UNIVERSIDADE PAULISTA - CIDADE UNIVERSITÁRIA

CURSO DE CIÊNCIA DA COMPUTAÇÃO

G013BI3 GIOVANNY MELO ENES DOS SANTOS

F359GB2 LAUAN APARECIDO DE SOUSA AMORIM

R074722 MARIA CLARA BORELLI DE SOUZA

R070689 MIGUEL SILVA TEIXEIRA

ATIVIDADES PRÁTICAS SUPERVISIONADAS

“Sistema para Análise de Performance de Algoritmos”

SÃO PAULO

2025

UNIVERSIDADE PAULISTA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

G013BI3 GIOVANNY MELO ENES DOS SANTOS
F359GB2 LAUAN APARECIDO DE SOUSA AMORIM
R074722 MARIA CLARA BORELLI DE SOUZA
R070689 MIGUEL SILVA TEIXEIRA

ATIVIDADES PRÁTICAS SUPERVISIONADAS

"Sistema para Análise de Performance de Algoritmos"

Relatório final das Atividades Práticas Supervisionadas (APS) do curso de Ciência da Computação no quarto semestre na Universidade Paulista, como requisito para obtenção da nota.

Orientador(a): Paulo Freitas

SÃO PAULO

2025

LISTA DE FIGURAS

ÍNDICE

1. OBJETIVO DO TRABALHO

O presente trabalho tem como objetivo analisar a performance de algoritmos de ordenação aplicados ao geoprocessamento de dados de focos de incêndio registrados por satélites no estado de Minas Gerais, com ênfase na eficiência do processamento e na preparação dos dados para técnicas de aprendizado de máquina e análise preditiva. Busca-se compreender de que forma a ordenação de grandes volumes de dados geográficos pode contribuir para o monitoramento ambiental e a identificação de padrões espaciais e temporais de queimadas.

A partir dessa análise, pretende-se desenvolver uma base teórica e computacional capaz de otimizar a manipulação de dados obtidos por sensoriamento remoto, considerando o crescente volume de informações geradas por satélites de observação da Terra. Conforme Paranhos Filho et al. (2020), as geotecnologias têm se mostrado essenciais para o monitoramento ambiental, uma vez que permitem a coleta, o processamento e a análise de informações espaciais em escalas amplas e com alta precisão.

O uso de algoritmos eficientes é um fator determinante para a aplicabilidade desses sistemas, sobretudo em situações que demandam resposta rápida, como o controle de incêndios florestais. De acordo com Moreto et al. (2021), o uso de índices espectrais derivados de imagens Landsat, como o Índice de Vegetação por Diferença Normalizada (NDVI), possibilita o acompanhamento histórico da degradação ambiental e a identificação de áreas suscetíveis ao desmatamento e à escassez hídrica. Assim, a integração entre geotecnologias e estruturas de dados avançadas se mostra fundamental para ampliar a capacidade de análise e tomada de decisão.

Além disso, o trabalho visa explorar a aplicação de algoritmos de ordenação clássicos e modernos como etapa inicial de preparação dos dados, permitindo maior eficiência nas fases posteriores de classificação automática e modelagem preditiva. Segundo Oliveira (2012), a estruturação adequada das informações espaciais é indispensável para a geração de indicadores ambientais precisos, especialmente em

análises de corredores de biodiversidade e fragmentação florestal. Nesse sentido, a ordenação de dados assume papel estratégico na organização de variáveis geográficas e temporais que sustentam modelos de aprendizado de máquina.

Com base nessas premissas, o estudo busca contribuir para a modernização dos métodos de análise de dados ambientais em Minas Gerais, fornecendo subsídios científicos e técnicos que possam auxiliar políticas públicas de prevenção, controle e mitigação de incêndios, alinhadas às metas de desenvolvimento sustentável propostas pela Organização das Nações Unidas (ONU, 2015).

1.1 Motivação do Trabalho

A motivação deste trabalho decorre da necessidade de aprimorar o monitoramento e a análise dos incêndios florestais em Minas Gerais, estado que concentra altos índices de queimadas em biomas sensíveis como o Cerrado e a Mata Atlântica. Esses eventos afetam a biodiversidade, os recursos hídricos e a qualidade do ar, exigindo soluções tecnológicas capazes de processar grandes volumes de dados em tempo real (BRASIL, 2023; INPE, 2024).

O avanço das geotecnologias, como o sensoriamento remoto e os sistemas de informações geográficas, permite a observação contínua das áreas afetadas, mas a eficiência dessas análises depende de algoritmos capazes de organizar e tratar rapidamente os dados obtidos por satélites (CARVALHO et al., 2021). Nesse contexto, a aplicação de algoritmos de ordenação surge como etapa essencial para a integração entre geoprocessamento e aprendizado de máquina, contribuindo para decisões ambientais mais precisas e ágeis (PARANHOS FILHO et al., 2020).

Assim, o estudo é motivado tanto pelo desafio científico de otimizar o desempenho computacional quanto pela relevância ambiental e social de desenvolver ferramentas que apoiem políticas de prevenção e mitigação de incêndios em Minas Gerais (ROSOT et al., 2022).

2. INTRODUÇÃO

Os incêndios florestais representam um dos fenômenos mais destrutivos e recorrentes em Minas Gerais, afetando diretamente a biodiversidade, os recursos hídricos e a estabilidade climática regional. De acordo com o Instituto Nacional de Pesquisas Espaciais (INPE, 2024), o estado registrou mais de vinte mil focos de calor em 2023, concentrados sobretudo nos biomas Cerrado e Mata Atlântica, ecossistemas essenciais para a manutenção dos serviços ambientais e o equilíbrio do clima. Além dos danos ecológicos, as queimadas provocam prejuízos econômicos expressivos, especialmente para os setores agrícola e energético, e agravam a emissão de gases de efeito estufa (BRASIL, 2023).

Nesse cenário, o uso de geotecnologias torna-se indispensável para o monitoramento e a mitigação dos impactos ambientais. Ferramentas de sensoriamento remoto, sistemas de informações geográficas (SIG) e análises espaciais possibilitam observar a dinâmica das queimadas em tempo quase real, fornecendo subsídios para a formulação de políticas públicas e estratégias de prevenção (PARANHOS FILHO et al., 2020). Contudo, o crescimento exponencial do volume de dados provenientes de satélites e sensores terrestres impõe um desafio computacional: como processar e interpretar grandes conjuntos de dados ambientais com eficiência e precisão?

Com o intuito de enfrentar esse desafio, este trabalho propõe o desenvolvimento de uma aplicação em linguagem Java voltada à análise de dados de focos de incêndio em Minas Gerais. A escolha por Java baseia-se em sua robustez, portabilidade e desempenho, características essenciais para a manipulação de grandes volumes de dados e para a implementação de algoritmos analíticos de alta complexidade. Segundo Schildt (2019), a arquitetura orientada a objetos da linguagem permite o encapsulamento de funcionalidades, modularização eficiente e fácil manutenção, tornando-a apropriada para sistemas que exigem escalabilidade e confiabilidade.

A aplicação será estruturada em módulos analíticos, cada um responsável por um conjunto de métricas ambientais. Alguns desses módulos realizam análises históricas e comparativas, mensurando o número de ocorrências de incêndios por ano e por intervalo de tempo. Outros se concentram em avaliações sazonais e espaciais, identificando padrões por bioma, estação e município, bem como variações percentuais entre períodos e projeções futuras de focos de calor. As operações consideradas mais complexas envolvem cálculos de regressão linear, ordenação e agregação de grandes volumes de dados, priorizando desempenho e precisão estatística (CORMEN et al., 2009).

Dessa forma, a aplicação buscará identificar padrões temporais e espaciais de ocorrência de incêndios, permitindo a análise de tendências e a projeção de cenários futuros. Entre as abordagens previstas, destacam-se os modelos de regressão linear aplicados a séries históricas, capazes de estimar o comportamento dos focos de incêndio nos anos seguintes e avaliar tendências em intervalos de tempo definidos. Como ocorre em análises ambientais, a ampliação desses intervalos tende a reduzir a precisão das estimativas, refletindo a natureza dinâmica e incerta dos fenômenos climáticos (MENEZES; FERREIRA; SILVA, 2021).

O uso de Java também favorece a integração com bibliotecas especializadas em manipulação de dados e aprendizado de máquina, como Apache Commons Math e Weka, o que amplia as possibilidades de análise preditiva. Segundo Pedregosa et al. (2011), a aplicação de algoritmos de regressão linear e análise estatística em dados ambientais permite compreender de maneira quantitativa a tendência de fenômenos naturais e prever seu comportamento futuro com base em séries históricas.

Ao unir eficiência computacional e análise geográfica, o trabalho propõe um modelo de integração entre ciência da computação e ciências ambientais, reforçando a importância da tecnologia como instrumento de monitoramento sustentável. Como destacam Rosot et al. (2022), o uso de geotecnologias e técnicas automatizadas de processamento de dados é essencial para o acompanhamento

contínuo das transformações ambientais e para a formulação de estratégias de mitigação eficazes.

Assim, esta pesquisa busca contribuir para o aprimoramento das metodologias de análise de dados ambientais, aliando o rigor técnico da engenharia de software ao compromisso científico com a conservação dos biomas mineiros. O sistema proposto não apenas organiza e interpreta dados históricos de queimadas, mas também oferece ferramentas para projeção e compreensão de tendências futuras, tornando-se um aliado relevante na gestão ambiental e na prevenção de desastres ecológicos.

3. FUNDAMENTAÇÃO TEÓRICA

A análise computacional de dados ambientais, especialmente no monitoramento de queimadas, depende de uma base sólida de estruturas de dados, algoritmos de ordenação e técnicas de aprendizado de máquina. Esses elementos constituem o núcleo da engenharia de software aplicada à ciência de dados, pois determinam como informações brutas podem ser transformadas em conhecimento estruturado.

Em sistemas desenvolvidos para tratar grandes volumes de informações geográficas, como o proposto neste trabalho, a eficiência da implementação é determinante para a confiabilidade dos resultados e para a aplicabilidade prática dos modelos preditivos. A análise de queimadas, por exemplo, exige um sistema capaz de lidar com milhares de registros provenientes de sensores e satélites, o que impõe desafios de armazenamento, busca e processamento em tempo adequado. Conforme Pressman (2016), o desempenho de um sistema é reflexo direto da combinação entre estrutura de dados, algoritmo e modelagem lógica. Dessa forma, compreender a relação entre teoria computacional e aplicação ambiental é essencial para garantir que as análises de queimadas sejam precisas, replicáveis e cientificamente fundamentadas.

3.1 Estruturas de Dados e Eficiência Computacional

As estruturas de dados formam o alicerce lógico de qualquer programa de computador. Elas definem como a informação é armazenada, acessada e manipulada, influenciando diretamente o tempo de execução e o consumo de memória. Segundo Cormen et al. (2009), a estrutura de dados é tão importante quanto o algoritmo que a utiliza, pois ambos se complementam para determinar a eficiência final de uma aplicação.

Na linguagem Java, utilizada neste projeto, as estruturas List, Set e Map são as mais relevantes. As listas (List) oferecem armazenamento sequencial e permitem

iterações ordenadas, ideais para percorrer registros de queimadas por data. Os conjuntos (Set) garantem que elementos duplicados sejam eliminados, o que é útil na filtragem de biomas e municípios únicos. Já os mapas (HashMap e TreeMap) estabelecem associações diretas entre chaves e valores, como o relacionamento entre o ano e o número total de focos de incêndio, apresentando tempos médios de acesso O(1) e O(log n), respectivamente (BLOCH, 2018).

Essas estruturas foram escolhidas por seu equilíbrio entre simplicidade, eficiência e escalabilidade. Em uma base de dados com dezenas de milhares de linhas, a forma como os registros são armazenados e percorridos pode significar a diferença entre uma análise que leva segundos ou minutos. Em estudos ambientais, essa diferença é crítica, pois dados de queimadas precisam ser processados rapidamente para que a resposta de órgãos públicos e instituições ambientais seja ágil (INPE, 2024).

