Anais do Congresso Brasileiro de Iniciação Científica
(https://periodicos.unisanta.br/COB)

ISSN 2317-1316

ACADEMAI: SISTEMA INTELIGENTE PARA *WEB* QUE GERENCIA RESERVAS E INFORMAÇÕES EM LABORATÓRIOS ACADÊMICOS

ACADEMAI: INTELLIGENT SYSTEM FOR THE WEB THAT MANAGES RESERVATIONS AND INFORMATION IN ACADEMIC LABORATORIES

Arthur Silles Fernandes, Fabrício Suzart Andrade, Juliana Pallin de Amorim, Maria Fernanda de Almeida Maneira, Sergio Schina de Andrade

Universidade Santa Cecília, Faculdade de Engenharia, Curso de Engenharia de Computação
E-mail para contato: juliana.pallin@yahoo.com

RESUMO – Diversos Fab Labs ainda administram recursos compartilhados por meio de formulários manuais, e estudantes com deficiência enfrentam barreiras de acesso a esses processos. Este trabalho desenvolveu e validou o ACADEMAI, plataforma web de código aberto que une a consulta documental à gestão de reservas por meio de um assistente de inteligência artificial executado localmente. A solução integra o modelo Llama 3.2 3B, via Ollama, à geração aumentada por recuperação (RAG), com persistência em SQLite mediada por chamadas estruturadas validadas pelo servidor. Em estudo de caso com seis usuários, o assistente conduziu reservas de forma conversacional, exibiu a fonte documental das respostas e obteve tempo mediano de resposta de 1,65 s em hardware sem GPU dedicada.

Palavras-chave: Inteligência artificial; *Retrieval-Augmented Generation*; Execução local de modelos; Acessibilidade digital; Gestão de laboratórios.

ABSTRACT – Many Fab Labs still manage shared resources through manual forms, and students with disabilities face barriers to accessing these processes. This work developed and validated ACADEMAI, an open-source web platform that combines document consultation with reservation management through a locally executed artificial intelligence assistant. The solution integrates the Llama 3.2 3B model, run via Ollama, with Retrieval-Augmented Generation (RAG), using SQLite persistence mediated by structured calls validated by the server. In a case study with six users, the assistant conducted reservations conversationally, displayed the documentary source of its answers and achieved a median response time of 1.65 s on hardware without a dedicated GPU.

Keywords: Artificial intelligence; Retrieval-Augmented Generation; Local model execution; Digital accessibility; Laboratory management.

ISSN 2317-1316

1 INTRODUÇÃO

A crescente utilização de assistentes conversacionais baseados em modelos de linguagem de grande escala alterou significativamente a forma como a sociedade se relaciona com os sistemas computacionais. A comunicação em linguagem natural, antes restrita aos sistemas de busca, tornou-se a interface convencional de diversas ferramentas digitais, moldando as expectativas dos usuários sobre como qualquer *software* deve se comunicar. Esse movimento, entretanto, ainda não se manifesta de forma homogênea nos procedimentos administrativos internos das instituições de ensino superior.

Os laboratórios de manufatura digital, como os Fab Labs, ainda administram o uso de máquinas compartilhadas por meio de formulários manuais ou sistemas pouco integrados, o que prejudica a eficiência e a rastreabilidade dos agendamentos. Tais sistemas não aproveitam a inteligência artificial (IA) para aperfeiçoar a gestão de reservas de equipamentos, evitar a subutilização ou a sobrecarga dos ativos e otimizar o acesso ao conhecimento institucional (Izourane et al., 2024).

Soma-se a isso uma barreira de acesso: os trâmites institucionais nas esferas científicas e tecnológicas, assim como a educação superior, constituem obstáculo para alunos com deficiência (James et al., 2019). Pesquisas recentes indicam que tecnologias de IA podem atender a demandas específicas de aprendizado e reduzir obstáculos interativos no contexto educacional (Hauck et al., 2025). Nesse contexto, a interação por voz é considerada uma modalidade adicional de acesso à interface multimodal, e não uma substituição ao teclado, ao leitor de tela ou ao HTML semântico.

