

I2A2 - INSTITUTE OF APPLIED ARTIFICIAL INTELLIGENCE
CURSO AGENTES AUTÔNOMOS COM REDES GENERATIVAS

GRUPO SIA (SOLUÇÕES INTELIGENTES AUTÔNOMAS)

Ícaro Ribeiro

PROJETO FINAL

Brasil

2025

ÍCARO RIBEIRO

PROJETO FINAL

Relatório do Projeto Final desenvolvido
pelo Grupo SIA (Soluções Inteligentes
Autônomas).

Professor: Celso Azevedo

Brasil

2025

LISTA DE ILUSTRAÇÕES

FIGURAS

Figura 1 - Fluxo de Trabalho do Sistema de Gestão de NF-e com IA.....	34
Figura 2 - Fluxo de Descompactação de Arquivo ZIP.....	43
Figura 3 - Fluxo de Mapeamento de Arquivos CSV.....	43
Figura 4 - Fluxo de Inserção de Registros no Banco de Dados.....	45
Figura 5 - Fluxo de Análise de Dados (Plotagem de Dados e Bate-Papo).....	46
Figura 6 - Tela Inicial.....	47
Figura 7 - Tela Modelagem de Dados.....	47
Figura 8 - Tela Ingestão de NF-e.....	48
Figura 9 - Tela Análise de Dados.....	48
Figura 10 - Tela Bate-Papo.....	49
Figura 11 - Tela Sobre.....	49

TABELAS

Tabela 1 - Tabela de Metadados para o Estado da Conversação.....	22
Tabela 2 - Tabela invoices (Cabeçalho da NF-e).....	23
Tabela 3 - Tabela invoice_items (Itens da NF-e).....	25

SUMÁRIO

1. INTRODUÇÃO.....	6
2. DESCRIÇÃO DO TEMA ESCOLHIDO.....	10
2.1 Extração de Dados e Automação de Processos: O Pipeline de Ingestão.....	10
2.2 Ferramentas Gerenciais e Assistente Consultor Especializado: Análise e Interação.	11
3. PÚBLICO-ALVO.....	14
4. JUSTIFICATIVA DO TEMA.....	16
5. DETALHAMENTO DO PROJETO.....	19
5.1 Arquitetura da Solução.....	19
5.2 Agentes de IA: Um Ecossistema Colaborativo.....	20
5.2.1 Agente Supervisor: O Maestro do Fluxo.....	20
5.2.2 Agentes Executores: Especialistas em Ação.....	20
5.3 Mecanismo de Memória e Persistência.....	21
5.3.1 O Papel do AsyncPostgresSaver (Checkpointner).....	21
5.3.2 Componentes Armazenados na Memória.....	22
5.3.3 Tabelas de Metadados para o Estado da Conversação.....	22
5.4 Modelagem de Dados das NF-e e Itens.....	23
5.4.1 Tabela invoices (Cabeçalho da NF-e).....	23
5.4.2 Tabela invoice_items (Itens da NF-e).....	24
5.4.3 Gerenciamento de Migrações com Alembic.....	26
5.5 Ferramentas de IA: Estendendo as Capacidades do LLM.....	26
5.5.1 Ferramenta de Delegação (Handoff Tool).....	27
5.5.2 Ferramenta de Descompactação de Arquivo (UnzipZipFileTool).....	27
5.5.3 Ferramenta de Mapeamento de CSVs (MapCSVsToIngestionArgsTool).....	28
5.5.4 Ferramenta de Inserção de Registros (InsertRecordsIntoDatabaseTool).....	29
5.5.5 Kit de Ferramentas de Banco de Dados (AsyncSQLDatabaseToolkit).....	30
5.5.6 Ferramenta de Obtenção de Esquemas das Tabelas (GetDetailedTableSchemasTool).....	30
5.5.7 Ferramenta de Geração de Gráfico de Barras (GenerateBarPlotTool).....	31
5.5.8 Ferramenta de Geração de Gráfico de Distribuição (GenerateDistributionPlotTool).....	32
5.6 Fluxo de Trabalho da Solução.....	32
6. AMBIENTE DE CONFIGURAÇÃO E EXECUÇÃO.....	35
6.1 Arquitetura de Implantação e Tecnologia de Containerização.....	35
6.2 Pré-requisitos de Software.....	36
6.2.1 Docker Desktop.....	36
6.2.2 Utilitário Make (Opcional).....	36
6.3 Configuração de Variáveis de Ambiente.....	37
6.4 Procedimentos de Execução.....	38
6.4.1 Método 1: Execução com Docker (Ambiente de Produção/Distribuído).....	38

6.5 Verificação e Acesso à Aplicação.....	41
7. ELEMENTOS ADICIONAIS.....	41
7.1 Diagramas de Sequência do Fluxos de Trabalho.....	42
7.2 Telas das Funções do Sistema.....	46
8. RESULTADOS, DISCUSSÕES E CONCLUSÕES.....	49
8.1 Resultados Alcançados.....	50
8.2 Discussão dos Resultados e Aprendizados.....	51
8.3 Oportunidades Futuras e Evolução do Projeto.....	52
8.4 Conclusão e Agradecimentos.....	53
9. REPOSITÓRIO DO PROJETO NO GITHUB.....	55
10. REFERÊNCIAS.....	56

1. INTRODUÇÃO

A transformação digital tem redefinido o cenário corporativo global, impondo às organizações a necessidade de processar um volume de dados sem precedentes. No Brasil, o ecossistema fiscal, com sua complexidade inerente e a obrigatoriedade de documentos eletrônicos como a Nota Fiscal Eletrônica (NF-e), representa um desafio significativo e, ao mesmo tempo, uma vasta oportunidade para a inovação tecnológica. A gestão manual desses documentos é um processo intensivo em mão de obra, propenso a erros de escrituração, demorado e que consome recursos valiosos que poderiam ser alocados em atividades estratégicas. Inconsistências em códigos como CFOP (Código Fiscal de Operações e Prestações) ou NCM (Nomenclatura Comum do Mercosul), erros de cálculo de impostos e a dificuldade em consolidar informações para a tomada de decisão gerencial são problemas recorrentes que afetam a eficiência operacional e a conformidade fiscal das empresas.

Nesse contexto, o presente trabalho de conclusão de curso propõe e detalha o desenvolvimento de uma plataforma inteligente denominada **Sistema de Gestão de NF-e com IA**. A solução foi concebida para endereçar diretamente os desafios mencionados, automatizando o ciclo de vida dos dados fiscais, desde a sua extração até a geração de insights acionáveis. Utilizando como fonte de dados os arquivos de NF-e em formato CSV, disponibilizados publicamente pelo **Portal da Transparência (BRASIL. Controladoria-Geral da União, [s.d.]**), o sistema oferece um ambiente unificado para ingestão, armazenamento, análise e interação com as informações fiscais. O projeto se destaca por empregar uma arquitetura moderna baseada em um fluxo de trabalho orquestrado por múltiplos agentes de Inteligência Artificial, cada um especializado em uma etapa do processo, garantindo modularidade, eficiência e escalabilidade.

A solução desenvolvida se alinha diretamente a diversos temas centrais propostos para o projeto, demonstrando uma abordagem multifacetada para a resolução do problema. Os principais temas abordados são:

- **Extração de Dados e Automação de Processos Fiscais/Contábeis:** O sistema implementa um robusto processo de extração, transformação e carga (ETL). A automação inicia-se com a capacidade de o usuário submeter um arquivo compactado contendo múltiplos documentos fiscais. Um agente de IA especializado é acionado para realizar a descompactação e identificar os arquivos CSV. Em seguida, outro agente assume a tarefa de mapear inteligentemente as colunas dos arquivos de origem para a estrutura de dados relacional do sistema, preparando-os para a persistência. Por fim, um terceiro agente gerencia a inserção dos registros em um banco de dados SQL, garantindo a integridade e a consistência dos dados. Este fluxo automatizado elimina a necessidade de entrada manual de dados, mitigando drasticamente o risco de erros humanos e otimizando o tempo necessário para consolidar as informações fiscais.
- **Classificação, Categorização e Ferramentas Gerenciais:** Uma vez que os dados estão estruturados no banco, o sistema revela seu maior potencial como uma poderosa ferramenta gerencial. A plataforma oferece uma seção de análise de dados com dashboards interativos e pré-configurados. Por meio de uma interface intuitiva, o usuário pode classificar e visualizar informações consolidadas a partir de diversas métricas, como faturamento total, valor médio por NF-e, contagem de notas, volume de itens e ranking de produtos mais relevantes. As análises podem ser categorizadas e agrupadas por diferentes dimensões, como a Unidade Federativa (UF) do emitente ou a descrição do produto, permitindo uma visão estratégica do negócio. A apresentação dos dados é enriquecida com recursos visuais sofisticados, incluindo mapas coropléticos para análise geográfica, gráficos de barras para comparações temporais e tabelas interativas que facilitam a exploração e a compreensão dos resultados.
- **Assistente Consultor Especializado:** O elemento mais inovador do projeto é, sem dúvida, a implementação de um assistente de IA conversacional. Através de uma interface de bate-papo, o usuário pode interagir com o sistema utilizando linguagem natural para realizar consultas complexas sobre os dados fiscais. O sistema é capaz de interpretar a pergunta do usuário, acionar o

agente de análise de dados para consultar o banco de dados e, mais notavelmente, gerar não apenas respostas textuais, mas também visualizações gráficas dinâmicas e personalizadas sob demanda. Essa funcionalidade transforma a plataforma de uma ferramenta de visualização passiva para um consultor ativo, democratizando o acesso à análise de dados e permitindo que usuários sem conhecimento técnico em bancos de dados ou Business Intelligence possam extrair insights valiosos de forma autônoma.

O objetivo principal deste projeto foi, portanto, projetar, desenvolver e validar um protótipo funcional de uma plataforma que utiliza agentes de Inteligência Artificial para automatizar a ingestão e potencializar a análise de documentos fiscais. A arquitetura, construída com Python, Streamlit, LangGraph e PostgreSQL, foi pensada para ser uma solução completa, abordando desde a otimização de tarefas operacionais até o suporte à decisão estratégica, demonstrando o impacto positivo da IA na modernização da gestão fiscal e contábil (LANGCHAIN, [s.d.]; POSTGRESQL GLOBAL DEVELOPMENT GROUP, [s.d.]; PYTHON SOFTWARE FOUNDATION, [s.d.]; STREAMLIT INC., [s.d.]).

Para detalhar o desenvolvimento e os resultados alcançados, este relatório está estruturado da seguinte forma: o Capítulo 2 trará a Descrição do Tema Escolhido, detalhando o escopo do que foi realizado. O Capítulo 3 definirá a Identificação do Público-Alvo, descrevendo quem utilizará a solução. O Capítulo 4 apresentará a Justificativa do Tema, explicando a importância do trabalho. O Capítulo 5 fará o Desenvolvimento da Solução, revelando o que foi desenvolvido, apresentando as funções, a operação da solução e seus aspectos técnicos. O Capítulo 6 abordará a Configuração e Execução do Sistema, detalhando os requisitos técnicos e as instruções de como instalar e operar a solução desenvolvida. O Capítulo 7 demonstrará os Elementos Adicionais, com diagramas de sequência dos fluxos de trabalho desempenhados pelos agentes de IA e telas das funções do sistema. O Capítulo 8 apresentará os Resultados, as Discussões e as Conclusões, juntamente com sugestões para trabalhos futuros. O Capítulo 9 fornecerá o Link para o Repositório do Projeto no GitHub que contém o projeto. Finalmente, o Capítulo 10 listará as Referências utilizadas.

2. DESCRIÇÃO DO TEMA ESCOLHIDO

O presente trabalho se aprofunda na interseção de três áreas fundamentais para a modernização da gestão fiscal e contábil: **1) Extração de Dados e Automação de Processos Fiscais/Contábeis**, **2) Classificação, Categorização e Ferramentas Gerenciais**, e **3) Assistente Consultor Especializado**. A solução desenvolvida não aborda esses temas de forma isolada, mas os integra em um ecossistema coeso e sinérgico, cujo núcleo é uma arquitetura avançada de múltiplos agentes de Inteligência Artificial (IA) que colaboram para atingir um objetivo comum. O escopo do projeto, portanto, consiste na criação de um sistema de software que automatiza o fluxo de ponta a ponta, desde o recebimento de dados brutos até a entrega de análises estratégicas interativas.

A estrutura central do sistema é um **fluxo de trabalho (workflow) inteligente**, implementado como um grafo de estados (StateGraph), que orquestra a interação entre um agente supervisor e um conjunto de agentes especialistas. Essa abordagem, inspirada em sistemas multiagentes, permite que tarefas complexas sejam decompostas em sub tarefas menores e delegadas ao agente com a habilidade mais apropriada para executá-la.

