PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS

ESCOLA POLITÉCNICA E DE ARTES

GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

# AE MESSAGING TOOL — UMA FERRAMENTA WEB PARA AUTOMAÇÃO DE MENSAGENS EM ESCOLAS DE IDIOMAS

FELIPE SOUSA BEZERRA

GOIÂNIA – GO

2026

FELIPE SOUSA BEZERRA

# AE MESSAGING TOOL — UMA FERRAMENTA WEB PARA AUTOMAÇÃO DE MENSAGENS EM ESCOLAS DE IDIOMAS

Trabalho de Conclusão de Curso apresentado à Escola Politécnica e de Artes, da Pontifícia Universidade de Goiás, como parte dos requisitos para obtenção do título de Bacharel em Engenharia da Computação.

Orientador(a): Profª. Drª. Solange da Silva
Banca examinadora:
Prof. Dr. Vicente Paulo de Camargo
Prof. Me. Olegário Correa da Silva Neto

GOIÂNIA – GO

2026

FELIPE SOUSA BEZERRA

# AE MESSAGING TOOL — UMA FERRAMENTA WEB PARA AUTOMAÇÃO DE MENSAGENS EM ESCOLAS DE IDIOMAS

Este Trabalho de Conclusão de Curso julgado adequado para obtenção o título de Bacharel em Engenharia de Computação, e aprovado em sua forma final pela Escola Politécnica e de artes, da Pontifícia Universidade Católica de Goiás, em 09/06/2026.

Banca Examinadora:

_____
Orientadora: Profa. Dra. Solange da Silva

_____
Prof. Dr. Vicente Paulo de Camargo

_____
Prof. Me. Olegário Correa da Silva Neto

GOIÂNIA 2026

## AGRADECIMENTOS

Agradeço, primeiramente, ao meu Pai Celestial, por sua infinita sabedoria, amor e paciência. Foi Ele quem me concedeu forças nos momentos de cansaço, clareza nas horas de dúvida e serenidade para persistir até o fim deste trabalho. Sem Sua orientação constante, nada disso teria sido possível.

Agradeço também à minha esposa, minha companheira de todas as horas, cuja presença e apoio incondicional foram fundamentais nesta caminhada. Ela compreendeu cada momento de ausência, esteve comigo em cada noite mal dormida e celebrou cada pequena conquista ao longo do curso, sempre oferecendo palavras de incentivo e gestos de carinho que renovaram minhas forças. Sua paciência, fé e amor foram combustível essencial para que eu pudesse alcançar este objetivo. Este trabalho é também resultado do seu sacrifício e da sua confiança em mim.

Agradeço, com todo o meu coração, aos meus pais, que mesmo sem terem tido todas as oportunidades, sempre me incentivaram a estudar e acreditaram no poder transformador do conhecimento. Foram eles que me ensinaram, pelo exemplo, o valor do trabalho duro, da honestidade e da perseverança. A eles devo muito do que sou e de tudo o que conquistei até aqui. Agradeço também aos meus sogros, pelo carinho, apoio e pelas palavras de incentivo que me acompanharam ao longo desta jornada, sempre torcendo pelo meu sucesso e acolhendo-me como parte da família. Agradeço ainda à minha irmã, por seu carinho, apoio e por sempre acreditar em mim, mesmo nos momentos mais desafiadores.

Estendo meus agradecimentos à minha orientadora, Professora Dra. Solange da Silva, por sua dedicação, paciência e valiosas orientações durante o desenvolvimento desta pesquisa.

Por fim, agradeço à Pontifícia Universidade Católica de Goiás e a todos os docentes do curso de Engenharia de Computação, por compartilharem conhecimento e inspirarem a busca constante pela excelência e pelo propósito no uso da tecnologia para o bem comum.

## RESUMO

Este trabalho teve como objetivo desenvolver uma aplicação web denominada *AE Messaging Tool*, voltada à automação de mensagens em escolas de idiomas, aplicando práticas consolidadas da Engenharia de *Software* para otimizar o fluxo de comunicação entre coordenação e professores. como agente de transformação digital e otimização de processos em instituições de ensino. Os testes realizados em ambiente real, com a equipe administrativa da escola parceira, evidenciaram ganhos expressivos de eficiência operacional. Observou-se uma redução média de aproximadamente 12 horas semanais no tempo gasto com tarefas manuais de comunicação, além de melhorias significativas na padronização das mensagens, rastreabilidade das informações e confiabilidade do processo de envio. A interface do sistema também apresentou boa aceitação pelos usuários, sendo considerada intuitiva e adequada ao contexto de uso. Dessa forma, conclui-se que o *AE Messaging Tool* atingiu plenamente os objetivos propostos, demonstrando viabilidade técnica, eficiência prática e aplicabilidade real no contexto educacional. A solução desenvolvida não apenas automatiza processos administrativos, mas também contribui para a redução de erros humanos e para a melhoria da organização interna da instituição. Além das contribuições práticas, o trabalho também reforça a importância da aplicação de conceitos de Engenharia de *Software* no desenvolvimento de sistemas reais, evidenciando como a adoção de boas práticas de arquitetura, modelagem e organização de código impacta diretamente na qualidade, manutenibilidade e evolução do produto final.



Palavras Chaves: Automação de mensagens. Aplicação Web. Engenharia de Software. Desenvolvimento de Software.

## ABSTRACT

This work aimed to develop a web application called *AE Messaging Tool*, focused on automating messaging processes in language schools by applying established Software Engineering practices to optimize the communication flow between coordination staff and teachers, acting as a tool for digital transformation and process optimization in educational institutions. The tests carried out in a real-world environment, together with the administrative staff of the partner school, demonstrated significant gains in operational efficiency. An average reduction of approximately 12 hours per week was observed in the time spent on manual communication tasks, in addition to substantial improvements in message standardization, information traceability, and reliability of the sending process. The system interface was also well accepted by users, being considered intuitive and appropriate for its context of use. Therefore, it can be concluded that the *AE Messaging Tool* fully achieved its proposed objectives, demonstrating technical feasibility, practical efficiency, and real applicability in the educational context. The developed solution not only automates administrative processes but also contributes to reducing human errors and improving the institution's internal organization. In addition to its practical contributions, this work also reinforces the importance of applying Software Engineering concepts in the development of real-world systems, highlighting how the adoption of good practices in architecture, modeling, and code organization directly impacts the quality, maintainability, and evolution of the final product.

## LISTA DE FIGURAS

Figura 1 - Tela principal do software desktop desenvolvido com *Qt Creator* ..24
Figura 2 – Diagrama de casos de uso.....26
Figura 3 - Diagrama lógico de banco de dados.....27
Figura 4 - Diagrama de componentes .....29
Figura 5 – Diagrama de classes da aplicação *AE Messaging Tool*.....30
Figura 6 – Diagrama de classes do modelo da aplicação.....31
Figura 7 - Tela de *login* da aplicação *web* com autenticação *Firebase* .....32
Figura 8 - Página inicial de geração de mensagens .....33
Figura 9 - Aba de personalização de *templates* de mensagens .....34
Figura 10 - Aba de registros e consultas ao banco de dados .....35
Figura 11 - Diagrama de sequência do processo de autenticação .....36
Figura 12- Diagrama de sequência do processo de geração de mensagens...37
Figura 13 – Diagrama de sequência do gerenciamento de templates.....38
Figura 14 – Diagrama de sequência do gerenciamento de registros.....39

## LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| *AI* | *Artificial Intelligence* |
| *API* | *Application Programming Interface* |
| *CSS* | *Cascading Style Sheets* |
| *CRUD* | *Create, Read, Update and Delete* |
| *HTML* | *HyperText Markup Language* |
| *IA* | *Inteligência Artificial* |
| *LGPD* | *Lei geral de proteção de dados* |
| *OSC* | *Organizações da Sociedade Civil* |
| *PLN* | *Processamento de Linguagem Natural* |
| *UML* | *Unified Modeling Language* |
| *WEB* | *World Wide Web* |

**SUMÁRIO**

1. INTRODUÇÃO ..... 9

2. Referencial teórico ..... 13

2.1 Definições e conceitos ..... 13

2.1.1 Definição de software ..... 13

2.1.2 Desenvolvimento web ..... 13

2.1.3 Autenticação e segurança em sistemas web ..... 15

2.1.4 Inteligência Artificial ..... 16

2.2 TRABALHOS RELACIONADOS ..... 17

2.2.1 Desenvolvimento de *Software* para Agendamentos de Trabalho Conclusão de Curso ..... 17

2.2.2 Aplicação Web para a Gestão de Beneficiários e Voluntários de Projetos Sociais ..... 18

2.2.3 Desenvolvimento de uma Aplicação Web para Controle de Tarefas na Área de Suporte de TI ..... 19

3. Método ..... 20

4. desenvolvimento da ferramenta *AE Messaging Tool* ..... 22

4.1 histórico do desenvolvimento ..... 22

4.2 Levantamento e Modelagem dos Requisitos ..... 24

4.3 Arquitetura e tecnologias utilizadas ..... 27

4.4 IMPLEMENTAÇÃO DO PROTÓTIPO ..... 30

4.5 testes e validação ..... 39

4.6 desafios e soluções encontradas ..... 40

5. Conclusão ..... 41

Referências ..... 43

APÊNDICEs ..... 45

# 1. INTRODUÇÃO

A automação digital tem se consolidado como um dos principais motores de transformação na sociedade contemporânea, ao substituir atividades manuais e repetitivas por processos tecnológicos mais ágeis e confiáveis. No contexto da tecnologia da informação, essa transformação está associada ao uso de sistemas baseados em *software* que executam tarefas operacionais de forma autônoma, contribuindo para o aumento da produtividade e a redução de erros. Tais sistemas permitem que organizações concentrem seus esforços em atividades estratégicas e de maior valor agregado (Valente, 2022).

No campo da tecnologia da informação, a Engenharia de *Software* constitui uma área essencial voltada à criação e manutenção de sistemas confiáveis, flexíveis e alinhados às demandas do mercado. Segundo Sommerville (2019), a engenharia de *software* é uma disciplina que trata de todos os aspectos da produção de *software* — desde sua concepção inicial até a operação e manutenção —, aplicando princípios de engenharia para garantir qualidade, confiabilidade e eficiência. Dessa forma, ela vai além dos aspectos técnicos do desenvolvimento, incorporando práticas de planejamento, documentação, gestão e melhoria contínua dos processos, de modo a assegurar produtos que atendam efetivamente às necessidades organizacionais e educacionais.

Com o avanço da internet e a consolidação de novas tecnologias de software, o desenvolvimento *World Wide Web* (web) tornou-se um dos principais meios para a criação de soluções acessíveis, escaláveis e interativas. Segundo Valente (2022), a web se firmou como a principal plataforma de desenvolvimento de sistemas devido à sua ubiquidade, ao baixo custo de distribuição e à facilidade de atualização contínua. As aplicações web baseiam-se, em sua maioria, no modelo cliente-servidor, que organiza a comunicação entre a interface do usuário e os serviços de processamento e dados. Essa arquitetura, apoiada em linguagens como *HyperText Markup Language* (*HTML)*, *Cascading Style Sheets* (*CSS)* e *JavaScript*, possibilita a construção de sistemas dinâmicos e conectados, permitindo que organizações de diferentes portes implementem ferramentas de gestão e comunicação de forma eficiente e de amplo alcance, característica essencial em contextos educacionais e corporativos.

No contexto de uma escola de idiomas, a comunicação entre coordenação, professores e alunos constitui um processo contínuo e dinâmico. Mudanças de turmas, ajustes de horários e informações administrativas exigem atualizações frequentes e, muitas vezes, são realizadas manualmente, consumindo tempo significativo dos gestores. A partir de uma análise interna realizada na empresa parceira, considerando a rotina da coordenação e o tempo dedicado às atividades manuais de comunicação, estimou-se que esse processo representa aproximadamente 12 horas de trabalho manual por semana, impactando diretamente a eficiência administrativa e a qualidade do serviço prestado.

Justifica estudar esse tema pois a aplicação web chamada *AE Messaging Tool* surge como uma solução desenvolvida para automatizar a geração e o envio em massa de informações aos professores da escola de idiomas, assegurando que as atualizações cheguem de forma rápida, padronizada e confiável. Ao reduzir o tempo despendido em tarefas repetitivas, a solução não apenas economiza recursos, mas também reforça a importância da integração entre automação, engenharia de *software* e desenvolvimento web na otimização de processos educacionais.

Diante deste contexto, este projeto visa responder à seguinte questão de pesquisa: **Como ancorar práticas de Engenharia de *Software* para desenvolver uma aplicação web de automação de mensagens que seja efetiva, confiável e aceitável para uso em escolas de idiomas?**

Este trabalho tem como objetivo geral desenvolver uma aplicação web de automação de mensagens que seja efetiva, confiável e aceitável para uso em escolas de idiomas, aplicando princípios e técnicas da Engenharia de Software.

Os objetivos específicos são:

- Levantar requisitos funcionais e não-funcionais por meio de entrevistas e análise documental com a equipe administrativa da escola parceira (a escola onde trabalho).
- Projetar a arquitetura do sistema (*front end*, *backend*, banco de dados, fila de mensagens e integração com provedores) e documentá-la.
- Implementar um protótipo funcional com: painel administrativo, criação de campanhas, *templates* de mensagem e integração com pelo menos um provedor de envio (e-mail ou WhatsApp).

