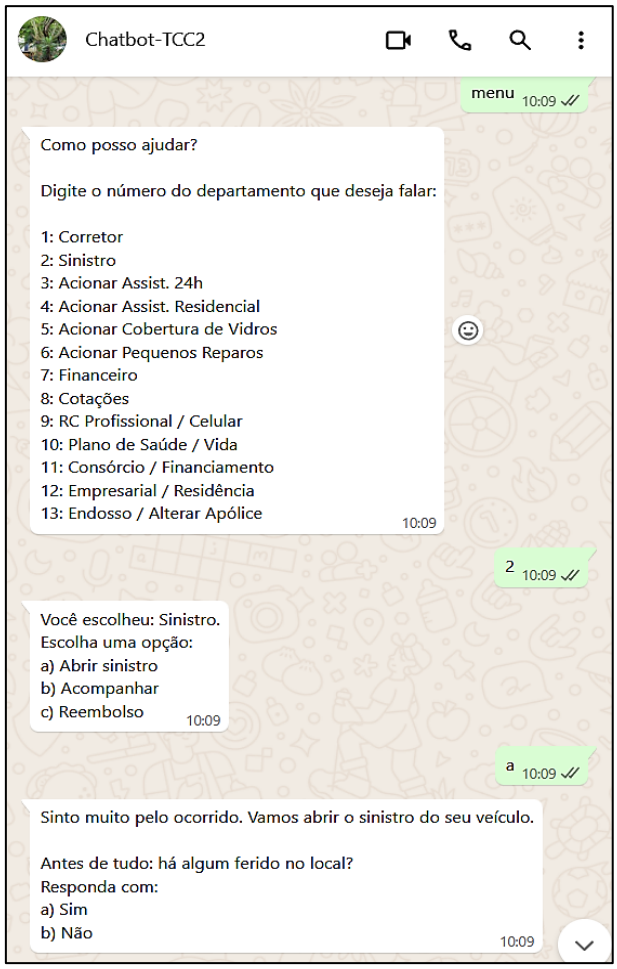
Neste tópico, apresento alguns exemplos reais das interações que foram executadas durante os testes, mostrando como o sistema se comportou em diferentes situações do atendimento.

A Figura 11 representa um exemplo da interação inicial realizada durante os testes. O fluxo de atendimento começa quando o usuário envia o comando “menu”, momento em que o sistema exibe o menu principal. A partir daí, o usuário já consegue visualizar os treze departamentos disponíveis e, no caso apresentado, seleciona a opção “2”, referente ao setor de Sinistro.

Na sequência, o chatbot apresenta as opções “a) Abrir sinistro”, “b) Acompanhar” e “c) Reembolso”. Ao escolher “a”, o sistema inicia automaticamente o fluxo guiado de abertura do atendimento. Logo depois, o

chatbot começa a coletar as informações necessárias e pergunta, por exemplo, se houve feridos no local.

Figura 11 - Menu principal do *Chatbot.*

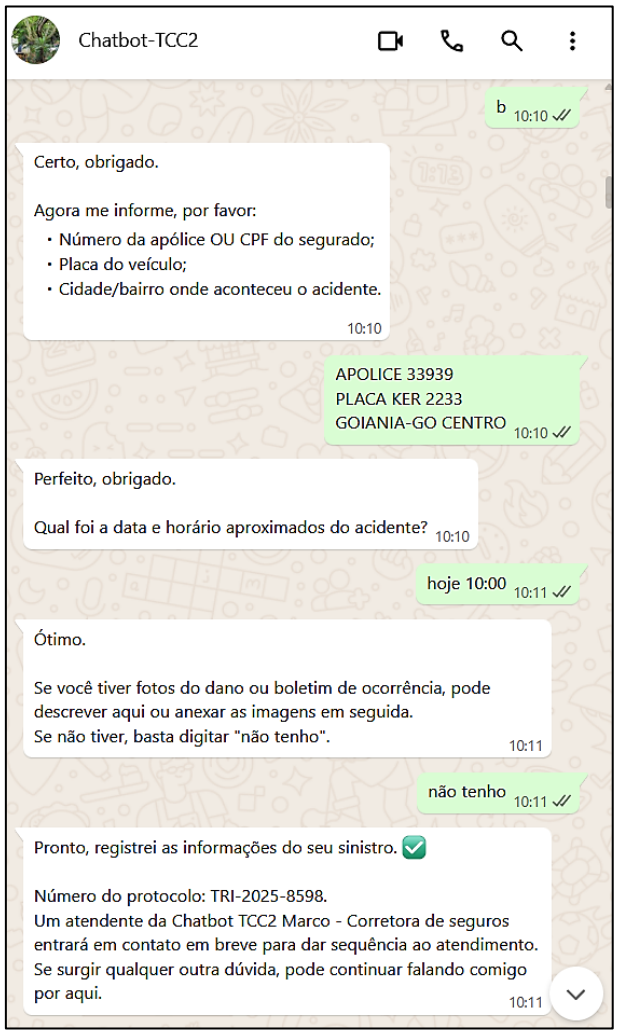


Fonte: Elaborado pelo autor (2025).

