PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA
GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

PUC
GOIÁS

LABORAI: AGENTES PARA GERAÇÃO AUTOMATIZADA DE DOCUMENTOS EM PROCESSOS DE LICITAÇÃO

KAYKY MOMOLI

GOIÂNIA
2026

KAYKY MOMOLI

LABORAI: AGENTES PARA GERAÇÃO AUTOMATIZADA DE DOCUMENTOS EM PROCESSOS DE LICITAÇÃO

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

Orientador: Prof. Ernesto Fonseca Veiga

GOIÂNIA
2026

RESUMO

O presente trabalho tem como objetivo desenvolver um protótipo de sistema voltado à geração automatizada de documentos em processos de licitação pública, com foco na elaboração do Termo de Referência e do Estudo Técnico Preliminar, com base nas diretrizes da Lei 14.133/2021. A aplicação, denominada *LaborAI*, foi desenvolvida utilizando React e Vite no *front-end*, NestJS no *back-end*, Supabase com PostgreSQL e pgvector como infraestrutura de dados, e LangChain.js com LangGraph para a criação e orquestração dos agentes de inteligência artificial. A arquitetura adotada segue o modelo de cliente-servidor com API REST, integrada a uma base de conhecimento construída com a técnica RAG (*Retrieval-Augmented Generation*), composta por 522 trechos indexados da Lei 14.133/2021 e de modelos reais de documentos utilizados em prefeituras. A pesquisa é de natureza aplicada e de caráter exploratório-descritivo, propondo uma solução tecnológica que conduz o servidor público por meio de uma conversa estruturada até a geração do documento final no formato DOCX. O sistema visa auxiliar na redução de erros, na padronização de documentos e na diminuição do retrabalho de inserção repetitiva de informações em múltiplos documentos, sem dispensar a revisão técnica e jurídica posterior. Os resultados obtidos reforçam a viabilidade técnica do uso de agentes conversacionais no contexto de contratações públicas, evidenciando o potencial da inteligência artificial na modernização e eficiência da administração pública.

Palavras-chave: Licitações Públicas. Inteligência Artificial. Lei 14.133/2021. NestJS. LangChain.

ABSTRACT

This research project aims to develop a prototype system for the automated generation of documents in public procurement processes, focusing on the Terms of Reference and the Preliminary Technical Study, based on the guidelines of Brazilian Law 14.133/2021. The application, called *LaborAI*, was developed using React and Vite on the front-end, NestJS on the back-end, Supabase with PostgreSQL and pgvector as data infrastructure, and LangChain.js with LangGraph for the creation and orchestration of artificial intelligence agents. The architecture follows the client-server model with REST API, integrated with a knowledge base built using the RAG (*Retrieval-Augmented Generation*) technique, consisting of 522 indexed excerpts from Law 14.133/2021 and real procurement documents used in municipalities. The research is applied and exploratory-descriptive in nature, proposing a technological solution that guides civil servants through a structured conversation until the final document is generated in DOCX format. The system aims to assist in reducing errors, standardizing documents and decreasing repetitive data entry across multiple documents, without replacing subsequent technical and legal review. The results obtained reinforce the technical feasibility of using conversational agents in the context of public procurement, highlighting the potential of artificial intelligence in the modernization and efficiency of public administration.

Keywords: Public Procurement. Artificial Intelligence. Law 14.133/2021. NestJS. LangChain.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

IA	Inteligência Artificial
RAG	Retrieval-Augmented Generation
LLM	Large Language Model
TR	Termo de Referência
ETP	Estudo Técnico Preliminar
API	Application Programming Interface
REST	Representational State Transfer
JWT	JSON Web Token
RLS	Row Level Security
JSON	JavaScript Object Notation
SQL	Structured Query Language
JSX	JavaScript XML
URL	Uniform Resource Locator
UX	User Experience
UI	User Interface
PWA	Progressive Web Application
SGBD	Sistema Gerenciador de Banco de Dados
IDE	Integrated Development Environment

SUMÁRIO

1 INTRODUÇÃO

A administração pública brasileira conduz expressivos volumes de contratações por meio de processos licitatórios regulamentados por lei. Com a promulgação da Lei nº 14.133, de 1º de abril de 2021, conhecida como a Nova Lei de Licitações e Contratos Administrativos, o ordenamento jurídico das contratações públicas passou por ampla reformulação, impondo novas exigências técnicas e procedimentais aos órgãos e entidades da administração direta e indireta (BRASIL, 2021).

Entre os documentos previstos pela legislação, destacam-se o Estudo Técnico Preliminar (ETP) e o Termo de Referência (TR). O ETP integra a fase preparatória da contratação e é tratado pela Lei nº 14.133/2021 no contexto do planejamento do processo licitatório, especialmente no Art. 18. O TR, por sua vez, definido no Art. 6º, inciso XXIII da mesma lei, especifica o objeto da contratação com todas as informações necessárias para que os licitantes elaborem suas propostas. A elaboração adequada desses documentos é diretamente relacionada à qualidade do planejamento da contratação, pois o ETP e o TR orientam a definição da necessidade administrativa, da solução escolhida e do objeto a ser contratado (BRASIL, 2021; BRASIL, 2022a; BRASIL, 2022b; TCU, [s.d.]a; TCU, [s.d.]b).

Na elaboração desses documentos, a qualidade, a padronização e a coerência das informações produzidas durante a fase preparatória da contratação são aspectos relevantes. Como o Estudo Técnico Preliminar serve de base para a construção de documentos posteriores, falhas ou lacunas nessa etapa podem repercutir na especificação do objeto e na elaboração do Termo de Referência. Dessa forma, a construção desses documentos demanda atenção à conferência das informações, à estruturação das peças e à adequação às exigências legais (BRASIL, 2021; TCE-AL, 2021).

Nesse contexto, o avanço dos modelos de linguagem de grande escala (*Large Language Models* - LLMs) e das técnicas de recuperação aumentada por geração (RAG - *Retrieval-Augmented Generation*) tornou possível a construção de agentes conversacionais capazes de conduzir o usuário na produção de documentos técnicos com apoio em bases de conhecimento específicas (LEWIS et al., 2020). A aplicação dessas tecnologias ao contexto de licitações públicas representa uma oportunidade concreta de modernização e eficiência nos processos de contratação pública.

Diante desse contexto, este trabalho visa responder à seguinte questão: **Como o uso de agentes de inteligência artificial pode auxiliar servidores públicos na elaboração automatizada de documentos de licitação com base na Lei 14.133/2021?**

1.1 Justificativa

A justificativa para o desenvolvimento do LaborAI fundamenta-se na complexidade técnica dos documentos exigidos pela Nova Lei de Licitações e no esforço operacional necessário para sua elaboração manual. A Lei 14.133/2021 ampliou a relevância da fase preparatória do processo de contratação, exigindo maior atenção ao planejamento, à justificativa da necessidade administrativa e à definição adequada do objeto a ser contratado.

Documentos como o Estudo Técnico Preliminar e o Termo de Referência possuem papel central nesse processo, pois organizam as informações que fundamentam a contratação e orientam as etapas posteriores do certame. A elaboração manual dessas peças pode gerar retrabalho, inconsistências e dificuldades de padronização, principalmente quando as mesmas informações precisam ser reaproveitadas em documentos distintos.

A automação assistida por inteligência artificial, fundamentada em legislação atualizada e em modelos reais de documentos, apresenta-se como alternativa viável para auxiliar na redução desses problemas, padronizar a produção documental e ampliar o acesso ao conhecimento técnico necessário para a elaboração inicial das peças do processo. A proposta do LaborAI, entretanto, não substitui a análise técnica ou jurídica realizada por servidores e profissionais responsáveis, mas oferece apoio à organização, padronização e geração inicial dos documentos.

1.2 Objetivos

1.2.1 Geral

- Desenvolver um sistema web baseado em agentes de inteligência artificial capaz de conduzir servidores públicos na elaboração automatizada de Termos de Referência e Estudos Técnicos Preliminares com base na Lei 14.133/2021.

1.2.2 Objetivos Específicos

- Construir uma base de conhecimento com a técnica RAG, indexando a Lei 14.133/2021 e modelos reais de documentos de licitação de municípios brasileiros;
- Implementar agentes conversacionais com fluxo controlado por meio do LangGraph, capazes de coletar informações do usuário e gerar documentos estruturados;
- Implementar autenticação, controle de acesso e persistência de sessões de conversa;
- Permitir o upload e a análise de arquivos PDF e DOCX para reaproveitamento de informações na geração documental;
- Integrar a geração de arquivos no formato DOCX com armazenamento em nuvem e disponibilização de link para download;
- Desenvolver interface web que permita ao usuário interagir com os agentes e gerenciar o histórico de sessões e documentos gerados.

1.3 Resultados Esperados

O desenvolvimento do LaborAI visa comprovar a viabilidade técnica e funcional de um sistema que automatiza a produção de documentos utilizados em processos licitatórios. O principal resultado esperado é um protótipo funcional que integre, em uma única plataforma, agentes de TR e ETP com base de conhecimento jurídico real, interface conversacional e geração de documentos formatados.

Isso poderá auxiliar servidores públicos e profissionais envolvidos na fase preparatória da contratação na organização das informações, na padronização inicial dos documentos e na redução de tarefas repetitivas, sem dispensar a revisão técnica e jurídica posterior.

1.4 Organização do Trabalho

Este trabalho está dividido em cinco capítulos, estruturados da seguinte maneira.

No presente capítulo foi apresentada a introdução, identificando o problema da elaboração manual de documentos de licitação, a motivação para o desenvolvimento do LaborAI e os objetivos propostos.

No Capítulo 2 é apresentada a fundamentação teórica, com os conceitos e tecnologias que sustentam a proposta, incluindo licitações públicas, inteligência artificial, modelos de linguagem, RAG e as principais ferramentas utilizadas no desenvolvimento.

O Capítulo 3 descreve os materiais e procedimentos metodológicos, detalhando a classificação da pesquisa, as ferramentas utilizadas e os equipamentos empregados.

O Capítulo 4 apresenta o desenvolvimento do LaborAI, descrevendo a arquitetura, as decisões técnicas, a construção dos agentes, a base de conhecimento RAG e as principais funcionalidades implementadas.

Por fim, o Capítulo 5 traz as considerações finais, discutindo os resultados obtidos em relação aos objetivos propostos e as sugestões de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os fundamentos teóricos e as tecnologias que sustentam o desenvolvimento do LaborAI, detalhando os conceitos de licitações públicas, inteligência artificial, recuperação aumentada por geração e as principais ferramentas utilizadas na construção do sistema.

2.1 Licitações Públicas e Lei nº 14.133/2021

A Lei nº 14.133/2021 estabelece normas gerais de licitação e contratação para as administrações públicas diretas, autárquicas e fundacionais da União, dos Estados, do Distrito Federal e dos Municípios. Entre as inovações introduzidas pela nova legislação, destaca-se o fortalecimento da fase preparatória da contratação, na qual são definidos os elementos técnicos, econômicos e administrativos que fundamentam o processo licitatório (BRASIL, 2021).

Essa etapa de planejamento é fundamental para a adequada condução das contratações públicas, pois permite identificar a necessidade administrativa, avaliar alternativas, definir o objeto e estabelecer condições para a futura contratação. Nesse contexto, documentos como o Estudo Técnico Preliminar e o Termo de Referência desempenham papel central, uma vez que organizam as informações necessárias para justificar e orientar o processo de contratação.

2.2 Estudo Técnico Preliminar

O Estudo Técnico Preliminar, previsto no contexto da fase preparatória da Lei 14.133/2021, é o documento responsável por caracterizar o interesse público envolvido na contratação e demonstrar a necessidade da solução escolhida para atendê-lo. Sua elaboração deve anteceder a redação do Termo de Referência, fornecendo fundamentação técnica, econômica e legal para justificar a contratação pretendida (BRASIL, 2021).

A Instrução Normativa SEGES nº 58/2022 dispõe sobre a elaboração dos Estudos Técnicos Preliminares para aquisição de bens e contratação de serviços e obras no âmbito da administração pública federal direta, autárquica e fundacional. Esse normativo reforça o papel do ETP como instrumento de planejamento, destinado a analisar a necessidade da

contratação, identificar soluções possíveis e avaliar a viabilidade da alternativa selecionada (BRASIL, 2022a).

2.3 Termo de Referência

O Termo de Referência, definido no Art. 6º, inciso XXIII, da Lei nº 14.133/2021, é o documento elaborado a partir dos estudos técnicos preliminares e deve conter os elementos necessários e suficientes para caracterizar o objeto da contratação. Entre esses elementos, estão a descrição do objeto, os requisitos técnicos, o modelo de execução, os critérios de medição, os prazos e as condições de pagamento (BRASIL, 2021).

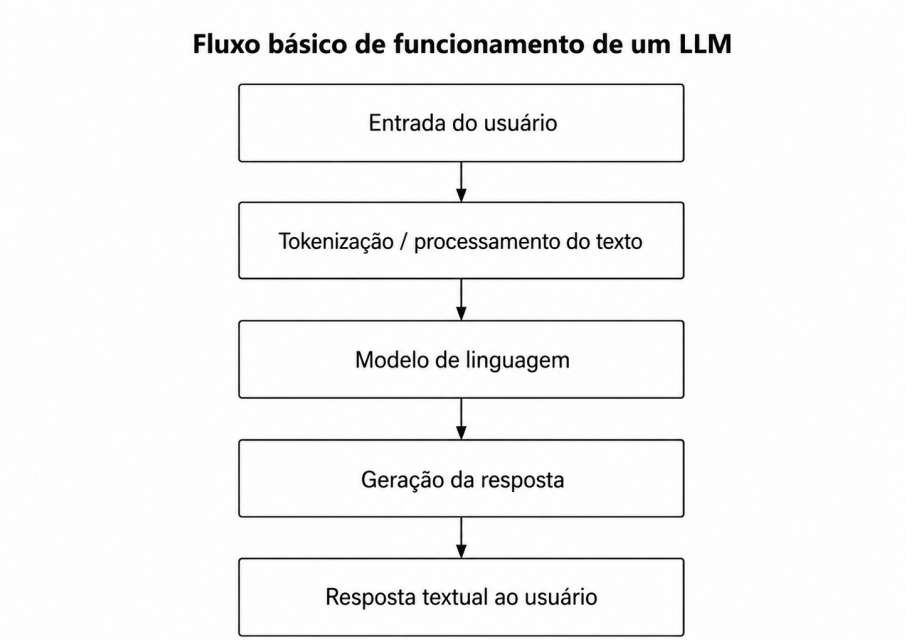
A Instrução Normativa SEGES/ME nº 81/2022 dispõe sobre a elaboração do Termo de Referência para aquisição de bens e contratação de serviços no âmbito da administração pública federal direta, autárquica e fundacional. Dessa forma, o TR consolida as informações necessárias para orientar os licitantes e permitir que a Administração conduza o processo de contratação de maneira mais clara, objetiva e padronizada (BRASIL, 2022b).