Diante desse quadro, este trabalho propõe o ACADEMAI (Fernandes et al., 2026), sistema *web* de código aberto que une a administração de agendamentos de recursos laboratoriais à consulta de uma base documental institucional, por meio de um assistente conversacional operado localmente. A execução local reduz a dependência de fornecedores externos de IA e restringe a exposição de solicitações, documentos e registros de reserva a terceiros; ao eliminar os custos variáveis do consumo de *tokens* de APIs, transfere-os para *hardware*, energia, manutenção e suporte técnico, cabendo a cada instituição ponderar essa substituição. Ressalta-se que a execução local, isoladamente, não assegura a privacidade, que também depende de autenticação, autorização, controle de acesso e segurança da infraestrutura institucional.

ISSN 2317-1316

O objetivo geral da pesquisa consiste em desenvolver e validar um sistema *web* acessível para a administração de reservas de equipamentos e espaços em laboratório, com recursos que visam ao uso autônomo das funcionalidades essenciais por usuários com deficiência visual ou motora, apoiados por um assistente de IA local; a validação desse uso autônomo junto ao público-alvo permanece como etapa futura. Os objetivos específicos são: descrever os requisitos, as regras de negócio e os casos de uso; conceber as camadas de apresentação, servidor, motor de IA e armazenamento; implementar autenticação, autorização e fluxo de aprovação de reservas por perfil; desenvolver um *pipeline* de geração aumentada por recuperação com citação obrigatória da fonte; estabelecer chamadas estruturadas que impeçam o modelo de escrever diretamente no banco de dados; e avaliar o protótipo por meio de um estudo de caso e de testes de desempenho.

2 MATERIAIS E MÉTODOS

A pesquisa é aplicada e exploratória, conduzida em duas etapas interdependentes: a implementação do sistema e sua avaliação, por meio de um estudo de caso complementado por testes de desempenho. Descrevem-se a seguir a arquitetura, as tecnologias adotadas, as alternativas descartadas e o método de avaliação.

2.1 Arquitetura do sistema

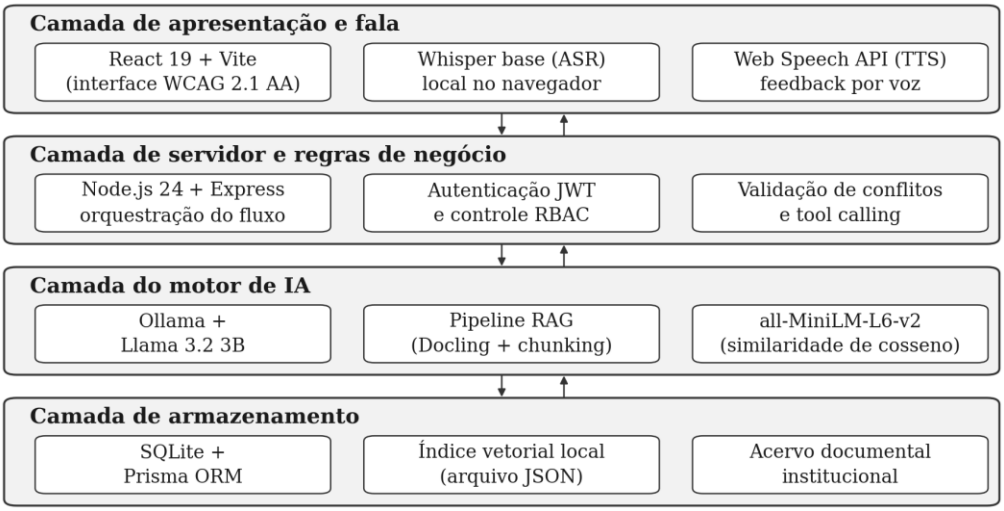
O sistema foi desenvolvido em um *monorepo*, repositório único no qual todos os componentes são versionados em conjunto, embora cada um seja executado como processo independente e se comunique com os demais por interfaces bem definidas, o que facilita o controle de versões e a consistência entre módulos sem exigir um monolito. A arquitetura foi estruturada em quatro camadas, conforme mostra a Figura 1.

