# CATCH: A Nova Fronteira dos Chatbots na Gestão de Força de Trabalho

Elon Oliveira Albuquerque [*] Bruno Justino Garcia Praciano [†]

Paulo Henrique Batista Rodrigues [‡] Flávio Garcia Praciano [§]

Fábio L. Lopes de Mendonça [¶] Geraldo P. Rocha Filho [‖]

**Resumo:** O Dimensionamento da Força de Trabalho (DFT) é um desafio essencial para a gestão de recursos humanos, especialmente no setor público, em que a eficiência e a precisão na alocação de pessoal são cruciais. A fase qualitativa do DFT, que depende de métodos manuais e subjetivos, enfrenta problemas significativos de imprecisão e ineficiência, criando uma lacuna na precisão das previsões e na gestão das demandas de trabalho. Em resposta a essas limitações, este trabalho propõe o CATCH – Chatbot para Automação de Tarefas e Comunicação Humanizada – como uma solução para aprimorar o DFT. O CATCH foi desenvolvido para automatizar a coleta e análise de dados qualitativos por meio de interações baseadas em inteligência artificial, o que resulta em uma abordagem mais precisa e eficiente para entender as necessidades de pessoal e otimizar a alocação de recursos. Em comparação com modelos existentes na literatura, o CATCH apresentou resultados superiores em termos de acurácia, com uma taxa de 0,9, tempo de resposta reduzido para 1,5 segundos e um detalhamento das respostas de 0,9. Esses resultados demonstram uma melhoria significativa na precisão das previsões e na eficiência do processo de DFT.

**Palavras-chave:** Inteligência Artificial, Dimensionamento da Força de Trabalho, Qualidade dos Serviços, Automação de Coleta de Dados, Chatbot.

[*]Universidade de Brasília - UnB, Brazil. Email: elon.albuquerque@gmail.com
[†]Universidade de Brasília - UnB, Brazil. Email: bruno.justino@ieee.org
[‡]Universidade de Brasília - UnB, Brazil. Email: paulo.rodrigues@redes.unb.br
[§]Universidade de Brasília - UnB, Brazil. Email: flavio.praciano@redes.unb.br
[¶]Universidade de Brasília - UnB, Brazil. Email: fabio.mendonca@redes.unb.br
[‖]Universidade Estadual do Sudoeste da Bahia - UESB, Brazil. Email: geraldo.rocha@uesb.edu.br

**Abstract:** Workforce Sizing (WFS) is an essential challenge for human resource management, especially in the public sector, where efficiency and accuracy in staff allocation are crucial. The qualitative phase of WFS, which relies on manual and subjective methods, faces significant issues of inaccuracy and inefficiency, creating a gap in forecast precision and workload management. In response to these limitations, this paper proposes CATCH – Chatbot for Automated Tasks and Humanized Communication – as a solution to enhance WFS. CATCH was developed to automate the collection and analysis of qualitative data through AI-based interactions, resulting in a more accurate and efficient approach to understanding personnel needs and optimizing resource allocation. Compared to existing models in the literature, CATCH showed superior results in terms of accuracy, with a 0.9 rate, reduced response time to 1.5 seconds, and a response detail score of 0.9. These results demonstrate a significant improvement in forecast precision and process efficiency in WFS.

**Keywords:** Artificial Intelligence, Workforce Sizing, Service Quality, Data Collection Automation, Chatbot.

## 1 Introdução

O Dimensionamento da Força de Trabalho (DFT) representa um desafio crucial para as unidades de gestão de pessoas Marques et al. (2022); Silva et al. (2024). Este processo é fundamental para garantir que o número ideal de profissionais especializados por categoria seja definido, de modo a atender às demandas da máquina pública Serrano et al. (2021). Em um contexto de incerteza macroeconômica, adotar práticas e processos inovadores é vital. Embora a iniciativa privada frequentemente adote tais práticas, elas ainda são limitadas no setor público da Silva et al. (2019). Novas tecnologias, como Robotic Process Automation (RPA), análise preditiva, chatbots e sistemas de gestão com inteligência artificial, são essenciais para aprimorar a eficácia operacional no setor público de Oliveira Mendes et al. (2020); Serrano et al. (2017).

O processo de DFT envolve várias etapas cruciais, incluindo a previsão da demanda, a previsão de suprimento e as estratégias para equilibrá-los Chowdhury (2016). As metodologias para o DFT podem ser divididas em duas fases. A fase qualitativa foca no levantamento das atividades, geralmente por meio de entrevistas, questionários e análise de tarefas, com o objetivo de entender as necessidades e o volume de trabalho. Em contraste, a fase quantitativa avalia a produtividade e o desempenho das atividades identificadas, utilizando dados históricos e métricas para calcular o número ideal de profissionais necessários. Essas duas fases juntas permitem uma análise abrangente e uma previsão mais precisa das necessidades de pessoal.

A fase qualitativa do DFT que é o foco desta pesquisa enfrenta várias limitações. Ela depende de métodos manuais e subjetivos, o que frequentemente resulta em imprecisões e ineficiências Marques et al. (2022); Silva et al. (2024). Além disso, a variabilidade nas respostas e interpretações dos envolvidos pode levar a inconsistências nos dados coletados, prejudicando a qualidade das informações utilizadas para o planejamento Silva et al. (2024). A introdução de chatbots transforma significativamente essa fase ao automatizar a coleta e análise de dados sobre as demandas de trabalho e a alocação de recursos. Chatbots são sistemas baseados em inteligência artificial projetados para interagir com usuários, fornecer respostas automatizadas e coletar informações detalhadas sobre processos e atividades. Essa automação não só melhora a precisão das previsões como também facilita a otimização da força de trabalho, contribuindo para a redução de custos e o aumento da eficiência do processo Pu et al. (2024); Rane et al. (2024); Silva et al. (2024).

Para sanar esse problema e otimizar o processo de DFT, este artigo apresenta o CATCH - **C**hatbot para **A**utomatização de **T**arefas e **C**omunicação **H**umanizada. Diferente de outras soluções que focam apenas na automação de tarefas repetitivas ou no atendimento ao cliente, o CATCH integra funcionalidades específicas para a análise e gestão de dados sobre a força de trabalho. O CATCH não apenas automatiza tarefas repetitivas, como responder a perguntas frequentes sobre o projeto, mas também melhora a coleta de dados qualitativos ao interagir diretamente com os funcionários e gerentes, coletando informações sobre as atividades e necessidades de pessoal. Isso permite uma análise mais precisa e em tempo real das demandas e do desempenho, facilitando a otimização da força de trabalho e contribuindo para a redução de custos e aumento da eficiência.

O restante deste artigo está organizado da seguinte forma: na Seção 2, são apresentados e discutidos os trabalhos relacionados na área. Na Seção 3, é descrito o modelo do CATCH. Na Seção 4, são apresentados os resultados obtidos para validar o CATCH. Por fim, a Seção 5 apresenta as conclusões e os trabalhos futuros.

## 2 Trabalhos Relacionados

Na literatura, existem trabalhos que adotam uma abordagem teórica e quantitativa para o DFT no setor público Serrano et al. (2017, 2021). Enquanto Serrano et al. (2021) foca na aplicação de modelos algébricos, estatísticos e de otimização para apoiar decisões estratégicas em cenários de incerteza econômica, Serrano et al. (2017) propõe uma metodologia baseada em modelagens matemáticas para instituições públicas federais, realizando uma análise das atividades organizacionais e o desenvolvimento de um sistema de apoio à decisão gerencial. Ambos os trabalhos compartilham uma visão teórica centrada no uso de modelos quantitativos para otimizar a gestão de recursos humanos, o que complementa a proposta do presente artigo, que descreve a aplicação prática de tecnologia, especificamente o desenvolvimento de um chatbot de inteligência artificial, para automatizar a coleta de dados qualitativos sobre as atividades das unidades de trabalho, otimizando processos não só no setor público.

Por outro lado, os estudos de da Silva et al. (2019) e de Oliveira Mendes et al. (2020) destacam a inovação na gestão pública, abordando a implementação de novos processos e tecnologias para melhorar a eficiência dos serviços. da Silva et al. (2019) foca na introdução de inovações no contexto da gestão pública municipal, analisando a percepção dos servidores sobre mudanças em um ambiente burocrático e legalmente rígido. Já de Oliveira Mendes et al. (2020) investiga o impacto do teletrabalho na administração pública, evidenciando os benefícios em termos de redução de custos administrativos e melhoria da qualidade de vida dos servidores. Ambos os estudos, embora focados em diferentes formas de inovação, reforçam a importância de modernizar a gestão pública. Entretanto, o presente trabalho diferencia-se por propor a automação por meio de um chatbot, integrando inteligência artificial para otimizar a coleta de dados e reduzir o uso de recursos humanos e financeiros no DFT.

Em Khennouche et al. (2024), os autores exploram uma taxonomia de chatbots, distinguindo entre modelos baseados em regras, recuperação e gerativos, como o ChatGPT. A pesquisa enfatiza a personalização e os desafios éticos na integração de chatbots com bases de conhecimento. Já o trabalho de Liu et al. (2024) investiga a eficácia de chatbots gerativos, confirmando que interações mais naturais, como as proporcionadas por esses chatbots, podem aumentar a eficácia das intervenções. Ambas as pesquisas destacam o potencial dos chatbots gerativos; porém, as limitações de ambos os trabalhos residem na falta de foco em aplicações para a gestão da força de trabalho, como a análise de métricas de desempenho ou a otimização de processos. No contexto do DFT, tais funcionalidades são cruciais, mas não são abordadas de maneira direta nos estudos citados.

Ao contrário das abordagens supracitadas, que se concentram em métodos teóricos, quantitativos e inovações gerenciais, esta pesquisa introduz uma aplicação prática para o DFT ao propor um chatbot de inteligência artificial para automatizar a coleta e análise de dados qualitativos sobre as atividades das unidades de trabalho, como apresentado na Tabela 1. O objetivo é não apenas otimizar a gestão da força de trabalho, mas também reduzir significativamente o esforço humano necessário e aprimorar a precisão dos dados coletados. A pesquisa se destaca por integrar tecnologias avançadas que visam aumentar a eficiência, reduzir custos e preencher lacunas deixadas por estudos anteriores, oferecendo uma solução prática e adaptável para a análise e otimização das atividades e métricas de desempenho no setor público.

Tabela 1: Sumarização das contribuições dos trabalhos relacionados e do trabalho proposto.

| Trabalhos | Abordagem teórica e quantitativa para o DFT | Inovação na gestão pública com novas tecnologias | Automação da coleta de dados qualitativos | Redução de recursos humanos e financeiros |
|---|---|---|---|---|
| **Serrano et al. (2021)** | Sim | Não | Não | Não |
| **Serrano et al. (2017)** | Sim | Não | Não | Não |
| **da Silva et al. (2019)** | Não | Sim | Não | Não |
| **de Oliveira Mendes et al. (2020)** | Não | Sim | Não | Não |
| **Khennouche et al. (2024)** | Não | Não | Não | Não |
| **Liu et al. (2024)** | Não | Não | Não | Não |
| **CATCH** | Sim | Sim | Sim | Sim |

## 3 CATCH - Chatbot para Automatização de Tarefas e Comunicação Humanizada

Esta seção apresenta o CATCH, um **C**hatbot para **A**utomatização de **T**arefas e **C**omunicação **H**umanizada com o intuito de otimizar o processo de DFT. Desenvolvido com base em Inteligência Artificial (IA) e em modelos de Processamento de Linguagem Natural (PLN), o CATCH visa automatizar a coleta e análise de dados qualitativos, permitindo que os gestores tomem decisões estratégicas com base em dados precisos e em tempo real. A integração dessas tecnologias busca garantir maior eficiência no gerenciamento da força de trabalho.

### 3.1 Arquitetura do CATCH

A arquitetura do CATCH é projetada para fornecer respostas precisas e contextualizadas através de uma estrutura modular que integra tecnologias de PLN e recuperação de informações, conforme ilustrada na Figura 1. O CATCH foi modelado com base no ChromaDB, um banco de dados vetorial para armazenar e recuperar grandes volumes de informações contextuais, e o LLaMA3, um modelo de linguagem que, em combinação com o ChromaDB, gera respostas contextualizadas. O ChromaDB é responsável pela indexação e busca de passagens relevantes, enquanto o LLaMA3 emprega técnicas de modelagem de linguagem para fornecer respostas informadas e pertinentes ao contexto da pergunta.

A interação com o CATCH inicia quando o usuário realiza a submissão de perguntas, a qual foi implementada com o framework FastAPI (Rótulo 1, Figura 1). Esta API gerencia as solicitações e garante a comunicação fluida entre o usuário e o backend do sistema. A implementação do CORS (Cross-Origin Resource Sharing) assegura a compatibilidade do CATCH com diferentes ambientes web, permitindo uma integração transparente com diversas plataformas.

Após a validação da solicitação, a pergunta é encaminhada para o processamento de linguagem (Rótulo 2, Figura 1). Neste estágio, a pergunta é convertida em uma representação vetorial utilizando o modelo de embeddings SentenceTransformer. Este modelo é especializado em capturar a semântica e o significado das questões, facilitando uma compreensão mais profunda do conteúdo. A representação vetorial resultante é então utilizada pelo ChromaDB para buscar passagens relevantes armazenadas, fornecendo uma base sólida de informações contextuais.

Com as passagens relevantes recuperadas, o CATCH utiliza o modelo de Geração Aumentada por Recuperação (RAG) para combinar essas informações com o conhecimento do modelo de linguagem LLaMA3 (Rótulo 3, Figura 1). O modelo RAG é eficaz na integração das informações recuperadas com a geração de linguagem natural, produzindo respostas enriquecidas e personalizadas. A resposta final é então apresentada ao usuário através da interface do CATCH, completando o ciclo de interação.

### 3.2 Gerenciamento de Consultas e Controle de Acesso no CATCH

O CATCH utiliza o FastAPI, um framework para construção de APIs web, com o objetivo de gerenciar de forma otimizada as requisições dos usuários. Esse framework foi escolhido por sua capacidade de lidar com operações assíncronas e sua compatibilidade com Python 3.7+, proporcionando um gerenciamento eficiente do ciclo de vida das requisições e melhorando a latência do sistema. A funcionalidade de submissão de consultas dos usuários (Rótulo 1,

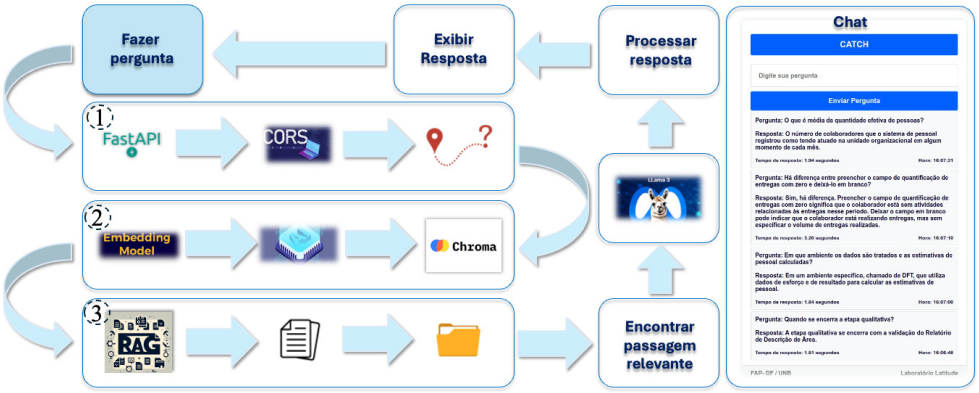


Figura 1: Visão geral da arquitetura do CATCH

Figura 1) é implementada diretamente pelo FastAPI, permitindo o processamento paralelo de requisições em tempo real. Além disso, o FastAPI oferece suporte a recursos como a validação de dados automatizada através do Pydantic, facilitando a detecção de inconsistências nos dados enviados e a redução de falhas no processamento. O FastAPI também gera automaticamente documentação interativa compatível com OpenAPI e Swagger, facilitando o uso da API e a integração com outras ferramentas de desenvolvimento. Dessa forma, o sistema melhora a responsividade e mantém a consistência nas interações.