A Figura 12 representa a continuação do fluxo de abertura de sinistro executado durante os testes funcionais. Após o usuário selecionar a opção correspondente ao departamento de sinistro, o *chatbot* solicita informações essenciais, como número da apólice ou CPF do segurado, placa do veículo e local do acidente.

O sistema então coleta dados adicionais na sequência, como data e horário aproximado do ocorrido, e permite ao usuário anexar fotos ou boletim de ocorrência, quando disponíveis. Ao final, o *chatbot* registra automaticamente as informações recebidas e gera um número de protocolo, salvando no banco de dados e permitindo posterior consulta e análise.

Figura 12 - Fluxo de abertura de sinistro.

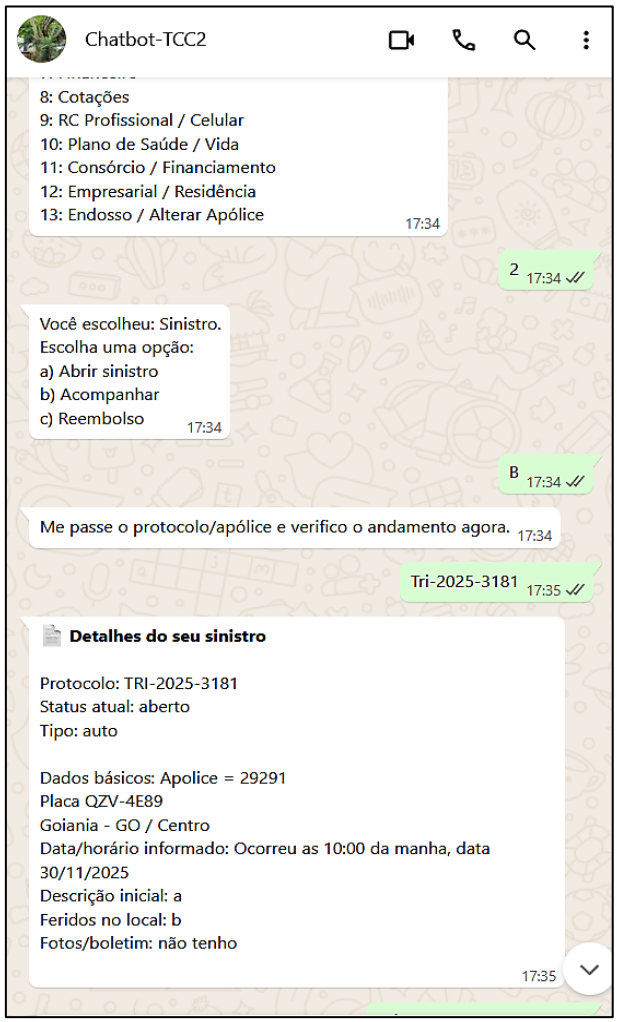


Fonte: Elaborado pelo autor (2025).

A figura 13 representa uma consulta posterior de sinistro no *WhatsApp* entre o usuário e o *chatbot.* Após selecionar o departamento Sinistro e a opção b) Acompanhar, o *chatbot* solicita que o usuário informe o número do protocolo ou da apólice para verificar o andamento do atendimento.

O usuário envia o código “Tri-2025-3181”, e o sistema consulta o banco de dados e responde imediatamente informando os detalhes completos do sinistro associado ao protocolo informado.

Figura 13 - Fluxo de consulta de sinistro.



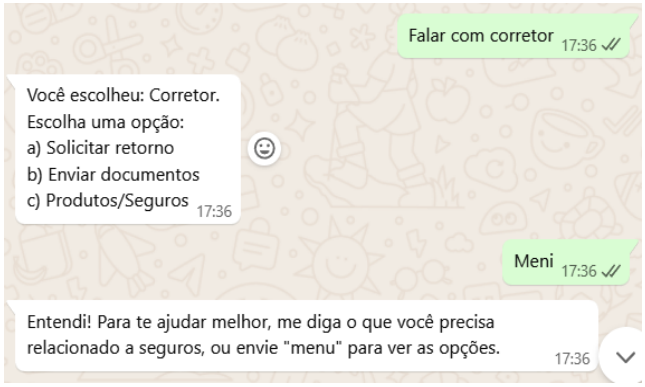
Fonte: Elaborado pelo autor (2025).

Um diferencial desse tipo de *bot* em relação aos modelos de regras fixas é a possibilidade de o sistema identificar intenções diversas, por exemplo, relacionadas a consultas de apólice, emissão de documentos, informações sobre coberturas e orientações iniciais de abertura de sinistro.

Depois de registrar um sinistro, o usuário quis falar com o corretor. Em um sistema tradicional ele teria que voltar ao menu e refazer todo o caminho, mas aqui isso não acontece. Basta digitar que quer falar com o corretor e o sistema identifica e faz a transferência direto.

