UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Vitor Magalhães de Souza

Observatório de Cibercrimes: desenvolvimento do backend e classificação automática de processos usando LLMs

Uberlândia, Brasil

2025

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Vitor Magalhães de Souza

Observatório de Cibercrimes: desenvolvimento do backend e classificação automática de processos usando LLMs

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Sistemas de Informação.

Orientador: Rodrigo Sanches Miani

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Bacharelado em Sistemas de Informação

Uberlândia, Brasil

2025

Vitor Magalhães de Souza

Observatório de Cibercrimes: desenvolvimento do backend e classificação automática de processos usando LLMs

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Sistemas de Informação.

Trabalho aprovado. Uberlândia, Brasil, 23 de setembro de 2025:

Rodrigo Sanches Miani
Orientador

Pedro Franklin Cardoso Silva
Professor

Paulo Henrique Ribeiro Gabriel
Professor

Uberlândia, Brasil
2025

Agradecimentos

A finalização e aprovação deste trabalho só foram possíveis graças a Deus, que me deu a vida e tornou possível esse momento. Gostaria de agradecer também:

À minha esposa, Nádia Ribeiro Bastos Villela Magalhães, por me amar, apoiar, incentivar e me dar suporte cuidando do nosso filho para que eu pudesse focar neste trabalho.

Aos meus pais, Jacques Douglas de Souza e Ionice Ferreira Magalhães, por cuidarem de mim, financiarem os meus estudos e me incentivarem a estudar desde criança.

Aos meus irmãos, Amanda Magalhães de Souza e Douglas Magalhães de Souza, por serem exemplos de dedicação e disciplina na minha vida.

Ao meu orientador, Rodrigo Sanches Miani, por me orientar da melhor forma possível e com paciência, e por me incentivar a finalizar este trabalho.

À Universidade Federal de Uberlândia (UFU), por todo o suporte fornecido.

Resumo

Este trabalho apresenta o desenvolvimento da camada de serviço (*backend*) do Observatório de Cibercrimes, uma plataforma projetada para a coleta, organização, classificação automatizada e disponibilização de dados de processos judiciais de crimes cibernéticos no Brasil. O problema central abordado é a dificuldade de acesso a informações judiciais de forma centralizada e estruturada, o que limita a análise de tendências e a compreensão do panorama dos delitos digitais. A solução proposta consiste em um sistema desenvolvido em Java com o *framework* Spring Boot, que automatiza a extração de dados do Tribunal de Justiça do Estado de São Paulo (TJSP) e os armazena em um banco de dados PostgreSQL. O principal diferencial do sistema é o módulo de enriquecimento de dados, que utiliza um Modelo de Linguagem Grande (LLM), por meio da API do Google Gemini, para classificar automaticamente os processos com base em uma taxonomia acadêmica. Os resultados demonstram que a plataforma é capaz de processar milhares de registros e disponibilizar os dados classificados de forma estruturada via uma API REST, criando uma fonte de conhecimento para pesquisadores, profissionais do direito e formuladores de políticas públicas.

Palavras-chave: crimes cibernéticos, sistema de informação, classificação, banco de dados, API REST.

Lista de ilustrações

Figura 1 – Painel de mapa do Observatório Nacional de Blockchain 18
Figura 2 – Interface Gráfica do Cuidaidoso 19
Figura 3 – Diagrama de Arquitetura do Sistema 20
Figura 4 – Diagrama de Entidade-Relacionamento 22

Lista de tabelas

Tabela 1 – Sistema Proposto de Classificação de Crimes Cibernéticos 12
Tabela 2 – Endpoints da API do sistema 29
Tabela 3 – Distribuição dos Processos Classificados por Categoria (Nível 1) 30
Tabela 4 – Detalhamento dos Processos da Categoria B (Nível 2) 30

Lista de abreviaturas e siglas

DER	Diagrama de Entidade-Relacionamento
URL	Uniform Resource Locator
HTTP	Hypertext Transfer Protocol
TJSP	Tribunal de Justiça do Estado de São Paulo
API	Application Programming Interface
ACID	Atomic, Consistent, Isolated and Durable
JSON	JavaScript Object Notation
IA	Inteligência Artificial
SQL	Structured Query Language
DSL	Domain-Specific Language
JVM	Máquina Virtual Java
NIST	National Institute of Standards and Technology
RNP	Rede Nacional de Ensino e Pesquisa
SGBD	Sistema de Gerenciamento de Banco de Dados
SSPDS	Secretaria da Segurança Pública e Defesa Social
UFU	Universidade Federal de Uberlândia
LLM	Large Language Model

Sumário

1 INTRODUÇÃO 9

2 FUNDAMENTAÇÃO TEÓRICA 11
2.1 Crime cibernético 11
2.2 Tecnologias e ferramentas 13
2.2.1 Java 13
2.2.2 Spring Framework 13
2.2.3 PostgreSQL 13
2.2.4 Gradle 14
2.2.5 Arquitetura Cliente-Servidor 14
2.2.6 Protocolo HTTP e APIs REST 14
2.2.7 Swagger 15
2.2.8 Flyway 15
2.2.9 Git e GitHub 15
2.2.10 Google Gemini 15

3 TRABALHOS RELACIONADOS 16
3.1 Cenário Jurídico e Taxonômico do Cibercrime 16
3.2 Sistemas Observatório como Ferramentas de Análise e Monitoramento 17

4 DESENVOLVIMENTO 20
4.1 Escolha das tecnologias 21
4.2 Modelagem do Banco de Dados 22
4.3 Implementação do Backend 24
4.3.1 Módulo de Ingestão de Dados 24
4.3.2 Módulo de Classificação 25
4.3.3 API REST para Consulta de Dados 29
4.3.4 Resultados 30

5 CONCLUSÃO 32
5.1 Limitações e Trabalhos Futuros 33

REFERÊNCIAS 34

1 Introdução

Desde o ano de 1995, em que a internet começou a ser comercializada no Brasil (RNP, 2022), o mundo digital não parou de evoluir. As pessoas estão cada vez mais conectadas através de redes sociais, jogos, entre outros programas digitais. Entretanto, apesar dessa tecnologia trazer grandes benefícios para a população, como lazer, praticidade em tarefas diárias e socialização, a Internet tem sido usada por pessoas com más intenções, para cometer todo tipo de crime usando o meio digital, como fraudes, acessos ilegais, roubo de identidade, divulgação de pornografia, ofensas, dentre muitos outros tipos de crimes digitais.

A quantidade de casos desse tipo estão cada vez mais comum e tendem a continuar crescendo conforme a tecnologia e o acesso à ela também evolui. Dentre esses crimes, o mais frequente é o estelionato virtual, cujo número de incidências quadruplicou do ano 2018 a 2022 (HONóRIO; STABILE; PAIVA, 2023), além disso uma pesquisa conduzida pelo DataSenado em parceria com a empresa Nexus revelou que 24% dos brasileiros foram vítimas de algum golpe digital num período de 12 meses entre 2023 e 2024 (CANALTECH, 2024).

Em face dessa situação, advogados, promotores e juízes enfrentam o desafio de encontrar precedentes e compreender como casos similares são julgados em diferentes comarcas. Para a sociedade e formuladores de políticas públicas, a falta de um panorama claro impede a criação de estratégias de segurança e campanhas de conscientização eficazes, pois não se sabe exatamente onde, como e com que frequência esses crimes são processados.

A análise acadêmica sobre a jurisprudência de crimes cibernéticos no Brasil é severamente limitada pela ausência de bases de dados centralizadas e estruturadas. Pesquisadores dependem de buscas manuais em diários de justiça ou sistemas de tribunais individuais, um processo moroso e propenso a erros que inviabiliza análises de tendências em larga escala.

Diante desse cenário, este trabalho propõe a criação do Observatório de Cibercrimes. Inspirado em outros "sistemas observatório", que são plataformas projetadas para centralizar, analisar e disseminar dados oficiais sobre determinado tema, esta iniciativa é concebida como um repositório digital focado em coletar, organizar e disponibilizar, de forma automatizada, dados sobre processos de crimes cibernéticos no Brasil. O objetivo é criar um ponto de referência para pesquisadores, profissionais do direito e para a sociedade, permitindo uma análise factual do panorama desses delitos.

O principal desafio para a viabilização de tal observatório reside no tratamento do

grande volume de dados não estruturados, uma vez que os processos judiciais são documentos textuais complexos e de difícil categorização manual em larga escala. É neste ponto que a utilização de um LLM (Modelo de Linguagem Grande) se torna o pilar tecnológico que conecta as duas frentes deste trabalho. Ao empregar a capacidade de interpretação de linguagem natural de um LLM, o Observatório supera a barreira da análise manual, automatizando a tarefa de classificar cada processo segundo uma taxonomia acadêmica específica. Dessa forma, o LLM não é uma ferramenta acessória, mas o componente essencial que transforma dados jurídicos brutos em uma base de conhecimento estruturada e inteligente.