Para garantir que o CATCH seja acessível a partir de diferentes domínios web, foi implementado o CORS, um componente que controla a política de compartilhamento de recursos entre origens distintas. O CORS é responsável por configurar regras que garantem que apenas requisições de domínios autorizados sejam aceitas, bloqueando tentativas não autorizadas que poderiam comprometer a integridade e a segurança do sistema. Esse mecanismo é utilizado em ambientes onde a API do CATCH pode ser consumida por diferentes aplicações web distribuídas, promovendo uma integração com múltiplas plataformas. O CORS, ao configurar políticas de controle de acesso, permite que o CATCH atenda a solicitações legítimas sem comprometer a segurança, promovendo interoperabilidade e acessibilidade em um ambiente web distribuído. Esse mecanismo assegura que a comunicação entre o frontend e o backend seja contínua e eficiente, independentemente da origem do acesso, permitindo que o CATCH opere de maneira coesa em diversos contextos de uso e integração.

### 3.3 Processamento de Embeddings no CATCH

No CATCH, o modelo SentenceTransformer é responsável pela conversão das perguntas dos usuários em representações vetoriais de alta dimensionalidade, facilitando o processo de busca e recuperação de informações contextuais. O SentenceTransformer é especializado em capturar a semântica e o significado das consultas, gerando vetores que encapsulam tanto o conteúdo explícito quanto o contexto implícito das perguntas. Esses vetores são posteriormente utilizados pelo ChromaDB, um banco de dados vetorial otimizado para armazenamento e recuperação de grandes volumes de dados, projetado para realizar buscas baseadas em si-

milaridade semântica de maneira eficiente.

O ChromaDB desempenha um papel fundamental na infraestrutura do CATCH, permitindo a indexação e recuperação de passagens relevantes com base nas representações vetoriais geradas pelo SentenceTransformer. Este banco de dados utiliza técnicas de indexação vetorial, como árvores de busca aproximada, para garantir que as consultas sejam respondidas com baixa latência, mesmo com grandes volumes de dados armazenados. Dessa forma, o CATCH é capaz de localizar informações pertinentes com precisão e eficiência.

A integração entre o SentenceTransformer e o ChromaDB possibilita ao CATCH realizar uma recuperação eficiente de informações contextuais em cenários complexos. O fluxo de trabalho inicia-se com a conversão das perguntas em vetores pelo SentenceTransformer, que gera uma representação densa de cada consulta. Em seguida, o ChromaDB utiliza essas representações para buscar as passagens mais relevantes, com base em medidas de similaridade semântica. Após a recuperação das passagens, o modelo RAG combina essas informações com o modelo de linguagem LLaMA3, que gera respostas contextualizadas. Esse processo de integração entre recuperação de informações e geração de linguagem natural assegura que as respostas fornecidas pelo CATCH sejam coerentes e alinhadas ao contexto original da consulta.

### 3.4 Mecanismo para Geração de Respostas no CATCH

No CATCH, a integração entre o modelo RAG e o LLaMA3 é fundamental para assegurar a precisão e relevância contextual das respostas geradas. Essa combinação não é trivial, pois envolve a coordenação entre a recuperação eficiente de informações e a geração de linguagem natural coerente. O RAG desempenha a função de recuperar passagens contextuais relevantes a partir de uma base de dados, enquanto o LLaMA3 utiliza essas passagens para gerar respostas que não apenas refletem o conteúdo das informações recuperadas, mas também estão alinhadas ao contexto específico da consulta do usuário. Esse processo, ilustrado no Algoritmo 1, garante que as respostas geradas pelo CATCH sejam tanto informativas quanto adequadamente contextualizadas, lidando com a complexidade e a necessidade de precisão em interações de linguagem natural.

O RAG foi modelado para combinar técnicas de recuperação de informações com geração de texto, operando em duas fases principais: a recuperação de passagens e a geração da resposta final. Na fase de recuperação, o RAG busca as informações relevantes em um banco de dados utilizando uma abordagem baseada em vetores. No CATCH, o RAG acessa o ChromaDB para recuperar passagens pertinentes que estão indexadas no banco de dados. A recuperação é realizada com base na representação vetorial da consulta do usuário, gerada pelo modelo SentenceTransformer. Essa representação vetorial captura a semântica e o significado da pergunta, permitindo que o RAG localize rapidamente as passagens que mais se aproximam do contexto da consulta, facilitando uma recuperação eficiente.

Após a fase de recuperação, o RAG passa para a fase de geração, em que combina as passagens recuperadas com o conhecimento do modelo de linguagem LLaMA3. O LLaMA3 utiliza as informações recuperadas para gerar uma resposta adaptada ao contexto específico da consulta. Essa abordagem de geração aumentada permite que o CATCH produza respostas que não são apenas precisas em termos de conteúdo, mas também altamente contextualizadas. O RAG garante que as passagens recuperadas sejam integradas de forma coesa e relevante, resultando em respostas que abordam diretamente a consulta do usuário, mantendo a precisão sem comprometer a fluidez.

O LLaMA3 tem um papel central na geração de respostas, empregando técnicas de PLN para interpretar as passagens recuperadas e produzir respostas que sejam gramaticalmente corretas e semanticamente adequadas. O modelo, desenvolvido com base na arquitetura Transformer, utiliza mecanismos de atenção multi-cabeça e codificação posicional para capturar relações de longo alcance entre as palavras, permitindo uma compreensão detalhada da estrutura textual. Pré-treinado em grandes volumes de dados, o LLaMA3 é ajustado para tarefas específicas, possibilitando uma interpretação robusta de nuances e contextos complexos, que são características essenciais para a aplicação em questão.

**Algorithm 1** Mecanismo de Geração de Respostas no CATCH

**Inicialização:**

Carregar modelo de embeddings $E$
Carregar banco de dados vetorial $D$
Carregar modelo de recuperação e geração $G$
Definir consulta do usuário $q$
Definir número de passagens $k$ e parâmetro de similaridade $\epsilon$

**Processamento da consulta:**

**function** GERAREMBEDDING($q$)
    $embedding_q \leftarrow E.encode(q)$
    **Retornar** $embedding_q$
**end function**

**Recuperação de passagens relevantes:**

**function** BUSCARINFORMACOES($embedding_q$, $D$, $k$, $\epsilon$)
    $resultados \leftarrow D.query(embedding_q, k, \epsilon)$
    **Retornar** $resultados$
**end function**

**Geração de resposta:**

**function** GERARRESPOSTA($resultados$, $G$)
    $resposta \leftarrow G.generate(resultados)$
    **Retornar** $resposta$
**end function**

**Execução completa:**

$embedding_q \leftarrow$ GERAREMBEDDING($q$)
$resultados \leftarrow$ BUSCARINFORMACOES($embedding_q$, $D$, $k$, $\epsilon$)
$resposta \leftarrow$ GERARRESPOSTA($resultados$, $G$)
**Retornar** $resposta$

No contexto do CATCH, o LLaMA3 processa as passagens recuperadas pelo RAG, utilizando técnicas de geração de texto baseadas em decodificadores para formular respostas que sejam detalhadas e relevantes. O modelo também aplica máscaras de contexto para garantir a coerência e continuidade na geração de texto, mantendo a integridade das informações recuperadas e gerando respostas fluentes. Essas técnicas permitem que o LLaMA3 forneça

respostas que não apenas respondam diretamente à pergunta, mas que também integrem e contextualizem as informações recuperadas, garantindo que a interação com o usuário seja precisa e adequada ao contexto da consulta.

## 4 Avaliação de Desempenho

Para avaliar o desempenho do mecanismo de geração de respostas implementado no CATCH, foram utilizados os modelos MISTRAL AI (2024) e PHI Microsoft (2024), além do baseline LLAMA 3.1 Llama (2024) para comparação. O objetivo principal foi medir a eficácia do CATCH em fornecer respostas adequadas, bem como avaliar a eficiência computacional durante o processamento das consultas.

O CATCH foi avaliado com base nas seguintes métricas:

- Tempo de resposta: O tempo médio para gerar uma resposta foi registrado para cada consulta, incluindo as etapas de geração de embeddings, recuperação de passagens e síntese da resposta final.
- Acurácia: A acurácia foi calculada como a proporção de respostas corretas (definidas como aquelas que correspondem à intenção da consulta) em relação ao total de consultas feitas.
- Detalhamento: Esta métrica avalia o grau de profundidade e especificidade das respostas fornecidas, medindo o quanto o sistema consegue explorar nuances e fornecer informações adicionais relevantes para a consulta.

Os experimentos a seguir foram conduzidos em um servidor com a seguinte configuração: sistema operacional Ubuntu 23.04 (Codinome: Lunar) com Kernel 6.2.0-39-generic. A CPU é um Intel(R) Core(TM) i7-10700F @ 2.90GHz, com arquitetura x86_64, 16 núcleos e suporte para 32 e 64 bits, utilizando 16 threads no total, com dois threads por núcleo. O servidor conta com 32 GiB de RAM e um volume de armazenamento LVM com capacidade de 921 GiB. Para processamento gráfico, foram utilizadas 6 GPUs NVIDIA GeForce RTX 3060 Ti, cada uma com 8 GiB de memória dedicada. A versão do Python utilizada foi a 3.12.

### 4.1 Impacto dos Resultados Obtidos

O gráfico de acurácia (Figura 2) apresenta uma comparação entre os diferentes modelos avaliados, destacando a performance de cada um em termos de assertividade nas respostas fornecidas a cinquenta perguntas sobre o DFT. O CATCH, com uma acurácia de 0,90, demonstrou superioridade em relação aos outros modelos avaliados, isso ocorre devido a combinação do modelo de linguagem LLaMA 3 com o ChromaDB e o RAG. Essa integração permitiu uma recuperação mais eficiente de informações contextuais e a geração de respostas mais precisas e relevantes. O modelo LLaMA 3.1, com uma acurácia de 0,85, também apresentou um bom desempenho, mas não conseguiu atingir o mesmo nível de precisão que o CATCH, devido à ausência da camada adicional de recuperação e integração de informações provida pelo ChromaDB e RAG. Já os modelos MISTRAL e PHI, com acurácias de 0,75 e 0,70, respectivamente, apresentaram desempenho significativamente inferior. Esses resultados indicam que as limitações nas técnicas de recuperação de informações e geração de respostas utilizadas por esses modelos impactaram sua capacidade de fornecer respostas precisas.

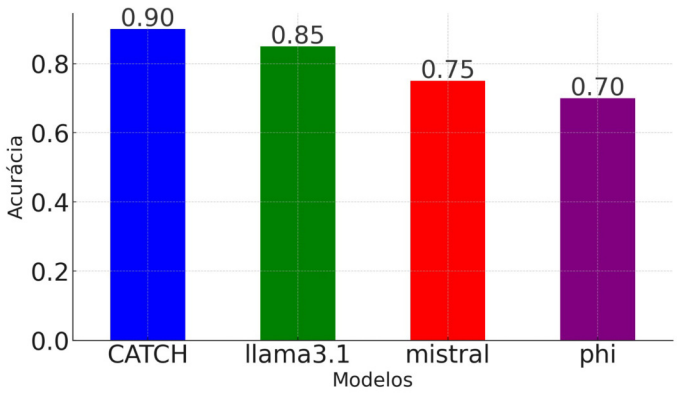


Figura 2: Impacto da acurácia no CATCH

O gráfico de tempo de resposta (Figura 3) ilustra uma análise comparativa entre os modelos avaliados, apresentando o tempo médio em segundos que cada modelo leva para processar e fornecer uma resposta. O CATCH apresentou o menor tempo de resposta, com uma média de 1,5 segundos, superando os demais modelos. O LLaMA 3.1, por sua vez, teve um tempo de resposta ligeiramente maior, de 1,7 segundos, enquanto o MISTRAL e o PHI demonstraram um desempenho inferior, com 2,5 e 3,0 segundos, respectivamente. O desempenho superior do CATCH pode ser atribuído à sua arquitetura otimizada, que utiliza o framework FastAPI para gerenciar as solicitações de forma eficiente, minimizando a latência. Essa infraestrutura, aliada à integração do ChromaDB e do SentenceTransformer, possibilita uma recuperação rápida e precisa das passagens relevantes. A combinação com o modelo RAG também contribui para a geração mais ágil de respostas contextuais, aproveitando as informações recuperadas com eficácia. Em comparação, os tempos de resposta mais altos dos outros modelos sugerem limitações em suas respectivas arquiteturas, especialmente nas fases de recuperação e geração de respostas, onde a integração entre os componentes parece ser menos eficiente. O MISTRAL e o PHI, em particular, demonstraram maior demora.

A Figura 4 ilustra a análise do grau de profundidade das respostas fornecidas por cada modelo, medido pela métrica de detalhamento. O CATCH alcançou o melhor desempenho, com uma pontuação de 0,9, superando os outros modelos avaliados. O LLaMA 3.1 obteve uma métrica de 0,85, enquanto o MISTRAL e o PHI tiveram resultados inferiores, com 0,8 e 0,7, respectivamente. Esse superioridade no CATCH pode ser explicado pela integração eficiente entre o modelo de RAG e o LLaMA3. Essa combinação permite que o sistema recupere informações contextuais de forma precisa e gere respostas ricas em conteúdo. A utilização do ChromaDB para a recuperação de passagens relevantes também desempenha um papel crucial, pois possibilita a busca rápida de informações semânticas específicas, enquanto o modelo SentenceTransformer realiza uma representação eficiente do conteúdo das perguntas, capturando suas nuances e significados com precisão. Os outros modelos apresentaram um nível de detalhamento inferior. Isso pode ser atribuído a limitações nas técnicas de recuperação e geração de respostas contextuais, o que afeta sua capacidade de fornecer respostas tão completas e informativas quanto as oferecidas pelo CATCH. No caso do PHI,

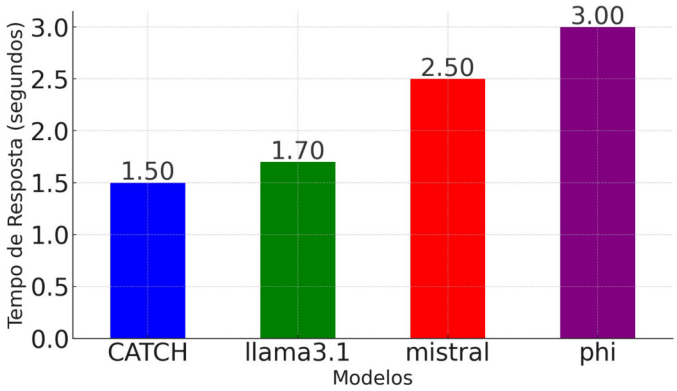


Figura 3: Impacto da tempo de resposta no CATCH

o menor detalhamento indica uma dificuldade maior em combinar e processar informações para gerar respostas com profundidade comparável.