Além disso, mesmo quando o usuário digita “meni” errado, o sistema não trava. Ele entende a intenção e orienta corretamente, sugerindo o comando “menu” para continuar o atendimento. A Figura 14 representa a demonstração destes dois fluxos:

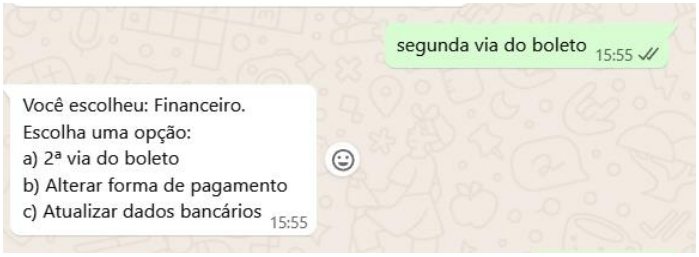
Figura 14 - Fluxo corretor.



Fonte: Elaborado pelo autor (2025).

A Figura 15 representa o usuário digitando “segunda via do boleto” e o *chatbot* já entendendo direto o que ele quer exibindo o setor Financeiro, e as alternativas corretas para continuar o atendimento.

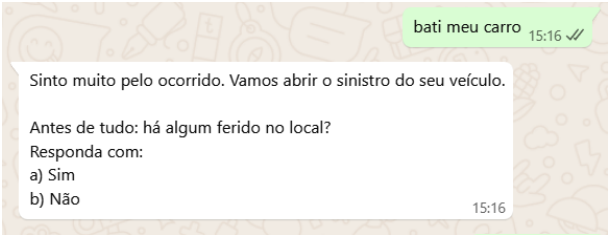
Figura 15 - Fluxo segunda via de boleto.



Fonte: Elaborado pelo autor (2025).

A Figura 16 representa o usuário digitando apenas “bati meu carro”, sem seguir menu, o *bot* já entende automaticamente que se trata de uma abertura de sinistro. A resposta é imediata: o sistema inicia o atendimento, demonstra empatia e inicia o fluxo padrão perguntando se houve feridos no local.

Figura 16 - Fluxo abertura de sinistro alternativo.



Fonte: Elaborado pelo autor (2025).

Este capítulo mostrou um pouco da capacidade do modelo de interpretar linguagem natural e abrir processos sem depender de comandos rígidos ou opções numéricas. Isso significa que o usuário não precisa seguir o menu passo a passo nem ficar navegando por várias opções. O sistema identifica a intenção automaticamente e já encaminha para o setor correto, tornando o atendimento mais ágil e intuitivo.

5.4 Avaliação de Desempenho

Este tópico apresenta o processo de avaliação de desempenho do chatbot, realizado por meio de vários testes controlados que simularam diferentes condições de uso. Os testes foram planejados para medir indicadores como tempo médio de resposta, estabilidade do diálogo e comportamento do sistema diante de múltiplas requisições consecutivas. essa métricas analisadas permitiram observar o desempenho geral do protótipo e identificar possíveis pontos de melhoria no fluxo de atendimento.

Após a fase de desenvolvimento e refinamento do *backend*, foi criado um módulo de monitoramento integrado ao servidor do *chatbot*, que capturou dados que chegavam direto ao *webhook* e ficou responsável por registrar em tempo real todas as interações processadas.

Posteriormente, esses registros foram exportados para um arquivo CSV, e interpretados em Python permitindo a análise complementar das métricas coletadas. O arquivo CSV utilizado na avaliação contém o registro de cada interação processada durante o período de testes. Cada linha representa uma interação individual e reuniu informações necessárias para o cálculo das métricas do estudo.

Esse módulo também contabilizou o total de mensagens recebidas, e calculou automaticamente o tempo de resposta a partir dos carimbos de data e hora, e identificou se cada atendimento foi resolvido pelo *chatbot* ou exigiu transbordo.

Considerando o envio entre 5 e 10 mensagens diárias durante os 42 dias de testes, estima-se que o volume total ficaria entre 210 e 420 interações. O arquivo final registrou 320 interações, valor compatível com a média prevista para o período.

Estes registros após consolidados possibilitaram o cálculo de métricas como tempo médio de resposta, taxa de resolução automática, taxa de compreensão e número de falhas técnicas observadas.

A Figura 17 apresenta os campos adotados no arquivo de métricas:

