MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

UMA PROPOSTA DE PORTAIS DE DADOS ABERTOS AMBIENTAIS

CHRYSTIAN MENDES FRANKLIN

chrystian.franklin@fatec.sp.gov.br

Relatório de Iniciação Científica do programa PIBIC, orientado pelo Dr. Eugenio Sper de Almeida

INPE

São José dos Campos

2024

AGRADECIMENTOS

Primeiramente, desejo agradecer ao meu professor e orientador, Dr. Eugenio Sper de Almeida, por me guiar e encorajar durante as diferentes etapas percorridas.

Sou grato também aos desenvolvedores das linguagens e bibliotecas utilizadas pela disponibilização dos softwares e da documentação compreensiva. Essa dedicação ao desenvolvimento de código livre e de fonte aberta tem deixado um impacto profundo na comunidade.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), pela disponibilização e auxílio financeiro desta bolsa de iniciação científica dentro do programa PIBIC/CNPq/INPE (Processo 145472/2023-0).

E ao Instituto Nacional de Pesquisas Espaciais (INPE) pela disponibilização dos dados de plataformas de coleta de dados e boias, oportunidade de estudo e certificações.

Este trabalho foi possibilitado graças a essas contribuições.

RESUMO

O Sistema Integrado de Dados Ambientais (SINDA) do Instituto de Instituto Nacional de Pesquisas Espaciais (INPE) gerencia dados georreferenciados de plataformas de coleta de dados (PCDs) e de boias marítimas do projeto *Prediction and Research Moored Array in the Tropical Atlantic* (PIRATA), coletando, armazenando e disponibilizando os dados. Nos processos de pesquisa PIBIC/CNPq anterior, Chrystian Mendes Franklin (processo 165501/2022-7), que teve como objetivo o desacoplamento da plataforma, efetuando melhorias de performance e construindo uma API para o acesso dos dados permitindo que esses sejam filtrados e parametrizados. O foco deste projeto foi estender as capacidades da API e analisar a interface construída na etapa anterior, efetuando modificações para manter a consistência visual e consertando problemas de responsividade, adicionando mecanismos de buscas e filtros. Durante o desenvolvimento do projeto, houveram lançamentos de novas versões da linguagem de programação Python, com melhorias de performance, *threading* e também das bibliotecas utilizadas. Portanto, os requerimentos do projeto foram atualizados de acordo com as versões disponíveis, com a criação de um pacote em python do software com pyproject para facilitar a instalação e execução do projeto. Ainda considerando os desafios com incompatibilidades de abordagens utilizadas dentro do software e os frequentes períodos de instabilidade do serviço do SINDA, foi adicionado um mecanismo de atualização periódica dos dados, de maneira que mesmo em períodos de indisponibilidade, os dados pré-existentes conseguem ser mantidos em dia sem requerer a acesso direto ao serviço ou a reinicialização do software. Houveram também melhorias no tratamento dos dados, com processos de limpeza de parâmetros inválidos, permitindo que os dados sejam mantidos mais compactos sem perder validade. Obteve-se como resultado a extensão com a documentação da API construída na etapa anterior e a reconstrução da interface gráfica com melhor responsividade e experiência de usuário, contando também com o uso de modos de interface clara e escura. Atingindo o objetivo principal do projeto, e possibilitando melhorias de disponibilidade, armazenamento de dados, e hospedagem do serviço e organização do projeto.

Palavras-chave: SINDA. Dados georreferenciados. Análise de dados. API. Aplicação Web. Experiência de usuário

ABSTRACT

The Integrated Environmental Data System (SINDA) from the Brazilian National Institute for Space Research (INPE) manages georeferenced data from data collection stations (PCDs) and marine buoys from the Prediction and Research Moored Array in the Tropical Atlantic (PIRATA) project, collecting, storing and distributing data. In the previous PIBIC/CNPq research processes, Chrystian Mendes Franklin (process 165501/2022-7) had a goal of decoupling the platform, improving the performance and building an API for the unified access of data, allowing those to be filtered by parameters. The focus of this project was to extend the capabilities of the API and to analyze the interface constructed previously, performing modifications to keep the visual consistency and fixing responsiveness problems, whilst also adding mechanisms for searching and filtering. During the development of the project, there were releases of more recent versions of the Python programming language with improvements in performance, threading and also in how the libraries were used. Therefore, the project requirements were updated in accordance to the available versions, with the creation of a python package of the software to facilitate the installation and execution of the project. In addition to this, considering the challenges with incompatibilities in the approaches taken in the software and the frequent periods of instability of the SINDA service, a mechanism of periodic updates was developed, in a way that even in such periods of unavailability, the pre-existing data can be kept up-to-date, without the need of direct access to the remote service or the restart of the system. There were also improvements in the way we handle data, by cleaning invalid parameters, allowing data to be kept in a compact format without losing validity. The result is the extension of the previously constructed API, with descriptive documentation and the reconstruction of the graphical interface with better responsiveness and user experience, whilst also featuring the possibility of choice between light and dark interface modes. Concluding the main objective of the project, and allowing for improvements in the behavior in regards to storage of data, availability and hosting of services and the project's organization.

Keywords: SINDA. Georeferenced data. Data analysis. Api. Web Application. User experience.

LISTA DE FIGURAS

Pág

SUMÁRIO

Pág

1 INTRODUÇÃO

Com o crescente tópico de mudanças climáticas e a ocorrência de eventos extremos que estão acometendo o planeta e os seus impactos na fauna, flora e infraestruturas urbanas, o uso de dados ambientais por meio de portais abertos torna-se fundamental para estabelecer uma adaptação climática (FORD et al., 2016). A construção desses portais tem como objetivo a disponibilização de dados e de informações estatísticas relevantes de maneira transparente, especialmente no setor público (LNENICKA, NIKIFOROVA, 2021).