A seguir, detalhamos como cada um dos temas propostos foi abordado e implementado no escopo desta solução.

2.1 Extração de Dados e Automação de Processos: O Pipeline de Ingestão

A base para qualquer análise de dados confiável é um processo de ingestão robusto e automatizado. Neste projeto, a automação do pipeline de dados fiscais é a primeira e mais crítica função do sistema, materializando o tema de **Extração e Automação de Processos**. A plataforma foi projetada para minimizar a intervenção humana e eliminar os erros associados à manipulação manual de arquivos.

O processo é iniciado quando o usuário submete um arquivo compactado contendo os dados das Notas Fiscais Eletrônicas. Neste ponto, o **Agente**

Supervisor, atuando como o maestro do fluxo de trabalho, recebe a requisição inicial e a direciona para o primeiro especialista:

1. **Agente de Descompactação:** Este agente é equipado com uma ferramenta específica para lidar com arquivos ZIP. Sua única responsabilidade é extrair o conteúdo do arquivo submetido, identificando e preparando os arquivos CSV internos para a próxima etapa do processo. Uma vez concluída sua tarefa, ele reporta o sucesso de volta ao fluxo.
2. **Agente de Mapeamento de Dados:** Após a extração, o Supervisor delega a responsabilidade para o agente de mapeamento. Este especialista possui a capacidade de analisar a estrutura dos arquivos CSV, identificar as colunas relevantes (como dados do emitente, destinatário, itens, valores e códigos fiscais) e traduzir (mapear) essas informações para um formato estruturado, compatível com o esquema do banco de dados relacional do sistema. O resultado de sua operação é um conjunto de “argumentos de ingestão” pronto para ser persistido.
3. **Agente de Inserção de Registros:** Com os dados devidamente mapeados e validados, o Supervisor aciona o terceiro e último agente do pipeline de ingestão. Este agente é responsável por executar a transação final: inserir os registros de forma massiva e eficiente nas tabelas do banco de dados PostgreSQL. Ele utiliza ferramentas de acesso direto ao banco, garantindo que a operação seja atômica e segura.

Esse fluxo sequencial e orquestrado de agentes autônomos representa a concretização da automação de processos fiscais. A transição de uma tarefa manual e suscetível a falhas para um pipeline automatizado, resiliente e monitorado é a principal contribuição do sistema neste tema.

2.2 Ferramentas Gerenciais e Assistente Consultor Especializado: Análise e Interação

Com os dados fiscais devidamente extraídos, limpos e armazenados, o sistema transita do domínio operacional para o estratégico, abordando os temas de

Ferramentas Gerenciais e Assistente Consultor Especializado. A inteligência da plataforma não reside apenas na automação, mas na sua capacidade de transformar dados brutos em conhecimento acionável.

Isso é alcançado por meio de um quarto especialista, o **Agente de Análise de Dados**. Este é o agente mais versátil do ecossistema, equipado com um arsenal de ferramentas para interagir com o banco de dados e gerar visualizações. Ele é o motor por trás tanto dos dashboards da aplicação quanto do assistente conversacional:

- **Para as Ferramentas Gerenciais (Dashboards):** Quando um usuário seleciona uma análise na interface gráfica (por exemplo, “Faturamento Total por UF”), uma requisição estruturada é enviada ao Agente Supervisor, que a delega ao analista. O Agente de Análise de Dados então executa consultas SQL complexas para agregar as informações, calcular as métricas necessárias (somas, médias, contagens) e retornar os dados em um formato JSON que a interface utiliza para renderizar os mapas, gráficos e tabelas.
- **Para o Assistente Consultor Especializado (Bate-Papo):** Esta é a funcionalidade mais inovadora. Quando um usuário digita uma pergunta em linguagem natural na interface de chat (ex: “qual o total de vendas para o estado de São Paulo em 2025?”), o Agente Supervisor novamente interpreta a intenção e passa a tarefa para o analista de dados. Este, por sua vez, não apenas executa a consulta correspondente, mas também é capaz de tomar decisões sobre a melhor forma de apresentar a resposta. Ele pode utilizar ferramentas para gerar dinamicamente especificações de gráficos (como gráficos de barra ou de distribuição) e retorná-las junto com a resposta textual.

O Agente Supervisor desempenha um papel crucial como o ponto central de roteamento e decisão, garantindo que a consulta do usuário, seja ela uma simples requisição da interface ou uma pergunta complexa no chat, seja direcionada ao especialista correto. A capacidade do sistema de manter o contexto das conversas, utilizando um mecanismo de checkpointing no banco de dados, permite interações

mais fluidas e complexas, solidificando seu papel como um verdadeiro assistente consultor. A sinergia entre o supervisor e o analista de dados transforma a aplicação de um simples visualizador de dados em uma plataforma de diálogo analítico.

3. PÚBLICO-ALVO

A solução **Sistema de Gestão de NF-e com IA** foi concebida para atender a uma ampla gama de profissionais e organizações que lidam diretamente com a complexidade do sistema tributário brasileiro. O projeto visa entregar valor tanto para o usuário operacional, focado na execução de tarefas rotineiras, quanto para o usuário estratégico, que necessita de informações consolidadas para a tomada de decisão. A definição do público-alvo foi um pilar central na concepção da arquitetura e da interface do sistema, buscando equilibrar poder de processamento com usabilidade. Segundo o Conselho Federal de Contabilidade ([s.d.]), a evolução das organizações contábeis no Brasil tem mostrado um crescimento significativo. Em 01/10/2025, o número total de organizações contábeis era de 100.676, que atendem empresas de diversos portes e segmentos. Este dado estatístico reforça a vastidão do mercado e a necessidade de ferramentas que tragam eficiência e inteligência para este setor.

Neste contexto, podemos segmentar o público-alvo da seguinte forma:

1. **Público Primário (Usuários Operacionais):** Este grupo é formado pelos profissionais que estão na linha de frente da gestão fiscal e contábil. Inclui **contadores, analistas fiscais e contábeis, e assistentes de departamento fiscal**. Para estes usuários, as principais dores são o tempo gasto em rotinas operacionais, repetitivas e de alto volume, como a conferência manual e o lançamento de notas fiscais. O risco de erros de escrituração que podem levar a multas e passivos fiscais é uma preocupação constante. A solução atende diretamente a este público ao automatizar todo o pipeline de ingestão de dados - desde a descompactação dos arquivos até a inserção no banco de dados -, liberando esses profissionais para se concentrarem em atividades de maior valor agregado, como a auditoria e a análise crítica das informações.
2. **Público Secundário (Usuários Estratégicos):** Além dos profissionais da linha de frente, o sistema foi projetado para empoderar **gestores financeiros (CFOs), diretores, consultores tributários e donos de pequenas e médias empresas**. Este público necessita de informações consolidadas e estratégicas

para tomar decisões mais informadas sobre vendas, compras, custos e planejamento tributário. Para eles, o valor da plataforma reside nas **ferramentas gerenciais**, como os dashboards interativos que permitem visualizar o faturamento por região, a performance de produtos e outras métricas chave com poucos cliques. Adicionalmente, **o assistente consultor especializado** (bate-papo) é um diferencial crucial, pois permite que estes gestores, muitas vezes sem conhecimento técnico profundo em linguagens de consulta de banco de dados, possam fazer perguntas de negócio em linguagem natural e obter respostas visuais e textuais de forma imediata.

Considerando o perfil das organizações, a solução é particularmente adequada para **escritórios de contabilidade e pequenas e médias empresas (PMEs)**. Esses dois grupos frequentemente carecem de sistemas ERP (Enterprise Resource Planning) de grande porte com módulos fiscais avançados, seja pelo alto custo de implementação ou pela complexidade de uso. A plataforma desenvolvida neste trabalho oferece a essas organizações um poder de análise de dados que antes era acessível apenas a grandes corporações, democratizando o acesso à inteligência fiscal e contábil.

Em suma, o público-alvo abrange desde o analista que realiza a ingestão dos dados no dia a dia até o diretor que consulta o faturamento por região no dashboard para planejar a expansão do negócio. A combinação de uma interface intuitiva com um poderoso motor de IA analítica torna a solução versátil e valiosa para diferentes perfis de usuários dentro do ecossistema fiscal brasileiro.

4. JUSTIFICATIVA DO TEMA

A relevância de um projeto de tecnologia é medida por sua capacidade de resolver problemas reais, gerar eficiência e criar valor em um contexto específico. O desenvolvimento do **Sistema de Gestão de NF-e com IA** se justifica plenamente ao endereçar uma das áreas mais críticas e onerosas para as empresas brasileiras: a gestão fiscal e tributária. A complexidade do sistema tributário nacional, historicamente marcada por uma profusão de regras, alíquotas e obrigações acessórias, representa um componente significativo do chamado “Custo Brasil”, impactando diretamente a competitividade e a sustentabilidade das organizações.

Neste cenário já desafiador, o Brasil se encontra em um ponto de inflexão histórica. A aprovação da Emenda Constitucional 132/23, que institui a reforma tributária e entrará em vigor de forma gradativa até 2032, desencadeou uma “corrida acelerada” das empresas brasileiras por adaptação e consultoria especializada. Essa antecipação às mudanças tem revelado um problema crônico no cenário fiscal do país: grande parte das organizações arca com despesas tributárias maiores do que o necessário. De fato, um levantamento realizado pelo Instituto Brasileiro de Planejamento e Tributação (IBPT) indica que **95% das organizações brasileiras pagam mais impostos do que deveriam** (INSTITUTO BRASILEIRO DE PLANEJAMENTO E TRIBUTAÇÃO, [s.d.]). Esse cenário impulsiona a busca por soluções que otimizem a gestão fiscal antes que as transformações profundas da reforma se consolidem.

A estatística do IBPT é alarmante e serve como a principal justificativa para este trabalho. Ela evidencia que a complexidade tributária não é apenas um fardo operacional, mas uma fonte direta de prejuízo financeiro para a esmagadora maioria das empresas. O pagamento indevido de impostos ocorre por diversas razões que a presente solução visa mitigar:

- **Erros na escrituração manual:** A digitação e a classificação manual de notas fiscais são processos inerentemente suscetíveis a falhas humanas, que podem levar a cálculos incorretos e, consequentemente, ao pagamento de tributos a maior.

- **Falta de visibilidade e análise estratégica:** Sem ferramentas adequadas, os gestores não conseguem ter uma visão clara e consolidada de suas operações fiscais. A dificuldade em analisar grandes volumes de dados de NF-e impede a identificação de padrões, a otimização de rotas logísticas, o planejamento de créditos fiscais e a detecção de oportunidades de economia tributária.
- **Dificuldade de acesso à informação:** A extração de insights de dados fiscais tradicionalmente exige conhecimento especializado em bancos de dados e ferramentas de Business Intelligence. Essa barreira técnica deixa muitos gestores dependentes de relatórios estáticos e incapazes de explorar os dados de forma dinâmica para responder a novas perguntas de negócio.

O sistema proposto justifica-se, portanto, por atacar diretamente essas três frentes. A **automação do processo de ingestão de dados** elimina a necessidade de entrada manual, reduzindo drasticamente a incidência de erros operacionais. As **ferramentas gerenciais** com dashboards interativos proporcionam a visibilidade estratégica que falta a muitas organizações, transformando dados fiscais brutos em indicadores de performance claros e acionáveis. Por fim, **o assistente consultor especializado**, baseado em um agente de IA conversacional, democratiza o acesso à análise de dados, permitindo que qualquer gestor, independentemente de seu conhecimento técnico, possa “conversar” com seus dados e obter respostas para questões críticas de negócio.

A escolha de uma arquitetura baseada em múltiplos agentes de IA não é meramente uma decisão técnica, mas uma resposta estratégica à natureza multifacetada do problema. Em vez de uma solução monolítica, o modelo de agentes especialistas (descompactação, mapeamento, inserção, análise) orquestrados por um supervisor permite maior modularidade, robustez e escalabilidade. Essa abordagem inovadora justifica-se por sua capacidade de lidar com a complexidade do domínio fiscal de forma mais eficiente e flexível do que as abordagens tradicionais.

Em suma, a importância deste trabalho reside em sua contribuição prática para um problema urgente e de alto impacto para o conjunto de empresas brasileiras. Em

um momento de transição e incerteza com a iminente reforma tributária, oferecer uma ferramenta que promove a eficiência operacional, a conformidade fiscal e, principalmente, a inteligência estratégica, não é apenas relevante, é essencial. A solução desenvolvida representa um passo concreto para ajudar as empresas, em especial as pequenas e médias, a navegar neste novo cenário com maior segurança e competitividade.