O objetivo geral deste trabalho é desenvolver o *backend* (camada de serviço) do sistema, criando uma solução robusta para a coleta, armazenamento, classificação automatizada usando IA e disponibilização via API dos dados de processos judiciais de crimes cibernéticos.

Os objetivos específicos são: modelar e implementar um banco de dados relacional para armazenar de forma normalizada os dados processuais; desenvolver um módulo de ingestão para processar e persistir informações extraídas do Tribunal de Justiça do Estado de São Paulo (TJSP); Implementar um serviço de classificação automática utilizando a API de Inteligência Artificial Google Gemini, com base em uma taxonomia acadêmica consolidada; expor os dados tratados por meio de uma API REST, garantindo acesso estruturado para futuras aplicações cliente.

Este trabalho está organizado da seguinte forma: O Capítulo 2 apresenta a fundamentação teórica, detalhando os conceitos de cibercrime e as tecnologias essenciais utilizadas no desenvolvimento do sistema. O Capítulo 3 revisa a literatura, contextualizando a pesquisa frente a outros trabalhos sobre o cenário jurídico do cibercrime e sistemas do tipo "Observatório". O Capítulo 4 descreve todo o processo de desenvolvimento do *backend*, incluindo a escolha das tecnologias, a modelagem do banco de dados, a implementação detalhada dos módulos de ingestão e classificação e os resultados da classificação. Por fim, o Capítulo 5 apresenta a conclusão do trabalho, consolidando os resultados obtidos, além das limitações da pesquisa e as direções para trabalhos futuros.

2 Fundamentação teórica

Este capítulo tem o objetivo de mostrar e explicar todos os conceitos, as ferramentas e tecnologias usadas para o desenvolvimento do sistema.

2.1 Crime cibernético

Crimes cibernéticos são ações criminosas cometidas com ou auxiliadas por um computador, uma rede de computadores ou um dispositivo conectado em rede. Tais crimes podem ser cometidos por indivíduos ou organizações, por criminosos estruturados e altamente capacitados tecnicamente ou por novatos (LAB, 2025).

O objetivo mais usual dos cibercriminosos é ganhar dinheiro, contudo não se limitam à isso. Além do estelionato, os cibercrimes podem estar relacionados à:

- Danos contra a confidencialidade, integridade e disponibilidade de dados e sistemas de computadores, como a espionagem de dados, interferência em sistemas ou dados, etc.
- Falsificação e roubo de identidade
- Ofensas relacionadas a conteúdo, como pornografia, intolerância religiosa, jogos de azar ilegais, *spam*, racismo e discurso de ódio
- Ofensas relacionadas a infrações de direitos autorais e direitos conexos

Para organizar a vasta gama de delitos digitais, diversas taxonomias foram propostas na literatura. A ausência de uma classificação padronizada dificulta a análise e o monitoramento sistemático desses crimes. Uma abordagem abrangente é o sistema de classificação proposto por Tsakalidis e Vergidis (2017), que se baseia em instrumentos internacionais como a Convenção sobre o Cibercrime para criar uma estrutura lógica e detalhada.

A taxonomia é organizada em um sistema de dois níveis. O Nível 1 agrupa os crimes em cinco categorias genéricas, enquanto o Nível 2 detalha os delitos específicos dentro de cada categoria, conforme apresentado na Tabela 1.

As cinco categorias principais do Nível 1 são definidas da seguinte forma:

- Tipo A - Delitos contra a confidencialidade, integridade e disponibilidade de dados e sistemas computacionais: Abrange o núcleo dos crimes de computador, como *hacking*

Tabela 1 – Sistema Proposto de Classificação de Crimes Cibernéticos

LEVEL-1	**LEVEL-2**
Type A Offences against the confidentiality, integrity and availability of computer data and systems	A1. Illegal Access (hacking, cracking) A2. Illegal data acquisition (data espionage) A3. Illegal Interception A4. Data Interference A5. System Interference A6. Misuse of devices
Type B Computer-related offences	B1. Computer-related forgery B2. Computer-related fraud B3. Identity theft
Type C Content-related offences	C1. Pornographic Material C2. Child Pornography C3. Religious Offences C4. Cyberbullying C5. Illegal gambling and online games C6. Spam and related threats C7. Racism and hate speech on the Internet
Type D Offences related to infringements of copyright and related rights	D1. Copyright-related offences D2. Trademark-related offences
Type E Combinational Offences	E1. Phishing E2. Cyber laundering E3. Cyberwarfare E4. Terrorist use of the Internet

Fonte: (TSAKALIDIS; VERGIDIS, 2017)

(acesso não autorizado), interceção ilegal e interferência em dados e sistemas. Esta categoria foca nos ataques que têm como alvo a própria infraestrutura tecnológica.

- Tipo B - Delitos informáticos: Inclui crimes em que o computador é utilizado como método para cometer delitos tradicionais já protegidos pela lei, como fraude, roubo de identidade e falsificação eletrônica.

- Tipo C - Delitos relacionados ao conteúdo: Engloba crimes que utilizam sistemas computacionais como meio para disseminar conteúdo abusivo ou ilegal. Exemplos incluem a distribuição de material pornográfico, *cyberbullying*, discurso de ódio e a promoção de jogos de azar ilegais.

- Tipo D - Delitos relacionados a infrações de direitos autorais e direitos conexos:

Categoria que trata de uma das formas mais difundidas de crime informático: a violação de direitos autorais e de marcas registadas através de meios digitais.

- Tipo E - Delitos combinados: Categoria proposta pelos autores para classificar crimes complexos que integram múltiplos delitos num único ato. Casos como *phishing*, lavagem de dinheiro virtual (*cyber laundering*) e o uso da internet para fins terroristas enquadram-se aqui, pois combinam elementos de fraude, roubo de identidade e interferência de sistemas, por exemplo.

Esta estrutura de classificação oferece um modelo teórico robusto para a categorização de incidentes de cibercrime, permitindo uma análise mais detalhada e padronizada dos diferentes tipos de delitos digitais.

2.2 Tecnologias e ferramentas

2.2.1 Java

Java é uma linguagem de programação de alto nível, orientada a objetos e multiplataforma, desenvolvida originalmente por James Gosling na Sun Microsystems e lançada em 1995. A ideologia principal da linguagem Java é "escreva uma vez, rode em qualquer lugar" (*write once, run anywhere*) (AWS, 2025), ou seja, a capacidade de rodar um código feito em java em qualquer dispositivo que possua a JVM (Máquina Virtual Java).

Os aspectos da linguagem de poder organizar o código em objetos e classes e rodar o programa em uma máquina virtual fazem do Java uma linguagem de programação versátil, com possibilidade de reutilização de código, modular, com alta manutenibilidade dos programas e fácil de aprender (ORACLE, 2025; AZURE, 2025).

2.2.2 Spring Framework

Spring Framework, como o próprio nome diz, é um *framework open source* que tem o objetivo de auxiliar no desenvolvimento de *softwares* usando a linguagem de programação Java. O Spring é uma ferramenta muito útil que simplifica a programação, resolvendo problemas comuns encontrados no desenvolvimento de aplicativos corporativos, como complexidade de configuração, segurança, acoplamento rígido e dificuldades de testabilidade (COMMUNITY, 2023; BROADCOM, 2025).

2.2.3 PostgreSQL

PostgreSQL é um SGBD relacional *open source* com mais de 35 anos de desenvolvimento ativo. Sua confiabilidade e integridade o tornaram muito popular entre os

desenvolvedores, além disso o software foi projetado para ser compatível com todos os principais sistemas operacionais incluindo Linux, Windows e Macintosh. O PostgreSQL oferece suporte a texto, imagens, sons e vídeos, além de possibilitar a definição de linguagens funcionais e tipos de dados, incluindo tipos personalizados ou tipos definidos pelo usuário (GROUP, 2023; MICROSOFT, 2025).

2.2.4 Gradle

O Gradle é uma ferramenta de automação de compilação de código *open source* projetada para suportar múltiplas linguagens e projetos de qualquer complexidade. Ele combina ideias dos predecessores Apache Ant e Apache Maven, mas utiliza uma DSL (Domain-Specific Language) baseada em Groovy ou Kotlin, em vez de XML (GRADLE, 2025).

2.2.5 Arquitetura Cliente-Servidor

A arquitetura Cliente-Servidor é bem comum e amplamente usada para a criação de sistemas distribuídos. Ela se baseia em uma comunicação através de rede entre dois ou mais nós (computadores), sendo um o cliente e o outro o servidor, em que geralmente o cliente solicita um serviço ao servidor e o servidor responde (TANENBAUM, 2016).

2.2.6 Protocolo HTTP e APIs REST

HTTP é o protocolo de comunicação mais usado para transferir dados (como páginas web, imagens e vídeos) entre cliente e servidor na web (CONTRIBUTORS, 2025).