O Instituto Nacional de Pesquisas Espaciais (INPE) tem como um de seus serviços o Sistema Integrado de Dados Ambientais (SINDA) que gerencia e disponibiliza dados de plataformas de coletas de dados (PCDs), que contém dados agrometeorológicos, hidrológicos e de temperatura, juntamente com as boias marítimas do projeto *Prediction and Research Moored Array in the Tropical Atlantic* (PIRATA), projeto colaborativo entre o Brasil, França e Estados Unidos para o monitoramento de dados de interação da atmosfera e do oceano. Atualmente contempla dados desde 1993, com mais de três milhões e cem mil linhas de dados.

Esse projeto é uma continuação dos seguintes processos anteriores, Ramon Brandi da Silva (processo 144538/2020-2), Jean Cavalcante Ribeiro (processo 160729/2021-1) e Chrystian Mendes Franklin (processo 165501/2022-7), que resultaram na construção da versão inicial da plataforma, que foi desacoplada e segmentada com melhorias de interoperabilidade e performance, com a construção de uma *Application Programming Interface* (API).

Levando em consideração as melhorias realizadas anteriormente com o desacoplamento do sistema, da criação da API e das melhorias de performance de acesso aos dados, esta etapa se caracteriza pela extensão das capacidades da API e da adaptação da interface antiga para efetuar o uso desses avanços, com melhor documentação, transparência e a

presença de informações estatísticas sobre as disponibilidades dos dados visando a democratização do acesso aos dados.

Para melhorar a documentação das rotas disponibilizadas pela API, foram documentados os parâmetros passados a cada rota, os dados enviados e devolvidos, segmentado a partir do tipo de dado tratado, permitindo operações de adição, consulta, edição e remoção de dados sobre regiões geográficas relevantes, das PCDs e boias disponíveis, em conjunto com seus respectivos dados em diferentes formatos.

Após análise da interface, foram observados alguns pontos de melhorias de responsividade, onde o carregamento das telas do mapa interativo com tempo de resposta de aproximadamente dez segundos, e dos dados e gráficos das plataformas, podendo variar de cinco segundos até minutos, dependendo da quantidade de dados presentes e da disponibilidade do serviço, que sofre frequentes e longos períodos de indisponibilidades, afetando negativamente a experiência do usuário final e portanto necessitando de alterações de funcionalidade para garantir melhor fluidez e interatividade.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Portais de Dados Abertos

De acordo com a definição da *Open Knowledge Foundation*, dados abertos podem ser utilizados de maneira livre e gratuita, sem restrições de criadores, localização ou propósito. Os portais de dados abertos utilizam esse conceito como base, trazendo em conjunto informações relevantes como análises estatísticas e gráficos informativos e interativos, garantindo a transparência e acessibilidade dos dados (JANSSEN et al., 2017).

O acesso às plataformas pelos usuários é realizado em acesso a websites, servidores FTP, ou por ferramentas especializadas. Dentro do escopo deste projeto, é considerado

como formas de acesso principais o website, que podem sofrer com experiências de usuários negativas – dificultando o entendimento dos dados contidos na plataforma (SILVA; ALMEIDA, 2022) - e a API construída para comunicação direta - que requer documentação de qualidade devido a estratificação dos dados transmitidos.

2.2 Dados Ambientais

Os dados georreferenciados utilizados são caracteristicamente ambientais, entre os dados agrometeorológicos e hidrológicos coletados, se destacam a coleta de temperaturas atmosféricas, terrestres e marítimas, de precipitação e umidade do ar, de pressão atmosférica e dados barométricos, de níveis de radiação solar e de voltagens de fontes de energia. Esses dados são coletados por sensores, que são equipamentos especializados incluídos nas PCDs e boias (MAGINA, 2007). Sendo dados que por natureza crescem gradativamente em complexidade, tamanho e resolução, permitindo análises mais extensas e requerendo maior atenção e validação para efetuar a sintetização de pontos de informação aos escopos relevantes (GIBERT et al., 2018).

2.3 Armazenamento e Disponibilidade de Dados

Existem várias tecnologias direcionadas aos métodos de armazenamento de dados, em especial as tecnologias de gerenciamento de banco de dados relacionais e não relacionais, com diferenças de abordagens e performance (GYŐRÖDI et al., 2020).

Para o escopo deste projeto foi definido o uso do SQLite3 devido a sua simplicidade e efetividade, em conjunto com *Object Relational Mappers* (ORMs) que abstraem a manipulação dos dados para efetuar buscas, inserções e alterações nos bancos de dados, facilitando o manuseio dos dados adquiridos. Em conjunto, são utilizados o envio de dados comprimidos pelo formato HDF5 para grandes volumes, devido a necessidade da alta velocidade de transferência necessária para a otimização de entrada e saída de dados numéricos (FOLK et al., 2011). Sendo um formato considerado estruturado, descritivo e versátil (COLLETTE, 2013).

2.4 API

Uma *Application Programming Interface* (API) é um dos principais mecanismos que permite a comunicação entre componentes de software por meio de diferentes protocolos e é diretamente responsável por um dos fatores principais de transformação digital (TRČEK, 2022). Portanto requerem maior atenção em como os dados são estruturados para disponibilização, atentando-se a anti padrões linguísticos (PALMA et al., 2022).

Dentre as estruturas de construção de APIs existentes, a *Representational State Transfer* (RESTful) se destaca pela sua natureza sem estado, com requisições feitas por consumidores sendo similares às URLs digitadas para acessar websites comuns, contendo somente dados simples em suas respostas, facilitando o seu uso dinâmico de maneira eficiente, perfeitamente se encaixando dentro do contexto de dados abertos (RICHARDSON e RUBY, 2008).

A documentação clara e concisa de uma API RESTful é essencial para a sua integração nas plataformas que a requerem por seus desenvolvedores, para isso ferramentas como Swagger[1] e Redocs[2] tem como objetivo realizar tal documentação a partir da especificação aberta da OpenAPI, que automaticamente produz uma documentação gráfica a partir de um documento em JSON com as especificações de pelo menos um campo de rotas, componentes ou *webhooks*.