5. DETALHAMENTO DO PROJETO

Este capítulo apresenta os detalhes técnicos e a arquitetura da solução desenvolvida. O objetivo é elucidar o funcionamento interno da plataforma, desde a sua estrutura de alto nível até os componentes específicos de Inteligência Artificial que a tornam funcional e inovadora. Serão abordados a arquitetura geral, o design do sistema multiagentes, as ferramentas que estendem as capacidades dos modelos de linguagem, os mecanismos de persistência de estado, a modelagem de dados de NF-e e seus itens no banco de dados e o fluxo de trabalho completo da solução.

5.1 Arquitetura da Solução

A plataforma foi projetada com base em uma arquitetura de três camadas principais, garantindo a separação de responsabilidades, modularidade e escalabilidade.

1. **Camada de Apresentação (Front-end):** Desenvolvida com o framework **Streamlit**, esta camada é responsável por toda a interação com o usuário. Ela provê a interface gráfica web, incluindo as páginas de ingestão de arquivos, os dashboards de análise de dados e o assistente de bate-papo. Sua função é capturar as requisições do usuário e exibir os resultados processados de forma intuitiva e visualmente agradável.
2. **Camada de Lógica e Inteligência (Back-end/AI Core):** O coração do sistema reside nesta camada. Construída em **Python** e utilizando o framework **LangGraph**, ela orquestra o fluxo de trabalho de múltiplos agentes de IA. É aqui que as requisições da camada de apresentação são recebidas, interpretadas e delegadas aos agentes apropriados para execução. Esta camada encapsula toda a lógica de negócio, a coordenação dos agentes e a interação com a camada de dados.
3. **Camada de Persistência (Base de Dados):** Utilizando o sistema de gerenciamento de banco de dados **PostgreSQL**, esta camada é responsável pelo armazenamento e recuperação de todos os dados da aplicação. Isso inclui não apenas os dados extraídos das NF-e, mas também os metadados

necessários para a persistência do estado das conversas e dos fluxos de trabalho dos agentes de IA.

5.2 Agentes de IA: Um Ecossistema Colaborativo

A abordagem central do projeto é a utilização de um sistema multiagentes. Em vez de um único modelo de IA monolítico, a solução emprega um conjunto de agentes autônomos e especializados, cada um com um objetivo claro e um conjunto de ferramentas específicas. Essa colaboração é orquestrada por um agente central, o Supervisor.

5.2.1 Agente Supervisor: O Maestro do Fluxo

O **Agente Supervisor** é o ponto de entrada para todas as requisições do sistema. Sua função não é executar tarefas diretamente, mas sim atuar como um roteador inteligente. Com base na análise da solicitação do usuário, ele determina qual agente especialista é o mais qualificado para a tarefa e delega a responsabilidade, fornecendo todo o contexto necessário para a execução. Conforme o código de seu prompt instrui, ele é programado para:

- Analisar a requisição do usuário.
- Delegar tarefas uma de cada vez, utilizando a ferramenta de handoff apropriada para cada especialista (ex: `delegate_to_unzip_file_agent_tool` para o agente de descompactação).
- Repassar quaisquer instruções de formatação específicas contidas na requisição original.

Após um agente executor completar sua tarefa, ele reporta o resultado de volta ao fluxo, e o Supervisor pode então decidir o próximo passo, seja delegar uma nova tarefa a outro agente ou formular a resposta final para o usuário.

5.2.2 Agentes Executores: Especialistas em Ação

Sob a coordenação do Supervisor, operam quatro agentes executores, cada um com uma persona e um conjunto de habilidades bem definidos:

- **Agente de Descompactação** (`unzip_file_agent`): Um especialista focado em uma única tarefa: descompactar arquivos no formato ZIP. Ele é acionado no início do fluxo de ingestão de dados.
- **Agente de Mapeamento de CSV** (`csv_mapping_agent`): Sua responsabilidade é receber os caminhos dos arquivos CSV extraídos e utilizar suas ferramentas para mapear os dados contidos neles para um formato estruturado que possa ser inserido no banco de dados.
- **Agente de Inserção de Registros** (`insert_records_agent`): Este agente recebe os dados já mapeados e executa a tarefa crítica de inseri-los nas tabelas do banco de dados PostgreSQL.
- **Agente de Análise de Dados** (`data_analysis_agent`): O agente mais versátil, responsável por responder a todas as consultas analíticas. Seu prompt o instrui a, primeiramente, inspecionar o esquema do banco de dados para entender a estrutura das tabelas e, em seguida, utilizar ferramentas de consulta SQL e geração de gráficos para responder às perguntas do usuário, seja através dos dashboards ou do bate-papo.

Essa especialização garante que cada etapa do processo seja tratada pela lógica mais adequada, resultando em um sistema mais robusto e fácil de manter.

5.3 Mecanismo de Memória e Persistência

Fluxos de trabalho complexos e interações de bate-papo são inerentemente “stateful” (dependentes de estado), o que significa que eles precisam se lembrar do que aconteceu anteriormente. Para gerenciar isso, o sistema implementa um robusto mecanismo de memória e persistência.

5.3.1 O Papel do AsyncPostgresSaver (Checkpointing)

O componente central deste mecanismo é o AsyncPostgresSaver, uma funcionalidade do LangGraph que permite salvar o estado completo do fluxo de

trabalho em um banco de dados PostgreSQL após cada etapa de execução. Este processo, conhecido como **checkpointing**, oferece duas vantagens críticas:

- 1. **Persistência de Longo Prazo:** Permite que conversas e processos que se estendem por múltiplas interações (como no bate-papo) mantenham seu contexto. O sistema pode ser interrompido e retomado exatamente de onde parou.
- 2. **Resiliência a Falhas:** Se ocorrer um erro durante a execução, o estado anterior fica salvo, permitindo a depuração e a recuperação do processo sem perda total de progresso.

5.3.2 Componentes Armazenados na Memória

O “estado” que é salvo a cada passo é um objeto que contém todas as informações relevantes para o fluxo, incluindo: o histórico completo de mensagens trocadas entre os agentes, quaisquer dados intermediários gerados (como a lista de argumentos de ingestão após o mapeamento e a especificação de um gráfico a ser renderizado).

5.3.3 Tabelas de Metadados para o Estado da Conversação

O AsyncPostgresSaver gerencia automaticamente um conjunto de tabelas de metadados no banco de dados para armazenar esses estados.

Tabela 1 - Tabela de Metadados para o Estado da Conversação

Nome da Tabela	Descrição (Comentário)
checkpoints	Armazena os instantâneos de estado (snapshots) das threads de conversação. Cada registro contém o estado interno completo do agente de IA em um determinado ponto de execução.
checkpoint_blobs	Usada para armazenar dados binários ou grandes estruturas de dados associadas aos checkpoints, que são separadas da tabela principal para otimizar o desempenho.
checkpoint_writes	Tabela de rastreamento usada para auxiliar na consistência transacional e garantir a atomicidade durante a escrita de novos checkpoints.
checkpoint_migrations	Tabela de controle de versão que registra quais alterações de esquema foram aplicadas às tabelas de checkpoint.

5.4 Modelagem de Dados das NF-e e Itens

A fundação de um sistema analítico robusto reside em sua modelagem de dados. A eficiência das consultas, a performance das visualizações e a própria capacidade dos agentes de IA de extraírem insights dependem de um esquema de banco de dados bem-planejado. Para este projeto, a estrutura das tabelas foi projetada com base em uma análise criteriosa dos campos disponíveis em uma amostra de arquivos de Notas Fiscais Eletrônicas, extraídos de pacotes .zip obtidos do Portal da Transparência.

A abordagem adotada foi a de um **esquema relacional com desnormalização estratégica**, visando otimizar a performance de leitura para as consultas analíticas, que são o foco principal do sistema. Para garantir a consistência, a rastreabilidade e a automação na evolução do esquema do banco de dados, foi utilizada a ferramenta de migração **Alembic** em conjunto com o **SQLAlchemy** (ALEMBIC, c2024; SQLALCHEMY, c2024). Cada tabela e suas alterações foram definidas em arquivos de migração Python, permitindo que a estrutura do banco de dados seja versionada e recriada de forma programática e confiável em qualquer ambiente.

5.4.1 Tabela invoices (Cabeçalho da NF-e)

Esta tabela é o repositório central para as informações de cabeçalho de cada Nota Fiscal Eletrônica. Ela foi projetada para armazenar os dados gerais que identificam uma transação, como as informações do emitente e do destinatário, datas, valores totais e códigos operacionais. A chave de acesso da NF-e (access_key), um identificador único de 44 dígitos, atua como a principal chave de negócio, garantindo a unicidade de cada documento fiscal e servindo como elo de ligação para os seus respectivos itens. A estrutura detalhada desta tabela, com todas as suas colunas, tipos de dados e comentários - que são utilizados pelo Agente de Análise de Dados para interpretar o esquema -, é apresentada a seguir.

Tabela 2 - Tabela invoices (Cabeçalho da NF-e)

Coluna	Tipo de Dado (PostgreSQL)	Descrição (Comentário)
id	UUID (PK)	Identificador UUID único

access_key	VARCHAR(44) (PK/Unique)	Chave de acesso única (CHAVE DE ACESSO)
model	VARCHAR(51)	Tipo de nota fiscal (MODELO)
series	INTEGER	Série da nota fiscal (SÉRIE)
number	INTEGER	Número da nota fiscal (NÚMERO)
operation_nature	VARCHAR(255)	Natureza da operação (NATUREZA DA OPERAÇÃO)
issue_date	DATETIME	Data de emissão (DATA EMISSÃO)
latest_event	VARCHAR(255)	Evento mais recente (EVENTO MAIS RECENTE)
latest_event_datetime	DATETIME	Data/hora do evento mais recente
emitter_cnpj_cpf	VARCHAR(14)	CPF/CNPJ do emitente
emitter_corporate_name	VARCHAR(255)	Razão social do emitente
emitter_state_registration	VARCHAR(20)	Inscrição Estadual do emitente
emitter_uf	VARCHAR(2)	UF do emitente
emitter_municipality	VARCHAR(100)	Município do emitente
recipient_cnpj	VARCHAR(14)	CNPJ do destinatário
recipient_name	VARCHAR(255)	Nome do destinatário
recipient_uf	VARCHAR(2)	UF do destinatário
recipient_ie_indicator	VARCHAR(50)	Indicador IE do destinatário
operation_destination	VARCHAR(50)	Destino da operação
final_consumer	VARCHAR(50)	Indicador de consumidor final
buyer_presence	VARCHAR(50)	Indicador de presença do comprador
total_invoice_value	NUMERIC(15, 2)	Valor total da nota fiscal
created_at	DATETIME (TZ)	Timestamp de criação do registro
updated_at	DATETIME (TZ)	Timestamp da última atualização do registro

5.4.2 Tabela invoice_items (Itens da NF-e)

Para otimizar a performance de consultas que envolvem a agregação de dados de itens - o caso de uso mais comum na análise fiscal -, a tabela invoice_items foi intencionalmente desnormalizada. Isso significa que, além dos detalhes específicos de cada produto ou serviço da nota (como descrição, NCM, CFOP, quantidade e

valor unitário), a tabela também armazena uma cópia de campos importantes do cabeçalho da nota fiscal (como dados do emitente/destinatário e data de emissão).

Esta abordagem de design reduz significativamente a necessidade de operações de junção (JOIN) com a tabela invoices durante as consultas analíticas, resultando em um ganho expressivo de performance na geração dos dashboards e nas respostas do assistente de IA. Cada linha nesta tabela representa um item único de uma nota fiscal, ligada à tabela invoices através da chave estrangeira access_key.

Tabela 3 - Tabela invoice_items (Itens da NF-e)

Coluna	Tipo de Dado (PostgreSQL)	Descrição (Comentário)
id	UUID (PK)	Identificador UUID único do item
access_key	VARCHAR(44) (FK)	Chave de Acesso da Fatura
model	VARCHAR(51)	Modelo da NF-e
series	INTEGER	Série da NF-e
number	INTEGER	Número da NF-e
operation_nature	VARCHAR(255)	Natureza da Operação
issue_date	DATETIME	Data de Emissão
emitter_cnpj_cpf	VARCHAR(14)	CPF/CNPJ do Emitente
emitter_corporate_name	VARCHAR(255)	Razão Social do Emitente
emitter_state_registration	VARCHAR(20)	Inscrição Estadual do Emitente
emitter_uf	VARCHAR(2)	UF do Emitente
emitter_municipality	VARCHAR(100)	Município do Emitente
recipient_cnpj	VARCHAR(14)	CNPJ do Destinatário
recipient_name	VARCHAR(255)	Nome do Destinatário
recipient_uf	VARCHAR(2)	UF do Destinatário
recipient_ie_indicator	VARCHAR(50)	Indicador IE do Destinatário
operation_destination	VARCHAR(50)	Destino da Operação
final_consumer	VARCHAR(50)	Consumidor Final
buyer_presence	VARCHAR(50)	Presença do Comprador
product_number	VARCHAR(20)	Número do Produto
product_service_description	VARCHAR(255)	Descrição do Produto/Serviço
ncm_sh_code	VARCHAR(8)	Código NCM/SH
ncm_sh_product_type	VARCHAR(255)	Tipo de Produto NCM/SH

cfop	VARCHAR(4)	Código CFOP
quantity	NUMERIC(15, 4)	Quantidade
unit	VARCHAR(10)	Unidade de Medida
unit_value	NUMERIC(21, 10)	Valor Unitário
total_value	NUMERIC(15, 2)	Valor Total do Item
created_at	DATETIME (TZ)	Timestamp de criação do registro
updated_at	DATETIME (TZ)	Timestamp da última atualização do registro

5.4.3 Gerenciamento de Migrações com Alembic

Para garantir a consistência e a rastreabilidade da estrutura do banco de dados ao longo do desenvolvimento, foi utilizada a ferramenta **Alembic**. Sendo o padrão de fato para migrações com SQLAlchemy, o Alembic permite tratar o esquema do banco de dados como código, uma prática conhecida como Database as Code.