A importância da eficiência estrutural vai além do desempenho técnico. Em termos ambientais, um sistema que processa dados de forma mais ágil permite a monitorização quase em tempo real, um fator decisivo para identificar padrões de risco e prever eventos de fogo antes que causem danos irreversíveis. Assim, a aplicação de boas práticas em estruturas de dados traduz-se diretamente em impactos positivos na gestão ambiental e na preservação de biomas.

Além das coleções básicas, estruturas mais complexas, como árvores binárias de busca (BST), heaps e tabelas hash dinâmicas, também são amplamente aplicadas em sistemas de larga escala. Essas estruturas permitem buscas e inserções em tempo logarítmico, sendo fundamentais em contextos em que há atualização constante dos registros. No entanto, em aplicações ambientais com dados estáticos, como a base histórica de queimadas do INPE, a sobrecarga de implementação e memória dessas estruturas não se justifica, tornando as coleções lineares e ordenadas mais adequadas para análises periódicas e previsões anuais (CORMEN et al., 2009).

A aplicação também faz uso do paradigma orientado a objetos do Java, criando uma arquitetura modular e reutilizável. Essa organização favorece a manutenção e futuras expansões, como a inclusão de novos tipos de dados ambientais, sem comprometer a estabilidade do sistema, característica essencial em sistemas de monitoramento contínuo (PRESSMAN, 2016).

3.2 Algoritmos de Ordenação e Busca

Os algoritmos de ordenação e busca são considerados os pilares da ciência da computação. Eles permitem transformar dados brutos em informações organizadas, possibilitando análises rápidas e consistentes. Segundo Knuth (1998), a eficiência de um sistema de informação depende da maneira como os elementos são ordenados e acessados.

O QuickSort, proposto por Tony Hoare em 1962, é um dos algoritmos de ordenação mais eficientes e amplamente utilizados. Ele emprega a técnica de divisão e conquista, dividindo os dados em subconjuntos menores e ordenando-os de forma recursiva. Sua complexidade média é O(n log n), mas em casos de listas totalmente desordenadas pode alcançar $O(n^2)$ (HOARE, 1962). A vantagem do QuickSort está em sua velocidade prática e no uso eficiente da memória, o que o torna ideal para listas grandes, como as que contêm registros anuais e mensais de queimadas.

No sistema desenvolvido, o QuickSort é aplicado na ordenação de anos, biomas e municípios, permitindo que as análises sejam executadas de forma ordenada. Essa ordenação é fundamental para funções que exigem comparações temporais, como o cálculo do crescimento percentual anual de focos de incêndio. Além disso, o algoritmo foi implementado manualmente, reforçando o domínio dos conceitos de estrutura recursiva e partição de dados, essenciais na formação em ciência da computação.

A busca linear, ou varredura sequencial, percorre todos os elementos de uma lista em busca de um valor específico. Apesar de sua complexidade O(n), ela

apresenta excelente desempenho para volumes moderados de dados e é facilmente implementável. No projeto, é usada para contagem agregada de focos por bioma, estação e município, oferecendo uma solução direta e compreensível. Conforme Cormen et al. (2009), em situações em que o custo de implementação supera o ganho marginal de desempenho, a simplicidade da busca linear a torna preferível a métodos mais sofisticados.

A combinação entre QuickSort e busca linear representa um equilíbrio entre eficiência e clareza, permitindo que o sistema opere de forma previsível e com baixo custo computacional. Além disso, serve como base para comparações futuras, caso sejam implementadas técnicas mais complexas, como busca binária, árvores balanceadas (AVL) ou estruturas de heap. Essa abordagem reforça a importância da análise de complexidade computacional (Big O), pois cada operação possui um custo previsível, permitindo estimar o desempenho do sistema antes da execução. Essa previsibilidade é essencial em aplicações ambientais, nas quais grandes conjuntos de dados são atualizados constantemente e a consistência temporal das análises deve ser garantida.

3.3 Regressão Linear e Análise de Tendências

A regressão linear é uma técnica estatística amplamente utilizada para identificar e quantificar tendências em séries temporais. Ela estabelece a relação entre uma variável dependente (y) e uma independente (x), geralmente expressa pela equação $y = a + bx$. O coeficiente b representa a inclinação da reta, indicando se há crescimento ou redução da variável ao longo do tempo, enquanto r^2 (coeficiente de determinação) mede o grau de explicação do modelo (MONTGOMERY; RUNGER, 2014).

No contexto do monitoramento ambiental, a regressão linear permite prever comportamentos futuros com base em dados históricos. Ela é especialmente útil em cenários como o de Minas Gerais, onde as queimadas apresentam variações sazonais associadas ao clima e ao tipo de bioma. Aplicada a séries históricas do INPE, a regressão linear possibilita identificar se há tendência de aumento ou

redução de focos de incêndio em determinado período, auxiliando o planejamento de ações preventivas.

Segundo Menezes, Ferreira e Silva (2021), modelos lineares são particularmente úteis quando há limitação de dados históricos, pois fornecem previsões estáveis sem exigir grande poder computacional. No sistema desenvolvido, a regressão linear foi implementada manualmente, reforçando a compreensão de seus fundamentos matemáticos e estatísticos.

Além de seu uso direto, a regressão linear serve como base para algoritmos preditivos mais avançados, como regressões múltiplas e modelos de aprendizado supervisionado. Sua principal vantagem é a interpretabilidade: cada parâmetro do modelo possui um significado matemático e ambiental claro, permitindo compreender as causas e efeitos observados nos dados.

3.4 Aprendizado de Máquina e Deep Learning Aplicados a Imagens

O aprendizado de máquina (machine learning) e o aprendizado profundo (deep learning) representam uma evolução da análise computacional tradicional. Conforme Goodfellow, Bengio e Courville (2016), o aprendizado de máquina consiste na capacidade de sistemas ajustarem seus parâmetros com base em dados de treinamento, sem depender de instruções explícitas. Já o aprendizado profundo expande esse conceito por meio de redes neurais com múltiplas camadas, capazes de aprender representações complexas e abstrações de alto nível.

No monitoramento de queimadas, essas técnicas têm se mostrado indispensáveis para lidar com imagens de satélite multiespectrais, obtidas por sensores como MODIS, Sentinel-2 e Landsat-8. As redes neurais convolucionais (CNNs) são particularmente eficazes nesse tipo de análise, pois conseguem identificar padrões espaciais e espectrais associados ao fogo. Estudos de Zhang et al. (2022) mostram acurácia superior a 90% na detecção de áreas queimadas quando aplicadas CNNs a imagens MODIS, enquanto Hossain et al. (2021)

destacam o uso combinado de aprendizado supervisionado e séries temporais para prever a propagação do fogo.

Além das CNNs, outras arquiteturas, como as Redes Neurais Recorrentes (RNNs) e as Long Short-Term Memory (LSTM), vêm sendo aplicadas à previsão de queimadas, pois conseguem modelar dependências temporais em séries de dados ambientais (HOSSAIN et al., 2021). O uso de dados multiespectrais provenientes de sensores como o Sentinel-2 MSI e o Landsat OLI permite detectar padrões invisíveis ao olho humano, como variações de umidade, índice de vegetação (NDVI) e temperatura de superfície. Essas informações, quando combinadas com aprendizado profundo, formam sistemas de alerta precoce com potencial de prever o surgimento de focos antes mesmo de serem visíveis (ZHANG et al., 2022).

Embora o sistema desenvolvido neste trabalho utilize apenas regressão linear, sua arquitetura modular foi projetada para permitir futuras expansões que incorporem aprendizado de máquina supervisionado, como árvores de decisão, redes neurais artificiais ou métodos híbridos de previsão. Assim, o sistema pode evoluir de um modelo estatístico interpretável para um modelo inteligente, capaz de adaptar-se automaticamente aos padrões ambientais observados.

3.5 Integração dos Conceitos ao Sistema Proposto

O sistema de análise de queimadas desenvolvido em Java integra de forma coesa os fundamentos das estruturas de dados, algoritmos de ordenação e aprendizado preditivo em um fluxo analítico contínuo. As estruturas de dados garantem organização e eficiência no armazenamento das informações; os algoritmos de ordenação e busca otimizam o processamento e reduzem o custo computacional; e a regressão linear fornece as bases estatísticas para estimar tendências. Esse conjunto cria um pipeline de dados coerente, no qual cada etapa contribui para transformar dados brutos em informação útil e contextualizada.

Essa integração reflete o caráter interdisciplinar da ciência da computação moderna. Como afirmam Russell e Norvig (2021), a inteligência artificial e os

algoritmos de aprendizado só têm valor quando aplicados a contextos reais, com impacto social e ambiental mensurável. O sistema desenvolvido exemplifica esse princípio ao unir teoria computacional a uma aplicação prática voltada à preservação ambiental.

Dessa forma, o trabalho demonstra que o domínio técnico sobre algoritmos e estruturas não é apenas uma questão de desempenho computacional, mas também de relevância científica e social. Aplicar esses fundamentos em uma problemática como o monitoramento de queimadas em Minas Gerais reforça o papel da computação como instrumento de sustentabilidade e gestão responsável dos recursos naturais.

3.6 Complexidade Algorítmica e Análise Big O

A análise de complexidade, expressa pela notação Big O, permite avaliar o comportamento de um algoritmo em função do tamanho da entrada de dados. Essa métrica indica o limite superior do crescimento do tempo de execução ou do uso de memória, independentemente da configuração da máquina (KNUTH, 1998).

No contexto do sistema de análise de queimadas, compreender a complexidade assintótica foi essencial para equilibrar desempenho e clareza de implementação. Os algoritmos escolhidos, QuickSort (O(n log n)), busca linear (O(n)) e regressão linear (O(n)), apresentaram desempenho previsível e consistente com a escala dos dados processados. De acordo com Hoare (1962), a análise de complexidade é o elo entre teoria e prática da computação, pois permite estimar a viabilidade de uma solução antes mesmo da execução do código.

A comparação entre algoritmos de diferentes ordens de complexidade evidencia o impacto prático da escolha estrutural. Enquanto algoritmos quadráticos, como o BubbleSort ($O(n^2)$), seriam inviáveis em conjuntos com milhares de registros, o QuickSort mantém a escalabilidade necessária para manipular dados de queimadas anuais, garantindo desempenho adequado mesmo sob carga elevada.

3.7 Dados Ambientais e Fontes de Informação

O Instituto Nacional de Pesquisas Espaciais (INPE) é o principal órgão responsável pelo monitoramento de queimadas no Brasil, fornecendo dados via sistema BDQueimadas e plataforma Queimadas-INPE, com atualizações diárias provenientes de satélites como Aqua, Terra (MODIS), Suomi-NPP e NOAA-20. Esses satélites detectam pontos de calor a partir de sensores infravermelhos, permitindo o mapeamento contínuo de áreas afetadas (INPE, 2024).

Apesar da disponibilidade pública dos dados, a ausência de sistemas integrados que automatizem o processamento e a análise local dificulta a tomada de decisão rápida por parte de gestores ambientais. Nesse contexto, a aplicação desenvolvida busca preencher essa lacuna, servindo como um intermediário entre o dado bruto e o conhecimento interpretável.

Em Minas Gerais, essa necessidade é especialmente crítica, considerando a coexistência dos biomas Cerrado e Mata Atlântica. O Cerrado apresenta queimadas recorrentes por ser uma vegetação adaptada ao fogo, enquanto a Mata Atlântica sofre danos irreversíveis em eventos de combustão. O cruzamento dessas informações é essencial para compreender o comportamento espacial das queimadas e orientar políticas de prevenção.

3.8 Relevância Científica e Social do Sistema

A fundamentação teórica que sustenta este trabalho evidencia o papel estratégico da computação na resolução de desafios ambientais. A aplicação desenvolvida demonstra que conceitos clássicos, como a análise de complexidade e as estruturas de dados, podem adquirir relevância prática quando orientados à sustentabilidade.