2.4 Inteligência Artificial e Modelos de Linguagem

A inteligência artificial é um campo da ciência da computação dedicado ao desenvolvimento de sistemas capazes de realizar tarefas que, quando executadas por humanos, exigiriam inteligência. Dentro desse campo, os modelos de linguagem de grande escala (*Large Language Models* - LLMs) representam uma categoria de sistemas treinados em grandes volumes de texto para compreender e gerar linguagem natural com alto grau de fluência e coerência (RUSSELL; NORVIG, 2022; BROWN et al., 2020).

Os LLMs são a base dos modernos agentes conversacionais. Esses modelos são capazes de seguir instruções, responder perguntas, resumir textos e gerar documentos estruturados, especialmente quando associados a técnicas de ajuste e treinamento voltadas ao alinhamento com instruções humanas (OUYANG et al., 2022). Essa capacidade torna os LLMs particularmente adequados para domínios que exigem geração de texto técnico e estruturado, como é o caso dos documentos de licitação pública. O fluxo básico de funcionamento pode ser observado na Figura 1.

Figura 1 – Fluxo básico de funcionamento de um LLM



Fonte: Elaborado pelo autor com base em Brown et al. (2020).

A Figura 1 representa o fluxo básico de um modelo de linguagem de grande escala. A entrada do usuário passa por um processo de tokenização e é processada internamente pelo modelo, que gera uma resposta textual ao final do fluxo.

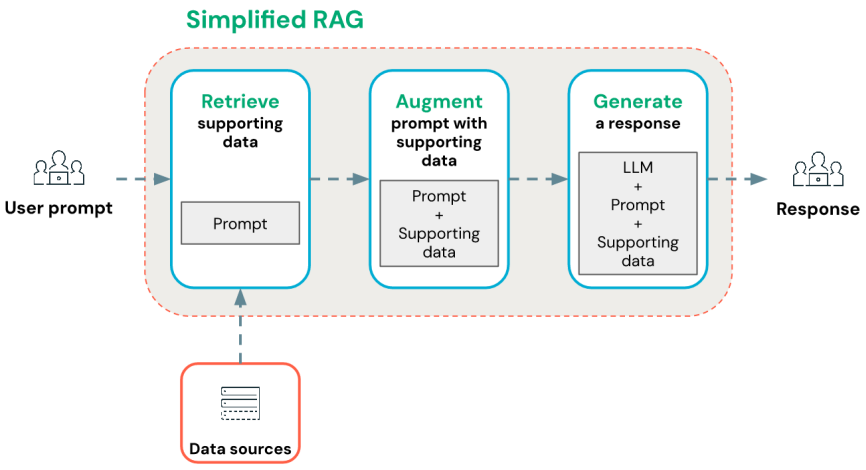
2.5 Geração Aumentada por Recuperação (RAG)

A técnica de Geração Aumentada por Recuperação, conhecida pela sigla RAG (*Retrieval-Augmented Generation*), foi proposta por Lewis et al. (2020) como uma abordagem para superar as limitações dos modelos de linguagem no acesso a conhecimentos específicos de domínio. Em vez de depender exclusivamente do conhecimento adquirido durante o treinamento, os modelos RAG consultam uma base de dados externa antes de gerar cada resposta, injetando os trechos mais relevantes como contexto adicional (LEWIS et al., 2020).

O funcionamento do RAG ocorre em duas etapas principais. Na etapa de recuperação, a entrada do usuário é convertida em um vetor numérico denominado *embedding*, que representa semanticamente o significado do texto. Esse vetor é então comparado com os vetores armazenados na base de conhecimento, e os trechos com maior similaridade semântica são recuperados. Na etapa de geração, esses trechos são injetados no *prompt*

enviado ao modelo de linguagem, que os utiliza como contexto para produzir uma resposta precisa e embasada. A Figura 2 apresenta essa arquitetura de forma esquemática.

Figura 2 – Arquitetura simplificada do RAG



Fonte: Microsoft (2026).

Conforme apresentado na Figura 2, o RAG é composto por três etapas principais. Na primeira, o sistema recupera dados de suporte a partir das fontes de dados externas. Na segunda, o *prompt* original é enriquecido com esse conteúdo. Na terceira, o modelo de linguagem combina *prompt* e dados de suporte para gerar a resposta final. No LaborAI, a fonte de dados é a base de conhecimento formada por trechos da Lei nº 14.133/2021 e por modelos documentais.

2.6 Linguagens

2.6.1 TypeScript

O TypeScript é uma linguagem de programação de código aberto desenvolvida pela Microsoft com o objetivo de tornar o JavaScript mais robusto e escalável. Por ser um superconjunto do JavaScript, qualquer código JavaScript é também válido em TypeScript. Sua principal diferença é a adição de um sistema de tipagem estática opcional, que permite declarar explicitamente os tipos de variáveis, parâmetros e retornos de funções, tornando possível identificar erros ainda na fase de desenvolvimento, antes da execução do programa (MICROSOFT, 2024).

O código TypeScript é transpilado para JavaScript padrão, preservando a compatibilidade com qualquer ambiente de execução baseado em JavaScript. No LaborAI, o TypeScript foi utilizado na *stack* do *back-end*, garantindo maior previsibilidade e segurança na manipulação dos estados dos agentes, das estruturas de dados do banco e das interfaces de comunicação com a API. O Código 2.1 apresenta um trecho do arquivo `jwt.strategy.ts`, responsável pela validação dos tokens JWT no *back-end* do LaborAI.

Listing 2.1 – Trecho do arquivo jwt.strategy.ts

```
@Injectable()
export class JwtStrategy extends PassportStrategy(Strategy, 'jwt') {
  constructor(
    config: ConfigService,
    private supabase: SupabaseService,
  ) {
    const supabaseUrl = config.getOrThrow<string>('SUPABASE_URL');
    super({
      jwtFromRequest: ExtractJwt.fromAuthHeaderAsBearerToken(),
      ignoreExpiration: false,
      secretOrKeyProvider: passportJwtSecret({
        cache: true,
        rateLimit: true,
        jwksRequestsPerMinute: 5,
        jwksUri: `${supabaseUrl}/auth/v1/.well-known/jwks.json`,
      }),
      algorithms: ['ES256', 'RS256'],
    });
  }

  async validate(payload: JwtPayload): Promise<AuthUser> {
    const { data, error } = await this.supabase
      .getAdminClient()
      .auth.admin.getUserById(payload.sub);

    if (error || !data.user) {
      throw new UnauthorizedException('Usuario nao encontrado');
    }

    return {
```

```
      id: payload.sub,
      email: payload.email,
      role: payload.role,
    };
  }
}
```

O Código 2.1 demonstra como o TypeScript permite declarar tipos explícitos nos parâmetros e retornos de funções, tornando o fluxo de autenticação mais seguro e previsível durante o desenvolvimento..

2.6.2 CSS com Tailwind

O Cascading Style Sheets (CSS) é a linguagem responsável pela apresentação visual de páginas web, separando a camada de estilo da camada de conteúdo. Ele permite definir cores, espaçamentos, fontes, dimensões e layouts responsivos (W3C, 2023).

O Tailwind CSS é um *framework* de utilitários que simplifica o processo de estilização por meio de classes pré-definidas aplicadas diretamente no JSX, eliminando a necessidade de arquivos CSS separados e reduzindo redundâncias. Sua abordagem baseada em utilitários proporciona maior produtividade, consistência visual e controle total sobre o *design* da interface (TAILWIND CSS, 2025). No LaborAI, o Tailwind CSS foi utilizado para construir uma interface responsiva, padronizada e compatível com a proposta visual da aplicação web.

2.7 Frameworks e Ferramentas

2.7.1 Vite

O Vite é uma ferramenta moderna de construção de projetos *front-end* que aproveita os módulos ECMAScript nativos dos navegadores para inicializar um servidor de desenvolvimento de forma quase instantânea. Diferentemente de ferramentas tradicionais, o Vite não empacota o projeto inteiro antes de iniciá-lo, enviando ao navegador apenas os módulos solicitados em cada momento (VITE, 2023).

No LaborAI, o Vite foi utilizado para estruturar o projeto React e otimizar o processo de desenvolvimento da interface, permitindo ciclos mais rápidos de alteração, teste e

validação visual.

2.7.2 React

O React é uma biblioteca JavaScript de código aberto desenvolvida pelo Facebook para criação de interfaces de usuário. Sua arquitetura baseada em componentes promove a divisão lógica da interface em partes coesas e reutilizáveis. O uso do Virtual DOM torna o processo de renderização mais eficiente, pois apenas os componentes afetados por mudanças de estado são atualizados na tela (BANKS, 2020).

A sintaxe JSX (*JavaScript XML*), característica do React, permite escrever estrutura e lógica no mesmo arquivo, tornando o código mais legível e organizado. No LaborAI, o React foi utilizado no *front-end* para construção de todas as telas da aplicação, incluindo a interface de conversação com os agentes, a tela inicial, a listagem de sessões e os componentes de navegação.

2.7.3 NestJS

O NestJS é um *framework* avançado para Node.js que utiliza TypeScript como base e incorpora conceitos de arquitetura modular, injeção de dependência e programação orientada a objetos. Fundamentado em padrões consolidados da engenharia de software, o NestJS favorece a modularização, a separação de responsabilidades e a testabilidade do código, características essenciais em sistemas de maior complexidade (NESTJS, 2025).

No LaborAI, o NestJS estrutura o *back-end* em módulos independentes para autenticação, agentes, RAG, documentos e armazenamento, permitindo que cada responsabilidade seja mantida de forma isolada.

2.7.4 LangChain.js

O LangChain é um *framework* voltado ao desenvolvimento de aplicações baseadas em modelos de linguagem, oferecendo recursos para integração com provedores de IA, construção de agentes, uso de ferramentas e composição de fluxos de processamento (LANGCHAIN, 2026).

O LangChain.js é a versão do *framework* para JavaScript e TypeScript, utilizada no LaborAI para integração com o modelo Claude da Anthropic, para execução das buscas

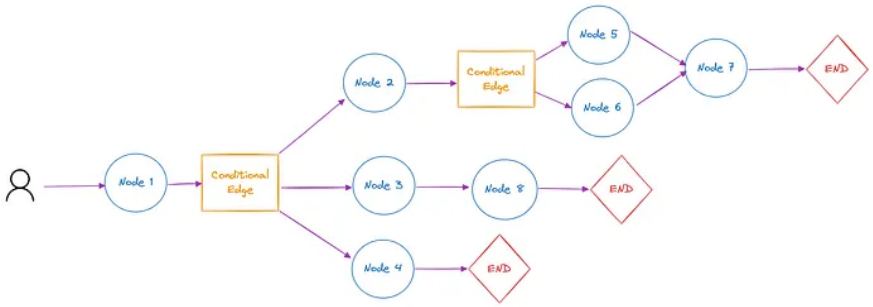
semânticas na base de conhecimento e para a construção dos grafos de estado dos agentes.

2.7.5 LangGraph

O LangGraph é uma extensão do ecossistema LangChain que permite criar agentes com fluxos de execução controlados por meio de grafos de estado. Em vez de deixar o modelo de linguagem decidir livremente o que fazer a cada etapa, o LangGraph permite definir explicitamente os nós que representam etapas do processo e as arestas que determinam as transições entre eles. As arestas podem ser fixas ou condicionais, permitindo que o fluxo se adapte ao estado atual da conversa (LANGCHAIN, 2026).

Essa abordagem é fundamental para o LaborAI, pois os agentes de TR e ETP precisam seguir uma sequência definida de coleta de informações antes de gerar o documento final. A estrutura de um grafo no LangGraph está representada na Figura 3.

Figura 3 – Estrutura de um grafo no LangGraph



Fonte: Burhan (2025).

A Figura 3 ilustra como o LangGraph organiza o fluxo de execução de um agente. Cada nó circular representa uma etapa do processo, enquanto os quadrados laranja indicam arestas condicionais que determinam qual caminho será seguido com base no estado atual da conversa. Essa estrutura permite que diferentes fluxos sejam percorridos a partir de uma mesma entrada, tornando o comportamento do agente controlado e previsível.

2.8 Infraestrutura de Dados

2.8.1 PostgreSQL

O PostgreSQL é um sistema gerenciador de banco de dados relacional de código aberto, reconhecido por sua estabilidade, extensibilidade e aderência ao padrão SQL. Ele

suporta transações ACID (*Atomicity, Consistency, Isolation, Durability*), *triggers*, funções armazenadas e tipos de dados avançados, sendo amplamente adotado em aplicações corporativas que exigem consistência e confiabilidade (POSTGRESQL, 2025).

No LaborAI, o PostgreSQL armazena as tabelas de usuários, sessões, mensagens, documentos, *templates* e a base de conhecimento vetorial, servindo como fundação de toda a persistência de dados da aplicação.

2.8.2 Supabase

O Supabase é uma plataforma de código aberto construída sobre o PostgreSQL que oferece serviços integrados de banco de dados, autenticação, armazenamento de arquivos e APIs. Por meio desses recursos, a plataforma permite centralizar diferentes necessidades de uma aplicação web em uma mesma infraestrutura (SUPABASE, 2026).

