



UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS ARARANGUÁ

Lucas Vitor Linhares Larroide

**Aplicação de Inteligência Artificial em Sistemas Web com Foco em
Processamento de Linguagem Natural**

Araranguá

2026

Lucas Vitor Linhares Larroide

**Aplicação de Inteligência Artificial em Sistemas Web com Foco em
Processamento de Linguagem Natural**

Trabalho de Conclusão de Curso submetido ao curso Bacharelado em Tecnologias de Informação e Comunicação do Campus Araranguá da Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de Bacharel em Tecnologias da Informação e Comunicação.

Orientador: Prof. Juarez Bento da Silva

Araranguá

2026

Larroide, Lucas Vitor Linhares

Aplicação de Inteligência Artificial em Sistemas Web
com Foco em Processamento de Linguagem Natural / Lucas
Vitor Linhares Larroide ; orientador, Juarez Bento Da
silva, 2026.

50 p.

Trabalho de Conclusão de Curso (graduação) -
Universidade Federal de Santa Catarina, Campus Araranguá,
Graduação em Tecnologias da Informação e Comunicação,
Araranguá, 2026.

Inclui referências.

1. Tecnologias da Informação e Comunicação. 2.
Inteligência Artificial. 3. Sistemas Web. I. Da silva,
Juarez Bento. II. Universidade Federal de Santa Catarina.
Graduação em Tecnologias da Informação e Comunicação. III.
Título.

Lucas Vitor Linhares Larroide

Aplicação de Inteligência Artificial em Sistemas Web com Foco em Processamento de Linguagem Natural

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do título de Bacharel e aprovado em sua forma final pelo Curso Bacharelado em Tecnologias da Informação e Comunicação.

Local Araranguá, 22 de junho de 2026



Documento assinado digitalmente
Fernando Jose Spanhol
Data: 06/07/2026 10:16:18-0300
CPF: ***.656.419-**
Verifique as assinaturas em <https://v.ufsc.br>

Coordenação do Curso

Banca examinadora



Documento assinado digitalmente
Juarez Bento da Silva
Data: 06/07/2026 09:59:33-0300
CPF: ***.921.580-**
Verifique as assinaturas em <https://v.ufsc.br>

Prof.(a) Juarez Bento da Silva, Dr.(a)
Orientador(a)



Documento assinado digitalmente
Marina Carradore Sergio
Data: 06/07/2026 13:19:50-0300
CPF: ***.746.359-**
Verifique as assinaturas em <https://v.ufsc.br>

Prof.(a) Marina Carradore Sérgio, Dr.(a)
Instituição UFSC



Documento assinado digitalmente
Cristian Cechinel
Data: 02/07/2026 22:46:17-0300
CPF: ***.628.449-**
Verifique as assinaturas em <https://v.ufsc.br>

Prof.(a) Cristian Cechinel, Dr.(a)
Instituição UFSC

Araranguá, 2026.

AGRADECIMENTOS

À Universidade Federal de Santa Catarina, em especial ao Campus Araranguá e a Coordenadoria Especial Interdisciplinar de Tecnologias da Informação, pela excelência no ensino e pela oportunidade de desenvolver este trabalho.

Ao Professor Juarez Bento da Silva, meu orientador, pela inestimável orientação, paciência e incentivo ao longo de todo o processo de desenvolvimento deste Trabalho de Conclusão de Curso. Seus conhecimentos e sua dedicação foram fundamentais para a concretização desta pesquisa.

Aos meus pais, pelo apoio constante durante toda a minha trajetória acadêmica.

A todos que, direta ou indiretamente, contribuíram para a realização deste trabalho, meu sincero obrigado.

RESUMO

A Inteligência Artificial, especialmente por meio de modelos de Processamento de Linguagem Natural (PLN), tem ampliado as possibilidades de interação entre usuários e sistemas digitais. Apesar desses avanços, muitas aplicações conversacionais ainda apresentam limitações relacionadas à autenticação de usuários, ao armazenamento estruturado das interações e à recuperação do histórico de conversas. Diante desse contexto, este trabalho teve como objetivo desenvolver e validar uma aplicação web integrada a um modelo de Inteligência Artificial baseado em linguagem natural, incorporando mecanismos de autenticação, interação em tempo real e persistência de dados em banco relacional. A pesquisa adotou uma abordagem aplicada e exploratória, combinando revisão bibliográfica, planejamento arquitetônico, desenvolvimento incremental e testes funcionais. A aplicação foi estruturada em arquitetura modular composta por front-end, back-end, banco de dados relacional e modelo de linguagem. O front-end foi desenvolvido com HTML, CSS e JavaScript; o back-end foi implementado em Node.js utilizando o framework Fastify; o PostgreSQL foi empregado para o armazenamento de usuários e históricos de conversas; e o modelo DeepSeek-R1:8B foi executado localmente por meio da plataforma Ollama. A validação compreendeu 30 testes funcionais, abrangendo autenticação, comunicação entre módulos, persistência e recuperação de dados. Os resultados demonstraram 100% de sucesso na autenticação (30/30) e 100% de sucesso na recuperação do histórico, além da execução de 20 prompts sem falhas de comunicação entre a aplicação e o modelo de linguagem. O sistema apresentou latência média de resposta de 2,8 segundos, com tempo mínimo de 1,9 segundo e máximo de 4,1 segundos, não sendo registradas falhas críticas durante os testes realizados. Esses resultados evidenciam a viabilidade técnica da arquitetura proposta para integração entre aplicações web e modelos de Inteligência Artificial executados localmente. Como contribuição para a literatura, o trabalho apresenta e valida uma arquitetura modular que integra Processamento de Linguagem Natural, autenticação de usuários, persistência relacional de dados e interface web utilizando tecnologias de código aberto, oferecendo um modelo reproduzível para aplicações acadêmicas e experimentais. Conclui-se que a solução desenvolvida atingiu os objetivos propostos, demonstrando que tecnologias abertas e infraestrutura local podem ser empregadas na construção de sistemas inteligentes, personalizados e organizados. Como limitações, destacam-se a realização dos testes em ambiente controlado e a ausência de avaliação com uma amostra ampla de usuários, indicando a necessidade de estudos futuros envolvendo testes de usabilidade em larga escala, avaliação de desempenho sob diferentes cargas de trabalho e comparação entre modelos de linguagem distintos.

Palavras-chave: Inteligência Artificial. Processamento de Linguagem Natural. Aplicações Web. Autenticação de Usuários. Banco de Dados Relacional.

ABSTRACT

Artificial Intelligence, particularly through Natural Language Processing (NLP) models, has expanded the possibilities for interaction between users and digital systems. Despite these advances, many conversational applications still face limitations regarding user authentication, structured interaction storage, and conversation history retrieval. Against this backdrop, this study aimed to develop and validate a web application integrated with a natural language-based AI model, incorporating mechanisms for authentication, real-time interaction, and relational database data persistence. The research adopted an applied and exploratory approach, combining a literature review, architectural planning, incremental development, and functional testing. The application was structured using a modular architecture comprising a front-end, back-end, relational database, and language model. The front-end was developed using HTML, CSS, and JavaScript; the back-end was implemented in Node.js using the Fastify framework; PostgreSQL was employed to store user data and conversation histories; and the DeepSeek-R1:8B model was executed locally via the Ollama platform. Validation involved 30 functional tests covering authentication, inter-module communication, and data persistence and retrieval. Results showed a 100% success rate in authentication (30/30) and history retrieval, alongside the execution of 20 prompts without communication failures between the application and the language model. The system exhibited an average response latency of 2.8 seconds—ranging from a minimum of 1.9 seconds to a maximum of 4.1 seconds—with no critical failures recorded during testing. These results demonstrate the technical feasibility of the proposed architecture for integrating web applications with locally executed AI models. As a contribution to the literature, this work presents and validates a modular architecture that integrates Natural Language Processing, user authentication, relational data persistence, and a web interface using open-source technologies, offering a reproducible model for academic and experimental applications. It is concluded that the developed solution achieved the proposed objectives, demonstrating that open technologies and local infrastructure can be employed to build intelligent, personalized, and organized systems. Limitations include the fact that testing was conducted in a controlled environment and the absence of evaluation with a large user sample; this indicates a need for future studies involving large-scale usability testing, performance evaluation under varying workloads, and comparisons between different language models.

Keywords: Artificial Intelligence. Natural Language Processing. Web Applications. User Authentication. Relational Database.

LISTA DE FIGURAS

Figura 1 - Tela de autenticação da aplicação.....	59
Figura 2 - Interface de conversação da aplicação.....	60
Figura 3 - Recuperação do histórico armazenado no PostgreSQL	61

LISTA DE QUADROS

Quadro 1 – Matriz de Alinhamento Metodológico.....	42
Quadro 2 – Tecnologias utilizadas e justificativas de escolha.....	51

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
API	Application Programming Interface
BDR	Banco de Dados Relacional
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
IA	Inteligência Artificial
LLMs	Large Language Models (Modelos de Linguagem de Grande Escala)
NPM	Node Package Manager
PLN	Processamento de Linguagem Natural
RNNs	Redes Neurais Recorrentes
TCC	Trabalho de Conclusão de Curso
TIC	Tecnologias de Informação e Comunicação
UX	Experiência do Usuário
VS Code	Visual Studio Code
RLHF	Reinforcement Learning from Human Feedback
GPUs	Unidades de Processamento Gráfico
REST	Representational State Transfer
SGBD	Sistema Gerenciador de Banco de Dados
RAG	Retrieval-Augmented Generation

SUMÁRIO

1. INTRODUÇÃO	16
1.1 PROBLEMA DE PESQUISA	17
1.2 OBJETIVOS	17
1.2.1 Objetivo geral.....	18
1.2.1 Objetivos específicos.....	18
1.3 JUSTIFICATIVA	18
1.4 ESTUDOS CORRELATOS	19
2. FUNDAMENTAÇÃO TEÓRICA	21
2.1 INTELIGÊNCIA ARTIFICIAL: CONCEITOS E EVOLUÇÃO.....	21
2.2 PROCESSAMENTO DE LINGUAGEM NATURAL (PLN)	23
2.3 LARGE LANGUAGE MODELS (LLMs) E ARQUITETURA TRANSFORMER	25
2.4 APLICAÇÕES WEB E INTEGRAÇÃO COM IA.....	28
2.5 SEGURANÇA EM APLICAÇÕES CONVERSACIONAIS.....	30
2.6 RETRIEVAL-AUGMENTED GENERATION (RAG).....	32
2.7 BANCO DE DADOS RELACIONAIS	33
2.8 EXPERIÊNCIA DO USUÁRIO (UX) E DESIGN DE INTERAÇÃO.....	35
2.9 TRABALHOS RELACIONADOS	38
3. METODOLOGIA	41
3.1 CARACTERIZAÇÃO DA PESQUISA	43
3.2 DESIGN SCIENCE RESEARCH METHODOLOGY (DSRM)	44
3.3 IDENTIFICAÇÃO DO PROBLEMA E MOTIVAÇÃO	46
3.4 DEFINIÇÃO DOS OBJETIVOS DA SOLUÇÃO.....	47
3.5 PROJETO E DESENVOLVIMENTO DO ARTEFATO UTILIZADAS	49
3.5.1 Arquitetura da Solução	49
3.5.2 Tecnologias Utilizadas.....	50
3.5.3 Implementação do Back-end.....	52
3.5.1 Implementação do Front-end	54
3.5.5 Integração com o LLM e Arquitetura RAG	55
3.5.6 Banco de Dados e Persistência	57
3.6 DEMONSTRAÇÃO DA APLICAÇÃO	58
3.6.1 Autenticação do Usuário	58
3.6.2 Interação com o Chatbot	59
3.6.3 Recuperação do Histórico	60

3.7 AVALIAÇÃO DO ARTEFATO.....	61
3.7.1 Procedimentos de Avaliação.....	62
3.7.2 Resultados dos Testes Funcionais.....	62
3.7.3 Análise dos Resultados	63
4. RESULTADOS E DISCUSSÃO	64
4.1 AMBIENTE EXPERIMENTAL	64
4.2 RESULTADOS DA AUTENTICAÇÃO	66
4.3 RESULTADOS DA PERSISTÊNCIA DOS DADOS.....	67
4.4 RESULTADOS DA INTEGRAÇÃO COM O DEEPSEEK	68
6. CONSIDERAÇÕES FINAIS	71
REFERÊNCIAS.....	74

1. INTRODUÇÃO

O avanço recente dos modelos de Inteligência Artificial baseados em Processamento de Linguagem Natural (PLN) transformou a forma como usuários interagem com sistemas computacionais. Modelos de linguagem de grande porte LLMs, fundamentados na arquitetura *Transformer*, passaram a oferecer respostas contextualizadas em linguagem natural, possibilitando sua integração em aplicações web voltadas ao atendimento automatizado, à educação, ao desenvolvimento de software e ao suporte à decisão (VASWANI et al., 2017; RUSSELL; NORVIG, 2021). Entretanto, a incorporação desses modelos em aplicações próprias ainda demanda soluções relacionadas à autenticação de usuários, ao gerenciamento de sessões e à persistência das interações, aspectos essenciais para garantir personalização, segurança e continuidade das conversas.

O desenvolvimento de modelos de linguagem capazes de compreender e gerar texto em linguagem natural ampliou consideravelmente as possibilidades de interação entre humanos e máquinas, tornando a comunicação com sistemas digitais mais intuitiva e próxima da linguagem cotidiana (VASWANI et al., 2017). Esse avanço abre espaço para a criação de aplicações cada vez mais acessíveis, eficientes e adaptadas às necessidades dos usuários.

No entanto, para que essas soluções sejam realmente úteis em contextos acadêmicos, profissionais e sociais, é necessário integrá-las a sistemas web robustos, que permitam autenticação de usuários, registro de interações e organização dos dados gerados. Essa integração favorece não apenas a personalização da experiência do usuário, mas também o armazenamento estruturado das informações, viabilizando análises futuras e ampliando as possibilidades de uso (GOODFELLOW; BENGIO; COURVILLE, 2016).

O foco central desta pesquisa reside no desenvolvimento e na análise técnica de uma aplicação web integrada a modelos de Inteligência Artificial baseados em Processamento de Linguagem Natural (PLN). A solução proposta prioriza a autenticação de usuários, a interação em tempo real e a persistência estruturada de diálogos em bancos de dados relacionais. Ao situar-se na interseção entre sistemas web, modelos de linguagem, persistência de dados e experiência do usuário (UX), o estudo investiga como esses componentes podem ser articulados para compor uma solução funcional, escalável e acessível.

1.1 PROBLEMA DE PESQUISA

Embora a Inteligência Artificial tenha avançado significativamente notadamente com os modelos de linguagem de larga escala, sua integração em ecossistemas web complexos ainda impõe desafios técnicos e metodológicos (RUSSELL; NORVIG, 2021; VASWANI et al., 2017). Frequentemente, aplicações conversacionais priorizam a latência e a fluidez das respostas em detrimento de camadas robustas de autenticação, armazenamento individualizado de sessões e recuperação estruturada de históricos, o que limita a continuidade e a organização dos dados gerados (McTEAR, 2020; ZHOU et al., 2020).

Tal cenário revela uma lacuna crítica: a necessidade de integrar, de forma coesa, a interação em linguagem natural, a persistência relacional de dados, a personalização da jornada e interfaces centradas no usuário (DATE, 2019; NIELSEN, 2012; PRESSMAN; MAXIM, 2021). A ausência dessa integração confina o uso da IA a interações efêmeras, prejudicando o acompanhamento histórico, a evolução das conversas e a análise analítica posterior das informações (McTEAR, 2020; ZHOU et al., 2020).

Com base nesse panorama, define-se a seguinte questão norteadora: de que maneira é possível desenvolver e validar uma aplicação web que integre Inteligência Artificial e Processamento de Linguagem Natural a mecanismos de autenticação e persistência de dados, garantindo uma experiência de uso organizada, personalizada e acessível?

1.2 OBJETIVOS

A definição dos objetivos deste trabalho orienta o desenvolvimento da aplicação proposta e estabelece os parâmetros para sua validação técnica e funcional. Considerando o problema de pesquisa apresentado, os objetivos foram estruturados de modo a contemplar tanto a construção da solução tecnológica quanto a verificação de sua aplicabilidade em um ambiente web integrado a recursos de Inteligência Artificial e Processamento de Linguagem Natural.