A camada de apresentação, desenvolvida em React 19 com Vite, abriga a interface de diálogo, o calendário de disponibilidade, os recursos de acessibilidade e o reconhecimento de fala, processado diretamente no navegador. A camada de servidor, construída com Express sobre Node.js, orquestra o fluxo da conversa, intermedia a comunicação com o motor de IA, aplica as regras de negócio e expõe as rotas consumidas pelo *frontend*. A camada do motor de IA combina o modelo de linguagem com o fluxo de recuperação de documentos, e a camada de armazenamento reúne o banco de dados relacional, acessado via Prisma ORM, e o índice

ISSN 2317-1316

vetorial dos fragmentos indexados. O serviço de conversão de documentos, desenvolvido em Python, é executado como processo distinto.

Figura 1: Arquitetura em quatro camadas do sistema ACADEMAI.



2.2 Modelo de linguagem e execução local

Os modelos de linguagem de grande escala funcionam de maneira autorregressiva, com base na arquitetura Transformer: cada *token* gerado é condicionado às entradas e aos *tokens* anteriores, o que os torna adequados a diálogos, resumos e tarefas que seguem instruções (Vaswani et al., 2017). Utilizou-se o Llama 3.2 com três bilhões de parâmetros, modelo multilíngue ajustado por instruções e otimizado para assistência conversacional, recuperação de informações e resumo. Ele resulta da aplicação de *pruning* e *distillation* sobre variantes maiores da geração anterior, preservando o raciocínio e o uso de ferramentas com custo computacional muito menor, e oferece janela de contexto de 128 mil *tokens* (Meta Llama, 2024).

As limitações do projeto foram decisivas: a versão de três bilhões de parâmetros é leve o bastante para *hardware* padrão, enquanto versões maiores ou serviços de API exigem *hardware* especializado ou o envio de dados a servidores externos, e a versão de um bilhão de parâmetros tem capacidade de raciocínio e de seguimento de instruções insuficiente para a coleta de dados de reserva.

O Ollama, *software* de código aberto, controla a execução do modelo, simplifica a configuração de modelos locais e disponibiliza uma API compatível com o padrão do setor, o que reduz o esforço de integração com os demais elementos do sistema. O llama.cpp, embora

ISSN 2317-1316

ofereça inferência ligeiramente mais rápida, exige o gerenciamento manual dos arquivos do modelo e a configuração explícita de parâmetros, o que contraria a proposta de simplicidade do projeto; o vLLM, por sua vez, foi projetado para atender muitas chamadas simultâneas em GPUs de alto desempenho, o que não é o objetivo deste trabalho.

2.3 Geração aumentada por recuperação

Um modelo de linguagem responde apenas com base em seus dados de treinamento, e não no conhecimento produzido internamente por uma instituição. Para contornar essa limitação, Lewis et al. (2020) propuseram a *Retrieval-Augmented Generation* (RAG), que combina a geração de texto com a recuperação de informações externas durante a inferência: os documentos são indexados previamente, e os trechos mais relevantes para a pergunta são adicionados ao contexto do modelo para fundamentar a resposta.

Com essa estratégia, é possível incorporar dados novos sem novo treinamento. Gao et al. (2024) observam grande redução de erros factuais em comparação com abordagens sem recuperação, superando inclusive métodos de ajuste fino não supervisionados em tarefas intensivas em conhecimento. Em um nicho da radiologia, Wada et al. (2025) reportaram a redução das alucinações identificadas de 8% para 0% ($p = 0{,}012$), além de tempo de resposta melhor que o dos modelos em nuvem estudados. A eficácia do RAG, contudo, varia com o domínio, a qualidade da documentação, a estratégia de segmentação, o modelo de vetorização e o método de avaliação. O ajuste fino foi descartado pela elevada demanda computacional e pela necessidade de reprocessar o modelo a cada atualização do acervo, enquanto o RAG incorpora novos documentos sem alterar o modelo.

