UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

GABRIEL BORCHARDT

IMPLEMENTAÇÃO DE FERRAMENTAS ANALÍTICAS PARA O SISTEMA OPEN SOCIAL CARE: RELATÓRIOS ESTRUTURADOS E DASHBOARDS

GUARAPUAVA

2025

GABRIEL BORCHARDT

IMPLEMENTAÇÃO DE FERRAMENTAS ANALÍTICAS PARA O SISTEMA OPEN SOCIAL CARE: RELATÓRIOS ESTRUTURADOS E DASHBOARDS

Implementation of Analytical Tools for the Open Social Care System: Structured Reports and Dashboards

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Tecnólogo em Tecnologia em Sistemas para Internet do Curso Superior de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná.

Orientador: Dr. Andres Jessé Porfirio

GUARAPUAVA

2025

4.0 Internacional

Esta licença permite compartilhamento, remixe, adaptação e criação a partir do trabalho, mesmo para fins comerciais, desde que sejam atribuídos créditos ao(s) autor(es). Conteúdos elaborados por terceiros, citados e referenciados nesta obra não são cobertos pela licença.

GABRIEL BORCHARDT

IMPLEMENTAÇÃO DE FERRAMENTAS ANALÍTICAS PARA O SISTEMA OPEN SOCIAL CARE: RELATÓRIOS ESTRUTURADOS E DASHBOARDS

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Tecnólogo em Tecnologia em Sistemas para Internet do Curso Superior de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná.

Data de aprovação: 12/novembro/2025

Dênis Lucas Silva
Mestre
Universidade Tecnológica Federal do Paraná - Campus Guarapuava

---

Sediane Carmem Lunardi Hernandes
Doutura
Universidade Tecnológica Federal do Paraná - Campus Guarapuava

---

Andres Jessé Porfirio
Doutor
Universidade Tecnológica Federal do Paraná - Campus Guarapuava

GUARAPUAVA
2025

AGRADECIMENTOS

Agradeço primeiramente à minha família, pelo apoio constante desde o primeiro dia em que ingressei na universidade. Sou imensamente grato pelas oportunidades e condições que sempre me proporcionaram, permitindo que eu estudasse e me desenvolvesse ao longo de toda a minha vida.

Expresso minha gratidão ao meu orientador, Dr. Andres Jessé Porfirio, pela dedicação, paciência e pela orientação segura ao longo do desenvolvimento deste trabalho. Sua experiência e disponibilidade foram fundamentais para o amadurecimento das ideias e para a conclusão deste trabalho.

Agradeço também aos colegas que fizeram parte da minha trajetória acadêmica, pelas parcerias, conversas, incentivo e pela amizade construída durante esses anos de graduação.

À minha namorada, manifesto minha profunda gratidão por todo o apoio incondicional — desde acompanhar momentos de teste e desenvolvimento até oferecer suporte nos desafios cotidianos. Sua presença foi essencial para que eu mantivesse o equilíbrio e a motivação ao longo do processo.

Agradeço a Deus pela força, pela oportunidade de realizar este trabalho e por me permitir chegar até aqui, concluindo mais uma etapa importante da minha vida.

Por fim, agradeço a mim mesmo pelo esforço dedicado, pela persistência diante das dificuldades e pela disciplina com que conduzi esta jornada. Reconheço a importância da minha própria dedicação para alcançar este resultado.

RESUMO

A gestão de dados na assistência social frequentemente enfrenta desafios devido a processos manuais e descentralizados, dificultando a extração de *insights* para a tomada de decisão informada. Este trabalho **justifica-se** pela necessidade de ferramentas que transformem dados operacionais em informações estratégicas, visando otimizar a alocação de recursos e o monitoramento de políticas públicas. O **objetivo** principal foi desenvolver um módulo analítico integrado ao sistema Open Social Care. A **metodologia** adotada envolveu a extensão do sistema de formulários existente para suportar perguntas de múltipla escolha, a criação de *templates* padrão (CCG e Albergue) via *Seeders* do Laravel para padronizar a coleta de dados, e a implementação de um *script* de população (*Populate*) para gerar dados de demonstração. No *backend*, foi criada uma *view* SQL (`analytics_view`) para unificar as respostas e um *controller* (`AnalyticsController`) para processar os dados, gerando análises categorizadas específicas para cada *template*. No *frontend* (Next.js), foi desenvolvida uma interface com *dashboards* interativos, utilizando ApexCharts para gráficos e componentes customizados para KPIs, além de filtros dinâmicos por *template* e período. A funcionalidade de exportação de relatórios foi implementada, gerando ficheiros CSV no servidor (com `league/csv`) e PDF no cliente (com `jsPDF` e `jspdf-autotable`). Como **resultados**, o módulo analítico implementado permite a visualização dinâmica de indicadores demográficos, socioeconômicos e operacionais, adaptando a apresentação dos dados ao contexto de cada formulário (CCG ou Albergue), e oferece a exportação dos dados filtrados. **Conclui-se** que o projeto alcançou seus objetivos técnicos, entregando uma ferramenta funcional que contribui para a modernização da gestão na assistência social, embora testes formais de usabilidade sejam recomendados como trabalho futuro para refinar a interface.