No LaborAI, o Supabase concentra três responsabilidades principais. A primeira é o gerenciamento de autenticação, utilizado para login dos usuários e emissão de tokens JWT. A segunda é o armazenamento de arquivos em *buckets*, utilizado para os arquivos enviados pelos usuários e para os documentos gerados pelos agentes. A terceira é a execução do banco de dados PostgreSQL com suporte ao pgvector, utilizado para armazenar os *embeddings* da base de conhecimento.

Além disso, o mecanismo RLS (*Row Level Security*) permite definir políticas de acesso diretamente no banco de dados, garantindo que cada usuário acesse apenas os dados compatíveis com suas permissões. No LaborAI, esse recurso foi utilizado como uma camada adicional de segurança para proteger sessões, mensagens, documentos e demais dados associados aos usuários.

2.8.3 pgvector

O pgvector é uma extensão do PostgreSQL que adiciona suporte nativo ao armazenamento e à busca de vetores de alta dimensão. Ele permite executar operações de similaridade entre vetores diretamente no banco de dados, viabilizando a busca semântica necessária para o funcionamento do RAG sem a necessidade de um banco de dados vetorial externo (PGVECTOR, 2024).

No LaborAI, o pgvector armazena os *embeddings* dos 522 trechos indexados da Lei

14.133/2021 e dos modelos reais de TR e ETP, permitindo que os agentes recuperem os trechos mais relevantes para cada etapa da geração do documento.

2.9 Arquitetura

2.9.1 Cliente-Servidor

A arquitetura cliente-servidor é um modelo distribuído no qual as responsabilidades são divididas entre dois componentes distintos. O cliente é responsável pela interação com o usuário e pelo envio de requisições ao servidor, que processa as solicitações, aplica as regras de negócio e retorna as respostas. Essa separação permite que múltiplos clientes acessem simultaneamente os serviços oferecidos pelo servidor, promovendo escalabilidade e organização do sistema (TANENBAUM; VAN STEEN, 2007).

No LaborAI, o *front-end* em React atua como cliente, e o *back-end* em NestJS atua como servidor, processando as mensagens dos usuários, invocando os agentes e retornando as respostas e os documentos gerados.

2.9.2 API REST

As APIs REST (*Representational State Transfer*) são uma abordagem de comunicação entre sistemas que utiliza o protocolo HTTP para realizar operações sobre recursos identificados por URLs bem definidas. Cada requisição é independente e contém todas as informações necessárias para ser processada pelo servidor, caracterizando o modelo como *stateless* (RICHARDSON; RUBY, 2013).

No LaborAI, toda a comunicação entre o *front-end* e o *back-end* ocorre por meio de uma API REST, com *endpoints* protegidos por JWT para autenticação e autorização das requisições.

3 MATERIAIS E PROCEDIMENTOS METODOLÓGICOS

Este capítulo tem como propósito descrever os procedimentos adotados e as estratégias aplicadas no desenvolvimento deste trabalho, correspondente ao Trabalho de Conclusão de Curso II. Serão apresentadas as principais ferramentas utilizadas na construção do LaborAI, sistema web destinado à geração automatizada de documentos em processos de licitação pública. A implementação foi conduzida com base nos fundamentos teóricos abordados no capítulo anterior, transformando os conceitos de arquitetura web, agentes de inteligência artificial e recuperação aumentada por geração em uma solução funcional.

3.1 Metodologia

3.2 Metodologia

Esta pesquisa, segundo sua natureza, caracteriza-se como aplicada, pois tem como finalidade o desenvolvimento de uma solução prática para um problema específico: a elaboração de documentos utilizados em processos de licitação pública. O trabalho não se limita ao estudo teórico das tecnologias envolvidas, mas utiliza esses conceitos na construção de um protótipo funcional, capaz de conduzir o usuário na criação de Termos de Referência e Estudos Técnicos Preliminares. Pesquisas aplicadas têm como objetivo gerar conhecimento para aplicação prática e solução de problemas específicos, ao contrário da pesquisa básica, que busca ampliar o conhecimento sem fins imediatos de aplicação (GIL, 2017; WAZLAWICK, 2014).

Segundo seus objetivos, a pesquisa pode ser classificada como exploratória e descritiva. É exploratória porque envolve o estudo e a aplicação de tecnologias recentes, como agentes de inteligência artificial, RAG e bases vetoriais, em um domínio específico da administração pública. A pesquisa exploratória tem como principal objetivo proporcionar maior familiaridade com o problema, tornando-o mais explícito ou construindo hipóteses a partir de sua análise (GIL, 2017). Também é descritiva porque apresenta, de forma organizada, as etapas de construção do sistema, as ferramentas utilizadas, a estrutura da aplicação e o funcionamento dos principais módulos implementados. A pesquisa descritiva tem como objetivo principal a descrição das características de determinado fenômeno ou o

estabelecimento de relações entre variáveis (GIL, 2017; WAZLAWICK, 2014).

Quanto aos procedimentos técnicos, o trabalho combina pesquisa bibliográfica, documental e desenvolvimento de protótipo. A pesquisa bibliográfica foi utilizada para fundamentar os conceitos de inteligência artificial, modelos de linguagem, RAG, arquitetura cliente-servidor, API REST e tecnologias de desenvolvimento web. A pesquisa documental foi empregada no estudo da Lei nº 14.133/2021, das normas relacionadas ao Estudo Técnico Preliminar e ao Termo de Referência, além de modelos reais de documentos utilizados em processos licitatórios. Por fim, o desenvolvimento do protótipo permitiu aplicar esses conceitos na construção do LaborAI.

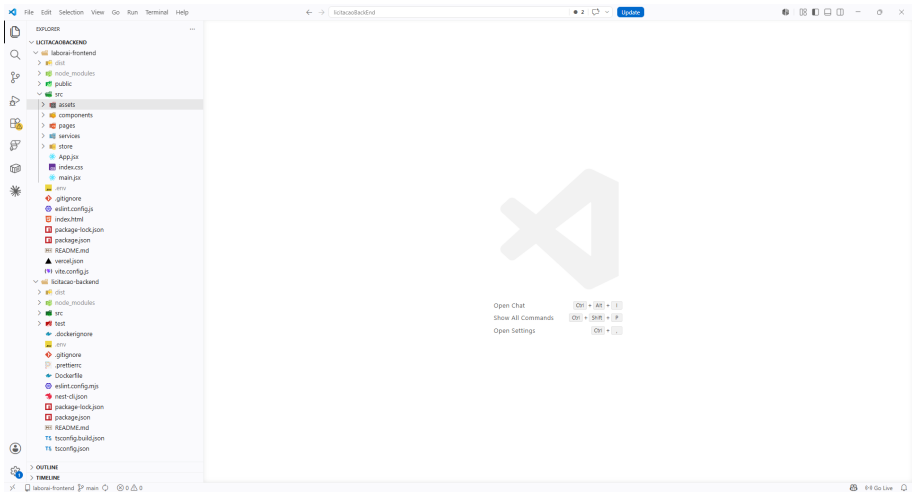
A condução do trabalho seguiu uma sequência prática de desenvolvimento. Primeiro, foi configurada a infraestrutura de dados e autenticação. Em seguida, foi construída a base de conhecimento utilizada pelo mecanismo RAG. Depois, foram implementados os agentes responsáveis pela geração dos documentos. Por fim, foi desenvolvida a interface web, permitindo que o usuário pudesse interagir com os agentes, enviar arquivos, retomar sessões anteriores e acessar os documentos gerados.

3.3 Materiais

3.3.1 Visual Studio Code

O Visual Studio Code foi utilizado como ambiente principal de desenvolvimento do projeto. A ferramenta permitiu a organização dos arquivos do *front-end* e do *back-end*, a execução de comandos no terminal integrado e o uso de extensões para JavaScript, TypeScript e React. A interface da ferramenta durante o desenvolvimento do LaborAI pode ser observada na Figura 4.

Figura 4 – Ambiente de desenvolvimento no Visual Studio Code



Fonte: O autor.

A Figura 4 apresenta o ambiente de desenvolvimento utilizado no LaborAI. No painel lateral é possível observar a estrutura do projeto, dividida entre as pastas `laborai-frontend` e `licitacao-backend`, com os módulos de agentes, RAG, autenticação, documentos e armazenamento organizados separadamente.

3.3.2 Node.js e NPM

O Node.js foi utilizado como ambiente de execução para o desenvolvimento da aplicação, permitindo a criação do *back-end* em NestJS e a execução do projeto *front-end* com React e Vite. Por meio do NPM, foram instaladas e gerenciadas as dependências necessárias para o funcionamento do sistema, como bibliotecas de roteamento, comunicação HTTP, autenticação, manipulação de arquivos, integração com IA e construção dos agentes.

Essa estrutura foi essencial para manter o projeto organizado, pois tanto o *front-end* quanto o *back-end* dependem de pacotes externos para executar suas principais funções. No caso do LaborAI, o ambiente Node.js permitiu integrar bibliotecas modernas em uma mesma aplicação, desde a interface web até os serviços responsáveis pela comunicação com modelos de linguagem.

3.3.3 React e Vite

O React foi utilizado no desenvolvimento do *front-end* do LaborAI, responsável pelas telas de login, página inicial, seleção de agentes, histórico de sessões, interface de chat

e gerenciamento de perfil. A escolha dessa biblioteca permitiu organizar a interface em componentes reutilizáveis, facilitando a manutenção e a evolução visual do sistema.

O Vite foi empregado como ferramenta de criação e execução do projeto React. Sua utilização tornou o processo de desenvolvimento mais rápido, principalmente durante os testes de interface, ajustes visuais e validação do fluxo entre as telas. O *front-end* também passou a consumir a API do *back-end* por meio de requisições HTTP, enviando o token de autenticação nas chamadas protegidas.

3.3.4 NestJS

O NestJS foi utilizado na construção do *back-end* do LaborAI. A estrutura modular do *framework* permitiu separar responsabilidades em serviços, controladores e módulos específicos, tornando mais clara a organização do sistema. Foram criados módulos voltados à autenticação, agentes, RAG, upload de arquivos, sessões, documentos e integração com o Supabase.

Essa separação foi importante porque o LaborAI possui regras de funcionamento distintas: autenticar usuários, proteger rotas, consultar a base vetorial, conduzir conversas com agentes, salvar mensagens, gerar arquivos e disponibilizar documentos. O NestJS serviu como camada central para organizar essas operações e expor os *endpoints* utilizados pela interface web.

3.3.5 Supabase, PostgreSQL e pgvector

O Supabase foi utilizado como infraestrutura principal de dados do LaborAI. Ele concentrou três responsabilidades no projeto: banco de dados PostgreSQL, autenticação de usuários e armazenamento de arquivos. O banco foi estruturado para registrar dados de usuários, sessões, mensagens, documentos, *templates* e a base de conhecimento utilizada pelos agentes.

O PostgreSQL foi utilizado como banco relacional da aplicação. Nele foram criadas as tabelas responsáveis por armazenar os dados principais do sistema, como os perfis dos usuários, o histórico das conversas e os documentos gerados. A extensão pgvector foi utilizada para armazenar os *embeddings* dos trechos da Lei nº 14.133/2021 e dos modelos documentais, permitindo a busca semântica necessária para o funcionamento do RAG.

Além do banco de dados, o Supabase Storage foi utilizado para armazenar os arquivos enviados pelos usuários e os documentos gerados pelos agentes. Para isso, foram separados *buckets* específicos para uploads e documentos finais. Também foram configuradas políticas de segurança com RLS, de modo que cada usuário acessasse apenas os dados relacionados à sua própria conta.

3.3.6 LangChain.js e LangGraph

O LangChain.js foi utilizado como camada de integração entre o sistema e os modelos de linguagem. No LaborAI, essa ferramenta auxiliou na construção de agentes, na definição de ferramentas internas e na comunicação com o modelo responsável pela geração textual.

O LangGraph foi utilizado para estruturar o fluxo dos agentes de Termo de Referência e Estudo Técnico Preliminar. A lógica dos agentes foi construída com base em grafos, nos quais cada nó representa uma etapa do processo, como carregar o *template*, fazer perguntas ao usuário, processar respostas, identificar seções pendentes e gerar o documento final. Essa abordagem permitiu que a conversa não dependesse apenas de uma resposta livre do modelo de linguagem, mas seguisse uma sequência controlada de coleta de informações.

3.3.7 LangSmith

O LangSmith foi utilizado como ferramenta de observabilidade das execuções dos agentes. Por meio dele, foi possível acompanhar os fluxos executados pelo LangGraph, visualizar os nós percorridos, analisar entradas e saídas de cada etapa, verificar metadados da sessão e observar o consumo de tokens durante as chamadas aos modelos de linguagem.

Essa ferramenta foi importante principalmente por se tratar de um sistema que gera documentos sensíveis para processos de licitação. Ao registrar os passos executados pelos agentes, o LangSmith facilita a identificação de falhas, a análise de respostas geradas e a melhoria dos prompts utilizados no sistema.

3.3.8 Bibliotecas de extração e manipulação de arquivos

Para permitir o reaproveitamento de informações a partir de documentos enviados pelo usuário, foram utilizadas bibliotecas capazes de extrair texto de arquivos PDF e DOCX.

Essa funcionalidade permite que o sistema leia o conteúdo de documentos enviados durante a conversa e utilize essas informações como apoio para o preenchimento das seções dos agentes.

No LaborAI, essa etapa foi usada principalmente para permitir que documentos já existentes servissem como referência durante a geração de novos arquivos. O texto extraído é associado à sessão ativa do usuário e pode ser utilizado pelo agente como contexto adicional no momento da geração documental.

3.4 Procedimentos de desenvolvimento

3.4.1 Configuração da infraestrutura de dados

A primeira etapa do desenvolvimento consistiu na configuração da infraestrutura de dados no Supabase. Foram criadas tabelas para armazenar os perfis dos usuários, as sessões de conversa, o histórico de mensagens, os documentos gerados, os *templates* utilizados pelos agentes e a base de conhecimento do sistema.