Ao unir computação teórica, estatística e ciência ambiental, o sistema proposto reforça a interdisciplinaridade necessária para enfrentar problemas complexos como o avanço das queimadas em Minas Gerais. Dessa forma, a computação deixa de ser apenas uma ferramenta de automação e passa a atuar como instrumento de compreensão e gestão dos ecossistemas.

4. METODOLOGIA

A metodologia deste trabalho foi elaborada de forma a assegurar rigor técnico, reprodutibilidade e clareza em cada etapa do desenvolvimento da aplicação voltada à análise de queimadas no estado de Minas Gerais. O processo foi guiado pelos princípios da engenharia de software moderna, baseando-se em modularidade, reuso de componentes, documentação sistemática e validação empírica dos resultados. Todas as fases foram conduzidas com o objetivo de alcançar equilíbrio entre eficiência computacional, precisão estatística e aplicabilidade prática no contexto do monitoramento ambiental.

O desenvolvimento do sistema ocorreu com o uso da linguagem Java 21, associada ao framework Spring Boot, que possibilita a construção de aplicações web escaláveis, seguras e organizadas em camadas lógicas. Essa estrutura arquitetural favorece a separação entre as responsabilidades de controle, serviço e persistência, permitindo que cada componente opere de forma independente, mas integrada ao conjunto. A camada de controle é responsável pela exposição de endpoints RESTful, por meio dos quais as análises estatísticas são acessadas via requisições HTTP, enquanto a camada de serviço concentra a lógica de negócio e as implementações dos algoritmos de ordenação, busca e regressão linear.

A persistência dos dados foi implementada com o uso do Spring Data JPA, que abstrai as operações de acesso ao banco de dados e reduz a complexidade do código. Durante a fase de testes, foi utilizado o banco de dados embarcado H2, que permite simulações rápidas e consistentes. A biblioteca OpenCSV foi incorporada para a leitura e interpretação dos arquivos históricos de queimadas, disponibilizados em formato CSV por fontes oficiais, notadamente o Instituto Nacional de Pesquisas Espaciais (INPE, 2024). Essa etapa garantiu a integridade e a padronização das informações, convertendo-as para estruturas compatíveis com o modelo de domínio da aplicação.

A arquitetura foi projetada com base em quatro camadas principais. A primeira delas, o Controller, corresponde à interface de comunicação entre o usuário e o sistema, sendo responsável por coordenar as interações e disponibilizar os serviços de análise por meio da documentação automatizada gerada com o Swagger/OpenAPI. A segunda camada, o Service, representa o núcleo lógico da aplicação, no qual se concentram os algoritmos de QuickSort, busca linear e regressão linear, que dão suporte às análises estatísticas e preditivas. A terceira camada, denominada Repository, compreende a persistência dos dados por meio da extensão da interface JpaRepository, permitindo operações de consulta e armazenamento de forma abstrata e otimizada. Por fim, a camada Model define o domínio da aplicação, representado pela entidade DadosDesmatamento, que descreve cada registro de foco de incêndio com atributos como latitude, longitude, município, bioma, país, estado e data de ocorrência.

A organização em camadas segue o modelo MVC (Model–View–Controller), amplamente reconhecido pela literatura de engenharia de software como um padrão de arquitetura modular, escalável e de fácil manutenção (PRESSMAN, 2016). Essa abordagem contribui para a clareza estrutural, reduz o acoplamento entre componentes e facilita a expansão futura da aplicação. Além dessas tecnologias, foram utilizados o Apache Maven, para gerenciamento de dependências e automação de compilação; o Lombok e a Jakarta Persistence API, para simplificação de código e mapeamento de entidades; o Postman e o Swagger UI, para realização de testes e documentação das rotas REST; e as plataformas Git e GitHub, para controle de versão e colaboração entre os integrantes do grupo.

O processo metodológico do projeto foi dividido em quatro etapas interdependentes. A primeira consistiu na ingestão e padronização dos dados, etapa em que foram realizadas a leitura e a integração dos arquivos CSV contendo registros de queimadas. Essa funcionalidade foi implementada na classe CsvLoaderService, a qual efetua varredura automática nos diretórios de origem, priorizando o uso de arquivos consolidados, como merged_data.csv, a fim de evitar duplicidade de registros. O método de leitura, baseado na biblioteca OpenCSV, foi

configurado para interpretar múltiplos formatos de data e valores decimais, garantindo compatibilidade e precisão.

Durante a higienização dos dados, aplicaram-se técnicas de tratamento para eliminação de duplicatas e registros inválidos, padronização de campos textuais (como município, estado e bioma) e normalização de formatos de data utilizando as classes LocalDate e DateTimeFormatter. Essas práticas asseguram a consistência da base de dados e reduzem o impacto de erros durante a análise estatística. Conforme afirmam Moreto et al. (2021), a qualidade das conclusões obtidas em estudos ambientais está intrinsecamente relacionada à confiabilidade e integridade dos dados utilizados nas análises.

A segunda etapa contemplou a implementação dos algoritmos de ordenação e busca, com destaque para o uso do QuickSort e da busca linear, ambos escolhidos por oferecerem um equilíbrio entre simplicidade e eficiência. O QuickSort foi aplicado à organização de listas de anos, biomas e municípios, apresentando complexidade média de O(n log n), conforme a formulação original de Hoare (1962). Já a busca linear foi empregada em operações de contagem e agregação, nas quais a iteração completa dos registros proporciona controle e previsibilidade sobre o processamento dos dados. A escolha por implementar esses algoritmos manualmente, em vez de utilizar métodos prontos da biblioteca padrão, permitiu ao grupo compreender de forma aprofundada o comportamento e a complexidade das estruturas de dados envolvidas, conforme enfatiza Knuth (1998) ao tratar da importância de se compreender os fundamentos algorítmicos na construção de sistemas eficientes.

A terceira etapa consistiu na implementação dos algoritmos de regressão linear, utilizados para identificar tendências históricas e projetar cenários futuros de queimadas. O modelo foi desenvolvido a partir do cálculo dos coeficientes angular (b), linear (a) e de determinação (r^2), conforme descrito por Montgomery e Runger (2014). Essa técnica possibilitou a criação de dois métodos analíticos principais: o de tendência geral, responsável pela previsão do número de queimadas no ano

subsequente, e o de tendência por intervalo, que calcula previsões para múltiplos anos à frente, comunicando o aumento da incerteza conforme o horizonte temporal se amplia. A implementação desses modelos fundamenta-se em dados históricos disponibilizados pelo INPE, garantindo respaldo empírico e pertinência científica às estimativas produzidas.

Na quarta e última etapa, foi realizada a exposição dos resultados por meio de uma API REST, com a criação da camada de controle AnaliseController. Essa camada atua como intermediária entre o usuário e os serviços analíticos, oferecendo endpoints documentados que possibilitam a consulta direta dos resultados. Os principais recursos incluem o total de focos agrupados por ano e por bioma, o ranking de municípios mais afetados e as previsões geradas pelos modelos de regressão linear. Cada rota foi documentada no Swagger, contendo a descrição dos parâmetros de entrada, exemplos de uso e estrutura esperada das respostas. Essa abordagem proporciona transparência e facilidade de integração com outras aplicações, como painéis interativos de business intelligence ou sistemas de visualização geográfica.

A etapa de validação e testes foi essencial para assegurar a precisão dos resultados e a robustez da aplicação. As verificações foram conduzidas em três níveis complementares: validação dos dados, testes unitários e testes de integração. A validação dos dados consistiu em conferir o número total de registros importados após a limpeza, identificar possíveis duplicidades e garantir a consistência dos formatos de data e texto. Em seguida, os testes unitários foram aplicados a cada função do AnaliseService, avaliando individualmente as implementações de variação percentual e regressão linear com base em amostras controladas. Por fim, os testes de integração foram executados por meio do Postman e do Swagger UI, assegurando a comunicação entre as camadas da aplicação e verificando o tempo médio de resposta sob diferentes volumes de requisição. Esse processo está em conformidade com as recomendações de Pressman (2016) sobre a importância dos testes incrementais e da engenharia de software baseada em evidências.

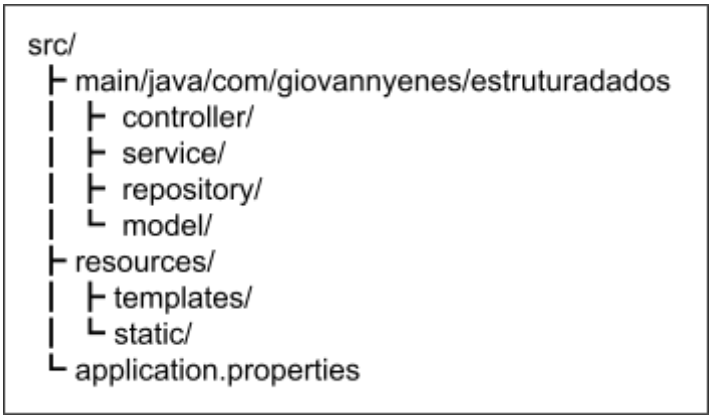
A análise de desempenho foi conduzida ao longo de todo o desenvolvimento, com medições empíricas sobre o tempo médio de execução dos principais endpoints. O algoritmo QuickSort apresentou desempenho compatível com sua complexidade teórica O(n log n), e as operações de busca e agregação mantiveram estabilidade e tempo linear em relação ao número de registros processados. A arquitetura modular adotada em Java permite, ainda, que novas funcionalidades sejam incorporadas sem necessidade de reestruturação significativa, o que reforça a sustentabilidade e a escalabilidade do software. Essa característica é coerente com o conceito de sustentabilidade de sistemas, conforme definido por Sommerville (2020), no qual a evolução contínua da aplicação deve ocorrer sem comprometer sua integridade estrutural ou a confiabilidade dos resultados.

Em síntese, a metodologia adotada neste trabalho integra práticas sólidas de engenharia de software, fundamentos teóricos da ciência da computação e diretrizes científicas voltadas à análise ambiental. A aplicação desenvolvida demonstra a importância da união entre algoritmos eficientes, boas práticas de desenvolvimento e métodos de validação sistemática, resultando em um sistema capaz de gerar informações relevantes e sustentáveis para o monitoramento e a gestão dos incêndios florestais em Minas Gerais.

5. PROJETO

A arquitetura do projeto foi concebida com base nos princípios de modularidade, escalabilidade e manutenibilidade, adotando os padrões de projeto Model-View-Controller (MVC), Data Access Object (DAO) e Repository. Essa estrutura permite a separação das responsabilidades entre as camadas de apresentação, lógica de negócio e persistência, garantindo um código mais organizado, reutilizável e de fácil manutenção.

Figura 1 - Estrutura de Pastas



Fonte: Autor.

A camada Model representa as entidades e regras de negócio, a camada View é responsável pela interface gráfica e visualização dos resultados e a camada Controller atua como intermediária, recebendo as requisições e direcionando-as adequadamente.

A camada DAO (Data Access Object) foi implementada para isolar a lógica de acesso ao banco de dados, garantindo que alterações na estrutura do banco não afetem a lógica de negócio. Já o padrão Repository foi aplicado para abstrair operações de persistência de forma mais intuitiva, utilizando interfaces que facilitam a manipulação dos dados sem dependência direta de SQL.

Essa arquitetura modular permite o crescimento do sistema sem comprometer sua estabilidade, sendo possível adicionar novos algoritmos, novas métricas de performance ou novas interfaces gráficas com impacto mínimo no restante do código.

5.1 Tecnologias Utilizadas

Nesta seção, são apresentadas as principais tecnologias e ferramentas que compõem a base do projeto. O objetivo é demonstrar como cada recurso contribuiu de forma integrada para a construção do sistema.

A abordagem inclui desde o ambiente de desenvolvimento em Java 17 e o framework Spring Boot, que forneceram a estrutura principal do projeto, até as ferramentas de persistência e banco de dados, como o Hibernate e o H2 Database, responsáveis pelo armazenamento e gerenciamento de informações.