1.2.1 Objetivo geral

Desenvolver uma aplicação web integrada a modelos de Inteligência Artificial baseados em linguagem natural, incorporando mecanismos de autenticação, interação em tempo real e persistência estruturada em banco de dados relacional, a fim de atestar sua viabilidade técnica e sua contribuição para uma experiência de usuário organizada, personalizada e funcional

1.2.1 Objetivos específicos

Para o alcance do objetivo geral, foram definidos objetivos específicos que contemplam as principais etapas técnicas e metodológicas do projeto, desde a estruturação da aplicação até a validação de seus componentes. Esses objetivos orientam o desenvolvimento do sistema e permitem verificar, de forma sistemática, se a solução proposta atende aos requisitos funcionais, de segurança, usabilidade e persistência de dados.

São objetivos específicos deste trabalho:

- OE.1. Implementar um sistema de autenticação e controle de acesso, assegurando a individualização das sessões, a segurança dos dados e o vínculo íntegro entre o usuário e seu histórico de interações.
- OE.2. Integrar modelos de Processamento de Linguagem Natural à interface web, viabilizando o processamento de prompts e a geração de respostas coerentes em tempo real.
- OE.3. Projetar e estruturar uma camada de persistência em banco de dados relacional para o armazenamento e a recuperação eficiente do histórico de conversas, garantindo a integridade das informações.
- OE.4. Desenvolver uma interface web responsiva e intuitiva, fundamentada em princípios de UX/UI, para otimizar a interação entre o usuário e o modelo de IA.

1.3 JUSTIFICATIVA

A escolha deste tema justifica-se pela relevância crescente da Inteligência Artificial como ferramenta de apoio em diferentes áreas do conhecimento e pela necessidade de explorar sua integração em sistemas web acessíveis e funcionais. O

uso de modelos de linguagem natural tem se consolidado como uma das tecnologias mais promissoras para tornar a interação homem-máquina mais intuitiva e eficaz (VASWANI *et al.*, 2017; RUSSELL; NORVIG, 2021).

Além disso, a implementação de mecanismos de autenticação e armazenamento estruturado de interações apresenta grande importância prática, tanto para garantir a segurança e a personalização da experiência do usuário quanto para possibilitar análises posteriores baseadas em dados organizados (DATE, 2019). A aplicação proposta, ao unir esses elementos, contribui não apenas para o avanço acadêmico, mas também para potenciais aplicações em contextos profissionais, como educação, atendimento automatizado e gestão de informações (ZHOU *et al.*, 2020).

Outro fator que reforça a justificativa é a oportunidade de desenvolvimento científico e formativo proporcionada ao estudante. A realização deste projeto permite o contato direto com tecnologias emergentes e metodologias de desenvolvimento de software, fortalecendo competências técnicas e acadêmicas, além de abrir caminho para a continuidade da pesquisa em trabalhos futuros e aplicações derivadas (MCTEAR, 2020).

O trabalho adota uma abordagem aplicada, combinando revisão bibliográfica com desenvolvimento incremental de software. As etapas incluem análise de requisitos, implementação de funcionalidades principais (login, interação com IA e histórico de conversas), integração com banco de dados relacional e testes de usabilidade. Essa estrutura metodológica visa assegurar tanto a fundamentação teórica quanto a validação prática dos resultados obtidos (PRESSMAN; MAXIM, 2021).

1.4 ESTUDOS CORRELATOS

Diversos estudos têm explorado a aplicação da Inteligência Artificial em sistemas voltados para interação com usuários, especialmente por meio de chatbots e assistentes virtuais. McTear (2020) destaca que os sistemas conversacionais vêm sendo amplamente utilizados em áreas como atendimento ao cliente e suporte técnico, mas muitas vezes limitam-se a respostas em tempo real, sem mecanismos avançados de armazenamento ou análise de dados.

No campo do processamento de linguagem natural, Vaswani *et al.* (2017) apresentaram a arquitetura Transformer, que revolucionou a forma como os modelos

compreendem e geram texto, tornando viável o desenvolvimento de aplicações mais sofisticadas e capazes de lidar com interações complexas. Complementando, Russell e Norvig (2021) discutem a evolução da IA em diferentes áreas e reforçam sua relevância na criação de sistemas mais inteligentes e adaptativos.

Outros trabalhos concentram-se na integração da IA a contextos específicos. Zhou *et al.* (2020), por exemplo, analisaram o uso de chatbots para atendimento ao cliente, ressaltando ganhos em eficiência e escalabilidade, mas também apontando limitações relacionadas à personalização da experiência do usuário. Já Goodfellow, Bengio e Courville (2016) apresentam fundamentos do aprendizado profundo que possibilitam a expansão da IA em aplicações diversas, incluindo aquelas voltadas a sistemas web.

Dessa forma, observa-se que os trabalhos existentes oferecem importantes contribuições para o avanço da área, mas ainda deixam lacunas no que se refere à combinação entre processamento de linguagem natural, persistência de dados e usabilidade em aplicações web. É nesse espaço que o presente projeto se insere, buscando integrar esses elementos em uma solução prática e funcional.

2. FUNDAMENTAÇÃO TEÓRICA

A fundamentação teórica constitui a base conceitual deste trabalho, reunindo os principais referenciais científicos relacionados ao desenvolvimento de aplicações web integradas a modelos de Inteligência Artificial (IA) baseados em Processamento de Linguagem Natural (PLN). A crescente adoção de modelos de linguagem de grande porte (*Large Language Models* – LLMs) tem impulsionado o desenvolvimento de sistemas conversacionais capazes de compreender e gerar textos em linguagem natural, ampliando suas aplicações em diferentes áreas, como educação, atendimento automatizado, desenvolvimento de software e suporte à tomada de decisão.

Entretanto, a implementação desses modelos em aplicações reais envolve desafios que vão além da geração de respostas em linguagem natural. Aspectos como arquitetura de software, integração entre front-end e back-end, autenticação de usuários, persistência de dados, gerenciamento de sessões, segurança da informação e experiência do usuário tornam-se fatores determinantes para a construção de sistemas robustos, escaláveis e confiáveis.

Dessa forma, este capítulo apresenta uma revisão da literatura sobre os principais conceitos que fundamentam a pesquisa. Inicialmente são discutidos os avanços da Inteligência Artificial e do Processamento de Linguagem Natural, com ênfase na evolução dos modelos de linguagem baseados na arquitetura *Transformer*. Em seguida, são abordados aspectos relacionados à integração de modelos de IA em aplicações web, segurança de aplicações conversacionais, persistência de dados em bancos relacionais e princípios de usabilidade e experiência do usuário. Por fim, são apresentados trabalhos correlatos, permitindo identificar as contribuições já existentes na literatura e evidenciar a relevância da arquitetura proposta neste trabalho.

2.1 INTELIGÊNCIA ARTIFICIAL: CONCEITOS E EVOLUÇÃO

A Inteligência Artificial (IA) constitui um dos principais campos da Ciência da Computação, dedicando-se ao desenvolvimento de sistemas capazes de executar tarefas que, tradicionalmente, requerem inteligência humana, como aprendizagem, raciocínio, percepção, planejamento e tomada de decisão. Embora as primeiras pesquisas na área tenham surgido na década de 1950, os avanços recentes em capacidade computacional, disponibilidade de grandes volumes de dados e algoritmos

de aprendizado profundo impulsionaram uma rápida expansão da IA, tornando-a presente em diversos setores da sociedade.

Segundo Russell e Norvig (2021), a Inteligência Artificial pode ser compreendida como o estudo de agentes inteligentes, isto é, sistemas capazes de perceber o ambiente em que estão inseridos e agir de maneira autônoma para alcançar objetivos específicos. Essa definição amplia a compreensão da IA para além da simples automação de tarefas, enfatizando a capacidade de adaptação e tomada de decisão baseada em informações disponíveis.

A evolução da IA pode ser dividida em diferentes fases. Inicialmente, predominavam sistemas baseados em regras (*rule-based systems*), nos quais o conhecimento era explicitamente codificado por especialistas. Apesar de apresentarem bons resultados em domínios específicos, esses sistemas possuíam baixa capacidade de adaptação a novos cenários e elevado custo de manutenção, tornando-se limitados diante de problemas complexos e dinâmicos.

Posteriormente, o desenvolvimento de técnicas de Aprendizado de Máquina (*Machine Learning*) permitiu que os sistemas passassem a identificar padrões diretamente a partir dos dados, reduzindo a dependência de regras previamente definidas. Mais recentemente, o Aprendizado Profundo (*Deep Learning*), fundamentado em redes neurais artificiais com múltiplas camadas, ampliou significativamente a capacidade dos modelos de resolver problemas relacionados ao reconhecimento de imagens, processamento de voz e compreensão de linguagem natural (GOODFELLOW; BENGIO; COURVILLE, 2016).

Nos últimos anos, a Inteligência Artificial Generativa consolidou-se como uma das áreas de maior crescimento dentro da IA. Diferentemente dos modelos voltados exclusivamente para classificação ou previsão, os modelos generativos são capazes de produzir novos conteúdos, como textos, imagens, códigos-fonte, áudio e vídeos, a partir de padrões aprendidos durante o treinamento. Esse avanço possibilitou o desenvolvimento de sistemas conversacionais cada vez mais sofisticados, ampliando suas aplicações em contextos acadêmicos, empresariais e industriais.

Nesse cenário, destacam-se os Modelos de Linguagem de Grande Escala (*Large Language Models* – LLMs), treinados com grandes volumes de dados textuais e capazes de compreender contexto, responder perguntas, gerar conteúdo e auxiliar na resolução de problemas complexos. Modelos como GPT, Llama, DeepSeek e Qwen representam essa nova geração de sistemas baseados em Inteligência

Artificial, demonstrando elevado desempenho em tarefas de Processamento de Linguagem Natural.

Apesar dos avanços alcançados, a adoção desses modelos em aplicações reais ainda apresenta desafios importantes. Questões relacionadas à privacidade dos dados, segurança das informações, gerenciamento de contexto, consumo de recursos computacionais, confiabilidade das respostas e integração com aplicações web continuam sendo temas amplamente discutidos na literatura recente. Dessa forma, o desenvolvimento de arquiteturas capazes de integrar modelos de linguagem a sistemas web de forma segura, modular e eficiente constitui uma área de pesquisa em constante evolução.

No contexto deste trabalho, a Inteligência Artificial é empregada como núcleo responsável pelo processamento das interações em linguagem natural entre o usuário e a aplicação. Entretanto, diferentemente de soluções baseadas exclusivamente em serviços externos, a proposta desenvolvida utiliza o modelo DeepSeek-R1:8B executado localmente por meio da plataforma Ollama. Essa abordagem proporciona maior controle sobre os dados processados, reduz a dependência de provedores externos e possibilita a construção de aplicações conversacionais alinhadas às necessidades de ambientes acadêmicos e institucionais.

2.2 PROCESSAMENTO DE LINGUAGEM NATURAL (PLN)

O Processamento de Linguagem Natural (PLN) constitui uma área interdisciplinar da Inteligência Artificial, da Ciência da Computação e da Linguística Computacional, dedicada ao desenvolvimento de métodos que permitam aos sistemas computacionais compreender, interpretar, processar e gerar linguagem humana de forma automática. Seu principal objetivo consiste em reduzir a distância entre a comunicação humana e a capacidade computacional, tornando possível que usuários interajam com sistemas utilizando linguagem natural em vez de comandos estruturados.

Segundo Jurafsky e Martin (2025), o PLN engloba um amplo conjunto de técnicas destinadas à análise e geração de texto, incluindo tarefas como classificação textual, tradução automática, reconhecimento de entidades nomeadas, análise de sentimentos, sumarização de documentos e sistemas de perguntas e respostas. Essas aplicações vêm sendo incorporadas em diferentes contextos, como assistentes

virtuais, plataformas educacionais, atendimento automatizado, recuperação de informações e desenvolvimento de software.

Historicamente, as primeiras abordagens em PLN baseavam-se em regras gramaticais e linguísticas definidas manualmente por especialistas. Embora eficientes em domínios restritos, esses sistemas apresentavam dificuldades para lidar com ambiguidades, variações linguísticas e contextos complexos. Posteriormente, métodos estatísticos passaram a utilizar grandes conjuntos de textos para estimar probabilidades de ocorrência das palavras, proporcionando maior flexibilidade e melhor desempenho em diversas tarefas linguísticas.

O avanço do Aprendizado de Máquina e, posteriormente, do Aprendizado Profundo representou uma mudança significativa nesse cenário. Redes neurais artificiais passaram a aprender representações semânticas diretamente dos dados, reduzindo a necessidade de engenharia manual de características (feature engineering) e aumentando a capacidade de generalização dos modelos. Como consequência, tarefas que anteriormente dependiam de regras específicas passaram a ser resolvidas por modelos treinados em grandes volumes de textos.

Essa evolução atingiu um novo patamar com a introdução da arquitetura Transformer, proposta por Vaswani et al. (2017), que revolucionou o Processamento de Linguagem Natural ao empregar mecanismos de atenção (self-attention) para identificar relações entre palavras independentemente da distância existente entre elas. Essa abordagem permitiu maior paralelização durante o treinamento e melhor compreensão do contexto das sentenças, tornando-se a base dos atuais Modelos de Linguagem de Grande Escala (Large Language Models – LLMs).

Os modelos contemporâneos de PLN não se limitam à identificação de padrões linguísticos, sendo capazes de manter contexto durante diálogos, responder perguntas complexas, produzir textos coerentes, gerar código-fonte e auxiliar em atividades de tomada de decisão. Essa capacidade tornou os LLMs elementos centrais na construção de sistemas conversacionais modernos, ampliando significativamente as possibilidades de interação entre usuários e aplicações computacionais.

Entretanto, a utilização desses modelos em aplicações práticas também impõe desafios relevantes. Aspectos como gerenciamento de contexto, persistência das conversas, tempo de resposta, consumo de recursos computacionais, segurança das informações e confiabilidade das respostas são fatores determinantes para o

sucesso de sistemas conversacionais baseados em Inteligência Artificial. Dessa forma, o desenvolvimento de aplicações que integrem modelos de linguagem de maneira eficiente exige não apenas modelos robustos, mas também arquiteturas de software capazes de garantir comunicação segura, armazenamento adequado das interações e uma experiência de uso satisfatória.

No contexto deste trabalho, o Processamento de Linguagem Natural constitui o mecanismo responsável pela interpretação dos prompts enviados pelos usuários e pela geração das respostas apresentadas na interface da aplicação. A utilização do modelo DeepSeek-R1:8B executado localmente por meio da plataforma Ollama demonstra a viabilidade da incorporação de modelos modernos de PLN em aplicações web desenvolvidas com tecnologias de código aberto, preservando maior controle sobre os dados processados e reduzindo a dependência de serviços externos.

2.3 LARGE LANGUAGE MODELS (LLMS) E ARQUITETURA TRANSFORMER

Os LLMs representam um dos avanços mais significativos da Inteligência Artificial contemporânea. Esses modelos são redes neurais treinadas com bilhões de palavras provenientes de livros, artigos científicos, páginas da Web, repositórios de código e outras fontes textuais, permitindo-lhes aprender padrões linguísticos, relações semânticas e estruturas sintáticas de maneira generalizada. Como resultado, tornam-se capazes de executar diversas tarefas utilizando um único modelo, como responder perguntas, resumir documentos, traduzir textos, gerar código-fonte e produzir conteúdo em linguagem natural.

O desempenho dos LLMs tornou-se possível principalmente devido à arquitetura *Transformer*, proposta por Vaswani et al. (2017). Diferentemente das arquiteturas recorrentes RNNs e das redes de memória de longo curto prazo LSTM, amplamente utilizadas em aplicações de Processamento de Linguagem Natural até então, o *Transformer* elimina a necessidade de processar palavras sequencialmente. Em seu lugar, emprega mecanismos de atenção (*self-attention*), permitindo que todas as palavras de uma sequência sejam analisadas simultaneamente e que o modelo identifique relações contextuais entre termos independentemente da distância existente entre eles.

O mecanismo de *self-attention* constitui o principal diferencial dessa arquitetura. Durante o processamento de uma sequência textual, o modelo atribui

diferentes pesos às palavras presentes no contexto, identificando quais elementos são mais relevantes para interpretar o significado de cada termo. Esse processo possibilita representar dependências semânticas complexas e compreender relações contextuais que seriam difíceis de capturar por modelos sequenciais tradicionais.

Além do mecanismo de atenção, a arquitetura *Transformer* caracteriza-se por sua elevada capacidade de paralelização durante o treinamento. Como todas as palavras podem ser processadas simultaneamente, o tempo necessário para treinamento é significativamente reduzido quando comparado às arquiteturas recorrentes. Essa característica, associada ao aumento da capacidade computacional proporcionada pelas Unidades de Processamento Gráfico (GPUs) e aos grandes conjuntos de dados disponíveis atualmente, tornou viável o desenvolvimento de modelos contendo bilhões de parâmetros.