2.5 Aplicação Web

Aplicações web são softwares que são acessados em navegadores Web. São utilizados para conectar consumidores de maneira conveniente e segura, permitindo o acesso de funcionalidades complexas sem a necessidade de instalar ou configurar softwares adicionais (AWS, [s.d.]). A portabilidade e facilidade de uso garantido por esse modelo

[1] https://swagger.io/

[2] https://redocly.com/docs/redoc

de aplicação foram fatores essenciais de decisão na construção da arquitetura do software final.

Para a construção e otimização de aplicações web, existem três tipos de limites de responsividade considerados: até um decisegundo (0.1 segundos) para que o usuário não perceba atraso na resposta do sistema, até um segundo para manter o fluxo de raciocínio do usuário intacto, e dez segundos para manter o foco de atenção do usuário, com tempos maiores necessitando de feedback visual para indicar que o sistema não foi interrompido durante o processo (NIELSEN, 1993). Esses valores aproximados são utilizados como referência de tempo de resposta ideal para os componentes do software.

3 METODOLOGIA

O projeto teve sua versão de Python atualizada para a v3.12.6 devido a melhorias de tipagem e performance, continuando uma série de melhorias realizadas na versão anterior (v3.11.3). A organização do projeto foi alterada para utilizar o padrão de pacote com pyproject.toml, um arquivo declarativo que define as propriedades comuns do projeto, como nome, descrição, versão do projeto e dependências com suas versões.

Figura 1 - Exemplo de configuração do pacote pelo pyproject.toml.

```
[project]
name = "sinda"
description = "A Proposal for an Open Data Portal"
version = "2.0.0"
readme = "README.md"
requires-python = "≥3.12"
dynamic = [ "dependencies" ]

[tool.setuptools.dynamic]
dependencies = { file = [ "requirements.txt" ] }

[project.scripts]
newsinda = "sinda.__main__:main"
newsinda-backend = "sinda.backend.__main__:main"
newsinda-frontend = "sinda.frontend.__main__:main"
```

Fonte: Produção do autor.

Essa alteração foi feita para estratificar os métodos de execução do projeto, levando em consideração as possibilidades de execução do software: somente a API, somente a aplicação web, e ambos - de maneira conjunta. Mantendo assim a facilidade de execução mesmo com a arquitetura desacoplada, além de possibilitar a publicação do pacote em plataformas com o pypi (*Python Package Index*) e a instalação por meio de ferramentas como pip (*Package Installer for Python*) e pipx.

A execução simultânea do projeto utiliza processos paralelos a partir da biblioteca *multiprocessing,* nativa do python, que disponibiliza mecanismos de inicializar, gerenciar a execução e comunicação entre processos e sua finalização. Devido à característica assíncrona do projeto foi feito o uso de corrotinas através da biblioteca nativa *asyncio*, porém por conflitos de tarefas bloqueantes que resultaram em deadlocks quando realizada alguma comunicação entre o back-end e o front-end da aplicação causando o congelamento de ambas as partes, foi decidido o uso de processos paralelos.

Juntamente, devido a quantidade de opções de customização de comportamentos do software disponibilizadas por variáveis de ambientes, foi incluído como componente essencial o uso de arquivo *dotenv*, onde são definidas as variáveis de ambientes, por meio da biblioteca python-dotenv (v1.0.1). Permitindo que a execução seja simplificada nos diversos ambientes de desenvolvimento.

A extração de dados é caracterizada pelo modelo de extração, transformação e carregamento dos dados (ETL) (SEENIVASAN, 2023) e é realizada por web scraping, por meio da análise das páginas HTML adquiridas com requisições HTTP usando as bibliotecas requests (v2.31.0) e BeautifulSoup (v4.12.3) com lxml (v5.2.2). Para ambos os dados de PCDs e de boias, processo é separado em três principais etapas:

Primeiramente, as entidades disponíveis são extraídas da lista disponível em sua fonte e para cada entidade existente é realizado o acesso da página de consulta, que podem

conter metadados geográficos e de disponibilidade das entidades, quando estes dados existem e são considerados válidos a entidade é marcada como disponível. Para as entidades que são marcadas como disponíveis, é feito o download dos dados contidos na plataforma em períodos de um ano (limite máximo permitido pelo SINDA).

Os metadados e dados baixados podem conter sequências de caracteres inválidos, como quebra de linhas e vírgulas em números de ponto flutuantes, espaços repetidos, ou valores inválidos como 9.999. Portanto, antes do armazenamento do banco de dados esses são passados por etapas de limpezas que tratam ou filtram essas incongruências.

A extração de dados é realizada de maneira ética e responsável, visando não sobrecarregar o SINDA. Os downloads são realizados com caching e mecânicas de envelhecimento, onde os arquivos adquiridos quando válidos são mantidos por até três dias por padrão, além de possuir mecanismos de respostas à *rate limiting* – esperando cinco segundos antes de realizar outra requisição por padrão. Evitando assim a repetição de requisições com os mesmos parâmetros em curtos períodos de tempos.

O armazenamento dos dados adquiridos é realizado a partir de modelos declarativos das entidades que contém a tipagem dos dados relevantes, e são repassados para o ORM SQLAlchemy (v2.0.30) que mapeia as entidades com seus modelos e permite a realização das operações no banco de dados em SQLite3. Os dados cadastrados são caracterizados pelo agrupamento dos metadados das entidades, os tipos de colunas contidas, com unidades e integridade em porcentagem, e os dados reais contidos pela entidade. Anteriormente, o armazenamento dos dados reais eram feitos a partir de arquivos comprimidos em HDF5, porém buscando unificar o tratamento dos dados a abordagem foi alterada para incluir os dados em conjunto com os metadados no banco relacional.