A tabela de perfis ficou responsável por armazenar informações dos usuários. A tabela de sessões registra cada conversa iniciada com um agente. A tabela de mensagens guarda o histórico das interações entre usuário e assistente. A tabela de documentos registra os arquivos gerados. A tabela de *templates* armazena a estrutura dos documentos utilizados pelos agentes, enquanto a base de conhecimento guarda os trechos indexados da legislação e dos modelos documentais.

Também foram configuradas políticas de RLS, buscando proteger o acesso aos dados no nível do banco. Dessa forma, mesmo com a aplicação consumindo serviços externos, a separação dos dados por usuário não depende apenas da lógica do *back-end*. Essa configuração foi complementada com o uso de dois clientes Supabase no NestJS: um cliente comum, que respeita as políticas de segurança, e um cliente administrativo, utilizado internamente para operações controladas pelo servidor.

3.4.2 Implementação da autenticação

A autenticação foi implementada utilizando Supabase Auth e validação de tokens JWT no *back-end*. O usuário realiza o login pela interface web e recebe um token de acesso. Esse token é enviado em cada requisição protegida por meio do cabeçalho *Authorization*.

No NestJS, foi criada uma estratégia de validação responsável por extrair e verificar o token recebido. Quando o token é válido, os dados do usuário ficam disponíveis para os controladores responsáveis pelas rotas protegidas. Caso o token esteja ausente ou inválido, o servidor retorna uma resposta de acesso não autorizado. Esse fluxo foi adotado para impedir que usuários não autenticados acessem sessões, documentos ou funcionalidades internas do sistema.

3.4.3 Construção da base de conhecimento

A base de conhecimento foi construída a partir da Lei nº 14.133/2021 e de modelos reais de documentos utilizados em processos de licitação. Esses documentos foram processados e divididos em trechos menores, permitindo que o sistema realizasse buscas mais precisas durante a interação com os agentes.

Após a divisão dos textos, cada trecho foi convertido em um *embedding*, isto é, uma representação numérica do seu conteúdo semântico. Esses vetores foram armazenados no PostgreSQL com o suporte da extensão pgvector. Ao final desse processo, a base contava com 522 trechos indexados, sendo formada por partes da legislação e por modelos de documentos utilizados como referência para TR e ETP.

Essa base é utilizada durante a execução dos agentes. Quando o usuário fornece uma informação ou solicita a geração de determinado trecho, o sistema busca os conteúdos mais próximos semanticamente e os envia como contexto para o modelo de linguagem. Essa estratégia reduz a dependência exclusiva do conhecimento interno do modelo e aproxima a resposta da base documental definida para o projeto.

3.4.4 Criação dos templates documentais

Os *templates* foram criados para definir a estrutura dos documentos gerados pelo LaborAI. Cada *template* armazena as seções necessárias, a ordem de preenchimento, a obrigatoriedade de cada parte e as perguntas utilizadas pelo agente para coletar informações do usuário.

No caso do Termo de Referência, o *template* foi estruturado com foco na definição do objeto, especificações, modelo de execução, obrigações, sanções e demais informações necessárias para orientar a contratação. Já no Estudo Técnico Preliminar, a estrutura foi

organizada para tratar da necessidade da contratação, levantamento de mercado, análise de alternativas e justificativa da solução escolhida.

Essa separação foi necessária porque TR e ETP possuem finalidades diferentes dentro do processo licitatório. Enquanto o ETP busca justificar a necessidade e a viabilidade da contratação, o TR detalha o objeto e as condições para sua execução.

3.4.5 Implementação dos agentes

A implementação dos agentes foi realizada com LangChain.js e LangGraph. Para cada tipo de documento, foi criado um fluxo específico de interação. O agente de Termo de Referência e o agente de Estudo Técnico Preliminar seguem estruturas semelhantes, mas utilizam *templates*, perguntas e critérios próprios.

Cada agente possui um estado de conversa, onde são armazenadas informações como o usuário, a sessão, o histórico de mensagens, o *template* carregado, as seções já preenchidas e a próxima etapa a ser executada. Esse estado permite que o sistema continue a conversa mesmo após novas requisições, evitando que o usuário precise reiniciar o processo caso a sessão seja retomada.

A lógica de funcionamento foi dividida em grafos de início e grafos de resposta. O grafo de início carrega o *template* e faz a primeira pergunta ao usuário. O grafo de resposta processa cada mensagem recebida, salva as informações coletadas, identifica se ainda existem seções pendentes e decide se deve continuar perguntando ou gerar o documento final. Essa divisão tornou o fluxo mais controlado e adequado ao preenchimento sequencial de documentos.

3.4.6 Upload e análise de arquivos

Foi implementada uma funcionalidade de upload e análise de arquivos para permitir que o usuário enviasse documentos em PDF ou DOCX durante a conversa com o agente. Após o envio, o sistema extrai o texto do arquivo, salva o documento no Supabase Storage e associa o conteúdo extraído à sessão ativa.

Essa funcionalidade permite que o agente utilize um documento anterior como apoio para preencher partes do TR ou do ETP. Por exemplo, um usuário pode enviar um documento já existente e solicitar que o sistema aproveite suas informações na geração de

outro arquivo. Assim, a ferramenta contribui para reduzir o retrabalho causado pela repetição de dados em diferentes documentos do processo licitatório.

3.4.7 Desenvolvimento do front-end

O *front-end* foi desenvolvido com React e Vite. A aplicação foi organizada em telas principais, como login, página inicial, seleção de agentes e interface de chat. Também foram criados componentes reutilizáveis, como barra de navegação, menu lateral, cards de agentes, área de mensagens e modais de perfil.

A tela de login permite que o usuário acesse o sistema por meio do Supabase Auth. A página inicial apresenta os agentes disponíveis e o histórico de sessões. A tela de chat concentra a interação principal do LaborAI, permitindo o envio de mensagens, upload de arquivos, visualização das respostas do agente e retomada da conversa.

Também foi implementado um controle de rotas protegidas, impedindo que usuários sem autenticação acessem as páginas internas. O token recebido no login é armazenado no cliente e enviado automaticamente nas chamadas à API, permitindo a comunicação segura entre o *front-end* e o *back-end*.

3.4.8 Observabilidade e testes

Durante o desenvolvimento, foram realizados testes funcionais nos principais fluxos do sistema. Foram verificados o login, a proteção das rotas, o envio de mensagens, a retomada de sessões, a busca semântica na base de conhecimento, o upload de arquivos e a geração de documentos.

Além dos testes diretos na aplicação, o LangSmith foi utilizado para observar a execução dos agentes. Com ele, foi possível acompanhar quais etapas do grafo foram executadas, quais dados foram enviados ao modelo, quais respostas foram retornadas e quais metadados estavam associados a cada sessão. Esse acompanhamento contribuiu para ajustar os fluxos dos agentes e identificar pontos de melhoria nos prompts e nas respostas geradas.

4 DESENVOLVIMENTO DO PROJETO

Este capítulo apresenta o desenvolvimento do LaborAI, descrevendo a arquitetura da aplicação, a estrutura de dados, a base de conhecimento utilizada pelos agentes e as principais funcionalidades implementadas. A proposta é demonstrar como os conceitos apresentados na fundamentação teórica foram aplicados na construção de um protótipo funcional voltado à geração automatizada de documentos em processos de licitação pública.

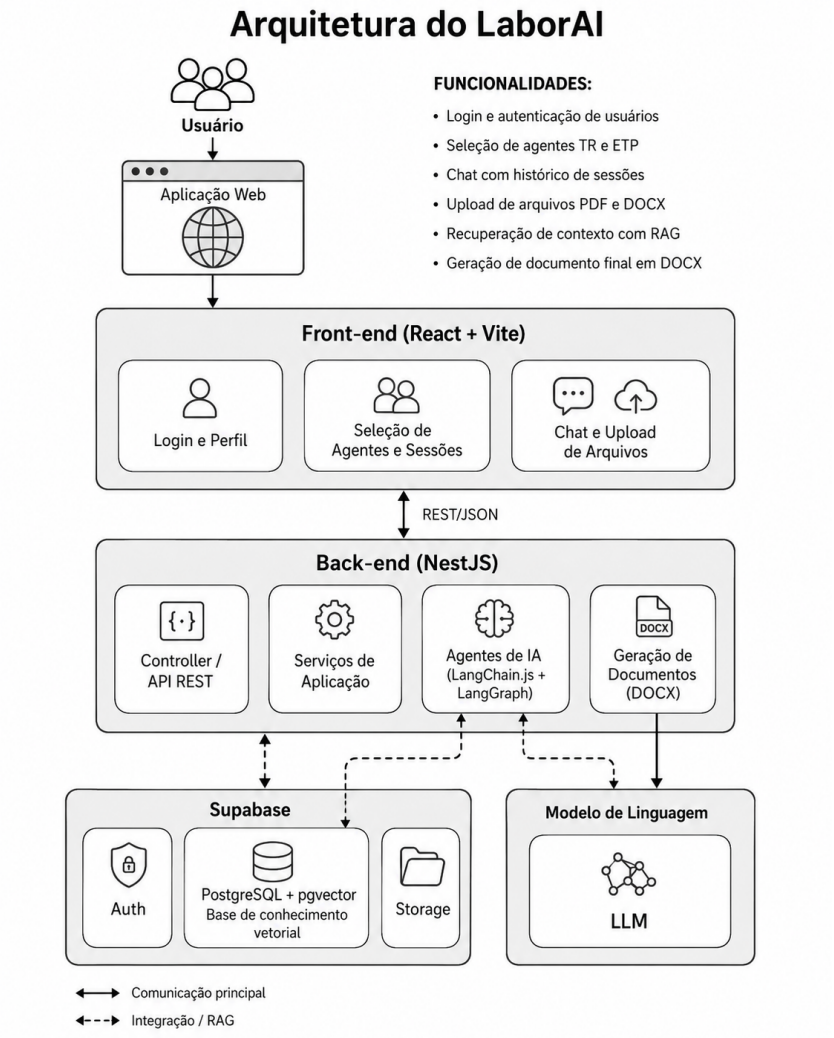
4.1 Visão geral da aplicação

O LaborAI foi desenvolvido como uma aplicação web baseada em arquitetura cliente-servidor. O *front-end*, construído com React e Vite, é responsável pela camada de interação com o usuário, contemplando funcionalidades como login, seleção dos agentes, acesso ao histórico de sessões, envio de mensagens e upload de arquivos. O *back-end*, desenvolvido em NestJS, concentra a lógica da aplicação, incluindo a exposição da API REST, a autenticação, a comunicação com o banco de dados, a execução dos agentes de inteligência artificial e a geração dos documentos.

A aplicação utiliza o Supabase como infraestrutura principal, reunindo autenticação de usuários, banco de dados PostgreSQL e armazenamento de arquivos. A extensão pgvector foi utilizada junto ao PostgreSQL para armazenar os *embeddings* da base de conhecimento vetorial, permitindo a recuperação de contexto durante a execução dos agentes. Já os agentes foram construídos com LangChain.js e LangGraph, compondo a camada responsável por conduzir a interação com o usuário e apoiar a geração dos documentos.

De forma geral, a arquitetura do sistema organiza-se em quatro blocos principais: a interface web, a API de aplicação, a infraestrutura de dados e armazenamento, e a camada de inteligência artificial. O usuário acessa o sistema pela aplicação web, que se comunica com o *back-end* por meio de requisições REST em formato JSON. O *back-end*, por sua vez, coordena as operações com o Supabase, aciona os agentes de IA, consulta a base de conhecimento quando necessário e realiza a geração do documento final em formato DOCX. Essa organização arquitetural está representada na Figura 5.

Figura 5 – Arquitetura do LaborAI



Fonte: O autor.

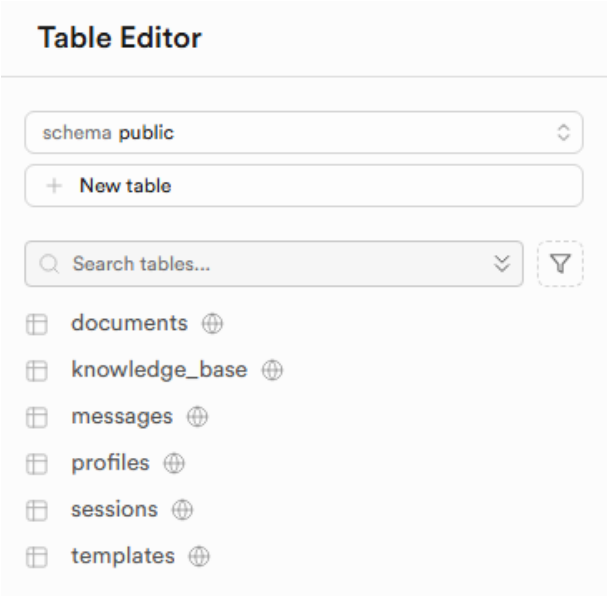
A Figura 5 apresenta a organização arquitetural do LaborAI. Na camada de *front-end*, concentram-se os recursos visuais utilizados pelo usuário, como login, perfil, seleção de agentes, sessões, chat e upload de arquivos. Na camada de *back-end*, ficam os controladores da API REST, os serviços de aplicação, os agentes de IA e o módulo responsável pela geração de documentos em DOCX. A infraestrutura do Supabase fornece autenticação, persistência de dados com PostgreSQL e pgvector, além do armazenamento de arquivos. Por fim, a integração com o modelo de linguagem permite que os agentes utilizem contexto recuperado da base vetorial para apoiar a geração dos documentos.

4.2 Infraestrutura de dados

A estrutura de dados do LaborAI foi organizada no Supabase, utilizando PostgreSQL como banco principal. Foram criadas tabelas para armazenar os perfis dos usuários, as sessões de conversa, o histórico de mensagens, os documentos gerados, os *templates* utilizados pelos agentes e a base de conhecimento do sistema.