Cada alteração na estrutura das tabelas, como a criação da invoices e, subsequentemente, da invoice_items, foi encapsulada em seu próprio arquivo de migração Python, como os revisions 859145bdd14c e 1657037f60b6. Esses arquivos contêm as instruções para aplicar (upgrade) e reverter (downgrade) as alterações de forma segura.

O processo de atualização do banco de dados para a versão mais recente do esquema é automatizado por meio de um script que executa o comando alembic upgrade head. Isso garante que, em qualquer ambiente de desenvolvimento ou produção, a estrutura do banco de dados possa ser criada ou atualizada para o estado desejado com a execução de um único comando, eliminando inconsistências e facilitando a manutenção do sistema.

5.5 Ferramentas de IA: Estendendo as Capacidades do LLM

Um Modelo de Linguagem Grande (LLM), por si só, é limitado a processar e gerar texto. Para interagir com o mundo exterior - como ler arquivos, consultar bancos de dados ou executar código -, ele precisa de **Ferramentas**. No contexto

deste projeto, as ferramentas são funções Python que os agentes de IA podem invocar para realizar ações específicas.

5.5.1 Ferramenta de Delegação (Handoff Tool)

A comunicação e a delegação entre o Supervisor e os agentes executores são viabilizadas por uma ferramenta crucial: a **Ferramenta de Delegação** ou **Handoff Tool**. O código desta ferramenta revela um design inteligente:

- **Criação Dinâmica:** Uma instância desta ferramenta é criada para cada agente executor (ex: `delegate_to_data_analysis_agent_tool`). Isso permite ao Supervisor ter um "verbo" de ação claro para cada especialista.
- **Encapsulamento da Tarefa:** Quando o Supervisor decide delegar uma tarefa, ele invoca a ferramenta correspondente, passando como argumento a `task_description`, que é uma descrição em linguagem natural de tudo o que o próximo agente precisa fazer.
- **Mecanismo de Roteamento:** A ferramenta não executa a tarefa em si. Em vez disso, seu método de execução (`_arun`) retorna um "artefato" de dados contendo duas informações: o nome do agente de destino e a descrição da tarefa. O framework LangGraph utiliza este artefato para rotear o estado do fluxo de trabalho para o nó do agente correto no grafo, efetivamente "entregando" a tarefa para o especialista designado.

Esta ferramenta é o mecanismo fundamental que permite a orquestração e a colaboração entre os diferentes componentes de IA do sistema.

5.5.2 Ferramenta de Descompactação de Arquivo (UnzipZipFileTool)

Esta ferramenta é a porta de entrada para os dados no sistema, sendo a primeira a ser acionada no pipeline de ingestão pelo **Agente de Descompactação**.

- **Propósito:** Sua função é estritamente operacional: receber o caminho de um arquivo .zip e extrair seu conteúdo para um diretório de destino especificado.
- **Funcionamento:** A ferramenta utiliza a biblioteca padrão zipfile do Python para realizar a descompactação. Ela foi construída com robustez em mente,

realizando verificações para garantir que o diretório de destino exista e que o arquivo de origem seja encontrado, tratando exceções comuns como `FileNotFoundException` e `BadZipFile`. Após a extração, ela inspeciona o conteúdo e compila uma lista com os caminhos completos de todos os arquivos extraídos.

- **Saída e Importância Estratégica:** A ferramenta retorna um content textual confirmando o sucesso da operação e, mais importante, um artifact que é uma lista de strings contendo os caminhos para os arquivos CSV recém-extraídos. Este artefato é a “ponte” para a próxima etapa do fluxo de trabalho, pois ele se torna a entrada de dados para o **Agente de Mapeamento**, garantindo uma transição de dados automatizada e sem falhas entre os especialistas.

5.5.3 Ferramenta de Mapeamento de CSVs (MapCSVsToIngestionArgsTool)

Esta é uma das ferramentas mais complexas e centrais do pipeline de ETL (Extração, Transformação e Carga), sendo a principal habilidade do **Agente de Mapeamento**. Ela é responsável pela etapa de “Transformação”.

- **Propósito:** Ler os arquivos CSV brutos, que podem ter formatos e nomes de colunas variados, e transformá-los em um conjunto de dados limpo, padronizado e pronto para ser inserido no banco de dados.
- **Funcionamento:** A ferramenta opera com base em um `ingestion_config_dict`, um dicionário de configuração que desacopla a lógica de mapeamento do código-fonte. Este dicionário define, para cada tipo de arquivo esperado (identificado via expressão regular no nome do arquivo), qual coluna do CSV corresponde a qual campo no modelo de dados do banco e quais funções de conversão (converter) devem ser aplicadas (ex: transformar uma string de data em um objeto `datetime` ou um valor monetário em um número). A ferramenta itera sobre cada arquivo CSV, aplica as regras de mapeamento e conversão linha por linha usando a biblioteca `Pandas` (`pandas, c2023`), e salva um novo arquivo CSV processado em um diretório de destino.
- **Saída e Importância Estratégica:** O artifact gerado por esta ferramenta é uma lista de dicionários, onde cada dicionário representa um “argumento de ingestão”. Ele contém o nome da tabela de destino (`table_name`) e o caminho

para o arquivo CSV já processado e limpo (file_path). Esta saída estruturada é fundamental, pois serve como uma instrução precisa para o **Agente de Inserção de Registros**, eliminando qualquer ambiguidade na etapa final do carregamento de dados. A abordagem baseada em configuração torna o sistema extensível e de fácil manutenção, permitindo que novos formatos de CSV sejam suportados sem a necessidade de alterar o código da ferramenta.

5.5.4 Ferramenta de Inserção de Registros (InsertRecordsIntoDatabaseTool)

Esta ferramenta conclui o pipeline de ETL, representando a etapa de “Carga”. É a principal ferramenta do **Agente de Inserção de Registros**.

- **Propósito:** Receber os argumentos de ingestão gerados pela ferramenta de mapeamento e persistir os dados de forma eficiente e segura no banco de dados PostgreSQL.
- **Funcionamento:** A ferramenta utiliza **SQLAlchemy**, um ORM (Mapeador Objeto-Relacional), para interagir com o banco de dados. Ela itera sobre a lista de argumentos de ingestão, lê cada arquivo CSV processado e, para cada linha, instancia o modelo SQLAlchemy correspondente à tabela de destino. Uma das suas características mais importantes é o tratamento robusto de erros, especialmente a IntegrityError. Se a ferramenta tenta inserir um registro que viola uma restrição de unicidade (como uma NF-e com uma chave de acesso já existente), ela captura a exceção, desfaz a inserção daquela linha específica (rollback), registra um aviso e continua o processamento das linhas seguintes.
- **Saída e Importância Estratégica:** Ao final do processo, a ferramenta retorna um artifact que é um dicionário (count_map) resumindo quantos novos registros foram inseridos em cada tabela. A capacidade de ignorar duplicatas de forma inteligente, sem interromper todo o processo de carga, é crucial para a usabilidade do sistema em cenários reais, onde a reimportação de arquivos pode ser comum. Isso garante a idempotência da operação de carga e a integridade do banco de dados.

5.5.5 Kit de Ferramentas de Banco de Dados (AsyncSQLDatabaseToolkit)

Este não é uma única ferramenta, mas um conjunto coeso de habilidades que equipa o **Agente de Análise de Dados** para realizar qualquer tipo de consulta.

- **Propósito:** Fornecer um conjunto completo de capacidades para que o agente possa explorar, validar e consultar o banco de dados SQL de forma autônoma.
- **Componentes do Kit:**
 - AsyncQuerySQLDatabaseTool: A ferramenta principal. Ela permite que o agente execute qualquer consulta SELECT de forma assíncrona. Esta é a ferramenta que o agente usa para, de fato, buscar os dados solicitados pelo usuário.
 - InfoSQLDatabaseTool e ListSQLDatabaseTool: Ferramentas auxiliares que permitem ao agente listar todas as tabelas disponíveis e obter informações sobre o esquema de tabelas específicas, respectivamente. São usadas em conjunto com a GetDetailedTableSchemasTool para a fase de exploração.
 - QuerySQLCheckerTool: Uma ferramenta de segurança e validação. Antes de executar uma consulta complexa, o agente pode usá-la para verificar se a sintaxe SQL está correta, reduzindo a chance de erros em tempo de execução.
- **Importância Estratégica:** Este kit de ferramentas é o que transforma o Agente de Análise em um verdadeiro “consultor”. Ele não segue apenas scripts pré-definidos; ele tem os meios para explorar o ambiente de dados, entender sua estrutura, formular suas próprias consultas, validá-las e executá-las. É essa autonomia que permite ao sistema responder a uma gama virtualmente infinita de perguntas em linguagem natural feitas no bate-papo, tornando a análise de dados verdadeiramente dinâmica e interativa.

5.5.6 Ferramenta de Obtenção de Esquemas das Tabelas (GetDetailedTableSchemasTool)

Para que o **Agente de Análise de Dados** possa construir consultas SQL corretas, ele primeiro precisa entender a estrutura do banco de dados. A

GetDetailedTableSchemasTool é a ferramenta que lhe confere essa capacidade de “visão”.

- **Propósito:** Sua função é fornecer ao agente um mapa detalhado das tabelas, incluindo não apenas os nomes das colunas e seus tipos de dados, mas, crucialmente, os **comentários** associados a cada coluna.
- **Funcionamento:** A ferramenta recebe uma lista de nomes de tabelas e executa uma consulta SQL diretamente nos metadados do PostgreSQL (information_schema e pg_catalog). Essa consulta extrai as descrições que foram definidas durante a modelagem de dados (ex: “Valor total da nota fiscal” para a coluna total_invoice_value).
- **Importância Estratégica:** Esta ferramenta é vital para mitigar o risco de “alucinação” do LLM, onde o modelo poderia adivinhar nomes de colunas incorretos. Ao ler as descrições em linguagem natural, o agente pode correlacionar de forma muito mais precisa a pergunta do usuário com os campos corretos do banco de dados, tornando suas consultas mais assertivas e confiáveis. É um mecanismo de autodescoberta que confere autonomia e precisão ao agente analítico.

5.5.7 Ferramenta de Geração de Gráfico de Barras (GenerateBarPlotTool)

Uma vez que o Agente de Análise de Dados obtém os dados, ele precisa de uma forma de apresentá-los visualmente. Esta ferramenta lhe dá a capacidade de criar gráficos de barras.

- **Propósito:** É utilizada para visualizar a frequência ou a contagem de **dados categóricos ou ordinais**. É ideal para responder a perguntas como “Quais os 5 CFOPs mais utilizados?” ou “Qual o número de notas por UF?”.
- **Funcionamento:** A ferramenta recebe uma consulta SQL e o nome da coluna a ser analisada. Ela executa a consulta, carrega o resultado em um DataFrame Pandas e realiza uma série de validações. Notavelmente, ela verifica se a coluna selecionada é de fato categórica; caso seja numérica, ela lança uma exceção, instruindo o LLM a usar a ferramenta de distribuição, o que demonstra uma lógica de autogovernança. Em seguida, ela calcula a contagem

de cada categoria, limitando-se às 20 principais para manter a clareza visual, e usa a biblioteca Altair para gerar uma especificação de gráfico em formato JSON.

- **Saída:** A ferramenta retorna dois resultados: um content textual que resume o que o gráfico mostra e um artifact contendo a especificação JSON do gráfico, que pode ser renderizada diretamente pela camada de apresentação (Streamlit).

5.5.8 Ferramenta de Geração de Gráfico de Distribuição (GenerateDistributionPlotTool)

Como contrapartida à ferramenta de gráfico de barras, esta é especializada em **dados numéricos e contínuos**.