Depois de armazenados, os dados podem ser acessados pela interface gráfica ou pela API, que foi estendida para conter documentação visual com o Swagger, aderindo à

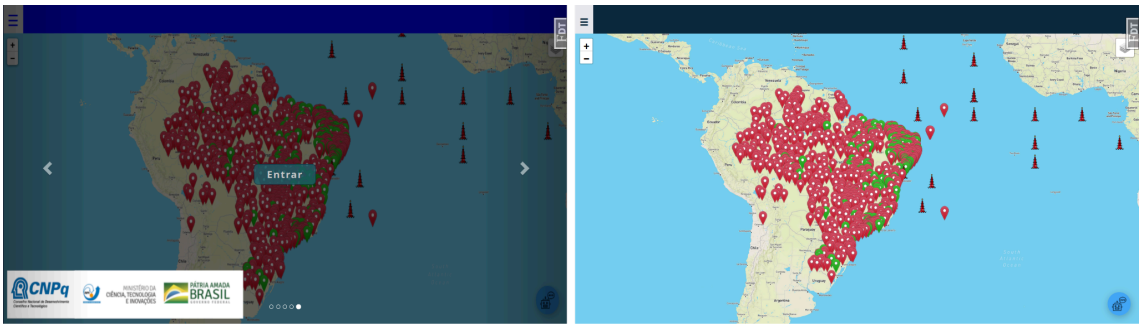
especificação definida pela OpenAPI. As rotas anteriormente existentes foram expandidas para conter parametrização de paginação ordenada, filtros e buscas, além de adicionar os diferentes códigos de resposta e os dados transitórios, que são explicitamente tipados por esquemas baseados nos modelos de dados utilizando o Pydantic (v2.7.1).

Para que os dados continuem sempre atualizados sem a interrupção da disponibilidade, o sistema contém uma funcionalidade de atualização de dados periódico a partir de rotinas definidas por entidade, onde a partir de um processo que executa em paralelo, utilizando as operações de extração de dados, o sistema verifica se há atualizações para as entidades relevantes.

Quando as atualizações são encontradas, o sistema verifica a diferença das datas e se existirem dados após a data final cadastrada ou dados anteriores a data inicial, esses são adquiridos e integrados aos dados pré-existentes, o que garante a integridade dos dados mesmo em casos de invalidação, remoção ou alteração dos dados contidos na fonte.

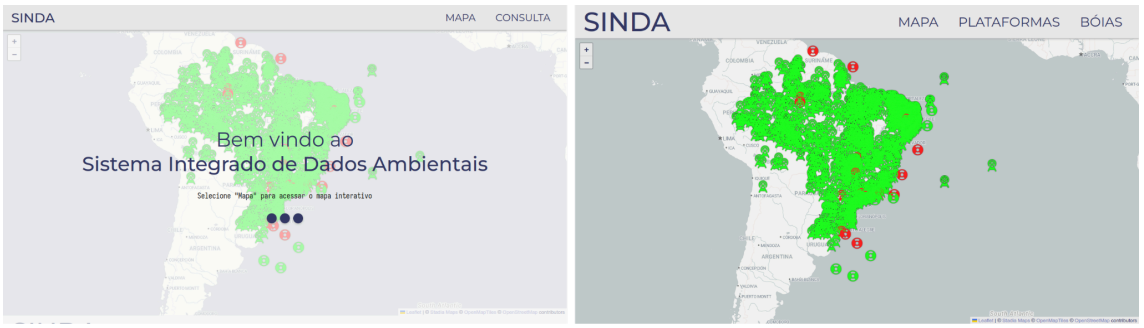
Além do segmento da API, nesse projeto foi realizada a reconstrução da interface gráfica, visando reestruturar a experiência de usuário utilizando as métricas de tempo de resposta e buscando manter uma estabilidade de padrão visual e de interação, ainda com templates HTML pelo Jinja (v3.1.4). A página de índice foi simplificada, mantendo a introdução à plataforma mas incluindo uma barra de navegação com links diretos para o mapa e consulta de dados.

Figura 2 - Versão anterior da interface de índice e mapa.



Fonte: Adaptada da versão do software de SILVA; ALMEIDA (2022)

Figura 3 - Versão atual da interface de índice e mapa.

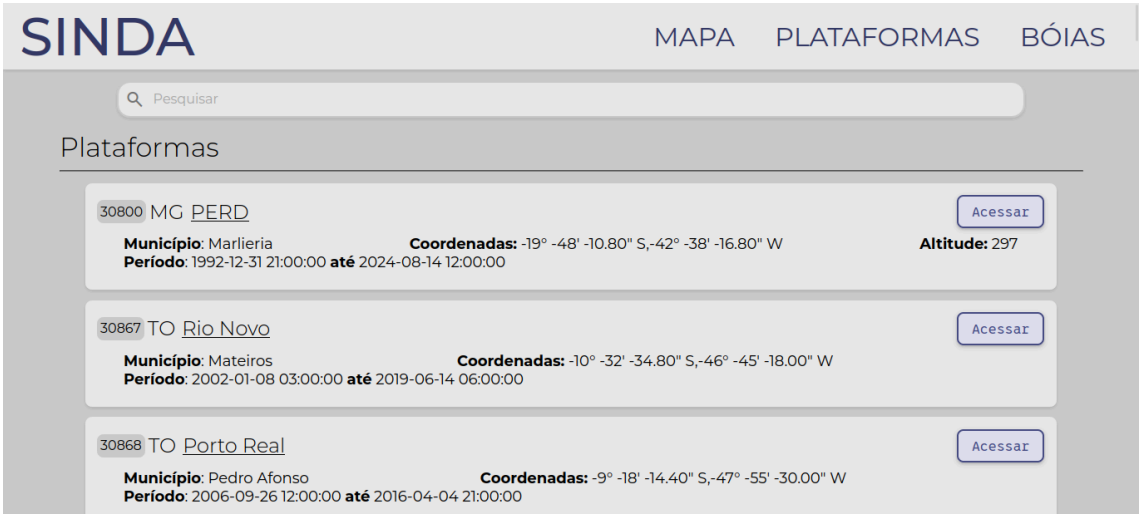


Fonte: Produção do autor.