Os LLMs modernos são normalmente treinados em duas etapas principais. Na primeira, denominada pré-treinamento (*pre-training*), o modelo aprende padrões gerais da linguagem a partir de grandes volumes de textos, sem necessidade de supervisão específica. Na segunda etapa, conhecida como ajuste fino (*fine-tuning*), o modelo pode ser especializado para tarefas ou domínios particulares utilizando conjuntos de dados menores e mais específicos. Em alguns casos, também são empregadas técnicas como o RLHF, que utilizam avaliações humanas para alinhar o comportamento do modelo às expectativas dos usuários.

Nos últimos anos, diversos LLMs foram disponibilizados tanto por organizações comerciais quanto pela comunidade de código aberto. Modelos como GPT, Llama, DeepSeek, Qwen, Gemma e Mistral apresentam elevado desempenho em tarefas de compreensão e geração de linguagem natural, diferindo principalmente em aspectos como quantidade de parâmetros, estratégia de treinamento, requisitos computacionais e formas de disponibilização. O crescimento do ecossistema de modelos abertos ampliou significativamente as possibilidades de pesquisa e desenvolvimento, permitindo que instituições acadêmicas e organizações implementem soluções baseadas em Inteligência Artificial sem depender exclusivamente de serviços em nuvem.

Apesar dos avanços alcançados, os LLMs ainda apresentam limitações importantes. Como seu funcionamento baseia-se na previsão probabilística da próxima palavra mais provável dentro de um determinado contexto, esses modelos podem produzir informações incorretas, fenômeno frequentemente denominado

hallucination. Além disso, questões relacionadas ao consumo de recursos computacionais, privacidade dos dados, viés algorítmico, explicabilidade das respostas e limitação da janela de contexto continuam sendo desafios amplamente discutidos na literatura científica.

Uma alternativa para reduzir parte dessas limitações consiste na execução local dos modelos de linguagem. Plataformas como o Ollama permitem executar LLMs diretamente na infraestrutura do usuário, dispensando o envio de informações para serviços externos e proporcionando maior controle sobre privacidade, segurança e gerenciamento dos dados processados. Embora essa abordagem exija recursos computacionais adequados, ela representa uma solução particularmente interessante para ambientes acadêmicos, institucionais e corporativos que demandam confidencialidade das informações.

No contexto desta pesquisa, foi adotado o modelo DeepSeek-R1:8B, executado localmente por meio da plataforma Ollama. A escolha desse modelo fundamentou-se no equilíbrio entre capacidade de geração textual, requisitos computacionais e viabilidade de integração com aplicações web desenvolvidas em Node.js. Essa abordagem permitiu construir uma arquitetura modular capaz de integrar autenticação de usuários, persistência relacional das conversas e processamento de linguagem natural em um ambiente local, reduzindo a dependência de serviços externos e ampliando o controle sobre os dados manipulados pela aplicação.

A literatura recente demonstra que os LLMs representam uma mudança de paradigma no desenvolvimento de sistemas inteligentes. Entretanto, seu desempenho em aplicações reais depende não apenas da qualidade do modelo utilizado, mas também da arquitetura de software responsável por sua integração, gerenciamento de contexto, persistência das interações, segurança das informações e experiência do usuário. Nesse sentido, o presente trabalho contribui ao investigar uma arquitetura que integra esses elementos em uma aplicação web baseada em tecnologias de código aberto, aproximando os avanços da Inteligência Artificial Generativa das necessidades práticas de desenvolvimento de software.

2.4 APLICAÇÕES WEB E INTEGRAÇÃO COM IA

A crescente adoção de LLMs impulsionou o desenvolvimento de aplicações web capazes de oferecer interações em linguagem natural em diferentes contextos, como atendimento automatizado, suporte técnico, educação, desenvolvimento de software e análise de informações. Entretanto, a incorporação desses modelos em sistemas computacionais envolve desafios que vão além da capacidade de geração textual, exigindo arquiteturas de software capazes de integrar componentes responsáveis pela interface com o usuário, processamento das requisições, execução do modelo de IA, persistência de dados e mecanismos de segurança.

Arquiteturas modernas para aplicações baseadas em IA normalmente adotam uma organização em camadas, separando as responsabilidades entre front-end, back-end, banco de dados e serviços responsáveis pela inferência do modelo. Segundo Pressman e Maxim (2021), a modularização favorece a manutenção, escalabilidade e reutilização de componentes, reduzindo o acoplamento entre as diferentes partes do sistema. Essa abordagem permite que alterações em um módulo possam ser realizadas com impacto mínimo sobre os demais componentes da aplicação.

Nesse contexto, a comunicação entre os módulos geralmente ocorre por meio de APIs baseadas em arquitetura REST, possibilitando que o front-end envie solicitações ao back-end, responsável por realizar validações, controlar autenticação, acessar o banco de dados e encaminhar as requisições ao modelo de IA. Essa separação proporciona maior flexibilidade para substituição de modelos de linguagem, atualização da infraestrutura e integração com diferentes tecnologias, preservando a estabilidade da aplicação.

Outro aspecto amplamente discutido na literatura refere-se ao gerenciamento do contexto das conversas. Diferentemente de aplicações tradicionais, sistemas conversacionais precisam manter informações sobre interações anteriores para produzir respostas coerentes ao longo de uma sessão. Esse gerenciamento exige mecanismos de persistência capazes de armazenar históricos de conversas, identificar usuários individualmente e recuperar informações relevantes sempre que novas requisições forem realizadas.

Além da persistência dos dados, a integração de IA em aplicações web também demanda atenção aos requisitos não funcionais do sistema. Desempenho,

escalabilidade, disponibilidade, segurança, confiabilidade e tempo de resposta influenciam diretamente a experiência do usuário e a qualidade percebida da aplicação. A literatura recente destaca que arquiteturas desacopladas facilitam a distribuição da carga de processamento entre diferentes componentes, permitindo maior eficiência na utilização dos recursos computacionais e simplificando futuras expansões da infraestrutura.

Outro fator relevante corresponde à escolha entre execução local e utilização de serviços em nuvem para hospedagem dos modelos de IA. Plataformas baseadas em serviços externos oferecem elevada capacidade computacional e atualizações constantes dos modelos, porém podem introduzir limitações relacionadas à privacidade dos dados, dependência de conectividade, custos operacionais e restrições impostas pelos provedores. Em contrapartida, soluções executadas localmente proporcionam maior controle sobre a infraestrutura, confidencialidade das informações e autonomia para personalização do ambiente de execução, embora exijam recursos computacionais compatíveis com o modelo utilizado.

A literatura evidencia que não existe uma arquitetura única capaz de atender a todos os cenários de utilização de IA em aplicações web. A escolha da solução depende de fatores como volume de usuários, requisitos de desempenho, necessidade de privacidade, custos de infraestrutura e facilidade de manutenção. Dessa forma, arquiteturas modulares e desacopladas têm sido amplamente adotadas por oferecerem maior flexibilidade para evolução do sistema e integração com novas tecnologias.

No presente trabalho, a integração entre IA e aplicação web foi implementada por meio de uma arquitetura composta por front-end desenvolvido em HTML, CSS e JavaScript, back-end implementado em Node.js utilizando o framework Fastify, banco de dados PostgreSQL para armazenamento das informações e um LLM executado localmente por meio da plataforma Ollama. A comunicação entre esses componentes ocorre por meio de APIs REST, permitindo o processamento das solicitações dos usuários, autenticação das sessões, armazenamento do histórico de conversas e geração de respostas em linguagem natural. Essa arquitetura demonstra a viabilidade da utilização de tecnologias de código aberto para construção de aplicações conversacionais capazes de integrar IA, persistência de dados e autenticação em um ambiente único, modular e de fácil manutenção.

A partir da revisão da literatura, observa-se que grande parte dos estudos concentra-se no desenvolvimento e na avaliação dos LLMs, enquanto aspectos relacionados à sua integração em aplicações completas recebem menor atenção. Nesse contexto, a arquitetura proposta neste trabalho contribui ao demonstrar uma implementação prática baseada em tecnologias abertas, conciliando processamento de linguagem natural, autenticação de usuários, persistência relacional das conversas e execução local do modelo de IA em uma única solução.

2.5 SEGURANÇA EM APLICAÇÕES CONVERSACIONAIS

A segurança constitui um dos principais requisitos não funcionais no desenvolvimento de aplicações web, especialmente em sistemas que processam informações pessoais ou utilizam IA para interagir com os usuários. Em aplicações conversacionais, além dos desafios tradicionais relacionados à proteção de dados e ao controle de acesso, torna-se necessário garantir a confidencialidade das interações, a integridade das informações armazenadas e a autenticidade dos usuários que utilizam o sistema.

Segundo a OWASP (2025), aplicações web permanecem entre os principais alvos de ataques cibernéticos devido à ampla exposição de seus serviços e à diversidade de vulnerabilidades que podem surgir durante o desenvolvimento. Problemas como falhas de autenticação, gerenciamento inadequado de sessões, controle insuficiente de acesso e exposição indevida de informações figuram entre os riscos mais recorrentes, podendo comprometer tanto os dados armazenados quanto o funcionamento da aplicação.

Em sistemas conversacionais baseados em IA, esses desafios tornam-se ainda mais relevantes. O histórico das conversas frequentemente contém informações sensíveis, dados pessoais ou conteúdos relacionados às atividades dos usuários. Dessa forma, mecanismos de autenticação e autorização desempenham papel fundamental para garantir que cada usuário tenha acesso exclusivamente às suas próprias informações, preservando a privacidade e evitando acessos indevidos.

Outro aspecto importante refere-se ao gerenciamento das sessões de usuário. Após a autenticação, a aplicação deve manter uma identificação segura durante toda a interação, permitindo que as requisições sejam corretamente associadas ao usuário autenticado. Segundo o NIST (2025), mecanismos seguros de

gerenciamento de identidade e sessão reduzem significativamente os riscos de sequestro de sessão (*session hijacking*), reutilização indevida de credenciais e acessos não autorizados.

Além do controle de acesso, a proteção das informações armazenadas constitui outro requisito essencial. A utilização de bancos de dados relacionais permite organizar usuários, conversas e demais informações de forma estruturada, possibilitando a implementação de restrições de integridade, controle de permissões e mecanismos de auditoria. A adoção de práticas como armazenamento seguro de credenciais, utilização de consultas parametrizadas e validação de entradas também contribui para reduzir vulnerabilidades conhecidas, como injeção de SQL e manipulação indevida de dados.

A crescente utilização de LLMs introduz ainda desafios específicos relacionados à segurança da IA. Ataques conhecidos como *prompt injection* procuram manipular as instruções fornecidas ao modelo para induzir comportamentos não previstos, enquanto tentativas de extração de informações podem explorar respostas geradas pelo sistema para obter dados confidenciais. Embora essas ameaças ainda sejam objeto de intensa pesquisa, recomenda-se que aplicações conversacionais implementem mecanismos de validação das entradas, restrições de acesso às informações armazenadas e monitoramento das interações realizadas pelos usuários.

Outro fator relevante refere-se à privacidade dos dados processados pelos modelos de IA. Soluções baseadas em serviços externos normalmente exigem o envio das conversas para servidores mantidos por terceiros, podendo gerar preocupações relacionadas à confidencialidade das informações e à conformidade com legislações de proteção de dados. Em contrapartida, a execução local dos modelos reduz a exposição dos dados durante o processamento e oferece maior controle sobre a infraestrutura utilizada, aspecto particularmente importante em ambientes acadêmicos, corporativos e institucionais.

No contexto deste trabalho, a segurança foi incorporada à arquitetura da aplicação por meio de mecanismos de autenticação de usuários, controle individual das sessões e persistência segregada dos históricos de conversas. Cada interação realizada é associada ao respectivo usuário autenticado, impedindo o compartilhamento indevido das informações entre diferentes contas. Além disso, a execução local do LLM por meio da plataforma Ollama elimina a necessidade de

transmissão das conversas para serviços externos durante o processamento, contribuindo para maior confidencialidade dos dados manipulados pela aplicação.

A literatura evidencia que a segurança em aplicações conversacionais não depende exclusivamente da robustez do modelo de IA, mas da integração entre mecanismos de autenticação, controle de acesso, proteção dos dados armazenados e boas práticas de desenvolvimento seguro. Nesse sentido, a arquitetura proposta neste trabalho busca alinhar esses elementos, demonstrando que aplicações baseadas em IA podem ser desenvolvidas com foco simultâneo em funcionalidade, privacidade e segurança da informação.

2.6 RETRIEVAL-AUGMENTED GENERATION (RAG)

O RAG constitui uma arquitetura que combina LLMs com mecanismos de recuperação de informações provenientes de bases de dados ou repositórios externos. Diferentemente dos modelos tradicionais, que geram respostas exclusivamente com base no conhecimento adquirido durante o treinamento, sistemas baseados em RAG complementam esse conhecimento consultando informações atualizadas durante a execução da aplicação.

Segundo Lewis et al. (2020), a arquitetura RAG é composta por duas etapas principais. A primeira corresponde à recuperação (*retrieval*), na qual documentos, registros ou outros dados relevantes são obtidos de uma fonte externa. Em seguida, essas informações são incorporadas ao contexto enviado ao LLM, responsável pela etapa de geração (*generation*), produzindo respostas fundamentadas tanto no conhecimento do modelo quanto nas informações recuperadas.

Essa abordagem apresenta vantagens importantes em relação aos modelos convencionais. Como as respostas podem ser construídas utilizando informações atualizadas e específicas do domínio da aplicação, reduz-se a ocorrência de respostas incorretas (*hallucinations*), além de eliminar a necessidade de novo treinamento do modelo sempre que a base de conhecimento é modificada. Dessa forma, sistemas RAG têm sido amplamente empregados em assistentes corporativos, sistemas de atendimento, plataformas educacionais e aplicações que dependem de informações frequentemente atualizadas.

Outro benefício do RAG consiste na separação entre o conhecimento do modelo e os dados da aplicação. Enquanto o LLM permanece responsável pela

compreensão da linguagem natural e geração das respostas, as informações específicas são mantidas em bases de dados independentes, permitindo atualização contínua sem alterar os parâmetros do modelo. Essa característica favorece a manutenção da aplicação, melhora a escalabilidade da arquitetura e amplia a confiabilidade das respostas produzidas.

A implementação de sistemas RAG também exige mecanismos eficientes de recuperação das informações. Dependendo da aplicação, essa recuperação pode ocorrer por meio de bancos de dados relacionais, bancos vetoriais ou mecanismos de busca especializados. Em aplicações estruturadas, como sistemas de gestão ou plataformas acadêmicas, bancos de dados relacionais podem desempenhar esse papel ao fornecer registros organizados que posteriormente são incorporados ao contexto enviado ao LLM.

No presente trabalho, a arquitetura desenvolvida utiliza o PostgreSQL como repositório das informações da aplicação, permitindo que o sistema realize consultas aos dados armazenados e utilize os resultados durante o processamento das solicitações dos usuários. Essa integração entre banco de dados, back-end e LLM caracteriza uma arquitetura baseada nos princípios do RAG, na qual o modelo DeepSeek-R1:8B complementa seu conhecimento utilizando informações recuperadas dinamicamente da base de dados da aplicação.

A literatura recente demonstra que arquiteturas RAG representam uma das principais tendências no desenvolvimento de aplicações conversacionais, pois conciliam a capacidade de compreensão dos LLMs com informações específicas e atualizadas do domínio da aplicação. Nesse contexto, a solução proposta neste trabalho integra recuperação de dados, autenticação de usuários, persistência das conversas e geração de respostas em linguagem natural, demonstrando a viabilidade da utilização de tecnologias abertas para construção de sistemas conversacionais inteligentes.

2.7 BANCO DE DADOS RELACIONAIS

O armazenamento estruturado das informações constitui um componente fundamental no desenvolvimento de aplicações web modernas. Em sistemas conversacionais baseados em IA, a persistência dos dados desempenha papel ainda mais relevante, pois permite registrar usuários, armazenar históricos de conversas,

recuperar interações anteriores e manter o contexto das sessões ao longo do tempo. Dessa forma, o banco de dados deixa de atuar apenas como um mecanismo de armazenamento, tornando-se parte integrante da arquitetura responsável pela continuidade da experiência do usuário.