APIs REST (ou RESTful) é uma abordagem para projetar e desenvolver interfaces de programação web que seguem os princípios do estilo arquitetural REST. REST é um conjunto de diretrizes que visa fornecer uma estrutura para a comunicação entre sistemas distribuídos, que são as seguintes:

- Utilização dos métodos HTTP, como GET, POST, PUT e DELETE, para realizar operações em recursos.
- Uso de URLs (Uniform Resource Locators) para identificar recursos específicos.
- Transferência de dados entre cliente e servidor em um formato padrão, geralmente JSON ou XML.
- Cada requisição contém todas as informações necessárias, o servidor não guarda estado da sessão do cliente.
- Dados armazenáveis em cache que simplificam as interações entre cliente e servidor.

(ANDRADE, 2023; HAT, 2023).

2.2.7 Swagger

Swagger (atual OpenAPI) é um conjunto de ferramentas *open source* para descrever, documentar, testar e consumir APIs REST (SOFTWARE, 2025). Com isso, é possível gerar a documentação automática da API com uma interface gráfica interativa, contendo as seguintes informações:

- Endpoints (URLs da API)
- Métodos HTTP (GET, POST, PUT, DELETE etc.)
- Parâmetros de entrada
- Estrutura das respostas (JSON, XML, etc.)
- Códigos de status (200, 404, 500 etc.)

2.2.8 Flyway

Flyway é uma ferramenta para versionamento de banco de dados, através de um controle de versões por script, além disso ele consegue garantir a integridade do banco com segurança para atualizar as versões. Ele faz isso usando uma tabela criada por ele onde guarda todo o histórico das versões e seus status. Esse gerenciamento dos scripts escritos para execução no banco de dados é muito útil quando se tem uma equipe de dois ou mais desenvolvedores trabalhando no mesmo projeto, pois todos da equipe teriam que compartilhar os scripts feitos entre si e também executar na ordem correta, sempre que precisassem replicar o banco de dados, por exemplo, ou até mesmo testar localmente (VITOR, 2024).

2.2.9 Git e GitHub

Git é um sistema de controle de versão, que é muito util para grandes projetos e que são desenvolvidos por duas ou mais pessoas. Já o GitHub é uma plataforma baseada em nuvem que utiliza o Git para armazenar os arquivos de projetos em um repositório, para monitorar e gerenciar as alterações feitas no projeto (GITHUB, 2025).

2.2.10 Google Gemini

O Gemini é uma família de modelos de inteligência artificial generativa da empresa Google. Ele serve para múltiplas tarefas, como geração de conteúdo criativo, resolução de problemas complexos, assistência em tarefas do dia a dia, integração com sistemas leitura e reconhecimento de imagens, áudios e vídeos (CARRARO, 2024).

3 Trabalhos relacionados

Este capítulo contextualiza o sistema Observatório de Cibercrimes, posicionando-o em relação a outras pesquisas e projetos relevantes. Para uma análise abrangente, a revisão da literatura foi dividida em duas vertentes principais.

A primeira seção aborda o cenário jurídico e as taxonomias do cibercrime, explorando trabalhos acadêmicos que discutem a tipificação, as consequências e os desafios da persecução penal de delitos digitais na jurisdição brasileira. Essa análise fundamenta a relevância social e a necessidade de classificação dos dados, que são a base da problemática que este trabalho busca resolver.

A segunda seção apresenta o conceito de Sistemas Observatório como Ferramentas de Análise e Monitoramento. Nela, são descritas iniciativas existentes, como o Observatório Nacional de Blockchain e o projeto CuidaIdoso, traçando um paralelo entre suas arquiteturas e objetivos e a solução proposta neste trabalho. O objetivo é demonstrar como o Observatório de Cibercrimes se inspira em modelos conceituais e arquitetônicos consolidados, ao mesmo tempo que inova ao aplicar tais conceitos a um domínio de dados distinto.

3.1 Cenário Jurídico e Taxonômico do Cibercrime

Existem diversos trabalhos relacionados aos crimes cibernéticos e seu cenário atual na jurisdição brasileira, além de tentativas de tipificação e sugestões de taxonomias. A publicação de Arruda e Lima (2023), por exemplo, teve como objetivo discutir as consequências desses crimes, o perfil das vítimas, as porcentagens dos crimes informáticos no Brasil e a proteção do direito de privacidade, com base nas alterações da legislação civil e penal vigente em nosso país. Utilizando métodos de abordagem dedutivo e de procedimento monográfico, as autoras comprovaram a necessidade de atualizações e modificações que assegurem o direito à privacidade e a prevenção de novos delitos virtuais, além da compreensão dos crimes cibernéticos e das leis que capacitam a proteção das vítimas. Esse fato evidencia a importância da criação de um sistema digital, como o proposto nesta monografia, para rastrear tais atividades criminosas que passaram pelo meio jurídico, facilitando a análise e, sucessivamente, a atualização das leis no Brasil.

O trabalho de Phillips et al. (2022) focou em explorar e consolidar definições, tipologias e taxonomias de cibercrime encontradas na literatura acadêmica, usando a metodologia estruturada de revisão de literatura. O resultado foi a criação de um novo *framework* de classificação para compreender o cibercrime, o qual serve como inspiração e

referência para a organização dos parâmetros utilizados no sistema proposto. Já o artigo de Bernik (2016) demonstra o quão custoso é para as organizações prevenir e investir contra ataques virtuais, investigando os investimentos em proteção contra o cibercrime. A conclusão do autor reforça a necessidade de ferramentas digitais que, como o Observatório de Cibercrimes, possam auxiliar as organizações a investir de forma mais eficiente, compreendendo o panorama real das ameaças julgadas.

De forma ainda mais central para este trabalho, destaca-se o artigo Tsakalidis e Vergidis (2017). Os autores abordam diretamente o problema da falta de uma classificação concisa e sistemática dos delitos cibernéticos, o que dificulta a aplicação de contramedidas e políticas eficazes.

A principal contribuição do artigo é a proposição de um sistema de classificação de dois níveis, projetado para ser uma referência comum e abrangente. O Nível 1 divide os crimes em cinco categorias principais (Tipo A a E), enquanto o Nível 2 detalha as ofensas específicas dentro de cada tipo. A taxonomia foi consolidada a partir de instrumentos internacionais, como a Convenção sobre o Cibercrime, e atualizada para incluir delitos combinados e complexos, como o *phishing* e a lavagem de dinheiro virtual.

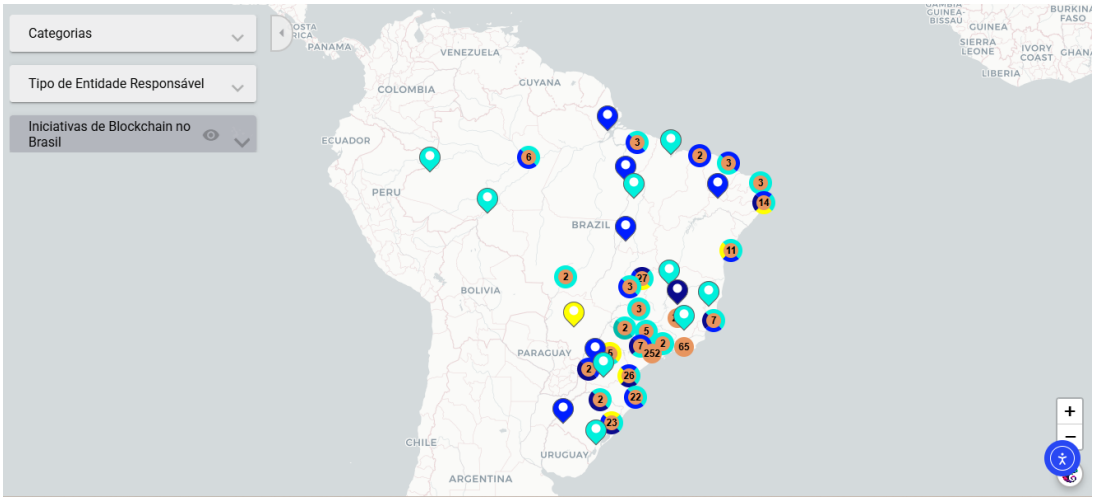
Diferentemente dos outros trabalhos citados, que servem como inspiração e contextualização, o artigo de Tsakalidis e Vergidis (2017) é a fundação teórica direta sobre a qual o módulo de classificação do Observatório de Cibercimes foi construído. A taxonomia apresentada no artigo não foi apenas uma referência, mas sim o modelo adotado e fornecido à Inteligência Artificial para a categorização automatizada dos processos judiciais, tornando este artigo o pilar metodológico para a principal inovação deste trabalho.

3.2 Sistemas Observatório como Ferramentas de Análise e Monitoramento

Ademais, existem outros trabalhos de sistemas do tipo Observatório. O conceito de um "sistema observatório" no campo da tecnologia da informação refere-se a plataformas projetadas para coletar, organizar, analisar e disseminar dados sobre um fenômeno específico, servindo como um ponto central de referência para pesquisadores, gestores e o público em geral. A seguir, são apresentados alguns observatórios e projetos de centralização de informação existentes, traçando um paralelo com a proposta deste trabalho.