No sistema, o conteúdo publicado pelo administrador é convertido em Markdown pela biblioteca Docling (Auer et al., 2025), no serviço de conversão em Python, etapa que influencia a qualidade de todo o processamento subsequente. Os documentos convertidos são segmentados em fragmentos (*chunks*) de 2.000 caracteres, sem sobreposição (*overlap*), granularidade que, nos testes iniciais, equilibrou a preservação do contexto semântico e a quantidade de conteúdo por fragmento. A representação vetorial é gerada pelo modelo all-MiniLM-L6-v2, em versão compatível com a biblioteca Transformers.js (Xenova, 2025), executável no navegador, que produz vetores de 384 dimensões. Os vetores dos fragmentos são armazenados em um arquivo JSON, carregado em memória no navegador do usuário. Assim, tanto a vetorização da consulta quanto a busca linear ocorrem integralmente no lado do cliente

ISSN 2317-1316

(*client-side*), dispensando banco de dados vetorial; ChromaDB, pgvector e FAISS foram descartados por imporem dependências de infraestrutura indesejadas.

Na etapa de inferência, a recuperação é feita de forma linear, com base na similaridade do cosseno. A consulta do usuário é vetorizada com o mesmo modelo de representação, e o vetor resultante é comparado com todos os fragmentos indexados, segundo a Equação (1), para selecionar os quatro mais similares, que passam a compor o contexto fornecido ao modelo de linguagem. O parâmetro de recuperação (top-k = 4) foi estabelecido empiricamente durante a fase de calibração do sistema, mostrando-se adequado para fornecer o contexto necessário sem extrapolar a janela de contexto do modelo de linguagem.

$$sim(q, d) = (q \cdot d) / (\|q\| \|d\|) \quad (1)$$

Na Equação (1), q é o vetor da consulta e d o vetor de um fragmento indexado. Para o volume documental previsto, a busca linear em memória mantém latência aceitável e dispensa a manutenção dos sistemas de busca aproximada, vantajosos apenas em bases muito maiores.

2.4 Persistência e chamadas estruturadas

Para evitar conflitos de disponibilidade, um sistema de reservas deve garantir a integridade e a consistência das transações. Escolheu-se o SQLite, banco de dados relacional de código aberto que dispensa servidor e consome poucos recursos. O MongoDB foi descartado por não oferecer as mesmas garantias transacionais, e o PostgreSQL, embora sólido e válido para futuras melhorias, impõe dependência excessiva de infraestrutura para um protótipo.

A conexão entre o modelo de linguagem e o banco de dados é realizada por chamadas estruturadas compatíveis com o padrão Model Context Protocol (MCP), implementadas com o recurso nativo de *tool calling* do Ollama (Ollama, 2025), sem a execução de um servidor MCP completo. O MCP é um padrão aberto de estrutura cliente-servidor no qual o modelo formula pedidos estruturados a serviços que disponibilizam recursos e ferramentas (Model Context Protocol, 2026). No ACADEMAI, o modelo dispõe de ferramentas para consulta, criação e cancelamento de agendamentos, descritas em JSON com os parâmetros que devem ser solicitados ao usuário. Identificada a intenção, o modelo emite a chamada pela interface do Ollama, e o servidor Express a recebe, encaminha à função apropriada e executa a ação, de modo que validação e execução ocorrem exclusivamente no *backend*. Cada requisição injeta o identificador da sessão ativa, vinculando as ações ao usuário autenticado. O modelo, portanto, nunca escreve diretamente no banco de dados: apenas solicita a ação, cabendo ao servidor

ISSN 2317-1316

validar autorização, regras de negócio, consistência temporal e confirmação explícita antes de qualquer gravação.