Palavras-chave: assistência social; análise de dados; dashboard; sistema web; Laravel; Next.js.

ABSTRACT

Data management in social assistance often faces challenges due to manual and decentralized processes, hindering the extraction of insights for informed decision-making. This work **is justified** by the need for tools that transform operational data into strategic information, aiming to optimize resource allocation and the monitoring of public policies. The main **objective** was to develop an analytical module integrated into the Open Social Care system. The adopted **methodology** involved extending the existing form system to support multiple-choice questions, creating default templates (CCG and Albergue) via Laravel Seeders to standardize data collection, and implementing a data population script (Populate) to generate demonstration data. On the *backend*, an SQL *view* (`analytics_view`) was created to unify responses, and a *controller* (`AnalyticsController`) was implemented to process the data, generating categorized analyses specific to each template. On the *frontend* (Next.js), an interface with interactive *dashboards* was developed, using ApexCharts for graphs and custom components for KPIs, along with dynamic filters for template and period. The report export functionality was implemented, generating CSV files on the server (with `league/csv`) and PDF files on the client-side (with `jsPDF` and `jspdf-autotable`). As **results**, the implemented analytical module allows for the dynamic visualization of demographic, socioeconomic, and operational indicators, adapting the data presentation to the context of each form (CCG or Albergue), and offers filtered data export. It **is concluded** that the project achieved its technical objectives, delivering a functional tool that contributes to the modernization of management in social assistance, although formal usability tests are recommended as future work to refine the interface.

Keywords: social assistance; data analysis; dashboard; web system; Laravel; Next.js.

Keywords: keyword 1; keyword 2; keyword 3; keyword 4; keyword 5.