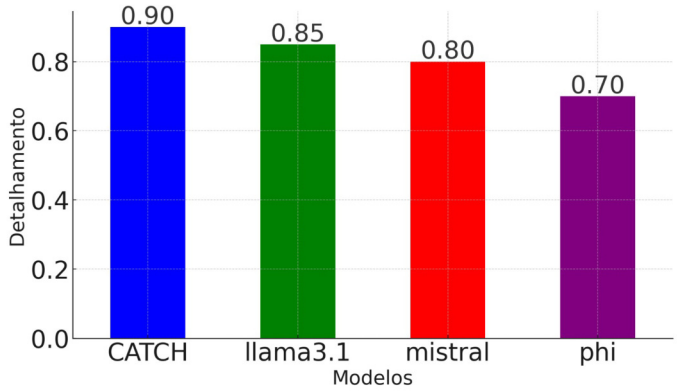


Figura 4: Impacto da detalhamento no CATCH

### 4.2 Discussão de resultados

Os resultados do CATCH mostram avanços no processo de DFT, com uma acurácia de 90%, tempo de resposta de 1,5 segundos e detalhamento de 0,9 nas respostas. Esses números indicam que o CATCH supera os modelos da literatura, tais como o MISTRAL e o PHI, em precisão, rapidez e qualidade das respostas geradas. A arquitetura do CATCH, que integra o

modelo LLaMA3 com o ChromaDB e o mecanismo de RAG, foi responsável pelo desempenho obtido. A capacidade de recuperar informações contextuais e gerar respostas detalhadas e alinhadas ao contexto permite que os gestores tomem decisões com base em dados, de forma mais eficiente no setor público.

O CATCH, no entanto, ainda apresenta limitações que precisam ser consideradas. Embora tenha sido eficaz no contexto público, sua aplicabilidade em outros setores e indústrias precisa ser avaliadas. Além disso, como a qualidade das respostas depende da interação dos usuários com o sistema, melhorias futuras podem focar na implementação de feedback contínuo e no uso de técnicas de aprendizado adaptativo para ajustar o chatbot. Com isso, o CATCH poderá se adaptar às mudanças nas demandas de trabalho e expandir seu uso para outros contextos.

## 5 CONCLUSÃO E TRABALHOS FUTUROS

O DFT no setor público enfrenta desafios complexos, especialmente na fase qualitativa, que depende de métodos manuais e subjetivos. Essas limitações frequentemente levam à imprecisão na previsão de necessidades de pessoal, aumentando a ineficiência na alocação de recursos. Este artigo apresentou o CATCH, um chatbot que visa automatizar a coleta e análise de dados qualitativos no DFT, utilizando tecnologias de IA e PLN.

Os resultados obtidos demonstram a eficácia do CATCH em melhorar tanto a precisão quanto a eficiência do processo de DFT. Com uma acurácia de 90%, um tempo médio de resposta de 1,5 segundos e um alto grau de detalhamento das respostas (0,9), o CATCH superou os modelos comparativos avaliados, tais como MISTRAL e PHI. Esses resultados reforçam o potencial da automação na transformação de processos críticos, especialmente no setor público, onde a otimização de recursos humanos é fundamental para melhorar a prestação de serviços e reduzir custos operacionais.

Além da automação de tarefas repetitivas, como responder a perguntas frequentes e realizar a coleta de dados qualitativos, o CATCH mostrou-se eficaz em proporcionar uma comunicação humanizada, ampliando a interação entre gestores e funcionários e oferecendo insights mais precisos e em tempo real para a tomada de decisões estratégicas. A combinação de tecnologias como o ChromaDB e o modelo LLaMA3 com o framework FastAPI possibilitou uma resposta rápida e relevante, assegurando um fluxo de trabalho ágil e eficiente.

Para trabalhos futuros, planeja-se o desenvolvimento de técnicas de aprendizado contínuo, que permitiriam ao CATCH adaptar-se dinamicamente às mudanças nas demandas de trabalho e nas condições do mercado. Além disso, a integração de um sistema de feedback em tempo real por parte dos usuários pode fornecer informações valiosas para ajustes finos no modelo, aprimorando ainda mais a capacidade do chatbot de entregar respostas precisas e personalizadas. Também se pretende explorar o uso do CATCH em outros setores, além do público, para ampliar sua aplicabilidade, como no gerenciamento de talentos em empresas privadas, onde o DFT também representa um desafio crítico.

## AGRADECIMENTOS

Este trabalho é apoiado pela Procuradoria Geral da Fazenda Nacional (nº PGFN 23106.148934/2019-67). Em parte pelo CNPq – Conselho Nacional de Pesquisas (Nº PQ-2 312180/2019-5 de Cibersegurança nº 465741/2014-2), em parte pelo Ministério da Economia do Brasil (N.º

DIPLA 005/2016) em parte pelo Conselho Administrativo de Defesa Econômica (Nº CADE 08700.000047/2019-14), em parte pela Advocacia Geral da União (nº AGU 697.935/2019), e em parte pela Fundação de Amparo à Pesquisa do Distrito Federal – FAPDF.

## Referências

AI, M. (2024). Mistral ai documentation. Acesso em 18 setembro 2024.

Chowdhury, S. (2016). *Estudos de otimização e melhoria de negócios na indústria upstream de petróleo e gás*.

da Silva, M. T., Pavan, D. P., Dechechi, E. C., da Silva Sampaio, V., and Panek, L. (2019). Dimensões da inovação no setor público: um estudo de caso nas prefeituras do oeste do paraná. *Brazilian Journal of Development*, 5(11):25650–25675.

de Oliveira Mendes, R. A., Oliveira, L. C. D., and Veiga, A. G. B. (2020). A viabilidade do teletrabalho na administração pública brasileira. *Brazilian Journal of Development*, 6(3):12745–12759.

Khennouche, F., Elmir, Y., Himeur, Y., Djebari, N., and Amira, A. (2024). Revolutionizing generative pre-traineds: Insights and challenges in deploying chatgpt and generative chatbots for faqs. *Expert Systems with Applications*, 246:123224.

Liu, I., Liu, F., Xiao, Y., Huang, Y., Wu, S., and Ni, S. (2024). Investigating the key success factors of chatbot-based positive psychology intervention with retrieval-and generative pre-trained transformer (gpt)-based chatbots. *International Journal of Human–Computer Interaction*, pages 1–12.

Llama (2024). Llama official website. Acesso em 18 setembro 2024.

Marques, A. L., Ferreira, L. O. G., Cavalcante, P. P. M. M., Mendes, N. C. F., da Costa, C. S., dos Santos Silvério, J. C., Neumann, C., Cruz, E. R. N., and de Souza Barbosa, E. (2022). Gestão dos custos na administração pública federal: um estudo de caso a partir das entregas. *Brazilian Journal of Development*, 8(4):25744–25768.

Microsoft (2024). Phi-2 model. Acesso em 18 setembro 2024.

Pu, H., Yang, X., Li, J., and Guo, R. (2024). Autorepo: A general framework for multimodal llm-based automated construction reporting. *Expert Systems with Applications*, 255:124601.

Rane, N., Choudhary, S., and Rane, J. (2024). A new era of automation in the construction industry: Implementing leading-edge generative artificial intelligence, such as chatgpt or bard. *Available at SSRN*.

Serrano, A., CUNHA, R. D., FRANCO, V. R., Assis, M., and SOUZA, F. (2017). Dimensionamento da força de trabalho aplicado a uma organização do poder executivo federal. *Anais do XX SEMEAD, São Paulo*.

Serrano, A. L. M., Ferreira, L. O. G., Mendes, N. C. F., Cavalcante, P. P. M. M., and Neumann, C. (2021). Modelos empíricos aplicados a análise da capacidade produtiva: Aplicações em cenários de incertezas empirical models applied to productive capacity analysis: Applications in uncertainty scenarios. *Brazilian Journal of Development*, 7(12):111940–111959.

Silva, G. W. et al. (2024). Dimensionamento da força de trabalho: desafios e possibilidades para a gestão universitária.

# Juris Syntax: Automação da Análise Jurídica Brasileira com IA Generativa e PLN

Fábio Lúcio Lopes de Mendonça[1], Bruno Justino Garcia Praciano[1], Flávio Garcia Praciano[1], Thiago Leite de Sousa[1], Elon Oliveira Albuquerque[1], José Péricles Pereira de Sousa[2]

**fabio.mendonca@redes.unb.br; bruno.justino@ieee.org; flavio.praciano@redes.unb.br; thiago.leite@redes.unb.br; elon.albuquerque@gmail.com; jose.pereira-sousa@pgfn.gov.br**

[1] Universidade de Brasília, Faculdade de Tecnologia - FT – 70910-900 – Brasília, Brasil

[2] Procuradoria Geral da Fazenda Nacional – PGAJUD/PGDAU – 70830-030 – Brasília, Brasil

**DOI: 10.17013/risti.n.pi-pf**

**Resumo:** Este artigo apresenta o Juris Syntax, um sistema inteligente para análise e contextualização de documentos jurídicos, desenvolvido com foco no Direito brasileiro. A solução integra Processamento de Linguagem Natural (PLN), modelos de similaridade semântica e Inteligência Artificial generativa, possibilitando extração de informações, fichamento, sumarização, busca por precedentes e geração de relatórios estruturados. Sua arquitetura combina backend em Python 3.11, embeddings com Sentence Transformers e integração ao modelo LLaMA 3.1 70B na Oracle Cloud, além de uma interface web responsiva. Avaliado com documentos reais da Procuradoria-Geral da Fazenda Nacional (PGFN), o Juris Syntax demonstrou redução significativa do tempo de análise e alta precisão na identificação de documentos similares, reforçando sua aplicabilidade prática. Os resultados indicam o potencial da ferramenta para otimizar fluxos jurídicos, apoiar a tomada de decisão baseada em precedentes e contribuir para a transformação digital no setor jurídico.

**Palavras-chave**: Processamento de Linguagem Natural; Inteligência Artificial Generativa; Direito; Similaridade Semântica; Oracle Cloud; Automação Jurídica**.**

***Juris Syntax: Intelligent System for Legal Analysis and Contextualization with Natural Language Processing Techniques***

***Abstract:*** *Juris Syntax is an intelligent system designed to streamline the analysis and contextualization of legal documents, with emphasis on Brazilian Law. The system integrates advanced Natural Language Processing (NLP), semantic similarity models, and generative Artificial Intelligence to support tasks such as information extraction, summarization, semantic search, and structured reporting. Its architecture combines a Python 3.11 backend, embeddings with Sentence Transformers, and integration with the LLaMA 3.1 70B model hosted on Oracle Cloud, together with a modern and responsive web interface. Tested with real cases from the Office of the National Treasury Attorney (PGFN), Juris Syntax demonstrated significant reduction in document analysis time and improved accuracy in identifying similar cases and legal precedents. The results highlight its potential to optimize legal workflows, strengthen decision-making, and contribute to the digital transformation of the legal field.*

***Keywords****: Natural Language Processing; Generative Artificial Intelligence; Law, Semantic Similarity; Oracle Cloud, Legal Automation.*

## 1. Introdução

O sistema jurídico brasileiro enfrenta uma crescente sobrecarga informacional. Apenas em 2024, havia mais de 77 milhões de processos em tramitação, muitos com duração superior a cinco anos em determinadas áreas (Conselho Nacional de Justiça, 2024). Órgãos como a Procuradoria-Geral da Fazenda Nacional (PGFN) lidam diariamente com milhares de petições, recursos e decisões, tornando inviável a análise manual diante da complexidade e do volume documental.

Nesse cenário, soluções de Inteligência Artificial (IA) e Processamento de Linguagem Natural (PLN) têm ganhado espaço em aplicações jurídicas internacionais, tais como sumarização automática, classificação de casos e geração de relatórios. Modelos de linguagem de grande porte (LLMs), tais como GPT e LLaMA, já demonstraram resultados significativos, mas ainda carecem de adaptação às especificidades do Direito brasileiro. Iniciativas nacionais como o BERTimbau e o RoBERTaLexPT representam avanços, porém, muitas vezes não contemplam integração ponta a ponta, segurança adequada ou usabilidade para operadores jurídicos (Garcia et al., 2024; R. et al., 2022).

Apesar desses avanços, ainda persistem lacunas importantes. Muitos sistemas comerciais e acadêmicos não se encontram adaptados às peculiaridades da legislação e da prática processual brasileiras, carecem de integração com fluxos de trabalho reais e, frequentemente, não oferecem garantias de privacidade e segurança adequadas para o tratamento de dados sensíveis. Além disso, poucos oferecem mecanismos de busca por similaridade semântica com foco específico no

direito brasileiro, recurso essencial para identificar precedentes relevantes mesmo quando redigidos com terminologias distintas.

O Juris Syntax surge como resposta a essas lacunas trata-se de uma plataforma modular que combina: (i) extração robusta de texto a partir de arquivos PDF, mesmo em documentos extensos ou digitalizados; (ii) geração de embeddings semânticos especializados no domínio jurídico; (iii) busca por similaridade baseada em índices vetoriais de alta performance; e (iv) sumarização contextualizada com o apoio de LLMs e pipelines de *Retrieval-Augmented Generation* (RAG). O sistema foi projetado para operar tanto em ambientes locais quanto em nuvem, com controle rigoroso sobre a confidencialidade e integridade dos dados.

Este artigo apresenta a concepção, implementação e avaliação do *Juris Syntax*, discutindo seus aspectos técnicos, metodológicos e práticos. São descritas a arquitetura do sistema, as tecnologias utilizadas, o pipeline de processamento, as estratégias de segurança, bem como um estudo de caso conduzido na PGFN. Também são analisados trabalhos relacionados nacionais e internacionais publicados nos últimos quatro anos, situando o *Juris Syntax* no estado da arte e evidenciando suas contribuições originais para a modernização da atividade jurídica no Brasil.

## 2. Comparativo com Trabalhos Relacionados

### 2.1. Análise Gráfica: Comparativo entre o Juris Syntax e Trabalhos Correlatos

O gráfico de linhas a seguir apresenta uma visualização comparativa entre o projeto Juris Syntax e os trabalhos brasileiros e estrangeiros, avaliando-os em diferentes dimensões. Essa abordagem permite identificar padrões de desempenho, evidenciar pontos fortes e apontar oportunidades de melhoria em relação ao estado da arte. Além disso, a comparação possibilita observar tendências de evolução tecnológica e metodológica no campo jurídico. Por fim, a análise facilita a compreensão do posicionamento do Juris Syntax no cenário nacional e internacional, oferecendo subsídios para decisões estratégicas de desenvolvimento. Conforme demonstrado na Figura 1 e Tabela I.

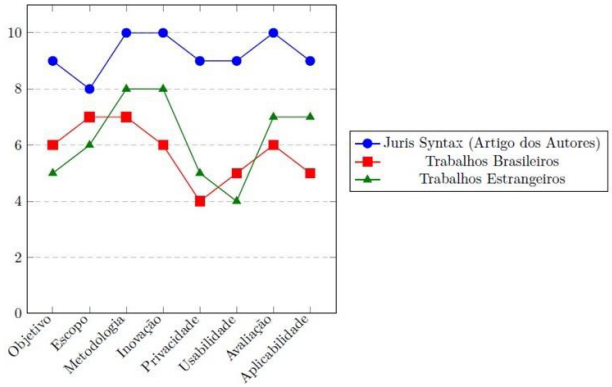


Figura 1. Comparativo: Juris Syntax vs. Outros Trabalhos

O gráfico destaca a performance superior do Juris Syntax em dimensões críticas como Metodologia e Inovação, atingindo a pontuação máxima de 10. Ele também demonstra um foco robusto em áreas negligenciadas por outros estudos, como Privacidade e Usabilidade, consolidando a sua abordagem como a mais completa e pronta para aplicação prática.