2.5 Reconhecimento e síntese de fala

O modelo Whisper, que transcreve áudio em texto, foi treinado por supervisão fraca em larga escala sobre aproximadamente 680 mil horas de gravações em diversos idiomas, o que lhe confere adaptação a diferentes contextos, sotaques e condições acústicas (Radford et al., 2023). A variante *base*, com 74 milhões de parâmetros, convertida para o formato ONNX, foi executada no navegador via Transformers.js, sem envio do áudio a qualquer servidor. Essa variante apresenta taxa de erro por palavra (WER) entre 5,0% e 12,4% em referências em inglês e de 21,9% em português (Radford et al., 2023), com latência compatível com o uso em tempo real em *hardware* padrão. Como essa taxa é relevante para comandos curtos, adotou-se uma salvaguarda: nenhuma reserva é gravada diretamente por voz, pois o sistema sempre exibe na tela um resumo da solicitação para confirmação explícita do usuário. As variantes *small* e *large* são mais precisas, mas excedem os limites de memória e processamento previstos, e a variante *tiny* sacrifica consideravelmente a precisão.

Para a síntese de voz, adotou-se a Web Speech API, interface nativa dos navegadores que dispensa bibliotecas externas, suporta o português brasileiro e elimina a latência de rede. Trata-se da única exceção declarada ao processamento local, pois o texto sintetizado pode ser enviado ao serviço do navegador; para ambientes com exigência estrita de privacidade, prevê-se alternativa local configurável, e o usuário é informado dessa limitação no primeiro uso. O Coqui TTS foi descartado por exigir um serviço adicional, e os serviços em nuvem, por não operarem sem conexão. As tecnologias de cada camada constam na Tabela 1.

Tabela 1: Tecnologias empregadas em cada camada do sistema

Camada	Tecnologia	Responsabilidade
Apresentação	React 19, TypeScript, Vite	Interface *web*, *chat*, calendário e recursos de acessibilidade
Servidor	Node.js 24, Express	Rotas, autenticação, regras de negócio e orquestração
Persistência	SQLite com Prisma ORM	Armazenamento de usuários, recursos e reservas
Modelo de linguagem	Llama 3.2 3B via Ollama	Interpretação conversacional e geração de respostas
RAG – conversão	Docling	Conversão de documentos e segmentação em 2.000 caracteres

ISSN 2317-1316

Camada	Tecnologia	Responsabilidade
RAG – vetores	Transformers.js, all-MiniLM-L6-v2	Representação vetorial e busca semântica dos quatro fragmentos mais similares
Reconhecimento de fala	Whisper *base* (ONNX), local	Transcrição de comandos de voz no navegador
Síntese de fala	Web Speech API	Retorno por voz de confirmações e erros

2.6 Procedimento de avaliação

A avaliação consistiu em um estudo de caso em contexto semelhante ao Fab Lab, laboratório de fabricação digital de uso comunitário da instituição onde a pesquisa foi realizada, que dispõe de impressoras 3D, cortadora a laser e prototipadora de placas de circuito impresso e atende predominantemente a estudantes de engenharia, perfil aderente às demandas de agendamento e de acesso ao conhecimento institucional assumidas pelo projeto.

Os testes foram executados em uma estação de trabalho sem GPU dedicada, com todos os modelos processados em CPU, cuja configuração consta na Tabela 2. O sistema também foi homologado em arquitetura ARM, em *hardware* Apple Silicon M1 com 8 GB de memória unificada, o que evidencia sua portabilidade. Foram cadastrados seis usuários, além do administrador, os equipamentos passíveis de agendamento e quatro documentos institucionais relevantes à realidade do laboratório, como manuais dos equipamentos, normas de utilização e materiais de apoio.