SUMÁRIO

1 INTRODUÇÃO 7

1.1 Considerações iniciais 7

1.2 Objetivos 8

1.2.1 Objetivo geral 8

1.2.2 Objetivos específicos 8

2 JUSTIFICATIVA 9

3 CONTEXTUALIZAÇÃO 10

4 DETALHAMENTO DA SOLUÇÃO 12

5 MATERIAIS E MÉTODOS 13

5.1 Materiais 13

5.1.1 Tecnologias Utilizadas 13

5.1.2 Ferramentas de Apoio 14

5.2 Métodos 14

5.2.1 Integração com o Sistema Existente 14

5.2.2 Modelagem dos Dashboards e Relatórios 15

5.2.3 Tratamento e Visualização dos Dados 16

5.2.4 Segurança e Confiabilidade 17

5.2.5 Organização do Desenvolvimento 18

6 RESULTADOS 20

6.1 Extensão do Sistema de Formulários 20

6.1.1 Implementação de Perguntas de Múltipla Escolha 20

6.1.2 Criação de Templates Padrão: CCG e Albergue 22

6.2 Implementação do Módulo Analítico 23

6.2.1 Filtros de Busca e Interatividade 24

6.2.2 Relatórios Estruturados e Desafios de Implementação 25

7 CONSIDERAÇÕES FINAIS 27

REFERÊNCIAS 29

A GUIA DE EXTENSIBILIDADE DO MÓDULO ANALÍTICO 30

A.1 Visão Geral da Arquitetura 30

A.2 Passo a Passo da Implementação 30
A.2.1 Parte 1: Modificações no Backend (Laravel) 30
A.2.1.1 Passo 1.1: Criar o Template de Formulário via Seeder 30
A.2.1.2 Passo 1.2: Registrar o Novo Seeder 31
A.2.1.3 Passo 1.3: Implementar a Lógica de Análise no Controller 31
A.2.2 Parte 2: Modificações no Frontend (Next.js) 33
A.2.2.1 Passo 2.1: Definir a Estrutura dos Dados (Schema Zod) 33
A.2.2.2 Passo 2.2: Criar o Componente do Dashboard 34
A.2.2.3 Passo 2.3: Integrar o Novo Dashboard na Página Principal 34
ANEXO A FICHA DE ATENDIMENTO DO CONSELHO DA COMUNIDADE DE GUARAPUAVA (CCG) 37
ANEXO B FICHA DO PLANO DE ACOMPANHAMENTO DO ALBERGUE . . 39

1 INTRODUÇÃO

Neste capítulo, será explorado o contexto de aplicabilidade do trabalho, destacando sua relevância, a necessidade de sua implementação e a problemática que busca solucionar.

1.1 Considerações iniciais

A era digital transformou significativamente a maneira como organizações e governos operam, colocando os dados analíticos no centro das decisões estratégicas. A capacidade de coletar, processar e interpretar grandes volumes de dados tem se tornado cada vez mais essencial para aprimorar a eficiência operacional, identificar tendências emergentes e responder proativamente às necessidades da sociedade. No setor público, especialmente na assistência social, a utilização de ferramentas analíticas permite uma compreensão mais profunda dos desafios enfrentados pelas populações vulneráveis, possibilitando a formulação de políticas mais assertivas e a alocação eficaz de recursos (FONSECA, 2023).

O Brasil tem avançado na modernização de seus processos de assistência social, reconhecendo a importância da tecnologia e da gestão da informação para aprimorar os serviços oferecidos à população. Uma iniciativa notável é o "Observatório do Cadastro Único"[1], lançado em 2023 pelo Ministério de Desenvolvimento e Assistência Social, Família e Combate à Fome (MDS). Esta ferramenta fornece um painel interativo de acesso público, atualizado dinamicamente pelos Centros de Referência de Assistência Social (CRAS), permitindo obter dados detalhados sobre indivíduos cadastrados, abrangendo aspectos como moradia, escolaridade e acesso ao trabalho. O Observatório visa promover transparência e humanizar o Cadastro Único, fornecendo diagnósticos precisos que subsidiam ações mais eficazes no combate à pobreza.

Nesse contexto, o Open Social Care é um sistema de código aberto desenvolvido para a gestão de atendimentos sociais. Diante do cenário de transformação digital, a implementação de um módulo analítico dentro desse sistema surge como uma solução estratégica para potencializar a gestão da assistência social na cidade de Guarapuava (ver Capítulo 3 para mais detalhes sobre o sistema). Ao integrar relatórios estruturados e *dashboards* interativos, o módulo permitirá aos gestores uma visualização clara e dinâmica dos dados, facilitando a identificação de padrões e tendências. Por exemplo, ao analisar dados referentes a indivíduos que buscam assistência social devido ao uso de substâncias específicas, os gestores poderão concentrar esforços em políticas preventivas direcionadas e cobrar ações mais eficazes do poder público. Essa abordagem baseada em dados não apenas otimiza a alocação de recursos, mas também aprimora a prestação de serviços à sociedade, garantindo que as intervenções sejam mais precisas e alinhadas às necessidades reais da população atendida. Portanto, esse documento propõe o desenvolvimento e implementação de um módulo analítico no sistema Open

[1] Disponível em: https://l1nq.com/rMfZy

Social Care, funcionalidades esperadas e os benefícios esperados para a gestão pública da assistência social em Guarapuava.

1.2 Objetivos

1.2.1 Objetivo geral

Desenvolver um módulo analítico para o Open Social Care, integrando relatórios estruturados e *dashboards* interativos para otimizar a gestão da assistência social, possibilitando a identificação de padrões e tendências para a tomada de decisões estratégicas.

1.2.2 Objetivos específicos

- Revisar e complementar o módulo de criação de questionários do sistema Open Social Care, identificando e permitindo a extração de dados estatísticos dos atendimentos realizados;
- Desenvolver relatórios estruturados que apresentem dados relevantes da assistência social de forma clara e acessível;
- Criar *dashboards* interativos que permitam a visualização dinâmica das informações, facilitando a análise por parte dos gestores;
- Identificar padrões e tendências nos dados da assistência social por meio das visualizações criadas;
- Proporcionar uma interface amigável que favoreça a usabilidade do módulo analítico para os usuários do sistema Open Social Care;
- Conduzir testes de usabilidade e eficiência para validar a eficácia do módulo desenvolvido em cenários reais de gestão.