Tabela 1 – Comparativo entre o Juris Syntax e os trabalhos nacionais e internacionais analisados

| **Dimensão** | **Juris Syntax (Este Artigo)** | **Trabalhos Brasileiros (10)** | **Trabalhos Estrangeiros (10)** |
|---|---|---|---|
| *Objetivo principal* | Plataforma modular, integrada e validada para análise, fichamento, sumarização contextualizada e busca semântica de documentos jurídicos no Brasil, com foco no Direito brasileiro e potencial de expansão internacional. | Revisões e estudos de PLN jurídico (Polo et al., 2021; Silva Junior et al., 2025); aplicações institucionais como o sistema Victor/STF (Ferreira, 2022) e Athos/STJ (Figueiredo, 2022); propostas de *embeddings* jurídicos (Carmo et al., 2023); e integração inicial de IA em tribunais (Pereira et al., 2024), ainda com foco limitado em integração ponta a ponta. | Propostas exploratórias e prototípicas para sumarização, classificação e geração automática de textos (Ariai & Demartini, 2024; Bertalan & Ruiz, 2022), geralmente sem validação em ambientes reais. |
| *Escopo* | Voltado ao Direito brasileiro, mas arquitetado de forma modular para adaptação a outros ordenamentos e idiomas. | Cobertura predominantemente restrita ao contexto jurídico nacional (Campos de Carvalho et al., 2022; Melo et al., 2023), com pouca atenção a aplicações multilíngues ou internacionais. | Cobertura ampla de contextos jurídicos e sistemas legais (Silva, 2025; Anonymous, 2025), mas sem ajustes às especificidades brasileiras. |
| *Metodologia técnica* | Pipeline completo que abrange extração robusta de texto (PDF), geração de embeddings jurídicos, indexação e busca vetorial, RAG, sumarização por LLaMA 3.1, interface web responsiva e protocolos de segurança e | Utilização de modelos BERTimbau e RoBERTaLexPT (Polo et al., 2021), *embeddings* específicos (Carmo et al., 2023) e aplicações pontuais de LLMs (Pereira et al., 2024), geralmente sem integração de todas as etapas em um único sistema. | Emprego de LLMs (GPT, LLaMA, RoBERTa) (Huang & Chang, 2025; Rahman et al., 2025; Goyal et al., 2022), RAG e Knowledge Graphs (Ariai & Demartini, 2024), com forte foco em desempenho técnico, mas menos em orquestração completa de |

| | | | |
|---|---|---|---|
| | privacidade | | processos. |
| *Inovação* | Combina múltiplas funcionalidades críticas em um único sistema validado em órgão público (PGFN), priorizando escalabilidade, modularidade e confidencialidade. | Avanços conceituais e metodológicos (Barros et al., 2024; Silva Junior et al., 2025), mas com pouca evidência de aplicação prática em larga escala. | Introduzem novos métodos e *benchmarks* (Akter et al., 2025; Silva, 2025), porém com escassa transposição para uso institucional. |
| *Tecnologias usadas* | Backend em Python, Flask, Py-MuPDF, Sentence Transformers, integração com LLaMA 3.1 na Oracle Cloud, containerização com Docker e frontend em Tailwind CSS. | Modelos BERTimbau, RoBERTaLexPT e *embeddings* jurídicos treinados localmente (Carmo et al., 2023); uso de linguagens como R em análises específicas (Barros et al., 2024). | Arquiteturas baseadas em Transformers, técnicas de *prompt engineering* e infraestrutura em nuvens como AWS e Azure (Ariai & Demartini, 2024; Goyal et al., 2022). |
| *Privacidade e segurança* | Processamento local como prioridade, uso de SDKs seguros para nuvem, variáveis de ambiente para credenciais e remoção automática de arquivos após processamento. | Pouco detalhamento técnico sobre segurança e conformidade com LGPD (Ferreira, 2022; Figueiredo, 2022). | Menções pontuais ao GDPR (Rahman et al., 2025), mas sem aprofundamento técnico operacional. |
| *Usabilidade* | Interface web intuitiva e responsiva, com drag-and-drop, exportação em PDF/CSV e histórico de análises. | Interfaces pouco exploradas ou documentadas (Barros et al., 2024; Campos de Carvalho et al., 2022), com foco maior em *backend*. | Grande parte das soluções acessível apenas via API ou linha de comando (Anonymous, 2025; Goyal et al., 2022), limitando o alcance a usuários técnicos. |
| *Avaliação* | Testado com documentos reais e volumosos da PGFN, medindo eficiência, precisão e adequação ao fluxo de trabalho. | Testes com *datasets* restritos ou simulados (Barros et al., 2024; Silva Junior et al., 2025), sem avaliação operacional contínua. | Avaliações baseadas em *corpora* públicos como COLIEE (Ariai & Demartini, 2024; Akter et al., 2025), que não refletem necessariamente a realidade jurídica brasileira. |
| *Aplicabilidade* | Implementável em fluxo de trabalho de órgãos públicos, escritórios de advocacia e departamentos jurídicos, com ganho mensurável de eficiência. | Potencial teórico identificado, mas com limitações de maturidade para uso em produção (Campos de Carvalho et al., 2022; Melo et al., 2023). | Aplicabilidade ampla em contextos internacionais (Silva, 2025; Silva, 2024), mas sem compatibilidade direta com a legislação e práticas brasileiras. |

## 3. Visão Geral do Juris Syntax

O projeto *Juris Syntax* foi concebido com a finalidade de atender a um conjunto de necessidades críticas identificadas no contexto do processamento e gestão de documentos jurídicos, particularmente no cenário brasileiro, caracterizado por elevado volume documental e complexidade normativa. A proposta busca combinar eficiência operacional com precisão analítica, explorando o potencial de técnicas avançadas de Processamento de Linguagem Natural (PLN) e Inteligência Artificial (IA) generativa para otimizar fluxos de trabalho em órgãos públicos, escritórios de advocacia e departamentos jurídicos corporativos.

Entre seus objetivos centrais, destaca-se a automatização integral do processo de análise e fichamento de petições, recursos e outros documentos processuais, permitindo a extração sistemática de informações essenciais, tais como identificação das partes, pedidos formulados, fundamentos jurídicos e decisões proferidas. Complementarmente, a ferramenta visa oferecer mecanismos robustos

para a geração de resumos estruturados e relatórios detalhados, a partir de arquivos em PDF ou texto puro, favorecendo a rápida compreensão e o tratamento de grandes volumes de informação.

Outro objetivo relevante consiste na implementação de um módulo especializado para busca por similaridade semântica, capaz de identificar, com elevado grau de acurácia, documentos juridicamente correlatos. Essa funcionalidade proporciona suporte valioso à pesquisa jurisprudencial, à análise comparativa de peças processuais e à tomada de decisão baseada em precedentes.

O projeto também contempla o desenvolvimento de uma interface web moderna, responsiva e intuitiva, projetada segundo princípios de usabilidade e acessibilidade, de forma a atender perfis heterogêneos de operadores do Direito, independentemente de seu nível de proficiência tecnológica. Adicionalmente, a integração de recursos avançados de IA garante não apenas a precisão das análises, mas também a sua contextualização, adaptando os resultados ao tipo e à natureza do documento analisado. Conforme demonstrado nas Figuras 2 e 3.

No que se refere à segurança e à privacidade, o *Juris Syntax* adota uma abordagem de proteção por padrão, privilegiando o processamento local sempre que possível, aliado a mecanismos seguros de comunicação com a nuvem por meio de SDKs oficiais e variáveis de ambiente. Com isso, o sistema se posiciona como uma solução escalável, flexível e de alta confiabilidade, apta a contribuir de forma significativa para a redução de custos operacionais, a mitigação de riscos e o incremento da eficiência na gestão de informações jurídicas.

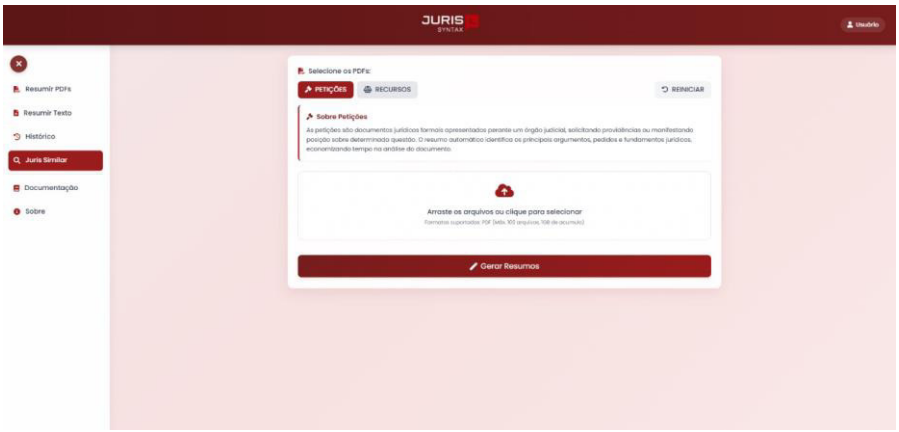


Figura 2. Juris Similar

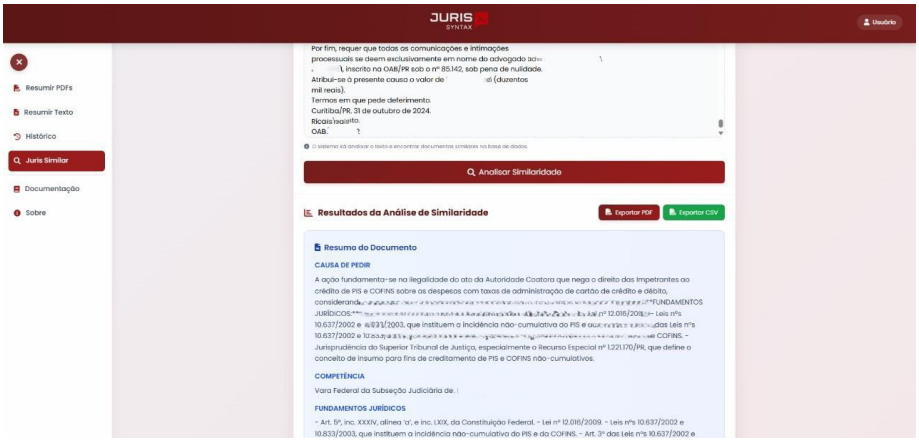


Figura 3. Análise de similaridade

## 4. Metodologia e Arquitetura

A concepção arquitetural do *Juris Syntax* fundamentase em uma abordagem modular e escalável, estruturada em três camadas principais: Backend, Frontend e Infraestrutura. Essa organização visa garantir separação de responsabilidades, flexibilidade para evolução tecnológica e manutenção facilitada, preservando a integridade do sistema em ambientes de alta demanda.

No Backend, desenvolvido integralmente em Python 3.11, concentra-se a lógica de negócio e o processamento central das funcionalidades. A aplicação é estruturada sobre o *framework* Flask 2.3.2, que fornece a base para a construção da API e para o roteamento das operações internas. A extração de conteúdo textual a partir de arquivos PDF é realizada por meio das bibliotecas *PyPDF2* e *PyMuPDF*, garantindo robustez mesmo diante de documentos extensos ou com formatação heterogênea. A geração de relatórios e a exportação em formato PDF são conduzidas pela biblioteca *FPDF2*, permitindo a produção de documentos profissionais e padronizados. Para a análise de similaridade semântica, o sistema emprega o modelo *Sentence Transformers* (*all-MiniLM-L6-v2*), responsável pela criação de *embeddings* vetoriais de alta qualidade. A integração com a *Oracle Cloud Generative AI*, utilizando o modelo Llama 3.1 70B, possibilita a sumarização contextualizada de textos jurídicos e a geração de insights adaptados ao tipo de documento processado. Operações de manipulação de dados e matrizes de *embeddings* são suportadas por *Pandas* e *NumPy*, enquanto a containerização e orquestração do ambiente são realizadas por *Docker* e *Docker Compose*. A

comunicação segura com os serviços em nuvem é viabilizada pelo *OCI SDK*, utilizando variáveis de ambiente para proteção de credenciais.

O *Frontend* é responsável pela interação direta com o usuário e foi implementado com tecnologias web modernas, incluindo HTML5 e JavaScript, proporcionando uma estrutura leve e de rápida resposta. A estilização é conduzida com *Tailwind CSS*, o que assegura responsividade, consistência visual e personalização da interface. Elementos gráficos e ícones são incorporados por meio da biblioteca *Font Awesome*, favorecendo a clareza e a atratividade da experiência visual.

A camada de Infraestrutura adota um modelo baseado em containerização, o que permite a execução tanto em ambientes locais quanto em plataformas de nuvem, como Google *Cloud Run* e Oracle *Cloud*. Essa arquitetura garante portabilidade e reprodutibilidade do ambiente de execução, reduzindo riscos de incompatibilidade. Arquivos de configuração, incluindo. env, .yml e documentação em .md, foram incorporados para simplificar o processo de implantação (*deploy*) e manutenção, além de padronizar o gerenciamento de variáveis e parâmetros operacionais.

Essa arquitetura, além de oferecer um elevado grau de modularidade e segurança, está orientada para o processamento eficiente de documentos jurídicos em larga escala, mantendo a flexibilidade necessária para futuras expansões e adaptações a novos cenários de aplicação. Conforme demonstrado na Arquitetura do sistema representado pela Figura 4.

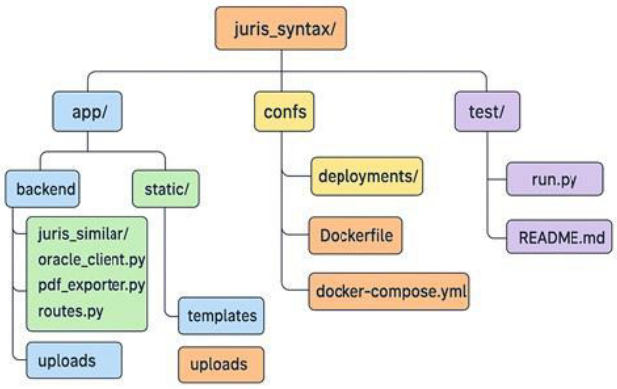


Figura 4. Arquitetura do Juris Syntax

A Figura 4 apresenta a arquitetura do projeto Juris Syntax, estruturada de forma modular para garantir organização, escalabilidade e facilidade de manutenção. No diretório raiz juris_syntax/ concentram-se os principais elementos que compõem o

sistema. O núcleo da aplicação encontra-se no diretório app/, subdividido em componentes essenciais para o funcionamento do sistema.

O subdiretório *backend/* concentra a lógica de processamento e integração, incluindo o módulo juris_similar/, responsável pela análise de similaridade textual entre documentos jurídicos; o script oracle_client.py, que realiza a integração com os serviços de inteligência artificial da Oracle Cloud AI; o módulo pdf_exporter.py, dedicado à exportação de documentos em formato PDF; e o arquivo routes.py, que define as rotas principais para comunicação entre o frontend e o backend. Outros módulos utilitários complementam a infraestrutura de processamento de dados.

A camada de apresentação é composta pelos diretórios *static*/, que armazena arquivos estáticos como CSS, JavaScript e imagens; e *templates*/, que contém os modelos HTML utilizados para renderização da interface web. O diretório uploads/ é destinado ao armazenamento temporário de arquivos enviados pelos usuários.

Em nível de configuração, o diretório confs/ centraliza parâmetros e prompts de IA utilizados pelo sistema. Já o diretório *deployments/* contém scripts e configurações voltadas à implantação do sistema em diferentes ambientes. O diretório test/ abriga os testes automatizados que asseguram a qualidade e a estabilidade do código.

No que diz respeito à infraestrutura, o arquivo requirements.txt define as dependências do projeto, enquanto *Dockerfile* e docker-compose.yml oferecem suporte à containerização e orquestração, facilitando a replicação e escalabilidade do ambiente. Por fim, o arquivo run.py atua como ponto de entrada da aplicação, e o README.md fornece documentação introdutória para desenvolvedores e usuários técnicos.

Essa organização hierárquica e modular foi concebida para otimizar o desenvolvimento, manutenção e evolução do Juris Syntax, permitindo que as equipes de pesquisa e tecnologia atuem de forma integrada e eficiente, com uma arquitetura robusta e preparada para futuras expansões.

## 5. Funcionalidades

O sistema desenvolvido incorpora um conjunto abrangente de funcionalidades voltadas à otimização do tratamento e análise de documentos jurídicos, com foco na automação e na precisão dos resultados. Entre os principais recursos, destaca-se a análise automatizada de documentos, que realiza a extração estruturada de informações relevantes, o fichamento dos conteúdos e a geração de resumos sintéticos, permitindo a rápida compreensão de grandes volumes de texto.