A tabela de sessões registra cada conversa iniciada pelo usuário, enquanto a tabela de mensagens armazena o histórico das interações entre usuário e agente. Já a tabela de documentos registra os arquivos gerados ao final do processo. Essa organização permite que o sistema mantenha o histórico das conversas e possibilite a retomada de sessões anteriores. A estrutura das tabelas pode ser visualizada na Figura 6.

Figura 6 – Tabelas criadas no Supabase



Fonte: O autor.

A Figura 6 apresenta as seis tabelas criadas no banco de dados do LaborAI: `documents`, que armazena os documentos gerados; `knowledge_base`, que contém os trechos indexados da base de conhecimento; `messages`, que guarda o histórico das conversas; `profiles`, com os dados dos usuários; `sessions`, que registra cada conversa iniciada; e `templates`, que armazena as estruturas dos documentos utilizadas pelos agentes.

A tabela `knowledgeBase` possui papel central no funcionamento do LaborAI, pois armazena os trechos da Lei nº 14.133/2021 e dos modelos documentais utilizados como

base de consulta. Cada trecho é salvo com seu respectivo *embedding*, permitindo que o sistema realize buscas semânticas durante a execução dos agentes.

Além do banco de dados, foram configurados espaços de armazenamento no Supabase Storage para receber arquivos enviados pelos usuários e documentos gerados pelos agentes. Também foram utilizadas políticas de RLS (*Row Level Security*), de modo que cada usuário acesse apenas os dados relacionados à sua própria conta.

4.3 Base de conhecimento e RAG

A base de conhecimento foi construída para reduzir a dependência exclusiva do conhecimento interno do modelo de linguagem. Para isso, foram utilizados trechos da Lei nº 14.133/2021 e modelos reais de documentos licitatórios, principalmente relacionados ao Termo de Referência e ao Estudo Técnico Preliminar.

O processo de construção da base ocorreu em etapas. Primeiro, os documentos de referência foram processados para extração do conteúdo textual. Em seguida, os textos foram divididos em trechos menores, chamados de *chunks*, permitindo que a recuperação das informações fosse mais precisa. Depois, cada trecho foi convertido em um *embedding* e armazenado no PostgreSQL com suporte do pgvector. O Quadro 1 apresenta um exemplo simplificado de como um trecho da Lei nº 14.133/2021 é representado após esse processo.

Tabela 1 – Exemplo de chunk e embedding na base de conhecimento

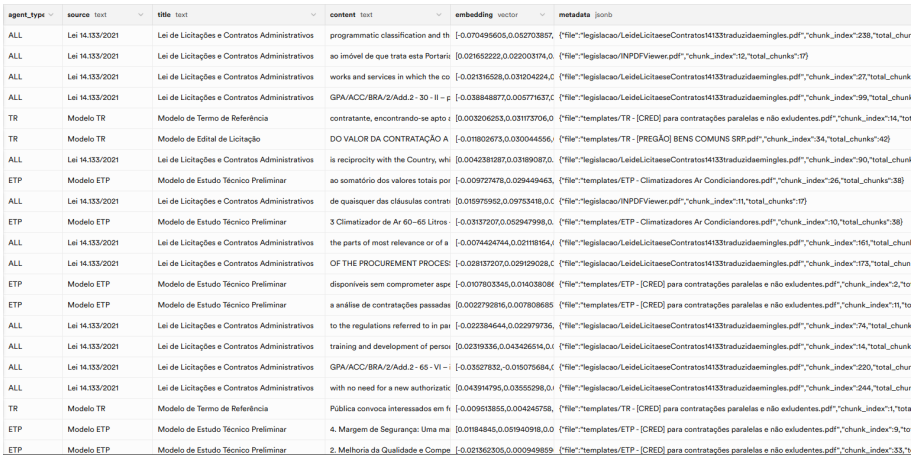
Trecho textual (chunk)	**Embedding (parcial)**
“Art. 18. A fase preparatória do processo licitatório é caracterizada pelo planejamento e deve compatibilizar-se com o plano de contratações anual [...]”	[-0.0704, 0.0216, -0.0213, 0.0042, -0.0388, ...]

Fonte: O autor.

A representação numérica exibida no Quadro 1 corresponde aos primeiros valores do vetor gerado pelo modelo de *embeddings*. Na prática, cada vetor possui 1536 dimensões, capturando o significado semântico do trecho de forma que trechos com conteúdo similar fiquem próximos no espaço vetorial. Essa proximidade é o que permite ao sistema recuperar os trechos mais relevantes durante a execução dos agentes.

Ao final desse processo, a base passou a contar com 522 trechos indexados. Esses trechos são consultados pelos agentes durante a conversa com o usuário. Quando uma informação precisa ser recuperada, o sistema compara semanticamente a solicitação com os vetores armazenados e retorna os conteúdos mais próximos. A estrutura da tabela de conhecimento está representada na Figura 7.

Figura 7 – Tabela da base de conhecimento vetorial



agent_type	source text	title text	content text	embedding vector	metadata jsonb
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	programmatic classification and th	[-0.070495605,0.052703857,	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":238,"total_chur
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	ao imóvel de que trata esta Portaria	[0.021652222,0.022003174,0.	{"file":"legislacao/INPDFViewer.pdf","chunk_index":12,"total_chunks":17}
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	works and services in which the co	[-0.021316528,0.031204224,(	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":27,"total_chunk
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	GPA/ACC/BRA/2/Add.2 - 30 - II – p	[-0.038848877,0.005771637,(	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":99,"total_chunk
TR	Modelo TR	Modelo de Termo de Referência	contratante, encontrando-se apto a	[0.003206253,0.031173706,0	{"file":"templates/TR - [CRED] para contratações paralelas e não exludentes.pdf","chunk_index":14,"tot
TR	Modelo TR	Modelo de Edital de Licitação	DO VALOR DA CONTRATAÇÃO A	[-0.011802673,0.030044556,(	{"file":"templates/TR - [PREGÃO] BENS COMUNS SRP.pdf","chunk_index":34,"total_chunks":42}
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	is reciprocity with the Country, whi	[0.0042381287,0.03189087,0.	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":90,"total_chunk
ETP	Modelo ETP	Modelo de Estudo Técnico Preliminar	ao somatório dos valores totais por	[-0.009727478,0.029449463,	{"file":"templates/ETP - Climatizadores Ar Condiciandores.pdf","chunk_index":26,"total_chunks":38}
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	de quaisquer das cláusulas contrat	[0.015975952,0.09753418,0.(	{"file":"legislacao/INPDFViewer.pdf","chunk_index":11,"total_chunks":17}
ETP	Modelo ETP	Modelo de Estudo Técnico Preliminar	3 Climatizador de Ar 60–65 Litros -	[-0.03137207,0.052947998,0.	{"file":"templates/ETP - Climatizadores Ar Condiciandores.pdf","chunk_index":10,"total_chunks":38}
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	the parts of most relevance or of a	[-0.0074424744,0.021118164,(	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":161,"total_chunk
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	OF THE PROCUREMENT PROCES	[-0.028137207,0.029129028,(	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":173,"total_chun
ETP	Modelo ETP	Modelo de Estudo Técnico Preliminar	disponíveis sem comprometer aspe	[-0.0107803345,0.014038086	{"file":"templates/ETP - [CRED] para contratações paralelas e não exludentes.pdf","chunk_index":2,"to
ETP	Modelo ETP	Modelo de Estudo Técnico Preliminar	a análise de contratações passadas	[0.002792816,0.007808685	{"file":"templates/ETP - [CRED] para contratações paralelas e não exludentes.pdf","chunk_index":11,"to
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	to the regulations referred to in par	[-0.022384644,0.022979736,	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":74,"total_chunk
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	training and development of perso	[0.02319336,0.043426514,0.(	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":14,"total_chunk
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	GPA/ACC/BRA/2/Add.2 - 65 - VI – i	[-0.03527832,-0.015075684,(	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":220,"total_chur
ALL	Lei 14.133/2021	Lei de Licitações e Contratos Administrativos	with no need for a new authorizatic	[0.043914795,0.03555298,0.(	{"file":"legislacao/LeideLicitaeseContratos14133traduzidaemingles.pdf","chunk_index":244,"total_chur
TR	Modelo TR	Modelo de Termo de Referência	Pública convoca interessados em fa	[-0.009513855,0.004245758,	{"file":"templates/TR - [CRED] para contratações paralelas e não exludentes.pdf","chunk_index":1,"tota
ETP	Modelo ETP	Modelo de Estudo Técnico Preliminar	4. Margem de Segurança: Uma ma	[0.0118484,0.051940918,0.0	{"file":"templates/ETP - [CRED] para contratações paralelas e não exludentes.pdf","chunk_index":9,"to
ETP	Modelo ETP	Modelo de Estudo Técnico Preliminar	2. Melhoria da Qualidade e Compe	[-0.021362305,0.000949859	{"file":"templates/ETP - [CRED] para contratações paralelas e não exludentes.pdf","chunk_index":33,"t

Fonte: O autor.

A Figura 7 apresenta registros da tabela `knowledge_base`. Cada linha corresponde a um trecho indexado e contém o tipo de agente ao qual pertence (`agent_type`: ALL, TR ou ETP), a fonte do documento (`source`), o título, o trecho de conteúdo textual, o vetor de *embedding* gerado e os metadados com informações sobre o arquivo e o índice do trecho. Essa estrutura permite que o sistema filtre os trechos mais relevantes para cada tipo de agente durante a busca semântica.

Essa abordagem segue a lógica do RAG, pois combina a geração textual dos modelos de linguagem com a recuperação de informações presentes em uma base externa. No LaborAI, isso permite que os agentes utilizem uma base documental previamente definida antes de gerar partes do TR ou do ETP.

4.4 Templates documentais

Os *templates* foram criados para definir a estrutura dos documentos gerados pelo LaborAI. Cada *template* contém as seções do documento, a ordem de preenchimento, a obrigatoriedade de cada parte e as perguntas utilizadas pelo agente durante a conversa

com o usuário.

No caso do Termo de Referência, o *template* foi organizado para coletar informações relacionadas ao objeto da contratação, especificações, modelo de execução, obrigações, condições de pagamento e demais elementos necessários para orientar a contratação. Já no Estudo Técnico Preliminar, o *template* foi estruturado com foco na necessidade da contratação, levantamento de alternativas, análise de viabilidade e justificativa da solução escolhida.

Essa separação foi necessária porque TR e ETP possuem finalidades diferentes dentro do processo licitatório. O ETP está mais relacionado à análise da necessidade e da viabilidade da contratação, enquanto o TR detalha o objeto e as condições para sua execução.

4.5 Agentes de inteligência artificial

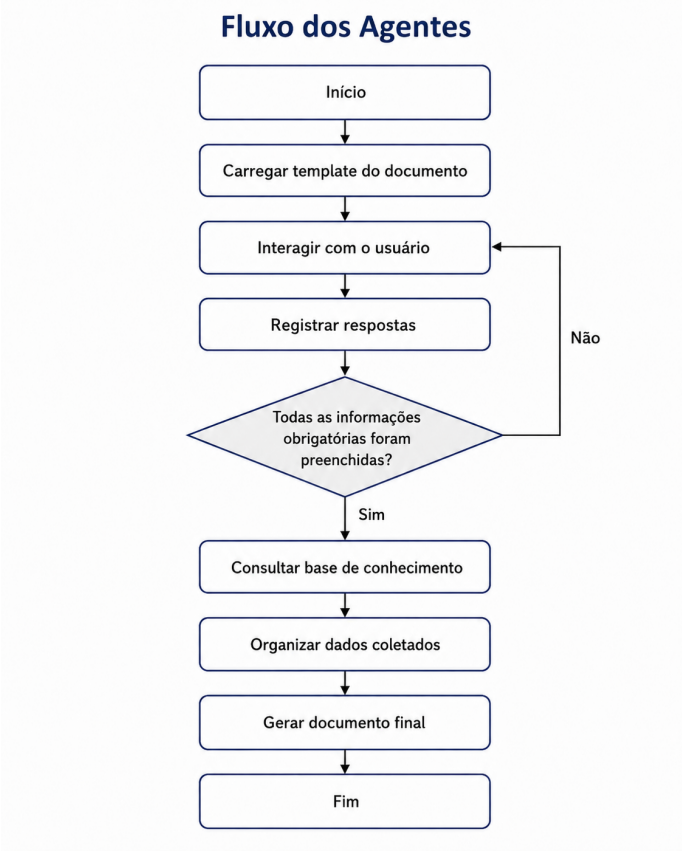
Os agentes do LaborAI foram desenvolvidos com LangChain.js e LangGraph. A principal função desses agentes é conduzir o usuário na elaboração dos documentos, fazendo perguntas, armazenando respostas, consultando a base de conhecimento e gerando o conteúdo final.

Foram implementados dois agentes principais: um para Termo de Referência e outro para Estudo Técnico Preliminar. Ambos seguem uma lógica semelhante, mas utilizam *templates* diferentes, pois cada documento exige informações próprias.

Cada agente possui um estado de conversa, no qual são armazenadas informações como o usuário, a sessão, o histórico de mensagens, o *template* carregado, as seções já preenchidas e a próxima etapa a ser executada. Esse estado permite que o sistema continue a conversa mesmo após novas requisições, sem perder as informações já fornecidas.

O fluxo dos agentes foi estruturado em etapas. Primeiro, o sistema carrega o *template* correspondente ao documento escolhido. Depois, o agente faz perguntas ao usuário e registra as respostas nas seções adequadas. Quando todas as informações obrigatórias são preenchidas, o sistema consulta a base de conhecimento, organiza os dados coletados e gera o documento final.

Figura 8 – Fluxo dos agentes no LaborAI



Fonte: O autor.

A Figura 8 apresenta o fluxo de execução dos agentes do LaborAI. O processo inicia com o carregamento do *template* e segue com a interação com o usuário e o registro das respostas. Enquanto houver seções pendentes, o agente retorna ao ciclo de interação. Ao concluir o preenchimento, o sistema consulta a base de conhecimento, organiza os dados e gera o documento final em formato DOCX.