- Avaliar a eficiência do sistema medindo o tempo médio gasto em tarefas de mensageria manual versus automatizada na própria escola parceira.
- Avaliar a usabilidade medindo taxa de sucesso em tarefas e tempo para completá-las pela equipe administrativa e coletar satisfação por questionário padronizado.

Espera-se que os resultados deste trabalho possam contribuir com:

- Um protótipo funcional da aplicação web chamada *AE Messaging Tool*, projetado não apenas para atender às demandas imediatas da coordenação de uma escola de idiomas, mas também com arquitetura escalável, capaz de ser expandida para outros departamentos da instituição. A ferramenta incluirá criação de perfis específicos para diferentes setores, *templates* personalizados de mensagens e disparo automatizado para WhatsApp.
- Redução significativa do tempo de trabalho manual, com economia estimada em cerca de 12 horas semanais atualmente despendidas em tarefas repetitivas de comunicação. Esse ganho de eficiência permitirá que gestores e coordenadores direcionem esforços para atividades de maior valor estratégico.
- Métricas objetivas de desempenho, apresentadas em relatório comparativo, incluindo tempo médio de execução das tarefas administrativas de mensageria manual versus automatizada, taxa de sucesso no envio e latência média de entrega.
- Métricas subjetivas de aceitação e usabilidade, como índices de satisfação dos usuários administrativos, facilidade de uso e percepção de confiabilidade do sistema, coletadas por meio de questionários padronizados.
- Com documentações das lições aprendidas e propor melhorias para futura implantação em produção na mesma escola ou em contextos semelhantes.
- Registro histórico de todas as mensagens enviadas, com filtros de consulta para identificação de padrões (ex.: professores frequentemente disponíveis em determinados horários ou com histórico de desempenho em avaliações internas).

- Tradução automática de mensagens para diversos idiomas, facilitando a comunicação em contextos multilíngues.
- Disparo em massa via WhatsApp, garantindo alcance imediato e uniforme para todos os professores de uma só vez.
- Controle e rastreabilidade, permitindo acompanhar status de entrega e resposta das mensagens.
- Conjunto completo de artefatos de Engenharia de Software, incluindo documento de requisitos, diagrama de arquitetura, diagrama de classes, casos de teste, relatório de usabilidade e registro das lições aprendidas, consolidando a pesquisa como referência prática e metodológica.

Esta monografia está estruturada da seguinte forma: o **Capítulo 1** traz a introdução com a questão de pesquisa e os objetivos, além dos resultados esperados. O **Capítulo 2** traz o referencial teórico, sendo que na primeira conceitos e definições e na segunda parte traz trabalhos relacionados. O **Capítulo 3** expõe os materiais e métodos utilizados neste projeto. O **Capítulo 4** mostra levantamento de requisitos, *Unified Modeling Language* (UML) etc e a ferramenta desenvolvida com as suas funcionalidades. Por fim, o **Capítulo 5** traz a conclusão do trabalho.

# 2. REFERENCIAL TEÓRICO

Este capítulo traz o referencial teórico dividido em duas partes: a primeira apresenta os conceitos e definições da área, enquanto a segunda aborda alguns trabalhos relacionados.

## 2.1 DEFINIÇÕES E CONCEITOS

### 2.1.1 Definição de software

De acordo com Sommerville (2019), o *software* é um conjunto de programas de computador e toda a documentação associada, incluindo manuais, bibliotecas e dados de configuração necessários para sua operação. Ele pode ser desenvolvido para atender a um cliente específico (*software* sob medida) ou para um mercado genérico (*software* de prateleira).

A engenharia de *software*, segundo Sommerville (2019), é a disciplina que se preocupa com todos os aspectos da produção de *software* — desde sua concepção até a operação e manutenção —, aplicando princípios de engenharia para garantir confiabilidade, qualidade e eficiência no desenvolvimento. Essa área surgiu formalmente na década de 1960, em resposta à chamada crise do *software*, quando projetos começaram a se tornar excessivamente complexos, caros e difíceis de manter.

Sommerville também destaca que o *software* é essencial para o funcionamento da sociedade moderna, estando presente em sistemas industriais, financeiros, de comunicação, transporte e entretenimento. A ausência de limitações físicas no *software* o torna potencialmente ilimitado, mas também mais suscetível à complexidade e a falhas caso não sejam aplicadas práticas adequadas de engenharia.

### 2.1.2 Desenvolvimento web

O desenvolvimento web consiste na criação de sistemas e aplicações acessíveis por meio da Internet, integrando tecnologias, linguagens e práticas de engenharia de *software* voltadas à construção de soluções escaláveis e de fácil manutenção. Conforme Valente (2022), a web consolidou-se como a principal

plataforma de desenvolvimento de *software* devido à sua ubiquidade, ao baixo custo de distribuição e à possibilidade de atualização contínua das aplicações.

As aplicações web são estruturadas, em sua maioria, sobre o modelo cliente-servidor, no qual o cliente — geralmente um navegador — solicita recursos e serviços a um servidor remoto. O servidor, por sua vez, processa as requisições e retorna respostas, frequentemente em formato HTML, CSS e JavaScript, linguagens que compõem a base da camada de apresentação (Valente, 2022). O HTML define a estrutura e o conteúdo da página, o CSS define o estilo e a aparência visual, e o JavaScript introduz interatividade e comportamento dinâmico, permitindo uma experiência de uso mais fluida e responsiva.

De acordo com Valente (2022), o desenvolvimento web moderno transcende a simples construção de páginas estáticas, incorporando conceitos de arquitetura de *software* que organizam os sistemas em componentes de maior nível, como módulos, camadas e serviços. Essa organização arquitetural representa decisões fundamentais de projeto, responsáveis por definir a estrutura do sistema e suas principais interações, sendo determinantes para sua manutenção e evolução ao longo do tempo.

Nesse contexto, a arquitetura em camadas destaca-se como uma das abordagens mais utilizadas, pois permite dividir o sistema em partes com responsabilidades bem definidas, reduzindo a complexidade e facilitando o desenvolvimento e a manutenção. Em aplicações web, essa divisão é comumente representada pelas camadas de apresentação, lógica de negócio e persistência de dados, que podem operar de forma distribuída entre cliente e servidor.

A separação entre *front end* (lado do cliente) e *backend* (lado do servidor) é uma consequência direta dessa organização em camadas. Enquanto o f*ront en*d é responsável pela interface e interação com o usuário, o *backend* concentra o processamento das regras de negócio, acesso a banco de dados e integração com serviços externos. Essa separação promove maior desacoplamento entre os componentes do sistema, facilitando sua evolução, reutilização e escalabilidade.

Para que sistemas heterogêneos possam integrar-se nesse contexto, utilizam-se interfaces de programação de aplicações (APIs). Uma API é uma interface que expõe parte da lógica ou dos dados de um sistema para uso de outros sistemas, de forma padronizada. Ao expor suas funcionalidades, uma API amplia a produtividade

de desenvolvedores internos e externos e pode agregar valor ao negócio (SACRAMENTO et al., 2022). APIs web são comumente modeladas no estilo REST, que impõe restrições de arquitetura (como o uso de verbos HTTP e recursos identificados por URLs) com vistas à simplicidade, escalabilidade, portabilidade e performance (SACRAMENTO et al., 2022). De acordo com os autores, uma API bem projetada e documentada torna-se vantagem competitiva ao permitir o acesso a dados e serviços de forma segura e padronizada.

No contexto deste trabalho, o desenvolvimento web é utilizado como base tecnológica para a implementação do sistema *AE Messaging Tool*, que emprega a arquitetura cliente-servidor e princípios de organização em camadas para automatizar o envio e a tradução de mensagens em escolas de idiomas. Essa abordagem possibilita que o sistema seja acessível a partir de qualquer dispositivo conectado à internet, garantindo maior flexibilidade, escalabilidade e eficiência operacional.

### 2.1.3 Autenticação e segurança em sistemas web

A segurança cibernética tornou-se elemento central na economia atual, sendo essencial para proteger dados e serviços expostos na Internet. De acordo com Albesher (2023), criar um ambiente online sustentável requer que as aplicações adotem procedimentos de autenticação capazes de evoluir com as tecnologias utilizadas pelos usuários. O autor destaca que proteger os utilizadores inclui aplicar os métodos de autenticação mais eficazes e que esses procedimentos estão em constante mudança devido aos avanços tecnológicos. Com o crescimento do uso da Internet, aumentou também o número de contas que exigem senha: enquanto há 15 anos o usuário médio possuía cerca de 25 contas com senha, hoje esse número ultrapassa 80. Tal cenário eleva o risco de vazamentos e o esgotamento do usuário, tornando os métodos tradicionais de senha isolada insuficientes; por isso, recomenda-se o uso de autenticação em múltiplos fatores e melhorias no processo de recuperação de senhas.

Do ponto de vista conceitual, a segurança da informação baseia-se nos pilares da confidencialidade, integridade e disponibilidade. Segundo Espedito et al. (2021), a confidencialidade garante que apenas pessoas autorizadas acessem os dados, a integridade assegura que os dados não sejam alterados sem permissão e a

disponibilidade garante que a informação esteja acessível quando necessária. Além desses aspectos técnicos, a segurança engloba fatores humanos e éticos, como destaca Dhillon (apud ESPEDITO et al., 2021), e normas como a ISO/IEC 27002 orientam políticas e controles para mitigar vulnerabilidades. O crescimento de serviços digitais e a entrada em vigor da Lei Geral de Proteção de Dados (LGPD) no Brasil aumentaram a exigência de mecanismos robustos de autenticação e de proteção de dados pessoais.

Os mecanismos de autenticação podem ser classificados em três categorias: (i) baseados em conhecimento, como senhas ou perguntas secretas; (ii) baseados em posse, como tokens físicos ou aplicativos geradores de códigos; e (iii) baseados em características biométricas, como impressão digital, reconhecimento facial ou de íris. Espedito et al. (2021) observam que métodos biométricos ganharam popularidade, mas ainda coabitam com senhas tradicionais devido a fatores como custo e preferência dos usuários. A combinação de fatores (autenticação multifator) aumenta a segurança ao exigir que o invasor comprometa mais de um tipo de credencial.

O estudo de Albesher (2023) alerta ainda que procedimentos de autenticação inadequados levam usuários a contorná-los, diminuindo a segurança. Para mitigar esse problema, o autor recomenda projetar mecanismos de autenticação que equilibrem segurança e usabilidade, adotando, por exemplo, autenticação em duas etapas como obrigatório quando apropriado. Em resumo, a literatura recente aponta que a proteção de sistemas web depende de combinar múltiplos fatores de autenticação, seguir padrões e normas de segurança da informação e considerar a experiência do usuário na concepção dos processos.

### 2.1.4 Inteligência Artificial

A Inteligência Artificial (IA) é um dos campos mais relevantes da ciência da computação contemporânea, voltado à criação de sistemas capazes de compreender, aprender e reproduzir comportamentos considerados inteligentes. Segundo Kaufman (2024), os fundamentos da IA estão enraizados na lógica e na matemática, que fornecem a estrutura formal para representar o raciocínio humano em linguagem computacional. A autora enfatiza que a IA deve ser compreendida não apenas como um conjunto de técnicas computacionais, mas como um fenômeno sociotécnico que

combina algoritmos, dados e valores humanos, influenciando diretamente a forma como indivíduos e organizações interagem com a informação e o conhecimento.

Ao longo das últimas décadas, a IA evoluiu de abordagens simbólicas, baseadas em regras e inferências lógicas, para modelos conexionistas, sustentados por redes neurais artificiais e técnicas de aprendizado de máquina. Conforme Lima, Miranda e Farias (2025), as redes neurais são compostas por camadas de unidades interligadas que ajustam seus parâmetros matematicamente, permitindo que o sistema aprenda a partir da experiência. Esse mecanismo de aprendizado é a base de diversas aplicações modernas, como reconhecimento de fala, visão computacional e processamento de linguagem natural (PLN).

Entre as áreas mais transformadas pela IA está o processamento de linguagem natural, responsável por possibilitar que sistemas compreendam e gerem textos em linguagem humana. Essa vertente combina técnicas linguísticas e estatísticas para permitir que algoritmos traduzam, interpretem e adaptem mensagens a diferentes contextos linguísticos e culturais (Lima; Miranda; Farias, 2025).

No contexto deste trabalho, a Inteligência Artificial é aplicada especificamente para a tradução automática de *templates* de mensagens, permitindo o envio de comunicações em múltiplos idiomas dentro do ambiente web. Essa funcionalidade amplia o alcance e a acessibilidade da ferramenta *AE Messaging Tool*, tornando a comunicação mais eficiente e inclusiva entre instituições multilíngues. Assim, a IA atua como elemento integrador e facilitador, promovendo a personalização das interações e reforçando o papel da Engenharia de *Software* na construção de soluções tecnológicas centradas na comunicação humana.

## 2.2 TRABALHOS RELACIONADOS

Esta seção apresenta trabalhos acadêmicos relevantes na área de desenvolvimento de web.