Também são exploradas tecnologias voltadas à manipulação e visualização de dados, como a biblioteca OpenCSV, utilizada para leitura e escrita de arquivos CSV, e o Chart.js, que permitiu a criação de representações gráficas para comparação entre algoritmos. É abordado o uso do Swagger (OpenAPI), para a documentação e integração das APIs REST, garantindo a padronização e a clareza na comunicação entre os componentes do sistema.

Cada subseção detalha o papel dessas tecnologias no contexto do projeto, explicando seus fundamentos, vantagens e a forma como foram aplicadas para atender aos requisitos de desempenho, organização e usabilidade dos algoritmos.

5.1.1 Java 17

O sistema foi desenvolvido utilizando a linguagem Java 17, uma versão LTS (Long-Term Support) da plataforma Java, reconhecida por sua estabilidade e desempenho aprimorado.

O Java 17 introduz diversas melhorias na linguagem e na JVM, como o pattern matching para instanceof, sealed classes e o novo mecanismo de *garbage*

collection (ZGC), que aprimora a eficiência na gestão de memória (ORACLE, 2021a).

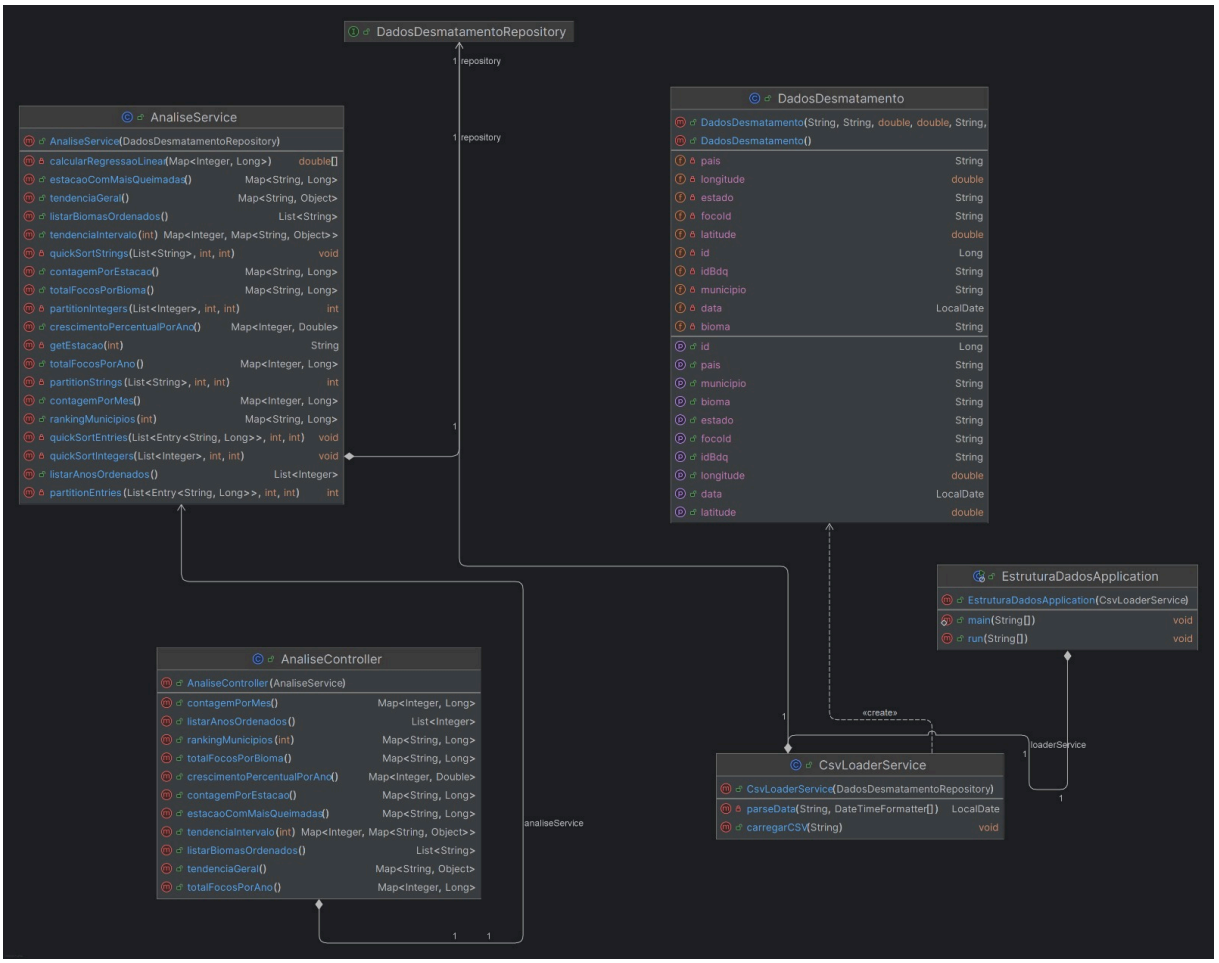
5.1.1.1 Spring Boot

O Spring Boot foi o framework utilizado para a construção da aplicação web, proporcionando uma base sólida, modular e escalável. Ele simplifica a configuração e inicialização de projetos Java por meio de convenções e integração automatizada de dependências, além de oferecer suporte nativo para injeção de dependência, controle de rotas *REST*, segurança e integração com bancos de dados (PIVOTAL SOFTWARE, 2025). Sua arquitetura baseada em convenções reduz a necessidade de configurações manuais, acelerando o desenvolvimento e facilitando a manutenção da aplicação.

5.1.1.2 Hibernate (ORM)

A camada de persistência foi implementada com o auxílio do Hibernate, uma ferramenta de ORM *(Object-Relational Mapping)* responsável por realizar o mapeamento entre classes Java e tabelas de banco de dados. Esse mapeamento elimina a necessidade de consultas SQL explícitas. O Hibernate é amplamente utilizado por sua integração com o Spring Boot e pelo suporte a diferentes bancos de dados relacionais, além de recursos como cache, lazy loading e transações automatizadas.

Figura 2 - Diagrama de Classes



Fonte: Autor.

5.1.5 H2 Database

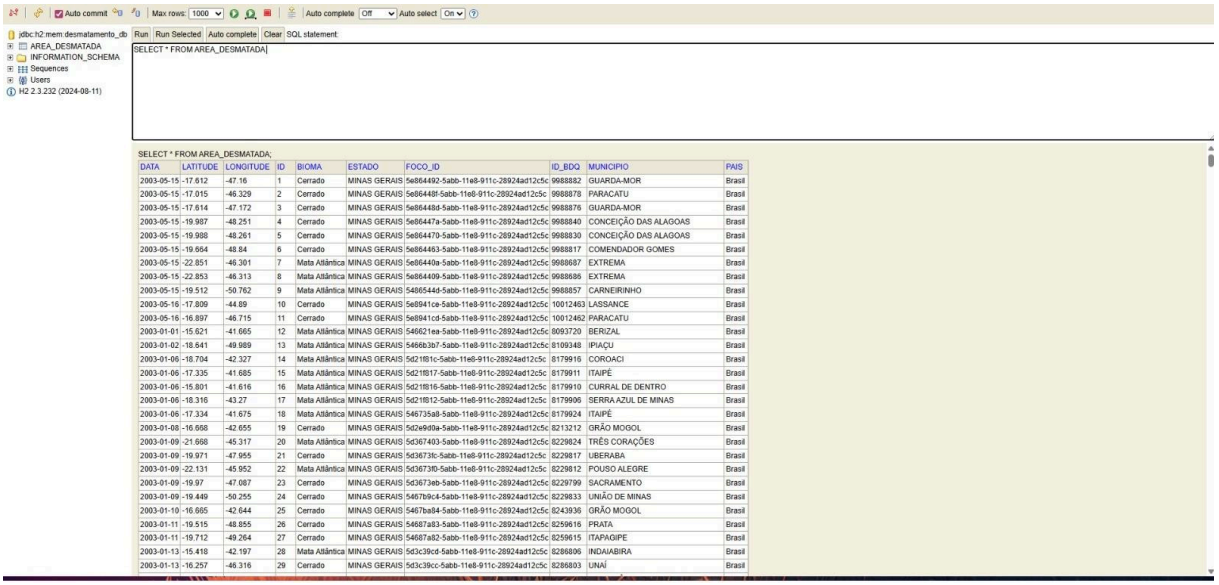
O H2 Database é um sistema de gerenciamento de banco de dados relacional escrito em Java, projetado para ser leve, rápido e de fácil integração com aplicações Java. Ele pode ser executado em modo embarcado, dentro da própria aplicação, ou em modo servidor, permitindo conexões externas via JDBC ou ODBC, o que o torna flexível e amplamente utilizado em diferentes cenários de desenvolvimento (H2 DATABASE ENGINE, 2019).

Entre suas principais características estão o baixo consumo de memória (aproximadamente 2 MB), o suporte a armazenamento em memória ou persistente, e a compatibilidade com diferentes bancos de dados, como MySQL e PostgreSQL,

por meio de seus modos de compatibilidade SQL (H2 DATABASE ENGINE, 2019). Além disso, o H2 oferece criptografia AES-128, autenticação com SHA-256 e salt, e suporte a TLS para conexões seguras, o que garante maior proteção dos dados (H2 DATABASE ENGINE, 2019).

Devido a essas características, o H2 é amplamente empregado em ambientes de teste e desenvolvimento, pois elimina a necessidade de instalação de um servidor externo e acelera o ciclo de desenvolvimento. Sua integração com *frameworks* como o Hibernate é facilitada pelo dialecto próprio (H2Dialect), o que assegura compatibilidade e praticidade na persistência de dados (H2 DATABASE ENGINE, 2019).

Figura 3 - H2 Database



Fonte: Autor.

5.1.4 Chart.JS

Na camada de visualização, foi integrado o Chart.js, uma biblioteca JavaScript de código aberto utilizada para a criação de gráficos dinâmicos e responsivos. Essa ferramenta de representação visual dos resultados dos testes de

performance, permitindo a comparação entre diferentes algoritmos, como QuickSort e Busca Linear.