4.6 Upload, geração de documentos e armazenamento

O LaborAI também permite o envio de arquivos em PDF e DOCX durante a conversa com o agente. Após o upload, o sistema extrai o conteúdo textual do arquivo e associa essas informações à sessão ativa. Assim, o agente pode utilizar o documento enviado como apoio para preencher partes do TR ou do ETP.

Essa funcionalidade foi pensada para reduzir o retrabalho na elaboração de documentos. Por exemplo, quando o usuário já possui um documento anterior, ele pode enviá-lo ao sistema e solicitar que suas informações sejam aproveitadas na geração de outro arquivo.

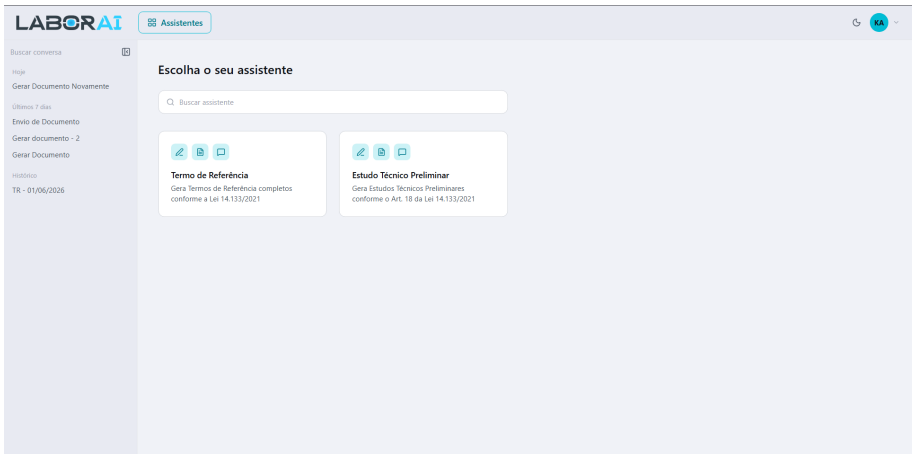
Com isso, a aplicação auxilia no reaproveitamento de dados que normalmente seriam digitados novamente.

Após a coleta das informações necessárias, o sistema gera o documento final em formato DOCX. Esse formato foi escolhido por ser editável e amplamente utilizado em ambientes administrativos. O arquivo gerado é armazenado no Supabase Storage, e suas informações são registradas na tabela de documentos, permitindo que o usuário acesse posteriormente o arquivo produzido.

4.7 Interface web

A interface web foi desenvolvida com React e Vite. A aplicação possui tela de login, página inicial com seleção de agentes, histórico de sessões e tela de chat. A tela de login permite que o usuário acesse o sistema por meio do Supabase Auth. A página inicial apresenta os agentes disponíveis e o histórico de sessões. A tela de chat concentra a interação principal do LaborAI, permitindo o envio de mensagens, upload de arquivos, visualização das respostas do agente e retomada da conversa. A interface inicial do sistema pode ser visualizada na Figura 9.

Figura 9 – Tela inicial do LaborAI



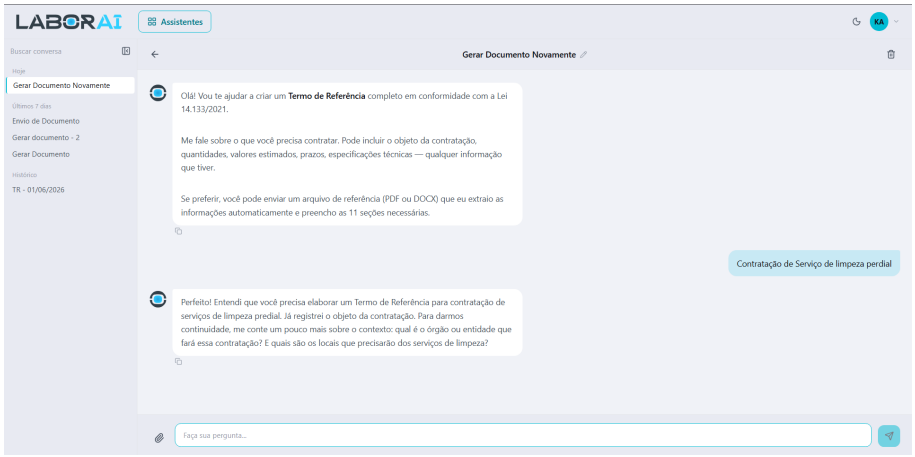
Fonte: O autor.

A Figura 9 apresenta a página inicial do LaborAI. Na área central, o usuário pode escolher entre os dois agentes disponíveis: Termo de Referência e Estudo Técnico Preliminar. Cada card exibe o nome do agente e uma breve descrição do documento que ele é capaz de gerar. Na barra lateral esquerda, o histórico de sessões anteriores é exibido com

agrupamento por data, permitindo o acesso rápido a conversas já iniciadas.

Também foram implementados recursos complementares, como retomada de conversas anteriores, edição do título da sessão, exclusão de sessões, modal de perfil e recuperação de senha. Esses elementos ajudam a tornar o sistema mais completo e adequado ao uso contínuo. A interface de chat está representada na Figura 10.

Figura 10 – Interface de conversa com o agente TR



Fonte: O autor.

A Figura 10 apresenta a tela de chat do LaborAI durante uma sessão de geração de Termo de Referência. O agente inicia a conversa de forma aberta, solicitando informações sobre o objeto da contratação. Após o usuário informar que se trata de um serviço de limpeza predial, o agente confirma a informação e avança naturalmente para coletar o contexto do órgão contratante. Na barra lateral, o histórico de sessões exibe as conversas anteriores agrupadas por data.

No *front-end*, as rotas internas foram protegidas para impedir o acesso de usuários não autenticados. O token recebido no login é enviado automaticamente nas requisições feitas ao *back-end*, permitindo que a comunicação com a API ocorra de forma autenticada.

Por fim, para fins de avaliação e demonstração do protótipo, a aplicação foi disponibilizada em ambiente web, permitindo o acesso externo ao sistema. O acesso pode ser realizado por meio do endereço:

https://laborai-frontend-two.vercel.app/

Para testes, foi criada uma conta específica, sem vínculo com dados reais, com as seguintes credenciais:

E-mail: `testetcc@gmail.com`
Senha: `12345678`

Essa conta tem como finalidade permitir a navegação pelas principais funcionalidades do LaborAI, incluindo seleção de agentes, criação de sessões, envio de mensagens, upload de arquivos, adição de novos usuários e geração de documentos.

4.8 Observabilidade e validação do funcionamento

Durante o desenvolvimento, o LangSmith foi utilizado para acompanhar a execução dos agentes. Por meio dele, foi possível visualizar as etapas percorridas no LangGraph, os dados enviados ao modelo de linguagem, as respostas retornadas e os metadados relacionados a cada sessão.

Esse acompanhamento foi importante para identificar falhas, ajustar *prompts* e compreender o comportamento dos agentes durante a geração dos documentos. Como o LaborAI trabalha com documentos sensíveis ao contexto administrativo, a observabilidade auxiliou na análise dos fluxos e na melhoria das respostas geradas.

Além disso, foram realizados testes funcionais nos principais fluxos da aplicação, incluindo autenticação, envio de mensagens, busca semântica, upload de arquivos, retomada de sessões e geração de documentos DOCX.

5 RESULTADOS E DISCUSSÃO

5.1 Visão geral dos resultados

O desenvolvimento do LaborAI resultou em um protótipo web funcional voltado à geração assistida de documentos em processos de licitação pública. A aplicação implementada integra interface web, autenticação de usuários, base de conhecimento vetorial, agentes conversacionais com fluxo controlado, upload de arquivos, memória de sessão e geração de documentos no formato DOCX.

De forma geral, o sistema atingiu o objetivo proposto neste trabalho, pois demonstrou a viabilidade técnica de utilizar agentes de inteligência artificial para auxiliar na produção inicial de Termos de Referência e Estudos Técnicos Preliminares. A solução não substitui a revisão técnica ou jurídica desses documentos, mas oferece uma forma mais organizada de coletar informações, reaproveitar dados e estruturar documentos com apoio em uma base de conhecimento específica.

O LaborAI também demonstrou que a técnica RAG pode ser aplicada em um contexto prático de documentação pública. Ao utilizar trechos indexados da Lei nº 14.133/2021 e modelos documentais como base de consulta, os agentes deixam de depender apenas do conhecimento interno do modelo de linguagem e passam a trabalhar com informações recuperadas de uma fonte previamente definida.

5.2 Prova de Conceito

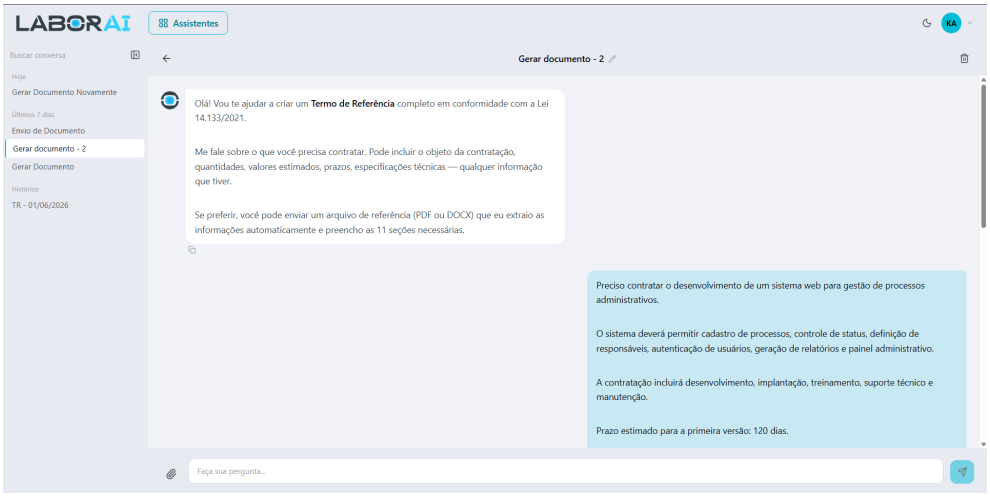
Para demonstrar o funcionamento do LaborAI em um cenário prático, esta seção apresenta dois fluxos de geração de Termo de Referência: o primeiro por meio de conversa guiada com o agente, e o segundo por meio do envio de um documento de referência. Em ambos os casos, o contexto utilizado é a contratação de desenvolvimento de um sistema web para gestão de processos administrativos de um órgão público.

5.2.1 Cenário 1 - Geração por conversa guiada

No primeiro cenário, o usuário inicia uma sessão com o agente de Termo de Referência e fornece as informações da contratação de forma conversacional, sem necessidade de preencher formulários ou seguir uma estrutura predefinida.

Conforme apresentado na Figura 11, o agente inicia a conversa solicitando informações sobre o objeto da contratação. O usuário descreve que precisa contratar o desenvolvimento de um sistema web para gestão de processos administrativos, informando funcionalidades esperadas, prazo estimado de 120 dias, valor estimado de R$ 300.000,00 e vigência contratual de 12 meses. O agente confirma o entendimento e avança naturalmente para a próxima etapa.

Figura 11 – Início da conversa com o agente TR

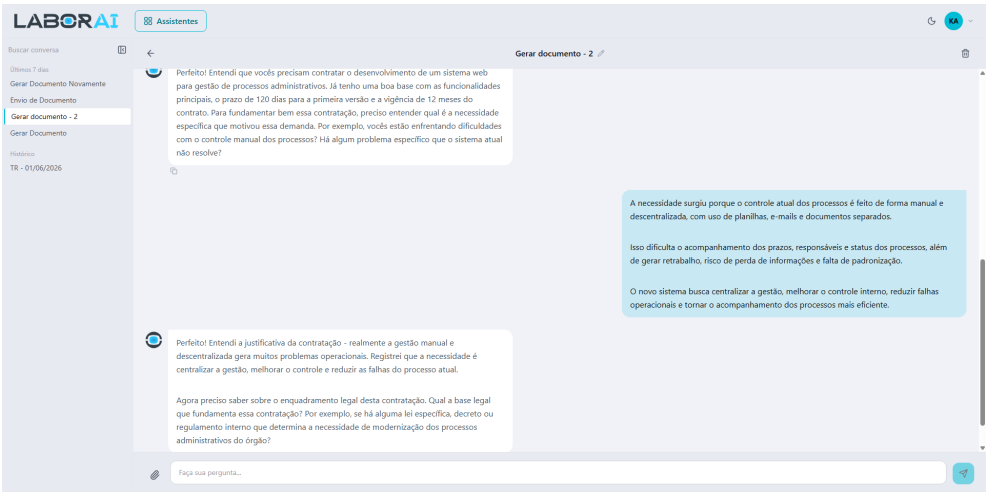


Fonte: O autor.

A Figura 11 ilustra como o agente processa múltiplas informações fornecidas em uma única mensagem, confirmando o que foi compreendido e solicitando complementações de forma natural. Nesse momento, o agente já registrou internamente o objeto da contratação, as especificações, o prazo e o valor estimado.

Na etapa seguinte, apresentada na Figura 12, o usuário complementa as informações explicando a necessidade da contratação: o controle atual é realizado de forma manual e descentralizada, com planilhas e e-mails separados, gerando retrabalho e falta de padronização. O agente confirma a justificativa e solicita o enquadramento legal da contratação.