Tabela 2: Configuração do ambiente de teste

Componente	Especificação
Processador	AMD Ryzen 5 4600G with Radeon Graphics – 6 núcleos/12 *threads* (3,7 GHz base)
Memória	32 GB de RAM
Gráficos	Radeon integrado (Vega); modelos executados em CPU, sem GPU dedicada
Configuração geral necessária	**Especificação**
Sistema operacional	Windows 11 (*build* 10.0.26200)
Ambiente de execução	Node.js v24.19.0
Modelo de *embeddings*	all-MiniLM-L6-v2 (384 dimensões)
Modelo de linguagem	Llama 3.2 (3B) via Ollama
Recuperação (RAG)	top-4 fragmentos por consulta

ISSN 2317-1316

Avaliaram-se a condução conversacional do agendamento, a conformidade das respostas com as fontes documentais, o reconhecimento de voz nas condições acústicas do espaço, a prevenção de reservas conflituosas e a adequação dos tempos de resposta. A avaliação funcional foi qualitativa e preliminar, com anotação manual das observações nas sessões. Complementarmente, mediu-se o desempenho: a vetorização de consulta corresponde à média de 10 medições, e os tempos de resposta, às medianas de cinco consultas representativas em três execuções, desconsiderada a carga inicial do modelo de linguagem em memória, evento único a frio.

3 RESULTADOS E DISCUSSÃO

Foram definidas três classes de usuários, com regras de aprovação próprias: a reserva solicitada por um aluno permanece com status PENDENTE até a avaliação de um professor ou administrador, enquanto as feitas por professores ou administradores são aprovadas automaticamente. Esses dois perfis não são autodeclarados no cadastro público, mas atribuídos por um administrador, o que elimina o risco de elevação indevida de privilégio. A verificação de conflito de horário, definida como a sobreposição temporal de dois intervalos com status PENDENTE ou APROVADA para o mesmo recurso, é realizada no servidor com transação e índice explícito, e não é delegada ao assistente.

Na gestão de reservas, o assistente conduziu o agendamento de forma autônoma e interativa: coletou as informações gradualmente, consultou a disponibilidade em tempo real no servidor, apresentou um resumo para confirmação e finalizou a reserva sem intervenção manual. Trata-se de um avanço em relação aos formulários manuais, que exigiam preencher cada campo e consultar o calendário separadamente. Como as sugestões de horários alternativos são geradas pelo servidor e apenas repassadas pelo modelo, não foram observadas propostas de horários inexistentes.

Os testes de desempenho (Tabela 3) indicam tempos compatíveis com o uso conversacional em *hardware* sem GPU dedicada. A vetorização de uma consulta (8,5 ms) corresponde a menos de 1% do tempo mediano de resposta completa (1,65 s), dominado pela geração do modelo de linguagem. A carga inicial desse modelo em memória (aproximadamente 116 s, a frio) ocorre uma única vez e indica a conveniência de mantê-lo residente em memória durante a operação.

ISSN 2317-1316

Tabela 3: Resultados dos testes de desempenho

Métrica	Valor observado	Observação
Fragmentos gerados (base RAG)	5	4 documentos, recorte de 2.000 caracteres
Carregamento do modelo de *embeddings*	≈ 2,6 s	Uma vez, na inicialização
Embedding por fragmento (≈ 2.000 caracteres)	≈ 159 ms	Inclui aquecimento do 1º fragmento
Embedding por consulta	≈ 8,5 ms	Média de 10 medições
Carga inicial do modelo de linguagem	≈ 116 s	A frio, evento único
Latência inicial (mediana)	760 ms	5 consultas representativas, 3 execuções
Tempo de resposta completa (mediana)	1,65 s	5 consultas representativas, 3 execuções

Quanto ao funcionamento do laboratório e às suas diretrizes, o sistema fez bom uso das fontes documentais indexadas, com baixa taxa de alucinação na avaliação qualitativa realizada, e exibiu a fonte sempre que a resposta se apoiou em um documento recuperado. O resultado é coerente com a literatura sobre a fundamentação documental de modelos locais, mas a comparação exige cautela: Wada et al. (2025) atuaram em um campo clínico específico, com protocolo instrumentado e mensuração quantitativa das alucinações, ao passo que este trabalho se baseia em um acervo institucional heterogêneo e em abordagem qualitativa. Ainda assim, a convergência sugere que a redução de erros factuais apontada por Gao et al. (2024) se estende a modelos menores executados localmente.