Adicionalmente, foi implementada uma busca por similaridade semântica, baseada em técnicas avançadas de Processamento de Linguagem Natural (PLN) e vetorização contextual, possibilitando a comparação entre documentos de forma mais precisa do que métodos tradicionais de busca por palavras-chave.

Outro diferencial é a integração com modelos de inteligência artificial generativa, que viabiliza a produção de contextualizações aprofundadas e a obtenção de insights jurídicos, contribuindo para uma análise mais qualificada. O sistema também oferece exportação de relatórios em múltiplos formatos, assegurando interoperabilidade com outras ferramentas de gestão e análise de dados, além de registrar todas as operações no histórico de análises, garantindo rastreabilidade e reprodutibilidade dos resultados. Por fim, dispõe de uma interface web moderna e responsiva, com navegação por abas, suporte a drag-and-drop e design adaptativo, assegurando usabilidade consistente em diferentes dispositivos.

## 6. Segurança e Privacidade

A segurança e a privacidade dos dados foram elementos centrais no desenvolvimento do sistema. Sempre que viável, o processamento dos documentos é realizado de forma local, minimizando a exposição de informações sensíveis a ambientes externos. Quando a integração com serviços em nuvem é necessária, esta ocorre por meio de um SDK seguro e do uso de variáveis de ambiente, garantindo a proteção das credenciais e a criptografia das comunicações. Além disso, os arquivos são armazenados apenas temporariamente durante o processamento, sendo automaticamente removidos após a conclusão das operações, em conformidade com as melhores práticas de governança de dados e privacidade.

## 7. Fluxo de Utilização do Sistema

O processo de interação com o sistema inicia-se quando o usuário acessa a interface web e seleciona o tipo de análise desejada. Em seguida, o usuário realiza o upload de arquivos no formato PDF ou insere o conteúdo manualmente em um campo de texto. O sistema processa o material submetido, executando tarefas de extração, fichamento e sumarização, bem como a busca por documentos semanticamente similares caso essa funcionalidade seja requisitada. Os resultados gerados são apresentados de forma clara na interface, com opções para visualização detalhada, exportação em múltiplos formatos (por exemplo, PDF e CSV) e compartilhamento. Adicionalmente, todo o histórico de análises é registrado com metadados relevantes (data, usuário, parâmetros da análise), permitindo rastreabilidade, reusabilidade e auditoria das operações realizadas.

*A. Diferenciais Competitivos*

O Juris Syntax reúne atributos que o posicionam de forma competitiva no domínio das soluções jurídicas automatizadas. Destaca-se pela especialização em Direito Brasileiro, com ajuste de modelos e prompts às especificidades legislativas e práticas forenses nacionais. Sua arquitetura permite o processamento concorrente de múltiplos documentos extensos, mantendo latências aceitáveis em cenários de alta carga. A flexibilidade e customização são pilares centrais: o sistema admite parametrizações para diferentes órgãos, áreas do direito ou fluxos processuais. A solução também foi projetada com escalabilidade, podendo ser executada localmente, em servidores dedicados ou em provedores de nuvem, com contêineres Docker e orquestração. Por fim, adota privacidade por padrão, priorizando processamento local, armazenamento temporário controlado e comunicação segura via SDKs e variáveis de ambiente, aspectos cruciais para dados jurídicos sensíveis.

## 8. Resultados e Discussão

Os experimentos conduzidos com um conjunto diversificado de documentos jurídicos reais evidenciaram resultados promissores quanto à eficiência e eficácia do sistema desenvolvido. Observou-se uma redução significativa no tempo médio de análise documental, o que indica uma otimização considerável dos processos que tradicionalmente demandam esforço manual e prolongado. Além disso, o sistema demonstrou elevada precisão na identificação de documentos similares, fator crucial para a recuperação e comparação de jurisprudência e outras referências jurídicas relevantes.

1. Outro aspecto relevante é a consistência e qualidade dos fichamentos e resumos gerados automaticamente, que apresentaram alinhamento com padrões técnicos esperados, proporcionando suporte confiável para profissionais do direito. A adoção do modelo Llama 3.1 70B para a contextualização de textos extensos mostrou-se particularmente eficiente, superando abordagens tradicionais de sumarização jurídica baseadas em métodos convencionais, sobretudo na manutenção da coerência e riqueza semântica dos conteúdos processados. Esses resultados reforçam a aplicabilidade do *Juris Syntax* como ferramenta capaz de auxiliar a automação inteligente em ambientes jurídicos, agregando valor por meio da integração avançada de técnicas de Processamento de Linguagem Natural (PLN) e Inteligência Artificial generativa.

Similaridade de Documentos O gráfico revela que os casos analisados apresentam um alto índice de similaridade, todos acima de 83%, indicando forte convergência temática e de fundamentos jurídicos. Isso sugere que os precedentes e decisões

correlacionadas possuem elevado potencial de influência mútua, permitindo identificar padrões consistentes na aplicação das teses. Conforme demonstrado na Figura 5.

2. Frequência de Fundamentos Legais A análise demonstra que a Constituição Federal (CF/88) e o Código Tributário Nacional (CTN) são as normas mais recorrentes, reforçando seu papel central nas disputas tributárias. A presença frequente da Lei nº 9.718/1998 e da Lei nº 11.033/2004 evidencia que a discussão se concentra especialmente na interpretação do regime de PIS/COFINS. Conforme demonstrado na Figura 6.

3. Tipos de Provas Apresentadas Notas fiscais representam a maior parcela das provas, seguidas de documentos societários e registros contábeis (EFD). Isso indica que a comprovação material das operações é o elemento central para sustentar o direito pleiteado, sendo complementada por laudos técnicos e extratos financeiros. Conforme demonstrado na Figura 7.

4. Principais Pedidos Formulados Os pedidos mais frequentes são o creditamento de PIS/COFINS e a restituição ou compensação de valores pagos indevidamente. Esses resultados apontam para um padrão nas ações judiciais, cujo foco é recuperar valores e manter o direito à dedução tributária. Conforme demonstrado na Figura 8.

5. Linha do Tempo de Normas Aplicadas A evolução normativa revela marcos regulatórios desde 1967 até 2022. Apesar da longa linha histórica, observa-se uma concentração de normas pós-2000, o que reflete o aumento de mudanças legislativas e a complexidade crescente do regime tributário aplicado ao setor de combustíveis e outras atividades econômicas. Conforme demonstrado na Figura 9.

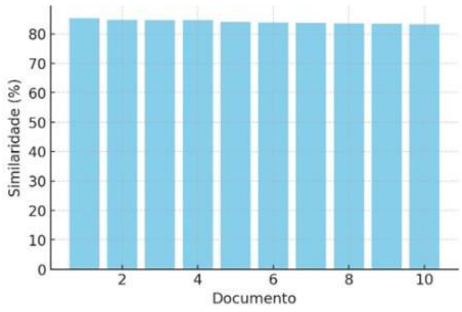


Figura 5. Similaridade de documentos

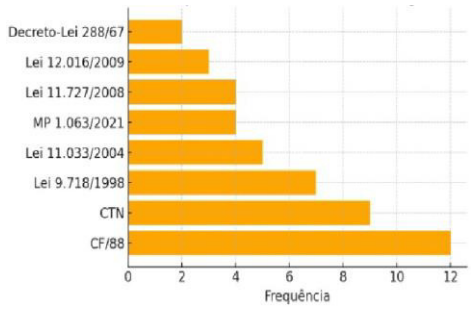


Figura 6. Fundamentos Legais

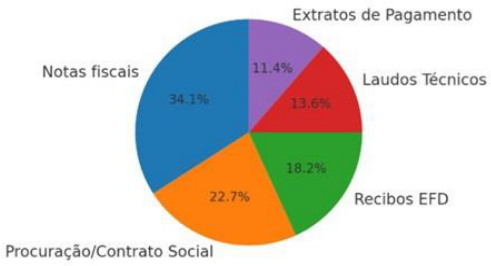


Figura 7. Provas apresentadas


Figura 8. Pedidos Formulados

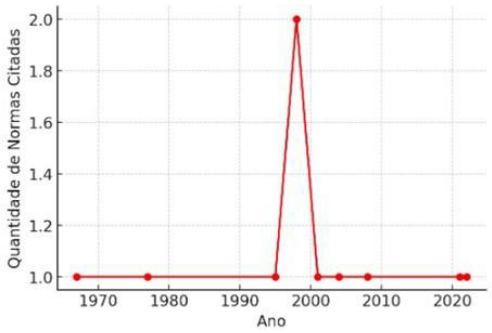


Figura 9. Normas aplicadas

## 9. Conclusão e Trabalhos Futuros

O sistema Juris Syntax representa um avanço expressivo na automação e no aprimoramento de processos jurídicos, ao incorporar tecnologias de ponta em Processamento de Linguagem Natural (PLN) e Inteligência Artificial (IA) generativa. A proposta evidenciou viabilidade técnica e impacto positivo na eficiência operacional, demonstrando elevado potencial para transformar a prática jurídica cotidiana, reduzir a carga de trabalho manual e aumentar a precisão na análise documental. As perspectivas de evolução contemplam integração com bases de jurisprudência em tempo real, proporcionando respostas mais precisas e alinhadas aos entendimentos recentes. Pretende-se também o aprimoramento dos modelos de *embedding*, ampliando a capacidade de representação semântica e fortalecendo o suporte multilíngue, possibilitando aplicação em diferentes jurisdições e contextos internacionais.

Adicionalmente, o Juris Syntax poderá incorporar módulos de autenticação e controle de acesso; integração direta com sistemas processuais eletrônicos — como

PJe e e-SAJ — promovendo interoperabilidade, extração automatizada de peças e otimização de fluxos processuais; dashboards analíticos para suporte à decisão; e processamento assíncrono para lidar com grandes volumes de documentos. Essas evoluções projetam o Juris Syntax como ferramenta robusta, versátil e estratégica na análise jurídica automatizada em escala nacional e internacional.

## Agradecimentos

Este trabalho é apoiado pela Procuradoria Geral da Fazenda Nacional (nº PGFN 23106.148934/2019-67), Em parte pelo CNPq – Conselho Nacional de Pesquisas (Nº PQ-2 312180/2019-5 de Cibersegurança nº 465741/2014-2), em parte pela Advocacia Geral da União (nº AGU 697.935/2019), e em parte pela Fundação de Apoio à Pesquisa do Distrito Federal – FAPDF.

## Referências

Akter, M., Çano, E., Weber, E., Dobler, D., & Habernal, I. (2025). A comprehensive survey on legal summarization: Challenges and future directions. arXiv preprint.

Anonymous. (2025). Analysing similarities between legal court documents using natural transformers (e.g., BERT, GPT-2, RoBERTa, LLaMA). PLOS ONE.

Ariai, F., & Demartini, G. (2024). Natural language processing for the legal domain: A survey of tasks, datasets, models, and challenges. arXiv preprint.

Barros, F. M. de C., Silva, C. D., Silva, I. R. de M., & Martins, V. S. (2024). Processamento de linguagem natural como ferramenta de suporte em documentos jurídicos: uma revisão sistemática. Revista de Casos e Consultoria, 15(1), e36701.

Bertalan, V. G. F., & Ruiz, E. E. S. (2022). Using attention methods to predict judicial outcomes. arXiv preprint.

Campos de Carvalho, T. P. F., Freitas, S. H. Z., Ribeiro, A. de S., et al. (2022). A inteligência artificial no poder judiciário brasileiro e a gestão de conflitos. Meritum, Revista de Direito da Universidade FUMEC.

Carmo, F. A. de, Serejo, F., Jacob Junior, A. F., et al. (2023). Embeddings jurídico: Representações orientadas à linguagem jurídica brasileira (pp. 188–199).

Conselho Nacional de Justiça. (2024). Justiça em números 2024. https://www.cnj.jus.br/wp-content/uploads/conteudo/arquivo/2024/08/relatorio-justica-em-numeros-2024.pdf

Ferreira, G. M. (2022). Inteligência artificial como auxiliar do poder judiciário: A experiência do sistema Victor no âmbito do STF [Monografia de graduação, Universidade Federal do Ceará].

Figueiredo, G. S. (2022). Projeto Athos: Um estudo de caso sobre a inserção do Superior Tribunal de Justiça na era da inteligência artificial [Dissertação de mestrado profissional, Universidade de Brasília].

Garcia, E., Silva, N., Siqueira, F., et al. (2024). Robertalexpt: A legal RoBERTa model pre-trained with deduplication for Portuguese. In Proceedings of PROPOR (ACL Anthology) – Legal NLP Workshop. https://aclanthology.org/2024.propor-1.38.pdf

Goyal, G., et al. (2022). Prompt-based summarization using GPT-3. arXiv preprint.

Huang, H., & Chang, C. (2025). Advancing prompt-based language models in the legal domain: Adaptive methods. Journal of Legal AI.

Melo, J. S. a. S., Nascente, V. F., & dos Santos, L. E. (2023). Discussão sobre a viabilidade técnica e jurídica para a aplicação de processamento de linguagem natural em decisões vinculantes em processos judiciais.

Pereira, J., Assumpção, A., Trecenti, J., et al. (2024). Inacia: Integrating large language models in Brazilian audit courts: Opportunities and challenges. arXiv preprint.

Polo, F. M., Mendonça, G. C. F., Parreira, K. C. J., et al. (2021). LegalNLP – Natural language processing methods for the Brazilian legal language. In Anais do XVIII Encontro Nacional de Inteligência Artificial e Computacional (SBC).

Rahman, M. M., et al. (2025). Natural language processing in legal document analysis software: A systematic review. International Journal of Innovative Research in Social Sciences (IJIRSS).

R., et al. (2022). Legal-BERTimbau: BERT for the Brazilian legal domain (model card / Hugging Face). https://huggingface.co/rufimelo/Legal-BERTimbau-base

Silva, D. de J., Oliveira, D. de, & Paes, A. (2025). Evaluating text representations for unsupervised legal semantic textual similarity in Brazilian Portuguese. Discover Data, 3, 23.

Silva, P. A. de L. A. S. (2025). Performance analysis of LLMs for abstractive summarization in legislative documents.

Silva, A. da. (2024). A legal framework for natural language processing model training in Portugal. arXiv preprint.

# ARQUITETURA PROPOSTA PARA INTEGRAÇÃO DE MIDDLEWARES IOT UTILIZANDO BLOCKCHAIN

Heitor Magalhães Vieira[1], Leonardo de Oliveira Almeida[1], Francisco L. de Caldas Filho[1], Elon de Oliveira Albuquerque[1], Clovis Neumann[1], Georges Daniel Amvame Nze[1],

*[1]Pós-graduação Profissional em Engenharia Elétrica – PPEE – Departamento de Engenharia Elétrica, Faculdade de Tecnologia, Universidade de Brasília (UnB), Brasília, Brasil, Zip Code 70910-900*

**RESUMO**

A proliferação crescente das tecnologias de Internet das Coisas (IoT) demanda soluções urgentes para aprimorar a confiabilidade na implementação, uso e auditoria desses ambientes, enquanto se preservam as características que os tornam atrativos, como custo-benefício, velocidade de comunicação, reatividade a dispositivos externos e capacidade de comunicação entre máquinas (m2m). Diante desse contexto, este artigo apresenta uma proposta de arquitetura de middleware IoT que se baseia em Fog Computing, aplicando tecnologias de blockchain e sistemas de detecção de intrusão para garantir a imutabilidade, descentralização, baixo custo e transparência do sistema. Através dessa abordagem inovadora, busca-se atender às demandas de confiança e segurança nas redes IoT, abrindo caminho para a expansão segura e confiável dessas tecnologias.

**PALAVRAS-CHAVES**

IoT, Fog Computing, Blockchain, IDS, sDN, Middleware.