A renderização do mapa interativo é realizada com Leaflet (v.1.9.4), utilizando camadas de imagem do OpenStreetMap, que incluem as bordas territoriais de países e estados, juntamente com as principais rodovias. As entidades agora dispõem de dicas de ferramentas que são visualizadas ao passar o cursor sobre a mesma.

Foi adicionado uma página dedicada à consulta de dados de maneira textual, similar ao que é presente no sistema atual, porém contendo de maneira imediata as informações geográficas, que atualmente requerem navegação adicional para acesso.

Figura 4 - Versão atual da interface de consulta.



Fonte: Produção do autor.

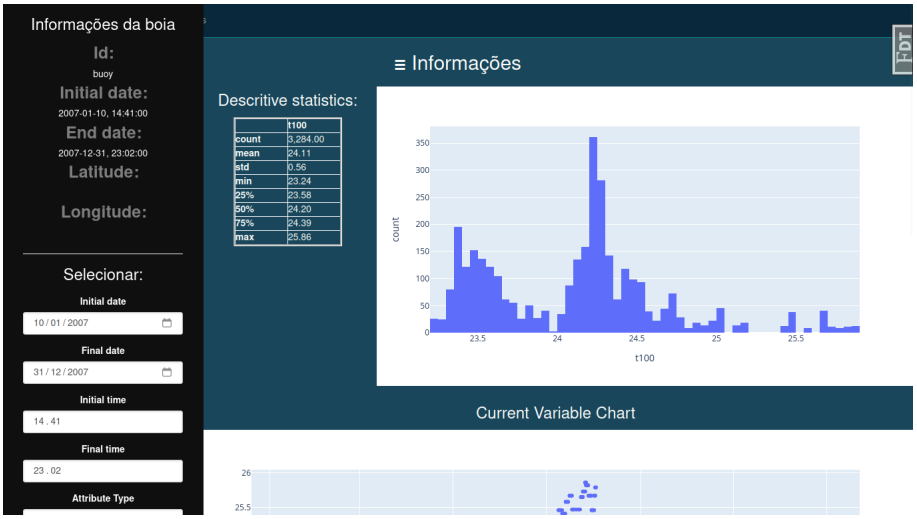
As consultas emas ambas as páginas de mapa interativo são realizadas a partir de requisições HTTP para a API, de maneira paginada, assíncrona e deferida em Javascript, essa metodologia permite que a página seja devolvida ao usuário imediatamente, enquanto os dados são buscados incrementalmente. Isso permite que a página de carregamento que existia anteriormente fosse substituída por um ícone de carregamento visível enquanto a aquisição de dados prossegue.

Além disso, para ambas as páginas existe a possibilidade de filtrar a exibição a partir de algum dos componentes das entidades, como nome, localização, data de disponibilidade. A estilização da interface dos filtros foi estabilizada para manter uma uniformidade em todas as páginas.

A uniformização e harmonização da interface foi realizada por arquivos em CSS que são servidos junto com as páginas HTML, dentro de um parâmetro no cabeçalho, que indicam o comportamento dos diferentes componentes e suas relações dentro do contexto gráfico do navegador. Sobrescrevendo a estilização de componentes das bibliotecas utilizadas para garantir a consistência da interface e melhorando assim a experiência do usuário final (LI et al., 2022).

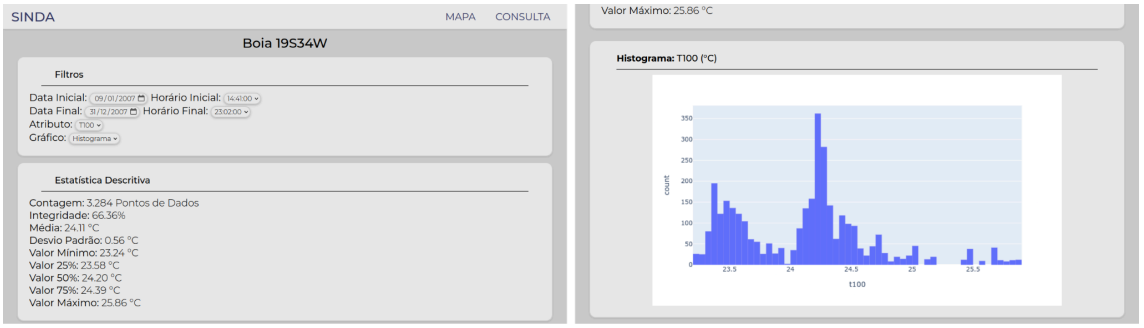
Por fim, o dashboard foi atualizado para corrigir incongruências de estilização, usando estatísticas descritivas, com seleção de período alvo e gráficos interativos de barras e histogramas por meio do plotly (v5.24.1) para representar os dados. Durante o projeto foi considerado o uso do "matplotlib", porém devido a fatores de interatividade dos gráficos que o "matplotlib" não apresentava, como os controles de posição e eventos de sobreposição, foi decidido manter o plotly.

Figura 5 - Versão anterior da interface de dashboard.



Fonte: Adaptada da versão do software de SILVA; ALMEIDA (2022)

Figura 6 - Versão atual da interface de dashboard.



Fonte: Produção do autor.