### 2.2.1 Desenvolvimento de *Software* para Agendamentos de Trabalho Conclusão de Curso

O trabalho de Souza (2025) teve como objetivo geral desenvolver uma aplicação web para automatizar o processo de agendamento de TCC da Pontifícia Universidade Católica de Goiás, oferecendo interface acessível a professores e administradores e evitando conflitos de horários e sobreposição de salas.

Realizou-se o levantamento de requisitos com *stakeholders*, projetou a arquitetura cliente-servidor e implementou a aplicação usando tecnologias web modernas (Next.js/*TypeScript* no *front end*; Nest.js, Prisma e PostgreSQL no *backend*. Foram modelados casos de uso, construído o banco de dados, implementadas telas de autenticação/gestão de usuários e realizadas validações e testes básicos do protótipo.

O autor concluiu que a aplicação desenvolvida atingiu os objetivos propostos, automatizando o processo de agendamento de apresentações acadêmicas e promovendo maior organização, controle e eficiência. O estudo também destacou a versatilidade do desenvolvimento web e a viabilidade de integração entre tecnologias modernas e demandas institucionais, indicando espaço para melhorias futuras (ex.: melhor gerenciamento de salas com projetores e integração com planilhas na nuvem).

### 2.2.2 Aplicação Web para a Gestão de Beneficiários e Voluntários de Projetos Sociais

O trabalho de Silva (2024) teve como objetivo geral desenvolver o aplicativo web Gestão Solidária, visando facilitar o acesso e a gestão das informações de Organizações da Sociedade Civil (OSC), melhorando a eficiência dos processos administrativos relacionados a beneficiários e voluntários.

Implementou-se uma aplicação web com módulos iniciais para beneficiários e voluntários, arquitetura limpa e modelos relacionais para autenticação/autorização; a solução centraliza dados, permite buscas e filtros personalizáveis, operações *Create, Read, Update and Delete* (CRUD) e controle de acesso. Tecnologias citadas no trabalho incluem ASP.NET *Core*, *Blazor* e Microsoft SQL Server. O desenvolvimento contemplou justificativa, modelagem, implementação e sugestões de evolução funcional.

O autor concluiu que a plataforma tem potencial para reduzir redundâncias, digitalizar processos administrativos das OSC e melhorar a precisão dos dados,

contribuindo para a gestão mais informada e eficaz dessas organizações. O trabalho aponta benefícios práticos e sugere extensões (ex.: controle de atividades dos voluntários, processamento de doações, relatórios personalizados) para aumentar o impacto social e econômico.

### 2.2.3 Desenvolvimento de uma Aplicação Web para Controle de Tarefas na Área de Suporte de TI

O trabalho de Oliveira (2023) teve como objetivo geral desenvolver uma aplicação web de controle de tarefas para uso na área de suporte de TI, tendo como referência o *software* livre GLPI, com foco em atribuição de atividades, definição de prioridades e acompanhamento de ordens de serviço.

Estudou-se linguagens e frameworks web (HTML5, CSS3, JavaScript, Angular, Node.js, MongoDB), realizou elicitação e documentação dos requisitos (Especificação de Requisitos do Software) e implementou o protótipo da aplicação de gestão de chamados. O sistema suporta criação/edição/consulta de chamados, tarefas pendentes, soluções registradas, avaliação e geração de ordens de serviço — além de funcionalidades de autenticação e gestão de permissões.

O autor concluiu que a aplicação desenvolvida melhora a gestão de chamados na área de suporte, aumentando a rastreabilidade e reduzindo falhas de comunicação entre equipe e solicitantes. O trabalho também enfatiza os ganhos de aprendizado na documentação e na implementação de um sistema com funcionalidades completas para controle de tarefas em TI, apontando limitações impostas por tempo e recursos e sugerindo melhorias futuras.

## 3. MÉTODO

Quanto à natureza é uma pesquisa secundária (Wazlawick, 2022).

Quanto aos objetivos esta pesquisa é exploratória. “As pesquisas exploratórias têm como propósito proporcionar maior familiaridade com o problema, com vistas a torná-lo mais explícito ou a construir hipóteses” (Gil, 2022).

Quanto aos procedimentos técnicos, esta pesquisa é bibliográfica e experimental. A pesquisa bibliográfica, de acordo com Gil (2017), é elaborada com base em material já publicado, incluindo livros, artigos científicos, teses, dissertações e anais de eventos. Essa abordagem permite o embasamento teórico necessário para o desenvolvimento do projeto.

As etapas da pesquisa bibliográfica, segundo Gil (2017) incluem:

a) Escolha do tema: Desenvolvimento de uma aplicação web para automação de mensagens em escolas de idiomas (*AE Messaging Tool*).
b) Levantamento bibliográfico preliminar: levantamento bibliográfico em artigos, trabalhos acadêmicos e livros da área de Engenharia de *Software* e Comunicação Digital, com o intuito de compreender os fundamentos de automação de mensagens e apoiar a definição do problema de pesquisa.
c) Formulação do problema: Como ancorar práticas de Engenharia de *Software* para desenvolver uma aplicação web de automação de mensagens que seja efetiva, confiável e aceitável para uso em escolas de idiomas?
d) Busca das fontes: Fontes em artigos científicos disponíveis em bases como *Google Scholar* e *SciELO*, além de livros da área obtidos em bibliotecas físicas e digitais. Essas fontes serviram de apoio para a formulação do problema de pesquisa e para a compreensão dos processos envolvidos.
e) Leitura do material: identificação de conceitos, técnicas e práticas correlacionadas ao tema da pesquisa, com foco em requisitos de software, usabilidade, segurança e mensageria digital.
f) Fichamento: fichamento de citações e conceitos relevantes extraídos do material bibliográfico selecionado, de modo a subsidiar a fundamentação teórica do trabalho.

g) Organização lógica: organização das ideias levantadas com o propósito de atender aos objetivos propostos.
h) Redação do texto: escrita do TCC.

Pesquisas experimentais consistem em determinar o objeto de estudo, selecionar variáveis capazes de influenciá-lo e definir as formas de controle e de observação dos efeitos que a variável produz no objeto de estudo (Gil, 2017).

Pesquisas experimentais apresentam as seguintes etapas segundo Gil (2017):

a) Formulação do problema: Como o desenvolvimento de um sistema web de automação de mensagens pode otimizar o processo de comunicação em uma escola de idiomas?
b) Definição do plano experimental: Foi utilizado o plano de única variável, aplicado a um único grupo de indivíduos (equipe administrativa da escola parceira). A observação foi feita de forma sequencial após cada versão/teste do sistema.
c) Determinação do ambiente: O ambiente utilizado foi virtual. Utilizando um notebook Dell G15*:* CPU: Intel i5 13450HX, RAM:16GB e GPU: Nvidia RTX 3050 6GB, Sistema Operacional: Windows 11 Pro, a IDE Visual Studio Code, Controle de Versão: GitHub.
d) Coleta de dados: A coleta foi realizada por meio da medição do tempo gasto em tarefas de mensageria manual versus automatizada, além de entrevistas e questionários para verificar a satisfação da equipe administrativa com o sistema.
e) Análise e interpretação dos dados: A análise foi feita comparando os tempos de execução e a percepção dos usuários. O modelo automatizado apresentou maior eficiência e foi aceito como o mais adequado.
f) Redação do relatório: Escrita do TCC2.

# 4. DESENVOLVIMENTO DA FERRAMENTA *AE MESSAGING TOOL*

Este capítulo descreve o processo de desenvolvimento do *AE Messaging Tool*, desde sua concepção inicial até a implementação atual como aplicação web. São abordados os requisitos levantados, a arquitetura definida, as tecnologias utilizadas, os módulos implementados, os testes realizados e os resultados obtidos. Além disso, apresenta-se o histórico do desenvolvimento, evidenciando como a aplicação evoluiu de uma solução experimental em desktop para uma ferramenta robusta de automação de fluxos de comunicação padronizada entre a coordenação e os professores.

## 4.1 HISTÓRICO DO DESENVOLVIMENTO

A *Academia Europea* é uma escola de idiomas com sede em El Salvador fundada em 1969. Atualmente conta com setenta e duas filiais espalhadas em sete países da américa central. Além da modalidade presencial, a escola também oferece a modalidade virtual, e para esta, conta com aproximadamente seiscentos professores localizados em diferentes partes do mundo. Esta ampla estrutura gera uma alta demanda operacional: a equipe de coordenação precisa gerenciar cerca de cem novas designações de turmas para professores por semana, exigindo a constante troca de informações, envio de instruções e confirmação das designações.

Foi nesse contexto que surgiu a ideia do *AE Messaging Tool*, ainda no terceiro período do curso de Engenharia de Computação, quando o autor atuava como coordenador dessa instituição de ensino. Na época, grande parte do tempo de trabalho era consumida por tarefas administrativas repetitivas, especialmente a comunicação de alterações de professores e turmas, o que gerava sobrecarga e aumentava o risco de falhas humanas.

Com o objetivo de reduzir esse esforço manual, iniciou-se o desenvolvimento de um *software desktop* em C++, utilizando o *Qt Creator*, conforme mostrado na Figura 1.


Figura 1 – Tela principal do *software* desktop desenvolvido com Qt Creator

Fonte: Autoria própria

No início o projeto explorava conceitos de programação orientada a objetos, criação de classes e integração com banco de dados local. Ao longo de aproximadamente dois anos, a aplicação foi continuamente aprimorada para atender às novas demandas da coordenação, alcançando uma economia de até nove horas de trabalho manual por semana.

Apesar de eficiente, o sistema desktop tinha limitações significativas: rodava apenas em Windows, não permitia acesso remoto e exigia que as mensagens fossem geradas em lotes separados (por tipo), o que ainda demandava certo esforço manual da coordenação.

No ano de 2025, com o amadurecimento acadêmico e a consolidação de novos conhecimentos em desenvolvimento web, foi realizada uma reestruturação completa, resultando em uma aplicação web multiplataforma acessível em qualquer dispositivo conectado à Internet. A nova versão trouxe ganhos ainda maiores, economizando até 12 horas de trabalho semanal, principalmente porque, além de gerar todas as

mensagens de todos os tipos de uma só vez, passou a enviá-las diretamente ao WhatsApp de cada professor por meio da API oficial da Meta.

Assim, o *AE Messaging Tool* evoluiu de um protótipo restrito e limitado a uma estação de trabalho para uma ferramenta completa, robusta e escalável de automação da comunicação administrativa.

## 4.2 LEVANTAMENTO E MODELAGEM DOS REQUISITOS

O levantamento de requisitos foi conduzido por meio de observação direta do trabalho da coordenação e conversas com a equipe administrativa da escola parceira. A partir desse processo, foram identificadas as principais necessidades operacionais relacionadas à comunicação interna entre coordenação e professores. Os requisitos funcionais e não funcionais levantados foram posteriormente organizados e documentados no Documento de Requisitos de Software, apresentado no Apêndice A deste trabalho, servindo como base para a modelagem e implementação da aplicação AE Messaging Tool. Foram identificados:

- Requisitos funcionais:
    - Criação e envio de fluxos de comunicação padronizada (substituição temporária, designação de novo professor permanente, desligamento de grupo, entre outros).
    - Envio automático das mensagens para o WhatsApp individual de cada professor por meio da API oficial da Meta.
    - Registro detalhado de todas as mensagens enviadas.
    - Sistema de confirmação de recebimento e aceitação por parte do professor.
    - Consulta com filtros (por professor, turma, período).
    - Tradução automática dos *templates*, quando necessário, utilizando a API do *MyMemory*.
- Requisitos não funcionais:
    - Escalabilidade, para permitir inclusão de novos tipos de mensagens e fluxos.
    - Confiabilidade, garantindo que todas as mensagens cheguem corretamente ao destino.

- Usabilidade, com interface clara para que a coordenação configure e envie comunicações rapidamente.
- Segurança, com autenticação de usuários via *Firebase Authentication* (login por e-mail e senha).
- Compatibilidade multiplataforma, acessível em navegadores modernos de qualquer dispositivo.

A modelagem dos requisitos foi representada por *user stories* e diagramas UML. A Figura 2 apresenta o Diagrama de Casos de Uso, que ilustra as principais interações entre os atores e o sistema *AE Messaging Tool*, permitindo compreender as funcionalidades oferecidas e as responsabilidades do usuário.