Figura 12 – Coleta de informações durante a conversa

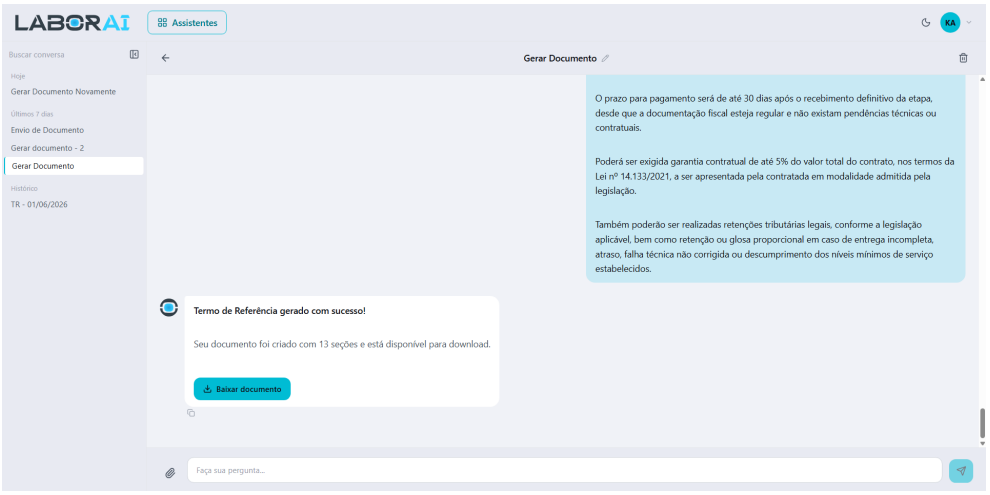


Fonte: O autor.

A Figura 12 demonstra o comportamento conversacional do agente, que avança progressivamente pelas seções do documento à medida que as informações são fornecidas. O fluxo não segue uma ordem rígida de formulário, mas se adapta ao contexto apresentado pelo usuário.

Ao concluir a coleta de todas as informações obrigatórias, o sistema consulta a base de conhecimento, organiza os dados e gera o documento final. O resultado pode ser observado na Figura 13.

Figura 13 – Documento gerado ao final da conversa



Fonte: O autor.

Conforme apresentado na Figura 13, o agente confirma a geração bem-sucedida

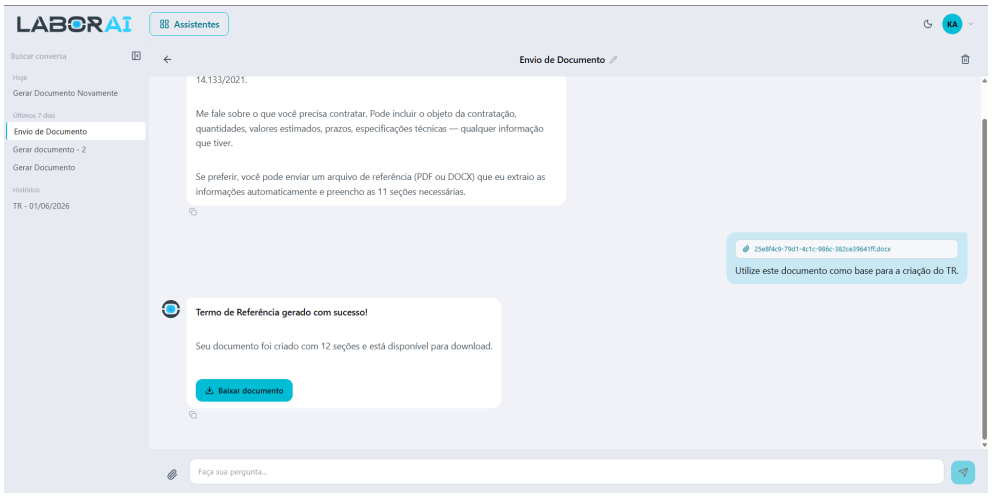
do Termo de Referência com 13 seções e disponibiliza o botão de download do arquivo DOCX. O documento gerado inclui seções como objeto, justificativa, especificações técnicas, modelo de execução, condições de pagamento e demais elementos exigidos pela Lei nº 14.133/2021.

5.2.2 Cenário 2 - Geração por envio de documento

O segundo cenário demonstra a funcionalidade de reaproveitamento de documentos existentes. Em vez de fornecer as informações por conversa, o usuário envia um arquivo DOCX como referência e solicita que o agente utilize seu conteúdo para gerar o Termo de Referência. Esse fluxo é especialmente útil quando o órgão já possui estudos preliminares ou rascunhos anteriores que podem ser aproveitados.

Conforme ilustrado na Figura 14, o usuário envia um arquivo DOCX e instrui o agente a utilizá-lo como base para a criação do TR. O sistema extrai automaticamente o conteúdo do documento, identifica as informações relevantes para cada seção e gera o Termo de Referência sem necessidade de interação adicional.

Figura 14 – Geração de TR a partir de documento enviado pelo usuário



Fonte: O autor.

A Figura 14 apresenta o resultado desse fluxo: o agente confirma a geração do Termo de Referência com 12 seções e disponibiliza o link para download. O documento foi gerado integralmente a partir das informações extraídas do arquivo enviado, sem que o usuário precisasse responder a nenhuma pergunta adicional.

Os dois cenários demonstram que o LaborAI é capaz de se adaptar a diferentes perfis de uso. Servidores com informações já organizadas podem utilizar o envio de documento para geração imediata. Servidores que ainda estão organizando as informações podem utilizar a conversa guiada para construir o documento progressivamente. Em ambos os casos, o resultado é um arquivo DOCX estruturado, pronto para revisão e uso administrativo.

5.3 Funcionalidades implementadas

A aplicação apresenta um conjunto de funcionalidades que permite executar o fluxo principal de geração assistida de documentos. Entre as entregas realizadas, destacam-se a autenticação de usuários, a criação de sessões de conversa, o armazenamento do histórico das mensagens, a busca semântica na base de conhecimento e a geração dos documentos finais em formato editável.

A autenticação foi implementada com Supabase Auth e validação de JWT no *back-end*. Com isso, apenas usuários autenticados conseguem acessar as funcionalidades internas da aplicação, como conversas com os agentes, upload de arquivos e acesso aos documentos gerados. Além disso, as políticas de segurança configuradas no banco contribuíram para isolar os dados de cada usuário.

A memória de sessão também foi uma entrega importante. Como a geração de um TR ou ETP ocorre em várias etapas, o sistema precisa manter o progresso da conversa. O LaborAI armazena tanto o histórico visível das mensagens quanto o estado interno do agente, permitindo que o usuário retome uma sessão anterior sem perder as informações já fornecidas.

Outro resultado relevante foi a construção da base de conhecimento vetorial. A base foi formada por 522 trechos indexados, extraídos da Lei nº 14.133/2021 e de modelos reais de documentos licitatórios. Esses trechos foram convertidos em *embeddings* e armazenados no PostgreSQL com suporte do pgvector, permitindo a realização de buscas por similaridade semântica.

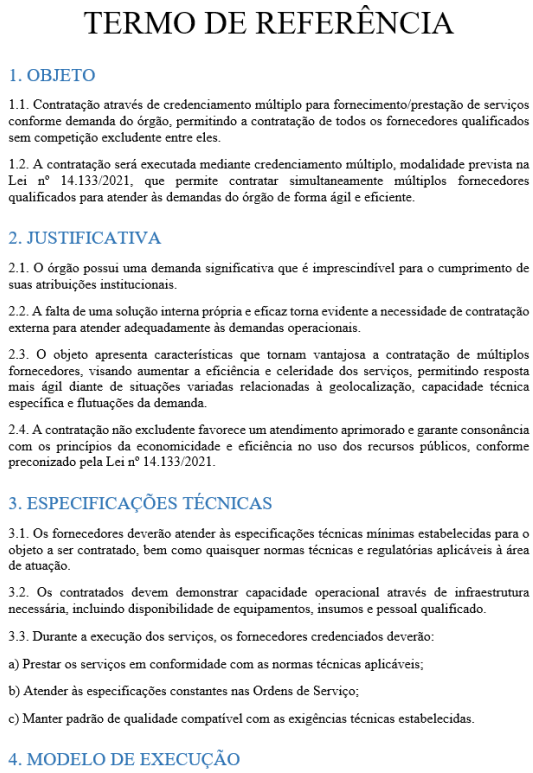
Também foram implementados dois agentes principais: o agente de Termo de Referência e o agente de Estudo Técnico Preliminar. Ambos seguem fluxos controlados com LangGraph, utilizando *templates* específicos para conduzir o usuário durante o preenchimento das informações. Essa abordagem permitiu que cada documento fosse gerado a

partir de uma sequência estruturada de perguntas e respostas, em vez de uma geração livre e sem controle.

A funcionalidade de upload e análise de arquivos ampliou o uso do sistema. Com ela, o usuário pode enviar documentos em PDF ou DOCX para que o conteúdo seja extraído e utilizado como apoio durante a conversa. Essa funcionalidade é especialmente útil em casos em que informações já existentes precisam ser reaproveitadas em outro documento, como ocorre quando dados de um ETP servem como base para a elaboração de um TR.

A geração em DOCX permitiu que o resultado final fosse entregue em um formato compatível com a rotina administrativa. O documento gerado pode ser baixado, revisado, editado e ajustado antes de qualquer utilização oficial. Um exemplo de documento gerado pelo sistema está apresentado na Figura 15.

Figura 15 – Documento DOCX gerado pelo LaborAI



TERMO DE REFERÊNCIA

1. OBJETO

1.1. Contratação através de credenciamento múltiplo para fornecimento/prestação de serviços conforme demanda do órgão, permitindo a contratação de todos os fornecedores qualificados sem competição excludente entre eles.

1.2. A contratação será executada mediante credenciamento múltiplo, modalidade prevista na Lei nº 14.133/2021, que permite contratar simultaneamente múltiplos fornecedores qualificados para atender às demandas do órgão de forma ágil e eficiente.

2. JUSTIFICATIVA

2.1. O órgão possui uma demanda significativa que é imprescindível para o cumprimento de suas atribuições institucionais.

2.2. A falta de uma solução interna própria e eficaz torna evidente a necessidade de contratação externa para atender adequadamente às demandas operacionais.

2.3. O objeto apresenta características que tornam vantajosa a contratação de múltiplos fornecedores, visando aumentar a eficiência e celeridade dos serviços, permitindo resposta mais ágil diante de situações variadas relacionadas à geolocalização, capacidade técnica específica e flutuações da demanda.

2.4. A contratação não excludente favorece um atendimento aprimorado e garante consonância com os princípios da economicidade e eficiência no uso dos recursos públicos, conforme preconizado pela Lei nº 14.133/2021.

3. ESPECIFICAÇÕES TÉCNICAS

3.1. Os fornecedores deverão atender às especificações técnicas mínimas estabelecidas para o objeto a ser contratado, bem como quaisquer normas técnicas e regulatórias aplicáveis à área de atuação.

3.2. Os contratados devem demonstrar capacidade operacional através de infraestrutura necessária, incluindo disponibilidade de equipamentos, insumos e pessoal qualificado.

3.3. Durante a execução dos serviços, os fornecedores credenciados deverão:

a) Prestar os serviços em conformidade com as normas técnicas aplicáveis;

b) Atender às especificações constantes nas Ordens de Serviço;

c) Manter padrão de qualidade compatível com as exigências técnicas estabelecidas.

4. MODELO DE EXECUÇÃO

Fonte: O autor.

A Figura 15 apresenta um exemplo de Termo de Referência gerado pelo LaborAI.

O documento é estruturado com título centralizado e seções numeradas, incluindo Objeto, Justificativa, Especificações Técnicas e Modelo de Execução. O conteúdo é gerado com base nas informações fornecidas pelo usuário durante a conversa com o agente e nos trechos recuperados da base de conhecimento, resultando em um arquivo editável no formato DOCX.

5.4 Resultados relacionados aos objetivos

O objetivo geral deste trabalho foi desenvolver um sistema web baseado em agentes de inteligência artificial capaz de conduzir servidores públicos na elaboração automatizada de Termos de Referência e Estudos Técnicos Preliminares com base na Lei nº 14.133/2021. Esse objetivo foi alcançado por meio da implementação do LaborAI, que reúne agentes conversacionais, base de conhecimento, geração de documentos e interface web em um mesmo protótipo.

O primeiro objetivo específico consistia em construir uma base de conhecimento com a técnica RAG, indexando a Lei nº 14.133/2021 e modelos reais de documentos de licitação. Esse objetivo foi atendido com a criação da tabela *knowledgeBase*, o uso de *embeddings* e o armazenamento dos vetores com pgvector.

O segundo objetivo específico previa a implementação de agentes conversacionais com fluxo controlado por meio do LangGraph. Esse objetivo também foi atendido, uma vez que foram criados fluxos específicos para TR e ETP, com controle de estado, identificação de seções pendentes e geração do documento ao final do preenchimento.

O terceiro objetivo específico tratava da autenticação, do controle de acesso e da persistência de sessões de conversa. Esse ponto foi implementado com Supabase Auth, validação de JWT no *back-end*, armazenamento das mensagens e salvamento do estado dos agentes.

O quarto objetivo específico previa o upload e a análise de arquivos PDF e DOCX para reaproveitamento de informações. Essa funcionalidade foi implementada com extração de texto dos arquivos enviados, armazenamento no Supabase Storage e associação do conteúdo à sessão ativa do usuário.

O quinto objetivo específico estava relacionado à geração de arquivos DOCX com armazenamento em nuvem e disponibilização de link para download. Esse objetivo foi

alcançado com a geração do documento final, o registro na tabela de documentos e o armazenamento do arquivo no Supabase Storage.

Por fim, o sexto objetivo específico previa o desenvolvimento de uma interface web que permitisse ao usuário interagir com os agentes e gerenciar o histórico de sessões. Esse objetivo foi atendido com a construção do *front-end* em React, incluindo tela de login, página inicial, interface de chat, histórico de conversas e recursos de gerenciamento da sessão.

5.5 Trabalhos Correlatos

Esta seção apresenta uma análise comparativa entre o LaborAI e soluções relacionadas ao seu escopo, abrangendo plataformas oficiais de licitação, ferramentas baseadas em inteligência artificial para documentos jurídicos e pesquisas acadêmicas sobre RAG em contextos legais.