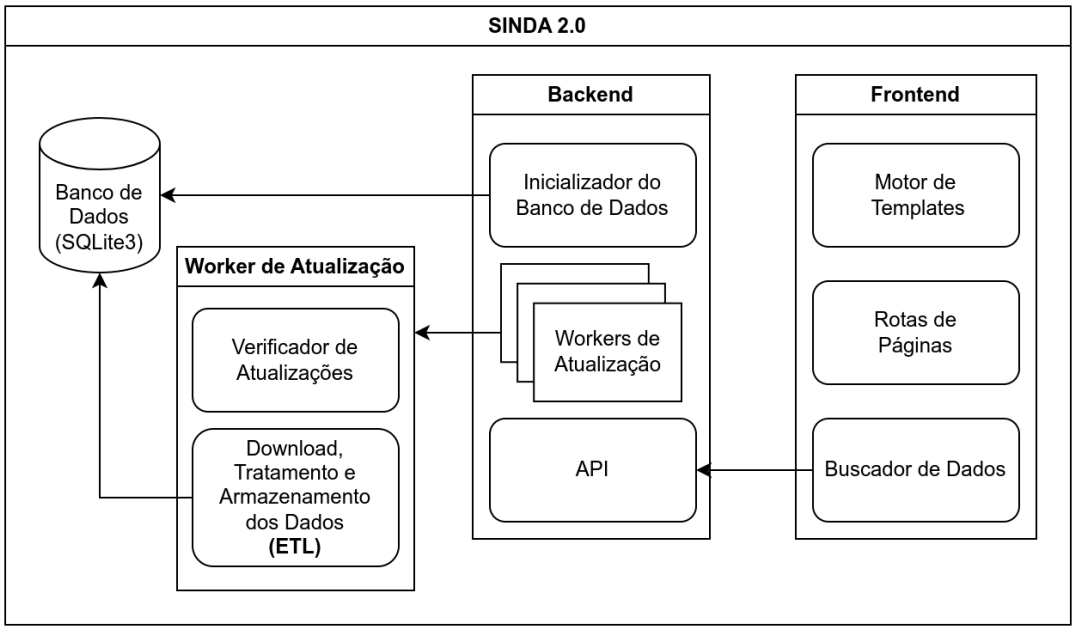
4 RESULTADOS E DISCUSSÕES

Com a realização da extensão da arquitetura do projeto (Figura 7 e 8), viabilizando a inicialização parcial ou simultânea dos componentes do software, em conjunto com a adição do mecanismo de atualização dos dados. Essa nova abordagem busca levar em consideração fatores como frequentes indisponibilidades do SINDA que assim como no processo anterior, continuaram a ocorrer durante o desenvolvimento do projeto, o tratamento de dados inválidos, agrupamento em um único banco de dados, para manter a constante disponibilidade para os usuários finais.

Utilizando das melhorias de performance realizadas nas atualizações mais recentes do Python e das bibliotecas incluídas, os requerimentos do projeto foram revisados para manter essa paridade. Em particular, a aplicação das bibliotecas pandas (v2.2.2) numpy (v1.26.4) para o processamento de dados foram alteradas para reduzir a quantidade de operações repetidas, por meio de aplicação de *caching* com a técnica *least recently used* (LRU) que consiste da adição e movimentação de nós em uma lista pela frequência de acesso – removendo os menos utilizados, e a redução do volume de data frames (a estrutura de dados que contém os dados em linhas e colunas na memória).

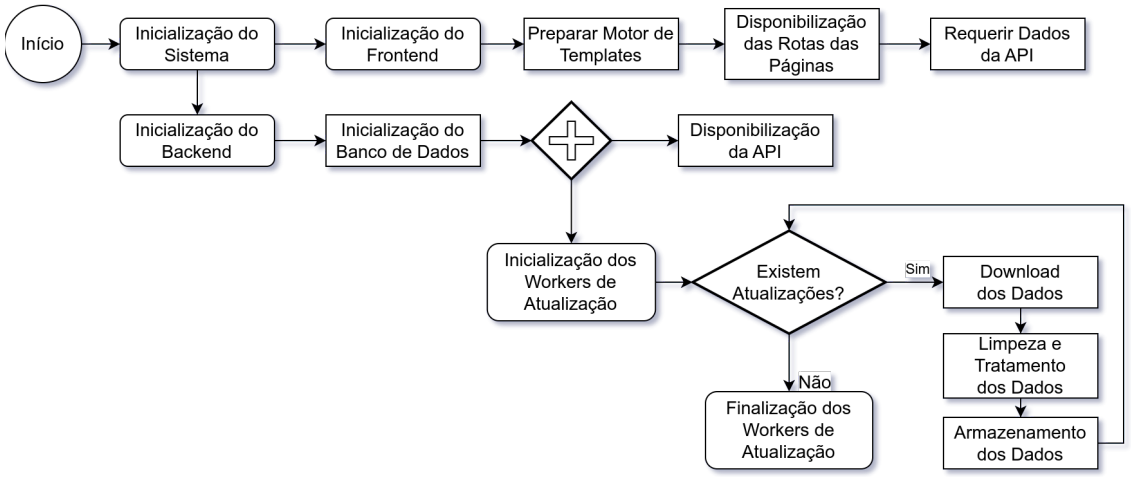
Essas melhorias resultam em um menor tempo de resposta e maior eficácia no processamento de dados e entrega de informação, que permite maior operação de dados em volume e velocidade, com menos erros e uma melhor experiência de usuário, além de permitir a adição de funcionalidades .

Figura 7 - Arquitetura revisada do projeto.



Fonte: Produção do autor.

Figura 8 - Fluxograma de execução do projeto.



