GaiaSenses: o Frontend e Backend de uma aplicação móvel para a geração automática de composições audiovisuais baseadas em dados climáticos

Álvaro Augusto Costa[1], **Artemis Moroni**[2]

a246909@dac.unicamp.br, artemis.moroni@cti.gov.br

[1]Faculdade de Tecnologia – FT
Universidade Estadual de Campinas (UNICAMP) – Limeira/SP

[2]Divisão de Sistemas Ciberfísicos – DISCF
CTI/MCTI Renato Archer – Campinas/SP

Abstract. *Raising public awareness regarding the need for action in face of climate change still persists as a significant challenge. Given such an issue, we present the GaiaSenses application, which has the goal of combining art and technology in a creative way, generating dynamic audiovisual compositions from climate data about the region the user is located in. In this article we describe the frontend and backend architecture, the challenges thereof and the solutions for such challenges. The main results of this work are the initial prototype of the application, the creation of a service for climate data processing as well as the integration of previous works done in the GaiaSenses project.*

Resumo. *A conscientização do público em relação à necessidade de ação diante das mudanças climáticas ainda permanece como um desafio significativo. Diante dessa questão, apresentamos o aplicativo GaiaSenses, que tem como propósito combinar arte e tecnologia de maneira criativa, gerando composições audiovisuais dinâmicas a partir de dados climáticos relativos à região em que o usuário está localizado. Neste artigo descrevemos a arquitetura do frontend e backend, as dificuldades encontradas e soluções para tais desafios. Os principais resultados deste trabalho são o protótipo inicial da aplicação, a criação de um serviço para processamento de dados climáticos e a integração de trabalhos anteriores realizados no projeto GaiaSenses.*

1. Introdução

Diante da realidade das mudanças climáticas, um dos grandes desafios que permanece é o de conscientizar o público não-especialista sobre a urgência do tema, criando engajamento que resulte em ações sustentáveis [1]. A arte aliada à tecnologia se mostra como uma promissora forma de abordar este desafio [1, 2, 3].

Neste contexto, o projeto GaiaSenses propõe a criação de um aplicativo para dispositivos móveis capaz de gerar composições audiovisuais a partir de dados climáticos relativos à região geográfica do usuário, que podem ser compartilhadas com demais pessoas dentro da plataforma. As composições convidam à reflexão sobre as condições climáticas locais, eventualmente levando o público à tomada de ações sustentáveis.

Descreveremos neste artigo a implementação da aplicação GaiaSenses, iniciando com uma visão geral de sua construção, dividida em duas grandes partes: o *frontend*, que diz respeito ao aplicativo utilizado pelos usuários, e o *backend*, responsável pelo armazenamento e processamento dos dados. Em seguida passamos a detalhar cada uma destas partes, apresentando o principal desafio encontrado: integrar a geração de animações com a geração de áudio e processamento de dados de fontes remotas.

2. Visão geral da aplicação

O GaiaSenses permite que usuários se cadastrem na plataforma e gerem composições audiovisuais de acordo com dados a respeito das condições climáticas em sua localização. As composições podem posteriormente ser tornadas públicas, permitindo que outros usuários as visualizem.

Devido à sua natureza distribuída e multiusuário, a aplicação é estruturada em uma arquitetura cliente-servidor (Figura 1). Os dispositivos móveis são clientes que se comunicam com um servidor que provê o serviço de aplicação, responsável pela autenticação de usuários e criação e edição de posts, e serviço de processamento de dados climáticos, responsável por acessar as fontes de dados meteorológicos e processar os dados recuperados, extraindo informações necessárias para a geração das composições.