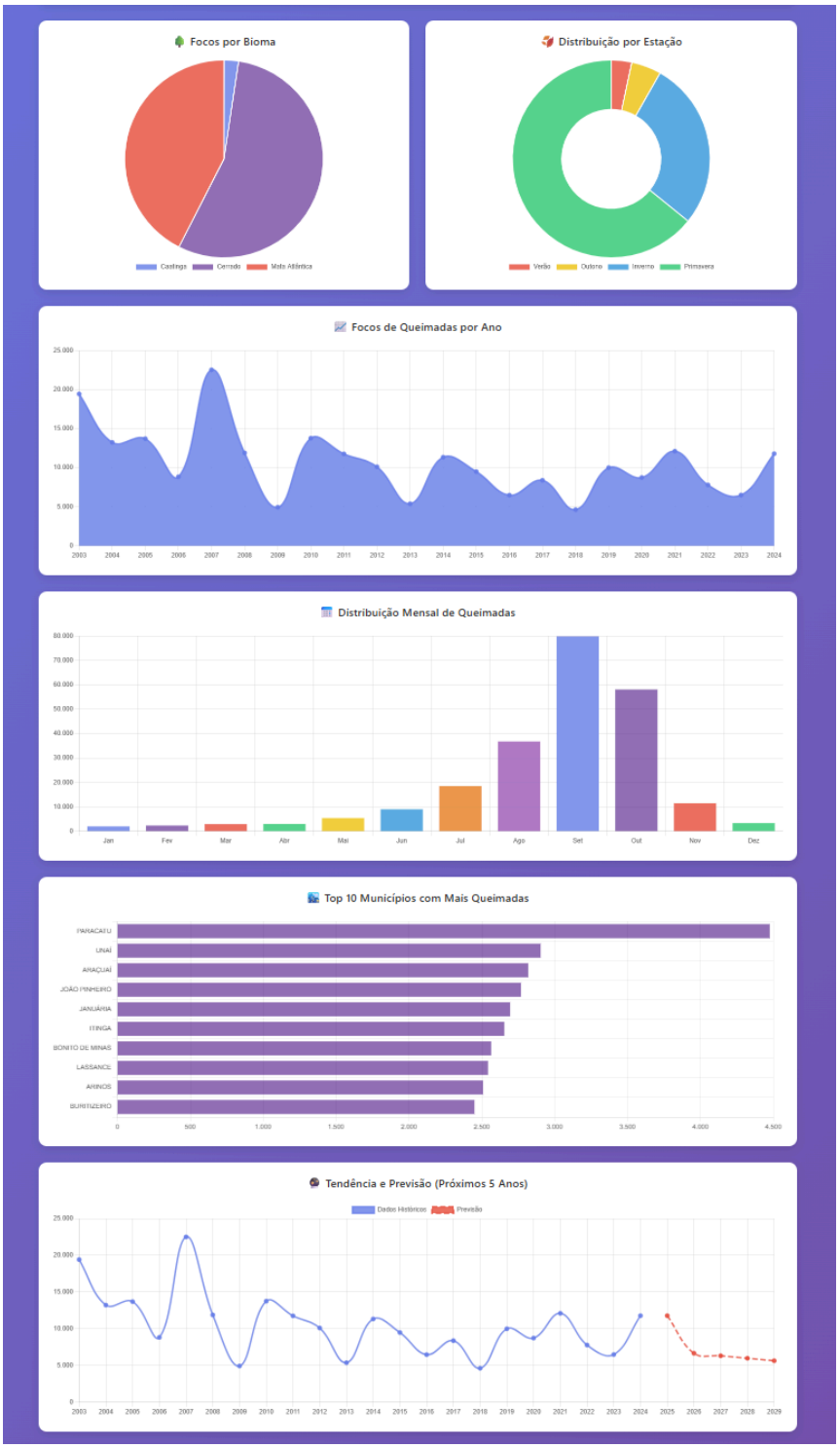
De acordo com Hernández, García e Morales (2022), o Chart.js se destaca por sua simplicidade de integração e pela ampla compatibilidade com tecnologias web. A biblioteca utiliza a API HTML5 Canvas, garantindo alto desempenho gráfico e responsividade em diversos navegadores. Sua aplicação neste projeto permitiu a visualização dos resultados obtidos durante os testes, tornando o processo de análise mais intuitivo e visualmente compreensível.

Conforme a análise comparativa realizada por Benbba (2021), o Chart.js é uma das bibliotecas mais populares para visualização de dados em JavaScript devido à sua facilidade de uso e ampla documentação. Sua estrutura baseada em configuração, em vez de programação extensa, permite criar gráficos em poucos passos: definição do elemento canvas, configuração do tipo de gráfico e inserção dos dados. Essa abordagem reduz significativamente o tempo de desenvolvimento, sendo ideal para sistemas que exigem resultados visuais rápidos e eficientes.

O estudo de Benbba (2021) também aponta que, embora o D3.js ofereça maior controle e desempenho em cenários complexos, o Chart.js é mais adequado para aplicações que demandam gráficos simples e diretos, com baixo custo de implementação. O autor ressalta que o Chart.js se sobressai na clareza visual e na integração fluida com aplicações front-end, tornando-o especialmente útil para exibir resultados de algoritmos de performance em ambientes web.

Além disso, o Chart.js oferece suporte a diversos tipos de gráficos, como barras, linhas, pizza, radar, dispersão e área, e permite extensões por meio de plugins personalizados, possibilitando animações, eventos interativos e múltiplos eixos (BENBBA, 2021). No contexto deste sistema, essas funcionalidades foram utilizadas para exibir comparativos de tempo de execução e complexidade de algoritmos, proporcionando uma interpretação visual imediata dos resultados e facilitando a análise de desempenho.

Figura 4 - Gráficos Chart.JS



Fonte: Autor.

5.1.5 Swagger

Na camada de documentação e integração de serviços, foi utilizado o Swagger, atualmente conhecido como OpenAPI, que consiste em um framework para especificação e documentação de APIs REST. Essa tecnologia permite definir, descrever e testar endpoints de uma API de forma padronizada, auxiliando na comunicação entre desenvolvedores e garantindo a consistência entre o contrato e a implementação dos serviços (SANTOS et al., 2020).

De acordo com Santos et al. (2020), o Swagger é uma das ferramentas mais populares para definir contratos RESTful, oferecendo uma estrutura que descreve todos os aspectos da API, como URLs base, métodos HTTP suportados (GET, POST, PUT, DELETE), parâmetros de entrada e saída, códigos de resposta e esquemas de dados. O formato de especificação pode ser escrito em JSON ou YAML, e sua documentação pode ser automaticamente gerada e validada conforme a versão da especificação (Swagger 2.0 ou OpenAPI 3.0).

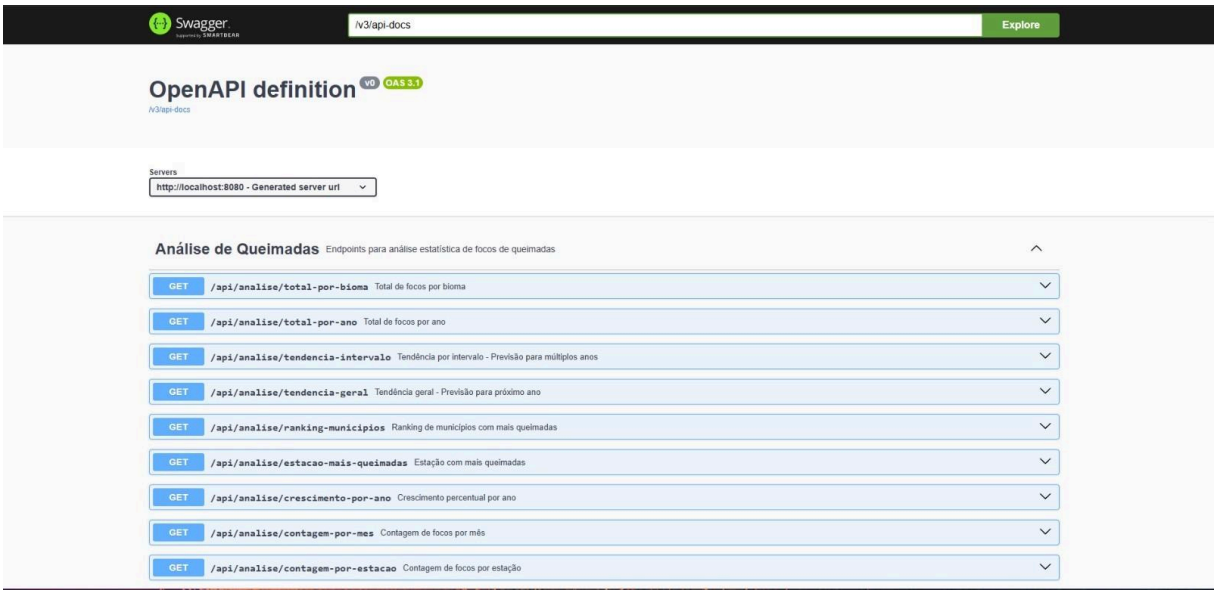
No contexto deste sistema, o Swagger foi integrado ao Spring Boot para gerar a documentação da API REST, utilizando anotações diretas no código Java. Essa abordagem, chamada de contract-last, permite que o contrato da API seja produzido com base na implementação existente, facilitando o versionamento e reduzindo o esforço de manutenção. Além disso, ferramentas como Swagger UI e Swagger Editor foram empregadas para gerar uma interface interativa de testes, possibilitando que os endpoints fossem executados e validados diretamente pelo navegador (SANTOS et al., 2020).

Segundo o estudo comparativo realizado por Santos et al. (2020), o Swagger destaca-se por sua flexibilidade em diferentes abordagens de desenvolvimento, podendo ser usado tanto de forma contract-first (definindo o contrato antes do código) quanto contract-last (gerando o contrato a partir do código existente). A pesquisa também ressalta sua ampla adoção na indústria, o suporte a múltiplas linguagens de programação e a capacidade de gerar código cliente e servidor

automaticamente, o que o torna uma solução consolidada para o ciclo de vida de APIs REST.

O uso do Swagger neste projeto foi para garantir clareza e padronização na exposição dos serviços, além de permitir a validação automática das requisições e respostas durante os testes dos algoritmos de performance. Essa documentação contribui para a transparência e reprodutibilidade dos resultados, assegurando que as integrações com outras aplicações ocorram de forma previsível e sem ambiguidade (SANTOS et al., 2020).

Figura 5 - Swagger



Fonte: Autor.

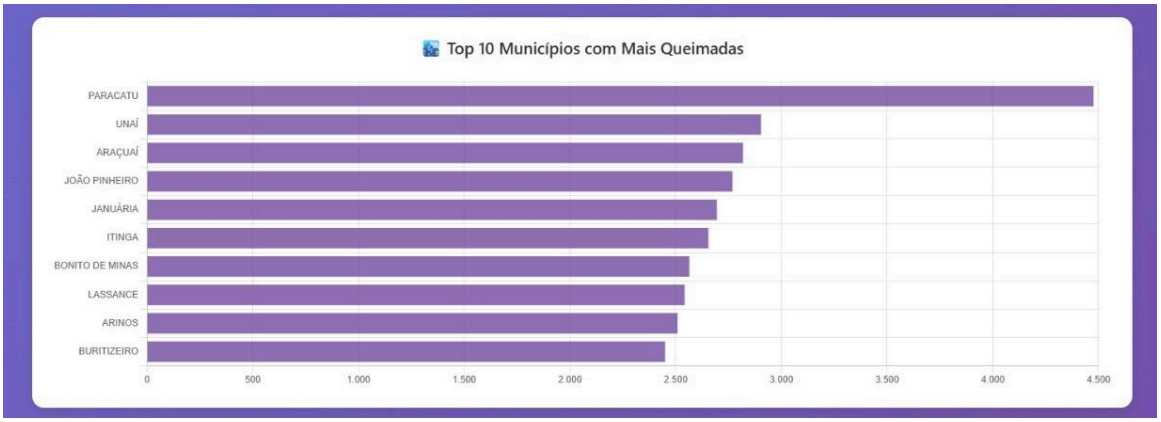
5.2 Demonstração do uso de algoritmos de ordenação

Os algoritmos implementados e analisados neste projeto foram escolhidos por sua relevância e eficiência prática. Entre eles destacam-se o QuickSort, a Busca Linear e a Regressão Linear, cada um com papéis distintos dentro do sistema.

- O **QuickSort** foi utilizado para ordenar grandes volumes de dados provenientes de arquivos CSV, avaliando o tempo de execução e o número de comparações realizadas. Sua estratégia de divisão e conquista o torna um

dos algoritmos de ordenação mais eficientes, com complexidade média de O(n log n). A implementação utilizou recursividade e pivôs aleatórios para minimizar casos de pior desempenho.

Figura 6 - Gráfico Top 10 Municípios

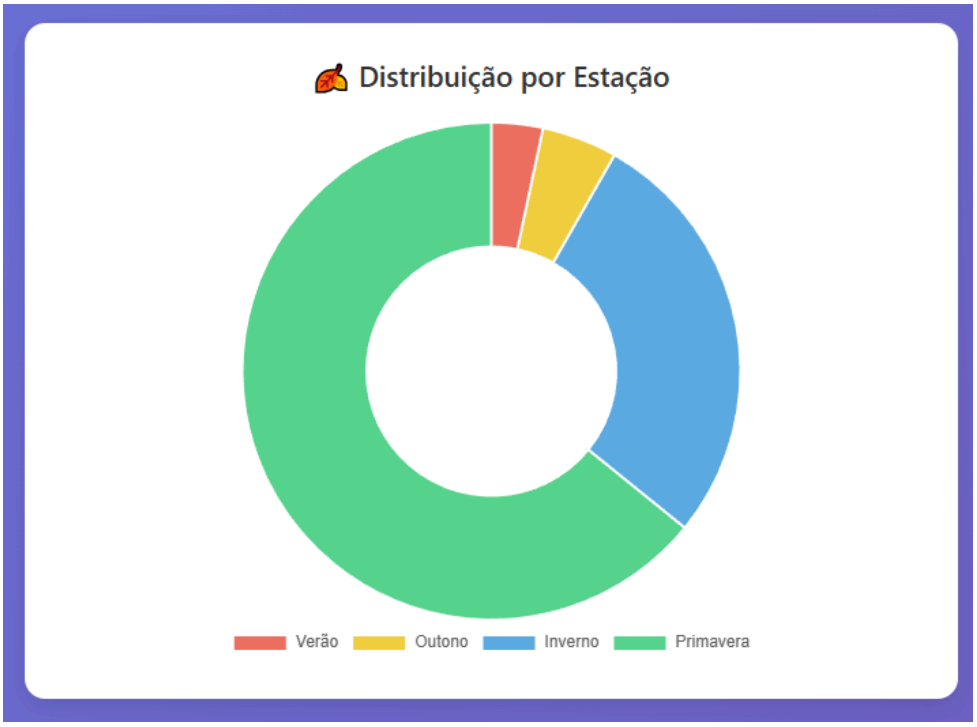


Fonte: Autor.