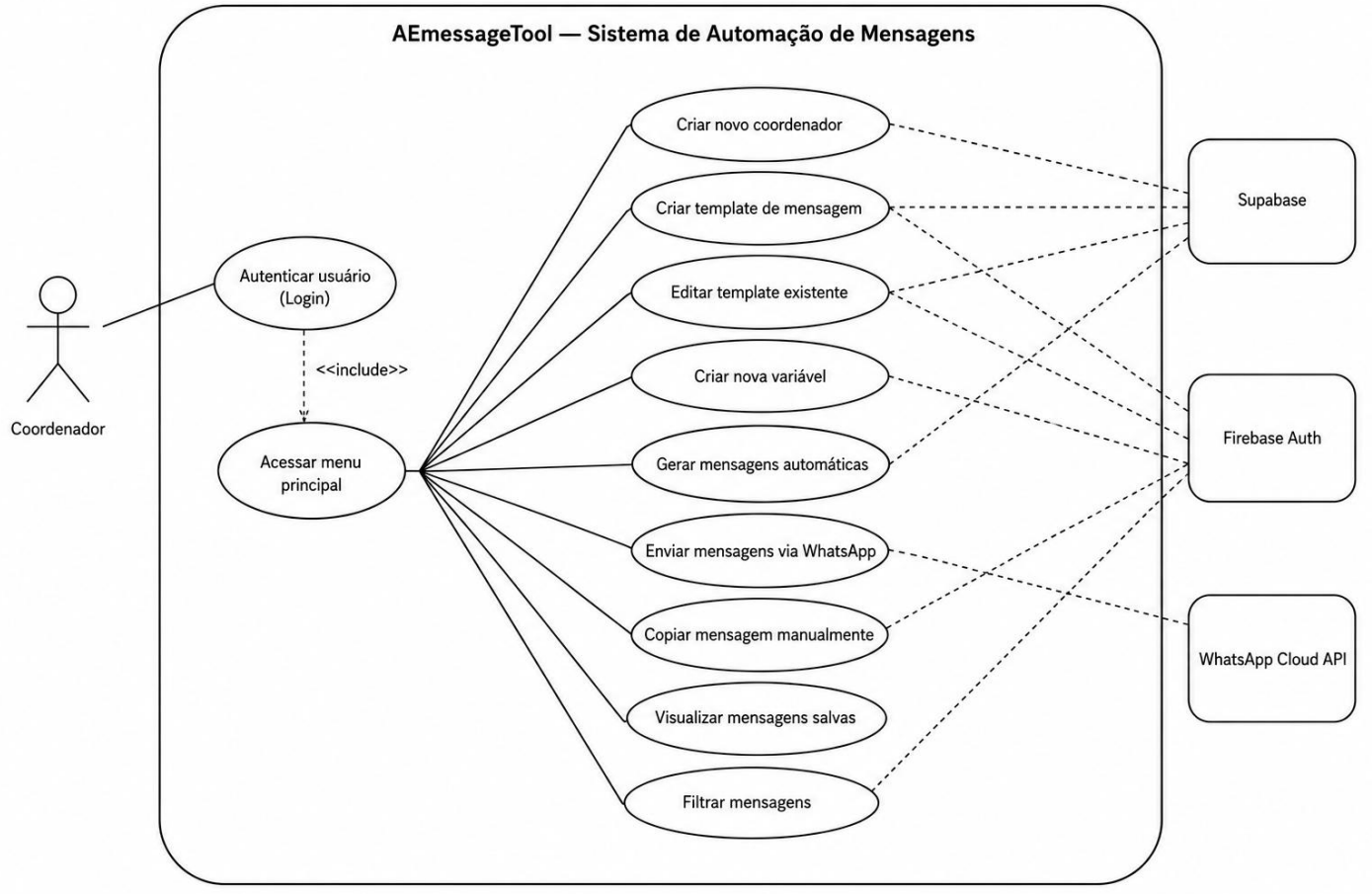


Figura 2 – Diagrama de casos de uso do *AE Messaging Tool*

Fonte: Autoria própria

Em seguida, a Figura 3 apresenta o Diagrama Lógico do Banco de Dados, que descreve a estrutura lógica das tabelas implementadas no *Supabase*, utilizando o *PostgreSQL* como sistema gerenciador de banco de dados. O diagrama evidencia as entidades, atributos e relacionamentos utilizados para o armazenamento das informações da aplicação, como templates, variáveis e mensagens geradas. Ressalta-se que a autenticação dos usuários não é representada como tabela nesse diagrama, pois esse controle é realizado por meio do *Firebase Authentication*, serviço externo responsável pelo cadastro, autenticação e gerenciamento dos coordenadores que acessam o sistema.

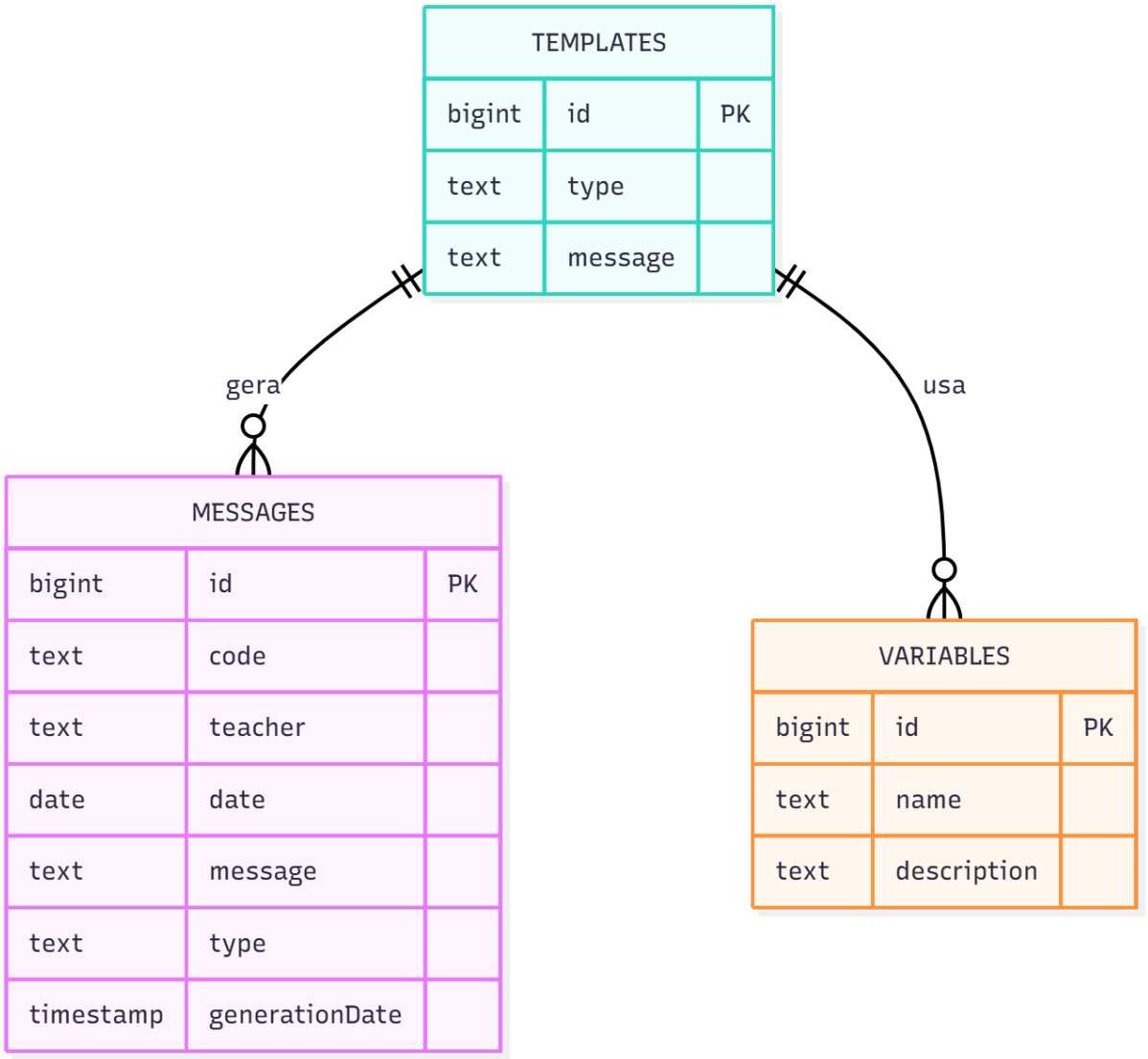


Figura 3 – Diagrama lógico de banco de dados do *AE Messaging Tool*

Fonte: Autoria própria

## 4.3 ARQUITETURA E TECNOLOGIAS UTILIZADAS

A arquitetura do AE Messaging Tool foi desenvolvida seguindo princípios da programação orientada a objetos e arquitetura em camadas, com o objetivo de promover modularização, reutilização de código, separação de responsabilidades e maior facilidade de manutenção e escalabilidade do sistema.

A aplicação foi estruturada em diferentes componentes especializados, responsáveis pela interface do usuário, processamento das regras de negócio, persistência de dados e integração com serviços externos. Essa abordagem permitiu organizar o sistema de forma desacoplada, facilitando futuras expansões e a implementação de novas funcionalidades.

A arquitetura do AE Messaging Tool contempla:

- *Front end*: desenvolvido em *React (TypeScript*), responsável pela interface gráfica intuitiva.
- *Backend*: construído em Node.js, responsável pelas regras de negócio, integração com APIs e verificação de confirmações.
- Banco de Dados: Supabase (PostgreSQL), para armazenar fluxos, registros de mensagens, confirmações e histórico.
- Integração de Mensagens: envio automático via API oficial da Meta para WhatsApp.
- Tradução Automática: integração com a *Application Programming Interface* (API) do *MyMemory* para tradução dos *templates* em diferentes idiomas, sempre que necessário.
- Autenticação: gerenciada pelo *Firebase Authentication*, utilizando login com e-mail e senha.
- Hospedagem: em nuvem, possibilitando acessibilidade, escalabilidade e disponibilidade contínua.

A Figura 4 ilustra o Diagrama de Componentes da aplicação, representando os principais módulos do sistema e as interações entre interface, serviços, banco de

dados e APIs externas utilizadas no processo de autenticação, tradução e envio automatizado de mensagens.

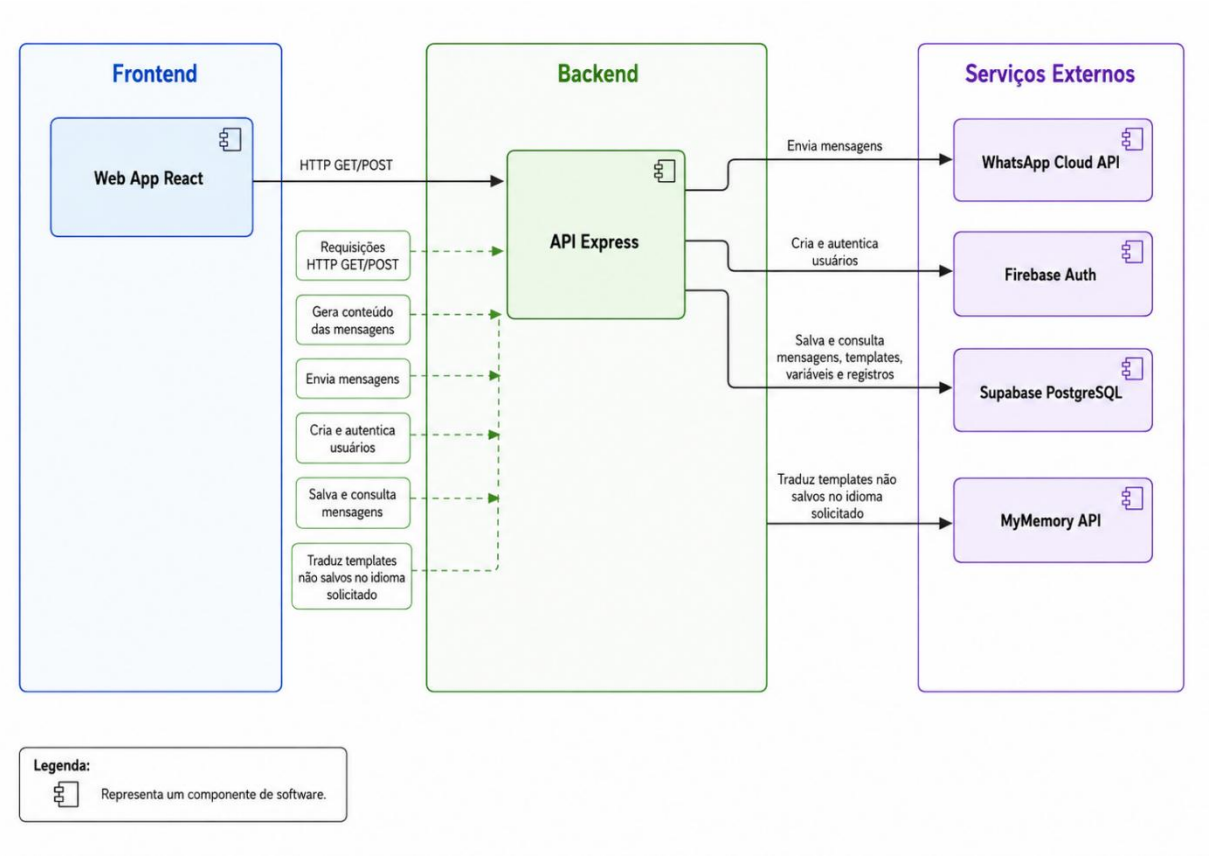


Figura 4 – Diagrama de componentes da aplicação *AE Messaging Tool*

Fonte: Autoria própria

A Figura 5 apresenta o Diagrama de Classes da aplicação AE Messaging Tool, evidenciando a organização estrutural dos principais componentes responsáveis pelas regras de negócio, manipulação de dados e interação entre as camadas do sistema. O diagrama representa classes relacionadas aos serviços da aplicação, entidades de domínio, repositórios e componentes de interface, demonstrando as dependências e relacionamentos existentes entre eles.