Um exemplo proeminente no cenário tecnológico brasileiro é o Observatório Nacional de Blockchain, cujo painel de mapa é mostrado na Figura 1, lançado em abril de 2025 como fruto de uma parceria entre a RNP e o CPQD, com apoio do MCTI (RNP; CPQD, 2025). A iniciativa nasceu com a missão de mapear, divulgar e fomentar o desenvolvimento da tecnologia blockchain no país. O objetivo do observatório é agregar informações

Figura 1 – Painel de mapa do Observatório Nacional de Blockchain



Fonte: Adaptado de RNP; CPQD (2025)

sobre iniciativas públicas e privadas, promovendo o diálogo e a integração entre academia, governo e o setor empresarial. Para isso, a plataforma oferece recursos como:

- Mapa de iniciativas: Identifica startups, grupos de pesquisa e projetos governamentais relacionados à blockchain.
- Indicadores do setor: Apresenta dados sobre a produção científica nacional.
- Curadoria de conteúdos: Disponibiliza relatórios, vídeos e oportunidades de pesquisa sobre o tema.
- Comunidade de Especialistas: Fomenta a troca de experiências por meio de encontros virtuais e grupos de discussão.

O paralelo com o Observatório de Cibercrimes reside no conceito fundamental de ser uma ferramenta estratégica de centralização e monitoramento. Ambas as plataformas buscam conectar diferentes atores (academia, governo, empresas) e servir como uma fonte agregadora de informações para fomentar o diálogo e a inovação em suas respectivas áreas.

A principal distinção está no escopo e na natureza do processamento de dados. O Observatório Nacional de Blockchain foca em uma tecnologia e seu ecossistema, atuando primariamente como um agregador e curador de informações já existentes. Em contrapartida, o Observatório de Cibercrime foca em um fenômeno social e jurídico e vai além da agregação, implementando um módulo de processamento e classificação ativa. Ele ingere dados brutos e não estruturados (processos judiciais) e, utilizando inteligência artificial, os transforma em uma base de dados estruturada e enriquecida, gerando uma nova camada de conhecimento que não existia previamente.

Figura 2 – Interface Gráfica do Cuidaidoso

Fonte: Adaptado de XR4Good Lab; Âmbar; LIMP (2025)

Embora não se autodenomine um "observatório", o projeto CuidaIdoso, cujo exemplo da interface gráfica pode ser observado na Figura 2, compartilha o mesmo princípio de centralizar e disseminar informações para resolver um problema social específico: a necessidade de fornecer informações de saúde confiáveis para idosos, combatendo a desinformação (*fake news*), especialmente no contexto da pandemia de COVID-19 (ERSE, 2025).

Para atingir seu objetivo de estar presente em múltiplos canais (sites, redes sociais, aplicativos móveis), a arquitetura do projeto foi dividida entre *frontend* e *backend*. Foi implementado um *backend* centralizado, com uma API REST, que fornece os mesmos recursos e informações para diferentes tipos de interface, evitando retrabalho e garantindo a simetria da informação entre as plataformas.

A semelhança arquitetônica com o Observatório de Cibercrimes é notável. Ambos os projetos adotaram a estratégia de um *backend* centralizado como "fonte única da verdade"para servir a diferentes clientes. Assim como o CuidaIdoso, a API do Observatório de Cibercrimes foi projetada para ser consumida por uma futura interface web, permitindo que o desenvolvimento das funcionalidades do sistema ocorra de forma independente e paralela à da camada de apresentação.

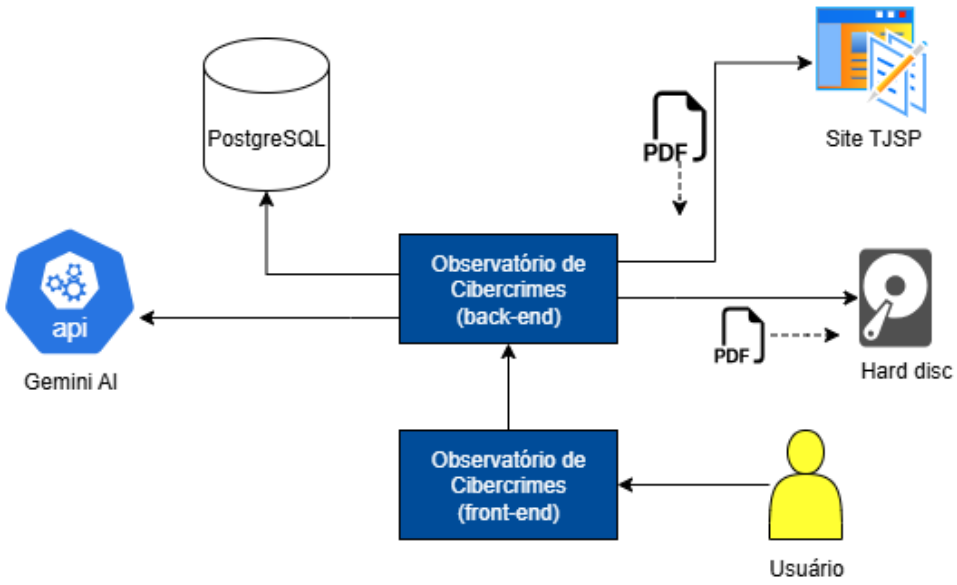
A diferença fundamental reside na origem e no tratamento dos dados. O *backend* do CuidaIdoso gerencia e dissemina, em grande parte, conteúdo curado e produzido pela própria equipe do projeto (dicas, informativos, etc.). Já o Observatório de Cibercrimes é projetado para a ingestão e transformação de um grande volume de dados externos e brutos (processos do Tribunal de Justiça), cujo valor principal é gerado pelo processo de classificação automatizada, tornando-os pesquisáveis e analisáveis. O foco do CuidaIdoso é primariamente informativo e de engajamento social, enquanto o Observatório de Cibercrimes é analítico e investigativo.

4 Desenvolvimento

Este capítulo tem o objetivo de mostrar o processo de desenvolvimento do *back-end* do sistema Observatório de Cibercrimes, o qual está salvo em um repositório[1] no Github, desde o levantamento de informações, as tomadas de decisões, a modelagem e a codificação. O sistema foi concebido como uma plataforma completa, composta por três componentes principais:

- Módulo de Coleta e Persistência (*backend*): O coração do sistema, responsável por buscar os dados brutos de fontes judiciais, processá-los e armazená-los de forma estruturada. Este componente é parte do escopo deste trabalho.
- Módulo de Classificação e Enriquecimento (*backend*): Um serviço que opera sobre os dados armazenados, utilizando inteligência artificial para classificá-los e enriquecer a base de dados. Este componente é parte do escopo deste trabalho.
- Módulo de Visualização (*frontend*): Uma interface web (painel) que consome a API REST para apresentar os dados de forma visual, com mapas, gráficos e filtros.

Figura 3 – Diagrama de Arquitetura do Sistema



Fonte: Elaborado pelo autor

Assim como demonstra a Figura 3, a arquitetura escolhida para o sistema Observatório de Cibercrimes é Cliente-Servidor, a qual permite o desacoplamento entre o *backend*

[1] <https://github.com/Zeitoum/Observatorio-de-cibercrimes/tree/main/backend>

e o *frontend*. Isso é possível, usando uma comunicação e tipo de dados, para a requisição e resposta, padronizados, os quais concede uma maior flexibilidade na criação do cliente (*frontend*). Além de facilitar o trabalho em conjunto no desenvolvimento do sistema, essa separação possibilita a criação de outros clientes no futuro, por exemplo uma aplicação para dispositivos móveis.

Quanto a funcionalidade do sistema, o processo inicia com a população do banco de dados relacional no momento em que o *backend* recebe uma requisição HTTP contendo o caminho do arquivo/planilha. Enquanto isso, é rodado um código em paralelo periodicamente, que busca o arquivo PDF de cada processo judicial, salva-o no armazenamento secundário e envia para a API do Google Gemini para classificá-lo. Após a classificação, os resultados são armazenados no PostgreSQL e disponibilizados atrávez de uma API REST para que o *frontend* possa consumir esses dados de forma estruturada e organiza-los em forma de gráficos, mapas e listas para que o usuário possa vê-los e entende-los.

4.1 Escolha das tecnologias

Para a criação do *back-end* foi escolhida a linguagem de programação Java em conjunto com o *framework* Spring. Essa escolha se deve ao fato de que Java é uma linguagem madura e estável, já que ela é usada a décadas por grandes empresas, como TOTVS, Mercado Livre, Itaú, PicPay, Sicredi, etc (DIO, 2025a). Além disso, por ser uma linguagem amplamente utilizada, ela possui uma grande variedade de bibliotecas úteis disponíveis gratuitamente para uso (DIO, 2025b).