## 1. INTRODUÇÃO

Os dispositivos IoT estão ganhando importância no mundo ao simplificar tarefas, abrir oportunidades de trabalho e por sua sinergia na interação com a Internet. Esses dispositivos IoT estão cada vez mais acessíveis e presentes nas atividades diárias, tanto que especialistas estimam cerca de 10 bilhões de dispositivos conectados em 2020 e cerca de 22 bilhões até 2025(What's IoT, 2021).

Os dispositivos IoT, em sua maioria, não têm foco em segurança e possuem várias vulnerabilidades, como ecossistemas de interface duvidosos, falta de controle de gerenciamento de dispositivos, serviços de rede não confiáveis e outros (Internet das Coisas (IoT): Vulnerabilidades de segurança e desafios, 2019). Essas falhas são consequências da alta complexidade na autenticação e validação da integridade de redes IoT à medida que escalam de tamanho.

Edge Computing é uma estratégia implantada em arquiteturas de IoT que fornece processamento compartilhado de dados e permite análises complexas feitas próximas a dispositivos reais, necessários com o aumento de dispositivos IoT e a demanda por respostas rápidas. Essa estratégia tem como benefício a baixa latência e largura de banda reduzida, pois não necessita enviar dados por grandes distâncias, redução de custos, facilidade de implementação e interoperabilidade de diferentes gerações (B. Varghese et al, 2016).

A Tangle é uma tecnologia de ledger distribuída com arquitetura baseada em gráficos acíclicos direcionados, que, como o Blockchain, constrói uma rede de transações independente e autogerenciada. Além disso, possui diferenciais como aumento de performance com o crescimento de dispositivos conectados à rede, ausência de taxas para verificação de transações e segurança garantida para a aplicação. Esses fatores propiciam a utilização dessa tecnologia em diversos segmentos de mercado. (S. Popov,2018)

O uso de blockchain em cenários de IoT tem sido amplamente debatido no âmbito das preocupações de segurança de aplicações e redes IoT, como blockchain pode ser usado para melhorá-los e quais aplicativos IoT são mais adequados para suportá-lo.

Uma análise profunda dos problemas de segurança encontrados em aplicações de blockchain pode ser encontrada no trabalho de Singh (S. Singh et al, 2021), que aponta riscos comuns que são impeditivos a adoção de blockchain em aplicações críticas e também debate sobre possíveis casos de uso do mundo real, concluem sobre os benefícios da blockchain e como ela pode melhorar problemas de segurança em ambientes IoT atuais.

As conclusões de Singh e colaboradores são utilizadas como referência no desenvolvimento das ideias deste trabalho. Elas fornecem informações complementares sobre a relevância das aplicações blockchain nos ambientes IoT.

Khan e Salah (M. A. Khan e K. Salah,2017) explicam minuciosamente as preocupações de segurança categorizadas para cada camada de ambientes IoT e interação de rede e como a blockchain pode ser implementada para resolver esses problemas. E são usados como referência para questões comuns que devem ser levadas em consideração ao propor o uso de blockchain dentro de redes IoT, como feito neste artigo.

Considerando tal, a arquitetura proposta nesse artigo utiliza níveis de proteção em diferentes camadas da comunicação para garantir a integridade dos dados em todos os níveis de comunicação desde a geração dos dados por dispositivos IoT até sua apresentação em aplicações na nuvem.

Minoli e Occhiogrosso (D. Minoli e B. Occhiogrosso,2018) concluem sobre os problemas de compatibilidade relacionados ao tamanho e capacidade dos dispositivos IoT e redes de blockchain, questionando quais aplicativos IoT são adequados para qual tipo de blockchain.

Essas preocupações foram consideradas na escolha do modelo operacional para a arquitetura proposta, sendo desenvolvida com arquitetura blockchain leve e rede de processamento distribuído para o ambiente IoT.

Liu et al (Y. Liu, J. Zhang, e J. Zhan,2020) propõem um aplicativo blockchain usado para resolver problemas de acesso e privacidade de controle de dados de ambientes IoT.

Comparativamente, este artigo propõe uma aplicação blockchain para melhorar a privacidade e a confiabilidade dos dados, mas com sistemas descentralizados de consenso e confiança, como o IDS, para melhorar a eficácia e a velocidade da rede.

Atlam et al (H. F. Atlam et al,2018) também propõem a viabilidade de implementações de blockchain como benéficas para um ambiente IoT, destacando os benefícios e os desafios ao fazê-lo.

Os desafios descritos como escalabilidade, armazenamento e poder de processamento são levados em consideração ao avaliar o modelo de arquitetura proposta neste trabalho, e são fortes incentivos para o uso deste tipo de aplicação blockchain em um ambiente de fog computing, uma vez que o mesmo proporciona soluções para tais problemas.

O trabalho de Bandary et al (M. Bhandary,2020) apresenta a compatibilidade de blockchain de grafos acíclicos direcionados (DAG) e ambientes IoT, transferindo com sucesso dados de dispositivos IoT de maneira segura e confiável.

Considerando as arquiteturas de fog computing aplicadas para IoT, é possível inferir, quando aplicadas, a eficácia de blockchain ledgers que são projetados para funcionar em ambientes leves com computação distribuída, como o Tangle, daí a proposta deste artigo.

Shabandri (B. Shabandri e P. Maheshwari,2019), implementa com sucesso ledgers distribuídos IOTA em ambientes IoT e conclui positivamente sobre os benefícios, compatibilidade e segurança IoT aprimorada também;
Sua implementação é utilizada como referência de possíveis aplicações de implantação para a arquitetura proposta neste artigo.

Na pesquisa de Elrawy (M. F. Elrawy et al,2018), é feita uma extensa descrição sobre sistemas de detecção de intrusão e como eles estão atualmente associados a ambientes IoT, apresentando desafios e recomendações.

Essas recomendações foram consideradas ao desenvolver a estrutura para a arquitetura proposta neste artigo, especialmente ao considerar a colocação do IDS no sistema de forma que não afete a integridade ou confidencialidade do ambiente.

Portanto, este estudo visa apresentar uma proposta de sistema de autenticação, validação de acesso e transmissão de dados para dispositivos e redes IoT, aplicando os conceitos do Blockchain, para melhorar a segurança das redes IoT garantindo a identidade dos dispositivos conectados e sua imutabilidade, descentralização da propriedade da informação, baixo custo de implementação e transparência nas transações de dados dentro da rede, explorando a sua sinergia com redes em Fog computing (M. Nofer,2014).

A solução proposta por este artigo está ancorada em pesquisas sobre segurança da informação, Edge Computing, dispositivos IoT e autenticação com Blockchain. Quanto à estruturação do trabalho, além desta introdução, ele terá os seguintes capítulos e seções: Arquitetura proposta, e Conclusões.

# 2. ARQUITETURA PROPOSTA

## 2.1 Topologia

A arquitetura proposta estabelece cada middleware IoT como um nó individual na rede, orquestrados por uma blockchain privada baseada em Tangle. Cada nó é equipado com diversos módulos para garantir a comunicação interna e externa, a conexão segura de dispositivos, um sistema de detecção de intrusão (IDS) e um ledger baseado em Tangle. A figura 1 ilustra a topologia esperada para a arquitetura proposta.

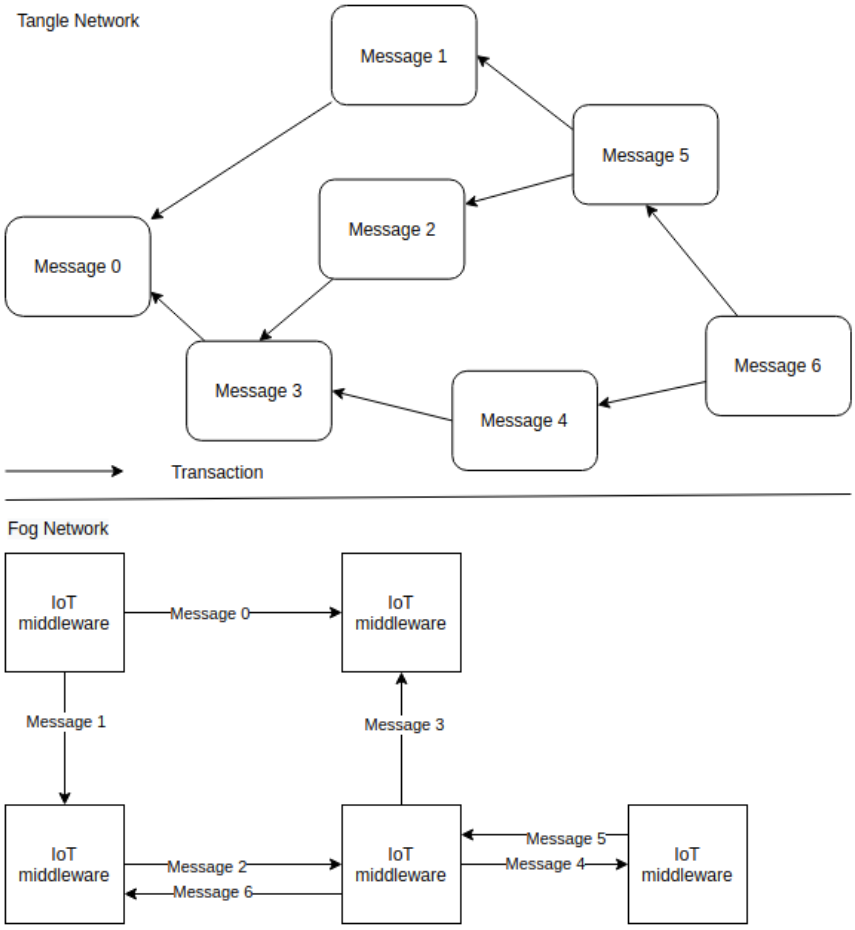


Figura 1. Topologia da arquitetura proposta

### 2.1.1 Fog computing

O fog computing é uma abstração da computação de borda que busca fornecer serviços aos dispositivos finais em uma camada mais próxima desses dispositivos. Na arquitetura proposta, utilizamos instâncias de middleware IoT que gerenciam redes de dispositivos IoT, oferecendo serviços de processamento distribuído de dados, armazenamento de memória rápida, segurança da informação e comunicação entre instâncias de middleware, dispositivos e a nuvem. As instâncias de middleware utilizadas são uma versão atualizada do middleware IoT especializado desenvolvido no laboratório UIoT (D. S. do Prado et al, 2019).

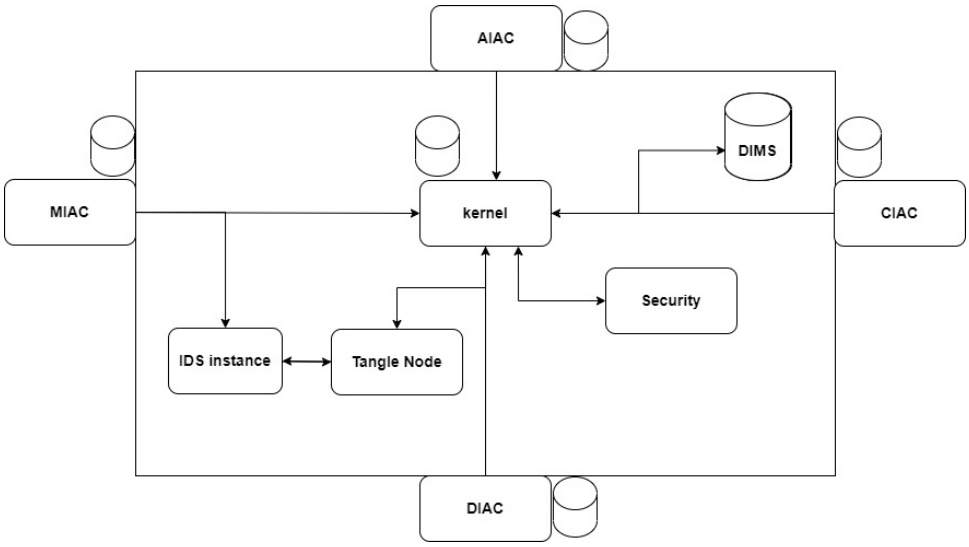


Figura 2. Arquitetura interna de serviços presente em cada instancia de middleware

### 2.1.2 Estrutura de serviços do middleware

Seguindo as estruturas apresentadas na figura 2, cada instância de middleware possui os módulos de serviços: Kernel, IDS, Ledger, Security, DIMS, e as interfaces de comunicação MIAC (Middleware Interface Access Control), CIAC ( Configuration Interface Access Control), DIAC (Device Interface Access Control), AIAC (Application Interface Access Control).

O Kernel é responsável pela orquestração dos serviços, reconhecendo a conexão adequada entre eles, garantindo a implementação correta de cada instância e afirmando políticas internas. O IDS é um módulo em desenvolvimento que integra um sistema de detecção de intrusão capaz de identificar padrões de transmissão de dados e sinalizar envios maliciosos ou suspeitos, protegendo as redes IoT que utilizam a arquitetura proposta. O módulo Security integra políticas de segurança, orquestra regras e valida o acesso aos dispositivos IoT. O módulo DIMS é o banco de dados local do middleware, responsável pelo armazenamento de dados a serem carregados para a nuvem ou outras instâncias de middleware, como informações de sensores e identificadores dos dispositivos.

As interfaces de comunicação, como CIAC, DIAC, AIAC e MIAC, controlam a transmissão de dados e a troca de informações entre as entidades da arquitetura.

### 2.1.3 Blockchain

A arquitetura proposta inclui um ledger de blockchain privada, capaz de agregar informações passadas entre os nós (instâncias de middleware) como transações criptografadas e postá-las na rede de validação em blockchain. Utilizamos a arquitetura de Grafos Acíclicos Direcionados (DAG) do blockchain Tangle para validar cada transação na rede de nós. A figura 3 exemplifica o fluxo de troca de informações entre os nós da rede.

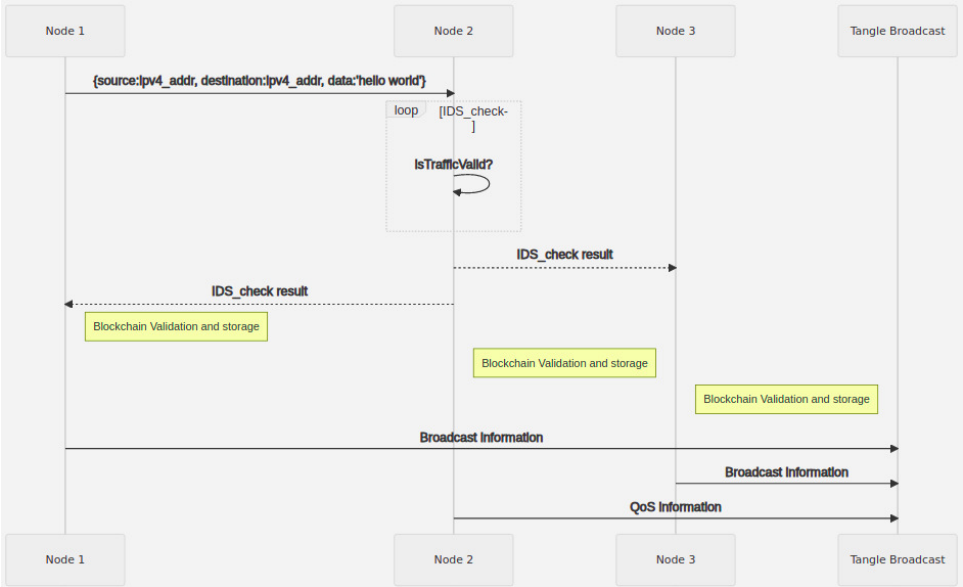


Figura 3. Fluxo de troca de informações entre nós e subsequente broadcast a rede blockchain

A abstração das trocas de informações no formato de transações permite que a própria rede de nós middleware verifique de forma segura a qualidade da informação, a integridade dos nós envolvidos na comunicação e agregue escalabilidade ao projeto.