A modelagem foi desenvolvida seguindo princípios da programação orientada a objetos, permitindo maior modularização, reutilização de código e separação de responsabilidades entre os componentes do sistema. Além disso, o diagrama evidencia a arquitetura baseada em camadas adotada na aplicação, na qual a interface em *React* se comunica com serviços especializados responsáveis pelo processamento das regras de negócio e pela persistência das informações.

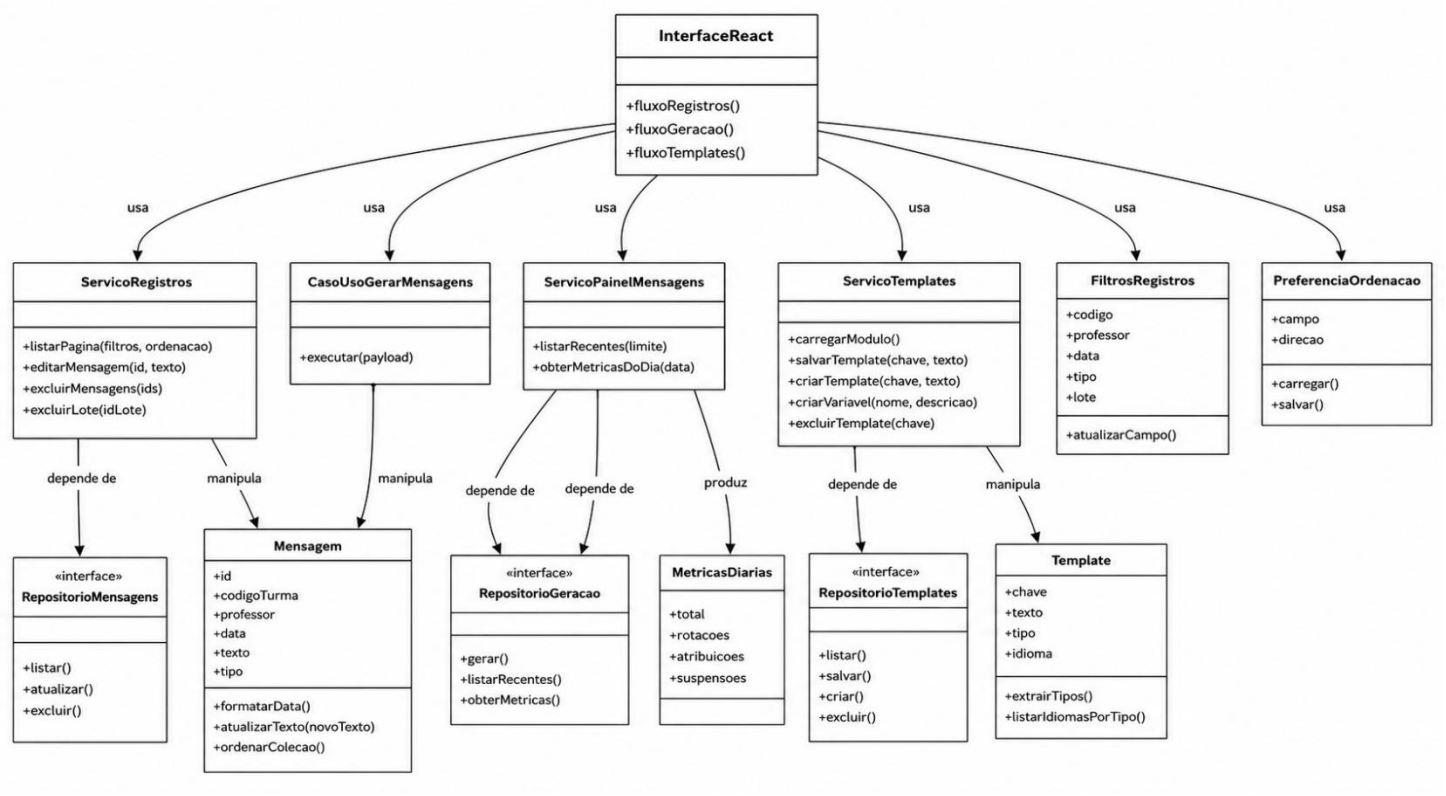


Figura 5 – Diagrama de classes da aplicação *AE Messaging Tool*

Fonte: Autoria própria

A Figura 6 apresenta o diagrama de classes do modelo da aplicação *AE Messaging Tool*, evidenciando as principais entidades persistidas no banco de dados e os serviços responsáveis por sua manipulação. Diferentemente do diagrama de classes geral da aplicação, esta representação possui foco na camada de persistência e na estrutura lógica dos dados armazenados pelo sistema.

O diagrama demonstra a relação entre a interface da aplicação desenvolvida em *React* e os serviços responsáveis pelas operações de listagem, criação, edição e exclusão dos registros. Também são apresentadas as entidades *Message*, *Template* e *Variable*, que correspondem diretamente às tabelas do banco de dados utilizadas pela aplicação. Essa modelagem contribui para a organização do sistema, promovendo maior separação de responsabilidades, facilidade de manutenção e escalabilidade da aplicação, princípios amplamente recomendados pela Engenharia de *Software* orientada a objetos.

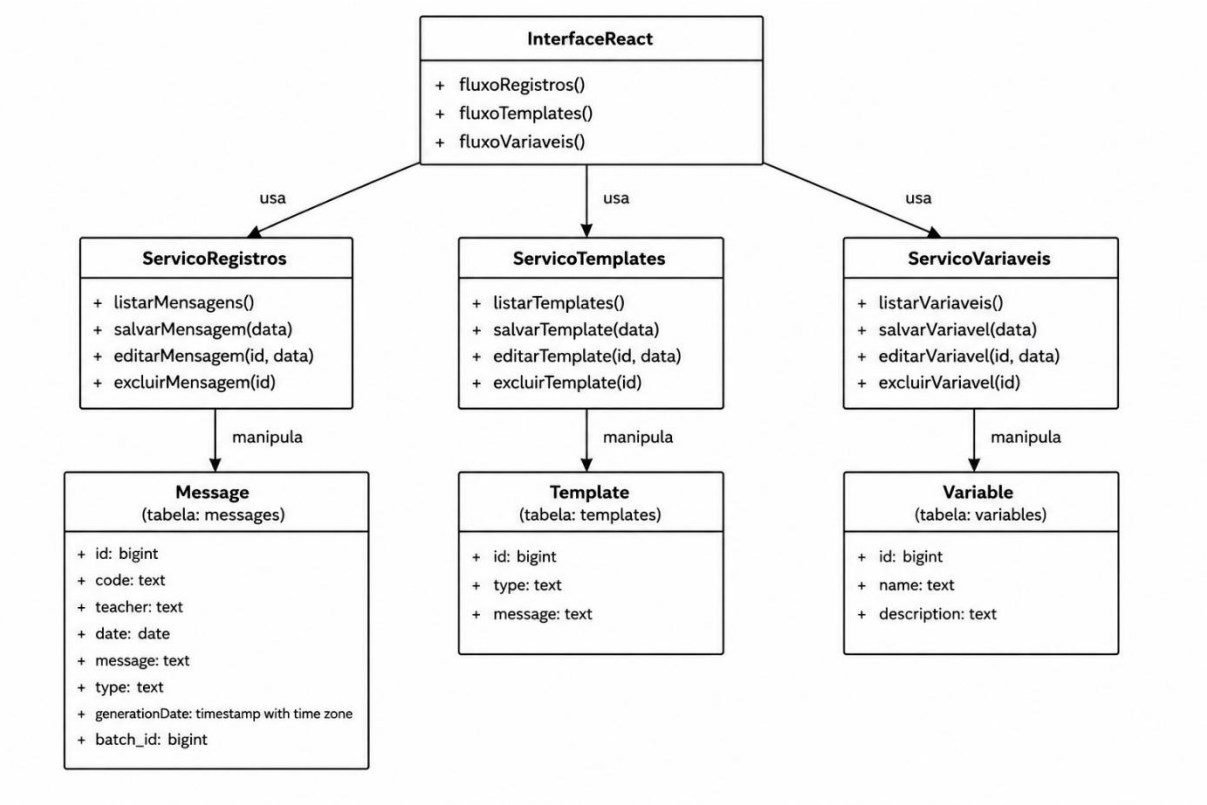


Figura 6 – Diagrama de classes do modelo da aplicação *AE Messaging Tool*

Fonte: Autoria própria

## 4.4 IMPLEMENTAÇÃO DO PROTÓTIPO

A versão atual do *AE Messaging Tool* foi desenvolvida como uma aplicação web completa, contemplando diferentes módulos que atendem às necessidades operacionais da equipe administrativa. O sistema foi projetado com foco em usabilidade, desempenho e integração com serviços externos, permitindo a automação eficiente dos fluxos de comunicação. A seguir, são apresentadas as principais interfaces do sistema, juntamente com a descrição de suas funcionalidades.

A Figura 7 apresenta a tela de autenticação do sistema, responsável por controlar o acesso dos usuários à aplicação.

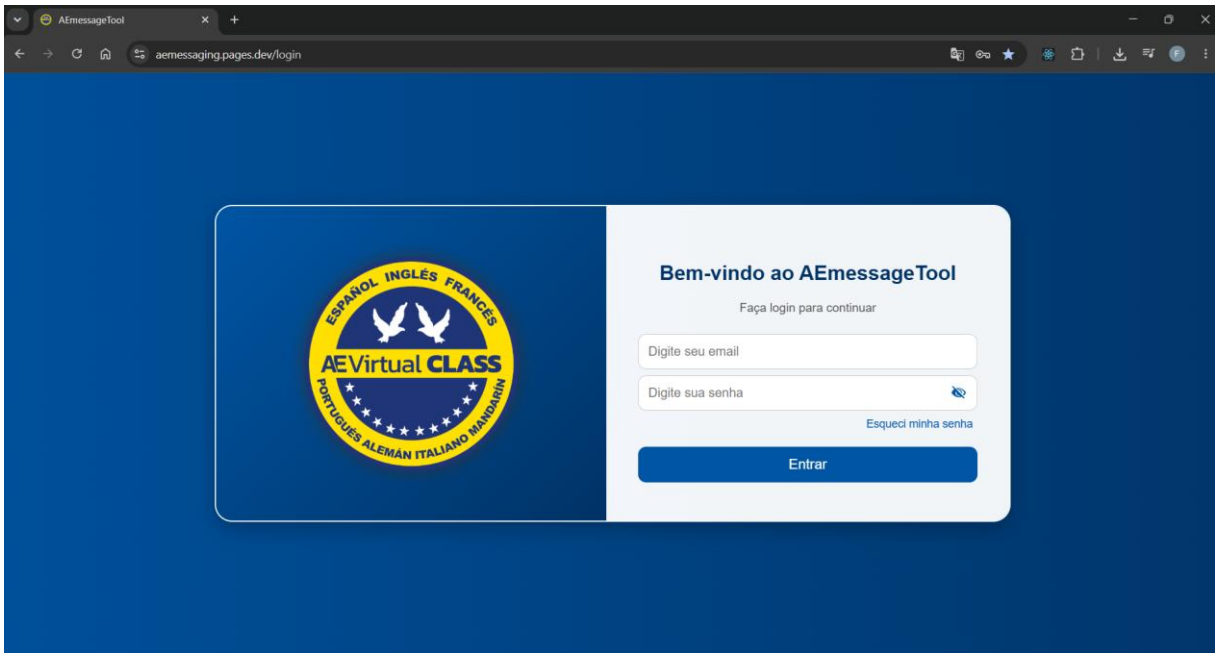


Figura 7 – Tela de login da aplicação web com autenticação Firebase

Fonte: Autoria própria

A interface foi projetada de forma simples e intuitiva, contendo campos para inserção de e-mail e senha, além de opção de recuperação de credenciais. A autenticação é gerenciada por meio do *Firebase Authentication*, garantindo segurança no processo de login e proteção dos dados dos usuários. Do ponto de vista técnico, essa abordagem permite a delegação da responsabilidade de autenticação para um serviço confiável em nuvem, reduzindo a complexidade do *backend* e aumentando a escalabilidade da aplicação. Além disso, o uso de autenticação baseada em *tokens* possibilita sessões seguras e controle de acesso eficiente às funcionalidades do sistema.

A Figura 8 ilustra a página principal da aplicação, onde ocorre a geração e o disparo das mensagens automatizadas.