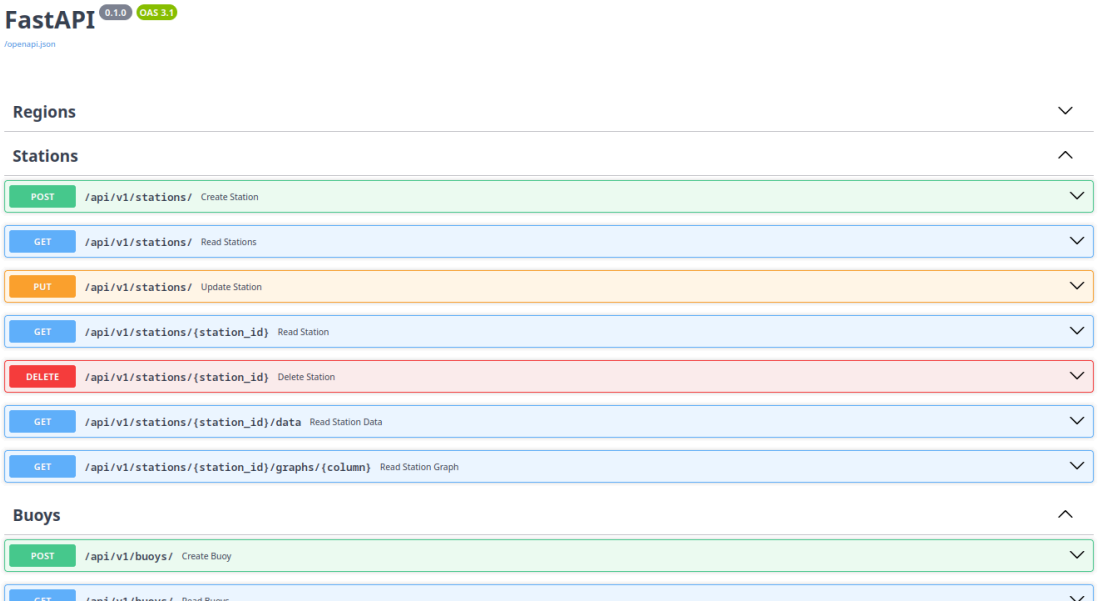
Fonte: Produção do autor.

A extensão da API entra como um fator principal desse projeto, com documentação das rotas, esquemas e códigos de respostas documentados aderindo mais adequadamente ao

conceito de portais de dados abertos, facilitando o uso e democratizando o acesso à API em caso de disponibilidade pública. Além de disponibilizar os dados de uma maneira mais eficiente, com paginação de dados e filtros aplicados em buscas, reduzindo o gasto de performance na hora de realizar operações nos dados existentes.

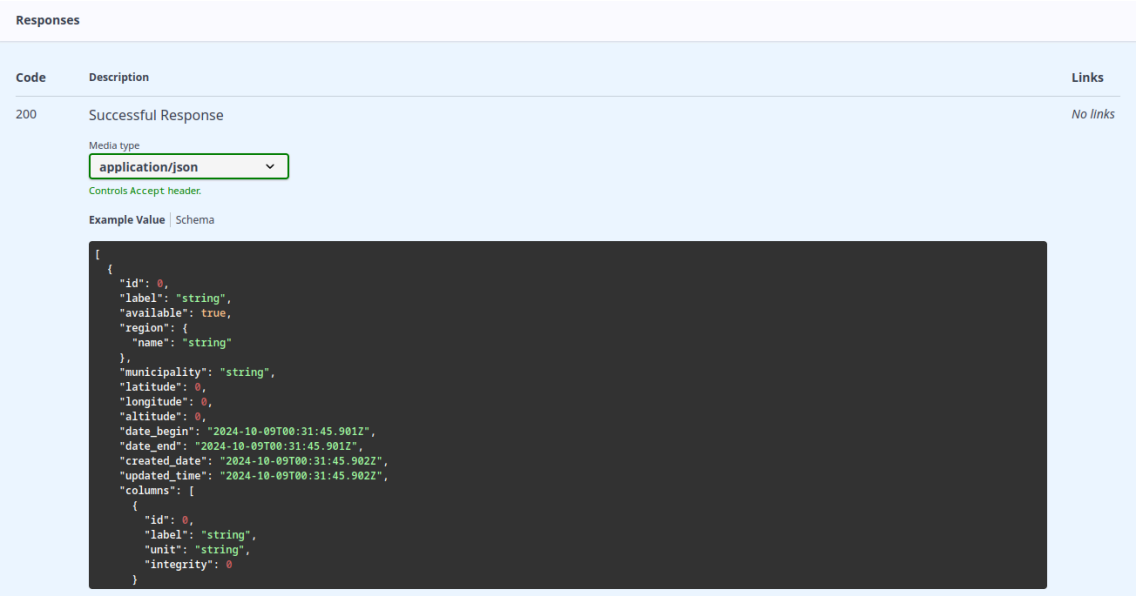
A disponibilização das rotas pelo ASGI em ambos os componentes de back-end e front-end foram realizadas com o FastAPI (v0.111.0) que possui dois mecanismos de documentação sejam disponibilizados de maneira automática a partir da definição das rotas, o Swagger e o Redoc, porém devido a consistência da interface visual do Swagger ser melhor definida, o Redoc foi desabilitado.

Figura 8 - Rotas pelo *Swagger*.



Fonte: Produção do autor.

Figura 9 - Respostas pelo Swagger.



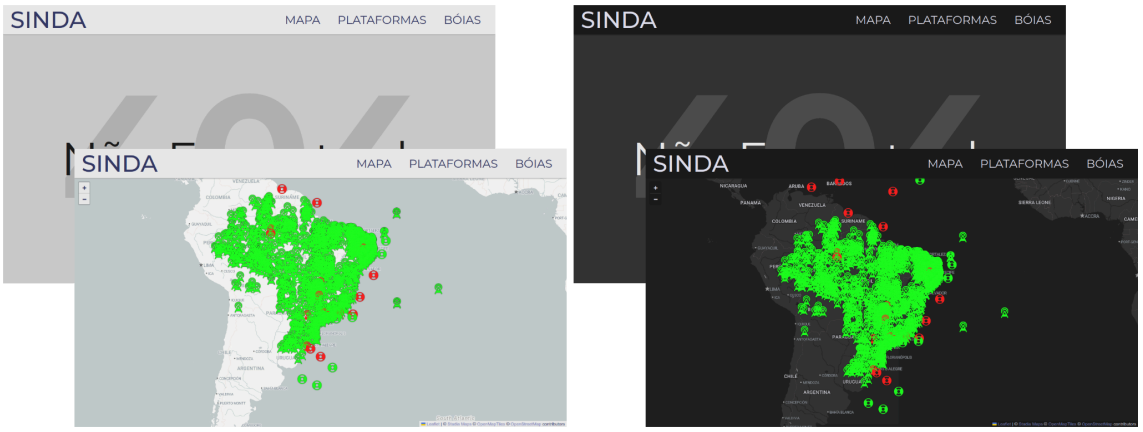
Fonte: Produção do autor.