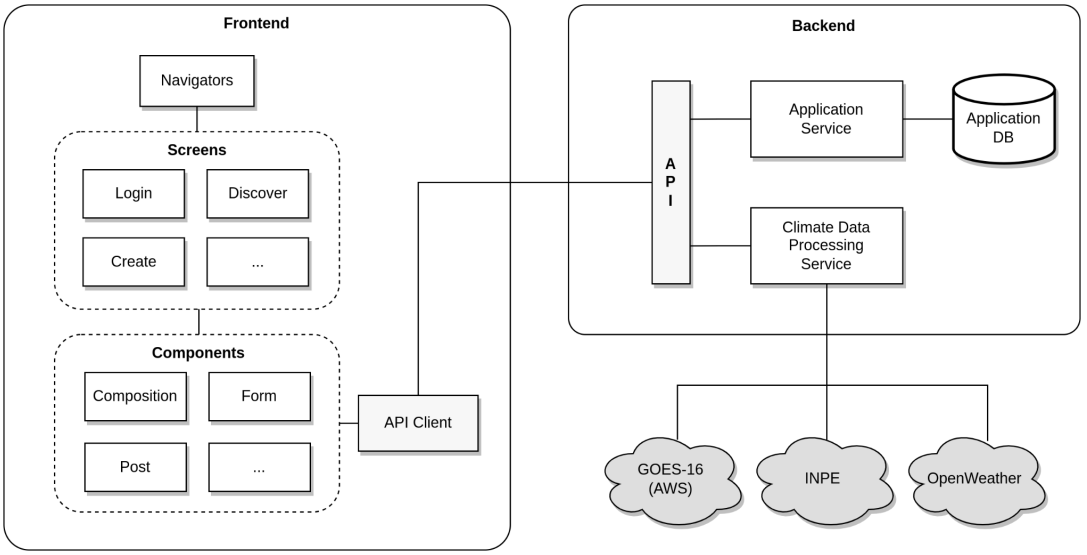

Figura 1: Visão geral da arquitetura da aplicação, destacando os elementos que compõem o *frontend* e *backend*

O *frontend* é composto por elementos chamados *navigators* que são responsáveis pela navegação do usuário entre as telas do aplicativo. As telas, por sua vez, são compostas por componentes reutilizáveis, que representam elementos gráficos na interface de usuário. Para acessar os dados necessários para seu funcionamento, os componentes utilizam uma camada cliente que se comunica com a API do *backend*.

3. O *frontend*

Para reutilização de código na interface gráfica do aplicativo, optou-se por desenvolvê-la utilizando o *framework* React Native [4], que permite que a aparência e comportamento da interface da aplicação sejam desenvolvidos em uma única linguagem e executem em diferentes plataformas, utilizando os componentes nativos de cada uma delas. A linguagem de desenvolvimento é JavaScript.

Inicialmente, a implementação do aplicativo se concentrou apenas na plataforma *Android*, dado sua popularidade e facilidade de acesso para desenvolvimento.

3.1. Interface de usuário

No total, o aplicativo é composto por cinco telas. As primeiras telas em que o usuário tem contato são "*Login*" e "*Register*" (Figura 2), em que há a autenticação e registro na plataforma, respectivamente.

Em seguida, é apresentado a tela de "*Discover*" (Figura 3), na qual as composições publicadas por outros usuários são visualizadas. Esta tela também dá acesso às telas "*Create*", onde se pode gerar uma nova composição, e "*Gallery*", onde se visualizam todas as composições geradas pelo usuário atualmente autenticado.

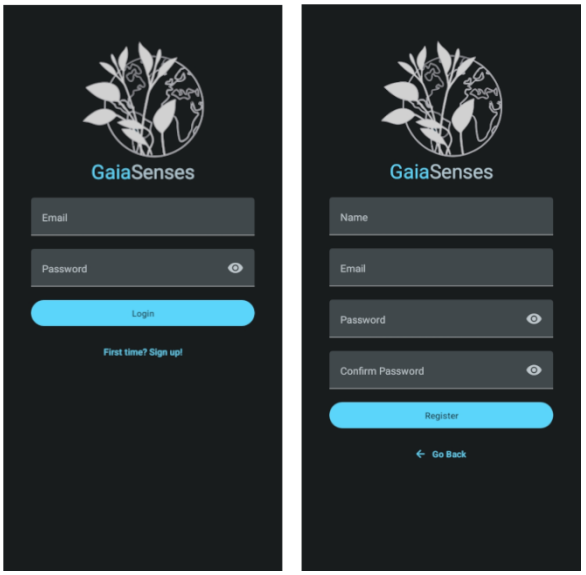

Figura 2: Telas de *Login* (esquerda) e *Register* (direita)