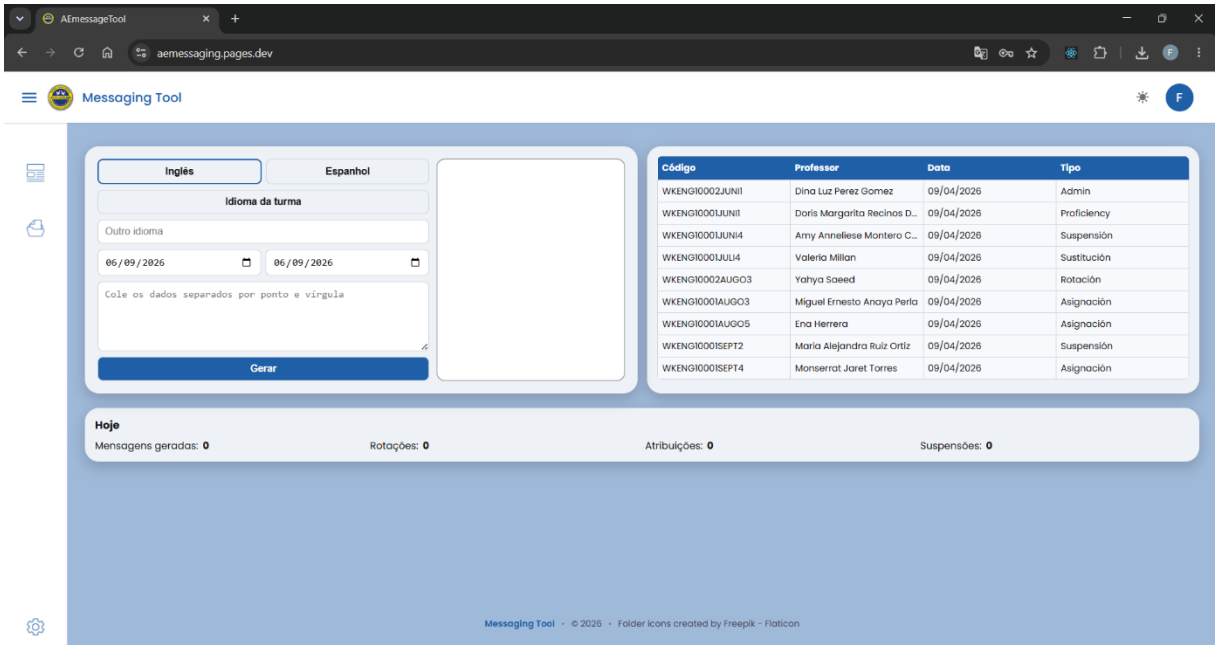


Figura 8 – Página inicial de geração de mensagens

Fonte: Autoria própria

Essa interface centraliza as principais funcionalidades do sistema, permitindo ao usuário selecionar o idioma das mensagens, definir parâmetros como período e tipo de comunicação, e inserir dados estruturados que serão processados pelo sistema. A partir dessas informações, o *AE Messaging Tool* é capaz de gerar múltiplas mensagens simultaneamente, utilizando *templates* previamente definidos. A interface também apresenta um painel com registros recentes e métricas do dia, como quantidade de mensagens geradas, rotações, designações e suspensões, fornecendo uma visão rápida do estado operacional. Tecnicamente, essa funcionalidade é suportada por uma integração entre o *front end* em *React* e o *backend* em Node.js, responsável por processar os dados, aplicar os *templates* e realizar o envio automatizado por meio da API do WhatsApp.

A Figura 9 apresenta a interface de gerenciamento e personalização de *templates* de mensagens, uma das funcionalidades centrais do sistema.

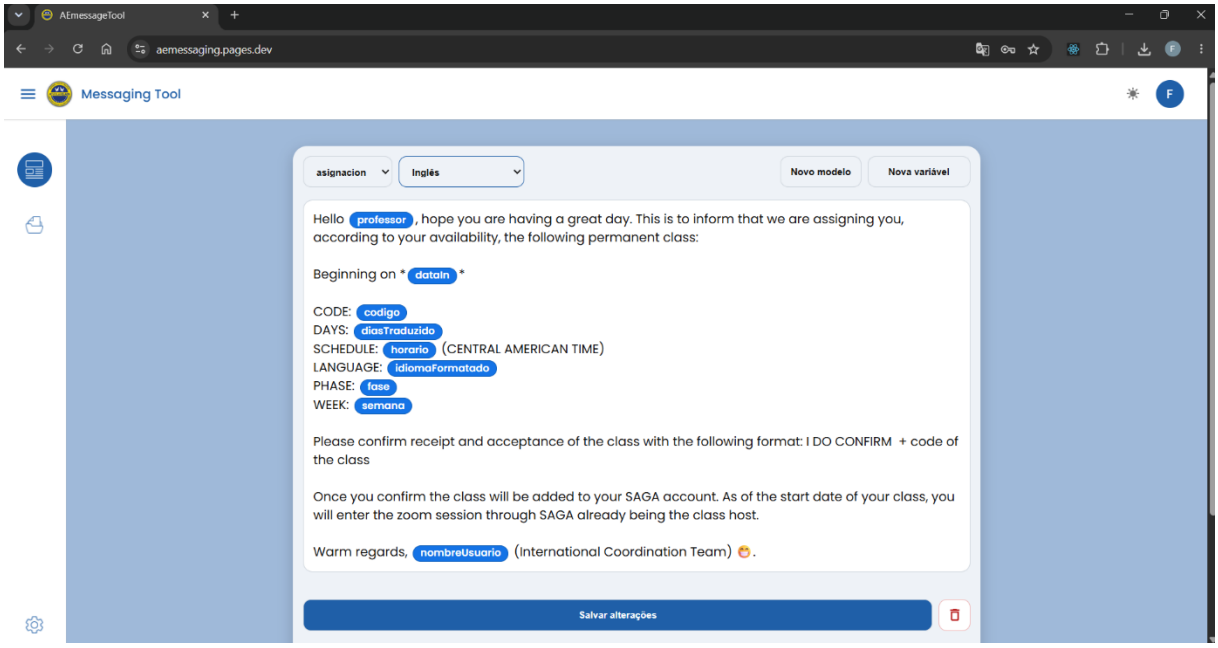


Figura 9 – Aba de personalização de *templates* de mensagens

Fonte: Autoria própria

Nessa tela, o usuário pode selecionar o tipo de mensagem e o idioma correspondente, além de editar o conteúdo textual com o uso de variáveis dinâmicas, como nome do professor, código da turma e datas. Essas variáveis são automaticamente substituídas durante o processo de geração das mensagens, permitindo alto nível de personalização sem necessidade de edição manual individual. A interface também disponibiliza opções para criação de novos *templates* e variáveis, tornando o sistema flexível e adaptável a diferentes cenários operacionais. Do ponto de vista técnico, essa funcionalidade é implementada por meio de um mecanismo de interpolação de dados, no qual o *backend* substitui os *placeholders* pelos valores reais provenientes do banco de dados ou da entrada do usuário. Além disso, a integração com a API de tradução possibilita que os *templates* sejam adaptados automaticamente para diferentes idiomas, ampliando a aplicabilidade da ferramenta em ambientes multilíngues.

A Figura 10 apresenta a interface de consulta e gerenciamento dos registros de mensagens enviadas pelo sistema garantindo desempenho adequado mesmo com grande quantidade de registros, além de facilitar futuras expansões do sistema.

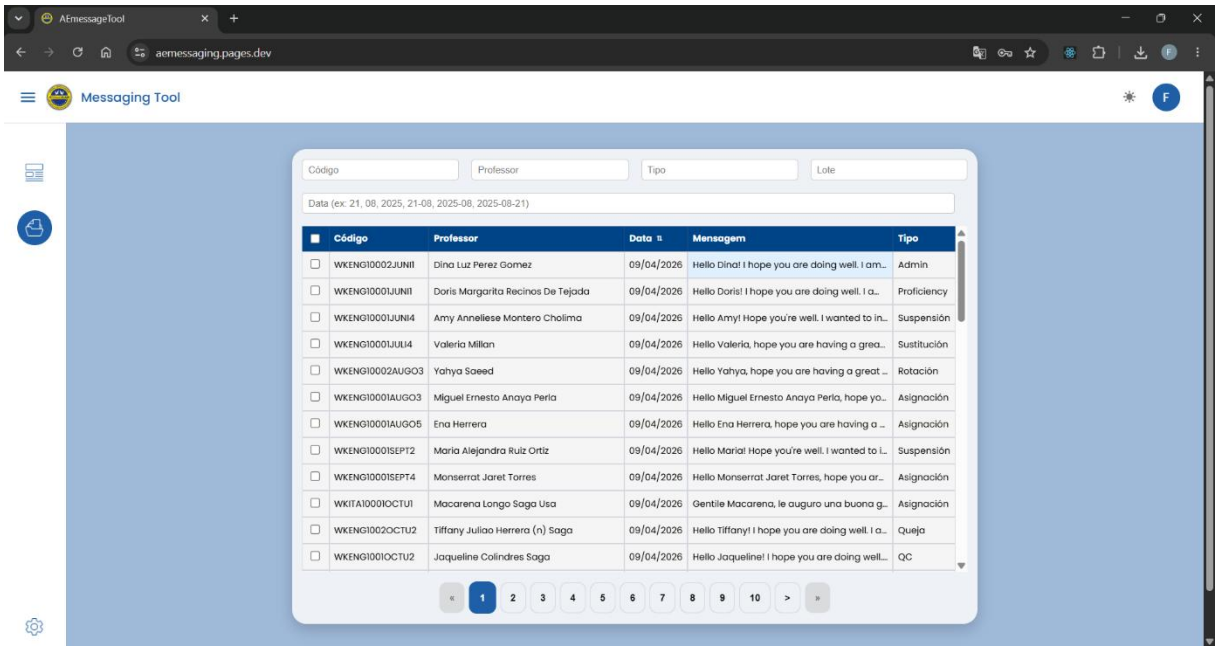


Figura 10 – Aba de registros e consultas ao banco de dados

Fonte: Autoria própria

Essa tela permite ao usuário visualizar o histórico completo das comunicações, com informações como código da operação, nome do professor, data de envio, conteúdo da mensagem e tipo de comunicação. A interface inclui funcionalidades de filtragem por diferentes critérios, como período, tipo e usuário, além de paginação para navegação eficiente em grandes volumes de dados. Essa funcionalidade é fundamental para garantir rastreabilidade e controle das operações realizadas, permitindo auditoria e análise posterior.

Do ponto de vista técnico, os dados são armazenados em um banco PostgreSQL gerenciado pelo Supabase, e recuperados por meio de requisições ao *backend*, que realiza o tratamento e a organização das informações antes de enviá-las ao *front end*. Essa arquitetura cliente servidor garante desempenho adequado mesmo com o fluxo interno de execução entre esses componentes.

A Figura 11 apresenta o Diagrama de Sequência do processo de autenticação de usuários no sistema *AE Messaging Tool*. O fluxo inicia-se com a interação do usuário na interface de *login*, onde são inseridas as credenciais de acesso. Essas

informações são enviadas ao *front end* da aplicação, que realiza a comunicação com o serviço de autenticação.

Em seguida, o *backend* interage com o *Firebase Authentication* para validar as credenciais fornecidas. Caso os dados estejam corretos, o serviço retorna um *token* de autenticação, permitindo o acesso do usuário ao sistema. Em caso de falha, uma resposta de erro é retornada ao *front end*, que informa o usuário sobre a inconsistência.

O diagrama evidencia a integração entre interface, *backend* e serviço externo de autenticação, destacando o fluxo de validação e controle de acesso, elemento essencial para garantir a segurança da aplicação.

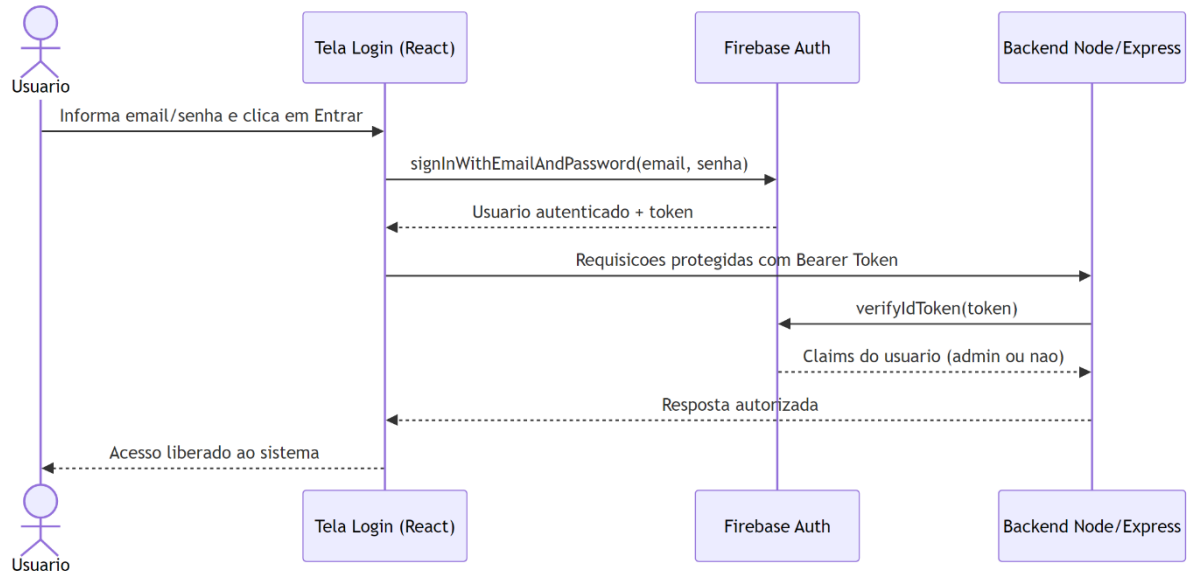


Figura 11 – Diagrama de sequência do processo de autenticação

Fonte: Autoria própria

A Figura 12 apresenta o Diagrama de Sequência do processo de geração e envio de mensagens automatizadas. O fluxo inicia-se com a interação do usuário na interface principal, onde são definidos os parâmetros necessários, como idioma, período e dados de entrada.