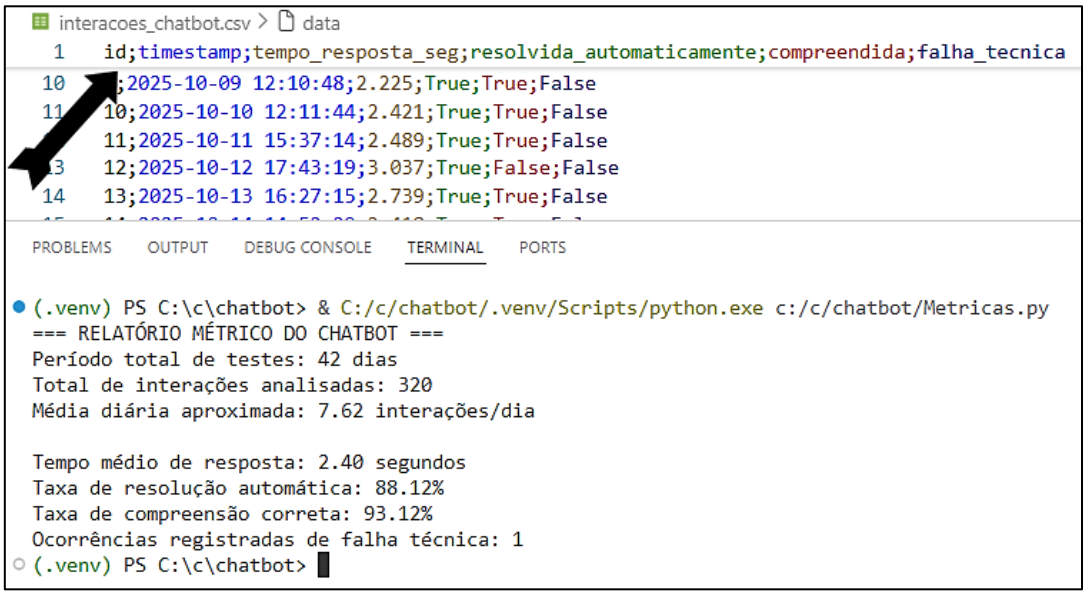
Figura 17 - Descrição dos campos utilizados nas métricas do chatbot

Campo	Descrição
id	Identificador numérico sequencial da interação, utilizado para organização e consultas posteriores.
timestamp	Data e horário exatos da interação, permitindo calcular a duração total dos testes e analisar a distribuição temporal dos eventos.
tempo_resposta_seg	Tempo, em segundos, entre o recebimento da mensagem e a resposta enviada pelo chatbot, utilizado no cálculo do tempo médio de resposta.
resolvida_aut.	Valor booleano que indica se a interação foi concluída sem intervenção humana, permitindo calcular a taxa de resolução automática.
compreendida	Campo booleano que identifica se a mensagem do usuário foi corretamente interpretada pelo modelo, base para o cálculo da taxa de compreensão.
falha_tecnica	Valor booleano que aponta ocorrências de instabilidade, como indisponibilidade de serviços externos ou timeout, utilizado para mensurar a robustez operacional do sistema.

Fonte: Elaborado pelo autor (2025).

A Figura 18 apresenta um trecho do arquivo CSV gerado, contendo todos os dados que representam cada mensagem processada ao longo dos testes.

Figura 18 - Amostra do arquivo CSV utilizado na coleta de métricas



Fonte: Elaborado pelo autor (2025).

Para a condução dos testes, foram enviados ciclos de mensagens previamente definidos, incluindo simulações de solicitações reais, como emissão de segunda via de apólice, consultas de dados e fluxos completos de atendimento. Cada ciclo envolveu grupos entre 5 e 10 mensagens, variando entre perguntas simples, fluxos completos e comandos enviados em sequência rápida.

Durante as interações, o *backend* registrou automaticamente, tanto nos logs do servidor quanto no *Firebase*, o tempo de processamento de cada requisição, além de qualquer falha eventual, resposta inválida ou situação de *timeout* que pudesse ocorrer.

A Figura 19 representa um trecho gerado pelo *script* de reprodução dos testes. Para cada interação, o sistema armazenou o *timestamp* registrado, a mensagem de teste enviada, o tempo de resposta medido, além dos indicadores de resolução automática, compreensão correta da mensagem e presença de falha técnica. Essa bateria de testes permitiu avaliar como o *bot* se comporta em cenários de atendimento durante o período de avaliação. A partir desses registros foram calculados os indicadores apresentados no trabalho.

Figura 19 - Bateria de Testes

```
--- Interação 210 ---
Timestamp registrado: 2025-11-11 10:15:07
Mensagem de teste enviada: Quero cancelar meu seguro atual.
Tempo de resposta registrado: 3.38 segundos
Resolvida automaticamente?: Sim
Mensagem compreendida?: Não
Falha técnica registrada?: Não

--- Interação 126 ---
Timestamp registrado: 2025-11-11 10:26:41
Mensagem de teste enviada: Quero cotar um seguro auto.
Tempo de resposta registrado: 2.52 segundos
Resolvida automaticamente?: Sim
Mensagem compreendida?: Sim
Falha técnica registrada?: Não

--- Interação 84 ---
Timestamp registrado: 2025-11-11 12:29:58
Mensagem de teste enviada: Pode me informar os dados da minha apólice atual?
Tempo de resposta registrado: 2.02 segundos
Resolvida automaticamente?: Sim
Mensagem compreendida?: Sim
Falha técnica registrada?: Não
```

Fonte: Elaborado pelo autor (2025).

5.5 Validação dos Requisitos

Para verificar se o *chatbot* realmente atendia ao que foi proposto no início do projeto, foi efetuada a validação direta dos requisitos funcionais e não funcionais. Foram efetuados diversos testes práticos de cada parte do sistema, para verificar se ele realmente fazia o que deveria e se o desempenho, a segurança e a estabilidade se manteriam durante o uso.

Essa validação foi baseada nos cenários de teste já descritos, incluindo simulações de atendimentos reais, testes de carga moderada e situações de falha. Com isso, pode-se observar o comportamento do sistema em diferentes condições e verificar, ponto a ponto, se cada requisito foi atendido.