Assim como o Java, o Spring Boot também é uma ferramenta amplamente usada por grandes empresas, além de que possui grandes vantagens, como configurações comuns prontas e algumas bibliotecas java incluídas em suas dependências, o que ajuda o desenvolvedor a focar mais na programação da regra de negócio do produto, ao invés de gastar muito tempo fazendo, por exemplo, configurações de conexões de banco de dados, de segurança e APIs REST (ADAK, 2025).

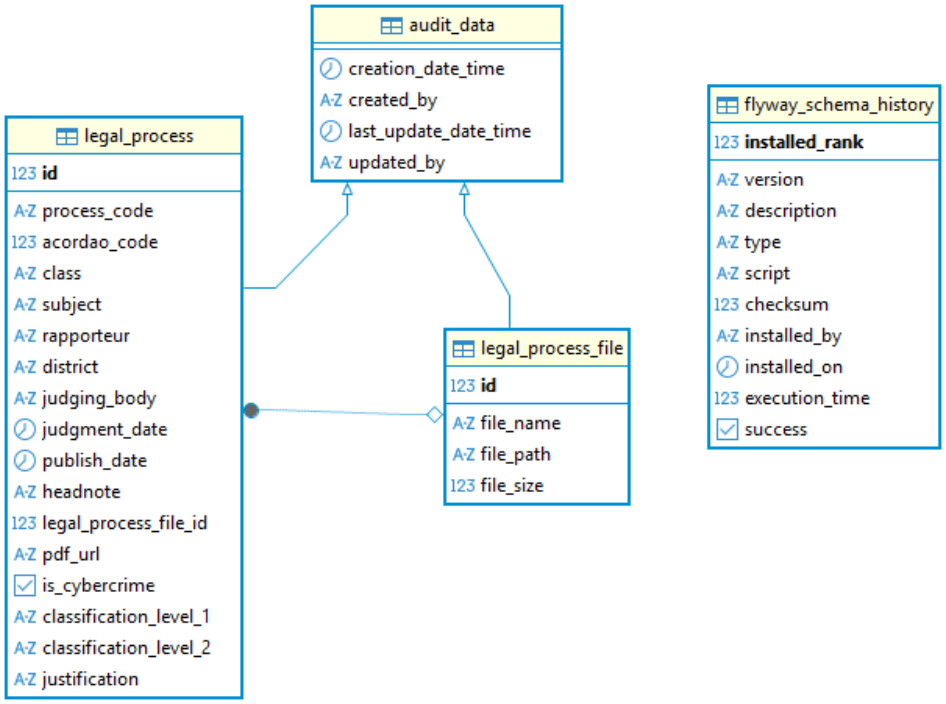
Além da linguagem de programação e do *framework*, foi utilizado o Gradle como ferramenta para a automação de compilação de código e gerenciador de dependências, com objetivo de não precisar baixar e configurar todos os pacotes necessários manualmente, economizando tempo de desenvolvimento. Os motivos de se escolher o Gradle ao invés de outros gerenciadores de dependência, foi sua linguagem de configuração mais legível, seu desempenho e alto nível de customização.

Para o armazenamento dos dados, foi escolhido um banco relacional, devido às suas características que permitem a integridade de dados, a normalização e transações ACID (Atomicidade, consistência, isolamento e durabilidade). Pois este não é um sistema que tem o foco em desempenho, apesar de ser um ponto importante para todo e qualquer

sistema digital, mas o objetivo principal é garantir a confiabilidade dos dados visualizados.

4.2 Modelagem do Banco de Dados

Figura 4 – Diagrama de Entidade-Relacionamento



Fonte: Elaborado pelo autor

A partir dos dados sobre os casos judiciais obtidos no site do TJSP, foi realizada a modelagem do banco de dados relacional do sistema. O Diagrama de Entidade-Relacionamento (DER) resultante, que representa a estrutura das tabelas e seus relacionamentos, pode ser visualizado na Figura 4.

Listing 4.1 – Script SQL para criação da tabela legal_process

```
CREATE TABLE public.legal_process (
        id int8 GENERATED BY DEFAULT AS IDENTITY( INCREMENT BY 1
           MINVALUE 1 MAXVALUE 9223372036854775807 START 1 CACHE 1
            NO CYCLE) NOT NULL,
        process_code varchar(255) NOT NULL,
        acordao_code int8 NOT NULL,
        "class" varchar(255) NOT NULL,
        subject varchar(255) NOT NULL,
        rapporteur varchar(255) NOT NULL,
        district varchar(255) NOT NULL,
        judging_body varchar(255) NULL,
        judgment_date date NULL,
```

```
        publish_date date NULL,
        headnote text NULL,
        legal_process_file_id int8 NULL,
        pdf_url varchar(255) NULL,
        is_cybercrime bool NULL,
        classification_level_1 varchar(255) NULL,
        classification_level_2 varchar(255) NULL,
        justification text NULL,
        CONSTRAINT legal_process_pkey PRIMARY KEY (id),
        CONSTRAINT process_code_uk UNIQUE (process_code)
)
INHERITS (public.audit_data);

ALTER TABLE public.legal_process ADD CONSTRAINT
   legal_process_file_fk FOREIGN KEY (legal_process_file_id)
   REFERENCES public.legal_process_file(id);
```

A tabela *legal_process*, cujo *script* de criação é mostrado na Listagem 4.1, é a entidade central, armazenando os dados extraídos de um processo. É importante notar que os campos *is_cybercrime*, *classification_level_1*, *classification_level_2* e *justification* são deliberadamente projetados para serem nulos inicialmente. Eles são preenchidos posteriormente pelo módulo de classificação, representando o processo de enriquecimento dos dados.

A decisão de criar a entidade *audit_data*, a qual tem a função de auditoria, segue o princípio de rastreabilidade. Em um sistema que será continuamente atualizado com novos processos, é crucial saber quando cada registro foi criado ou modificado, garantindo a integridade e a governança dos dados.

Já tabela *legal_process_file*, tem a função de armazenar os metadados dos arquivos PDFs salvos na memória secundária, os quais são usados para localizar esses arquivos para enviá-los à API do Google Gemini.

Quanto a tabela *flyway_schema_history* serve para uso do Flyway, o qual usa essa tabela para controle de versionamento dos scripts SQL. É importante ressaltar que o próprio Flyway faz a criação da tabela e suas respectivas atualizações de dados.

4.3 Implementação do Backend

4.3.1 Módulo de Ingestão de Dados

No primeiro momento, foram coletadas informações sobre casos judiciais do TJSP. Esses dados foram armazenados em uma planilha[2]. São 4540 linhas de dados, em que cada linha representa um caso judicial e cada coluna da planilha contém uma informação diferente sobre o caso.

Antes de iniciar a classificação automática em larga escala, foi realizada uma análise amostral para validar a metodologia e obter uma estimativa da prevalência de cibercrimes nos dados. Para tanto, alguns casos judiciais foram escolhidos para serem classificados manualmente, com base na leitura do respectivo documento em PDF. Este processo seguiu estritamente a taxonomia de Tsakalidis e Vergidis (2017), utilizando suas cinco categorias principais: Tipo A (Delitos contra a confidencialidade, integridade e disponibilidade), Tipo B (Delitos informáticos, como fraudes), Tipo C (Delitos relacionados ao conteúdo), Tipo D (Violações de direitos autorais) e Tipo E (Delitos combinados). Essa classificação manual foi útil para aferir a porcentagem aproximada de casos que se encaixavam no contexto de crime cibernético, servindo como um ponto de referência para comparar e avaliar os resultados que seriam posteriormente gerados pela Gemini AI.

Para dar início ao processamento da planilha contendo as informações, foi criada uma API REST, a qual recebe o caminho em que se encontra o arquivo no corpo de uma requisição HTTP. Portanto, sempre que for criado um novo arquivo com novos dados sobre crimes cibernéticos, é necessário apenas salvar o arquivo na memória secundária do servidor e enviar uma requisição HTTP para o serviço para popular o banco de dados com essas novas informações.

O *script* foi criado para ler uma linha por vez, porém a demora era considerável, ainda mais para um arquivo com mais de quatro mil e quinhentas linhas. Então, foi alterado para ler as linhas de forma assíncrona.