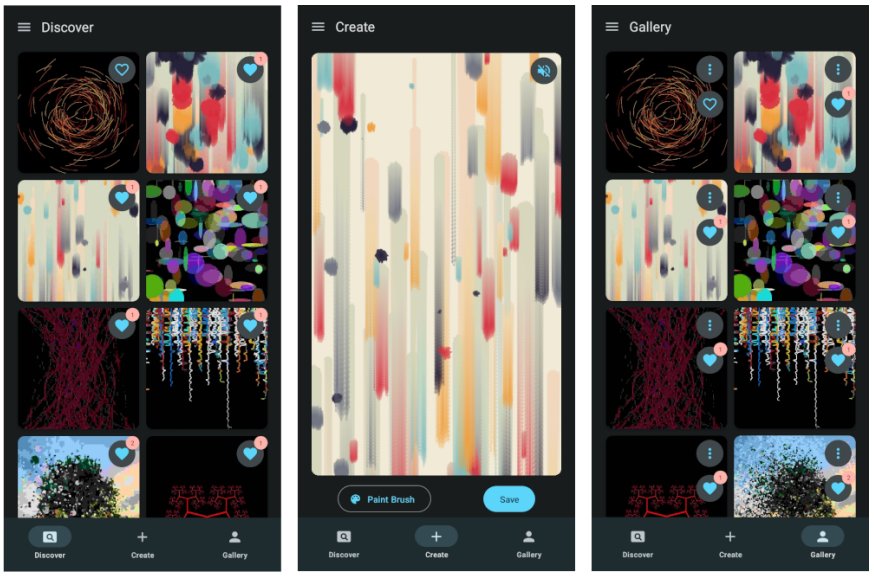

Figura 3: Telas *Discover* (esquerda), *Create* (centro), *Gallery* (direita)

3.2. Composições audiovisuais

As composições audiovisuais são compostas por dois elementos: a geração da animação gráfica e a geração do efeito sonoro que acompanha a animação. O elemento visual é criado com o uso do *p5.js*, uma biblioteca em JavaScript para código criativo que utiliza os recursos disponíveis em uma página web para criar desenhos [5]. O elemento sonoro é criado utilizando-se da linguagem de programação visual *Pure Data*. Devido ao caráter multiplataforma do aplicativo, a escolha dessas ferramentas possibilita a definição das composições de maneira independente do ambiente em que executam.

Os *scripts* em *p5.js* necessitam ser executados em um ambiente web, uma vez que esta biblioteca utiliza as APIs disponíveis para a web. Em um aplicativo móvel é possível incorporar a *WebView* do sistema e executar os *scripts* dentro deste ambiente. Em React Native, a *WebView* é acessada utilizando um *Native Component* (Figura 4): um componente de interface definido em código nativo e acessado em JavaScript. Devido à abordagem adotada, passam a existir dois ambientes JavaScript: um para o React e outro dentro da *WebView*. Isto torna o compartilhamento dos dados climáticos para a geração das composições uma tarefa não trivial. A solução encontrada é "injetar" os dados no ambiente do *p5.js* através de um *script* definido em tempo de execução que cria uma variável global, acessível pelos códigos de geração de composições.

Por sua vez, os programas em *Pure Data* necessitam da biblioteca nativa *libpd* que implementa a linguagem para a plataforma utilizada. Neste aspecto, o desafio encontrado é o de integrar esta biblioteca com o React Native, a fim de controlar o carregamento e a execução dos programas através do JavaScript. A solução proposta é a

implementação de um *Native Module* que expõe ao ambiente React funções da biblioteca *libpd* (Figura 4).