A análise final é de que todos os requisitos funcionais foram atendidos nos cenários testados. O *chatbot* executou todas as operações principais previstas, como consultas, abertura de sinistro e emissão de documentos. Nos requisitos não funcionais, o desempenho também foi positivo, com destaque para a segurança dos dados, a estabilidade do *Backend* e o tempo de resposta.

As validações parciais ocorreram em aspectos relacionados ao uso de *APIs* externas e à interpretação de mensagens muito fora do padrão esperado. Ainda assim, os mecanismos de *fallback* e transbordo garantiram continuidade no atendimento, mitigando impactos significativos na experiência do usuário. De forma geral, a validação indica que o comportamento adequado para um protótipo funcional aplicado ao setor de seguros. A Figura 20 representa a consolidação dos requisitos funcionais avaliados e o resultado obtido para cada um deles.

Figura 20 - Resultados dos Requisitos Funcionais (RF).

Requisito	Tipo	Resultado	Observação
Redução de intervenções humana	Não funcional	Atendido	88% dos atendimentos concluídos sem necessidade de transbordo
Tempo médio de resposta	Não funcional	Atendido	Média de 2,4 segundos, dentro do esperado para sistemas em tempo real
Segurança dos dados	Não funcional	Atendido	Uso de variáveis de ambiente, HTTPS e regras do Firebase
Robustez linguística	Não funcional	Atendido parcialmente	Algumas variações incomuns exigiram reenvio ao modelo
Estabilidade do Backend	Não funcional	Atendido	Funcionamento contínuo 24/7 durante 42 dias de teste
Disponibilidade das APIs externas	Não funcional	Atendido parcialmente	Dependência de serviços da Meta e OpenAI gerou uma ocorrência de falha

Fonte: Elaborado pelo autor (2025).

A Figura 21 representa a consolidação dos requisitos não funcionais avaliados e o resultado obtido para cada um deles.

Figura 21 - Resultados dos Requisitos não Funcionais (RNF).

Requisito	Tipo	Resultado	Observação
Emitir segunda via de apólice	Funcional	Atendido	Mensagens processadas corretamente e resposta imediata ao usuário
Consultar boletos	Funcional	Atendido	Consulta concluída com sucesso e retorno estruturado
Informar coberturas	Funcional	Atendido	Sistema interpretou corretamente perguntas variadas e abreviações
Abertura de sinistro	Funcional	Atendido	Fluxo completo executado; inclusão de informações, fotos e registro no Firebase
Acompanhar sinistro	Funcional	Atendido	Consulta de protocolo retornou dados corretos em tempo real

Fonte: Elaborado pelo autor (2025).

6 ANÁLISE DOS RESULTADOS OBTIDOS

Este capítulo apresenta a análise dos resultados obtidos com a implementação e os testes realizados sobre o *chatbot* inteligente baseado no modelo GPT e integrado à plataforma *WhatsApp*. A avaliação considera aspectos quantitativos e qualitativos, tomando como referência os dados registrados durante o uso do sistema em ambiente de testes com usuários reais.

O objetivo é verificar em que medida os objetivos propostos no início do projeto foram alcançados, identificar pontos de melhoria e discutir limitações da solução desenvolvida.

Durante a análise dos resultados funcionais, O chatbot apresentou desempenho satisfatório nos cenários avaliados. Desde entender as intenções do usuário, mesmo quando a pessoa escrevia com abreviações ou pequenos erros, seguiu corretamente os fluxos de atendimento. Isso tornou a interação mais natural e ágil além de mais parecida com o jeito que os clientes realmente conversam no dia a dia.

Mesmo com o menu disponível, o usuário não precisa seguir aquele passo a passo. Ele pode simplesmente digitar o assunto que deseja resolver, e o próprio sistema irá reconhecer a intenção e o direcionar para o fluxo correto. Isso tende a reduzir etapas desnecessárias e deixa o atendimento mais ágil, especialmente nos processos de abertura de sinistro, que normalmente exigem maiores informações.

O transbordo foi projetado para funcionar sempre que o sistema identificar que a solicitação não pode ser resolvida automaticamente. Com os registros coletados, foi possível organizar indicadores de desempenho e avaliar como o *chatbot* se comportou em termos de eficiência e estabilidade durante os testes. A Figura 22 representa o resultado das principais métricas que foram analisadas.

Figura 22 - Indicadores de desempenho do protótipo.

Indicador	Resultado obtido
Total de interações simuladas	320
Tempo médio de resposta do chatbot	2,4 segundos
Taxa de resolução sem intervenção humana	88%
Taxa de mensagens compreendidas corretamente	93%
Ocorrências de falhas técnicas	01 (instabilidade de conexão)

Fonte: Elaborado pelo autor (2025).

Os resultados mostraram que o tempo médio de resposta, de 2,4 segundos, foi suficiente para manter a interação fluida e bastante compatível com o uso em tempo real. A taxa de resolução sem intervenção dos colaboradores, de 88%, indica que o *chatbot* conseguiu resolver a maior parte das solicitações sem precisar encaminhar o usuário para um atendente.