Listing 4.2 – Método para processamento da planilha de forma assíncrona

```
1 public void processSpreadsheetFile(String spreadsheetFilePath) {
2     log.info("Extracting file information from spreadsheet: {}",
          spreadsheetFilePath);
3
4     Path filePath = Path.of(spreadsheetFilePath);
5
6     final int threads = Runtime.getRuntime().availableProcessors();
7     ExecutorService executor = Executors.newFixedThreadPool(threads);
8
```

[2] <https://github.com/Zeitoum/Observatorio-de-cibercrimes/blob/main/backend/casos-judiciais.csv>

```
9       try (Stream<String> lines = Files.lines(filePath, StandardCharsets.
            UTF_8)) {
10          lines.forEach(line ->
11              executor.submit(() -> {
12                  try {
13                      processFileLine(line);
14                  } catch (Exception e) {
15                      throw new RuntimeException("Error processing line: "
                            + line, e);
16                  }
17              }));
18      } catch (IOException e) {
19          throw new RuntimeException("Error reading file: " +
                spreadsheetFilePath, e);
20      } finally {
21          executor.shutdown();
22      }
23  }
```

Como ilustrado no código 4.2, utilizamos um *ExecutorService* com um *pool* de *threads* dimensionado pelo número de processadores disponíveis para a JVM, que foi extraído a partir do método *Runtime.getRuntime().availableProcessors()*. Isso permite que múltiplas linhas sejam processadas em paralelo, reduzindo drasticamente o tempo de ingestão. O motivo para usar uma quantidade de threads igual e não maior que o número de núcleos de processadores disponíveis é porque não é possível executar dois fluxos de código simultâneamente no mesmo núcleo, portanto não aumentará o desempenho mesmo se fosse adicionadas mais threads.

É possível notar que pode ocorrer o problema de duplicação de informações já que é possível processar os dados de um mesmo crime duas vezes, caso seja necessário processar a planilha de dados novamente, ou até mesmo um novo arquivo que tenha salvado informações de casos judiciais iguais. Por isso, foi feito uma tratativa no código executado para cada linha da planilha, no intuito de verificar se tal informação já está salva no banco de dados ou não. Essa verificação é feita usando o código do processo junto com o código do acordão, os quais demonstram ser únicos para cada caso jurídico.

4.3.2 Módulo de Classificação

Listing 4.3 – Método que é executado periodicamente e inicia a classificação

```
1   @Scheduled(fixedDelay = 1000) // 1 second
2   public void execute() throws InterruptedException {
3       log.info("Starting Legal Process Classifier Scheduler...");
4
5       service.classifyLegalProcesses();
```

```
6
7           log.info("Legal Process classification finished !");
8       }
```

O método java descrito no *script* 4.3, agendado com a anotação *@Scheduled* do Spring Boot é executado periodicamente para classificar os casos judiciais. Observa-se que foi passado o parâmetro *fixedDelay* com o valor *1000*, que significa que esse método executará a cada mil milissegundos (um segundo) de intervalo entre o fim e o inicio da execução do mesmo. Assim sendo, não haverá sobreposição de execução da classificação, ou seja, enquanto um fluxo de classificação estiver rodando, não terá como executar outro ao mesmo tempo.

O processamento da classificação dos casos judiciais inicia-se com a busca de 100 casos judiciais, que ainda não foram classificados. E a maneira que foi feito isso é fazendo uma *query* na na tabela *legal_process*, buscando as linhas que têm o campo *is_cybercrime* nulo, como pode-se observar na consulta SQL 4.4.

Listing 4.4 – Código SQL para buscar casos judiciais não classificados

```
SELECT lp.id AS legal_process_id, lp.acordao_code, lp.pdf_url,lpf
   .file_path
FROM legal_process lp
LEFT JOIN legal_process_file lpf ON lpf.id = lp.
   legal_process_file_id
WHERE lp.is_cybercrime IS NULL
LIMIT 100;
```

Para evitar sobrecarregar o site do TJSP e devido às limitações de taxa da API do Gemini, que mesmo com a licença de estudante possui um limite de 10 requisições por minuto e 250 por dia, foi implementado um código para processar uma linha por vez e também um *delay* (atraso) fixo de 6 segundos entre cada processamento de um caso judicial. Essa medida garante a operação estável e sustentável do sistema.

A cada linha da tabela *legal_process* que módulo processa, o sistema faz o *download* do arquivo usando o valor do campo "*pdf_url*", salva-o no armazenamento secundário, cria uma nova linha na tabela *legal_process_file* com os metadados do arquivo e usa a chave primária retornada para atualizar a linha da tabela *legal_process* correspondente, adicionando a *foreign key* (chave estrangeira).

Após esse processo de salvar o arquivo, é necessário fazer uma requisição à API do Google Gemini passando os parâmetros necessários. A documentação pública e o pacote Java disponibilizados do Google Gemini facilita bastante este procedimento, tornando tão simples como chamar um método Java para executar (GEMINI, 2025a).

Listing 4.5 – Exemplo de uso da biblioteca da API do Google Gemini

```
1 Client client = new Client();
2
3 GenerateContentResponse response =
4         client.models.generateContent(
5                 "gemini-2.0-flash",
6                 "Write a story about a magic backpack.",
7                 null);
8
9 System.out.println(response.text());
```

Verifica-se que um dos parâmetros que é passado na linha 5 do código 4.5, trata-se do nome da variante do modelo do Gemini. O Gemini disponibiliza vários modelos, os quais cada um possui tipos de saídas de respostas diferentes, tal quais texto, áudio, imagem e vídeos (GEMINI, 2025b).

Para este trabalho, foi considerado apenas os modelos da versão mais recente (2.5) que tem a saída exclusiva do tipo "texto", que são gemini-2.5-pro, gemini-2.5-flash e gemini-2.5-flash-lite. De acordo com o Gemini (2025b), o modelo "pro"é o mais inteligente, e consequentemente o mais caro. Enquanto o modelo "gemini-2.5-flash-lite"é considerado o mais rápido, mas também com o menor raciocínio. Já o gemini-2.5-flash é considerado o meio termo e o mais eficiente economicamente. Como o processamento de classificação dos casos judiciais não necessita de velocidade rápida, devido às limitações mencionadas acima, foi escolhido o modelo "gemini-2.5-flash", que demonstrou "inteligência"o suficiente para classificar os casos de forma precisa.

Então o *backend* faz uma requisição para a API do Google Gemini para classificar, junto com o PDF do caso judicial, a Tabela 1 no formato de imagem, um arquivo texto contendo parte do artigo (TSAKALIDIS; VERGIDIS, 2017) e o *prompt* a seguir:

> *Você é um expert em classificar casos judiciais se são um crime cibernético ou não. Além disso, você sabe classificar qual o tipo de crime cibernético ocorreu, levando em consideração o artigo em anexo, e principalmente a imagem da tabela em anexo, que também se encontra no artigo. a sua classificação deverá ser simples e direta. Você deve responder usando apenas 3 tópicos: Status, Classificação e Justificativa. No tópico Status você deverá colocar apenas Sim quando for um crime cibernético e Não quando não for um crime cibernético. No tópico Classificação você deverá colocar apenas a classificação, sem mais explicações, além disso deverá ter apenas uma classificação, se você achar que tem mais de uma classificação, então coloque apenas aquela que é a principal do caso. No tópico Justificativa você deverá dar uma breve explicação do caso, com no máximo 50 palavras.*
>
> *Exemplo:*
>
> ** Status: Sim*

** Classificação: B2*

** Justificativa: A fraude consistiu em uma pessoa se passar por gerente de banco no WhatsApp para induzir a vítima a fazer uma transferência bancária para a conta da ré.*

A eficácia da IA depende criticamente da engenharia de *prompt*. O *prompt* desenvolvido foi estruturado para: (1) definir o papel do modelo ("expert em classificar casos judiciais"); (2) fornecer o contexto necessário (o artigo e a tabela de classificação); e (3) exigir um formato de resposta estrito (Status, Classificação, Justificativa), o que é fundamental para permitir o *parsing* programático da resposta e a atualização consistente do banco de dados.

O desenvolvimento deste *prompt* foi um processo iterativo de engenharia, seguindo um método de tentativa e erro. Foram necessárias aproximadamente cinco iterações para que o modelo de IA retornasse consistentemente a resposta no formato exato requerido pelo sistema, garantindo que o *parsing* para salvar os dados no banco de dados fosse bem-sucedido. Durante esta fase, descobriu-se uma limitação técnica: tentativas de enviar apenas o link do PDF do caso judicial para a IA resultavam em classificações incorretas, pois o modelo parecia não conseguir acessar o conteúdo ou analisava um documento diferente. Isso tornou indispensável a implementação do *download* do arquivo PDF e o envio de seu conteúdo textual diretamente para a API, o que se mostrou a abordagem correta para garantir que a IA analisasse o caso judicial exato.

Para validar a coerência e a qualidade das classificações geradas, foi realizada uma análise amostral. Após a IA processar um lote de processos, foi selecionada uma amostra de 10 a 20 casos judiciais classificados como cibercrime ou não. Realizou-se então a leitura manual do documento PDF de cada um desses casos para verificar se a classificação (Nível 1 e 2) e a justificativa fornecidas pela IA eram corretas e condizentes com o conteúdo do processo. Nesta verificação amostral, não foram encontradas inconsistências, indicando que a IA, instruída com o *prompt* refinado, estava operando com alta assertividade e compreendendo corretamente o contexto jurídico e a taxonomia fornecida.