Segundo Elmasri e Navathe (2017), um banco de dados consiste em um conjunto organizado de dados relacionados entre si, estruturado de forma a permitir armazenamento, consulta, atualização e recuperação eficiente das informações. Entre os diferentes modelos existentes, o modelo relacional permanece como um dos mais utilizados no desenvolvimento de aplicações corporativas devido à sua consistência, integridade e capacidade de representar relações entre diferentes entidades.

Os Sistemas Gerenciadores de Banco de Dados Relacionais (SGBDRs) organizam as informações em tabelas interligadas por chaves primárias e estrangeiras, permitindo reduzir redundâncias, preservar a integridade referencial e facilitar consultas complexas por meio da Linguagem de Consulta Estruturada (SQL). Essas características tornam o modelo relacional particularmente adequado para aplicações que exigem controle de usuários, autenticação, rastreabilidade das operações e armazenamento estruturado de grandes volumes de informações.

No contexto de aplicações conversacionais, a persistência das conversas representa um requisito funcional importante. Embora os LLMs sejam capazes de considerar o contexto presente em uma mesma interação, eles não mantêm memória permanente entre diferentes sessões. Assim, a responsabilidade pela preservação do histórico recai sobre a própria aplicação, que deve registrar as mensagens trocadas, associá-las aos respectivos usuários e disponibilizá-las sempre que necessário para continuidade das interações.

A literatura recente destaca que a persistência do histórico proporciona benefícios que vão além da simples recuperação de mensagens anteriores. O armazenamento estruturado das conversas possibilita auditoria das interações, análise de utilização do sistema, geração de relatórios, personalização da experiência do usuário e desenvolvimento de funcionalidades futuras baseadas no histórico de uso. Além disso, contribui para reduzir inconsistências durante sessões prolongadas, permitindo que informações relevantes sejam recuperadas sempre que necessário.

Outro aspecto relevante refere-se ao desempenho e à escalabilidade das operações de armazenamento. À medida que aplicações conversacionais passam a

atender um número crescente de usuários, torna-se necessário adotar estruturas capazes de realizar consultas rápidas, garantir consistência dos dados e suportar múltiplas operações simultâneas. Nesse contexto, os SGBDRs continuam sendo amplamente empregados devido à sua maturidade tecnológica, mecanismos de controle de concorrência e suporte a transações, características essenciais para aplicações críticas.

Entre as soluções disponíveis, o PostgreSQL destaca-se como um dos SGBDRs de código aberto mais utilizados em aplicações modernas. Sua conformidade com os princípios do modelo relacional, aliada ao suporte a índices, transações ACID, mecanismos avançados de segurança e elevada capacidade de escalabilidade, torna-o uma alternativa adequada para aplicações que demandam confiabilidade, desempenho e integridade dos dados.

No presente trabalho, o PostgreSQL foi utilizado para armazenar informações de autenticação, usuários e históricos de conversas gerados durante a utilização da aplicação. Cada mensagem enviada ao sistema é registrada juntamente com os dados necessários para identificação do usuário e recuperação posterior das interações. Essa estratégia permite manter o histórico individualizado das conversas, garantindo organização das informações, continuidade das sessões e suporte à personalização da experiência de uso.

A literatura demonstra que aplicações conversacionais modernas dependem da integração entre modelos de IA e mecanismos robustos de persistência de dados. Enquanto o LLM é responsável pelo processamento da linguagem natural e geração das respostas, o banco de dados assegura a continuidade das interações, a rastreabilidade das informações e o gerenciamento eficiente do contexto das conversas. Nesse sentido, a arquitetura desenvolvida neste trabalho integra essas duas camadas de maneira complementar, conciliando processamento inteligente da linguagem com armazenamento estruturado das informações em ambiente relacional.

2.8 EXPERIÊNCIA DO USUÁRIO (UX) E DESIGN DE INTERAÇÃO

A Experiência do Usuário (UX) corresponde ao conjunto de percepções, emoções e respostas resultantes da interação entre um usuário e um sistema computacional. Em aplicações conversacionais baseadas em IA, a qualidade da experiência não depende exclusivamente da capacidade do LLM em gerar respostas

coerentes, mas também de fatores relacionados à usabilidade, desempenho, organização das informações, continuidade das conversas e percepção de confiança durante a interação.

Segundo a ISO 9241-210 (2019), a UX abrange todas as percepções e respostas do usuário decorrentes do uso ou da expectativa de uso de um produto, sistema ou serviço. Essa definição evidencia que aspectos técnicos, funcionais e subjetivos influenciam diretamente a aceitação de uma aplicação, tornando a experiência do usuário um requisito essencial no desenvolvimento de sistemas interativos.

Nielsen (2020) destaca que aplicações com boa usabilidade devem apresentar facilidade de aprendizado, eficiência de utilização, consistência na interface, prevenção de erros e facilidade de recuperação quando falhas ocorrem. Embora esses princípios tenham sido originalmente propostos para interfaces tradicionais, continuam sendo amplamente aplicáveis ao desenvolvimento de aplicações conversacionais, nas quais a simplicidade da interação influencia diretamente a satisfação do usuário.

No contexto de sistemas baseados em IA, estudos recentes apontam que a confiança representa um dos fatores mais importantes para a adoção dessas tecnologias. Usuários tendem a aceitar melhor respostas produzidas por modelos inteligentes quando conseguem compreender o funcionamento geral da aplicação, identificar claramente suas funcionalidades e perceber consistência nas respostas fornecidas ao longo das interações. Da mesma forma, tempos elevados de processamento, respostas inconsistentes ou perda do contexto das conversas podem comprometer significativamente a qualidade da experiência.

Outro aspecto amplamente discutido na literatura refere-se à continuidade das interações. Diferentemente de aplicações convencionais, sistemas conversacionais frequentemente envolvem diálogos compostos por múltiplas mensagens relacionadas entre si. Assim, a possibilidade de armazenar e recuperar o histórico das conversas contribui para que o usuário mantenha o contexto da interação, retome discussões anteriores e acompanhe a evolução das informações trocadas com o sistema. Essa característica favorece maior organização das atividades e reduz a necessidade de repetição de comandos ou perguntas já realizadas.

Além da continuidade das conversas, a organização visual da interface desempenha papel importante na experiência do usuário. Interfaces limpas,

responsivas e intuitivas reduzem a carga cognitiva durante a utilização da aplicação, facilitando a localização das funcionalidades disponíveis e tornando a interação mais eficiente. Em aplicações conversacionais, elementos como identificação clara das mensagens, separação visual entre usuário e sistema, histórico organizado e navegação simplificada contribuem para uma utilização mais natural.

Outro fator relevante corresponde ao desempenho percebido pelo usuário. Ainda que modelos de IA demandem processamento computacional significativo, a literatura demonstra que tempos de resposta reduzidos favorecem a sensação de fluidez durante a interação. Dessa forma, a otimização da comunicação entre front-end, back-end, banco de dados e LLM exerce influência direta sobre a percepção de qualidade da aplicação, mesmo quando a capacidade de geração textual do modelo permanece inalterada.

No presente trabalho, a experiência do usuário foi considerada desde a definição da arquitetura da aplicação. A interface foi desenvolvida com foco na simplicidade de utilização, permitindo autenticação individual, acesso ao histórico de conversas, envio de mensagens em linguagem natural e visualização organizada das respostas geradas pelo LLM. A persistência das interações no banco de dados possibilita a continuidade das sessões, enquanto a arquitetura modular contribui para reduzir atrasos na comunicação entre os componentes do sistema.

Embora a presente pesquisa tenha priorizado a validação funcional da aplicação, a literatura evidencia que avaliações futuras podem empregar métodos específicos de mensuração da UX, como questionários padronizados, testes de usabilidade, escalas de satisfação e métricas de eficiência, eficácia e percepção de qualidade. A incorporação dessas técnicas permitirá avaliar de forma mais abrangente o impacto da arquitetura proposta sobre a experiência dos usuários.

Dessa forma, observa-se que a UX em aplicações conversacionais resulta da integração entre diversos fatores técnicos e humanos. A qualidade do LLM constitui apenas um dos elementos envolvidos, sendo igualmente importantes aspectos relacionados à arquitetura da aplicação, persistência das conversas, desempenho, segurança e organização da interface. Nesse contexto, a solução desenvolvida neste trabalho procura integrar esses elementos para oferecer uma interação mais eficiente, organizada e adequada às necessidades dos usuários.

2.9 TRABALHOS RELACIONADOS

O avanço dos LLMs impulsionou o desenvolvimento de diversas soluções voltadas à construção de sistemas conversacionais inteligentes. Paralelamente à evolução dos modelos de linguagem, surgiram plataformas e arquiteturas destinadas à integração desses modelos com aplicações web, mecanismos de persistência de dados, autenticação de usuários e recuperação de informações por meio de arquiteturas RAG. A análise desses trabalhos permite compreender o estado da arte da área e identificar as principais contribuições e limitações das soluções atualmente disponíveis.

Entre as plataformas mais difundidas destaca-se o ChatGPT, desenvolvido pela OpenAI, que popularizou a utilização de LLMs em aplicações conversacionais. A plataforma demonstra elevado desempenho em tarefas de compreensão e geração de linguagem natural, oferecendo respostas contextualizadas para diferentes domínios do conhecimento. Entretanto, sua utilização depende predominantemente de infraestrutura em nuvem, o que pode limitar a personalização da arquitetura, o controle sobre os dados processados e a integração direta com bases de dados específicas da aplicação.

Outra solução amplamente utilizada é o Ollama, plataforma destinada à execução local de LLMs. Seu principal objetivo consiste em simplificar a instalação, gerenciamento e utilização de modelos de linguagem em computadores pessoais ou servidores locais. A execução local reduz a dependência de serviços externos e amplia o controle sobre privacidade e segurança das informações processadas. Contudo, o Ollama não fornece, de forma nativa, funcionalidades como autenticação de usuários, gerenciamento de sessões, persistência de conversas ou interfaces completas para aplicações web, exigindo integração com outros componentes de software.

No contexto das interfaces para modelos locais, destaca-se o Open WebUI, uma plataforma de código aberto que disponibiliza uma interface gráfica para interação com diferentes LLMs executados localmente. A ferramenta oferece funcionalidades como gerenciamento de conversas, organização de sessões e suporte à integração com múltiplos modelos. Entretanto, sua arquitetura é voltada ao uso geral, apresentando limitações quanto à personalização da lógica de negócio, integração com bancos de dados específicos e desenvolvimento de funcionalidades adaptadas às necessidades de aplicações acadêmicas ou corporativas.

A literatura recente também apresenta diversas pesquisas relacionadas à arquitetura RAG, destacando sua importância na redução de respostas incorretas e na utilização de informações atualizadas durante a geração das respostas pelos LLMs. Esses estudos demonstram que a combinação entre mecanismos de recuperação de informações e modelos de linguagem melhora a qualidade das respostas em aplicações que dependem de bases de conhecimento específicas. Entretanto, muitos trabalhos concentram-se na utilização de bancos vetoriais e mecanismos de recuperação semântica, dedicando menor atenção à integração com bancos de dados relacionais utilizados em aplicações web convencionais.

Observa-se, portanto, que as soluções existentes geralmente abordam apenas parte dos requisitos necessários para o desenvolvimento de uma aplicação conversacional completa. Enquanto algumas plataformas concentram-se na execução eficiente dos LLMs, outras priorizam interfaces gráficas ou mecanismos de recuperação de informações. Poucos trabalhos integram, em uma única arquitetura, autenticação de usuários, persistência relacional das conversas, recuperação de dados, execução local do modelo de IA e comunicação entre os diferentes componentes da aplicação.

Nesse contexto, o presente trabalho propõe uma arquitetura modular baseada em tecnologias de código aberto que integra um LLM executado localmente por meio do Ollama, autenticação de usuários, banco de dados PostgreSQL, comunicação via APIs REST e mecanismos de recuperação de informações fundamentados nos princípios do RAG. Essa integração permite construir uma aplicação conversacional capaz de combinar processamento de linguagem natural, armazenamento estruturado dos dados e recuperação dinâmica de informações, preservando maior controle sobre privacidade, manutenção e evolução da aplicação.

A Tabela 1 apresenta uma comparação entre as principais características das soluções analisadas e a arquitetura proposta neste trabalho.

Característica	ChatGPT	Ollama	Open WebUI	Arquitetura proposta
Execução local do LLM	Não	Sim	Sim	Sim
Interface web própria	Sim	Não	Sim	Sim

Autenticação de usuários	Limitada ao serviço	Não	Sim	Sim
Persistência em PostgreSQL	Não	Não	Parcial	Sim
Integração com banco relacional	Limitada	Não	Parcial	Sim
Arquitetura RAG	Limitada	Não	Parcial	Sim
Código aberto	Não	Sim	Sim	Sim
Arquitetura modular	Parcial	Parcial	Parcial	Sim

A análise comparativa evidencia que a principal contribuição deste trabalho não consiste no desenvolvimento de um novo LLM, mas na proposição e validação de uma arquitetura integrada para aplicações conversacionais. Ao reunir execução local de IA, autenticação, persistência de dados relacionais, recuperação de informações e comunicação modular em uma única solução, a pesquisa amplia as possibilidades de utilização de tecnologias abertas em ambientes acadêmicos e institucionais, oferecendo uma arquitetura reproduzível para futuros desenvolvimentos.

3. METODOLOGIA

Este capítulo apresenta os procedimentos metodológicos adotados para o desenvolvimento da pesquisa, descrevendo a abordagem científica utilizada, as etapas de concepção e desenvolvimento do artefato, os métodos empregados para sua demonstração e avaliação, bem como as tecnologias utilizadas na implementação da solução proposta.

Considerando que o objetivo deste trabalho consiste no desenvolvimento e validação de uma aplicação web integrada a um modelo de Inteligência Artificial, a pesquisa foi conduzida com base na **Design Science Research Methodology (DSRM)**, proposta por Peffers et al. (2007). Essa metodologia é amplamente utilizada em pesquisas da área de Computação e Sistemas de Informação por orientar a criação de artefatos tecnológicos destinados à solução de problemas identificados em contextos reais.

Diferentemente de metodologias voltadas exclusivamente à investigação de fenômenos, a DSRM concentra-se no desenvolvimento, demonstração e avaliação de artefatos, como modelos, métodos, arquiteturas e sistemas computacionais. Sua aplicação permite que o pesquisador identifique um problema, proponha uma solução, implemente o artefato e avalie sua eficácia em relação aos objetivos estabelecidos, garantindo rigor científico durante todo o processo de desenvolvimento.

Neste trabalho, a metodologia foi empregada para orientar a construção de uma arquitetura modular capaz de integrar autenticação de usuários, comunicação entre aplicação web e modelo de linguagem, recuperação de informações por meio de uma abordagem baseada em **RAG**, persistência de dados em banco relacional e geração de respostas em linguagem natural utilizando um **LLM** executado localmente. A avaliação do artefato foi realizada por meio de testes funcionais que verificaram o correto funcionamento dos principais componentes da aplicação.

Com o objetivo de evidenciar a coerência entre os objetivos específicos, as etapas de desenvolvimento, os procedimentos técnicos e os critérios de validação adotados, apresenta-se, no Quadro 1, a Matriz de Alinhamento Metodológico da pesquisa. Esse instrumento permite demonstrar a rastreabilidade entre o problema investigado, a implementação da aplicação web e a análise dos resultados obtidos.