A taxa de compreensão das mensagens, que chegou a 93%, mostra que o modelo conseguiu interpretar bem as entradas dos usuários dentro do escopo definido. Durante o período de testes, foi registrado um evento de falha associado a uma instabilidade global no serviço da *Cloudflare*, que acabou afetando também a Meta, a OpenAI e vários outros sistemas. Mesmo assim, o *chatbot* se manteve estável ao longo dos testes, embora fique evidente que ele ainda depende diretamente da qualidade da infraestrutura de rede.

A comparação com estudos da literatura permite situar o projeto no contexto mais amplo da adoção de *chatbots* e inteligência artificial no setor de seguros. Ferreira et al. (2021) analisam a utilização de *chatbots* no pós-venda de seguros de pessoas, com foco em tarefas simples como emissão de segunda via de documentos e atualização cadastral. Os autores relatam boa aceitação dos usuários, mas apontam limitações na compreensão de perguntas mais complexas, uma vez que o sistema se baseava em regras e fluxos pré-definidos. Em contraste, o protótipo desenvolvido neste trabalho, apoiado no modelo GPT-3.5, apresentou maior flexibilidade para interpretar perguntas ambíguas e adaptar respostas ao contexto imediato da interação.

No estudo de Silva, Marques e Rocha (2024), o *chatbot* proposto é voltado à comunicação interna entre departamentos em uma cooperativa de crédito, priorizando a eficiência operacional. Trata-se de uma solução com fluxos fixos e funcionamento restrito ao ambiente institucional, sem foco em linguagem natural avançada. Em comparação, o sistema apresentado opera em um canal amplamente utilizado pelo público geral (*WhatsApp*) e explora modelos de linguagem de última geração, ampliando a aplicabilidade em cenários de relacionamento com clientes externos.

Já Silva, Chan e Decoster (2025) identificam, em análise documental, que apenas uma parcela limitada das seguradoras brasileiras menciona explicitamente o uso de inteligência artificial em seus relatórios institucionais, com aplicações concentradas em atendimento ao cliente e automação de

processos administrativos. O protótipo que foi desenvolvido neste projeto se insere diretamente nessa tendência, e forneceu um exemplo concreto de aplicação de IA conversacional em atendimento, com potencial de redução de sobrecarga dos colaboradores e aumento da disponibilidade do serviço.

A Figura 23 representa o comparativo entre estudos relacionados e o projeto desenvolvido.

Figura 23 - Comparativo entre estudos relacionados e o projeto desenvolvido.

Estudo	Aplicação	Tecnologia	Diferencial
Ferreira et al. (2021)	Pós-venda em seguros de pessoas	Chatbot baseado em regras simples	Boa aceitação em tarefas simples, mas limitações em perguntas complexas
Silva, Marques e Rocha (2024)	Chatbot interno em cooperativa de crédito	Fluxos fixos para comunicação interna	Melhoria organizacional interna, mas sem foco em NLP
Silva, Chan e Decoster (2025)	Relatórios de IA no setor segurador	Análise documental de uso de IA	Identificação de baixa adesão de IA no setor
Este Projeto	Atendimento automatizado via WhatsApp com GPT	GPT-3.5 + API do WhatsApp + Firebase	Alta flexibilidade, linguagem natural e integração em tempo real

Fonte: Elaborado pelo autor (2025).

Em comparação deste projeto com os outros trabalhos analisados, verificou-se que alguns estudos focam em chatbots mais simples, baseados em regras ou usados só dentro de uma organização, para o desenvolvimento deste a proposta foi testar um modelo de linguagem mais avançado em um cenário real.

O uso do GPT ofereceu mais flexibilidade nas respostas, mas também introduziu limitações como custo, dependência de APIs e possíveis falhas de disponibilidade, aspectos que os outros estudos, baseados em regras fixas, não enfrentam da mesma forma. Por outro lado, esses sistemas mais simples também não alcançam a mesma capacidade de interpretação contextual. Assim, o projeto se encaixa como uma alternativa intermediária: não é a solução definitiva para o setor, mas pode contribuir para o avanço de modelos de linguagem avançados em aplicações práticas, desde que suas restrições operacionais sejam consideradas.

Durante o desenvolvimento do protótipo, algumas limitações foram identificadas:

- Integração com a plataforma da Meta: o processo de verificação comercial e a configuração de variáveis de ambiente, *tokens* e permissões de *webhook* mostraram-se etapas sensíveis. Inconsistências nessas configurações resultaram em falhas de autenticação e ausência de eventos, exigindo revisões sucessivas.
- Ajustes nos *prompts* do modelo GPT-3.5: pequenas alterações na forma de formular instruções geravam respostas inconsistentes. Foi necessário um refinamento contínuo para garantir clareza, precisão e alinhamento ao contexto da corretora.
- Adaptação do fluxo de atendimento: ajustes foram necessários para evitar repetições no diálogo e assegurar que o sistema encaminhasse o usuário ao atendimento humano quando apropriado. Critérios de transbordo foram aprimorados ao longo dos testes.