2 JUSTIFICATIVA

A crescente complexidade dos desafios enfrentados pela assistência social em cidades como Guarapuava evidencia a necessidade urgente de ferramentas que promovam uma gestão mais eficiente e baseada em dados. Profissionais da área frequentemente lidam com informações dispersas, o que dificulta a identificação de padrões, a formulação de políticas públicas e a prestação de serviços adequados à população vulnerável (DATASUAS, 2023). A criação de um módulo analítico, que inclua relatórios estruturados e dashboards interativos, surge como uma solução essencial para otimizar as operações na assistência social, permitindo que os gestores tenham acesso a dados organizados e significativos.

Os dados coletados por assistentes sociais são fundamentais para compreender a realidade das comunidades atendidas. No entanto, sem um sistema eficiente de análise, esses dados permanecem subutilizados, limitando a capacidade dos profissionais em tomar decisões baseadas em dados. Ao estruturar e apresentar essas informações de maneira clara e acessível, é possível identificar rapidamente as áreas que necessitam de intervenção. Por exemplo, ao analisar dados sobre o acesso a serviços de saúde e educação, fica evidente quais grupos específicos enfrentam dificuldades, permitindo direcionar recursos de forma mais eficiente e eficaz (FEDERAL, 2023).

Profissionais da assistência social podem monitorar indicadores-chave de desempenho em tempo real, possibilitando uma atuação mais ágil e proativa. Essa abordagem não apenas aumenta a eficiência das operações, mas também contribui para um atendimento mais humano e personalizado, ajustando as intervenções às necessidades específicas de cada caso. O acesso a informações organizadas e visualmente intuitivas capacita os profissionais a responderem rapidamente a situações emergenciais, garantindo que as necessidades da população sejam atendidas de forma oportuna (GOV.BR, 2023).

Além disso, a implementação de um módulo analítico proporciona maior transparência e prestação de contas. Com dados apresentados de forma clara, os gestores podem justificar as ações tomadas e os recursos alocados de maneira mais convincente. Essa transparência é essencial para construir a confiança da comunidade nos serviços de assistência social e o engajamento da mesma sobre a assistência social, o que contribui para a melhoria contínua das práticas adotadas.

Por fim, ao possibilitar uma gestão mais eficiente e baseada em dados, o módulo analítico atende tanto às demandas dos profissionais da área quanto às necessidades da sociedade. A promoção de um atendimento mais eficaz e direcionado às populações vulneráveis contribui para a construção de uma cidade mais justa e solidária, onde todos têm acesso a serviços que atendem suas necessidades básicas. Portanto, a necessidade de desenvolver um módulo analítico é evidente e crucial para a melhoria da assistência social em Guarapuava, impactando diretamente a qualidade de vida da população atendida.

3 CONTEXTUALIZAÇÃO

A assistência social desempenha um papel fundamental na promoção do bem-estar de populações vulneráveis, garantindo o acesso a direitos básicos e serviços essenciais. Com o avanço da tecnologia, diversas áreas da gestão pública e privada passaram por processos de modernização que permitiram maior eficiência na organização de dados e na tomada de decisões. No entanto, a incorporação de tecnologias avançadas para análise e gestão de informações na assistência social ainda se encontra em um estágio inicial de desenvolvimento (SOUZA, 2024). Tradicionalmente, o registro de atendimentos e a administração de informações no setor ocorrem de forma descentralizada, dificultando a obtenção de dados estruturados que auxiliem no planejamento de políticas sociais.

O Open Social Care surgiu como uma resposta a essas limitações, buscando fornecer uma plataforma unificada para o registro e acompanhamento de atendimentos. O sistema permite que profissionais da assistência social organizem e documentem interações com os usuários, promovendo maior controle sobre os atendimentos realizados e fornecendo um histórico detalhado para a tomada de decisões (BOEIRA, 2024). Inicialmente projetado para atender demandas institucionais específicas ligadas à execução penal, sua aplicação foi expandida para diferentes contextos, tornando-se uma ferramenta para a gestão da assistência social como um todo. No entanto, sua evolução trouxe consigo novos desafios, principalmente no que diz respeito à experiência do usuário e à capacidade de análise dos dados coletados.