O reconhecimento de fala mostrou-se funcional nas condições acústicas habituais do ambiente, com transcrição local em tempo real, sem medição quantitativa da taxa de erro por palavra nesta etapa. A síntese de fala forneceu retorno auditivo (*feedback*) para confirmações e correções, permitindo concluir a conversa sem teclado nas funções básicas. Os critérios analisados e as observações registradas constam na Tabela 4.

Tabela 4: Critérios avaliados no estudo de caso e observações registradas

Critério avaliado	Meta ou referência de projeto	Observação registrada
Condução do fluxo de reserva pelo assistente	Coleta, confirmação e registro sem intervenção manual	Fluxo concluído nos cenários testados

ISSN 2317-1316

Critério avaliado	Meta ou referência de projeto	Observação registrada
Aderência das respostas à base documental	Exibição obrigatória da fonte recuperada	Fonte exibida nas respostas apoiadas em documentos; baixa taxa de alucinação (avaliação qualitativa)
Reconhecimento de voz	Transcrição local, sem envio de dados externos	Transcrição local funcional nas condições do espaço (WER não medida)
Prevenção de conflitos de reserva	Bloqueio de sobreposição verificado no servidor	Sobreposições bloqueadas nos casos testados
Aprovação automática por perfil	APROVADA para professor e administrador	Comportamento conforme especificado
Tempo de resposta do assistente	Adequado ao uso conversacional	Medianas: 760 ms (latência inicial) e 1,65 s (resposta completa)

Os resultados devem ser lidos dentro dos limites do estudo. O escopo restrito, com poucos usuários e volume documental limitado, não permite generalizar o comportamento do sistema sob carga concorrente. É razoável conjecturar que a expansão da base indexada e o aumento da demanda resultarão em degradação, tanto pela capacidade limitada de um modelo com três bilhões de parâmetros quanto pelo custo da busca linear em bases mais extensas. A arquitetura, porém, permite substituir o modelo de linguagem por opções mais eficientes sem alterar sua estrutura, o que atenua essa limitação à medida que o *hardware* progride.

Há, ademais, uma restrição considerável quanto ao objetivo de acessibilidade. A interface atende às diretrizes WCAG 2.1, nível AA, e a navegação por comandos de voz opera corretamente; entretanto, ainda não foi efetuada validação com usuários de leitores de tela nem com o público-alvo. A autonomia sem o uso do teclado permanece, portanto, um resultado a ser evidenciado empiricamente, e não uma conclusão deste estudo.

4 CONCLUSÃO

O trabalho cumpriu as metas estabelecidas ao descrever, implementar e analisar a plataforma de código aberto ACADEMAI, assistente inteligente para a administração de agendamentos e o acesso ao conhecimento institucional em laboratórios acadêmicos. Evidenciou-se que é possível utilizar IA executada localmente para coordenar reservas de maneira conversacional e esclarecer dúvidas fundamentadas na documentação da instituição,

ISSN 2317-1316

com exibição da fonte consultada, baixa taxa de alucinação no contexto investigado e tempo mediano de resposta de 1,65 s em *hardware* sem GPU dedicada. A principal contribuição é uma ferramenta de código aberto que permite às instituições de ensino gerenciarem seus espaços sem depender de serviços externos pagos, reduzindo obstáculos à adoção de tecnologia em contextos com limitações financeiras e computacionais. Destacam-se como limitações o escopo restrito do estudo de caso e a ausência de validação da acessibilidade com o público-alvo. Para trabalhos futuros, sugerem-se avaliação sob carga mais representativa do uso real, testes de autonomia com usuários de leitores de tela, otimização do *pipeline* de recuperação com segmentação adaptativa e indexação vetorial mais eficiente e análise da integração com mecanismos de busca da internet, preservando os requisitos de privacidade. Agradece-se ao Professor Sergio Schina de Andrade, orientador, e ao Professor Luis Fernando Pompeo Ferrara, coorientador, pelo acompanhamento da pesquisa, e à Universidade Santa Cecília pela infraestrutura disponibilizada.