7 CONCLUSÃO

Este trabalho teve o intuito de responder à seguinte questão de pesquisa: **um *chatbot* com IA integrada, baseado no modelo *GPT* e conectado ao *WhatsApp* pode otimizar o atendimento ao cliente em corretoras de seguros?**

O objetivo geral foi desenvolver um *chatbot* inteligente com integração ao *WhatsApp*, utilizando as *APIs* da Meta e o modelo *GPT*, para otimizar o atendimento ao cliente em corretoras de seguros.

O sistema foi desenvolvido com capacidade de automatizar atendimentos, reduzir o tempo de resposta, manter interações em linguagem natural e atender de forma eficiente demandas recorrentes. A integração entre o modelo *GPT-3.5* e a plataforma *WhatsApp* contribuiu para a otimização do atendimento nesse contexto.

Assim a solução apresentou funcionamento estável, com arquitetura modular que possibilitou a integração das tecnologias e a geração de respostas contextualizadas em tempo real para o setor segurador. O estudo teórico aliado à prática, com uso de modelos baseados na arquitetura *Transformer*, ampliou a compreensão sobre inteligência artificial e *APIs* de mensageria.

Ao efetuar o levantamento das demandas mais recorrentes do setor foram identificadas as principais necessidades de atendimento e assim definir funcionalidades como o esclarecimento de dúvidas frequentes, atendimento inicial automatizado e encaminhamento para transbordo. Os testes indicaram bom desempenho do sistema, com tempo de resposta reduzido e adequada taxa de resolução automática no atendimento.

Os resultados indicam impacto positivo da automação, com redução do tempo de espera, maior disponibilidade do atendimento e liberação da equipe para casos mais complexos. O estudo realizado também permitiu concluir que desenvolver um chatbot desse tipo envolve desafios que geram retrabalho e exigem ajustes constantes, especialmente em um desenvolvimento individual.

Unir teoria e prática em um sistema real exigiu assumir múltiplos papéis, como projetar a arquitetura, integrar APIs, configurar o backend e testar fluxos conversacionais.

Durante o desenvolvimento, foi necessário revisar decisões técnicas e adaptar partes do sistema devido à complexidade das integrações, especialmente com a API da Meta, que passou por um processo de liberação demorado. Mesmo com esses desafios, os conhecimentos adquiridos ao longo do curso deram a base necessária para seguir com segurança.

Com isso, conclui-se que o objetivo geral foi alcançado. O protótipo mostrou capacidade de automatizar interações simples, reduzir o tempo de resposta e oferecer suporte inicial de forma eficiente. Além disso, o trabalho trouxe amadurecimento técnico e uma visão mais prática sobre como integrar APIs, sistemas distribuídos e modelos de linguagem em um contexto real de atendimento.

Para continuidade deste trabalho sugere-se as seguintes sugestões de trabalhos futuros:

- Implementar autenticação segura utilizando CPF, número de apólice ou identificadores internos;
- Aprimorar o mecanismo de transbordo, tornando-o mais preciso em situações que exigem intervenção manual;
- Desenvolver técnicas de personalização e recuperação de contexto para tornar as respostas mais consistentes em atendimentos prolongados.

8 REFERÊNCIAS

ALMALKI, Abdul Rahman; AZIZ, Mohd Yusof; WANG, Xiang. **A review of conversational AI: *chatbot* development and future directions**. *Information*, v. 12, n. 3, p. 110–130, 2021.

COLACE, Francesco et al. ***Chatbot* for e-learning: a case of study**. *Journal of e-Learning and Knowledge Society*, v. 18, n. 1, p. 10–20, 2022. Disponível em: *https*://www.je-lks.org/ojs/index.php/Je-LKS_EN/article/view/1136. Acesso em: 14 fev. 2025.

DALE, Robert. **The handbook of natural language processing**. 2. ed. Boca Raton: CRC Press, 2020.

FERREIRA, Juliana Costa et al. **O uso de *chatbot* como estratégia de atendimento de pós-venda no seguro de pessoas**. *Revista Gestão & Tecnologia*, Pedro Leopoldo, v. 21, n. 2, p. 211–238, abr./jun. 2021. Disponível em: *https*://revistagt.fpl.emnuvens.com.br/get/article/view/1883/1223. Acesso em: 17 fev. 2025.

GIL, Antônio Carlos. **Métodos e técnicas de pesquisa social**. 6. ed. São Paulo: Atlas, 2008.

GIL, Antônio Carlos. **Métodos e técnicas de pesquisa social**. 7. ed. São Paulo: Atlas, 2021.

JURAFSKY, Daniel; MARTIN, James H. **Speech, and language processing: an introduction to natural language processing, computational linguistics, and speech recognition**. 3. ed. Draft. Stanford: Pearson/Prentice Hall, 2025. Disponível em: *https*://web.stanford.edu/~jurafsky/slp3/. Acesso em: 20 fev. 2025.

LANDSTEINER, Norbert**. *Eliza (elizabot.js).*** 2005. Disponível em: *https*://www.masswerk.at/eliza/. Acesso em: 20 fev. 2025.