Figura 10 - Esquemas pelo Swagger.

Fonte: Produção do autor.

A Reconstrução da Interface Gráfica conta com técnicas de streaming de dados, mantendo a consistência de sua estilização, com temas claro e escuro que são definidos automaticamente pela preferência configurada no navegador do usuário, com alterações focando na intuitividade de interação com o usuário, dispondo de análises estatísticas sobre a integridade dos dados contidos e mecanismos de interatividade por meio dos mapas e dos gráficos no *dashboard*.

Figura 11 - Variação de temas de interface clara e escura.



Fonte: Produção do autor.

5 CONCLUSÃO

Esse estudo teve como objetivo a continuação da reestruturação da arquitetura do projeto e o seu desacoplamento, corrigindo potenciais problemas de indisponibilidade e tratamento de dados, além de efetuar melhorias de performance e usabilidade. Concluindo assim o objetivo principal estabelecido no início do projeto, com a construção de uma portal de dados abertos, trabalhando com os dados de PCDs e boias do projeto PIRATA.

Como trabalhos futuros, considera-se o estudo e comparação de estratégias de armazenamento em banco de dados, por exemplo, a utilização de bancos de dados não relacionais, devido às características intrínsecas os dados georreferenciados, podendo haver melhorias de performance e usabilidade, já que o modelo atual do banco requer que os dados sejam recortados, removendo do modelo principal as colunas que não são

utilizadas. Outra consideração seria a utilização de um sistema de cache como o Redis, podendo tirar vantagem da repetição das requisições para uma maior eficiência na entrega dos dados.

A aplicação de *sharding*, uma técnica de divisão dos bancos de dados em fragmentos menores e a extensão dos dados trabalhados como os dados de queimadas e desmatamento contidos no portal do terrabrasilis do INPE[3], devido a sua natureza georreferenciada, aumentado o escopo e transformando a plataforma em uma plataforma de dados variados de maneira unificada. Além da consideração da conteinerização da aplicação e a inclusão de serviços em nuvem como a AWS e suas funcionalidades.

6 REFERÊNCIAS

AWS. **What is a Web Application? - Web Applications Explained - AWS**. Disponível em: <https://aws.amazon.com/what-is/web-application/>.

COLETTE, A. Python and HDF5: **Unlocking Scientific Data**. Beijing: “O’Reilly Media, Inc”, 2013.

FOLK, M. et al. A**n overview of the HDF5 technology suite and its applications**. Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases - AD ‘11, 25 mar. 2011.

FORD, J. D. et al. Opinion: Big data has big potential for applications to climate change adaptation. **Proceedings of the National Academy of Sciences**, v. 113, n. 39, p. 10729–10732, 27 set. 2016.

[3] https://terrabrasilis.dpi.inpe.br/queimadas/portal/

GIBERT, K. et al. Environmental Data Science. **Environmental Modelling & Software**, v. 106, p. 4–12, 1 ago. 2018.

GYŐRÖDI, C. A. et al. Performance Analysis of NoSQL and Relational Databases with CouchDB and MySQL for Application's Data Storage. **Applied Sciences**, v. 10, n. 23, p. 8524, 28 nov. 2020.

JANSSEN, M. et al. Transparency-by-design as a foundation for open government. **Transforming Government: People, Process and Policy**, v. 11, n. 1, p. 2–8, 20 mar. 2017.

LI, W. et al. Design Factors to Improve the Consistency and Sustainable User Experience of Responsive Interface Design. **Sustainability**, v. 14, n. 15, p. 9131, 25 jul. 2022.

LNENICKA, M.; NIKIFOROVA, A. Transparency-by-design: What is the role of open data portals? **Telematics and Informatics**, v. 61, p. 101605, ago. 2021.

MAGINA, F. DE C. **Parâmetros coletados, calculados e transmitidos via satélite SCD pelas Plataformas de Coleta de Dados**. Disponível em: <https://cimehgo.meioambiente.go.gov.br/rede_obs/dados.html>.

NIELSEN, J. **Response Time Limits: Article by Jakob Nielsen**. Disponível em: <https://www.nngroup.com/articles/response-times-3-important-limits/>.

OpenAPI Specification v3.1.0. Disponível em: <https://spec.openapis.org/oas/v3.1.0.html>.

PALMA, F. et al. Assessing the linguistic quality of REST APIs for IoT applications. **Journal of Systems and Software**, v. 191, p. 111369, set. 2022.

RICHARDSON, L.; RUBY, S. **RESTful Web Services: Web Services for the Real World**. [s.l.] Beijing: “O’Reilly Media, Inc”, 2007.

SEBESTYÉN, V.; CZVETKÓ, T.; ABONYI, J. The Applicability of Big Data in Climate Change Research: The Importance of System of Systems Thinking. **Frontiers in Environmental Science**, v. 9, 17 mar. 2021.

SEENIVASAN, D. ETL (Extract, Transform, Load) Best Practices. **International Journal of Computer Trends and Technology**, v. 71, n. 1, p. 40–44, 31 jan. 2023.

SILVA, R.B.; ALMEIDA, E.S. **Um framework para a identificação e diferenciação de plataformas de coleta de dados ambientais do Sistema Nacional de Dados Ambientais (SINDA)**. In Anais Estendidos do XVIII Simpósio Brasileiro de Sistemas de Informação (pp. 17-20). SBC, 2022.

TRČEK, D. E-Government 4.0: Managing APIs as Facilitators for Digital Transformation. **Academic Journal of Interdisciplinary Studies**, v. 11, n. 1, p. 1, 3 jan. 2022.