5 DECLARAÇÃO DE USO DE IA

Declaração de Uso de IA: Durante a elaboração deste trabalho, utiliza-se a ferramenta Claude (versão Opus 5), na etapa de revisão do manuscrito, com as finalidades de revisão linguística, apoio à organização textual e verificação de consistência entre as seções. O conteúdo gerado foi revisado, editado e validado pelos autores, que assumem total responsabilidade pelo texto final.

6 REFERÊNCIAS

AUER, C. et al. Docling: an efficient open-source toolkit for AI-driven document conversion. In: *AAAI Conference on Artificial Intelligence*, 2025. Disponível em: https://arxiv.org/abs/2501.17887. Acesso em: 3 maio 2026.

FERNANDES, A.; ANDRADE, F.; AMORIM, J.; MANEIRA, M. *AcademAI*. GitHub, 2026. Disponível em: https://github.com/fabriciosuzart/academai. Acesso em: 16 maio 2026.

GAO, Y. et al. *Retrieval-augmented generation for large language models: a survey*. arXiv, 2024. Disponível em: https://arxiv.org/abs/2312.10997. Acesso em: 3 maio 2026.

ISSN 2317-1316

HAUCK, K. et al. Inclusive education with AI: supporting special needs and tackling language barriers. *AI and Ethics*, v. 5, p. 5729-5757, 2025. Disponível em: https://link.springer.com/article/10.1007/s43681-025-00824-3. Acesso em: 2 maio 2026.

IZOURANE, F. et al. Smart campus based on AI and IoT in the era of Industry 5.0: challenges and opportunities. In: CHAKIR, A.; BANSAL, R.; AZZOUAZI, M. (eds.). *Industry 5.0 and Emerging Technologies*. Studies in Systems, Decision and Control, v. 565. Cham: Springer, 2024. Disponível em: https://doi.org/10.1007/978-3-031-70996-8_3. Acesso em: 2 maio 2026.

JAMES, W. et al. Barriers to physics identity development for students with disabilities. In: *Physics Education Research Conference Proceedings*, 2019, Provo. College Park: American Association of Physics Teachers, 2019. p. 229-234. Disponível em: https://doi.org/10.1119/perc.2019.pr.James. Acesso em: 7 jun. 2026.

LEWIS, P. et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In: *Advances in Neural Information Processing Systems*, v. 33, p. 9459-9474, 2020. Disponível em: https://arxiv.org/abs/2005.11401. Acesso em: 3 maio 2026.

META LLAMA. *Model card do Llama 3.2*. GitHub, 2024. Disponível em: https://github.com/meta-llama/llama-models. Acesso em: 16 maio 2026.

MODEL CONTEXT PROTOCOL. *What is the Model Context Protocol (MCP)?* 2026. Disponível em: https://modelcontextprotocol.io/docs/getting-started/intro. Acesso em: 16 maio 2026.

OLLAMA. *Tool calling*. 2025. Disponível em: https://docs.ollama.com/capabilities/tool-calling. Acesso em: 16 maio 2026.

RADFORD, A. et al. Robust speech recognition via large-scale weak supervision. In: *International Conference on Machine Learning*, p. 28492-28518, 2023. Disponível em: https://proceedings.mlr.press/v202/radford23a.html. Acesso em: 3 maio 2026.

ISSN 2317-1316

VASWANI, A. et al. Attention is all you need. In: *Advances in Neural Information Processing Systems*, v. 30, p. 5998-6008, 2017. Disponível em: https://arxiv.org/abs/1706.03762. Acesso em: 3 maio 2026.

WADA, A. et al. Retrieval-augmented generation elevates local LLM quality in radiology contrast media consultation. *npj Digital Medicine*, v. 8, n. 395, 2025. Disponível em: https://doi.org/10.1038/s41746-025-01802-z. Acesso em: 3 maio 2026.

XENOVA. *Model all-MiniLM-L6-v2*. 2025. Disponível em: https://huggingface.co/Xenova/all-MiniLM-L6-v2. Acesso em: 15 jun. 2026.