LOCATELLI, Flávia Pereira. **Inteligência artificial no poder judiciário: riscos, limites e garantias à proteção de dados**. *Revista Jurídica da Escola Superior do Ministério Público da União*, v. 7, n. 1, p. 1–17, 2023.

McTEAR, Michael F. **Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots**. Cham: Springer, 2020. Disponível em: https://direct.mit.edu/coli/article/49/1/257/113849/Conversational-AI-Dialogue-Systems-Conversational. Acesso em: 20 mar. 2025.

MEDURI, Sudeep. **Revolutionizing Customer Service: The Impact of *Large Language Model*s on *Chatbot* Performance**. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, v. 10, n. 5, p. 721-730, 2024.

NIELSEN, Jakob; MACK, Robert L. **Usability Inspection Methods**. New York: Wiley, 1994.

OECD – Organisation for Economic Co-operation and Development. **Recomendação do conselho de inteligência artificial**. Paris: OCDE, 2019. Disponível em: *https*://legalinstruments.oecd.org/en/instruments/oecd-legal-0449. Acesso em: 3 mar. 2025.

OPENAI. ***OpenAI API* documentation**. 2024. Disponível em: *https*://platform.*OpenAI*.com/docs. Acesso em: 6 mar. 2025.

ORACLE. **O que é um *chatbot*?** São Paulo: Oracle Brasil, 2023. Disponível em: *https*://www.oracle.com/br/*chatbots*/what-is-a-*chatbot*/. Acesso em: 8 mar. 2025.

RADFORD, Alec; NARASIMHAN, Karthik; SUTSKEVER, Ilya. ***Language models* are Few-Shot Learners**. 2020. Disponível em: *https*://arxiv.org/abs/2005.14165. Acesso em: 15 mar. 2025.

RAWAT, Shivani; SHARMA, Priya; VERMA, Rakesh. **Machine learning approaches for *chatbot* development: a comprehensive review**. *Journal of Applied AI Research*, v. 12, n. 2, p. 221–239, 2021.

RUSSELL, Stuart; NORVIG, Peter. **Artificial intelligence: a modern approach**. 4. ed. Upper Saddle River: Pearson, 2021. Disponível em: *https*://aima.cs.berkeley.edu/. Acesso em: 20 mar. 2025.

SHUM, Heung-Yeung; HE, Xiao-dong; LI, Di. **From Eliza to Xiaolce: challenges and opportunities with social *chatbots***. *ACM Transactions on Human-Robot Interaction*, v. 5, n. 2, p. 22–57, 2018.

SILVA, Fabiana Lopes da; CHAN, Betty Lilian; DECOSTER, Sonia Rosa Arbues. **Utilização de Inteligência Artificial no mercado segurador: uma abordagem baseada no nível de divulgação dos relatórios financeiros.** RedeCa *v.* 12, n.
6, p. 1–20, 2025. Disponível em: *https*://doi.org/10.23925/2446-9513.2025v12id69774. Acesso em: 27 jan. 2025.

SILVA, Wilian Toneli da; MARQUES, Heráclides Veloso; ROCHA, Verônica Alkimim; et al. **Proposta de implantação de uma central interna com *chatbot* integrado em uma cooperativa de crédito.** *Revista Caderno Pedagógico*, Curitiba, v. 21, n.12, p. 1–25, 2024. Disponível em: *https*://ojs.studiespublicacoes.com.br/ojs/index.php/cadped/article/view/10989. Acesso em: 27 jan. 2025.

VASWANI, Ashish et al. **Attention is all you need**. In: *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. Disponível em: *https*://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf. Acesso em: 5 abr. 2025.

WAZLAWICK, Raul Sidnei. **Metodologia da pesquisa para ciência da computação**. 2ª ed. Rio de Janeiro: Elsevier, 2014.

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS GABINETE DO REITOR
Av. Universitária, 1069 • Setor Universitário
Caixa Postal 86 • CEP 74605-010
Goiânia • Goiás • Brasil Fone: (62) 3946.1000 www.pucgoias.edu.br • reitoria@pucgoias.edu.br

RESOLUÇÃO n° 038/2020 – CEPE

ANEXO I

APÊNDICE ao TCC

Termo de autorização de publicação de produção acadêmica

O estudante Marco Aurélio Ayres Rocha de Oliveira do Curso de Engenharia de Computação, matrícula 2019200330023-1, telefone: 62 998628457, e-mail m.aurelio@msn.com, na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do Autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado **DESENVOLVIMENTO DE UM *CHATBOT* BASEADO EM INTELIGÊNCIA ARTIFICIAL**, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto(PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 09 de Dezembro de 2025.

gov.br Documento assinado digitalmente
MARCO AURELIO AYRES ROCHA DE OLIVEIRA
Data: 09/12/2025 10:22:04-0300
Verifique em https://validar.iti.gov.br

Assinatura do autor: _____

Nome completo do autor: Marco Aurélio Ayres Rocha de Oliveira_____

Assinatura do professor–orientador: ___ gov.br Documento assinado digitalmente
SOLANGE DA SILVA
Data: 09/12/2025 10:30:23-0300
Verifique em https://validar.iti.gov.br ___

Nome completo do professor-orientador: Solange da Silva _____