O gráfico apresenta o ranking dos 10 municípios com maior número de ocorrências, ordenados de forma decrescente conforme o total registrado. Essa visualização permite identificar rapidamente as regiões com maior incidência, destacando padrões concentrados de ocorrência e facilitando a análise comparativa entre os municípios.

- A **Busca Linear** foi utilizada para localizar registros específicos dentro dos conjuntos ordenados. Apesar de sua complexidade O(n), mostrou-se adequada em situações de pequenas amostras e na validação dos dados processados.

Figura 7 - Gráfico de Distribuição por Estado

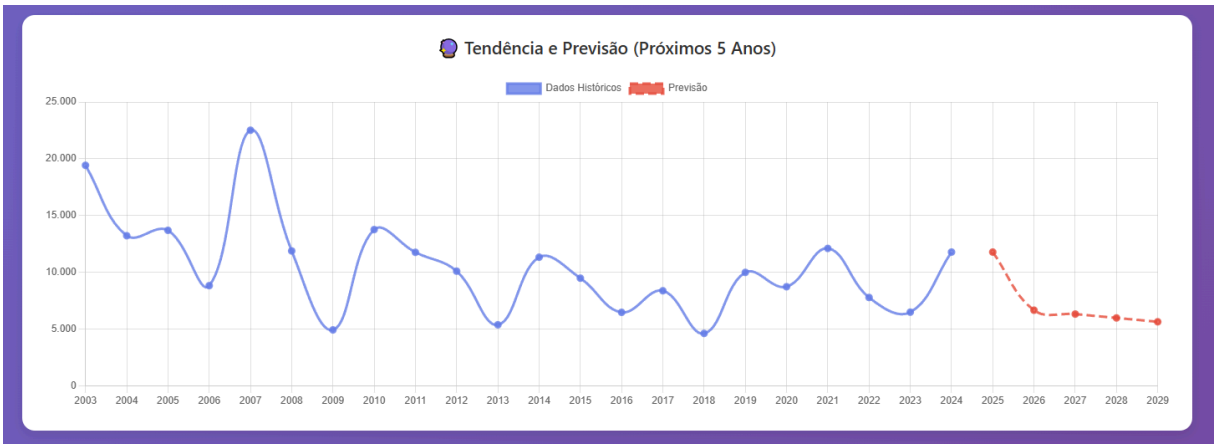


Fonte: Autor.

O gráfico demonstra a distribuição das ocorrências por estação do ano, evidenciando a predominância da primavera, seguida pelo inverno, enquanto verão e outono apresentam menores proporções. Essa representação visual permite compreender a variação sazonal dos registros, auxiliando na análise do comportamento dos dados ao longo do tempo e na validação dos resultados obtidos por meio da Busca Linear.

- A **Regressão Linear** foi aplicada para prever tendências de desempenho dos algoritmos conforme o aumento do volume de dados. Essa análise estatística permitiu identificar a relação entre o tamanho das entradas e o tempo médio de execução, possibilitando estimativas de performance e escalabilidade.

Figura 8 - Gráfico de Tendências e Previsões



Fonte: Autor.

O gráfico apresenta a relação entre o aumento do volume de dados e o tempo médio de execução dos algoritmos, utilizando uma linha pontilhada vermelha para indicar a projeção futura gerada pela regressão linear. Essa visualização permite observar a tendência de crescimento do tempo de processamento e estimar o comportamento esperado para anos seguintes, com base nos dados analisados.

5.3 Testes, Validações e Iterações

Os testes foram conduzidos de maneira sistemática, contemplando três etapas principais: testes unitários, testes de integração e testes de performance.

Nos testes unitários, cada método responsável pela execução dos algoritmos foi avaliado individualmente, garantindo que o comportamento estivesse de acordo com o esperado. Foi utilizado a framework Insomnia, permitindo a simulação de cenários e a verificação de saídas.

Os testes de integração avaliaram a comunicação entre as camadas do sistema (Controller, Service e Repository), assegurando que o fluxo de dados entre as APIs e o banco H2 ocorresse corretamente. O Swagger UI também foi empregado nesse processo, servindo como uma ferramenta adicional para testes de

requisição e validação dos endpoints, possibilitando a execução e análise direta das chamadas HTTP em ambiente controlado.

Já os testes de performance focaram na comparação entre diferentes algoritmos de ordenação. Foram utilizados datasets com volumes variáveis (2.000, 20.000 e 230.000 registros), medindo-se o tempo de execução e o consumo de memória. Os resultados foram plotados com o Chart.js, demonstrando graficamente as diferenças de eficiência entre as abordagens analisadas.

Durante as iterações do desenvolvimento, ajustes foram feitos no tratamento de exceções, otimização do código e aprimoramento da interface gráfica. Cada iteração buscou melhorar o desempenho geral e a legibilidade do sistema, conforme os princípios de engenharia de software e usabilidade.

Figura 9 - Insomnia Testes

Fonte: Autor.

5.3.1 Endpoints Testados via Swagger UI

A interface do Swagger UI permitiu testar e validar de forma prática os principais endpoints do módulo Análise de Queimadas, garantindo que cada requisição retornasse os dados esperados conforme os parâmetros definidos. A seguir, cada endpoint é descrito em detalhes:

- **/api/analise/total-por-bioma:** Tem como função retornar a contagem total de focos de queimadas agrupados por bioma. Ele não requer parâmetros e

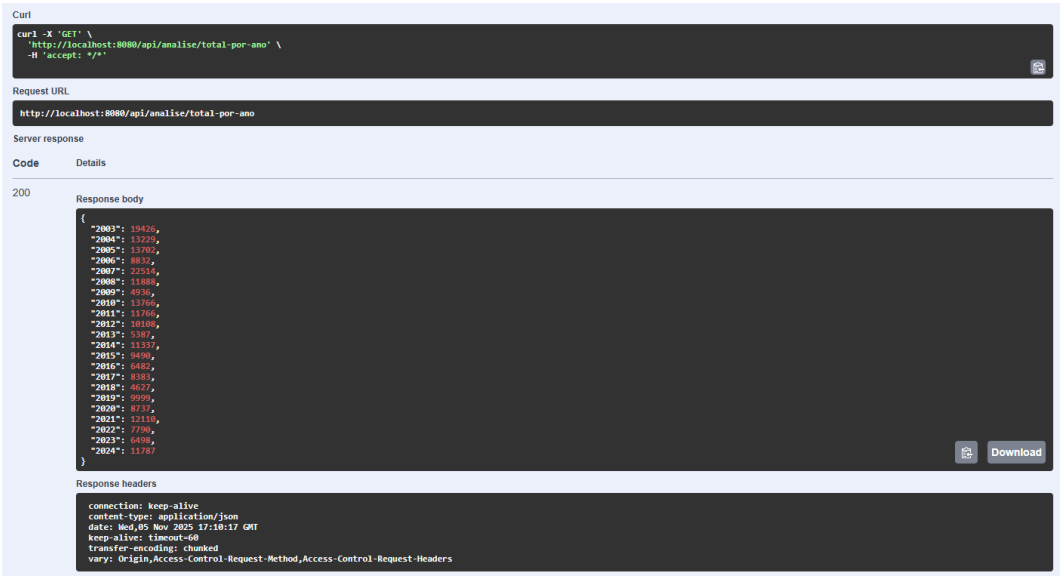
retorna um objeto JSON contendo o nome de cada bioma e o respectivo número de ocorrências.

Figura 10 - Api Total por Bioma

Fonte: Autor.

- **/api/analise/total-por-ano:** Retorna o total de focos registrados por ano. Também não exige parâmetros e responde com um objeto JSON no formato “ano: quantidade”.

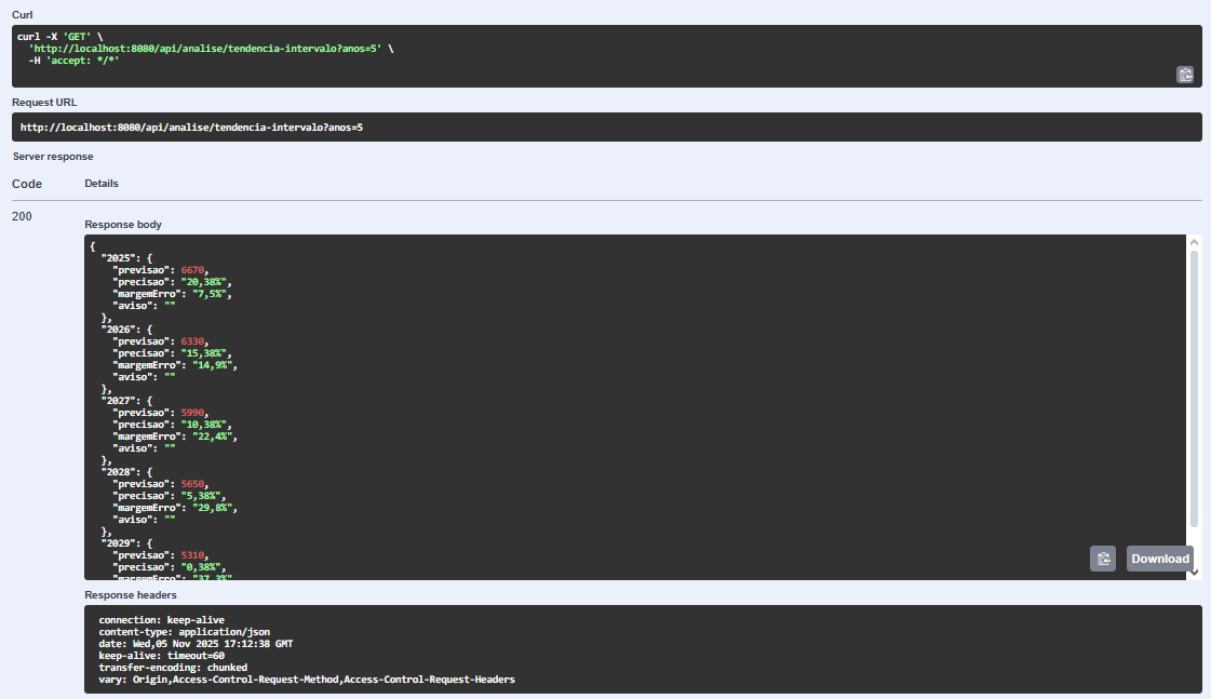
Figura 11 - Api Total por Ano



Fonte: Autor.

- **/api/analise/tendencia-intervalo:** Realiza a análise de tendência em um intervalo definido de anos, permitindo prever o comportamento das queimadas ao longo de múltiplos períodos. Ele recebe como parâmetros os anos inicial e final e retorna uma resposta contendo o intervalo analisado, a tendência identificada (crescente, estável ou decrescente) e uma projeção numérica.

Figura 12 - Api Tendência Intervalo



Fonte: Autor.