A seguir está um exemplo real de resposta da API do Google Gemini:

** Status: Sim*

** Classificação: B2*

** Justificativa: A vítima foi induzida por fraudadores, via aplicativo de telefone, a contratar um empréstimo e pagar um boleto falso para uma suposta portabilidade, configurando fraude auxiliada por sistemas computacionais.*

A partir da resposta da IA, é feita a atualização da linha no banco de dados, adicionando o valor *true* ou *false* no campo *is_cybercrime* e também o valor da classificação

nos campos *classification_level_1*, *classification_level_2* e *justification*, caso o processo judicial esteja no contexto de um crime cibernético.

4.3.3 API REST para Consulta de Dados

Tabela 2 – Endpoints da API do sistema

Método HTTP	**Endpoint**	**Descrição**
GET	/v1/legal-process	Retorna a lista de crimes cibernéticos de forma paginada.
POST	/v1/legal-process/process-file	Inicia o processo de ingestão de dados através de uma planilha.
GET	/v1/health/status-check	Retorna o status da saúde do backend.

Fonte: Elaborado pelo autor.

Para que os dados processados e as funcionalidades do sistema fossem acessíveis a uma aplicação cliente, foram desenvolvidos diversos endpoints. A Tabela 2 resume todos os pontos de acesso da API REST do Observatório de Cibercrimes, detalhando o método HTTP utilizado e a finalidade de cada um.

O primeiro *endpoint* criado para o *backend* do Observatório de Cibercrimes foi o "/v1/health/status-check", o qual retorna o texto "OK"com o *status* HTTP 200 se tiver sucesso ou retorna um *status* HTTP 500 caso dê algum erro, para que pudesse ser testado a implementação de uma API REST e também para conferir a saúde do sistema de tempos em tempos.

Logo em seguida foi criado o *endpoint* "/v1/legal-process/process-file", o qual espera uma *string* como valor do corpo da requisição HTTP e retorna o *status* HTTP 200 caso tenha processado a planilha com sucesso.

Então, finalmente foi criado o *endpoint* "/v1/legal-process", o qual possui alguns *Query Parameters* (Parâmetros de Consulta) opcionais:

- pageSize: representa a quantidade de itens (cibercrimes) totais por página, sendo que o valor mínimo aceito é 1 e o valor máximo é 100
- pageNumber: representa o número da página a ser retornada, sendo o valor mínimo igual à 1
- orderBy: representa a forma como deve ser ordenado os itens retornados, sendo o seguinte formato aceito: "id+ASC"
- search: representa a busca feita pelo usuário no campo de pesquisa, o qual usa os campos *process_code*, *acordao_code*, *class*, *subject* e *rapporteur* da tabela *legal_process* do banco de dados.

- isCybercrime: é o parâmetro usado para filtrar apenas o casos que são classificados como cibercrimes, caso o valor seja "*true*".

4.3.4 Resultados

A execução do módulo de classificação sobre o conjunto de dados coletados do TJSP produziu os primeiros resultados quantitativos do Observatório de Cibercrimes. Esta seção apresenta uma análise desses resultados, as tendências identificadas e as limitações inerentes ao processo.

Do total de 4.540 processos judiciais coletados, foi possível submeter 2.277 ao processo de classificação pela API do Google Gemini, uma limitação imposta pela cota de uso da ferramenta. Deste subconjunto analisado, o sistema identificou que 509 processos (aproximadamente 22,4%) se enquadravam na definição de crime cibernético, conforme a taxonomia de Tsakalidis e Vergidis (2017).

A distribuição desses 509 casos entre as cinco categorias principais (Nível 1) da taxonomia é detalhada na Tabela 3.

Tabela 3 – Distribuição dos Processos Classificados por Categoria (Nível 1)

Categoria	**Descrição Resumida**	**Quantidade de Casos**
Tipo A	Delitos contra a confidencialidade e integridade	49
Tipo B	Delitos informáticos	295
Tipo C	Delitos relacionados ao conteúdo	69
Tipo D	Violações de direitos autorais	12
Tipo E	Delitos combinados	84

Fonte: Elaborado pelo autor.

A análise da Tabela 3 revela uma predominância expressiva da Categoria B, que sozinha corresponde a mais da metade de todos os cibercrimes identificados. Para compreender melhor a natureza desses delitos, a Tabela 4 detalha a classificação dos 295 casos da Categoria B em seus subtipos (Nível 2).

Tabela 4 – Detalhamento dos Processos da Categoria B (Nível 2)

Subtipo	**Descrição**	**Quantidade de Casos**
B2	Fraude por computador	266
B3	Roubo de identidade	29

Fonte: Elaborado pelo autor.

Os dados apresentados permitem extrair uma primeira conclusão importante sobre o panorama dos cibercrimes que chegam ao judiciário paulista: a esmagadora maioria está relacionada a fraudes financeiras facilitadas por meios digitais (estelionato virtual). Com 266 casos, o subtipo B2 (Fraude por computador) representa cerca de 52% de todos os

cibercrimes classificados. Este achado quantitativo, gerado de forma automática pelo sistema, corrobora as estatísticas de segurança pública citadas na introdução deste trabalho, em que é citado o artigo do site Canaltech (2024), e evidencia que na prática jurídica, o cibercrime é predominantemente sinônimo de crimes patrimoniais que migraram para o ambiente digital.

Em contrapartida, crimes que visam a infraestrutura tecnológica em si, como ataques de *hacking* e interferência de sistemas (Tipo A), e os delitos combinados, como *phishing* (Tipo E), aparecem em menor número, embora ainda relevantes. A baixa incidência de casos de violação de direitos autorais (Tipo D) pode sugerir que este tipo de delito é menos processado criminalmente ou segue por outras vias judiciais.

5 Conclusão

O problema da dificuldade de acesso a dados estruturados sobre a jurisprudência de crimes cibernéticos no Brasil, apresentado na Introdução, foi endereçado por meio do desenvolvimento de uma plataforma de software robusta. Conforme demonstrado nos Capítulos 3 e 4, a implementação da camada de serviço *backend* do Observatório de Cibercrimes permitiu atingir todos os objetivos propostos. Foi desenvolvido um sistema que automatiza a coleta, o armazenamento e, crucialmente, a classificação de processos judiciais, disponibilizando os resultados de forma acessível via uma API REST.

O trabalho resultou na criação de um panorama inicial sobre a natureza dos cibercrimes que chegam ao judiciário paulista. Dos 2.277 processos analisados pela IA, 509 foram classificados como crimes cibernéticos, revelando que a fraude eletrônica (Categoria B2), com 266 casos, representa a esmagadora maioria dos delitos, seguida pelo uso indevido de dispositivos (Categoria E) com 84 casos. Este resultado quantitativo valida a eficácia da abordagem e fornece uma primeira "fotografia"do cenário, antes inacessível de forma agregada.

Estes resultados quantitativos têm o potencial de apoiar diretamente tribunais e órgãos de segurança. A identificação de que a “Fraude por computador” (B2) é a categoria predominante permite, por exemplo, que o judiciário aloque recursos para varas especializadas ou crie mutirões para lidar com esse gargalo específico. Para órgãos de segurança pública, esses dados refinam o foco de campanhas de conscientização, direcionando-as para os tipos de golpe mais comuns. Além disso, o monitoramento contínuo possibilitado pelo Observatório pode revelar tendências: um crescimento súbito em crimes do “Tipo A” (ataques contra a integridade de sistemas), por exemplo, poderia sugerir a necessidade de novas medidas legais ou de políticas de segurança tecnológica mais robustas em nível nacional.

A principal contribuição de conhecimento novo foi demonstrar a viabilidade de, com uma estratégia de engenharia de *prompt* adequada, utilizar um modelo de linguagem de grande escala de propósito geral (Google Gemini) para analisar e categorizar processos judiciais com base em uma taxonomia acadêmica. A análise amostral de coerência, descrita no Capítulo 4, reforça que a metodologia é capaz de gerar dados estruturados e assertivos a partir de texto não estruturado, superando o desafio da análise manual em larga escala.

Uma das principais lições aprendidas ao longo do trabalho foi a importância crítica da engenharia de *prompt* para o sucesso da classificação por LLMs. A capacidade de instruir o modelo de forma precisa, exigindo um formato de resposta estrito, foi fundamental para automatizar o *parsing* dos resultados. Ademais, aprendeu-se que a gestão de

limitações de APIs externas (custo e taxa de requisições) é um fator de engenharia de software decisivo, que impacta diretamente o escopo e a velocidade de processamento dos dados.

5.1 Limitações e Trabalhos Futuros

As limitações deste estudo estão relacionadas ao volume de dados processados e à sua representatividade geográfica. Primeiramente, devido às restrições de taxa de requisições da API do Google Gemini, dos 4.540 processos coletados, foi possível classificar 2.277 até o momento. Em segundo lugar, a base de dados atual se restringe apenas a processos do Tribunal de Justiça de São Paulo, não representando, portanto, um panorama nacional.