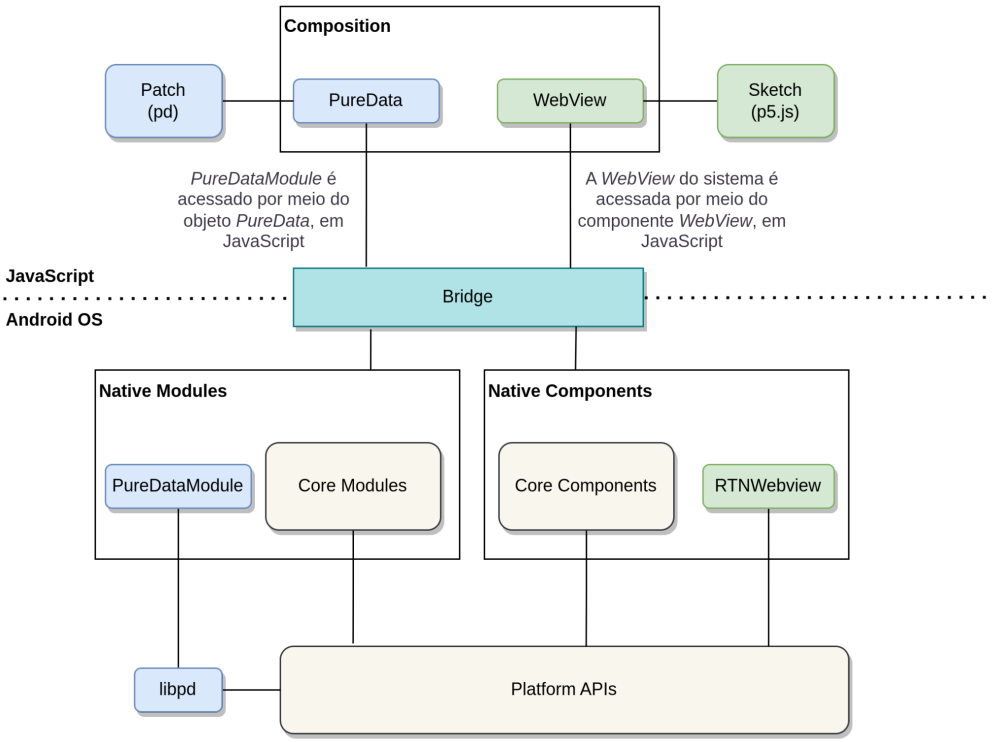

Figura 4: Arquitetura do React Native integrada com *WebView* e *libpd*

4. O *backend*

No *backend* dois serviços são necessários, um para lidar com autenticação de usuários, criação e manipulação de composições salvas, e outro para processar e fornecer dados climáticos. O primeiro, chamado de Serviço de Aplicação, foi desenvolvido em JavaScript utilizando o *framework* web *Express*. O segundo, chamado de Serviço de Processamento de Dados, foi escrito em Python, com o *framework* web *Flask* e fazendo uso do rico ecossistema dessa linguagem para processamento de dados.

4.1. Serviço de Aplicação

A arquitetura deste serviço está dividida em quatro componentes: *Routers*, *Controllers*, *Services* e *Repositories* (Figura 5). Os *Routers* definem as rotas que a API expõe para lidar com os três tipos de recursos que este serviço manipula: usuários, *posts* e *likes*. Os *Controllers*, extraem da requisição HTTP recebida os parâmetros esperados, realizam uma ação provida pelos *Services*, e retornam uma resposta ao cliente. Consequentemente, os *Services* definem ações como publicar um *post* e registrar um novo *like*. Estas ações necessitam acessar o banco de dados da aplicação, e este acesso é abstraído pelos *Repositories*, evitando que os *Services* manipulem diretamente o banco.