- **Propósito:** Serve para gerar histogramas que mostram a distribuição de valores em uma determinada coluna. É a ferramenta ideal para perguntas como “Mostre a distribuição dos valores totais das notas fiscais” ou “Como se distribuem as quantidades dos itens?”.
- **Funcionamento:** Recebe uma consulta SQL, o nome da coluna numérica e um parâmetro opcional `split_by`. Após executar a query, a ferramenta utiliza as bibliotecas NumPy e Pandas para calcular eficientemente os “bins” (intervalos) do histograma. Sua funcionalidade mais poderosa é a capacidade de usar o parâmetro `split_by` para segmentar a distribuição por uma categoria, permitindo análises comparativas ricas (ex: visualizar a distribuição do valor das notas, separada por UF, no mesmo gráfico). Assim como a outra ferramenta de plotagem, ela utiliza Altair para gerar a especificação do gráfico.
- **Saída:** Retorna um resumo textual, que inclui uma métrica estatística útil como a assimetria (skewness) da distribuição, e o artifact com a especificação JSON do histograma para renderização.

5.6 Fluxo de Trabalho da Solução

Para ilustrar a operação integrada de todos os componentes descritos, podemos detalhar dois cenários de uso principais:

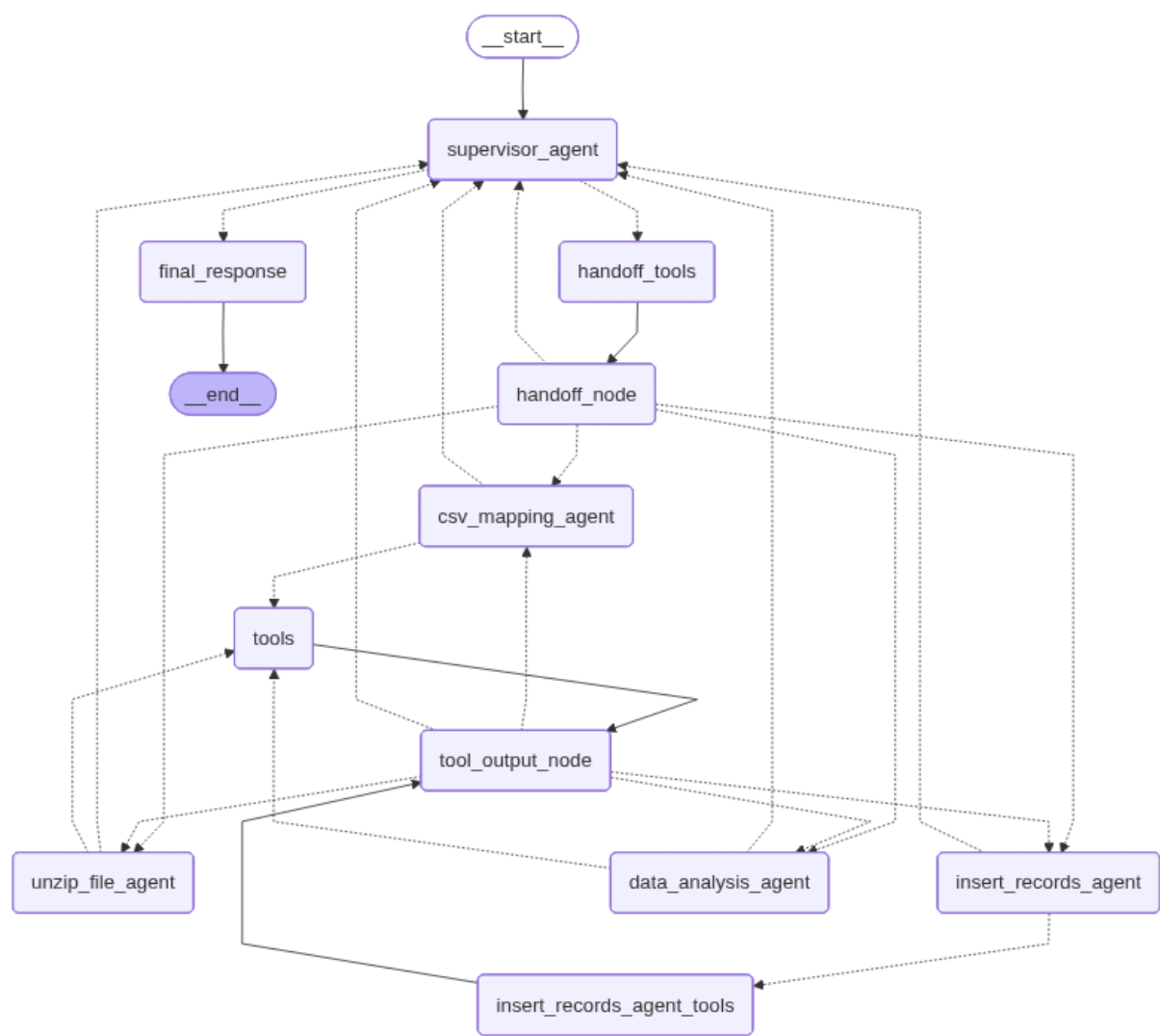
1. Cenário de Ingestão de Dados:

- O usuário faz o upload de um arquivo .zip na interface.
- A requisição chega ao **Agente Supervisor**, que delega a tarefa de descompactação ao **Agente de Descompactação**.
- Após a extração dos arquivos CSV, o Supervisor delega a tarefa de mapeamento ao **Agente de Mapeamento**, que transforma os dados em argumentos de ingestão.
- Em seguida, o Supervisor aciona o **Agente de Inserção de Registros**, que persiste os dados no banco PostgreSQL.
- Finalmente, o Supervisor informa ao usuário que o processo foi concluído com sucesso.

2. Cenário de Análise de Dados via Bate-Papo:

- O usuário digita uma pergunta em linguagem natural, como “Gere um gráfico de barras com o faturamento total dos 5 principais estados”.
- A pergunta é enviada ao **Agente Supervisor**, que a interpreta e delega a tarefa de análise ao **Agente de Análise de Dados**.
- O **Agente de Análise de Dados** primeiramente usa a ferramenta para obter o esquema do banco, identifica as colunas corretas (total_invoice_value e emitter_uf), executa uma consulta SQL para agregar os dados e, por fim, utiliza a GenerateBarPlotTool para criar a especificação do gráfico.
- O resultado (uma resposta textual e o código do gráfico) é retornado ao fluxo.
- O **Supervisor** formata a resposta final, que é enviada à camada de apresentação para ser exibida ao usuário na interface de bate-papo.

Figura 1 - Fluxo de Trabalho do Sistema de Gestão de NF-e com IA



6. AMBIENTE DE CONFIGURAÇÃO E EXECUÇÃO

Este capítulo detalha os procedimentos técnicos necessários para a configuração do ambiente e a execução da solução. A aplicação foi projetada para ser executada em um ambiente containerizado, utilizando a tecnologia Docker, a fim de garantir portabilidade, consistência e facilidade de implantação em diferentes sistemas operacionais.

6.1 Arquitetura de Implantação e Tecnologia de Containerização

Para garantir que a aplicação funcione de maneira idêntica e confiável independentemente da máquina do desenvolvedor ou do ambiente de produção, optou-se pela **containerização com Docker**. Esta abordagem encapsula a aplicação e todas as suas dependências (bibliotecas Python, frameworks e configurações de sistema) em unidades isoladas e padronizadas chamadas contêineres.

A arquitetura de implantação da solução é composta por dois contêineres distintos, orquestrados pelo docker-compose:

1. **Contêiner do Banco de Dados (postgresql):** Uma imagem oficial do **PostgreSQL** é utilizada para criar um serviço de banco de dados dedicado. Este contêiner é responsável pelo armazenamento do estado das conversas do agente de IA (mecanismo de checkpointing) e, crucialmente, pela persistência dos dados das NF-e. Isolar o banco de dados em seu próprio contêiner é uma boa prática que facilita o gerenciamento de dados, backups e atualizações de forma independente da aplicação.
2. **Contêiner da Aplicação (streamlit_app):** Esta imagem customizada contém todo o ecossistema da aplicação, incluindo o código-fonte em **Python**, os frameworks **Streamlit** (para a interface do usuário) e **LangGraph** (para o fluxo de trabalho dos agentes), além de todas as dependências de projeto listadas no arquivo de requisitos.

Essa separação garante um ambiente robusto, onde a lógica da aplicação e a camada de dados podem ser gerenciadas e escaladas de forma independente.

6.2 Pré-requisitos de Software

Antes de executar a aplicação, é necessário garantir que o seguinte software esteja instalado e configurado na máquina local.

6.2.1 Docker Desktop

A instalação do Docker é um requisito fundamental. Ele gerencia a criação e a execução dos contêineres.

- **Windows e macOS:** A instalação deve ser feita a partir do site oficial (<https://www.docker.com/products/docker-desktop/>). Após a instalação, é crucial verificar se o Docker Engine está em plena execução, o que é geralmente indicado pelo ícone do Docker na bandeja do sistema.
- **Linux:** A instalação varia conforme a distribuição. Após o processo, deve-se garantir que o serviço do Docker (dockerd) esteja ativo e que o usuário corrente possua as permissões necessárias para interagir com o daemon do Docker (geralmente, adicionando o usuário ao grupo docker).

Para confirmar a instalação bem-sucedida, o seguinte comando pode ser executado em um terminal, que deverá retornar a versão do Docker instalada:

```
Bash  
docker --version
```

6.2.2 Utilitário Make (Opcional)

Para simplificar os comandos de execução, o projeto disponibiliza um arquivo Makefile. O uso do utilitário Make é opcional, mas altamente recomendado para automatizar os comandos do Docker Compose.

- **Windows:** O Make pode ser instalado através do Git for Windows (<https://git-scm.com>), que inclui o ambiente Git Bash, onde o comando make é reconhecido.
- **Linux e macOS:** A maioria das distribuições baseadas em Unix já possui o Make pré-instalado. A verificação pode ser feita com o comando `make --version`.

6.3 Configuração de Variáveis de Ambiente

Para separar as configurações sensíveis e específicas do ambiente do código-fonte, a aplicação utiliza um arquivo de variáveis de ambiente.

O projeto inclui um arquivo de modelo chamado `.env.example`. É necessário criar uma cópia deste arquivo e renomeá-la para `.env`. A maioria das variáveis já possui valores padrão adequados para a execução local. A única variável que exige configuração manual obrigatória é a chave de API para o Modelo de Linguagem Grande (LLM).

As principais variáveis de ambiente relacionadas à IA que devem ser configuradas no arquivo `.env` são:

- **AI_LLM_MODEL:** Define o modelo de linguagem a ser utilizado.
 - Exemplo: `gpt-4.1-nano`
- **AI_LLM_TEMPERATURE:** Controla a criatividade ou aleatoriedade das respostas do modelo. Valores mais baixos (próximos de 0.0) tornam as respostas mais determinísticas e focadas.
 - Exemplo: `0.1`
- **AI_LLM_API_KEY:** A chave de API secreta para autenticação com o provedor do LLM (ex: OpenAI). **Este valor é obrigatório.**
 - Exemplo: `sk-xxxxxxxxxxxxxxxxxxxxxxxxxxxx`

6.4 Procedimentos de Execução

A aplicação foi projetada com flexibilidade, permitindo dois métodos de execução principais, cada um adequado a um cenário de uso distinto: um ambiente containerizado com Docker, ideal para distribuição e produção, e um ambiente de desenvolvimento local, que oferece maior agilidade para depuração e implementação de novas funcionalidades.

6.4.1 Método 1: Execução com Docker (Ambiente de Produção/Distribuído)

Este é o método recomendado para garantir a máxima consistência e portabilidade da aplicação. A containerização encapsula a aplicação e seu banco de dados em ambientes isolados, eliminando problemas de compatibilidade entre diferentes sistemas.

Execução Simplificada com Makefile

Para usuários com o utilitário Make instalado, o processo é simplificado por meio de comandos pré-configurados. No diretório raiz do projeto, execute os seguintes comandos em ordem:

1. Iniciar o contêiner do banco de dados: Este comando constrói e inicia o contêiner do PostgreSQL em segundo plano (-d).
 - Bash
`make startup-postgresql-db`
2. Formata o arquivo `entrypoint.sh`: Este comando converte os finais de linha do arquivo `entrypoint.sh` de terminações estilo DOS (CR+LF) para terminações estilo Unix (LF), resolvendo potenciais problemas de execução em sistemas Unix/Linux:
 - Bash
`make format-entrypoint`
3. Iniciar o contêiner da aplicação: Este comando constrói a imagem da aplicação Python e inicia seu contêiner, conectando-o à rede do banco de dados.