Uma das principais lacunas encontradas no contexto da assistência social digital é a ausência de ferramentas para análise de dados. Atualmente, os dados registrados no Open Social Care servem essencialmente para operação, mas há pouco suporte para a extração de *insights* estratégicos a partir dessas informações. A ausência de *dashboards* interativos e relatórios personalizados limita a capacidade de identificar padrões, avaliar tendências e prever demandas futuras, o que reduz a eficiência da gestão social e a capacidade de resposta a desafios em crescimento.

A carência de mecanismos analíticos na assistência social contrasta com outras áreas que já adotaram sistemas de inteligência de dados para otimizar processos e fundamentar decisões estratégicas. Enquanto setores como saúde e segurança pública contam com plataformas que permitem visualizações avançadas, a assistência social ainda opera em sua maior parte com registros estáticos e análises manuais. Esse atraso tecnológico impacta diretamente a formulação de políticas públicas e a alocação de recursos, dificultando intervenções assertivas e baseadas em evidências.

O aprimoramento do Open Social Care por meio de melhorias na funcionalidade de cadastro e na introdução de um módulo analítico representa uma oportunidade significativa para transformar a gestão da assistência social. A possibilidade de visualizar dados em tempo real, gerar relatórios automatizados e monitorar indicadores-chave permitirá que gestores e profissionais atuem de forma mais estratégica e proativa. Acredita-se que essas melhorias não apenas

facilitarão a execução do trabalho diário dos assistentes sociais, mas também contribuirão para uma gestão mais transparente e eficiente, beneficiando diretamente a população atendida.

4 DETALHAMENTO DA SOLUÇÃO

A solução proposta visa desenvolver um módulo analítico para aprimorar a gestão de dados na área de assistência social. Este módulo integrará relatórios estruturados e *dashboards* interativos, permitindo a visualização e análise eficiente das informações coletadas pelos profissionais da área. A iniciativa busca solucionar a dificuldade de interpretação dos dados, que comprometem a eficácia na identificação de demandas e na formulação de políticas públicas adequadas.

A solução será implementada por meio do desenvolvimento de um módulo analítico que se integrará ao sistema existente. Este módulo permitirá a geração de relatórios detalhados e a criação de *dashboards* interativos, facilitando a visualização de indicadores-chave e tendências. A abordagem adotará técnicas de visualização de dados e análise estatística, como gráficos de barras para comparações quantitativas, gráficos de linha para identificar tendências e gráficos de dispersão para correlações entre variáveis, garantindo que as informações sejam apresentadas de forma clara e acessível aos usuários. Além disso, serão incorporadas funcionalidades que possibilitem a personalização dos relatórios e *dashboards*, atendendo às necessidades específicas dos gestores e profissionais da assistência social.

Os principais beneficiários desta proposta são os gestores e profissionais que atuam na área de assistência social, que terão acesso a ferramentas mais eficazes para a análise e interpretação de dados. Isso resultará em uma gestão mais eficiente e em intervenções mais precisas junto às populações vulneráveis. Indiretamente, a sociedade como um todo se beneficiará, pois as políticas públicas poderão ser formuladas com base em informações mais precisas, promovendo um atendimento mais adequado às necessidades da população.

Espera-se que a implementação do módulo analítico resulte em uma melhoria significativa na gestão de dados na assistência social. Os profissionais terão à disposição ferramentas que facilitarão a identificação de padrões e tendências, permitindo uma alocação mais eficiente de recursos e a definição de estratégias de intervenção mais eficazes. Além disso, a transparência e a prestação de contas serão aprimoradas, uma vez que os dados estarão organizados e acessíveis para análises e auditorias.

A solução contribuirá para a melhoria dos serviços prestados à população em situação de vulnerabilidade, promovendo uma sociedade mais justa e equitativa.

5 MATERIAIS E MÉTODOS

Este capítulo descreve os recursos tecnológicos e as ferramentas de apoio que constituem a base do sistema Open Social Care, sobre a qual o módulo analítico foi desenvolvido. Adicionalmente, detalha os métodos aplicados para a construção da nova funcionalidade, desde a extensão da base de dados até a implementação da interface de visualização e exportação de dados.

5.1 Materiais

A construção do módulo analítico aproveitou a infraestrutura tecnológica já existente no projeto Open Social Care, complementada por bibliotecas específicas para atender aos novos requisitos de visualização e exportação de dados.