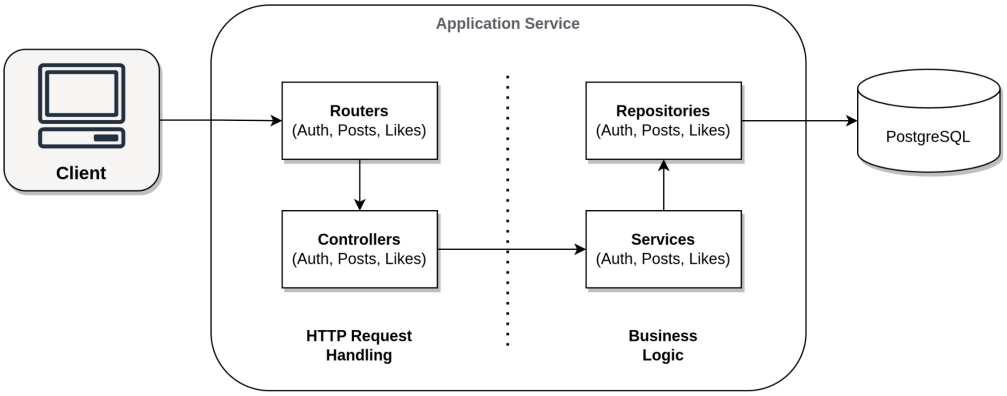

Figura 5: Arquitetura do serviço de aplicação

4.2. Serviço de Processamento de Dados

O processamento de dados climáticos é um dos elementos vitais para o projeto GaiaSenses. Portanto, o grande desafio na elaboração deste serviço é conceber uma arquitetura que facilite sua extensão, seja adicionando novas rotinas de processamento ou novas fontes de dados.

A proposta deste serviço é a estruturação em três componentes: uma API, *Processors* e *Data Sources* (Figura 6). A API fornece a interface para que clientes obtenham os dados processados, de acordo com o produto de interesse. Inicialmente, os produtos disponíveis são dados a respeito de precipitação, focos de incêndio e registro de raios.

Para cada tipo de produto existe um *Processor* e *Data Source* equivalentes. Os *Data Sources* representam a camada que faz o acesso às bases de dados remotas. Estas bases são a *OpenWeather*, que fornece dados sobre a condição do tempo atual na localização desejada, o Programa Queimadas do INPE (Instituto Nacional de Pesquisas Espaciais), que fornece dados sobre focos de incêndio no Brasil, e dados do satélite GOES-16, que reúne dados e imagens sobre o clima no hemisfério ocidental [6].

Os *Processors* recebem os dados brutos fornecidos pelas fontes e realizam o processamento para extrair métricas e informações relevantes para a geração das composições audiovisuais no *frontend*. As rotinas de processamento foram desenvolvidas em trabalho anterior [6] e baseadas nos modelos encontrados na bibliografia [7, 8].