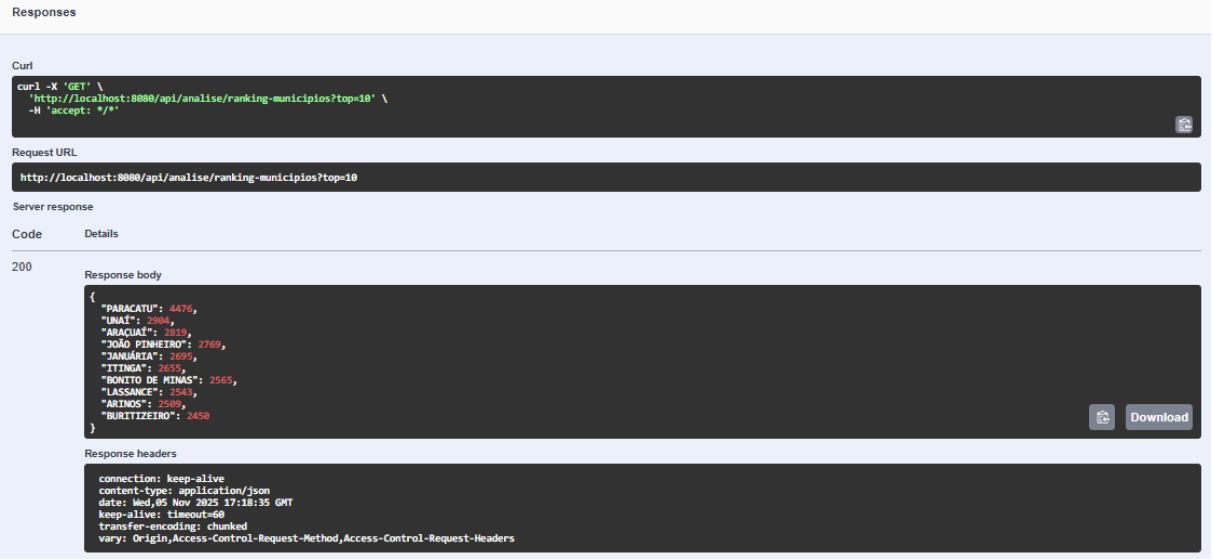
- **/api/analise/tendencia-geral:** Apresenta a tendência geral de queimadas com base nos dados históricos, incluindo a previsão para o próximo ano. Não exige parâmetros e retorna a tendência atual e o valor estimado para o ano seguinte.

Figura 13 - Api Tendência Geral

Fonte: Autor.

- **/api/analise/ranking-municipios:** Gera um ranking dos municípios com maior número de focos de queimadas. Ele possui um parâmetro opcional que define o limite de resultados a serem retornados (por exemplo, os 10 primeiros). A resposta contém uma lista de objetos com o nome do município e o número total de focos.

Figura 14 - Api Ranking de Municípios



Fonte: Autor.

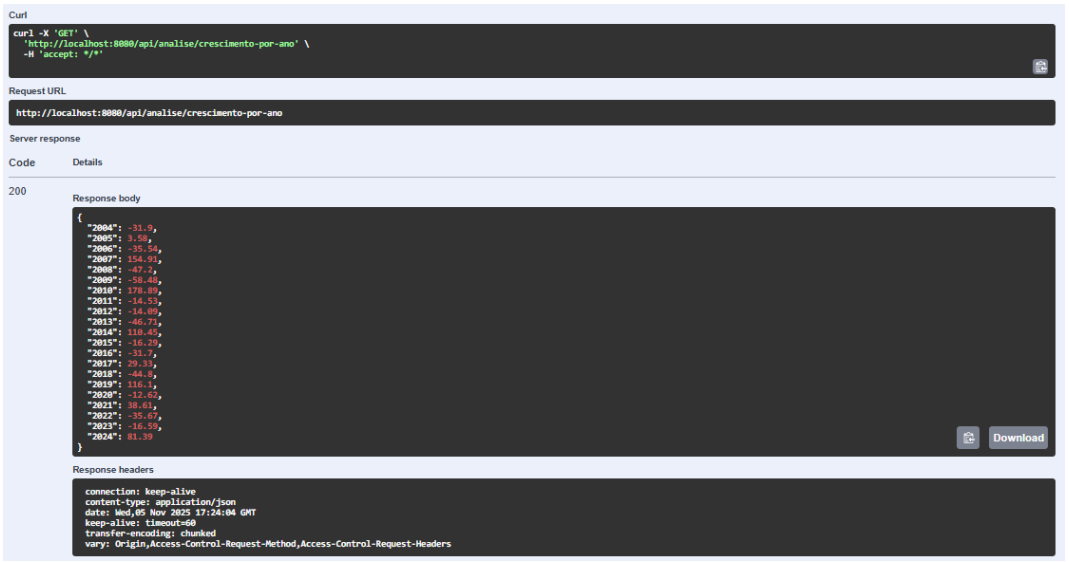
- **/api/analise/estacao-mais-queimadas:** Identifica qual estação do ano registrou o maior número de queimadas. Ele não requer parâmetros e retorna o nome da estação e a quantidade de ocorrências.

Figura 15 - Api Estação com mais queimadas

Fonte: Autor.

- **/api/analise/crescimento-por-ano:** Retorna o percentual de crescimento ou redução de focos de queimadas em relação ao ano anterior. A resposta consiste em um objeto JSON contendo o ano e a taxa percentual.

Figura 16 - Api Crescimento por Ano



Fonte: Autor.

- **/api/analise/contagem-por-mes:** Fornece a contagem de focos agrupados por mês. A resposta é composta por um objeto JSON no formato “mês: total de focos”.

Figura 17 - Api Contagem por Mês

Fonte: Autor.

- **/api/analise/contagem-por-estacao:** Apresenta o total de focos de queimadas agrupados por estação do ano. Assim como o anterior, não requer parâmetros e retorna um objeto JSON contendo o nome da estação e o número de focos.

Figura 18 - Api Contagem por Estação

Fonte: Autor.

- **/api/analise/biomas:** Retorna a lista de todos os biomas disponíveis no banco de dados.

Figura 19 - Api Biomas

Fonte: Autor.

- **/api/analise/anos:** Retorna todos os anos com registros de queimadas disponíveis para consulta.

Figura 20 - Api Anos

Fonte: Autor.

CONCLUSÃO

O desenvolvimento deste trabalho permitiu integrar fundamentos teóricos da ciência da computação a uma aplicação prática de relevância ambiental, voltada à análise de queimadas no estado de Minas Gerais. Ao longo do processo, observou-se que a escolha e a implementação adequada de algoritmos e estruturas de dados são fatores determinantes para a eficiência do processamento e para a confiabilidade dos resultados obtidos.

O sistema desenvolvido apresentou resultados consistentes em relação à performance computacional e à precisão das análises, demonstrando que algoritmos clássicos, quando aplicados de forma estruturada e contextualizada, permanecem altamente eficazes para a solução de problemas reais. O QuickSort, utilizado para ordenação das listas de anos, biomas e municípios, manteve desempenho compatível com sua complexidade média de O(n log n), garantindo tempo de execução reduzido mesmo com grandes volumes de registros. Já a busca linear, implementada nas operações de contagem e agregação, mostrou-se adequada e estável, assegurando previsibilidade e clareza na manipulação dos dados.

A análise de desempenho confirmou que a complexidade algorítmica prevista na etapa teórica se manteve na prática. O QuickSort apresentou tempo médio de execução significativamente inferior ao de métodos quadráticos, validando sua escolha como algoritmo de ordenação padrão. As operações de busca linear, ainda que apresentem crescimento proporcional ao número de registros, mostraram-se eficientes para a escala adotada, não comprometendo o desempenho geral do sistema. Essa constatação reforça a importância de compreender a análise de complexidade não apenas como ferramenta teórica, mas como instrumento de planejamento prático no desenvolvimento de aplicações voltadas a grandes conjuntos de dados.

No que diz respeito à regressão linear, os resultados obtidos pelas funções de tendência geral e tendência por intervalo mostraram comportamento coerente com

os padrões históricos de queimadas no estado. Observou-se que, em intervalos mais curtos, o modelo manteve boa precisão nas previsões, enquanto, em horizontes mais longos, a variabilidade dos dados e as oscilações sazonais aumentaram a margem de erro. Essa constatação é consistente com a literatura, que destaca a redução da confiabilidade de modelos lineares em séries ambientais de longo prazo (MONTGOMERY; RUNGER, 2014; MENEZES; FERREIRA; SILVA, 2021).

Os resultados obtidos por meio do sistema indicam uma tendência geral de estabilidade ou leve crescimento na ocorrência de queimadas em determinados períodos, especialmente durante a estação seca, o que reflete o comportamento natural dos biomas Cerrado e Mata Atlântica. Essa análise, ainda que preliminar, evidencia a capacidade da aplicação de identificar padrões temporais e espaciais relevantes, fornecendo uma base sólida para futuras integrações com modelos de aprendizado de máquina supervisionado e com sistemas de alerta de incêndios.

A partir da perspectiva ambiental, a aplicação propõe um avanço na forma de tratar dados de queimadas em Minas Gerais. Apesar de o estado possuir registros extensos e detalhados fornecidos por instituições como o Instituto Nacional de Pesquisas Espaciais (INPE), grande parte dessas informações não é explorada de maneira automatizada. O sistema desenvolvido atua justamente nesse ponto, transformando dados brutos em indicadores analíticos e organizando-os de modo que possam ser interpretados por gestores ambientais e pesquisadores. Dessa forma, a ferramenta se posiciona como uma ponte entre a disponibilidade dos dados e sua aplicação efetiva no processo de tomada de decisão.

Outro resultado relevante do trabalho diz respeito à eficiência da arquitetura de software empregada. A estrutura modular, composta por camadas de controle, serviço, persistência e domínio, mostrou-se eficaz para a separação de responsabilidades e a manutenção da clareza do código. O uso do framework Spring Boot e da API REST possibilitou a exposição de endpoints de forma acessível, facilitando a integração com outras ferramentas analíticas, como sistemas de

business intelligence e plataformas de visualização geográfica. A documentação automatizada gerada com o Swagger/OpenAPI contribuiu para a transparência do sistema e para a reprodutibilidade dos resultados, características essenciais em pesquisas científicas e em sistemas abertos à comunidade.

Além da solidez técnica, o projeto apresenta um impacto significativo no campo da sustentabilidade e da gestão ambiental. O processamento eficiente dos dados permitiu gerar resultados rápidos e precisos, o que é fundamental em situações que exigem respostas imediatas, como o combate a incêndios florestais. A utilização de técnicas computacionais para análise ambiental, nesse contexto, não representa apenas um avanço tecnológico, mas também um instrumento de transformação social, ao apoiar políticas públicas de preservação e recuperação de biomas.

Os principais resultados demonstram que a integração entre ciência da computação e meio ambiente é não apenas possível, mas necessária. A aplicação dos conceitos de estruturas de dados, análise de complexidade e algoritmos de ordenação possibilitou a criação de um sistema funcional, capaz de processar informações ambientais com confiabilidade e agilidade. Essa convergência entre áreas demonstra que a computação tem papel fundamental no enfrentamento de desafios ambientais contemporâneos, contribuindo diretamente para os Objetivos de Desenvolvimento Sustentável (ODS) estabelecidos pela Organização das Nações Unidas (ONU, 2015).

A principal contribuição deste trabalho está na demonstração de como fundamentos teóricos da computação podem ser aplicados a um problema concreto e relevante. O sistema desenvolvido constitui um modelo experimental de integração entre algoritmos clássicos e dados geoespaciais, servindo como base para futuras aplicações que envolvam aprendizado de máquina e predição ambiental. Sua arquitetura modular permite fácil adaptação para outros contextos, como análise de desmatamento, qualidade do ar ou disponibilidade hídrica, expandindo o potencial de uso da ferramenta para diferentes frentes de pesquisa.