# 3. CONCLUSÃO

A arquitetura proposta tem como principal objetivo a implementação de uma aplicação de blockchain em middleware IoT estruturados em Fog computing, buscando resolver problemas de segurança e escalabilidade dessas arquiteturas. O presente artigo parte do ciclo de inovação comandado pelo laboratório de pesquisa UIoT, que aponta vantagens na implementação de ambientes IoT tendendo a descentralização e independência, pelo compartilhamento de recursos de processamento de forma confiável e segura.

Exemplos disso são encontrados em trabalhos como a pesquisa sobre auto registro para dispositivos IoT(C. C. M. Silva,2016), que conclui sobre processos consideravelmente mais rápidos, quando os dispositivos estão cientes de seus serviços e podem se registrar de forma independente no ambiente IoT. O trabalho realizado por Bruno Dutra (B. V. Dutra et al,2019) sobre HIDS para sistemas IoT embarcados, que conclui sobre os benefícios que um sistema especializado de detecção de intrusão pode ter quando especializado para ambientes IoT. O trabalho de Patrão(R. L. Patrão, et al, 2020) concluiu como viável a implementação de ambientes IoT baseados em fog computing e machine learning, que juntos levaram ao estágio atual de melhoria, com a adição de consenso descentralizado à rede para que ela possa de maneira não supervisionada, identificar participantes confiáveis, compartilhar configurações e informações usando um conceito de publicação/assinatura, para que a própria rede possa suportar crescimento e comunicação mais rápidos (compartilhamentos de dados).

A arquitetura proposta neste trabalho ainda está no estágio de prova de conceito e é um projeto em andamento, portanto, não possui dados suficientes para testes e resultados sólidos. A versão melhorada do IDS que é compatível com os ledgers Tangle, bem como o kernel do nó middleware estão atualmente em desenvolvimento, e os testes com resultados concretos são considerados como próximos passos para este trabalho. Para garantia da implementação, deve-se validar que a instância privada da arquitetura Tangle pode suportar as implementações propostas, como a assinatura de novos nós de middleware e dispositivos IoT,

compartilhamento de informações com base em zonas de assinatura, auto quarentena e prevenção de ataques, comunicação mais rápida com base em rotas otimizadas e gerenciamento de políticas com base nas informações geradas pela rede.

# REFERENCIAS


Singh, S., Hosen, A. S. M. S. and Yoon, B. (2021) 'Blockchain Security Attacks, Challenges, and Solutions for the Future Distributed IoT Network', in Special Section on internet-of-things attacks and defenses: recent advances and challenges.

Khan, M. A. and Salah, K. (2017) 'IoT security: Review, blockchain solutions, and open challenges', in Future Generation Computer Systems.

Minoli, D. and Occhiogrosso, B. (2018) 'Blockchain mechanisms for IoT security', in Internet of Things.

Liu, Y., Zhang, J. and Zhan, J. (2020) 'Privacy protection for fog computing and the internet of things data based on blockchain', in Cluster Computing.

Atlam, H. F. et al. (2018) 'Blockchain with Internet of Things: Benefits, Challenges, and Future Directions', in I.J. Intelligent Systems and Applications.

Bhandary, M., Parmar, M. and Ambawade, D. (2020) 'A Blockchain Solution based on Directed Acyclic Graph for IoT Data Security using IoTA Tangle', in Proceedings of the Fifth International Conference on Communication and Electronics Systems (ICCES 2020).

Shabandri, B. and Maheshwari, P. (2019) 'Enhancing IoT Security and Privacy using Distributed Ledgers with IOTA and The Tangle', in 6th International Conference on Signal Processing and Integrated Networks (SPIN).

do Prado, D. S. et al. (2019) 'Design of a Fog Controller to Provide an IoT Middleware with Hierarchical Interaction Capability', in Information Technology and Systems.

Popov, S. (2018) 'The Tangle', in white paper.

Silva, C. C. M. et al. (2016) 'Proposta de auto-registro de serviços pelos dispositivos em ambientes de IoT', in XXXIV SIMPOSIO BRASILEIRO DE TELECOMUNICAÇÕES- SBrT2016.

Dutra, B. V. et al. (2019) 'HIDS by Signature for embedded devices in IoT networks', in Jornadas Nacionales de Investigacíon en Ciberseguridad (JNIC 2019), pp. 53–61.

Patrão, R. L. et al. (2020) 'Environmental building monitoring and control based on machine learning and fog computing on an IoT architecture', in INCT em Segurança Cibernética.

Nofer, M. et al. (2017) 'Blockchain', in Business and Information Systems Engineering, pp. 183–187.

Leite, L. R. C. (2019) 'Internet das Coisas (IoT): Vulnerabilidades de Segurança e Desafios', in Monografia (Curso Superior de Tecnologia em Segurança da Informação).

Varghese, B. et al. (2016) 'Challenges and Opportunities in Edge Computing', in 2016 IEEE International Conference on Smart Cloud (SmartCloud), pp. 20–26.

Oracle (2021) 'What is IoT', in Internet of Things.

McAfee (2018) 'Beware: Zombie IoT Botnets', in Security News.

Elrawy, M. F., Awad, A. I. and Hamed, H. F. A. (2018) 'Intrusion detection systems for IoT-based smart environments: a survey', in Journal of Cloud Computing: Advances, Systems and Applications.

Rahman, A. et al. (2020) 'DistBlockBuilding: A Distributed Blockchain-Based SDN-IoT Network for Smart Building Management', IEEE Access, 8, pp. 140008–140018.

# Data Pipelines Implementation and Management for Data Engineering: A Case Study Applied to the Public Sector

Elon Oliveira Albuquerque[1], Wesley Gongora de Almeida[1], Bruno Justino Garcia Praciano[1], Márcio Bastos de Medeiros[2], Fábio Lúcio Lopes de Mendonça[1], and Robson de Oliveira Albuquerque[1(✉)]

[1] Professional Post-Graduate Program in Electrical Engineering (PPEE), Department of Electrical Engineering (ENE), Faculty of Technology, University of Brasilia (UnB), Brasília 70910-900, Brazil
{fabio.mendonca,robson}@redes.unb.br
[2] Attorney General's Office (AGU), Brasília 70070-030, Brazil
marcio.medeiros@agu.gov.br

**Abstract.** This article explores the implementation and management of automated Data Pipeline (DP) in the Attorney General's Office (AGU), a public institution in Brazil, resulting in process simplification, increased efficiency, and best practices in data management used for big data. Challenges such as integrating heterogeneous data, governance, updates, and data transformations were identified. As a result of modifying the current infrastructure and processes supported by best practices that significantly contributed to scalability and operational efficiency, the results are demonstrated with better performance. The results also show a significant reduction in the process's complexity and improvements in management efficiency, operational scalability, and data governance, where information becomes available with fewer efforts from the data engineering team.

**Keywords:** Data pipelines · Data Engineering · Case Study · Big Data

## 1 Introduction

Data are one of the most important assets for a modern organization. However, integrating data from various sources to generate valuable insights requires efficient data processing, coordination of multiple processes, and management of the increasing data volume, leading to a big data approach. Achieving this practical goal is possible only through well-designed DP solutions, supported by effectively implemented IT services. All these features enable automated task execution, minimize manual intervention, and meet established requirements and deadlines to have the information available. Despite the growing popularity of data pipelines, significant challenges and a gap persist in the literature regarding the implementation and real-world applicability of these solutions in large-scale data processing environments, considering the public sector view.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Rocha et al. (Eds.): ICITS 2025, LNNS 1448, pp. 212–222, 2025.
https://doi.org/10.1007/978-3-031-93106-2_18

The difficulty in data integration presents a challenge for both the private and public sectors; however, in the public sector, the challenge becomes more complex due to the bureaucracy involved in governmental processes and the widespread occurrence of data silos. This situation was observed in the Attorney General's Office (AGU), which is responsible, among other duties, for defending the Brazilian government in judicial cases. To effectively provide its services, the AGU is equipped with a vast array of data sets from various ministries and legislative and judicial branches in Brazil. AGU is responsible for integrating these data sources to extract value. Historically, this integration process was performed manually, involving the execution of labor-intensive routines, which often led to delays and increased susceptibility to errors. This study examines the AGU's complex data integration environment to provide an automated solution for replacing manual processing import routines with automated data pipelines.

The contributions of this study include the demonstration of workflow orchestration, the identification of challenges and best practices, and an evaluation of the applicability of such solutions in real-world scenarios. A comprehensive perspective is provided, encompassing aspects from architectural design to environment management, expanding on existing concepts, and shedding light on theoretical and practical issues often underexplored in the current literature about data pipelines to the public sector. This work gives practical examples of implementation in the field of Data Engineering (DE) and offers valuable insights into the application of data pipeline solutions.

The structure of this paper is as follows:. After this brief introduction, Sect. 2 presents the background and related work. Section 3 describes the case study and its requirements. Section 4 presents the implementation details. Section 5 presents the results and discussions, and Sect. 6 concludes with final considerations.

## 2 Background and Related Works

In this section, we introduce the fundamental concepts that support our work and then review related works. In theory, DP is defined as a Directed Acyclic Graph (DAG), consisting of a sequence of nodes [5]. These nodes perform operations on data. The graph begins with at least one source node that generates the data and ends with at least one destination node that receives the processed data. In practice, a data pipeline generally refers to software that automates data manipulation, moving it from various source systems to specific destinations [3].

On the other hand, data engineering refers to the process of handling raw data to generate high-quality and consistent information [13]. In data engineering, several factors influence the data lifecycle within an organization, as shown in Fig. 1 [13]. In addition, data governance should be considered essential in the public sector to ensure legal compliance and avoid sanctions. It involves ensuring compliance with legal requirements, privacy regulations, and data security standards [5].

In studies such as [14], the application of cloud-based data pipelines has been explored, but their applicability in public institutions remains under researched.

Our work addresses this gap by focusing on a large-scale scenario with specific data governance requirements. Recent studies, including [4,5], and [3], identify deficiencies in data quality and challenges associated with complex pipelines and large volumes of data. Our work aims to address these gaps by offering practical solutions tailored to the public sector context. Although there are case studies, such as [7] and [12], and analyses of general pipeline challenges, such as [11], many issues still require further investigation. Additionally, practical analyses, such as [17] and [15], as well as comparative studies of tools, such as [2,10,16], and [9], are useful, but our research applies these tools in a real environment and delves deeper into assessing operational effectiveness.

Our study aims to fill this gap by providing a detailed analysis of the implementation of data pipeline technologies in the Attorney General's Office, a Brazilian public institution, offering significant insights and advancing the understanding of the topic through a practical case study.

## 3 The Case Study

In this section, we present the case study we conducted. We adopted the Case Study methodology described by [6], a qualitative research approach aimed at investigating contemporary phenomena within real-life contexts.

### 3.1 Scenario Description: The Attorney General's Office of Brazil

The case study was conducted at AGU, a key institution in the Brazilian justice system, responsible for both judicial and extrajudicial representation of the Brazilian Government and involved in financial restitution and asset recovery through the Asset Recovery Laboratory (LABRA). Initially, LABRA faced significant challenges in processing large volumes of sensitive data about Brazilian citizens and companies. Data processing was managed manually and in a decentralized manner, using a variety of heterogeneous technologies, such as routines Java, Python, Pentaho, and SQL routines. The absence of centralized orchestration for data pipelines led to poor data governance, frequent delays, inconsistencies, and rework.

Considering the above situation, the institution faced significant challenges with its data, directly impacting critical activities necessary for populating decision support systems and dashboards and generating insights. The initial diagnostic assessment identified vital issues such as missing data, inconsistencies, and processing delays; complexity in the implementation and management of data processing; manual execution of routines without tracking dependencies; lack of historical records of executions; and difficulties in recruiting and training specialists in heterogeneous technologies.

These problems are exacerbated in government institutions. According to the review presented in [8], the five critical challenges in implementing innovations are addressed and contextualized in our work as follows: a) **Cultural Change -** Resistance to change is generally more pronounced; b) **Collaboration with External Agents -** There is greater difficulty in cooperation between the

public sector and external agents; c) **Development of Skills and Knowledge -** There are significant challenges in recruiting, training, and retaining Data Engineering in the public sector; d) **Bureaucracy and Existing Processes -** Rigid processes, predefined technologies and tools, and fixed financial budgets influence the implementation of Data Engineering projects; and e) **Support from Senior Management -** Decision-making is often centralized among senior management and executives and influenced by public policies and external factors.

### 3.2 Methods

The analysis provided in this article focuses on the impact of holistic data solutions on the public sector, particularly in terms of operational efficiency, data governance, and the formulation of public policies supported by available information. The process used to create and implement the pipelines to AGU was based on the data engineering lifecycle presented in Fig. 1.

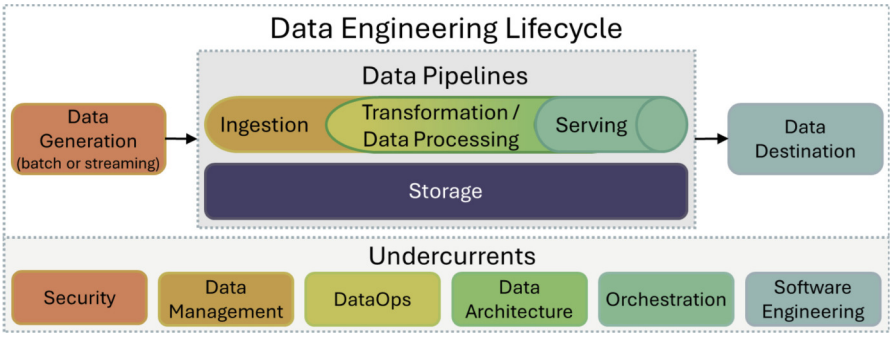


**Fig. 1.** Data Engineering Lifecycle with a Focus on Data Pipelines: Modified from [13].

For the development of the solution, it was essential to identify the needs and objectives of the institution. The proposed approach must balance budget feasibility, deadlines, and legal and regulatory requirements in the public sector context. The process for defining the solution was conducted using the following methods: 1) **Interviews with institution managers:** Conducted to gain a deep understanding of the organization's needs and expectations; 2) **Systematic meetings with IT specialist teams:** Held to assess technical capabilities and existing solutions; 3) **Review of the architecture and technological solutions in use:** Analyze the technologies and tools currently in the production environment; and 4) **Scientific and academic review:** Search the relevant literature to ensure theoretical grounding and the adoption of best practices.

### 3.3 Proposed Solution for Data Pipelines Architecture

This section presents an overview of the proposed solution for a data pipeline architecture. It consists of the following modular components on a cloud-based infrastructure, as illustrated in Fig. 2.

Each of these items serves the following purposes: a) *User Interface:* Graphical interface for interaction with the environment; b) *WebServer:* Responsible for receiving and processing requests related to the execution and management of workflows via an HTTP server; c) *Code Repository:* Storage of coded routines in a standardized and organized manner, classified and ready for execution by DAGs; d) *Scheduler/Executor:* Monitor Codes/DAGs to decide what needs to be added to the execution queue based on dependencies and schedules. Sending to the *Executor* for orchestrate task execution e manage parallelism; e) *Workers:* Receive task flows from the Executor for multiple executions; and f) *Metadata DB:* Stores all information about the execution states.

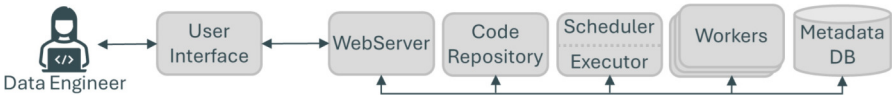


**Fig. 2.** Overview of the Proposed Solution for Data Pipelines Architecture.

## 4 Implementation

The next sub-subsections describe in detail the approach to selecting the technology stack and the implementation of pipelines.