O Compras.gov.br é a principal plataforma digital de compras públicas do Governo Federal, reunindo funcionalidades para publicação de avisos, realização de pregões eletrônicos e gestão de contratos (BRASIL, [s.d.]). Embora incorpore as exigências da Lei nº 14.133/2021, a plataforma não dispõe de recursos de inteligência artificial para apoiar a redação de documentos. O preenchimento do ETP e do TR permanece como tarefa manual do servidor.

No cenário internacional, ferramentas como Harvey AI e CoCounsel oferecem assistência jurídica baseada em LLMs, com foco em pesquisa legal e revisão de contratos. Contudo, essas soluções são voltadas ao ordenamento jurídico estrangeiro e não contemplam as especificidades da legislação brasileira de licitações.

Na literatura acadêmica, Dahl et al. (2024) demonstraram que LLMs alucinam em pelo menos 58% das consultas jurídicas sem base de conhecimento estruturada, reforçando a necessidade de RAG e validação humana em aplicações legais. Magesh et al. (2024) avaliaram ferramentas comerciais com RAG e identificaram que mesmo em sistemas especializados a ocorrência de respostas imprecisas permanece relevante.

6 CONCLUSÕES

6.1 Síntese do trabalho

Este trabalho apresentou o desenvolvimento do LaborAI, um sistema web baseado em agentes de inteligência artificial voltado à geração automatizada de Termos de Referência e Estudos Técnicos Preliminares em conformidade com a Lei nº 14.133/2021. A proposta partiu da identificação de um problema prático da administração pública brasileira: a elaboração manual desses documentos é um processo trabalhoso, suscetível a erros e que exige conhecimento técnico-jurídico especializado dos servidores envolvidos.

Para responder à questão de pesquisa proposta, foi desenvolvido um protótipo funcional que combina agentes conversacionais com fluxo controlado por LangGraph, uma base de conhecimento construída com a técnica RAG a partir da Lei nº 14.133/2021 e de modelos documentais reais, e uma interface web que permite ao usuário interagir com os agentes, enviar arquivos de referência e acessar os documentos gerados. Os resultados obtidos demonstram que a aplicação de agentes de inteligência artificial ao contexto de licitações públicas é tecnicamente viável e capaz de produzir documentos estruturados, embasados em legislação verificada e prontos para revisão administrativa.

6.2 Limitações do sistema

Apesar dos resultados obtidos, o LaborAI apresenta limitações que devem ser consideradas antes de qualquer uso em ambiente real de trabalho.

A principal delas é o risco de alucinação dos modelos de linguagem. Mesmo com a técnica RAG reduzindo a dependência do conhecimento interno do modelo, estudos demonstram que LLMs aplicados ao domínio legal ainda produzem respostas incorretas ou imprecisas em parcela relevante das consultas (DAHL et al., 2024; MAGESH et al., 2024). No contexto de licitações públicas, uma informação gerada incorretamente pode comprometer a validade jurídica do documento e expor o órgão a questionamentos nos órgãos de controle.

Relacionada a essa questão está a dependência do modelo de linguagem utilizado. O desempenho do sistema está diretamente vinculado às capacidades e limitações do modelo contratado. Atualizações, descontinuações ou mudanças de comportamento do

modelo podem afetar a qualidade das respostas sem que o sistema seja alterado.

Outra limitação relevante diz respeito à atualização da legislação. A base de conhecimento foi construída sobre a Lei nº 14.133/2021 e as instruções normativas vigentes no momento do desenvolvimento. Alterações posteriores na legislação, novas instruções normativas ou entendimentos dos órgãos de controle não são automaticamente incorporados ao sistema, exigindo atualização periódica da base para manter a conformidade legal das respostas geradas.

Há ainda o risco de interpretação jurídica equivocada. Documentos licitatórios envolvem decisões administrativas que demandam análise contextual, conhecimento do caso concreto e responsabilidade técnica do servidor. O LaborAI organiza e estrutura informações fornecidas pelo usuário, mas não substitui o julgamento humano necessário para avaliar a adequação jurídica do conteúdo gerado. Por isso, a revisão técnica e jurídica posterior é indispensável e não deve ser dispensada em nenhuma circunstância.

6.3 Trabalhos futuros

Como trabalhos futuros, uma das possibilidades é a implementação de um agente voltado à geração de editais. Esse agente poderia utilizar como base os documentos já produzidos pelo sistema, especialmente o TR, para auxiliar na construção de uma peça mais completa do processo licitatório.

Outra melhoria possível é a ampliação da base de conhecimento. A inclusão de novos modelos, manuais, instruções normativas e entendimentos de órgãos de controle poderia tornar as respostas dos agentes mais adequadas a diferentes cenários de contratação. Também seria possível criar bases específicas para tipos distintos de objeto, como serviços continuados, aquisições de bens, obras ou contratações de tecnologia.

Também se propõe, como evolução do sistema, a criação de um mecanismo de revisão e controle de versões dos documentos gerados. Esse recurso permitiria acompanhar alterações realizadas após a geração inicial, comparar versões e manter um histórico mais claro da evolução de cada documento.

Outra possibilidade é a implementação de exportação em PDF, além do formato DOCX. Embora o DOCX seja adequado para edição e revisão, o PDF pode ser útil para compartilhamento, arquivamento e apresentação formal dos documentos.

Por fim, a realização de testes com usuários reais seria uma etapa importante para avaliar a usabilidade e a efetividade do sistema. A aplicação poderia ser testada por servidores, equipes de planejamento ou profissionais envolvidos em contratações públicas, permitindo coletar percepções sobre clareza das perguntas, qualidade dos documentos gerados, tempo economizado e adequação da ferramenta à rotina administrativa.

6.4 Considerações finais

O desenvolvimento do LaborAI representou uma oportunidade de aplicar, em um problema real da administração pública, conceitos atuais de desenvolvimento web, inteligência artificial, recuperação aumentada por geração e agentes conversacionais. O trabalho partiu de uma necessidade prática, relacionada à elaboração de documentos de licitação, e resultou em um protótipo capaz de conduzir o usuário na geração inicial de TR e ETP.

A aplicação construída demonstra que o uso de agentes de inteligência artificial pode contribuir para organizar a coleta de informações, reduzir tarefas repetitivas e padronizar a estrutura inicial de documentos licitatórios. A integração com RAG também mostrou ser relevante, pois permite que o sistema consulte uma base documental própria antes de gerar suas respostas.

Ainda assim, o LaborAI deve ser entendido como uma ferramenta de apoio. A geração de documentos em processos públicos envolve responsabilidade técnica, jurídica e administrativa, o que torna indispensável a revisão humana. O sistema não elimina essa etapa, mas pode reduzir parte do esforço inicial de elaboração e melhorar a organização das informações.

No geral, o protótipo mostrou-se tecnicamente viável e coerente com os objetivos propostos. A solução desenvolvida reúne autenticação, persistência de sessões, base vetorial, agentes especializados, upload de arquivos, geração DOCX e interface web, formando uma base consistente para evoluções futuras.

REFERÊNCIAS

BURHAN, Peerzada. **LangGraph Tutorials - Part 1: Let us introduce LangGraph and build a simple graph**. *Generative AI*, 18 mar. 2025. Disponível em: https://generativeai.pub/langgraph-tutorials-part-1-89e6a5adace9. Acesso em: jan. 2026.

BANKS, Alex. **Learning React: Modern Patterns for Developing React Apps**. O'Reilly Media, 2020.

BRASIL. **Lei nº 14.133, de 1º de abril de 2021**. Estabelece normas gerais de licitação e contratação para as Administrações Públicas. Diário Oficial da União, Brasília, DF, 1 abr. 2021. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2019-2022/2021/lei/l14133.htm. Acesso em: jan. 2026.

BRASIL. Ministério da Economia. Secretaria de Gestão. **Instrução Normativa SEGES nº 58, de 8 de agosto de 2022**. Dispõe sobre a elaboração dos Estudos Técnicos Preliminares - ETP. Brasília, DF: Gov.br, 2022. Disponível em: https://www.gov.br/compras/pt-br/acesso-a-informacao/legislacao/instrucoes-normativas/instrucao-normativa-seges-no-58-de-8-de-agosto-de-2022. Acesso em: jan. 2026.

BRASIL. Ministério da Economia. Secretaria de Gestão. **Instrução Normativa SEGES/ME nº 81, de 25 de novembro de 2022**. Dispõe sobre a elaboração do Termo de Referência - TR. Brasília, DF: Gov.br, 2022. Disponível em: https://www.gov.br/compras/pt-br/acesso-a-informacao/legislacao/instrucoes-normativas/instrucao-normativa-seges-me-no-81-de-25-de-novembro-de-2022. Acesso em: jan. 2026.

BRASIL. Ministério da Gestão e da Inovação em Serviços Públicos. **Compras.gov.br**: Portal de Compras do Governo Federal. Disponível em: https://www.gov.br/compras/pt-br. Acesso em: jan. 2026.

BROWN, Tom B. et al. **Language Models are Few-Shot Learners**. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 33., 2020. **Proceedings [...]**. [S. l.]: NeurIPS, 2020. Disponível em: https://arxiv.org/abs/2005.14165. Acesso em: jan. 2026.

DAHL, Matthew et al. Large Legal Fictions: Profiling Legal Hallucinations in Large Language Models. **Journal of Legal Analysis**, v. 16, n. 1, p. 64–93, 2024. Disponível em: https:

//academic.oup.com/jla/article/16/1/64/7699227. Acesso em: jan. 2026.

GIL, Antônio Carlos. **Como elaborar projetos de pesquisa**. 6. ed. São Paulo: Atlas, 2017.

LANGCHAIN. **Agents**. Disponível em: https://docs.langchain.com/oss/javascript/langchain/agents. Acesso em: jan. 2026.

LANGCHAIN. **LangGraph Overview**. Disponível em: https://docs.langchain.com/oss/python/langgraph/overview. Acesso em: jan. 2026.

LEWIS, Patrick et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 33., 2020, Virtual. **Proceedings [...]**. [S. l.]: NeurIPS, 2020. p. 9459–9474. Disponível em: https://arxiv.org/abs/2005.11401. Acesso em: jan. 2026.

MAGESH, Varun et al. **Hallucination-Free? Assessing the Reliability of Leading AI Legal Research Tools**. arXiv, 2024. Disponível em: https://arxiv.org/abs/2405.20362. Acesso em: jan. 2026.

MAMMOTH. **Mammoth .docx to HTML converter**. Disponível em: https://github.com/mwilliamson/mammoth.js. Acesso em: jan. 2026.

MICROSOFT. **TypeScript Documentation**. Disponível em: https://www.typescriptlang.org/docs/. Acesso em: jan. 2026.

MICROSOFT. **RAG (Geração Aumentada de Recuperação) no Azure Databricks**. Microsoft Learn, 2026. Disponível em: https://learn.microsoft.com/pt-br/azure/databricks/generative-ai/retrieval-augmented-generation. Acesso em: fev. 2026.

MOZILLA. **PDF.js**. Disponível em: https://mozilla.github.io/pdf.js/. Acesso em: jan. 2026.

NESTJS. **Documentation**. Disponível em: https://docs.nestjs.com. Acesso em: jan. 2026.

NODE.JS. **Node.js Documentation**. Disponível em: https://nodejs.org/docs/latest/api/. Acesso em: jan. 2026.

OUYANG, Long et al. **Training language models to follow instructions with human feedback**. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 35., 2022. **Proceedings [...]**. [S. l.]: NeurIPS, 2022. Disponível em: https://arxiv.org/abs/

2203.02155. Acesso em: jan. 2026.

PGVECTOR. **pgvector: Open-source vector similarity search for Postgres**. Disponível em: https://github.com/pgvector/pgvector. Acesso em: jan. 2026.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. **PostgreSQL Documentation**. Disponível em: https://www.postgresql.org/docs/. Acesso em: jan. 2026.

RICHARDSON, Leonard; RUBY, Sam. **RESTful Web APIs**. O'Reilly Media, 2013.

RUSSELL, Stuart; NORVIG, Peter. **Artificial Intelligence: A Modern Approach**. 4. ed. Pearson, 2022.

SUPABASE. **Row Level Security**. Disponível em: https://supabase.com/docs/guides/database/postgres/row-level-security. Acesso em: jan. 2026.

SUPABASE. **pgvector: embeddings and vector similarity**. Disponível em: https://supabase.com/docs/guides/database/extensions/pgvector. Acesso em: jan. 2026.

SUPABASE. **Documentation**. Disponível em: https://supabase.com/docs. Acesso em: jan. 2026.

TAILWIND CSS. **Documentation**. Disponível em: https://tailwindcss.com/docs. Acesso em: jan. 2026.

TANENBAUM, Andrew S.; VAN STEEN, Maarten. **Distributed Systems: Principles and Paradigms**. 2. ed. Pearson, 2007.

TRIBUNAL DE CONTAS DA UNIÃO. **Estudo Técnico Preliminar (ETP)**. Licitações e Contratos. Brasília, DF: TCU, [s.d.]. Disponível em: https://licitacoesecontratos.tcu.gov.br/4-1-estudo-tecnico-preliminar-etp/. Acesso em: jan. 2026.

TRIBUNAL DE CONTAS DA UNIÃO. **Termo de Referência (TR)**. Licitações e Contratos. Brasília, DF: TCU, [s.d.]. Disponível em: https://licitacoesecontratos.tcu.gov.br/4-3-termo-de-referencia-tr/. Acesso em: jan. 2026.

TRIBUNAL DE CONTAS DO ESTADO DE ALAGOAS. **Estudo Técnico Preliminar (ETP) e Termo de Referência (TR)**. Maceió: TCE-AL, 2021. Disponível em: https://www.tceal.tc.br/view/documentos/doc28042021132920000000006089633033a68.pdf. Acesso em: jan. 2026.

VITE. **Getting Started**. Disponível em: https://vite.dev/guide/. Acesso em: jan. 2026.

W3C. **Cascading Style Sheets (CSS) Snapshot 2023**. Disponível em: https://www.w3.org/TR/css-2023/. Acesso em: jan. 2026.

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