- Bash
make startup-streamlit-app

Execução Manual com Docker Compose

Caso o Make não esteja disponível, os mesmos resultados podem ser alcançados executando os comandos docker-compose diretamente:

2. Iniciar o contêiner do banco de dados:
 - Bash
docker-compose up --build -d postgresql_db
3. Formata o arquivo entrypoint.sh:
 - Bash
dos2unix entrypoint.sh
4. Iniciar o contêiner da aplicação:
 - Bash
docker-compose up --build -d streamlit_app

6.4.2 Método 2: Execução Local (Ambiente de Desenvolvimento)

Este método é ideal para desenvolvedores que precisam interagir diretamente com o código-fonte. Ele requer a instalação local das dependências Python do projeto (gerenciadas pelo uv) e executa a aplicação diretamente na máquina hospedeira, enquanto ainda utiliza um contêiner Docker para o banco de dados.

O processo de execução local consiste nos seguintes passos:

1. Iniciar o Banco de Dados: O serviço do PostgreSQL ainda é executado via Docker para simplificar sua configuração. Utilize um dos seguintes comandos no diretório raiz para iniciar apenas o contêiner do banco:
 - Com Makefile:
 - i. Bash
make startup-postgresql-db
 - Com Docker Compose:

- i. Bash
`docker-compose up --build -d postgresql_db`
- 2. Configurar a Variável de Ambiente PYTHONPATH: Para que o interpretador Python localize corretamente os módulos do projeto, é necessário configurar a PYTHONPATH para apontar para o diretório raiz.
 - o No Windows (Prompt de Comando):
 - i. Bash
`set PYTHONPATH=.`
 - o No Linux/macOS (Bash/Zsh):
 - i. Bash
`export PYTHONPATH=.`
- 3. Executar a Migração do Banco de Dados: Antes de iniciar a aplicação pela primeira vez, é necessário criar as tabelas invoices (Cabeçalho da Fatura) e invoice_items (Itens da Fatura) no banco de dados. Este processo é realizado pelo script de migração:
 - o Com Makefile:
 - i. Bash
`make migrate-postgresql-db`
 - o Manualmente com uv:
 - i. Bash
`uv run migrate_postgresql_db.py`
- 4. Iniciar a Aplicação Streamlit: Com o banco de dados em execução e as tabelas criadas, a aplicação pode ser iniciada:
 - o Com Makefile:
 - i. Bash
`make launch-streamlit-app`
 - o Manualmente com uv:
 - i. Bash
`uv run launch_streamlit_app.py`

6.5 Verificação e Acesso à Aplicação

Independentemente do método de execução escolhido, a aplicação web estará acessível em um navegador através do seguinte endereço, após a conclusão bem-sucedida dos passos:

- `http://localhost:8501`

O acesso a esta URL deve exibir a página inicial do **Sistema de Gestão de NF-e com IA**.

Para o método de execução com Docker, é possível verificar o status dos contêineres a qualquer momento com o comando:

- Bash
`docker ps`

A saída deverá listar os contêineres `postgresql_db` e/ou `streamlit_app` com o status “Up”, confirmando que os serviços estão em pleno funcionamento.

7. ELEMENTOS ADICIONAIS

Este capítulo complementa as descrições técnicas e funcionais apresentadas ao longo deste relatório com um conjunto de elementos visuais. O objetivo é fornecer uma compreensão mais clara e aprofundada tanto da dinâmica interna do fluxo de trabalho dos agentes de Inteligência Artificial quanto da interface final da aplicação, que representa a experiência do usuário.

7.1 Diagramas de Sequência do Fluxos de Trabalho

Para ilustrar a interação cronológica e a colaboração entre os diferentes componentes do sistema - o Agente Supervisor, os Agentes Especialistas e suas respectivas Ferramentas - foram elaborados os seguintes diagramas de sequência. Cada diagrama detalha um dos fluxos de trabalho centrais discutidos no Capítulo 5, demonstrando visualmente a ordem das mensagens trocadas e as ações executadas para completar uma tarefa específica.