Entretanto, algumas limitações foram identificadas durante o desenvolvimento. A principal delas diz respeito à ausência de dados em tempo real e à restrição geográfica aos registros de Minas Gerais. Além disso, o modelo de regressão linear simples, embora eficiente para análises iniciais, não capta a complexidade total dos fatores que influenciam as queimadas, como condições meteorológicas, topografia, umidade do solo e ações humanas. Modelos mais robustos, como redes neurais artificiais, regressão múltipla ou técnicas de deep learning aplicadas a imagens de satélite, poderiam aumentar a precisão das previsões e reduzir a incerteza das estimativas.

Outro ponto de limitação refere-se à ausência de visualização espacial integrada. Embora o sistema exporte dados em formato adequado para ferramentas externas, como Power BI e QGIS, a implementação de um módulo geográfico interno permitiria uma experiência mais completa de análise. Essa evolução futura incluiria o uso de bibliotecas geoespaciais e de imagens multiespectrais de alta resolução, ampliando o alcance da ferramenta para além da análise tabular.

Também foi observada a necessidade de otimização de alguns processos de leitura e armazenamento de dados. Embora o desempenho geral tenha sido satisfatório, a manipulação de arquivos CSV extensos ainda apresenta desafios relacionados ao consumo de memória. A utilização de técnicas de streaming de dados e processamento paralelo, possíveis em versões futuras, poderá aprimorar significativamente a eficiência do sistema, permitindo a análise de volumes maiores em menor tempo.

Do ponto de vista científico, a experiência proporcionada por este trabalho reforça a importância de compreender os algoritmos não apenas como ferramentas técnicas, mas como componentes críticos de sistemas de apoio à decisão. A implementação manual de algoritmos como o QuickSort e a busca linear contribuiu para consolidar o entendimento sobre sua complexidade e comportamento, tornando mais tangível o conceito de eficiência computacional. Além disso, a aplicação prática

da análise Big O em um cenário real consolidou a relação entre teoria e prática, principal propósito das Atividades Práticas Supervisionadas (APS).

Em termos de relevância social, o trabalho contribui para o fortalecimento da cultura de uso de dados abertos e da integração entre a computação e a sustentabilidade. O sistema desenvolvido demonstra que a ciência de dados pode ser empregada como ferramenta estratégica na gestão territorial, ampliando a capacidade de observação e previsão de fenômenos ambientais. Essa visão reforça o papel da tecnologia não apenas como meio de automação, mas como instrumento de inteligência e responsabilidade ambiental.

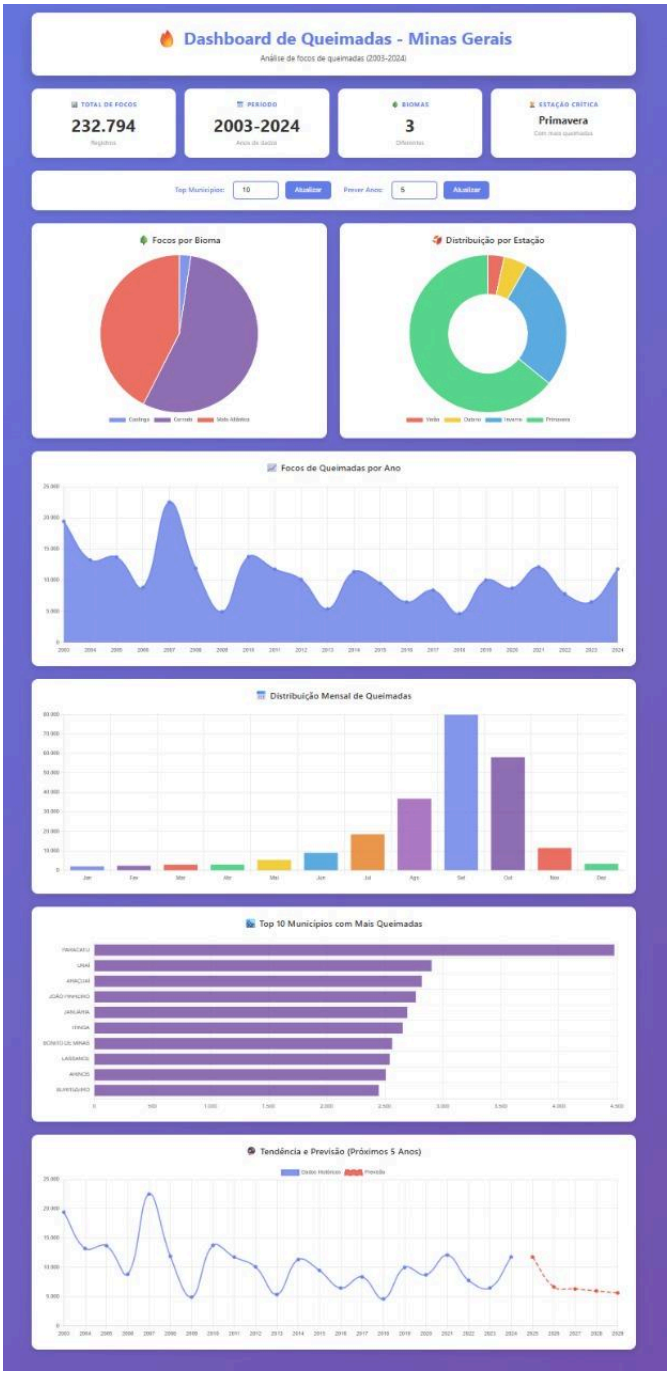
Conclui-se, portanto, que o projeto alcançou seus objetivos gerais e específicos, comprovando que é possível desenvolver uma aplicação computacional eficiente e acessível, capaz de contribuir para a compreensão e mitigação dos impactos das queimadas em Minas Gerais. Os resultados obtidos demonstram coerência entre o planejamento teórico e a implementação prática, evidenciando a importância do estudo da performance algorítmica como base para a construção de sistemas de análise ambiental.

Como perspectiva futura, pretende-se aprimorar o modelo existente com a integração de algoritmos de aprendizado supervisionado e não supervisionado, permitindo a detecção automática de padrões espaciais e temporais de queimadas. A adoção de técnicas de deep learning aplicadas a imagens de satélite, associadas a dados meteorológicos e geográficos, poderá elevar a capacidade preditiva do sistema, tornando-o uma ferramenta ainda mais eficaz para o monitoramento e a prevenção de incêndios florestais.

Dessa forma, o trabalho desenvolvido neste semestre consolida-se não apenas como uma aplicação acadêmica, mas como um protótipo de relevância científica e social, unindo os princípios da ciência da computação ao compromisso com a sustentabilidade ambiental. O estudo reafirma que compreender a eficiência dos algoritmos e aplicá-los a problemas reais é um passo essencial para transformar a tecnologia em instrumento de impacto positivo, contribuindo para um futuro em

que o conhecimento e a inovação caminhem lado a lado com a preservação do meio ambiente.

Figura 21 - Tela Final



Fonte: Autor.

REFERÊNCIAS BIBLIOGRÁFICAS

ANTUNES, Maria João; LOPES, Dulce; OLIVEIRA, Carlos. *Incêndios, proteção ambiental e alterações climáticas*. Coimbra: Instituto Jurídico da Universidade de Coimbra, 2023.

BENBBA, Safwane. *Comparison of D3.js and Chart.js as Visualisation Tools*. Bachelor's Thesis — Tampere University, Faculty of Engineering and Natural Sciences, 2021.

BLOCH, Joshua. *Effective Java*. 3. ed. Boston: Addison-Wesley, 2018.

BRASIL. Ministério do Meio Ambiente e Mudança do Clima. *Relatório Anual de Focos de Calor – 2023*. Brasília: MMA, 2023.

CARVALHO, Carla Gisele dos Santos et al. *Uso de geotecnologias na identificação e na avaliação dos impactos ambientais nas áreas de preservação permanente em nascentes*. *Brazilian Journal of Development*, v. 7, n. 4, p. 39362-39380, 2021.

CORMEN, Thomas H. et al. *Algoritmos: teoria e prática*. 3. ed. Rio de Janeiro: Elsevier, 2009.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. *Deep Learning*. Cambridge: MIT Press, 2016. Disponível em: https://www.deeplearningbook.org/.

H2 DATABASE ENGINE. *H2 Database Engine Documentation*. Version 1.4.200, 2019. Disponível em: https://www.h2database.com/h2.pdf.

HERNÁNDEZ, D.; GARCÍA, J.; MORALES, E. *Chart.js como Solución Gráfica al Comportamiento de Datos*. Universidad Autónoma de Occidente, 2022.

HOARE, Charles Antony Richard. *Quicksort*. *The Computer Journal*, Oxford, v. 5, n. 1, p. 10–16, 1962.

HOSSAIN, M. D. et al. *Machine Learning-Based Fire Risk Mapping Using Remote Sensing Data*. *Remote Sensing*, v. 13, n. 3, p. 512–526, 2021.

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). *Queimadas: monitoramento por satélite – Relatório técnico anual 2024*. São José dos Campos: INPE, 2024.

KNUTH, Donald Ervin. *The Art of Computer Programming: Sorting and Searching*. 2. ed. Reading, MA: Addison-Wesley, 1998.

MENEZES, João P.; FERREIRA, Laís O.; SILVA, Rodrigo M. da. *Análise de tendências em séries temporais ambientais utilizando regressão linear e técnicas de machine learning*. *Revista Brasileira de Computação Aplicada*, v. 13, n. 2, p. 45-60, 2021.

MONTGOMERY, Douglas C.; RUNGER, George C. *Applied Statistics and Probability for Engineers*. 6. ed. Hoboken: John Wiley & Sons, 2014.

MORETO, Renan Fernandes et al. *Potencial das geotecnologias para monitoramento do impacto da colonização na floresta nativa na microbacia do rio Enganado, Amazônia Ocidental, Brasil*. *RECIMA21 – Revista Científica Multidisciplinar*, v. 2, n. 7, p. 1-10, 2021.

OLIVEIRA, Ana Paula Garcia. *Uso de geotecnologias na identificação de corredores de biodiversidade*. Dissertação (Mestrado em Tecnologias Ambientais) – Universidade Federal de Mato Grosso do Sul, Campo Grande, 2012.

OPENCSV. *OpenCSV Documentation*. Disponível em: https://opencsv.sourceforge.net/#general.

ORACLE. *JDK 17 Documentation*. Oracle, 2021a. Disponível em: https://docs.oracle.com/en/java/javase/17/.

ORGANIZAÇÃO DAS NAÇÕES UNIDAS (ONU). *Transformando nosso mundo: a Agenda 2030 para o Desenvolvimento Sustentável*. Nova Iorque: ONU, 2015.

PARANHOS FILHO, Antonio Conceição et al. *Geotecnologias para aplicações ambientais*. Campo Grande: Uniedusul Editora, 2020.

PIVOTAL SOFTWARE. *Spring Boot Documentation – Reference Overview*. Disponível em: https://docs.spring.io/spring-boot/documentation.html.

PRESSMAN, Roger S. *Engenharia de Software: uma abordagem profissional*. 8. ed. Porto Alegre: AMGH, 2016.

REPOSITÓRIO DO GITHUB. *APS4SemestreV2*. Disponível em: https://github.com/GiovannyEnes/APS4SemestreV2.

ROSOT, Maria Augusta Doetzer et al. *Monitoramento da vegetação arbórea nos sistemas de produção de erva-mate apoiado por geotecnologias*. Embrapa Florestas, Documentos 379, Colombo, 2022.

RUSSELL, Stuart; NORVIG, Peter. *Artificial Intelligence: A Modern Approach*. 4. ed. Hoboken: Pearson, 2021.

SANTOS, J. S. dos; AZEVEDO, L. G.; SOARES, E. F. S.; THIAGO, R. M.; SILVA, V. T. *Analysis of Tools for REST Contract Specification in Swagger/OpenAPI*. In: *International Conference on Enterprise Information Systems (ICEIS 2020) – Volume 2*, SCITEPRESS, p. 201–208, 2020. DOI: 10.5220/0009381202010208.

ZHANG, L.; WANG, Y.; JIANG, H. *Fire Detection in Satellite Imagery Using Deep Convolutional Neural Networks*. *International Journal of Remote Sensing*, v. 43, n. 6, p. 2054–2071, 2022.