### 4.1 Technology Stack Selection

Selecting a suitable technology stack for current and future needs is crucial in any practical project. Our approach consists of three steps where: 1) Pre-select the tools in order to identify market tools and solutions applicable to the context and requirements of our case study; 2) Technical review of the pre-selected tools considering review supported by academic and technical analyses, was conducted on the pre-selected tools; 3) Validate with the government institution with the objective to evaluate and jointly validate with the government institution the advantages and disadvantages of each tool for the final decision. All this is based accordingly to [2,9,10,15–17].

### 4.2 Orchestration with *Apache Airflow*

The choice of *Apache Airflow* was based on its modular and scalable architecture for orchestrating workflows through DAGs [17]. The use of *Python* facilitates the recruitment of specialists, and as an open-source tool, it meets the budgetary constraints common in public institutions. The architecture implemented in the cloud infrastructure is shown in Fig. 3. Six *Docker Containers* are used to isolate different workloads. Additionally, *Airflow* requires an *Broker* for communication and message (tasks) queuing, which are then processed by the *Workers*.

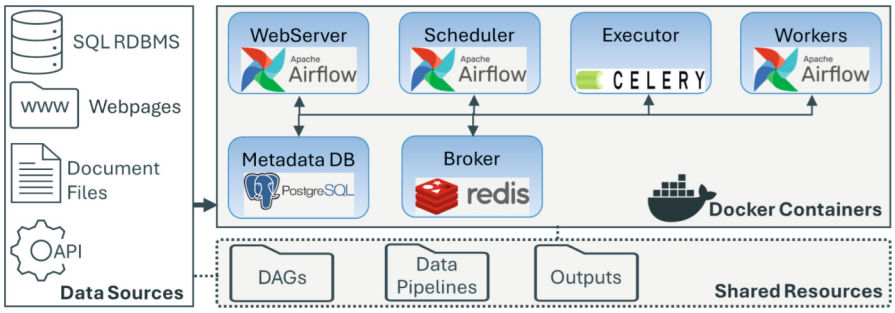


**Fig. 3.** Data Pipeline Orchestration Architecture Overview with *Apache Airflow*.

It also provides additional benefits by exploring a tool of growing popularity, whose implementation and application in real-world, large-scale data processing environments remain underexplored. This gap is particularly relevant for the public sector, where effective use of data orchestration tools can significantly enhance operational efficiency.

**Data Pipelines with *Python* Batch Processing:** Since coding DAGs with the *PythonOperator* was problematic for maintenance, testing, and logging, we adopted a new solution: all routines were converted to *Python* and executed using batch scheduling with the *BashOperator*. The pipeline implementation was phased: first, code from other languages was converted to *Python* for testing and peer review. After validation, we automated the full workflow, including the scheduler and task dependency management through DAGs. Finally, notification and report generation routines were implemented. The pipelines can be classified into two types based on their nature and purpose: 1) *ETL/ELT and Auditing Processes:* Loading of large volumes of external data, and access log auditing for sensitive datas (8 scheduled DAGs); and 2) *Machine Learning (ML):* Address deduplication, and uncovering family connections (3 scheduled DAGs).

## 5 Results and Discussions

Figure 4 presents the centralized management of DPs in the public sector. This web interface consolidates various functionalities into a single access point, allowing users to monitor task execution and review detailed historical data. It provides critical insights into the status and performance of DPs in real time. Feedback from stakeholders has been positive, highlighting that the efficiency and quality of DPs have increased by eliminating manual and repetitive tasks, enabling the team to focus on more relevant activities. Furthermore, the cloud infrastructure utilizing *Docker Containers* has provided scalability, security, and effective management. However, despite the improved management, implementation challenges have been identified, which are discussed in the following sections.

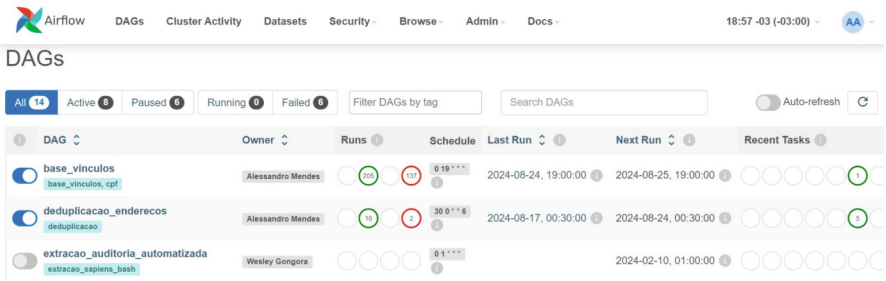


**Fig. 4.** Data Pipeline Management with *Apache Airflow*.

### 5.1 Evaluating Performance and Scalability

The mentioned benefits resulted, objectively, in a reduction of the development and implementation time of DP, which in some cases exceeded 50 days. There is no definitive answer in the literature regarding the average time required for this activity, as it can vary widely based on factors such as complexity and the experience of stakeholders.

**Table 1.** Comparative Analysis of the Implementation of DP in the Initial Scenario versus the Proposed Approach

| Scenarios | Infrastructure<br>Xeon Silver 4114 CPU @ 2.20 GHz,<br>4 vCPUs, 16RAM | ETL/ELT and Auditing<br>Processes (8 batch scheduling) | | Machine Learning<br>(3 batch scheduling) | |
|---|---|---|---|---|---|
| | | Average time<br>from coding<br>to deploy | Average<br>execution time | Average time<br>from coding<br>to deploy | Average<br>execution time |
| Inicial Scenario | Java code,<br>Windows Server 2019,<br>Windows Task Scheduler with<br>Manual Oversight | 30 days | 2 h 25 min<br>(145 min) | – | – |
| Proposed Approach | Python code,<br>Docker Containers in Debian Linux,<br>Apache Airflow Scheduler | 15 days | 3 h 40 min<br>(230 min)<br>increase of 59% | 60 days | 8h20min |

Table 1 presents a comparative analysis between the initial scenario and the proposed approach. The first set showed an average time reduction of approximately 50%, decreasing from 30 days to 15 days. The results highlighted that computational environments with appropriate technology stacks, standardization, and automation of steps are essential for reducing the development and implementation time of DPs, thereby minimizing errors and rework.

In the second category, related to Machine Learning DPs, no DPs were implemented, as the team faced time constraints due to workload, along with the lack of an adequate environment for the development and implementation of these pipelines. Our approach resulted in an average time of 60 days until deployment.

Performance was evaluated through the average execution time, with SQL queries remaining unchanged in both scenarios. The result revealed an average

increase of 59% in execution time. This value is considered satisfactory, especially given the use of multithreading optimization techniques in *Python*, as *Java* implementations generally exhibit superior performance.

### 5.2 Highlighting Challenges and Best Practices

Although we adopted best practices [1,15], and a consolidated technology stack, we faced challenges that tested our technological choices.

The following topics address the main challenges we found with data pipelines: a) **Security:** Maintain data integrity and security [13]; Policies and procedures to comply with government regulations and institutional requirements for access controls [13]; Protection of sensitive personal data; b) **Data Management:** Assurance of Data Governance [5] and institutional Compliance; Data Quality [11] [12] [16] [13]: Missing data files, Operational erros, Logical changes [11]; Query requirements, Budget constraints, and Limited infrastructure [13]; Increase responsibilities of Data Pipeline manager [11]; Dependency on other organizations [11]; Quality Assurance and Maintenance [5,17]: metadata, monitoring [14,15] and logging; c) **DataOps:** Serving environment [5] [16]: performance, scalability [11,15]; Tools & technology [5]: Appropriatenes, Compatibility, Debugging, Capabilities, Usability; Upgrades to IT infrastructure; Lack of standardized processes; d) **Data Architecture:** Integrating new data sources [11] and External services [15] and systems [11,17]; Data Sharing between tasks [17]; Ineffective rules and policies for data use; A company without a solid Data; Architecture will not remain relevant for long; Trade-off between data pipeline complexity and robustness [11]; e) **Orchestration**: Building interdependent task flows, Correct execution of scheduling and triggers [15,17]; Ensuring dependencies are met [17]; Alarms [11] and notifications; Task definition and configuration [10,15,17]; and f) **Software Engineering**: Requirements specifications; Expertise e technical knowledge [5]; Code quality assurance; Testing scope [5]; Testing depth and space; Rapid advancement of technology; Changing requirements: Software requirements are often fluid due to internal and external political factors; Lack of communication between teams [5,11]; DataOps-DevOps Collaboration [11].

We adopted an underexplored approach by considering DP in the context of the data engineering lifecycle. Government institutions, often rigid, need to adopt well-established approaches. Data Engineering emerges as crucial to overcoming these barriers. Figure 1 shows the six key factors for DE. This approach can be significant, as similar challenges may arise at different stages of DP.

In **Data Ingestion** phase, for example, integration incompatibilities were identified in *Airflow* when using advanced *Oracle RDBMS* instructions and *Pentaho*. These issues were reclassified in our research as critical factors in DP: a) **Data Management:** *Data Governance* by integrating new data sources, and *Data Quality* due to operational errors and query requirements; b) **DataOps:** Serving environment due to performance and tools & technology; and c) **Software Engineering:** Technical knowledge, Limited testing process.

The same approach was implemented across the remaining stages of the DP. The transformation was affected by the loss of history when renaming files/DAGs and by an excessively complex testing environment that was poorly compatible with legal and internal control requirements.

Preliminary assessments indicate that *Python*-converted routines maintained performance equivalence, and additional adjustments effectively handled increasing data volumes. Best practices with the DP included: code simplification and standardization; more complex SQL; and testing outside of *Airflow*. Regarding infrastructure, we adopted continuous monitoring of containers and optimized DB libraries decoupled from *Airflow*.

Our strategies to overcome these challenges of technological changes in the public sector have been grouped into the following obstacles to implementation: *a) Cultural Change:* Conducting workshops and training sessions highlighting practical benefits; Showcasing success stories, test environments, and pilot projects; Demonstrating the practical benefits of adopting proposed new Technologies; *b) Collaboration with External Agents:* Regular communication and meetings for follow-up, providing feedbacks; Building strong professional relationships; Managing expectations regarding the reality of available resources; Clearly defining roles and responsibilities; *c) Development of Skills and Knowledge:* Training actions through workshops, courses, and ongoing education; *d) Bureaucracy and Existing Processes:* Flexibility and adaptation of activities during implementation; Close collaboration with managers to review and adapt existing procedures; Conducting pilot projects and proof of concepts before full deployment; Facilitate the integration of IT with mature and adaptable technologies; *e) Support from Senior Management:* Continuously review project objectives with institutional goals; Ensuring effective communication; Highlighting results and successes.

### 5.3 Exploring Real-World Applications

Although this study was conducted within a Brazilian government institution, the proposed solution has potential applications across various sectors. For instance, in healthcare, it could automate clinical data processing, while in the financial sector, it could efficiently manage large volumes of transactions. Adaptations to the proposed data pipeline architecture can be achieved by replicating the steps outlined in previous sections, which detail the study methodology. Finally, the implementation can be evaluated based on the chosen technology stack. While specific customizations will likely be required for different contexts, adopting similar solutions can significantly improve operational efficiency and data management in public and private sectors.

## 6 Conclusion

The implementation of DP clearly demonstrated the benefits of automation and data orchestration for the public sector. The use of *Apache Airflow* significantly reduced operational complexity and improved efficiency and management of DE activities. These results highlight the importance of appropriate approaches and

tools in environments with high processing demands and implementation obstacles. Challenges, such as technological adaptations and strategies for managing cultural changes, were also revealed. The lessons learned and best practices identified can serve as a reference for other governmental institutions facing similar challenges in adopting DE solutions.

Beyond the results, our work identified areas for future research. One is expanding the implementation to other public sectors to explore different contexts, while another is advanced monitoring automation using machine learning techniques to predict failures and bottlenecks, ensuring greater resilience. Finally, inter-institutional collaboration through shared DP among various public entities is a relevant proposal that could foster a broader and more efficient ecosystem for data sharing and orchestration.

**Acknowledgments.** The authors would like to thank the technical and computational support provided by the LATITUDE Laboratory of the University of Brasília, TED 01/2019 of the Attorney General's Office (Grant AGU 697.935/2019), TED 01/2021 of the National Secretariat for Social Assistance SNAS/DGSUAS/CGRS, TED 01/2021 of the General Coordination of Information Technology (CGTI) of the Attorney General's Office of the National Treasury PGFN, the SISTER City Project Secure and Real-Time Intelligent Systems for Smart Cities (Grant 625/2022), the Project "Control and Unification System for the Federal District Government Sispro-DF" (Grant 497/2023), the Dean's Office for Research and Innovation DPI/UnB, and FAP/DF.

## References

1. Apache Airflow. Documentation (2024). https://airflow.apache.org/docs/. Accessed 01 June 2024
2. Bergmann, R., Theusch, F., Heisterkamp, P., Grigoryan, N.: Comparative analysis of open-source ml pipeline orchestration platforms (2024)
3. Biswas, S., Wardat, M., Rajan, H.: The art and practice of data science pipelines: a comprehensive study of data science pipelines in theory, in-the-small, and in-the-large. In: Proceedings of the 44th International Conference on Software Engineering, pp. 2091–2103 (2022)
4. Blohm, I., Jarvis, E.: Big data and analytics with driving data: implementation and analysis of data pipeline and data processing resources (2023)
5. Foidl, H., Golendukhina, V., Ramler, R., Felderer, M.: Data pipeline quality: influencing factors, root causes of data-related issues, and processing problem areas for developers. J. Syst. Softw. **207**, 111855 (2024)
6. Leavy, P.: Research Design: Quantitative, Qualitative, Mixed Methods, Arts-Based, and Community-Based Participatory Research Approaches. Guilford Publications (2022)
7. Manowon, S., Boonma, P.: Development of batch data pipeline system for flight delay prediction (2023)
8. Fontana, R.M., Marczak, S.: Characteristics and challenges of agile software development adoption in Brazilian government. J. Technol. Manage. Innov. **15**(2), 3–10 (2020)

9. Matskin, M., et al.: A survey of big data pipeline orchestration tools from the perspective of the datacloud project. In: DAMDID/RCDL (Supplementary Proceedings), pp. 63–78 (2021)
10. Mbata, A., Sripada, Y., Zhong, M.: A survey of pipeline tools for data engineering. arXiv preprint arXiv:2406.08335 (2024)
11. Munappy, A.R., Bosch, J., Olsson, H.H.: Data pipeline management in practice: challenges and opportunities. In: Product-Focused Software Process Improvement: 21st International Conference, PROFES 2020, Turin, Italy, 25–27 November 2020, Proceedings 21, pp. 168–184. Springer (2020)
12. Nazabal, A., Williams, C.K.I., Colavizza, G., Smith, C.R., Williams, A.: Data engineering for data analytics: a classification of the issues, and case studies. arXiv preprint arXiv:2004.12929 (2020)
13. Reis, J., Housley, M.: Fundamentals of Data Engineering. O'Reilly Media, Inc. (2022)
14. Shukla, S.: Developing pragmatic data pipelines using apache airflow on google cloud platform. Int. J. Comput. Sci. Eng. **10**(8), 1–8 (2022)
15. Stilinski, D., Potter, K.: Building a scalable and robust data extraction pipeline with apache airflow and cloud platforms. Technical report, EasyChair (2024)
16. Talia, D., Trunfio, P.: Programming tools for high-performance data analysis. In: Proceedings of the 33rd International Symposium on High-Performance Parallel and Distributed Computing, pp. 352–355 (2024)
17. Yasmin, J., Wang, J., Tian, Y., Adams, B.: An empirical study of developers' challenges in implementing workflows as code: a case study on apache airflow. arXiv preprint arXiv:2406.00180 (2024)