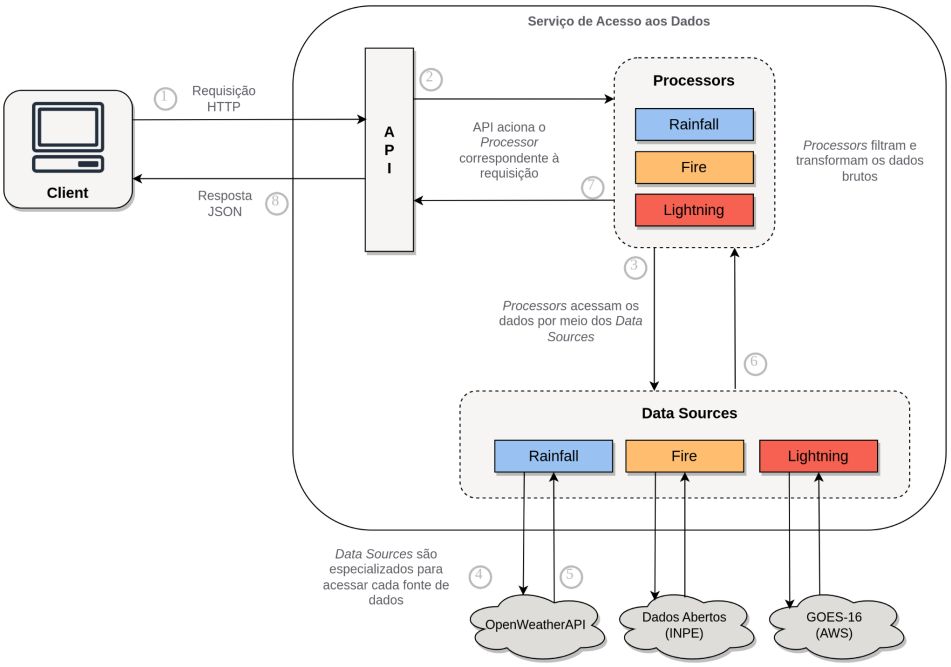

Figura 6: Arquitetura do serviço de processamento de dados

Graças a essa organização, o processo de adicionar novos produtos e novas fontes de dados torna-se viável, e a complexidade fica contida apenas no algoritmo de processamento dos dados e não em como integrá-lo ao sistema.

5. Resultados

As contribuições principais deste trabalho para o projeto GaiaSenses são o protótipo inicial do aplicativo móvel e implementação e aprimoramento do *backend*, sobretudo no que diz respeito ao serviço de processamento de dados. Este trabalho, portanto, contribui para a integração dos desenvolvimentos realizados anteriormente no projeto [6, 9]. A Figura 7 demonstra a integração viabilizada por este trabalho: à esquerda a resposta recebida pelo serviço de processamento de dados com relação ao produto de precipitação e à direita uma composição que faz uso destes dados.

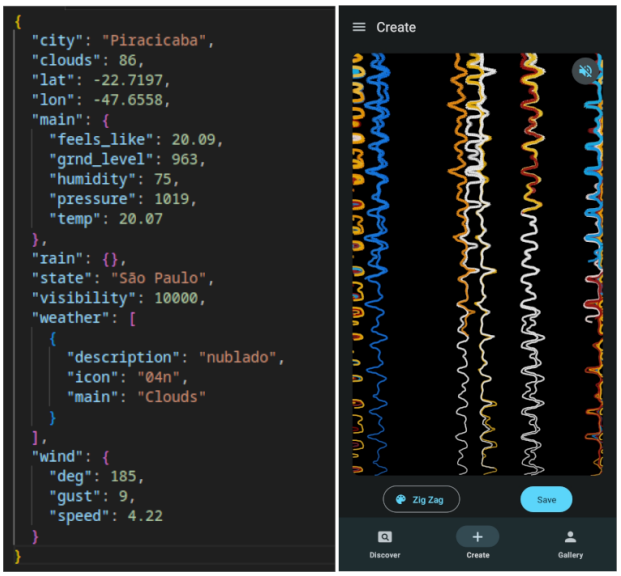

Figura 7: Composição visual (direita) gerada a partir de dados de precipitação (esquerda)

Outro ponto a se destacar é que, no contexto do aplicativo móvel, foi desenvolvida a funcionalidade de executar programas em *Pure Data* no Android com integração com o *framework React Native*. Esta contribuição pode ser útil para trabalhos em outros contextos que necessitem utilizar estas tecnologias em conjunto.

Todo código relativo à implementação do GaiaSenses se encontra disponível de maneira aberta nos repositórios pertencentes à organização GaiaSenses no GitHub [10].

6. Próximos Passos

Com relação ao *frontend*, é importante que trabalhos futuros abordem a identificação de problemas de usabilidade na interface do aplicativo e implementem melhorias.