Figura 2 - Fluxo de Descompactação de Arquivo ZIP

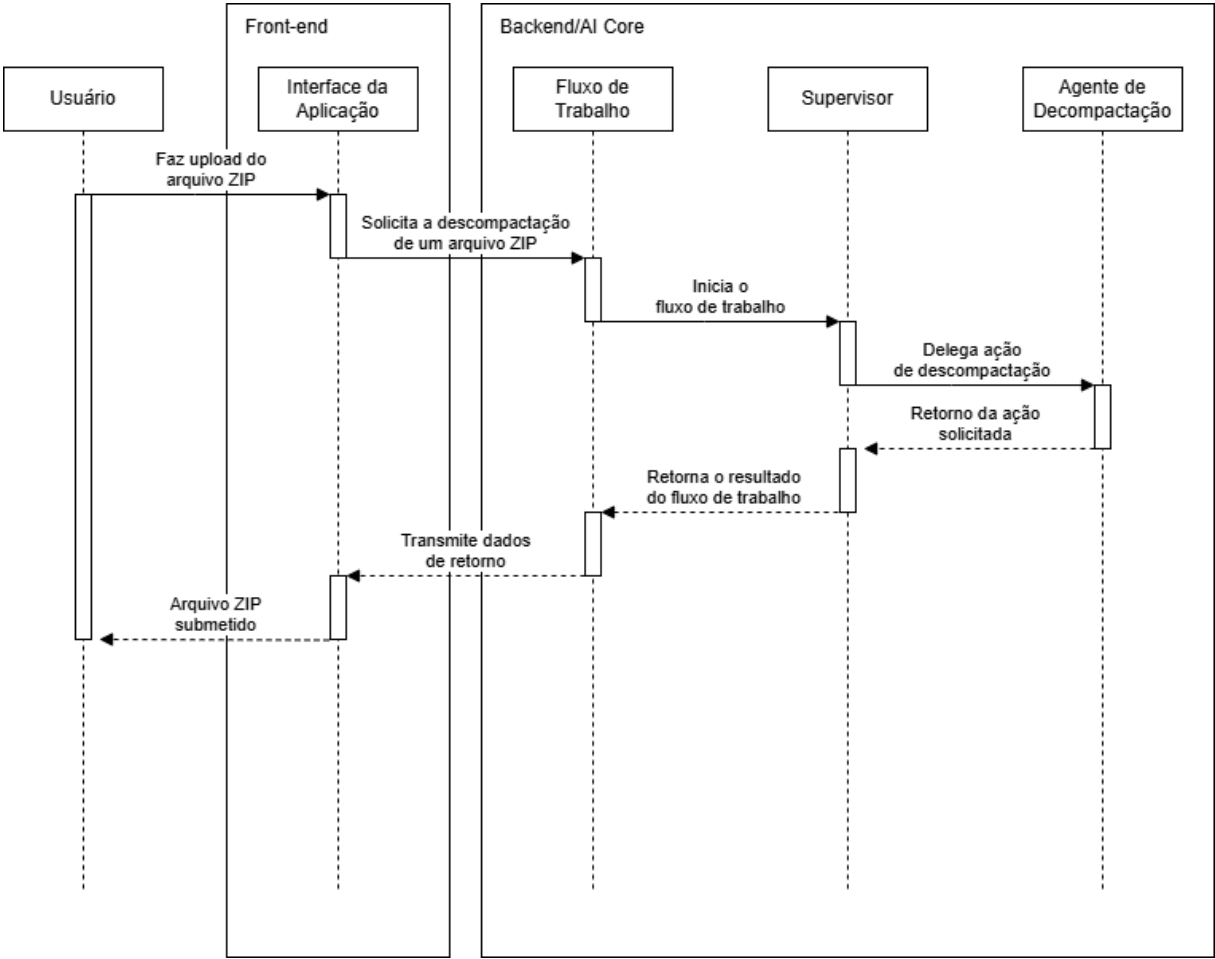


Figura 3 - Fluxo de Mapeamento de Arquivos CSV

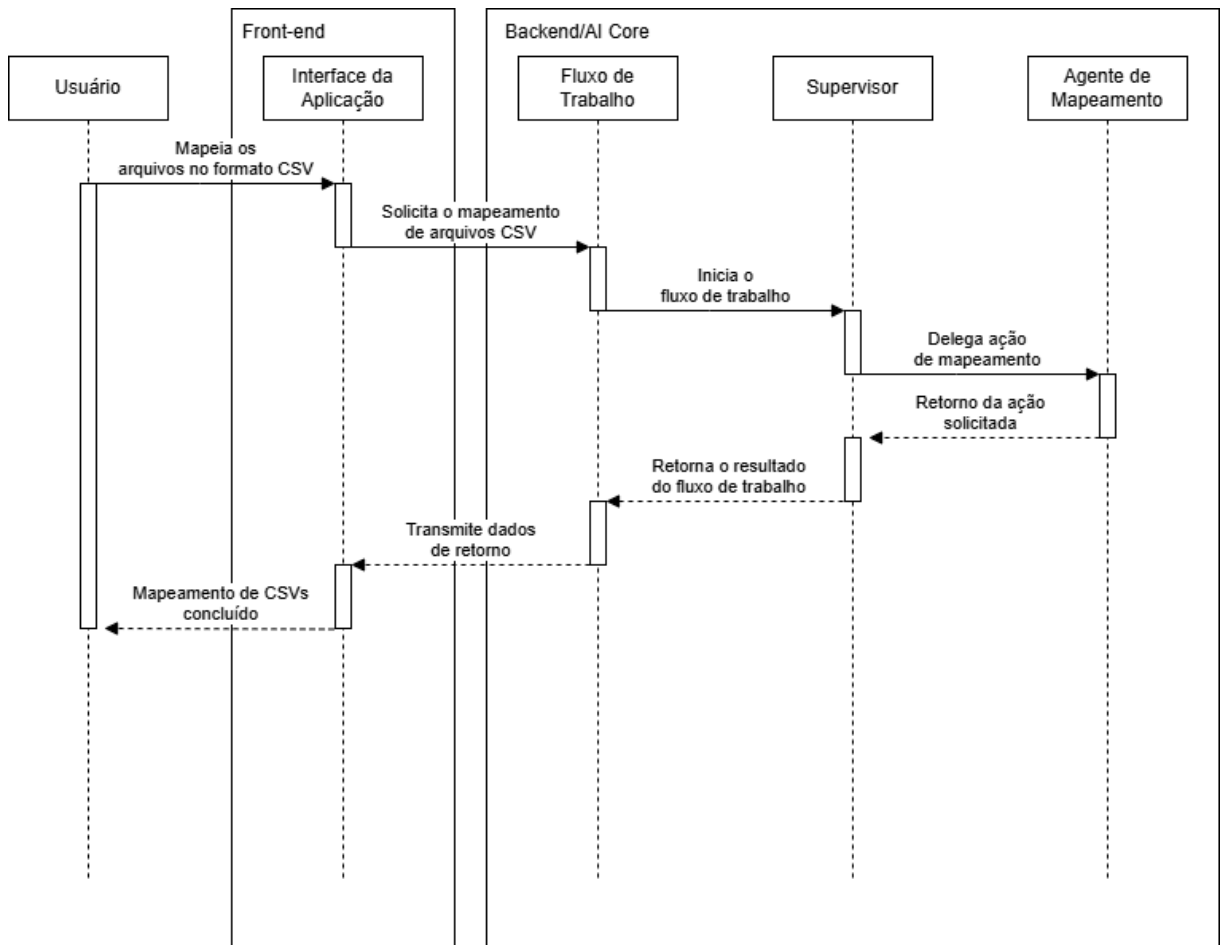


Figura 4 - Fluxo de Inserção de Registros no Banco de Dados

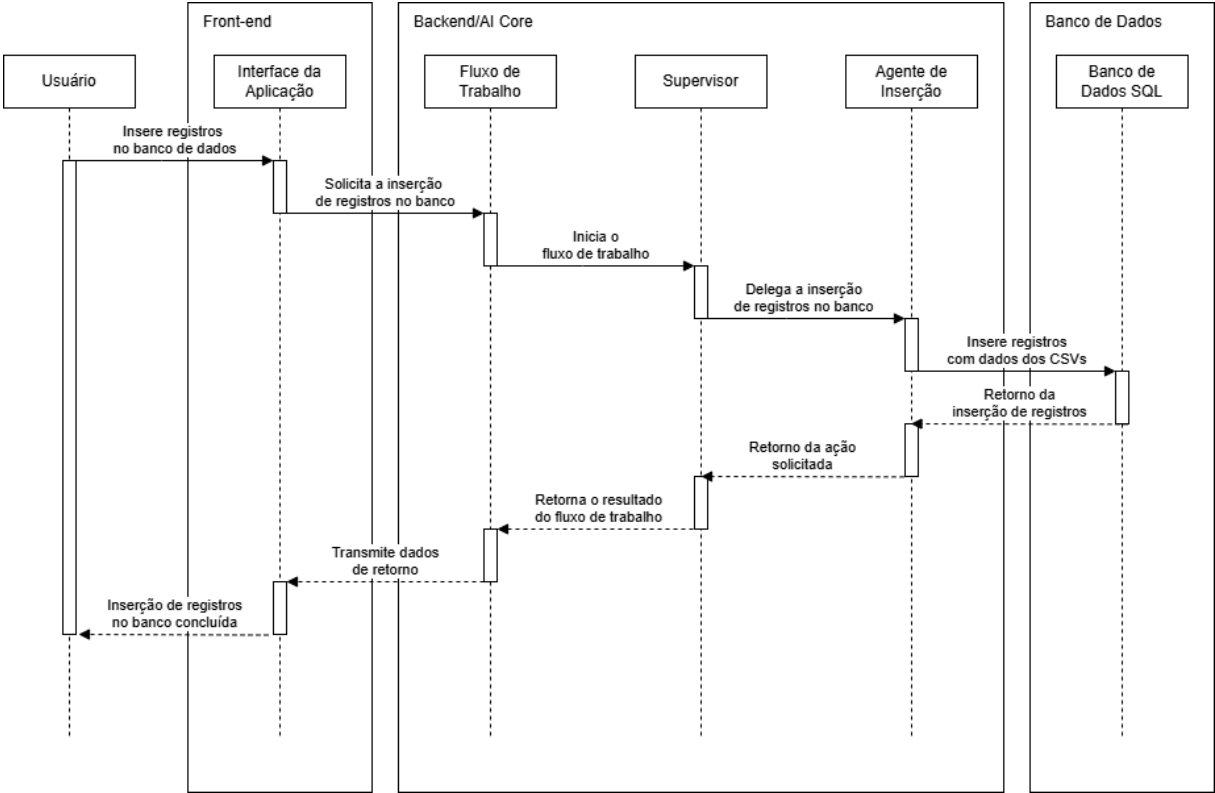
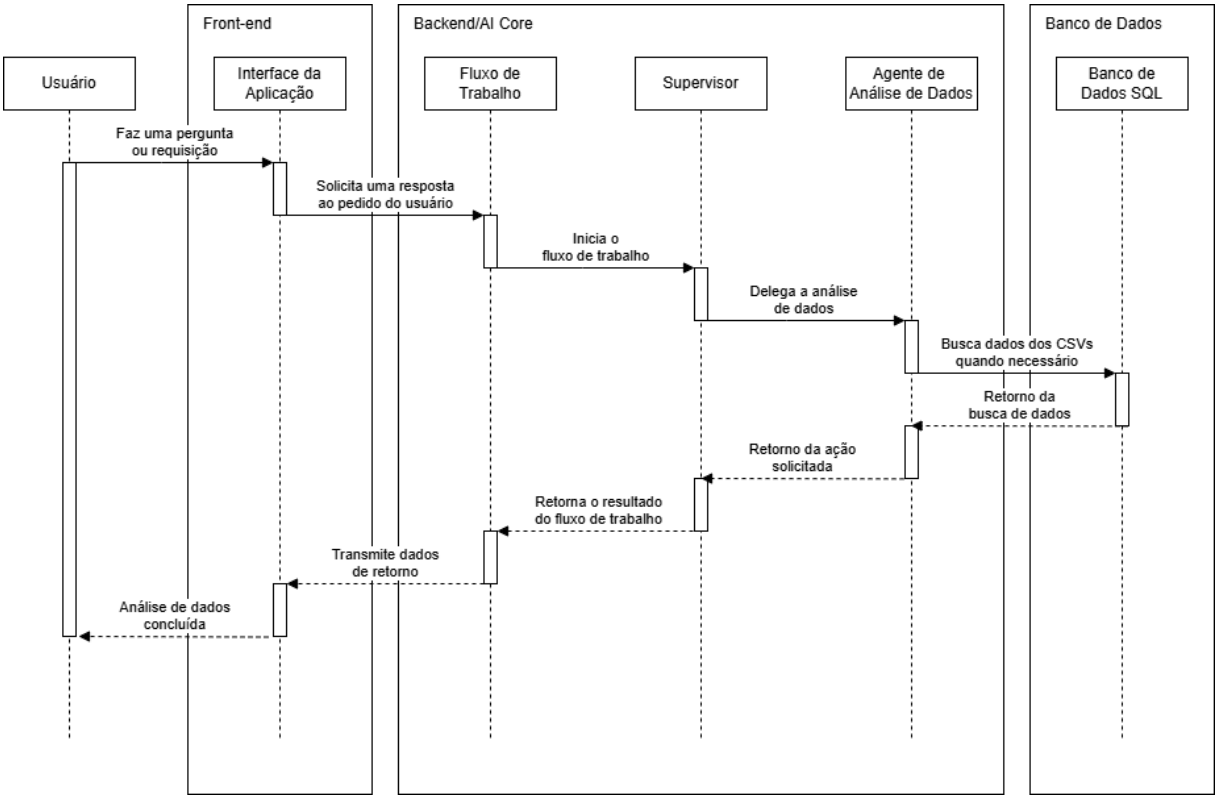


Figura 5 - Fluxo de Análise de Dados (Plotagem de Dados e Bate-Papo)



7.2 Telas das Funções do Sistema

Além da arquitetura interna, a usabilidade e a funcionalidade da interface do usuário são resultados cruciais deste projeto. As figuras a seguir apresentam as principais telas do Sistema de Gestão de NF-e com IA, desenvolvidas com o framework Streamlit. Elas oferecem um percurso visual através das funcionalidades da aplicação, desde a página inicial até as ferramentas de ingestão, análise e interação, demonstrando a concretização dos objetivos propostos.