Quadro 1 – Matriz de Alinhamento Metodológico

Objetivo específico	Etapas metodológicas correspondente	Procedimentos adotados	Ferramentas/tecnologias utilizadas	Crterios de validao	Evidncias esperadas
Implementar um sistema de autenticao e controle de acesso, assegurando a individualizao das sessoes, a segurana dos dados e o vnculo entre o usurio e seu histrico de interaes.	Levantamento de requisitos, modelagem da autenticao e implementao do back-end.	Definio dos requisitos de acesso; criao da lgica de login; associao do usurio s sessoes; armazenamento do identificador de usurio; controle de acesso s conversas individuais.	Node.js, Fastify, JavaScript, cookies de sesso, PostgreSQL.	O sistema deve permitir o acesso apenas mediante credenciais vlidas; cada usurio deve visualizar somente seu prprio histrico; as sessoes devem ser mantidas de forma individualizada.	Tela de login funcional; registros de usurios no banco de dados; sessoes associadas corretamente aos histricos de conversas.
Integrar modelos de Processamento de Linguagem Natural interface web, viabilizando o processamento de prompts e a gerao de respostas coerentes em tempo real.	Implementao da comunicao entre front-end, servidor e modelo de IA.	Configurao do modelo DeepSeek-R1:8B via Ollama; criao de rotas de comunicao; envio dos prompts ao modelo; recebimento e exibio das respostas na interface.	DeepSeek-R1:8B, Ollama, Node.js, Fastify, Axios, JavaScript.	O sistema deve enviar corretamente os prompts ao modelo de IA; as respostas devem retornar interface em tempo adequado e com coerncia textual; a comunicao entre os mdulos deve ocorrer sem falhas crticas.	Interaes bem-sucedidas entre usurio e IA; respostas exibidas na interface; logs ou testes funcionais demonstrando a comunicao entre aplicao e modelo.
Projetar e estruturar uma camada de persistncia em banco de dados relacional para o armazenamento e a recuperao eficiente do histrico de conversas.	Modelagem do banco de dados e implementao da persistncia das interaes.	Definio das entidades do banco; criao das tabelas de usurios e conversas; implementao das operaes de gravao, consulta e recuperao dos histricos; vinculao das conversas	PostgreSQL, biblioteca pg, Node.js, SQL.	As mensagens enviadas e recebidas devem ser registradas corretamente; o histrico deve ser recuperado sem perda de dados; cada registro deve manter vnculo com o usurio	Tabelas relacionais estruturadas; histricos armazenados no banco; recuperao das conversas anteriores na interface da aplicao.

		aos usuários autenticados.		correspondent e.	
Desenvolver uma interface web responsiva e intuitiva, fundamentada em princípios de UX/UI, para otimizar a interação entre o usuário e o modelo de IA.	Desenvolvimento do front-end e organização visual da aplicação.	Construção da tela de login; criação da área de conversação; exibição do histórico de interações; aplicação de layout responsivo; adoção de design minimalista e modo escuro; formatação das mensagens geradas.	HTML, CSS, JavaScript, princípios de UX/UI.	A interface deve ser clara, legível e funcional; os elementos principais devem ser facilmente identificados; a aplicação deve permitir navegação simples e interação contínua com o modelo de IA.	Tela de login; dashboard da aplicação; área de mensagens; histórico de conversas; capturas de tela apresentadas nas figuras do trabalho.
Executar protocolos de testes funcionais, de integração e de usabilidade para validar se os componentes do sistema cumprem os requisitos técnicos e as expectativas de desempenho estabelecidas.	Testes, validação e análise dos resultados.	Realização de testes de login; verificação da comunicação entre front-end, back-end e IA; testes de persistência e recuperação dos dados; avaliação da exibição das respostas; observação da estabilidade do sistema durante múltiplas interações.	Ambiente local de testes, navegador web, Node.js, PostgreSQL, Ollama, DeepSeek-R1:8B.	As funcionalidades principais devem operar de forma integrada; não devem ocorrer falhas críticas de autenticação, comunicação ou armazenamento; o sistema deve apresentar comportamento estável durante o uso.	Registro dos testes realizados; funcionamento integrado dos módulos; resultados descritos na seção de resultados e discussão; identificação de limitações e possibilidades de melhoria.

Fonte: Elaborado pelo autor (2026).

3.1 CARACTERIZAÇÃO DA PESQUISA

A presente pesquisa caracteriza-se como aplicada, pois teve como finalidade desenvolver uma solução tecnológica para um problema prático relacionado à integração entre aplicações web e modelos de Inteligência Artificial. Segundo Gil (2022), a pesquisa aplicada busca produzir conhecimentos direcionados à solução de problemas específicos, visando sua utilização em situações reais.

Quanto aos objetivos, a pesquisa possui caráter exploratório, uma vez que investiga tecnologias recentes relacionadas à IA, LLMs e arquiteturas RAG, buscando

compreender sua aplicação no desenvolvimento de sistemas conversacionais integrados a aplicações web. De acordo com Gil (2022), pesquisas exploratórias têm como propósito proporcionar maior familiaridade com um problema, contribuindo para o aprimoramento de ideias, identificação de soluções e construção de novos conhecimentos.

Em relação à abordagem do problema, a pesquisa é classificada como predominantemente qualitativa, por concentrar-se no projeto, desenvolvimento e análise do artefato computacional proposto. Entretanto, a avaliação da solução também incorporou elementos quantitativos, por meio da análise de métricas obtidas durante os testes funcionais, incluindo taxa de sucesso das funcionalidades implementadas, número de testes realizados e tempo de resposta da aplicação. Essa combinação permitiu avaliar tanto o funcionamento da arquitetura quanto seu desempenho operacional.

Como procedimento metodológico, adotou-se a Design Science Research Methodology (DSRM), proposta por Peffers et al. (2007), por tratar-se de uma metodologia voltada ao desenvolvimento e avaliação de artefatos tecnológicos. A DSRM orienta o processo de pesquisa desde a identificação do problema até a demonstração e avaliação da solução desenvolvida, proporcionando rigor metodológico na construção de sistemas computacionais.

No contexto deste trabalho, o artefato desenvolvido consiste em uma aplicação web modular capaz de integrar autenticação de usuários, recuperação de informações por meio de uma arquitetura baseada em RAG, persistência de dados em PostgreSQL e processamento de linguagem natural utilizando o modelo DeepSeek-R1:8B executado localmente por meio da plataforma Ollama. A utilização da DSRM permitiu estruturar sistematicamente todas as etapas do desenvolvimento, garantindo que a solução proposta fosse projetada, implementada, demonstrada e avaliada de acordo com um método científico consolidado na área de Computação.

3.2 DESIGN SCIENCE RESEARCH METHODOLOGY (DSRM)

A Design Science Research Methodology (DSRM), proposta por Peffers et al. (2007), constitui uma metodologia de pesquisa voltada ao desenvolvimento de artefatos capazes de solucionar problemas identificados em contextos reais. Diferentemente de abordagens que buscam apenas descrever ou explicar fenômenos, a DSRM concentra-se na concepção, implementação, demonstração e avaliação de

soluções tecnológicas, sendo amplamente utilizada em pesquisas nas áreas de Computação, Sistemas de Informação e Engenharia de Software.

A metodologia é composta por seis etapas principais: (i) identificação do problema e motivação; (ii) definição dos objetivos da solução; (iii) projeto e desenvolvimento do artefato; (iv) demonstração; (v) avaliação; e (vi) comunicação dos resultados. Essas etapas não representam necessariamente um processo estritamente linear, podendo ser revisitadas ao longo da pesquisa conforme novas necessidades sejam identificadas durante o desenvolvimento do artefato.

A primeira etapa corresponde à identificação do problema e motivação, na qual são analisadas as dificuldades existentes e justificada a necessidade de desenvolvimento de uma nova solução. No contexto deste trabalho, observou-se que muitas aplicações conversacionais concentram-se apenas na interação com modelos de IA, apresentando limitações relacionadas à autenticação de usuários, persistência das conversas, recuperação de informações e integração entre os diferentes componentes da aplicação.

Na segunda etapa, denominada definição dos objetivos da solução, estabeleceram-se os requisitos que deveriam ser atendidos pelo artefato desenvolvido. Como objetivo principal, definiu-se a construção de uma aplicação web capaz de integrar autenticação de usuários, comunicação com um LLM executado localmente, recuperação de informações utilizando uma abordagem baseada em RAG e armazenamento persistente das interações em um banco de dados relacional.

A terceira etapa consiste no projeto e desenvolvimento do artefato, envolvendo a definição da arquitetura da aplicação, seleção das tecnologias utilizadas e implementação dos componentes do sistema. Para isso, foi adotada uma arquitetura modular composta por interface web, servidor de aplicação, banco de dados PostgreSQL e modelo DeepSeek-R1:8B executado localmente por meio da plataforma Ollama, comunicando-se por APIs REST.

A quarta etapa corresponde à demonstração do artefato, na qual a aplicação desenvolvida foi utilizada para executar os principais casos de uso previstos, incluindo autenticação de usuários, recuperação de informações, processamento de solicitações em linguagem natural, armazenamento das conversas e apresentação das respostas ao usuário.

Na quinta etapa, denominada avaliação, foram realizados testes funcionais para verificar o atendimento aos requisitos definidos durante o planejamento da

pesquisa. A avaliação contemplou a autenticação dos usuários, persistência do histórico das conversas, comunicação entre os módulos da aplicação e desempenho do sistema durante o processamento das solicitações, permitindo validar a viabilidade técnica da arquitetura proposta.

Por fim, a sexta etapa refere-se à comunicação dos resultados, materializada neste Trabalho de Conclusão de Curso, no qual são apresentados os fundamentos teóricos, os procedimentos metodológicos, a arquitetura desenvolvida, os resultados obtidos durante a avaliação e as contribuições da pesquisa para o desenvolvimento de aplicações conversacionais baseadas em IA.

3.3 IDENTIFICAÇÃO DO PROBLEMA E MOTIVAÇÃO

A primeira etapa da DSRM consiste na identificação do problema que motiva a pesquisa e na justificativa para o desenvolvimento de um artefato capaz de solucioná-lo. Segundo Peffers et al. (2007), essa etapa é fundamental para delimitar o contexto da investigação e demonstrar a relevância da solução proposta.

A evolução dos LLMs ampliou significativamente as possibilidades de desenvolvimento de sistemas conversacionais capazes de compreender e gerar linguagem natural. Entretanto, a incorporação desses modelos em aplicações web ainda apresenta desafios relacionados à integração entre os diferentes componentes da arquitetura, incluindo autenticação de usuários, persistência de dados, recuperação de informações e comunicação eficiente entre a aplicação e o modelo de IA.

Embora existam plataformas consolidadas para utilização de LLMs, muitas delas dependem de infraestrutura em nuvem, apresentam limitações quanto à personalização da aplicação ou não disponibilizam mecanismos integrados para gerenciamento de usuários e armazenamento estruturado das interações. Da mesma forma, ferramentas destinadas à execução local de modelos de linguagem concentram-se principalmente na inferência dos modelos, exigindo o desenvolvimento de componentes adicionais para atender aos requisitos de aplicações completas.

Outro desafio identificado refere-se ao gerenciamento das informações utilizadas durante as interações. Em aplicações conversacionais, a capacidade de recuperar dados armazenados e incorporá-los ao contexto da resposta torna-se um fator importante para ampliar a utilidade do sistema. Nesse cenário, arquiteturas baseadas em RAG possibilitam integrar o conhecimento do LLM com informações

provenientes de bases de dados externas, permitindo que as respostas sejam fundamentadas em dados atualizados e específicos do domínio da aplicação.

Diante desse contexto, identificou-se a necessidade de desenvolver uma solução que integrasse, em uma única arquitetura, autenticação de usuários, persistência das conversas em banco de dados relacional, recuperação de informações por meio de uma abordagem baseada em RAG e processamento de linguagem natural utilizando um LLM executado localmente. Essa integração busca oferecer maior controle sobre os dados processados, reduzir a dependência de serviços externos e disponibilizar uma arquitetura modular capaz de atender às necessidades de aplicações acadêmicas e institucionais.

A motivação para o desenvolvimento deste trabalho fundamenta-se, portanto, na proposição de uma arquitetura que reúna essas funcionalidades de forma integrada, empregando tecnologias de código aberto e priorizando aspectos como modularidade, segurança, persistência das informações e facilidade de manutenção. A partir da identificação dessas necessidades, foram definidos os objetivos da solução, apresentados na etapa seguinte da DSRM.

3.4 DEFINIÇÃO DOS OBJETIVOS DA SOLUÇÃO

Após a identificação do problema e da necessidade de integração entre aplicações web e modelos de IA, a segunda etapa da DSRM consistiu na definição dos objetivos que orientariam o desenvolvimento do artefato. Conforme Peffers et al. (2007), essa etapa tem como finalidade estabelecer os requisitos que a solução deve atender para responder ao problema identificado e servir como referência para as etapas posteriores de projeto, desenvolvimento e avaliação.

O objetivo principal da solução foi desenvolver uma aplicação web capaz de integrar autenticação de usuários, processamento de linguagem natural por meio de um LLM executado localmente, recuperação de informações utilizando uma arquitetura baseada em RAG e persistência das interações em um banco de dados relacional. Além de atender aos requisitos funcionais da aplicação, buscou-se desenvolver uma arquitetura modular que favorecesse a manutenção, a escalabilidade e a reutilização de seus componentes.

Para atingir esse objetivo, foram estabelecidos os seguintes requisitos específicos:

- desenvolver uma interface web intuitiva para autenticação e interação com o sistema;
- implementar um servidor de aplicação responsável pelo gerenciamento das requisições e comunicação entre os componentes da arquitetura;
- integrar um LLM executado localmente por meio da plataforma Ollama para processamento das solicitações em linguagem natural;
- implementar mecanismos de recuperação de informações armazenadas no PostgreSQL para composição do contexto enviado ao modelo, seguindo os princípios da arquitetura RAG;
- armazenar usuários, sessões e históricos de conversas em um banco de dados relacional, garantindo persistência e recuperação das informações;
- assegurar que cada usuário tenha acesso apenas às suas próprias informações por meio de mecanismos de autenticação e controle de acesso;
- validar o funcionamento da aplicação por meio de testes funcionais e métricas de desempenho.

Além dos requisitos funcionais, também foram considerados requisitos não funcionais relacionados ao desempenho, modularidade, segurança, facilidade de manutenção e utilização de tecnologias de código aberto. Esses requisitos orientaram a escolha da arquitetura da aplicação e das ferramentas empregadas durante o desenvolvimento, buscando garantir que a solução fosse tecnicamente viável e passível de evolução futura.

A definição desses objetivos estabeleceu as diretrizes para o desenvolvimento do artefato proposto, servindo como base para as decisões arquiteturais e tecnológicas apresentadas na etapa seguinte da DSRM, correspondente ao projeto e desenvolvimento da aplicação.

Código	Requisito	Tipo
RF01	Autenticar usuários	Funcional
RF02	Permitir envio de prompts	Funcional
RF03	Consultar dados do PostgreSQL para compor o contexto (RAG)	Funcional
RF04	Armazenar histórico de conversas	Funcional
RF05	Recuperar conversas anteriores	Funcional

Código	Requisito	Tipo
RF06	Executar o LLM localmente	Funcional
RNF02	Arquitetura modular	Não funcional
RNF03	Utilizar tecnologias de código aberto	Não funcional
RNF04	Garantir segurança e controle de acesso	Não funcional

3.5 PROJETO E DESENVOLVIMENTO DO ARTEFATO UTILIZADAS

3.5.1 Arquitetura da Solução

A etapa de projeto e desenvolvimento do artefato teve como objetivo definir uma arquitetura capaz de integrar os diferentes componentes necessários ao funcionamento da aplicação conversacional proposta. Para isso, foi adotada uma arquitetura modular, organizada em camadas independentes, permitindo a separação das responsabilidades entre interface do usuário, processamento das requisições, recuperação de informações, armazenamento de dados e processamento por IA.

A arquitetura desenvolvida é composta por cinco componentes principais: interface web, servidor de aplicação, banco de dados relacional, mecanismo de recuperação de informações e modelo de linguagem executado localmente. Essa organização favorece a manutenção do sistema, reduz o acoplamento entre os módulos e facilita futuras evoluções da aplicação.

A interface web foi desenvolvida utilizando HTML, CSS e JavaScript, sendo responsável pela interação entre o usuário e a aplicação. Por meio dessa interface, o usuário realiza a autenticação, envia solicitações em linguagem natural, visualiza o histórico das conversas e recebe as respostas geradas pelo sistema.

O processamento das requisições é realizado por um servidor desenvolvido em Node.js utilizando o framework Fastify. Esse componente atua como intermediário entre a interface web, o banco de dados PostgreSQL e o modelo de IA, sendo responsável pelo gerenciamento das requisições HTTP, autenticação dos usuários, recuperação das informações necessárias e encaminhamento dos dados ao modelo de linguagem.

O PostgreSQL foi empregado como sistema gerenciador de banco de dados relacional, armazenando informações de autenticação, usuários, histórico das conversas e demais dados necessários ao funcionamento da aplicação. Além da persistência das informações, o banco de dados também desempenha papel importante na recuperação de dados utilizados para compor o contexto enviado ao modelo de IA.

A integração entre a aplicação e o modelo DeepSeek-R1:8B foi realizada por meio da plataforma Ollama, responsável pela execução local do LLM. Antes do envio da solicitação ao modelo, o servidor consulta as informações relevantes armazenadas no PostgreSQL, incorporando esses dados ao contexto da requisição. Essa estratégia segue os princípios da arquitetura RAG, permitindo que as respostas sejam geradas considerando tanto o conhecimento do modelo quanto as informações recuperadas da base de dados.