No *backend*, uma oportunidade de estudo está em melhorar a maneira como as composições são salvas. Atualmente apenas uma imagem capturando um instante da animação é salva, o que não é ideal. Deve-se investigar formas de salvar as composições, incluindo vídeo e som, e visualizá-las posteriormente, mantendo o caráter audiovisual das mesmas.

Outra necessidade é a inclusão de novas rotinas de processamento de dados no *backend*. Para tal se torna importante verificar possíveis pontos de otimização no sistema de processamento de dados, a fim de que seja possível lidar com volumes de dados cada vez maiores.

7. Conclusão

A arte pode ser uma grande aliada à tecnologia para conscientizar as pessoas sobre a urgência das mudanças climáticas e da necessidade de tomada de ações sustentáveis. Enquanto a arte convida o público à reflexão quanto ao tema, a tecnologia possibilita um

maior alcance e novas possibilidades para a criatividade [3]. O objetivo do projeto GaiaSenses é explorar esta interseção.

Como parte fundamental desta exploração está a implementação inicial do aplicativo e do serviço de processamento de dados. Esta infraestrutura básica integra desenvolvimentos prévios do projeto e viabiliza a sua continuidade para investigar maneiras criativas de representar as condições e mudanças climáticas através de composições audiovisuais.

Referências

[1] Megarry, W. e Hadick, K. (2021) “Lessons from the Edge: Assessing the impact and efficacy of digital technologies to stress urgency about climate change and cultural heritage globally”, The Historic Environment: Policy & Practice, 12:3-4, 336-355, DOI: 10.1080/17567505.2021.1944571

[2] Aragón, C., Jasim, M. e Mahyar, N. (2021) “RisingEMOTIONS: Bridging Art and Technology to Visualize Public’s Emotions about Climate Change”. In Creativity and Cognition (C&C ’21), June 22–23, 2021, Virtual Event, Italy. ACM, New York, NY, USA, 10 pages. DOI: 10.1145/3450741.3465259

[3] Randerson, J. (2018) “Weather as a Medium: Toward a Meteorological Art”. Cambridge, MA: The MIT Press, 2018.

[4] React Native documentation (2023). “Core Components and Native Components”, https://reactnative.dev/docs/getting-started, 2023.

[5] p5.js documentation (2023). “Hello!”, https://p5js.org/, 2023.

[6] Rigue, I. e Moroni, A. (2022) “GaiaSenses: Acesso à base de dados de satélite e tratamento de seus produtos”. Anais da XXIV Jornada de Iniciação Científica do CTI Renato Archer - JICC 2022. Campinas: CTI Renato Archer, 2022.

[7] Noel Gorelick, Matt Hancher, Mike Dixon, Simon Ilyushchenko, David Thau, Rebecca Moore, Google Earth Engine: Planetary-scale geospatial analysis for everyone, Remote Sensing of Environment, Volume 202, 2017, Pages 18-27, ISSN 0034-4257.

[8] V. Génot, L. Beigbeder, D. Popescu, N. Dufourg, M. Gangloff, M. Bouchemit, S. Caussarieu, J.-P. Toniutti, J. Durand, R. Modolo, N. André, B. Cecconi, C. Jacquey, F. Pitout, A. Rouillard, R. Pinto, S. Erard, N. Jourdane, L. Leclercq, S. Hess, M. Khodachenko, T. Al-Ubaidi, M. Scherf, E. Budnik, Science data visualization in planetary and heliospheric contexts with 3DView, Planetary and Space Science, Volume 150, 2018, Pages 111-130.

[9] Cardoso, T. e Moroni, A. (2021) “GaiaSenses: desenvolvimento de protótipo para aplicativo móvel para geração de composições audiovisuais a partir de dados de plataformas planetárias. Anais da XXIII Jornada de Iniciação Científica do CTI Renato Archer – JICC 2021. Campinas: CTI Renato Archer, 2021.

[10] Organização GaiaSenses (2023). “GaiaSenses”, https://github.com/GaiaSenses, 2023.