Após a submissão, o *front end* encaminha as informações ao *backend*, que realiza o processamento dos dados e aplica os templates previamente configurados. Durante esse processo, o sistema pode realizar integrações com serviços externos, como a API de tradução, quando necessário.

Em seguida, as mensagens são geradas e enviadas por meio da API do WhatsApp, enquanto simultaneamente são registradas no banco de dados para fins de rastreabilidade. O diagrama evidencia a coordenação entre múltiplos componentes do sistema, incluindo interface, lógica de negócio, serviços externos e persistência de dados, caracterizando um fluxo distribuído e automatizado.

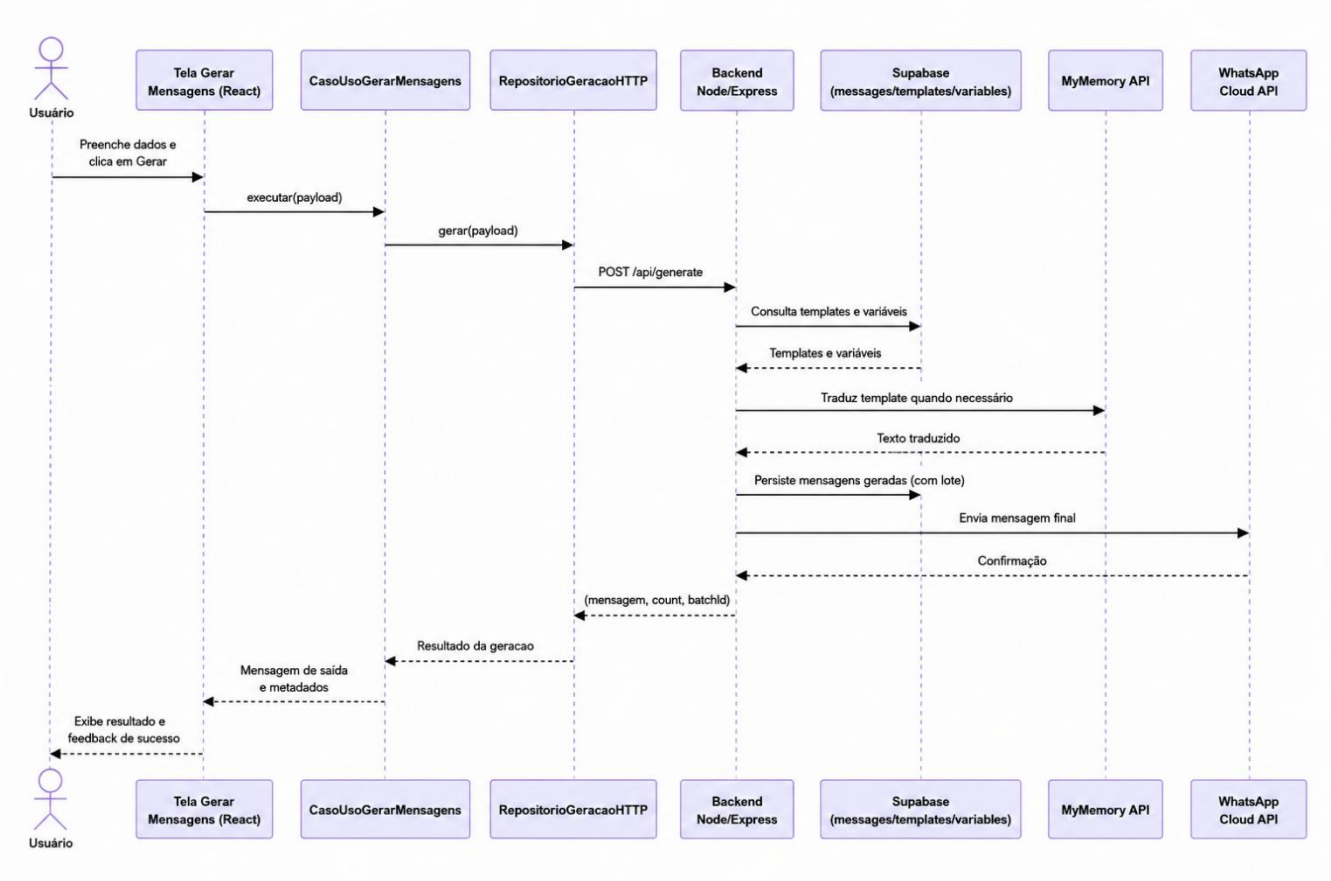


Figura 12 – Diagrama de sequência do processo de geração e envio de mensagens

Fonte: Autoria própria

A Figura 13 apresenta o Diagrama de Sequência relacionado ao processo de criação e edição de *templates* de mensagens. O fluxo tem início com a interação do usuário na interface de *templates*, onde é possível selecionar, editar ou criar modelos de mensagens.

As alterações realizadas são enviadas ao *backend*, que processa as informações e as armazena no banco de dados. Durante esse processo, o sistema também gerencia variáveis dinâmicas, que serão posteriormente substituídas por dados reais no momento da geração das mensagens.

O diagrama demonstra a separação entre a camada de interface e a camada de persistência, bem como o papel do *backend* na validação e armazenamento das informações, garantindo flexibilidade e personalização no uso dos templates.

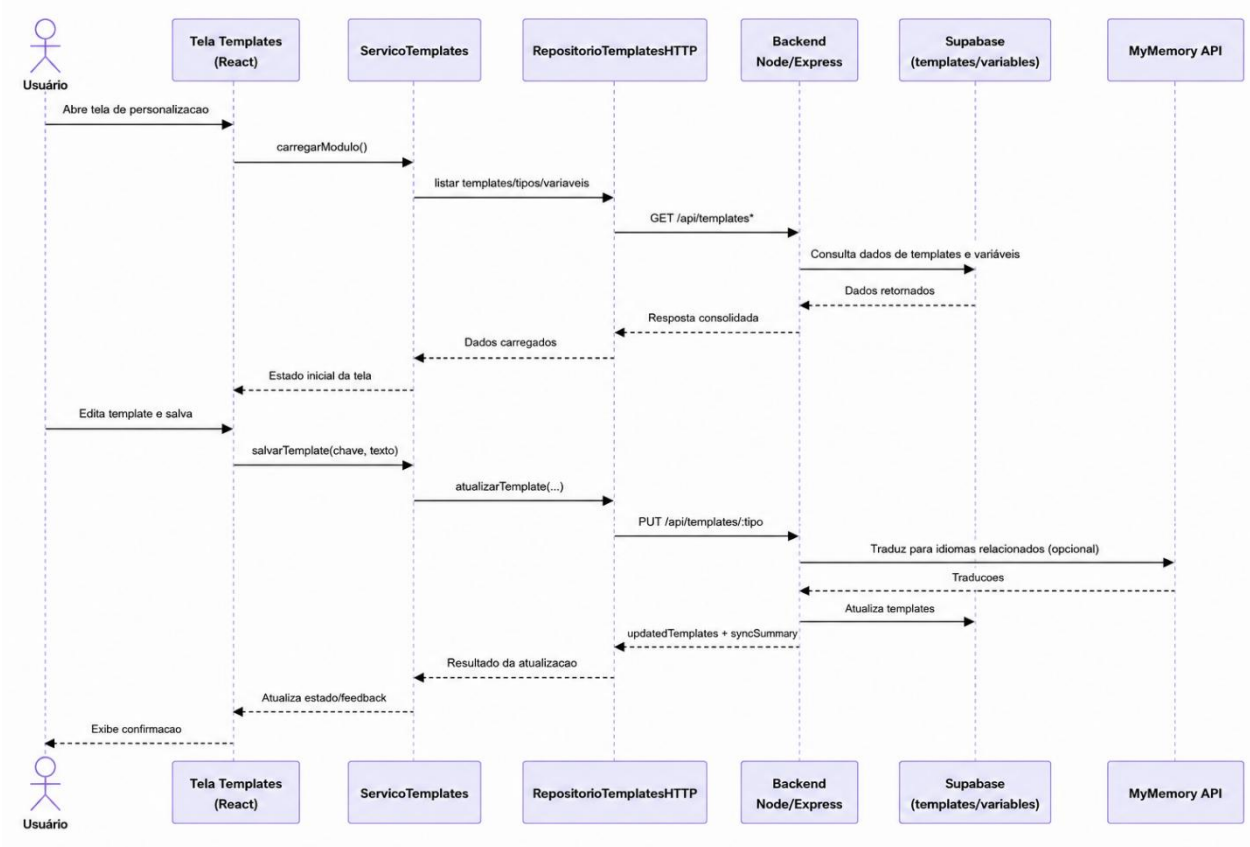


Figura 13 – Diagrama de sequência do processo de criação e edição de *templates*

Fonte: Autoria própria

A Figura 14 apresenta o Diagrama de Sequência do processo de consulta aos registros de mensagens enviadas. O fluxo inicia-se com a solicitação do usuário na interface de registros, onde podem ser aplicados filtros como período, tipo de mensagem ou usuário.

O *front end* encaminha a requisição ao *backend*, que realiza a consulta ao banco de dados, recuperando os registros correspondentes. Em seguida, os dados são processados e retornados à interface, onde são exibidos de forma estruturada e paginada.

O diagrama evidencia o fluxo de recuperação de dados em uma arquitetura cliente-servidor, destacando a interação entre interface, *backend* e banco de dados, além de reforçar o papel da persistência na garantia de rastreabilidade e controle das operações realizadas pelo sistema.

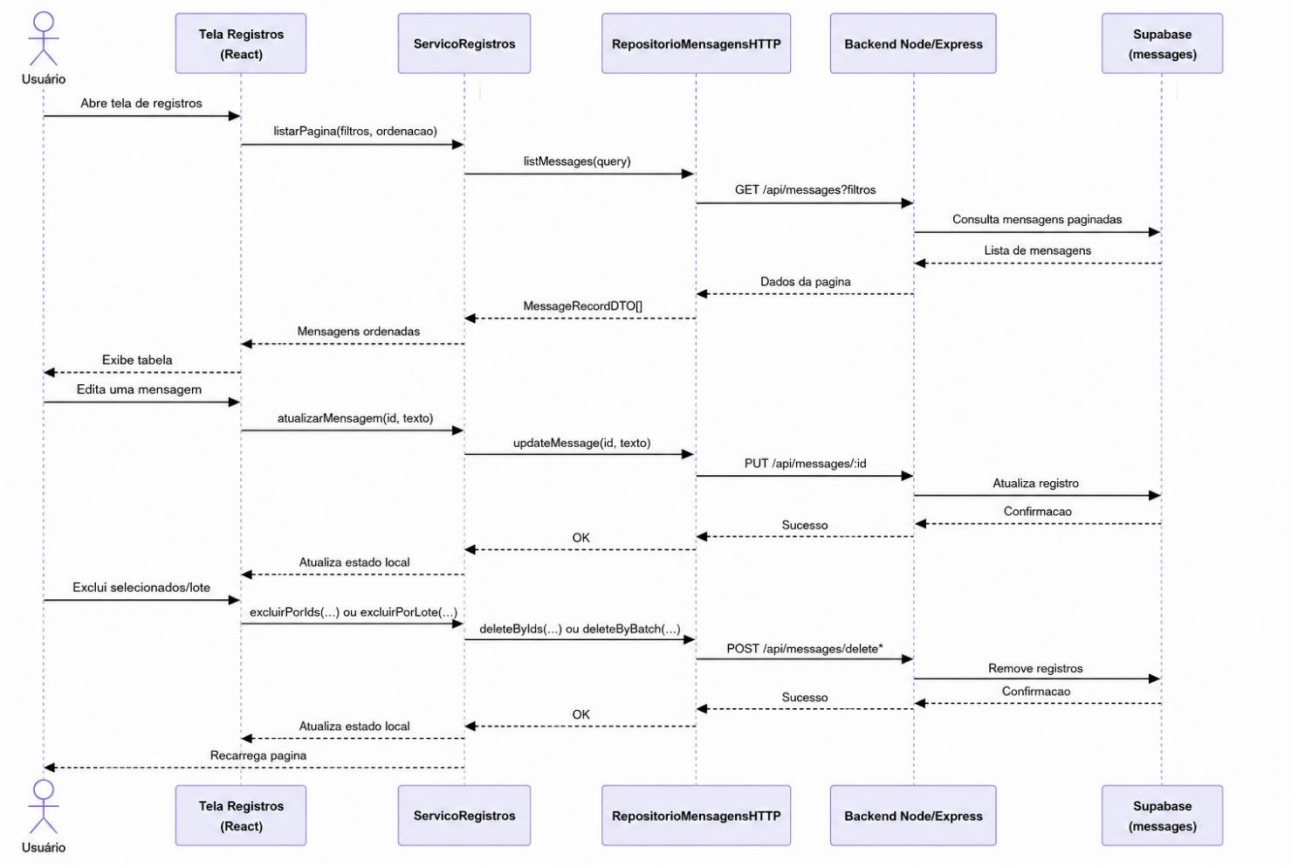


Figura 14 – Diagrama de sequência do processo de consulta aos registros

Fonte: Autoria própria

## 4.5 TESTES E VALIDAÇÃO

A etapa de testes e validação do *AE Messaging Tool* teve como objetivo verificar a corretude das funcionalidades implementadas, avaliar o desempenho do sistema em ambiente real e analisar a aceitação por parte dos usuários finais. Para isso, foram conduzidos testes em diferentes níveis, contemplando aspectos funcionais, de usabilidade e de desempenho.