Após o processamento pelo LLM, a resposta é encaminhada novamente ao servidor de aplicação, registrada no banco de dados juntamente com a interação realizada e, por fim, apresentada ao usuário na interface web. Esse fluxo garante a continuidade das conversas, a persistência das informações e a associação de cada interação ao respectivo usuário autenticado.

A Figura 1 apresenta a arquitetura geral da aplicação, evidenciando o fluxo de comunicação entre os componentes e a integração entre a interface web, o servidor de aplicação, o PostgreSQL e o modelo de IA executado localmente.

3.5.2 Tecnologias Utilizadas

A implementação da aplicação exigiu a utilização de diferentes tecnologias de software, cada uma desempenhando funções específicas na arquitetura proposta. A seleção dessas ferramentas foi realizada considerando critérios como desempenho, compatibilidade entre os componentes, disponibilidade de documentação, suporte da comunidade, facilidade de integração e utilização de soluções de código aberto.

No desenvolvimento da interface web foram empregados HTML5, CSS3 e JavaScript, responsáveis pela estruturação das páginas, estilização da interface e implementação das funcionalidades executadas no navegador. Para o processamento das requisições no servidor foi utilizado o ambiente de execução Node.js em conjunto

com o framework Fastify, responsável pela implementação das APIs REST, gerenciamento das rotas e comunicação entre os diferentes módulos da aplicação.

O armazenamento das informações foi realizado por meio do PostgreSQL, utilizando a biblioteca **pg** para estabelecer a comunicação entre o servidor da aplicação e o banco de dados relacional. Essa combinação permitiu implementar autenticação de usuários, persistência do histórico de conversas e recuperação das informações utilizadas durante o processamento das solicitações.

A integração com o modelo de IA foi implementada utilizando a plataforma Ollama, responsável pela execução local do modelo DeepSeek-R1:8B. A comunicação entre o servidor da aplicação e o modelo ocorreu por meio da biblioteca Axios, possibilitando o envio das solicitações, recuperação das respostas geradas pelo LLM e integração da arquitetura baseada em RAG implementada neste trabalho.

Durante o desenvolvimento também foram utilizados o gerenciador de pacotes npm, responsável pelo gerenciamento das dependências do projeto, o ambiente de desenvolvimento Visual Studio Code para implementação e depuração do código-fonte e o Git, em conjunto com o GitHub, para controle de versão e rastreabilidade das alterações realizadas ao longo da pesquisa.

O Quadro 3 apresenta as principais tecnologias empregadas durante o desenvolvimento da aplicação, suas respectivas versões, finalidades e justificativas de utilização.

Quadro 2 – Tecnologias utilizadas e justificativas de escolha

Tecnologia/ferramenta	Versão utilizada	Finalidade no projeto	Justificativa de escolha
JavaScript	ES6 ou superior	Linguagem utilizada no desenvolvimento do front-end e em parte da lógica da aplicação.	Foi escolhida por sua ampla adoção em aplicações web, flexibilidade e compatibilidade com o ecossistema Node.js.
Node.js	v24.18.0	Ambiente de execução do back-end da aplicação.	Permite a execução de JavaScript no servidor, oferece suporte a operações assíncronas e facilita a criação de aplicações web escaláveis.
Fastify	v5.9.0	Framework utilizado para criação das rotas e gerenciamento das requisições HTTP.	Foi escolhido por sua leveza, desempenho e adequação à construção de APIs RESTful.
PostgreSQL	v18	Banco de dados relacional utilizado para armazenar	Apresenta robustez, integridade referencial, confiabilidade e ampla utilização em aplicações que

		usuários e históricos de conversas.	exigem persistência estruturada de dados.
pg	v8.22.0	Biblioteca de conexão entre Node.js e PostgreSQL.	Permite executar consultas SQL e gerenciar a comunicação entre o servidor da aplicação e o banco de dados relacional.
Axios	v1.18.1	Biblioteca utilizada para requisições HTTP entre os módulos da aplicação.	Facilita a comunicação com APIs e serviços externos ou locais, contribuindo para a integração entre servidor e modelo de IA.
Ollama	v0.30.11	Plataforma utilizada para execução local do modelo de linguagem.	Possibilita a execução de modelos de IA em ambiente local, reduzindo dependência de serviços externos e ampliando o controle sobre os dados processados.
DeepSeek-R1:8B	DeepSeek-R1:8B	Modelo de Inteligência Artificial utilizado para processamento de prompts e geração de respostas em linguagem natural.	Foi selecionado por permitir execução local, apresentar capacidade de geração textual coerente e ser adequado ao escopo experimental do projeto.
npm	v11.18.0	Gerenciador de pacotes utilizado para instalação e controle das dependências do projeto.	Facilita o gerenciamento das bibliotecas utilizadas no desenvolvimento da aplicação.
HTML5	HTML5	Estruturação das páginas e elementos da interface web.	Permite organizar semanticamente os componentes da interface e garantir compatibilidade com navegadores modernos.
CSS3	CSS3	Estilização visual da interface, incluindo layout, responsividade e modo escuro.	Foi utilizado para aprimorar a experiência visual, a legibilidade e a adaptação da aplicação a diferentes telas.
Visual Studio Code	v1.126	Ambiente de desenvolvimento utilizado para escrita, organização e depuração do código.	Foi escolhido por sua integração com extensões, terminal, controle de versão e suporte ao ecossistema JavaScript/Node.js.
Git e GitHub	v2.55.0	Controle de versão e armazenamento do código-fonte.	Foram utilizados para registrar alterações, organizar versões do projeto e apoiar a rastreabilidade do desenvolvimento.

Fonte: Elaborado pelo autor (2026).

3.5.3 Implementação do Back-end

O back-end da aplicação foi desenvolvido utilizando o ambiente de execução Node.js e o framework Fastify, responsáveis pelo processamento das requisições enviadas pela interface web, gerenciamento das regras de negócio e integração entre os diferentes componentes da arquitetura. A escolha dessas tecnologias deve-se ao

seu desempenho, suporte a operações assíncronas e facilidade para desenvolvimento de APIs REST.

A arquitetura do servidor foi organizada de forma modular, separando as responsabilidades em componentes específicos, como gerenciamento de rotas, autenticação de usuários, comunicação com o banco de dados, recuperação de informações e integração com o modelo de IA. Essa organização favorece a manutenção do código, reduz o acoplamento entre os módulos e facilita futuras expansões da aplicação.

O Fastify foi empregado para implementar os *endpoints* responsáveis pelas principais funcionalidades do sistema, incluindo autenticação de usuários, envio de mensagens, recuperação do histórico de conversas e comunicação com o modelo de IA. Cada requisição recebida pelo servidor é submetida a validações antes de ser processada, garantindo que apenas solicitações válidas sejam encaminhadas às etapas seguintes da aplicação.

Após o recebimento de uma solicitação, o servidor identifica o usuário autenticado e consulta o PostgreSQL para recuperar as informações necessárias ao processamento da requisição. Quando aplicável, os dados recuperados são utilizados para compor o contexto enviado ao LLM, seguindo os princípios da arquitetura RAG. Essa estratégia permite que as respostas geradas pelo modelo sejam fundamentadas tanto em seu conhecimento interno quanto nas informações armazenadas pela própria aplicação.

A comunicação entre o servidor e o modelo DeepSeek-R1:8B foi realizada por meio da plataforma Ollama, utilizando requisições HTTP para envio dos *prompts* e recebimento das respostas geradas. Após a inferência do modelo, a resposta é processada pelo servidor, registrada no banco de dados juntamente com a interação correspondente e posteriormente encaminhada à interface web.

Além do processamento das mensagens, o back-end também é responsável pelo gerenciamento das sessões de usuários e pelo controle de acesso às informações armazenadas. Cada interação permanece associada ao usuário autenticado, assegurando que o histórico de conversas seja recuperado exclusivamente pelo respectivo proprietário da sessão. Esse mecanismo contribui para preservar a privacidade dos dados e garantir a consistência das informações durante a utilização da aplicação.

A implementação modular adotada permitiu integrar autenticação, persistência de dados, recuperação de informações e processamento por IA em uma única arquitetura, mantendo baixo acoplamento entre os componentes e facilitando a manutenção da aplicação. Essa organização também possibilita a substituição ou atualização de módulos específicos sem comprometer o funcionamento dos demais componentes do sistema.

3.5.1 Implementação do Front-end

O front-end da aplicação foi desenvolvido utilizando HTML5, CSS3 e JavaScript, tecnologias responsáveis pela estruturação da interface, estilização visual e implementação das funcionalidades executadas no navegador. O desenvolvimento da interface teve como objetivo proporcionar uma interação simples e intuitiva, permitindo que o usuário utilizasse os recursos da aplicação sem necessidade de conhecimentos técnicos específicos.

A estrutura da interface foi organizada em diferentes componentes, contemplando as telas de autenticação, área principal de conversação e visualização do histórico das interações. Essa organização permitiu separar as funcionalidades da aplicação, facilitando a navegação entre as diferentes operações disponibilizadas ao usuário.

O processo de autenticação foi implementado por meio de um formulário de acesso, no qual o usuário informa suas credenciais para validação pelo servidor. Após a autenticação bem-sucedida, a interface disponibiliza o ambiente de conversação, mantendo a sessão ativa durante a utilização da aplicação e restringindo o acesso às informações do respectivo usuário autenticado.

A comunicação entre o front-end e o back-end ocorre por meio de requisições HTTP direcionadas às APIs REST implementadas no servidor. Sempre que o usuário envia uma mensagem, a interface encaminha a solicitação ao servidor, que realiza o processamento necessário e retorna a resposta gerada pelo modelo de IA. Após o recebimento da resposta, a interface atualiza dinamicamente a área de conversação, proporcionando uma interação contínua e em tempo real.

Além do envio de mensagens, a interface também permite recuperar o histórico de conversas armazenado no PostgreSQL. Durante o carregamento da sessão, as interações anteriormente registradas são recuperadas pelo servidor e

apresentadas ao usuário de forma organizada, possibilitando a continuidade das conversas e preservando o contexto das interações realizadas.

A estilização da aplicação foi desenvolvida utilizando CSS3, priorizando critérios de legibilidade, organização visual e responsividade. O layout foi estruturado para adaptar-se a diferentes resoluções de tela, mantendo a disposição dos elementos da interface e proporcionando uma experiência de utilização consistente em diferentes dispositivos.

O JavaScript foi empregado para controlar a interação entre os elementos da interface, validar entradas do usuário, enviar requisições assíncronas ao servidor e atualizar dinamicamente o conteúdo da página sem necessidade de recarregamento completo. Essa abordagem contribui para reduzir o tempo de resposta percebido pelo usuário e tornar a utilização da aplicação mais fluida.

A implementação do front-end foi planejada de forma integrada ao restante da arquitetura da aplicação, permitindo comunicação eficiente com o servidor, acesso seguro às funcionalidades disponíveis e apresentação organizada das respostas produzidas pelo modelo de IA. Dessa forma, a interface atua como elemento de integração entre o usuário e os demais componentes do sistema, proporcionando uma experiência de interação compatível com os objetivos estabelecidos para o artefato desenvolvido.

3.5.5 Integração com o LLM e Arquitetura RAG

A integração entre a aplicação web e o modelo de IA foi implementada com o objetivo de permitir que as solicitações enviadas pelos usuários fossem processadas por um LLM executado localmente, utilizando informações armazenadas na própria aplicação para complementar o contexto das respostas. Para isso, foi adotada uma arquitetura baseada nos princípios do **Retrieval-Augmented Generation (RAG)**, integrando o servidor da aplicação, o banco de dados PostgreSQL e o modelo DeepSeek-R1:8B executado por meio da plataforma Ollama.

O fluxo de processamento inicia-se quando o usuário envia uma mensagem pela interface web. A solicitação é encaminhada ao servidor desenvolvido em Node.js utilizando o framework Fastify, responsável por validar a requisição, identificar o usuário autenticado e coordenar a comunicação entre os diferentes componentes da arquitetura.

Após a validação da requisição, o servidor realiza consultas ao PostgreSQL para recuperar as informações necessárias ao processamento da solicitação. Dependendo do contexto da interação, podem ser recuperados dados do histórico de conversas, informações relacionadas ao usuário ou outros registros armazenados pela aplicação. Esses dados são organizados e utilizados para compor o contexto que será enviado ao modelo de IA.

A etapa de construção do contexto representa um dos principais elementos da arquitetura RAG implementada neste trabalho. Nessa etapa, as informações recuperadas do banco de dados são incorporadas ao *prompt* juntamente com a solicitação original do usuário, permitindo que o modelo utilize dados atualizados e específicos da aplicação durante a geração da resposta. Essa estratégia reduz a dependência exclusiva do conhecimento adquirido durante o treinamento do LLM e possibilita respostas mais adequadas ao domínio da aplicação.

Após a composição do contexto, o servidor encaminha a requisição ao modelo DeepSeek-R1:8B utilizando a API disponibilizada pela plataforma Ollama. O modelo processa o *prompt*, interpreta tanto a solicitação do usuário quanto as informações recuperadas do banco de dados e gera uma resposta em linguagem natural.

Concluído o processamento, a resposta é devolvida ao servidor, que realiza o registro da nova interação no PostgreSQL, preservando o histórico da conversa e permitindo sua recuperação em sessões futuras. Em seguida, o resultado é enviado ao front-end, onde é apresentado ao usuário na interface de conversação de forma imediata.

A adoção dessa arquitetura proporciona benefícios importantes para a aplicação. A execução local do modelo preserva maior controle sobre os dados processados, enquanto a recuperação dinâmica das informações permite que o LLM utilize conteúdos atualizados sem necessidade de novo treinamento. Além disso, a separação entre recuperação de dados, processamento da lógica da aplicação e inferência do modelo reduz o acoplamento entre os componentes, favorecendo a manutenção e futuras expansões do sistema.

A implementação dessa arquitetura permitiu integrar autenticação de usuários, recuperação de informações, persistência de dados e processamento de linguagem natural em uma única solução. Dessa forma, o artefato desenvolvido demonstra a viabilidade da utilização de tecnologias de código aberto para construção de aplicações conversacionais inteligentes, combinando uma arquitetura modular com

mecanismos de recuperação de informações e geração de respostas em linguagem natural.

3.5.6 Banco de Dados e Persistência

O armazenamento persistente das informações da aplicação foi implementado utilizando o Sistema Gerenciador de Banco de Dados Relacional PostgreSQL. A escolha dessa tecnologia deveu-se à sua confiabilidade, elevado desempenho, suporte ao padrão SQL e ampla utilização em aplicações corporativas e acadêmicas.

O banco de dados desempenha papel fundamental na arquitetura desenvolvida, sendo responsável pelo armazenamento das informações necessárias ao funcionamento da aplicação. Entre os principais dados persistidos estão os registros de usuários, credenciais de autenticação, histórico de conversas e demais informações utilizadas durante o processamento das interações. Essa persistência permite que os dados permaneçam disponíveis mesmo após o encerramento da sessão do usuário ou reinicialização da aplicação.

A comunicação entre o servidor desenvolvido em Node.js e o PostgreSQL foi realizada por meio da biblioteca **pg**, responsável pelo estabelecimento das conexões, execução das consultas SQL e gerenciamento das operações de leitura e escrita no banco de dados. Todas as consultas foram implementadas no servidor da aplicação, centralizando o acesso aos dados e reduzindo o acoplamento entre os componentes da arquitetura.

Durante o processo de autenticação, o servidor consulta o banco de dados para validar as credenciais fornecidas pelo usuário e recuperar as informações necessárias para a criação da sessão. Após a autenticação, todas as interações realizadas permanecem associadas ao usuário autenticado, garantindo que cada histórico de conversas seja armazenado e recuperado de forma individualizada.

Além da função de persistência, o PostgreSQL também atua como fonte de informações para a arquitetura baseada em RAG implementada neste trabalho. Sempre que necessário, o servidor consulta os dados armazenados e incorpora essas informações ao contexto enviado ao LLM. Dessa forma, o modelo pode utilizar informações atualizadas e específicas da aplicação durante a geração das respostas, ampliando a precisão e a relevância das interações.

A modelagem do banco de dados foi estruturada para preservar a integridade referencial entre as tabelas, estabelecendo relacionamentos entre usuários, sessões e conversas. Essa organização favorece a consistência dos dados, reduz redundâncias e facilita futuras expansões da aplicação, permitindo a inclusão de novas funcionalidades sem alterações significativas na estrutura existente.