Figura 6 - Tela Inicial

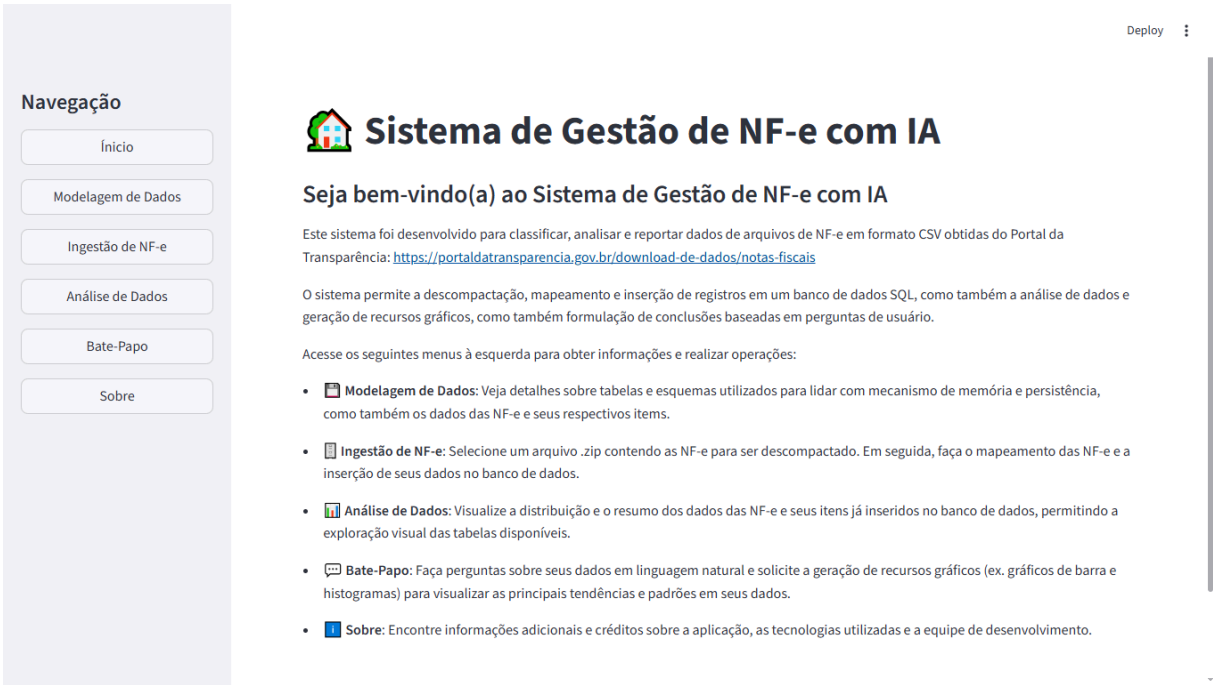


Figura 7 - Tela Modelagem de Dados



Figura 8 - Tela Ingestão de NF-e

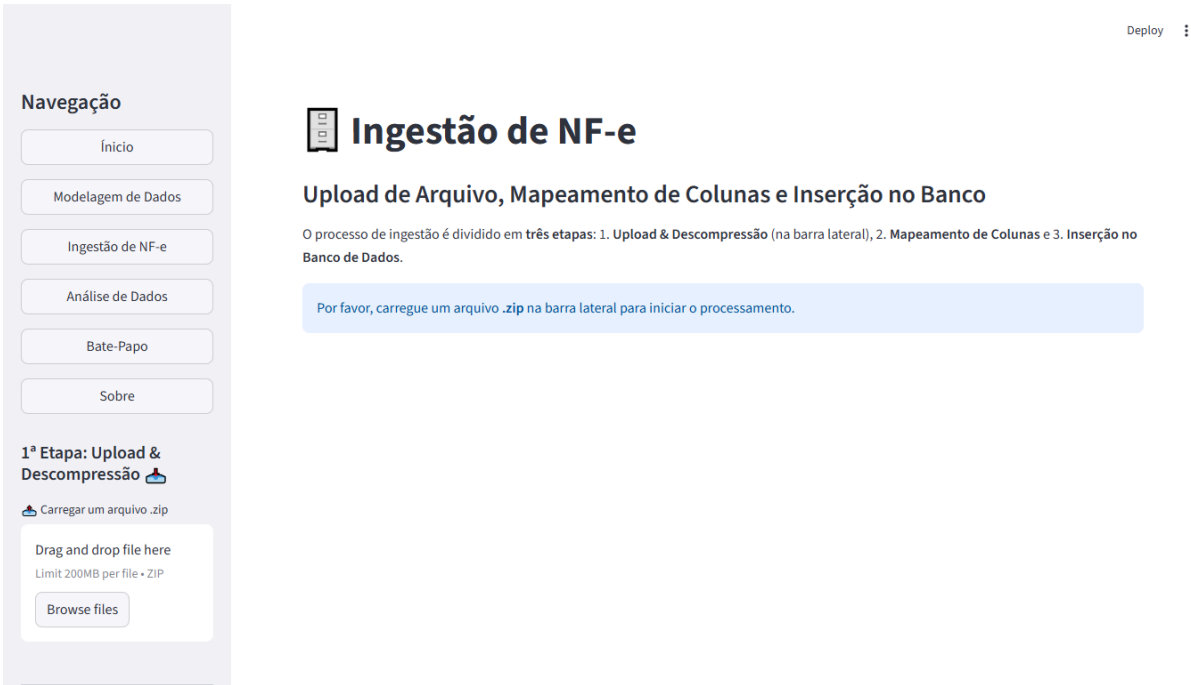


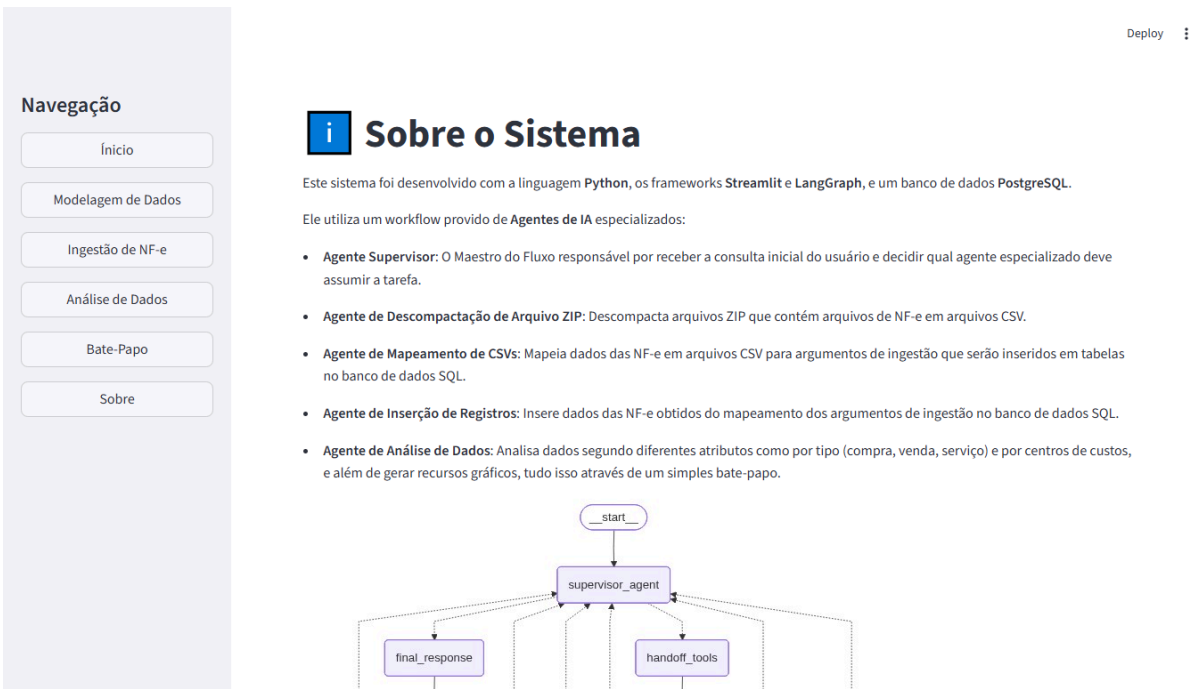
Figura 9 - Tela Análise de Dados



Figura 10 - Tela Bate-Papo



Figura 11 - Tela Sobre



8. RESULTADOS, DISCUSSÕES E CONCLUSÕES

A jornada de desenvolvimento de um projeto de conclusão de curso é, em sua essência, um exercício de transformação: de um problema teórico a uma solução tangível, de um conjunto de requisitos a um sistema funcional, e de um estudante a um profissional com experiência prática. Este capítulo final busca consolidar essa transformação, apresentando os resultados concretos alcançados pelo **Sistema de Gestão de NF-e com IA**, discutindo os aprendizados profundos adquiridos no processo e delineando as conclusões e os caminhos futuros que se abrem a partir deste trabalho. É também um momento de reflexão e agradecimento por uma jornada que, embora repleta de desafios técnicos e conceituais, culminou em um imenso crescimento pessoal e profissional.

8.1 Resultados Alcançados

O objetivo primário deste projeto foi desenvolver um protótipo funcional que demonstrasse a viabilidade e o valor da aplicação de um sistema multiagentes de Inteligência Artificial para automatizar e enriquecer a gestão de documentos fiscais. Ao final do ciclo de desenvolvimento, este objetivo foi plenamente alcançado. O resultado é uma plataforma web coesa e funcional que integra com sucesso os três temas centrais propostos:

1. **Automação do Pipeline de Dados Fiscais:** Foi implementado um fluxo de trabalho de ponta a ponta que automatiza com sucesso as etapas de **Extração, Transformação e Carga (ETL)**. O sistema é capaz de receber um arquivo .zip contendo dados de NF-e, orquestrar agentes de IA para descompactar, mapear as colunas para o esquema do banco de dados e inserir os registros de forma segura e idempotente, tratando duplicatas de forma inteligente. O resultado é a eliminação completa da necessidade de entrada manual de dados, validando a hipótese de que a automação pode reduzir drasticamente o tempo e os erros operacionais.
2. **Geração de Ferramentas Gerenciais Dinâmicas:** A plataforma oferece uma robusta camada de visualização de dados. Os dashboards interativos, que permitem a análise de métricas como faturamento, contagem de notas e

performance de produtos por meio de mapas, gráficos e tabelas, foram implementados com sucesso. O sistema transforma dados fiscais brutos, antes isolados em arquivos, em uma fonte rica de inteligência de negócio, permitindo a classificação e categorização das informações de forma intuitiva.

3. **Implementação de um Assistente Consultor Especializado:** O resultado mais inovador do projeto é a interface de bate-papo funcional. O sistema demonstrou a capacidade de interpretar perguntas complexas formuladas em linguagem natural, acionar o agente de análise para consultar o banco de dados e gerar não apenas respostas textuais, mas também visualizações gráficas sob demanda. Este assistente efetivamente democratiza o acesso à análise de dados, cumprindo o objetivo de criar um consultor de IA ativo.

8.2 Discussão dos Resultados e Aprendizados

A construção de um sistema que não apenas executa tarefas, mas também “raciocina” sobre elas, proporcionou aprendizados que transcendem a mera aplicação de tecnologias.

A Validade e Utilidade da Solução

Conforme destacado na justificativa deste trabalho, 95% das empresas brasileiras pagam mais impostos do que o devido. Este dado alarmante não é apenas uma estatística, mas a realidade de um ecossistema empresarial sobrecarregado pela complexidade. A ferramenta desenvolvida se posiciona como uma resposta direta a este problema. Ao automatizar a entrada de dados, ela ataca a fonte primária de erros de escrituração. Ao fornecer dashboards claros, ela dá aos gestores a visibilidade necessária para um planejamento tributário mais estratégico. E ao oferecer um assistente conversacional, ela capacita os tomadores de decisão a fazerem as perguntas certas e obterem respostas imediatas, sem intermediários. Em um momento de transição como a reforma tributária, ter uma ferramenta que organiza e dá sentido ao histórico fiscal não é um luxo, mas uma necessidade estratégica para a sobrevivência e competitividade do negócio.

Aprendizados no Desenvolvimento com Inteligência Artificial

Trabalhar intensivamente com um framework moderno como o LangGraph e com a arquitetura de agentes autônomos revelou lições valiosas:

1. **A Engenharia de Prompts é a Nova Programação:** Ficou evidente que a qualidade do sistema depende diretamente da precisão e da clareza dos prompts que definem a “personalidade” e as instruções de cada agente. A inclusão de “CRITICAL RULES” e a definição explícita de objetivos e ferramentas não são apenas detalhes, mas a essência da programação de um LLM. O processo é iterativo e exige uma mentalidade de “treinador” de IA, refinando as instruções até que o comportamento desejado seja alcançado de forma consistente.
2. **O Poder da Especialização em Sistemas Multiagentes:** A decisão de utilizar um Agente Supervisor orquestrando especialistas (Descompactação, Mapeamento, Inserção, Análise) provou-se fundamental para o sucesso do projeto. Em vez de um agente monolítico que tenta fazer tudo, a decomposição de responsabilidades tornou o sistema mais modular, mais fácil de depurar e mais robusto. Quando uma falha ocorria, era possível isolar exatamente qual “especialista” havia falhado e por quê.
3. **LLMs são Motores de Raciocínio, Ferramentas são a Ação:** Um dos maiores aprendizados foi a compreensão de que um LLM, por mais avançado que seja, é um cérebro sem braços. Sua capacidade de agir no mundo real (ler um arquivo, consultar um banco, gerar um gráfico) depende inteiramente da qualidade e da granularidade das ferramentas que lhe são fornecidas. O desenvolvimento de ferramentas robustas, como a `GetDetailedTableSchemasTool` (para evitar alucinações de colunas) ou as ferramentas de plotagem (para traduzir dados em imagens), foi tão ou mais importante quanto a engenharia dos prompts.

8.3 Oportunidades Futuras e Evolução do Projeto

Este trabalho estabelece uma base sólida sobre a qual muitas novas funcionalidades podem ser construídas. Algumas das oportunidades mais promissoras para a evolução do projeto incluem:

- **Processamento Direto de Documentos (XML/PDF):** A próxima fronteira seria eliminar a necessidade de arquivos CSV, integrando ferramentas de OCR (Reconhecimento Óptico de Caracteres) e NLP (Processamento de Linguagem Natural) para extrair dados diretamente de arquivos XML de NF-e ou até mesmo de documentos PDF escaneados.
- **Módulo de Validação e Auditoria Fiscal:** Implementar um novo agente especialista em regras fiscais. Este agente poderia, proativamente, analisar os dados inseridos em busca de inconsistências (ex: CFOP incompatível com a natureza da operação, cálculos de ICMS incorretos) e gerar relatórios de auditoria, alertando os usuários sobre potenciais riscos fiscais.
- **Integração Direta com Sistemas Contábeis e ERPs:** Desenvolver conectores para integrar a plataforma com sistemas de mercado como Domínio, Alterdata ou Protheus, permitindo a sincronização de dados e a automação de lançamentos contábeis.
- **Análise Preditiva:** Evoluir dos dashboards descritivos (o que aconteceu) para análises preditivas (o que irá acontecer). Utilizando modelos de Machine Learning, o sistema poderia prever tendências de faturamento, sazonalidade de produtos ou até mesmo estimar a carga tributária futura com base em cenários da reforma.

8.4 Conclusão e Agradecimentos

Este projeto é o resultado final de um ciclo de intenso aprendizado e dedicação. A construção do **Sistema de Gestão de NF-e com IA** demonstrou, na prática, o imenso potencial da Inteligência Artificial, especificamente da arquitetura de agentes autônomos, para resolver problemas complexos e de alto valor no domínio fiscal e contábil brasileiro. Mais do que um software, o resultado é um protótipo que materializa a visão de um futuro onde a gestão fiscal é menos sobre tarefas operacionais repetitivas e mais sobre inteligência estratégica e tomada de decisão informada.

A realização deste trabalho não teria sido possível sem o apoio, a orientação e a paciência de muitas pessoas. Por isso, é com grande satisfação que dedico este

espaço para agradecer às pessoas e instituições que tornaram este projeto uma realidade. Gostaria de expressar minha mais profunda gratidão aos meus familiares e amigos, que ofereceram o suporte emocional e o incentivo indispensáveis para superar os desafios. Agradeço o professor Celso Azevedo, cujos ensinamentos e direcionamentos estratégicos guiaram o desenvolvimento de cada etapa deste trabalho. Por fim, também registro meus agradecimentos ao **I2A2 - Institute of Applied Artificial Intelligence**, às empresas parceiras e a toda a equipe de planejamento e execução do curso **Agentes Autônomos com Redes Generativas** pela oportunidade de aprendizado em conceitos, ferramentas e tecnologias inovadoras. Cada obstáculo superado ao longo desta jornada não foi apenas uma barreira vencida, mas um degrau no fortalecimento do conhecimento aqui apresentado, preparando-me para os próximos desafios da minha carreira profissional.

9. REPOSITÓRIO DO PROJETO NO GITHUB

O código-fonte completo do sistema, que inclui todos os módulos, a arquitetura implementada e os scripts de execução, está disponível para acesso e consulta pública no seguinte repositório oficial do projeto no GitHub:
<https://github.com/icaroribeiro/grupo-sia/tree/main/projeto-final>

10. REFERÊNCIAS

ALEMBIC. **Alembic** [Ferramenta de migração de banco de dados para o ecossistema Python]. [S.l.]: Alembic, c2024. Disponível em: <https://alembic.sqlalchemy.org/en/latest/>. Acesso em: 27 out. 2025.

CONSELHO FEDERAL DE CONTABILIDADE. Profissionais da Contabilidade e Organizações Contábeis. Disponível em: <https://cfc.org.br/registro/quantos-somos-2/>. Acesso em: 22 out. 2025.

INSTITUTO BRASILEIRO DE PLANEJAMENTO E TRIBUTAÇÃO (IBPT). O IBPT se une ao IDV para apresentar um estudo inédito. Disponível em: <https://ibpt.org.br/levantamento-do-ibpt-mostra-que-95-das-organizacoes-brasileiras-pagam-mais-impostos-do-que-deveriam/>. Acesso em: 22 out. 2025.

LANGCHAIN. **LangGraph** [Biblioteca Python para aplicações stateful e multiagentes]. [S.l.]: LangChain, [s.d.]. Disponível em: <https://langchain.dev/langgraph>. Acesso em: 26 out. 2025.

MCKINNEY, W. **pandas** [Biblioteca Python para análise de dados e estruturas de dados de alto desempenho]. [S.l.]: pandas, c2023. Disponível em: <https://pandas.pydata.org/>. Acesso em: 27 out. 2025.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. **PostgreSQL** [Sistema Gerenciador de Banco de Dados Relacional Open Source]. [S.l.]: PostgreSQL Global Development Group, [s.d.]. Disponível em: <https://www.postgresql.org/>. Acesso em: 26 out. 2025.

PYTHON SOFTWARE FOUNDATION. **Python** [Linguagem de Programação]. Versão 3.x (ou a versão utilizada, se conhecida). [S.l.]: Python Software Foundation, [s.d.]. Disponível em: <https://www.python.org/>. Acesso em: 26 out. 2025.

SQLALCHEMY. **SQLAlchemy** [Toolkit SQL e Mapeador Objeto-Relacional (ORM) para Python]. [S.l.]: SQLAlchemy, c2024. Disponível em: <https://www.sqlalchemy.org/>. Acesso em: 27 out. 2025.

STREAMLIT INC. **Streamlit** [Software para construção de aplicações de dados].
[S.l.]: Streamlit Inc., [s.d.]. Disponível em: <https://streamlit.io/>. Acesso em: 26 out. 2025.