5.1.1 Tecnologias Utilizadas

Frontend: A interface de usuário (*frontend*) do Open Social Care é desenvolvida utilizando o *framework* **Next.js** (versão 14.1.0), baseado na biblioteca **React** (versão 18) e escrito em **TypeScript**. A arquitetura adotada é a do **App Router**, que permite a composição de interfaces através de Componentes de Servidor (para busca de dados e lógica no servidor) e Componentes de Cliente (para interatividade no navegador). A estilização é realizada com **Tailwind CSS**, e a validação de dados no lado do cliente utiliza a biblioteca **Zod**. O ambiente de execução necessário para o desenvolvimento e compilação do *frontend* é o **Node.js** (versão 18), utilizando `npm` ou `yarn` como gerenciadores de pacotes. Para o módulo analítico, a biblioteca **ApexCharts**, juntamente com seu wrapper para React (`react-apexcharts`), foi adicionada para a renderização dos gráficos interativos.

Backend: A lógica de negócio e a API (*backend*) do sistema são implementadas com o *framework* **Laravel** (versão 10.10), utilizando a linguagem **PHP** (versão 8.1). A arquitetura segue o padrão MVC (Model-View-Controller), com o **Eloquent ORM** para interação com o banco de dados e a API sendo exposta através de rotas RESTful. O ambiente de desenvolvimento é padronizado e gerenciado via **Docker** através do **Laravel Sail**, que simplifica a configuração e execução dos contêineres necessários (servidor web, PHP, banco de dados, etc.). A gestão de dependências do PHP é realizada pelo **Composer**. Para a funcionalidade de exportação de relatórios, inicialmente planejou-se o uso das bibliotecas `Laravel Excel` e `DomPDF`. A `Laravel Excel` foi escolhida por sua popularidade e capacidade de manipulação de planilhas, facilitando a criação de arquivos CSV compatíveis com o Microsoft Excel. A `DomPDF` foi selecionada por sua habilidade em converter HTML e CSS para o formato PDF, permitindo a

geração de relatórios com layout definido através de *views* Blade. Contudo, desafios de implementação levaram a uma reavaliação (detalhada no Capítulo 6).

Banco de Dados: O sistema utiliza o **PostgreSQL** como Sistema de Gerenciamento de Banco de Dados (SGBD) relacional. A interação entre a aplicação Laravel e o banco de dados é primariamente realizada através do **Eloquent ORM**, que abstrai as consultas SQL. A estrutura do banco de dados é gerenciada e versionada utilizando o sistema de **Migrations** do Laravel, que permite a criação e modificação de tabelas de forma controlada. Para a população inicial de dados essenciais (como usuários padrão, perfis de acesso e, neste trabalho, os *templates* de formulário padrão), utiliza-se o sistema de **Seeders** do Laravel, garantindo que novas instâncias da aplicação possuam a configuração mínima necessária. Este trabalho expandiu o uso de *Seeders* para incluir os *templates* CCG e Albergue, além de um *seeder* opcional para dados de demonstração.

5.1.2 Ferramentas de Apoio

O desenvolvimento do projeto Open Social Care, incluindo o módulo analítico, foi suportado por um conjunto de ferramentas que visam facilitar a colaboração, o controle de versão e a gestão do projeto.

O controle de versão do código-fonte é realizado utilizando o **Git**, com o repositório hospedado na plataforma **GitHub**. O GitHub serve não apenas como repositório central, mas também como plataforma para revisão de código (*Pull Requests*) e discussão de implementações.

O ambiente de desenvolvimento local foi padronizado uso de **Docker** e **Laravel Sail**, garantindo que a aplicação rodasse de forma consistente independentemente do sistema operacional utilizado por quem instalar o código-fonte.

5.2 Métodos

Esta seção detalha o processo metodológico adotado para o desenvolvimento do módulo analítico, desde a integração com a arquitetura preexistente do Open Social Care até a implementação das funcionalidades de visualização, filtragem e exportação de dados. O objetivo é fornecer um roteiro claro que permita a compreensão e, potencialmente, a reprodução do trabalho realizado.