A utilização do PostgreSQL como mecanismo de persistência contribuiu para garantir segurança, confiabilidade e organização das informações armazenadas pela aplicação. Sua integração com o servidor e com a arquitetura RAG permitiu combinar armazenamento estruturado, recuperação eficiente de dados e suporte ao processamento realizado pelo modelo de IA, constituindo um dos elementos centrais da solução proposta.

3.6 DEMONSTRAÇÃO DA APLICAÇÃO

A etapa de demonstração, prevista na Design Science Research Methodology (DSRM), tem como objetivo evidenciar o funcionamento do artefato desenvolvido em um cenário representativo de sua utilização. Segundo Peffers et al. (2007), essa etapa consiste em demonstrar que a solução proposta é capaz de atender aos objetivos definidos durante o planejamento da pesquisa, permitindo sua utilização em um contexto real ou simulado.

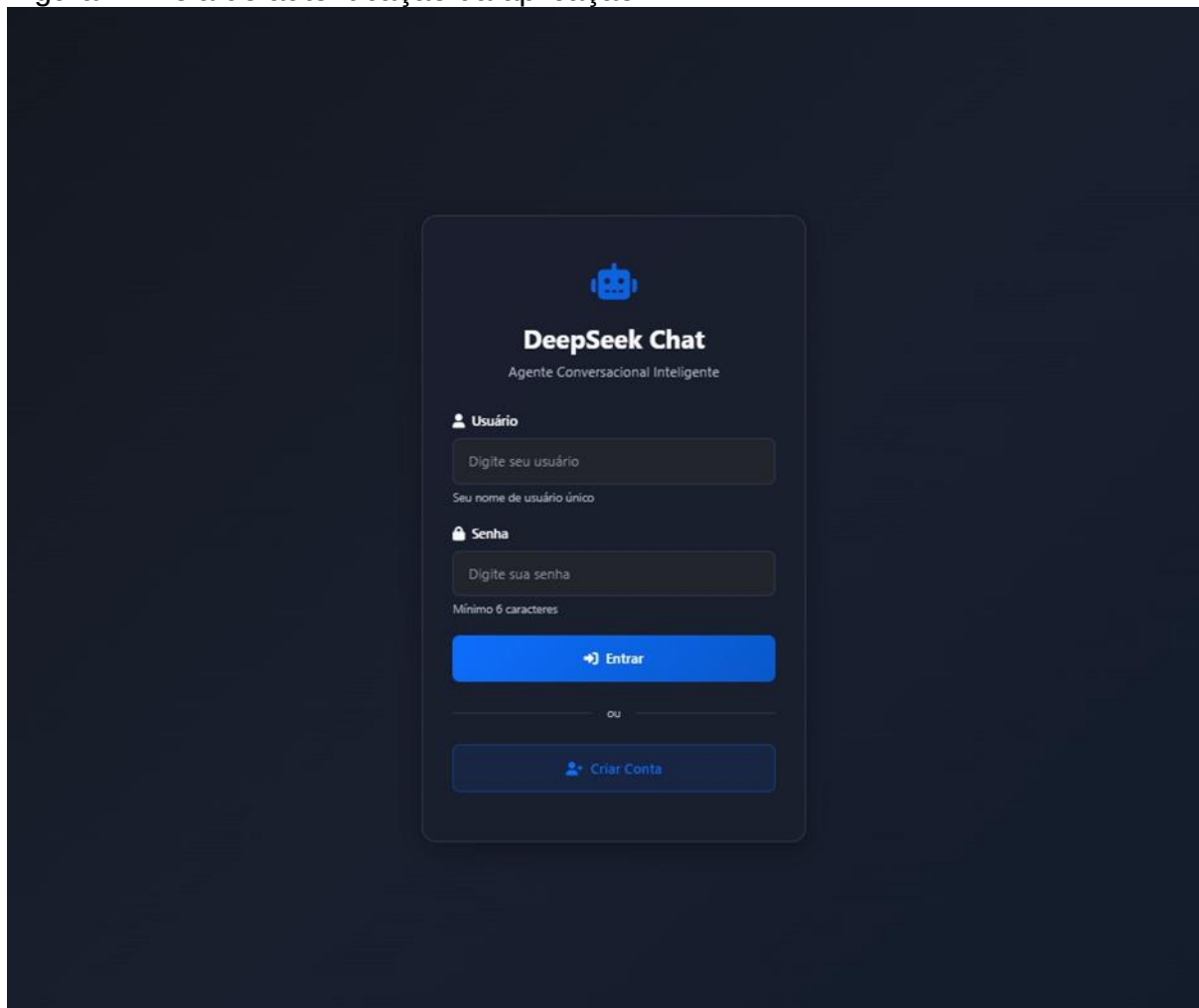
No presente trabalho, a demonstração foi realizada por meio da execução das principais funcionalidades da aplicação, abrangendo o processo de autenticação, interação com o modelo de IA, recuperação do histórico de conversas e persistência das informações no banco de dados PostgreSQL. Durante essa etapa, buscou-se reproduzir o fluxo normal de utilização do sistema, verificando a integração entre seus diferentes componentes.

3.6.1 Autenticação do Usuário

O fluxo de utilização da aplicação inicia-se com a autenticação do usuário. Nessa etapa, as credenciais informadas na interface são encaminhadas ao servidor, que realiza a validação das informações consultando o banco de dados PostgreSQL. Quando a autenticação é realizada com sucesso, uma sessão é criada e o usuário

passa a ter acesso às funcionalidades da aplicação. A **Figura 1** apresenta a tela de autenticação utilizada durante a demonstração do sistema.

Figura 1 - Tela de autenticação da aplicação



Fonte: Elaborado pelo autor (2026).

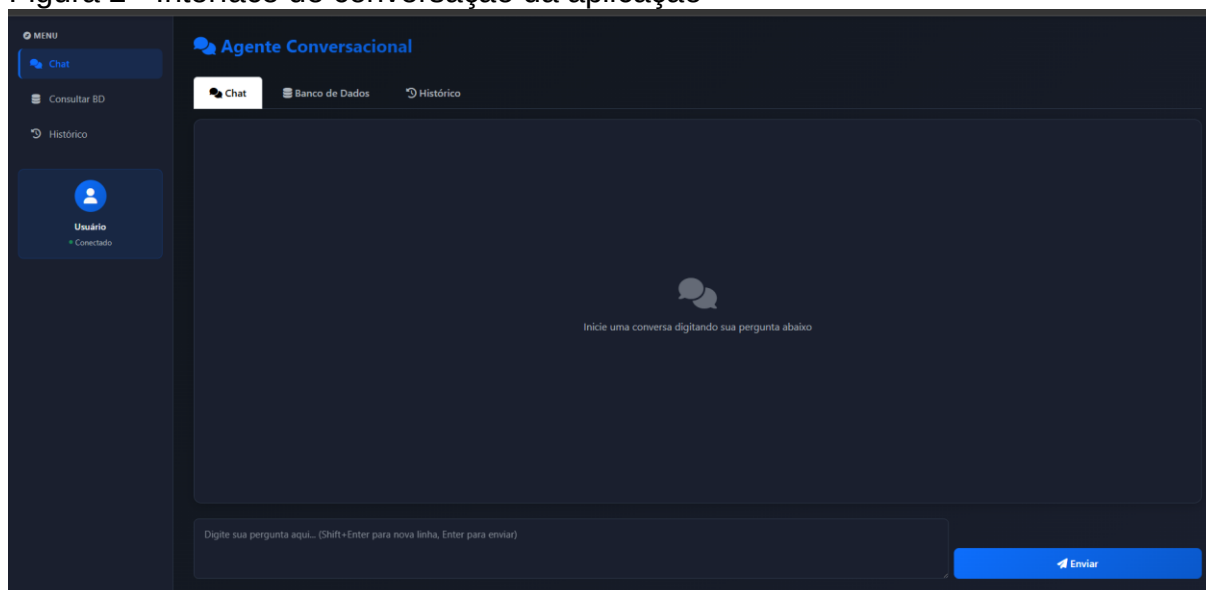
3.6.2 Interação com o Chatbot

Após a autenticação, o usuário é direcionado para a interface principal da aplicação, onde pode enviar solicitações em linguagem natural ao chatbot. Cada mensagem enviada é encaminhada ao servidor, que realiza a recuperação das informações necessárias, constrói o contexto da requisição e envia o *prompt* ao modelo DeepSeek-R1:8B executado localmente por meio do Ollama.

Após o processamento pelo LLM, a resposta é retornada ao servidor e é apresentada na interface de conversação, permitindo uma interação contínua entre usuário e sistema.

A **Figura 2** apresenta a interface principal durante uma sessão de conversação.

Figura 2 - Interface de conversação da aplicação



Fonte: Elaborado pelo autor (2026).

3.6.3 Recuperação do Histórico

Uma das funcionalidades implementadas consiste na recuperação do histórico de conversas armazenado no PostgreSQL. Sempre que o usuário inicia uma nova sessão, o servidor consulta o banco de dados e recupera as interações previamente registradas, permitindo a continuidade das conversas e preservando o contexto das informações.

Esse mecanismo também fornece suporte à arquitetura baseada em RAG, permitindo que informações armazenadas na aplicação sejam utilizadas durante o processamento das solicitações pelo modelo de IA.

A **Figura 3** apresenta um exemplo da recuperação do histórico de conversas.

Figura 3 - Recuperação do histórico armazenado no PostgreSQL



Fonte: Elaborado pelo autor (2026).

A demonstração evidenciou que os componentes da arquitetura operam de forma integrada, permitindo a autenticação dos usuários, a recuperação das informações persistidas, a comunicação com o LLM e o armazenamento das novas interações em uma única aplicação. Essa etapa confirmou a viabilidade técnica do artefato desenvolvido e serviu de base para a etapa seguinte da DSRM, correspondente à avaliação do sistema por meio de testes funcionais e métricas de desempenho.

3.7 AVALIAÇÃO DO ARTEFATO

A etapa de avaliação constitui a quinta fase da Design Science Research Methodology (DSRM) e tem como objetivo verificar se o artefato desenvolvido atende aos requisitos estabelecidos durante a definição da solução. Segundo Peffers et al. (2007), essa etapa consiste na aplicação de métodos que permitam analisar o desempenho da solução e comprovar sua capacidade de resolver o problema inicialmente identificado.

Neste trabalho, a avaliação foi realizada por meio de testes funcionais executados sobre a aplicação desenvolvida, verificando o correto funcionamento dos principais componentes da arquitetura, incluindo autenticação de usuários, comunicação entre a interface e o servidor, recuperação das informações armazenadas no PostgreSQL, integração com o modelo DeepSeek-R1:8B executado pelo Ollama e persistência das conversas.

3.7.1 Procedimentos de Avaliação

Os testes foram conduzidos em ambiente controlado, simulando a utilização normal da aplicação por usuários autenticados. Cada funcionalidade foi executada diversas vezes com o objetivo de verificar sua estabilidade, consistência e correto funcionamento durante diferentes interações.

- Foram avaliados os seguintes aspectos:
- autenticação dos usuários;
- envio de mensagens ao chatbot;
- recuperação do histórico de conversas;
- comunicação entre o servidor e o modelo de IA;
- armazenamento das interações no PostgreSQL;
- tempo de resposta da aplicação;
- estabilidade geral do sistema.

Além da verificação funcional, foram registradas métricas relacionadas ao desempenho da aplicação, permitindo uma análise quantitativa dos resultados obtidos.

3.7.2 Resultados dos Testes Funcionais

Ao todo, foram realizados **30 testes funcionais**, abrangendo todas as funcionalidades consideradas essenciais para o funcionamento da aplicação.

Os resultados demonstraram que todas as operações avaliadas apresentaram funcionamento consistente durante os testes. A autenticação dos usuários obteve **100% de sucesso**, com validação correta das credenciais em todas as execuções realizadas. Da mesma forma, a recuperação do histórico de conversas apresentou **100% de sucesso**, permitindo que todas as interações previamente armazenadas fossem recuperadas corretamente após o início de uma nova sessão.

Durante os testes também foram enviados **20 prompts** ao modelo DeepSeek-R1:8B por meio da plataforma Ollama. Em todas as execuções, a comunicação entre o servidor da aplicação e o modelo ocorreu sem falhas, demonstrando estabilidade na integração entre os componentes da arquitetura.

Em relação ao desempenho, a aplicação apresentou **latência média de resposta de 2,8 segundos**, variando entre **1,9 segundo** e **4,1 segundos**, valores considerados adequados para aplicações conversacionais executadas localmente.

Durante todo o processo de avaliação não foram registradas **falhas críticas**, interrupções do serviço ou inconsistências na comunicação entre os módulos da aplicação, evidenciando a estabilidade da arquitetura implementada.

A Tabela 2 apresenta um resumo dos principais indicadores obtidos durante os testes.

Indicador	Resultado
Testes funcionais realizados	30
Sucesso na autenticação	100% (30/30)
Sucesso na recuperação do histórico	100% (30/30)
Prompts processados pelo LLM	20
Falhas de comunicação com o Ollama	0
Latência média	2,8 s
Menor latência	1,9 s
Maior latência	4,1 s
Falhas críticas	0

3.7.3 Análise dos Resultados

Os resultados obtidos demonstram que a arquitetura proposta foi capaz de integrar corretamente todos os componentes previstos durante o desenvolvimento do artefato. A ausência de falhas durante os testes evidencia a estabilidade da comunicação entre a interface web, o servidor desenvolvido em Node.js, o PostgreSQL e o modelo DeepSeek-R1:8B executado localmente pelo Ollama.

A recuperação consistente das informações armazenadas no banco de dados confirma o correto funcionamento da estratégia baseada em RAG implementada na aplicação. Ao incorporar informações persistidas ao contexto enviado ao modelo de IA, o sistema foi capaz de utilizar dados específicos da aplicação durante a geração das respostas, atendendo aos objetivos definidos para o desenvolvimento do artefato.

As métricas de desempenho também indicam que a solução apresenta tempo de resposta compatível com aplicações conversacionais executadas localmente. A latência média observada demonstra que a integração entre os diferentes componentes da arquitetura ocorreu de forma eficiente, sem comprometer significativamente a experiência de utilização.

Dessa forma, a avaliação realizada permitiu verificar que o artefato desenvolvido atende aos requisitos funcionais estabelecidos na pesquisa, demonstrando sua viabilidade técnica para integração entre aplicações web, banco de dados relacional e modelos de linguagem executados localmente. Esses resultados validam a proposta apresentada neste trabalho e evidenciam sua contribuição para o desenvolvimento de aplicações conversacionais baseadas em Inteligência Artificial utilizando tecnologias de código aberto

4. RESULTADOS E DISCUSSÃO

4.1 AMBIENTE EXPERIMENTAL

Os experimentos foram realizados em ambiente controlado, utilizando a arquitetura desenvolvida ao longo desta pesquisa. O objetivo foi avaliar o funcionamento integrado da aplicação, contemplando a comunicação entre a interface web, o servidor de aplicação, o banco de dados PostgreSQL e o modelo de linguagem executado localmente.

A aplicação foi executada em um computador equipado com processador **AMD Ryzen 5 5600X**, **32 GB de memória RAM**, placa de vídeo **AMD Radeon RX 6750 XT** e **SSD de 512 GB**, utilizando o sistema operacional Windows. O servidor da aplicação foi desenvolvido utilizando o ambiente de execução Node.js e o framework Fastify, enquanto o PostgreSQL foi empregado para armazenamento persistente das informações. O processamento das solicitações em linguagem natural foi realizado pelo modelo **DeepSeek-R1:8B**, executado localmente por meio da plataforma Ollama.

Durante os experimentos, o Ollama utilizou aceleração por hardware da GPU por meio da plataforma **ROCm (Radeon Open Compute)**, permitindo que o processamento do modelo de linguagem fosse realizado diretamente pela placa gráfica. A utilização dessa tecnologia reduziu significativamente o tempo necessário para a inferência do modelo, contribuindo para os tempos de resposta observados durante a avaliação da aplicação.

A comunicação entre os componentes da arquitetura ocorreu por meio de APIs REST, permitindo a troca de informações entre a interface web, o servidor, o banco de dados e o modelo de IA. Todos os testes foram executados utilizando a mesma configuração de hardware e software, garantindo uniformidade das condições experimentais e maior confiabilidade dos resultados obtidos.

A Tabela 3 - Configuração utilizada durante os experimentos.