Essas limitações abrem caminho para diversas oportunidades de pesquisa futura, que vão além de meras implementações técnicas:

- Análise Comparativa da Jurisprudência: Expandir o módulo de coleta para outros tribunais de justiça permitiria um trabalho futuro de pesquisa focado em comparar como diferentes estados brasileiros julgam tipos semelhantes de cibercrime, investigando possíveis variações regionais na aplicação da lei.
- Investigação de Modelos de IA Especializados: Um trabalho futuro de grande relevância seria explorar o *fine-tuning* de modelos de linguagem com um corpus de textos jurídicos do Brasil. A pesquisa poderia investigar se um modelo especializado supera a acurácia de modelos de propósito geral na classificação de processos e se é capaz de identificar nuances mais sutis nos textos.
- Estudo de Visualização de Dados para Análise Criminal: Com base na API desenvolvida, uma pesquisa futura poderia focar nas formas mais eficazes de visualização de dados (geoespaciais, temporais, redes de relacionamento) para a identificação de padrões e tendências no comportamento dos cibercrimes, contribuindo para a área de análise criminal e formulação de políticas públicas.

Referências

ADAK, S. **Spring Boot Beginner's Guide: Build Java Applications Fast with Auto-Configuration**. 2025. <https://www.codingshuttle.com/blogs/what-is-spring-boot/>. [Online; accessed 15-Setembro-2025]. Citado na página 21.

ANDRADE, E. **Uma visão geral do HTTP**. 2023. <https://www.dio.me/articles/entendendo-as-diferencas-entre-apis-rest-e-restful>. [Online; accessed 09-Setembro-2025]. Citado na página 15.

ARRUDA, V. C. D.; LIMA, M. F. **CRIMES CIBERNÉTICOS: ANÁLISE DA LEGISLAÇÃO CIVIL E PENAL VIGENTE NO BRASIL**. 2023. <https://doi.org/10.47820/recima21.v4i12.4578>. [Online; accessed 19-Novembro-2024]. Citado na página 16.

AWS. **O que é Java ?** 2025. <https://aws.amazon.com/pt/what-is/java/>. [Online; accessed 07-Setembro-2025]. Citado na página 13.

AZURE, M. **O que é o Java?** 2025. <https://azure.microsoft.com/pt-br/resources/cloud-computing-dictionary/what-is-java-programming-language>. [Online; accessed 18-Setembro-2025]. Citado na página 13.

BERNIK, I. **Cybercrime: The Cost of Investments into Protection**. 2016. <https://www.fvv.um.si/rV/arhiv/2014-2/01_Bernik.pdf>. [Online; accessed 22-Novembro-2024]. Citado na página 17.

BROADCOM. **Spring Framework**. 2025. <https://spring.io/projects/spring-framework>. [Online; accessed 08-Setembro-2025]. Citado na página 13.

CANALTECH. **24% dos brasileiros sofreram golpes digitais nos últimos 12 meses, diz pesquisa**. 2024. <https://canaltech.com.br/seguranca/24-dos-brasileiros-sofreram-golpes-digitais-nos-ultimos-12-meses-diz-pesquisa>. [Online; accessed 13-Outubro-2024]. Citado 2 vezes nas páginas 9 e 31.

CARRARO, F. **O que é o Google Gemini e o que esse modelo de IA é capaz de fazer — com exemplo prático**. 2024. <https://www.alura.com.br/artigos/google-gemini?srsltid=AfmBOooLgnYFZGSs0ZLVWjowHYpyUby7b0wBOvj9j1A6GYjqpgHm2V1->. [Online; accessed 16-Setembro-2025]. Citado na página 15.

COMMUNITY, D. **O que é o Spring framework e para que ele é usado?** 2023. <https://www.dio.me/articles/o-que-e-o-spring-framework-e-para-que-ele-e-usado>. [Online; accessed 08-Setembro-2025]. Citado na página 13.

CONTRIBUTORS, M. **Uma visão geral do HTTP**. 2025. <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Guides/Overview>. [Online; accessed 09-Setembro-2025]. Citado na página 14.

DIO. **Empresas Brasileiras que Utilizam Java em seu Desenvolvimento**. 2025. <https://www.dio.me/articles/empresas-brasileiras-que-utilizam-java-em-seu-desenvolvimento-455bfed4395c>. [Online; accessed 15-Setembro-2025]. Citado na página 21.

_____. **Empresas Brasileiras que Utilizam Java em seu Desenvolvimento**. 2025. <https://www.dio.me/articles/empresas-brasileiras-que-utilizam-java-em-seu-desenvolvimento-455bfed4395c>. [Online; accessed 15-Setembro-2025]. Citado na página 21.

ERSE, A. V. **DESENVOLVIMENTO E ANALISE DO BACKEND DO PROJETO CUIDAIDOSO**. 2025. <https://monografias.ufop.br/handle/35400000/3274>. [Online; accessed 20-Setembro-2025]. Citado na página 19.

GEMINI, G. **Gemini API reference**. 2025. <https://ai.google.dev/api>. [Online; accessed 19-Setembro-2025]. Citado na página 26.

_____. **Modelos do Gemini**. 2025. <https://ai.google.dev/gemini-api/docs/models>. [Online; accessed 19-Setembro-2025]. Citado na página 27.

GITHUB. **Sobre o GitHub e o Git**. 2025. <https://docs.github.com/pt/get-started/start-your-journey/about-github-and-git>. [Online; accessed 16-Setembro-2025]. Citado na página 15.

GRADLE. **Gradle User Manual**. 2025. <https://docs.gradle.org/current/userguide/userguide.html>. [Online; accessed 08-Setembro-2025]. Citado na página 14.

GROUP, T. P. G. D. **What is PostgreSQL?** 2023. <https://www.postgresql.org/about/>. [Online; accessed 08-Setembro-2025]. Citado na página 14.

HAT, R. **O que é uma API REST?** 2023. <https://www.redhat.com/pt-br/topics/api/what-is-a-rest-api>. [Online; accessed 09-Setembro-2025]. Citado na página 15.

HONóRIO, G.; STABILE, A.; PAIVA, D. **Estelionatos no Brasil mais que quadruplicam em cinco anos, e golpes virtuais disparam após pandemia, revela Anuário**. 2023. <https://g1.globo.com/sp/sao-paulo/noticia/2023/07/20/estelionatos-no-brasil-mais-que-triplicam-em-cinco-anos-e-golpes-virtuais-disparam-apos-pandemia-ghtml>. [Online; accessed 17-Setembro-2025]. Citado na página 9.

LAB, A. K. **O que é crime cibernético? Aprenda a como se proteger**. 2025. <https://www.kaspersky.com.br/resource-center/threats/what-is-cybercrime>. [Online; accessed 18-Setembro-2025]. Citado na página 11.

MICROSOFT. **O que é PostgreSQL?** 2025. <https://azure.microsoft.com/pt-br/resources/cloud-computing-dictionary/what-is-postgresql>. [Online; accessed 08-Setembro-2025]. Citado na página 14.

ORACLE. **O que é Java ?** 2025. <https://www.java.com/pt-BR/download/help/whatis_java.html>. [Online; accessed 07-Setembro-2025]. Citado na página 13.

PHILLIPS, K.; DAVIDSON, J. C.; FARR, R. R.; BURKHARDT, C.; CANEPPELE, S.; AIKEN, M. P. **Conceptualizing Cybercrime: Definitions, Typologies and Taxonomies**. 2022. <https://doi.org/10.3390/forensicsci2020028>. [Online; accessed 19-Novembro-2024]. Citado na página 16.

RNP. **Evolução da internet no Brasil**. 2022. <https://www.rnp.br/noticias/evolucao-da-internet-no-brasil>. [Online; accessed 13-Outubro-2024]. Citado na página 9.

RNP; CPQD. **CONHEÇA O OBSERVATÓRIO**. 2025. <https://observatorioblockchain.org.br/sobre/>. [Online; accessed 20-Setembro-2025]. Citado 2 vezes nas páginas 17 e 18.

SOFTWARE, S. **O que é Swagger**. 2025. <https://swagger.io/docs/specification/v2_0/what-is-swagger/>. [Online; accessed 10-Setembro-2025]. Citado na página 15.

TANENBAUM, M. v. S. A. S. **Distributed Systems Principles and Paradigms**. [S.l.]: Maarten van Steen, 2016. Citado na página 14.

TSAKALIDIS, G.; VERGIDIS, K. A Systematic Approach Toward Description and Classification of Cybercrime Incidents. **IEEE**, 2017. Citado 5 vezes nas páginas 12, 17, 24, 27 e 30.

VITOR, O. **O que é Flyway e por que usa-lo? Com Java e Spring!** 2024. <https://medium.com/@perez_vitor/o-que-%C3%A9-flyway-e-por-que-usa-lo-com-java-e-spring-312219ebf840>. [Online; accessed 16-Setembro-2025]. Citado na página 15.

XR4GOOD LAB; ÂMBAR; LIMP. **Cuidaidoso**. 2025. <http://cuidaidoso.net.br/>. [Online; accessed 22-Outubro-2025]. Citado na página 19.