Inicialmente, foram realizados testes funcionais, incluindo testes unitários e de integração, com o intuito de garantir o correto funcionamento dos principais módulos do sistema, como autenticação de usuários, geração de mensagens, personalização de *templates*, armazenamento de registros e envio automatizado via API do WhatsApp. Esses testes permitiram identificar e corrigir inconsistências na comunicação entre *front end* e *backend*, bem como validar a integridade dos dados processados.

Em seguida, foram conduzidos testes de usabilidade com membros da equipe administrativa da escola parceira, que utilizaram o sistema em situações reais de trabalho. Os usuários executaram tarefas como geração de mensagens em lote, edição de *templates* e consulta de registros históricos. Observou-se que a interface foi considerada intuitiva e de fácil utilização, permitindo a realização das tarefas com menor esforço e reduzindo significativamente a necessidade de retrabalho.

Além disso, foram realizados testes de desempenho com foco na comparação entre o processo manual anteriormente utilizado e o processo automatizado proporcionado pelo sistema. Os resultados evidenciaram uma redução média de aproximadamente 12 horas semanais no tempo gasto com atividades de mensageria, especialmente devido à automação do envio em massa e à eliminação de etapas repetitivas.

A validação do sistema também considerou aspectos de confiabilidade e rastreabilidade, sendo possível verificar que todas as mensagens enviadas são devidamente registradas e podem ser consultadas posteriormente por meio da interface de registros. Esse fator contribui para maior controle operacional e transparência nas comunicações realizadas.

Dessa forma, os testes realizados demonstraram que o *software AE Messaging Tool* atende aos requisitos definidos, apresentando desempenho satisfatório, boa

aceitação pelos usuários e capacidade de otimizar significativamente o processo de comunicação em escolas de idiomas.

## 4.6 DESAFIOS E SOLUÇÕES ENCONTRADAS

Entre os desafios enfrentados, destacam-se:

- Migração de desktop para web, exigindo aprendizado em *React* e Node.js.
- Integração confiável com o WhatsApp, solucionada por meio da API oficial da Meta.
- Implementação de tradução automática, viabilizada pela integração com a API do *MyMemory*.
- Gestão de autenticação e segurança, solucionada com *Firebase Authentication.*
- Escalabilidade, viabilizada pela adoção do Supabase e arquitetura modular em nuvem.

Cada obstáculo superado fortaleceu a maturidade técnica do projeto e contribuiu para a solidez da solução final.

# 5. CONCLUSÃO

Este trabalho teve como objetivo desenvolver uma aplicação web denominada *AE Messaging Tool*, voltada à automação de mensagens em escolas de idiomas, aplicando práticas consolidadas da Engenharia de *Software* com o propósito de otimizar o fluxo de comunicação entre coordenação e professores.

A pesquisa buscou responder à seguinte questão: “Como ancorar práticas de Engenharia de *Software* para desenvolver uma aplicação web de automação de mensagens que seja efetiva, confiável e aceitável para uso em escolas de idiomas?”. Com base nos resultados obtidos, verifica-se que essa questão foi devidamente respondida por meio da concepção, implementação e validação do sistema desenvolvido.

A aplicação foi construída seguindo uma arquitetura baseada em camadas, com separação clara entre *frontend*, *backend* e banco de dados, além de integração com serviços externos como autenticação via *Firebase*, envio de mensagens por meio da API do WhatsApp e tradução automática de conteúdo. O sistema foi desenvolvido utilizando princípios de programação orientada a objetos, promovendo melhor organização do código, reutilização de componentes e maior facilidade de manutenção e evolução. Essa combinação de decisões arquiteturais e estruturais contribuiu para a construção de uma solução robusta, escalável e alinhada às práticas modernas de engenharia de *software*.

Os testes realizados em ambiente real, com a equipe administrativa da escola parceira, evidenciaram ganhos expressivos de eficiência operacional. Observou-se uma redução média de aproximadamente 12 horas semanais no tempo gasto com tarefas manuais de comunicação, além de melhorias significativas na padronização das mensagens, rastreabilidade das informações e confiabilidade do processo de envio. A interface do sistema também apresentou boa aceitação pelos usuários, sendo considerada intuitiva e adequada ao contexto de uso.

Dessa forma, conclui-se que o *AE Messaging Tool* atingiu plenamente os objetivos propostos, demonstrando viabilidade técnica, eficiência prática e aplicabilidade real no contexto educacional. A solução desenvolvida não apenas automatiza processos administrativos, mas também contribui para a redução de erros humanos e para a melhoria da organização interna da instituição.

Além das contribuições práticas, o trabalho também reforça a importância da aplicação de conceitos de Engenharia de *Software* no desenvolvimento de sistemas reais, evidenciando como a adoção de boas práticas de arquitetura, modelagem e organização de código impacta diretamente na qualidade, manutenibilidade e evolução do produto final.

Por fim, destaca-se que o desenvolvimento do *AE Messaging Tool* proporcionou não apenas a construção de uma solução tecnológica funcional, mas também a consolidação de conhecimentos práticos em engenharia de software, desenvolvimento web, orientação a objetos e integração de sistemas, evidenciando a relevância deste trabalho como contribuição acadêmica e profissional.

Para continuidade dessa pesquisa, sugere-se como trabalho futuro a ampliação da solução por meio da integração com sistemas acadêmicos institucionais, da implementação de módulos analíticos para geração de relatórios e indicadores de desempenho, do aprimoramento da experiência do usuário com base em novos ciclos de feedback e da expansão da aplicação para outros contextos organizacionais que demandem automação da comunicação. Além disso, embora o sistema já esteja estruturado para envio de mensagens via WhatsApp, ainda depende da liberação de acesso completo à API e do cadastro dos dados de todos os professores para possibilitar novas funcionalidades, como o envio de confirmações de aceitação ou rejeição de turmas, o acompanhamento das respostas dos docentes e outros fluxos automatizados de comunicação.

## REFERÊNCIAS

ALBESHER, Abdulmohsen Saud. Reviewing the usability of web authentication procedures: comparing the current procedures of 20 websites. *Sustainability*, v. 15, n. 14, p. 11043, 2023. Disponível em: https://www.mdpi.com/2071-1050/15/14/11043/pdf. Acesso em: 30 maio 2026.

ESPEDITO, Beatriz Cardoso; TAKIZAWA, Thiago Takeshi; GASETA, Edson Roberto. *Comparativo dos processos de autenticação usados em smartphones*. Americana: Faculdade de Tecnologia de Americana, 2021. Trabalho de graduação. Disponível em: https://ric.cps.sp.gov.br/bitstream/123456789/12869/4/20211S_Beatriz%20Cardoso%20Espedito_OD1121.pdf. Acesso em: 30 maio 2026.

GIL, Antônio Carlos. **Como Elaborar Projetos de Pesquisa**. 6. ed. São Paulo: Editora Atlas Ltda, 2017.

GIL, Antônio Carlos. Como elaborar projetos de pesquisa. 7. ed. Rio de Janeiro: Editora Atlas Ltda, 2022.

GOMES, Dennis dos Santos. **Inteligência artificial: conceitos e aplicações**. Revista Olhar Científico – Faculdades Associadas de Ariquemes, Ariquemes, v. 1, n. 2, p. 234–237, ago./dez. 2010.

KAUFMAN, Dora. *Lógica e fundamentos da inteligência artificial e reações da sociedade para maximizar benefícios e mitigar danos.* Revista de Filosofia Unisinos, v. 25, n. 1, p. 1–17, 2024. Disponível em: https://revistas.unisinos.br/index.php/filosofia/article/view/27011/60749937. Acesso em: 10 nov. 2025.

LIMA, Gubio; MIRANDA, Gustavo; FARIAS, Tiago de Souza. *Introdução a rede neural para físicos.* arXiv preprint arXiv:2503.06272, 2025. Disponível em: https://arxiv.org/pdf/2503.06272. Acesso em: 10 nov. 2025.

SACRAMENTO, Lucas S. C.; CUNHA, Ítalo; CARDOSO, Gabriel P. O.; SOUZA, Artur; FRANCO, Antônio; OLIVEIRA, Leonardo B. Geração automática de APIs REST a partir de um modelo aberto de descrição de serviços. In: *Conferência Latino-Americana de Software Livre e Tecnologias Abertas (Latinoware)*. Cascavel: Sociedade Brasileira de Computação, 2022. Disponível em: https://sol.sbc.org.br/index.php/latinoware/article/download/22970/22797. Acesso em: 30 maio 2026.

SOMMERVILLE, Ian. **Engenharia de software**. 10. ed. São Paulo: Pearson Education do Brasil, 2019.

VALENTE, Marco Túlio. *Engenharia de Software Moderna.* 2. ed. Independente, 2022. Disponível em: https://engsoftmoderna.info/cap1.html e https://engsoftmoderna.info/cap7.html. Acesso em: 10 nov. 2025.

VALENTE, Marco Túlio. *Fundamentos de Manutenção de Software*. 1. ed. Independente, 2026. Disponível em: https://manutencaosoftware.org/cap3. Acesso em: 30 maio 2026.

WAZLAWICK, R. S. **Metodologia da Pesquisa para Ciência da Computação**. 2ª. ed. [S.l.]: Campus, 2014.

WAZLAWICK, R. S. Metodologia de pesquisa para ciência da computação. [S. l.]: Elsevier Editora Ltda., 2022.

# APÊNDICES

## APÊNDICE A – REQUISITOS FUNCIONAIS E NÃO FUNCIONAIS DO SISTEMA

### A.1 Requisitos Funcionais

RF01 – O sistema deve permitir autenticação de usuários.
RF02 – O sistema deve permitir geração automática de mensagens.
RF03 – O sistema deve permitir geração de mensagens em diferentes idiomas.
RF04 – O sistema deve substituir automaticamente variáveis nos *templates* de mensagens.
RF05 – O sistema deve registrar no banco de dados todas as mensagens geradas.
RF06 – O sistema deve permitir envio automático de mensagens via API do *WhatsApp*.
RF07 – O sistema deve permitir consulta ao histórico de mensagens enviadas.
RF08 – O sistema deve permitir filtrar registros por professor, tipo, lote e data.
RF09 – O sistema deve permitir edição e personalização de *templates* de mensagens.
RF10 – O sistema deve permitir tradução automática de *templates*.
RF11 – O sistema deve permitir gerenciamento de usuários por administradores.
RF12 – O sistema deve permitir alternância de idioma da interface.
RF13 – O sistema deve permitir alternância entre modo claro e escuro.
RF14 – O sistema deve exibir métricas relacionadas às mensagens geradas.

### A.2 Requisitos Não Funcionais

RNF01 – O sistema deve possuir interface web responsiva acessível por navegadores modernos.
RNF02 – O sistema deve utilizar *React*, *TypeScript* e *Vite* no *front end*.
RNF03 – O sistema deve utilizar *Node.js* e *Express* no *backend*.
RNF04 – O sistema deve utilizar *Firebase Authentication* para autenticação de usuários.

RNF05 – O sistema deve utilizar banco de dados *Supabase* para armazenamento persistente.

RNF06 – O sistema deve disponibilizar APIs REST em formato JSON.

RNF07 – O sistema deve utilizar variáveis de ambiente para armazenamento de informações sensíveis.

RNF08 – O sistema deve suportar internacionalização da interface.

RNF09 – O sistema deve impedir acesso não autorizado a funções administrativas.

RNF10 – O sistema deve utilizar HTTPS nas integrações externas em produção.

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
GABINETE DO REITOR

Av. Universitária, 1069 • Setor Universitário
Caixa Postal 86 • CEP 74605-010
Goiânia • Goiás • Brasil
Fone: (62) 3946.1000
www.pucgoias.edu.br • reitoria@pucgoias.edu.br

**RESOLUÇÃO n° 038/2020 – CEPE**

**ANEXO I**

APÊNDICE ao TCC

**Termo de autorização de publicação de produção acadêmica**

O estudante Felipe Sousa Bezerra do Curso de Engenharia de Computação, matrícula 20211003300533 telefone: 62 995033331 e-mail flipesbezerra@gmail.com, na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do Autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado AE MESSAGING TOOL — UMA FERRAMENTA WEB PARA AUTOMAÇÃO DE MENSAGENS EM ESCOLAS DE IDIOMAS, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto(PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 28 de MARÇO de 2026.

gov.br Documento assinado digitalmente
FELIPE SOUSA BEZERRA
Data: 28/03/2026 23:09:14-0300
Verifique em https://validar.iti.gov.br

Assinatura do autor: _____

Nome completo do autor: Felipe Sousa Bezerra

Assinatura do professor–orientador: gov.br Documento assinado digitalmente
SOLANGE DA SILVA
Data: 29/03/2026 06:59:13-0300
Verifique em https://validar.iti.gov.br

Nome completo do professor-orientador: Solange da Silva