Componente	Especificação
Processador	AMD Ryzen 5 5600X (6 núcleos / 12 threads)
Memória RAM	32 GB DDR4
Placa de vídeo	AMD Radeon RX 6750 XT
Aceleração do LLM	GPU via ROCm
Armazenamento	SSD 512 GB
Sistema Operacional	Windows
Ambiente de execução	Node.js
Framework	Fastify
Banco de Dados	PostgreSQL
Modelo de IA	DeepSeek-R1:8B
Plataforma do modelo	Ollama
Comunicação	APIs REST (HTTP)
Front-end	HTML5, CSS3 e JavaScript

Durante a etapa experimental foram realizados **30 testes funcionais**, abrangendo as principais funcionalidades da aplicação, incluindo autenticação de usuários, recuperação do histórico de conversas, persistência das informações e integração entre o servidor e o modelo de IA. Além disso, foram processados **20 prompts** pelo modelo DeepSeek-R1:8B, permitindo avaliar a estabilidade da comunicação entre os componentes da arquitetura.

As métricas analisadas compreenderam a taxa de sucesso das funcionalidades implementadas, o tempo de resposta da aplicação, a estabilidade da comunicação entre os módulos e a consistência da persistência dos dados

armazenados no PostgreSQL. Os resultados dessas avaliações são apresentados nas seções subsequentes.

4.2 RESULTADOS DA AUTENTICAÇÃO

A autenticação dos usuários constitui uma das funcionalidades essenciais da arquitetura proposta, sendo responsável por garantir que apenas usuários devidamente cadastrados possam acessar os recursos da aplicação. Além do controle de acesso, esse mecanismo assegura que cada usuário tenha acesso exclusivamente às suas informações e ao seu histórico de conversas armazenado no banco de dados.

Com o objetivo de validar essa funcionalidade, foram realizados **30 testes funcionais**, contemplando diferentes cenários de autenticação, incluindo acesso com credenciais válidas, verificação da criação da sessão do usuário e acesso às funcionalidades protegidas da aplicação.

Durante todos os testes realizados, o sistema apresentou funcionamento consistente, validando corretamente as credenciais informadas e estabelecendo a sessão do usuário sempre que a autenticação foi concluída com sucesso. Não foram observadas falhas durante o processo de validação das credenciais ou na comunicação entre o front-end, o servidor da aplicação e o banco de dados PostgreSQL.

Tabela 4 – Resultados dos testes de autenticação

Indicador	Resultado
Testes realizados	30
Autenticações realizadas com sucesso	30
Taxa de sucesso	100%
Falhas de autenticação do sistema	0
Sessões criadas corretamente	30
Inconsistências identificadas	0

Os resultados obtidos evidenciam que a arquitetura implementada foi capaz de integrar corretamente a interface web, o servidor desenvolvido em Node.js, o banco de dados PostgreSQL e os mecanismos de controle de acesso. A taxa de sucesso de

100% durante os testes demonstra que a autenticação atendeu aos requisitos funcionais definidos para o artefato, constituindo uma base confiável para as demais funcionalidades da aplicação, como recuperação do histórico de conversas e integração com o modelo de IA.

4.3 RESULTADOS DA PERSISTÊNCIA DOS DADOS

A persistência dos dados constitui um dos componentes centrais da arquitetura desenvolvida, sendo responsável pelo armazenamento permanente das informações geradas durante a utilização da aplicação. Para essa finalidade, foi utilizado o PostgreSQL, encarregado de armazenar dados de autenticação, informações dos usuários e o histórico completo das conversas realizadas com o chatbot.

Com o objetivo de avaliar a confiabilidade desse mecanismo, foram realizados testes envolvendo o registro de novas interações, a recuperação do histórico de conversas e a verificação da integridade das informações armazenadas. Os testes buscaram confirmar se os dados permaneciam disponíveis após o encerramento e reinício das sessões dos usuários.

Durante a avaliação experimental, foram executados **30 testes de recuperação do histórico**, abrangendo diferentes cenários de utilização da aplicação. Em todos os casos avaliados, as informações previamente armazenadas foram recuperadas corretamente pelo sistema, sem perda de dados ou inconsistências entre os registros armazenados e os apresentados ao usuário.

Os resultados obtidos são apresentados na Tabela 5.

Tabela 5 – Resultados dos testes de persistência dos dados

Indicador	Resultado
Testes de recuperação realizados	30
Recuperações realizadas com sucesso	30
Taxa de sucesso	100%
Perda de registros identificada	0
Inconsistências nos dados	0
Falhas na recuperação	0

Os resultados demonstram que o mecanismo de persistência implementado apresentou funcionamento estável durante toda a avaliação. Todas as interações registradas permaneceram armazenadas no banco de dados e puderam ser recuperadas corretamente em sessões posteriores, confirmando a eficácia da estratégia adotada para armazenamento das informações.

Além de garantir a continuidade das conversas, a persistência das informações desempenha papel fundamental na arquitetura baseada em RAG implementada neste trabalho. A disponibilidade dos dados armazenados permite que informações relevantes sejam recuperadas e incorporadas ao contexto enviado ao modelo de IA contribuindo para a geração de respostas mais contextualizadas e alinhadas às informações mantidas pela aplicação.

A taxa de sucesso de **100%** observada nos testes confirma que o PostgreSQL atendeu adequadamente aos requisitos de armazenamento e recuperação definidos para o sistema. Esses resultados evidenciam que a solução desenvolvida é capaz de preservar a integridade dos dados e garantir acesso consistente às informações necessárias para o funcionamento da aplicação.

4.4 RESULTADOS DA INTEGRAÇÃO COM O DEEPSEEK

Um dos principais objetivos deste trabalho foi integrar um modelo de linguagem de grande porte (LLM) a uma aplicação web, permitindo que as solicitações dos usuários fossem processadas localmente e enriquecidas com informações armazenadas no banco de dados PostgreSQL por meio de uma arquitetura baseada em Retrieval-Augmented Generation (RAG).

Para avaliar essa integração, foram realizados testes envolvendo o envio de solicitações em linguagem natural ao modelo DeepSeek-R1:8B executado na plataforma Ollama. Durante os experimentos, cada requisição percorreu todo o fluxo da aplicação, incluindo o recebimento da mensagem pelo servidor, recuperação das informações necessárias no PostgreSQL, construção do contexto da requisição, processamento pelo modelo de IA e retorno da resposta ao usuário.

Ao todo, foram processados **20 prompts**, contemplando diferentes tipos de consultas relacionadas às funcionalidades da aplicação. Em todas as execuções, a comunicação entre o servidor desenvolvido em Node.js, a plataforma Ollama e o modelo DeepSeek-R1:8B ocorreu de forma estável, não sendo registradas falhas de

comunicação, interrupções no processamento ou inconsistências nas respostas retornadas pelo sistema.

Os resultados obtidos são apresentados na Tabela 6.

Tabela 6 – Resultados da integração entre a aplicação e o modelo DeepSeek-R1:8B

Indicador	Resultado
Prompts enviados ao modelo	20
Prompts processados com sucesso	20
Taxa de sucesso	100%
Falhas de comunicação com o Ollama	0
Respostas retornadas corretamente	20
Interrupções durante o processamento	0

Os resultados demonstram que a integração entre a aplicação e o modelo de linguagem apresentou elevado nível de estabilidade durante os testes realizados. Todas as solicitações enviadas ao LLM foram processadas corretamente, evidenciando o funcionamento adequado da comunicação entre o servidor da aplicação e a plataforma Ollama.

Além da estabilidade da comunicação, verificou-se o correto funcionamento da estratégia baseada em RAG implementada neste trabalho. Antes do envio de cada solicitação ao modelo de IA, o servidor recuperou as informações relevantes armazenadas no PostgreSQL e incorporou esses dados ao contexto do *prompt*. Essa abordagem permitiu que o modelo utilizasse informações específicas da aplicação durante a geração das respostas, ampliando a contextualização e a relevância das interações.

Os resultados obtidos evidenciam que a arquitetura proposta foi capaz de integrar de forma eficiente a interface web, o servidor desenvolvido em Node.js, o banco de dados PostgreSQL e o modelo DeepSeek-R1:8B executado localmente por meio da plataforma Ollama. A taxa de sucesso de **100%** nas comunicações realizadas demonstra que a solução apresentou comportamento consistente durante a avaliação experimental, confirmando a viabilidade técnica da utilização de LLMs locais em aplicações conversacionais desenvolvidas com tecnologias de código aberto.

Adicionalmente, a utilização do modelo em ambiente local, com aceleração por GPU utilizando a plataforma ROCm, proporcionou independência de serviços

externos, maior controle sobre os dados processados e redução da dependência de APIs comerciais. Essas características reforçam uma das principais contribuições deste trabalho, demonstrando que é possível desenvolver aplicações conversacionais inteligentes utilizando infraestrutura própria e ferramentas de código aberto, mantendo desempenho adequado e integração eficiente entre os componentes da arquitetura.

6. CONSIDERAÇÕES FINAIS

O presente trabalho teve como objetivo desenvolver e validar uma aplicação web integrada a um modelo de Inteligência Artificial baseado em Processamento de Linguagem Natural, utilizando tecnologias modernas e acessíveis para proporcionar uma experiência de interação organizada, personalizada e eficiente. A proposta surgiu da necessidade de investigar como modelos de linguagem podem ser incorporados a sistemas web capazes de oferecer autenticação de usuários, armazenamento estruturado de informações e recuperação do histórico de interações, superando limitações frequentemente observadas em aplicações conversacionais tradicionais.

A partir dos estudos teóricos realizados e da implementação prática desenvolvida, foi possível responder ao problema de pesquisa proposto. Os resultados demonstraram que a integração entre Inteligência Artificial, mecanismos de autenticação, persistência de dados e interface web é tecnicamente viável e pode ser implementada por meio de uma arquitetura modular composta por front-end, back-end, banco de dados relacional e modelo de linguagem natural. A validação funcional realizada confirmou que os diferentes componentes do sistema operam de forma integrada, permitindo a realização de interações em linguagem natural, armazenamento das conversas e recuperação posterior das informações registradas.

Em relação ao objetivo geral, considera-se que este foi alcançado no escopo definido para este trabalho.. A aplicação desenvolvida mostrou-se capaz de integrar um modelo de Inteligência Artificial baseado em linguagem natural a um ambiente web funcional, oferecendo ao usuário uma experiência organizada e personalizada. Da mesma forma, todos os objetivos específicos foram atendidos. O sistema de autenticação permitiu o controle individualizado de acesso; o modelo DeepSeek-R1:8B foi integrado com sucesso ao ambiente da aplicação; o banco de dados PostgreSQL garantiu a persistência das conversas; a interface web apresentou características de responsividade e usabilidade adequadas; e os testes realizados possibilitaram validar a funcionalidade dos principais módulos implementados.

Do ponto de vista técnico, este trabalho demonstra que soluções baseadas em modelos de linguagem podem ser implementadas utilizando tecnologias de código aberto e infraestrutura local, reduzindo custos operacionais e ampliando a autonomia no desenvolvimento de aplicações inteligentes. Além disso, a utilização do Ollama para execução local do modelo de IA evidenciou uma alternativa viável para projetos

acadêmicos e institucionais que necessitam de maior controle sobre os dados processados e sobre a infraestrutura computacional utilizada.

Sob a perspectiva acadêmica, a pesquisa contribui para a compreensão da integração entre áreas distintas, como Inteligência Artificial, Processamento de Linguagem Natural, Engenharia de Software, Banco de Dados e Experiência do Usuário. A articulação desses conhecimentos permitiu transformar conceitos teóricos em uma solução prática, reforçando a importância da interdisciplinaridade no desenvolvimento de sistemas computacionais modernos. O trabalho também evidencia o potencial dos modelos de linguagem como ferramentas capazes de ampliar a interação entre usuários e sistemas digitais, tornando-a mais natural, acessível e eficiente.

Entretanto, algumas limitações devem ser reconhecidas. A avaliação da aplicação foi realizada em ambiente local e com foco predominantemente funcional, não contemplando uma amostra ampla de usuários reais nem a utilização de métricas quantitativas detalhadas de desempenho. Dessa forma, aspectos relacionados à satisfação do usuário, escalabilidade e comportamento do sistema sob cargas elevadas não puderam ser analisados de maneira aprofundada. Além disso, a qualidade das respostas geradas permanece condicionada às capacidades e limitações inerentes ao modelo de linguagem utilizado.

Como possibilidades de continuidade desta pesquisa, recomenda-se a realização de estudos envolvendo usuários reais, permitindo a aplicação de métodos formais de avaliação de usabilidade e experiência do usuário. Também são sugeridas comparações entre diferentes modelos de linguagem, análises quantitativas de desempenho, implementação de mecanismos avançados de segurança e privacidade, além da integração da aplicação com serviços externos e ambientes em nuvem. Outra linha promissora consiste na incorporação de recursos de memória contextual aprimorada, personalização adaptativa e análise inteligente do histórico de interações.

Por fim, conclui-se que a pesquisa atingiu os resultados esperados e demonstrou o potencial da Inteligência Artificial aplicada a sistemas web como instrumento de apoio à comunicação, organização da informação e interação homem-computador. A solução desenvolvida evidencia que a combinação entre modelos de linguagem natural, autenticação de usuários, persistência de dados e interfaces intuitivas pode contribuir para a construção de aplicações mais inteligentes, eficientes

e alinhadas às demandas tecnológicas contemporâneas. Dessa forma, o trabalho não apenas atende aos objetivos acadêmicos propostos, mas também estabelece uma base sólida para futuras investigações e evoluções tecnológicas na área de Inteligência Artificial aplicada a sistemas web.

REFERÊNCIAS

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS (ABNT). *NBR 14724:2023 Trabalhos acadêmicos: Apresentação*. Rio de Janeiro: ABNT, 2023.

DATE, C. J. *An Introduction to Database Systems*. 8. ed. Boston: Addison-Wesley, 2019.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. *Deep Learning*. Cambridge: MIT Press, 2016.

McTEAR, Michael F. *Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots*. California: Morgan & Claypool Publishers, 2020.

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (NIST). Digital Identity Guidelines: Authentication and Authenticator Management. NIST Special Publication 800-63B-4. Gaithersburg: National Institute of Standards and Technology, 2025. Disponível em: <https://csrc.nist.gov/pubs/sp/800/63/b/4/final>. Acesso em: 30 maio 2026.

NECULA, Alexandru et al. A Systematic Literature Review on Using Natural Language Processing in Software Requirements Engineering. *Electronics*, v. 13, n. 11, p. 2055, 2024. DOI: <https://doi.org/10.3390/electronics13112055>.

NIELSEN, Jakob. *Usability 101: Introduction to Usability*. Fremont: Nielsen Norman Group, 2012. Disponível em: <https://www.nngroup.com/articles/usability-101-introduction-to-usability/>. Acesso em: 9 nov. 2025.

OPEN WORLDWIDE APPLICATION SECURITY PROJECT (OWASP). Authentication Cheat Sheet. OWASP Cheat Sheet Series, 2026. Disponível em: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html. Acesso em: 30 maio 2026.

PRESSMAN, Roger S.; MAXIM, Bruce R. *Software Engineering: A Practitioner's Approach*. 9. ed. New York: McGraw-Hill Education, 2021.

RUSSELL, Stuart; NORVIG, Peter. *Artificial Intelligence: A Modern Approach*. 4. ed. Hoboken: Pearson, 2021.

SILVA MAFRA, Rafael; BARBOSA, Simone; GOMES, Carlos. Proposing Usability-UX Technologies for the Design and Evaluation of Text-Based Chatbots. *Journal on Interactive Systems*, v. 15, n. 1, p. 1-16, 2024. Disponível em: <https://journals-sol.sbc.org.br/index.php/jis/article/view/3856>. Acesso em: 9 nov. 2025.

SOMMERVILLE, Ian. *Engenharia de Software*. 10. ed. São Paulo: Pearson Education, 2020.

VASWANI, Ashish et al. Attention Is All You Need. *Advances in Neural Information Processing Systems (NeurIPS)*, v. 30, 2017. Disponível em: <https://arxiv.org/abs/1706.03762>. Acesso em: 9 nov. 2025.

XUE, Yujie; ZHANG, Wen; LI, Tao; GAO, Jing; ZHANG, Min. Bias and Fairness in Chatbots: An Overview. *arXiv preprint*, 2023. Disponível em: <https://arxiv.org/abs/2309.08836>. Acesso em: 9 nov. 2025.

ZHOU, Li; GAO, Jianfeng; LI, Di; SHUM, Heung-Yeung. The Design and Implementation of Chatbots for Customer Service. *Computational Linguistics*, v. 46, n. 1, p. 53–81, 2020. DOI: https://doi.org/10.1162/coli_a_00368