5.2.1 Integração com o Sistema Existente

O Open Social Care opera sob uma arquitetura de microsserviços desacoplada, composta por uma *API backend* desenvolvida em Laravel e uma interface *frontend* separada, cons-

truída com Next.js. A comunicação entre estas duas camadas ocorre via chamadas *RESTful*, onde cada funcionalidade depende de *endpoints* específicos definidos na API.

Para a implementação do módulo analítico, foi necessário estender a API existente com novas rotas, dedicadas exclusivamente a fornecer os dados processados e as opções de filtragem para o *frontend*. A Tabela 1 detalha as novas rotas criadas como parte deste trabalho, especificando o método HTTP, o URI e a finalidade de cada *endpoint*. É importante ressaltar que todas estas rotas foram adicionadas aos arquivos `routes/api.php` do *backend*.

Tabela 1 – Novas Rotas da API Criadas para o Módulo Analítico

Método	URI	Descrição
GET	/api/[perfil]/form-templates/select/{organization}	Retorna uma lista simplificada (ID e Título) dos *templates* de formulário associados a uma organização, utilizada para popular a caixa de seleção de filtros.
GET	/api/[perfil]/analytics/form-template/{form_template}/{period?}	Endpoint principal que busca, processa e retorna os dados analíticos agregados para um *template* específico, com filtro de período opcional.
GET	/api/[perfil]/analytics/form-template/{form_template}/export/{format}	Endpoint responsável por gerar e fornecer o *download* do relatório analítico nos formatos CSV ou PDF, aplicando os mesmos filtros de *template* e período.

Nota: O segmento `[perfil]` representa `manager` ou `social-assistant`, dependendo do grupo de middleware.

A criação destas rotas seguiu o padrão já estabelecido no projeto, garantindo a integração coesa do novo módulo com a arquitetura existente.

5.2.2 Modelagem dos Dashboards e Relatórios

Com base nos requisitos levantados e nos *templates* de formulário definidos (CCG e Albergue), foram modelados dois *dashboards* analíticos específicos. O objetivo foi transformar os dados brutos coletados em informações visuais e de fácil interpretação para os gestores e assistentes sociais.

- **Dashboard CCG:** Focado em fornecer uma visão geral do perfil socioeconômico e demográfico dos atendidos pelo Conselho da Comunidade. Os cards implementados incluem:
  - *Distribuição de Gênero* (Gráfico de Pizza): Proporção entre gêneros.
  - *Distribuição por Faixa Etária* (Gráfico de Barras): Contagem de atendidos por idade.
  - *Tamanho do Núcleo Familiar* (Gráfico de Barras): Distribuição do número de pessoas por domicílio.

  - *Nível de Escolaridade* (Gráfico de Barras Horizontais): Comparação dos níveis educacionais.
  - *Situação de Trabalho* (Gráfico de Pizza): Proporção de status de emprego.
  - *Principais Fontes de Renda* (Gráfico de Barras): Frequência das fontes de renda declaradas.
  - *Principais Encaminhamentos* (Gráfico de Barras): Serviços mais acionados.
  - *Bairros Mais Atendidos* (Gráfico de Barras Horizontais): Ranking geográfico.
  - *KPIs de Destaque:* Total de atendimentos, total de bairros mapeados, etc.

- **Dashboard Albergue:** Direcionado ao perfil específico do público atendido pelo Albergue, com ênfase em mobilidade, saúde e cidadania. Os cards incluem:
  - *Visitantes Recorrentes* (Gráfico de Pizza): Proporção de atendidos que já estiveram na cidade.
  - *Principais Cidades de Origem* (Gráfico de Barras Horizontais): Ranking de origem.
  - *Perfil de Saúde* (Gráficos de Pizza e KPIs): Análise sobre doenças crônicas, uso de substâncias, condições psicológicas e acesso a tratamento/medicação.
  - *Perfil de Cidadania* (KPIs): Indicadores sobre posse de documentos (RG, CPF) e inscrição no CadÚnico.

Ambos os *dashboards* são integrados com os **filtros de Template e Período**, permitindo que o usuário refine a análise conforme a necessidade. A seleção destes filtros na interface atualiza os parâmetros na URL, que são então utilizados pela API para buscar e processar os dados correspondentes no servidor. Os relatórios exportados em PDF e CSV refletem exatamente os dados exibidos no *dashboard* com os filtros aplicados no momento da exportação.

5.2.3 Tratamento e Visualização dos Dados

A transformação dos dados brutos em *insights* é realizada no *backend*, especificamente no `AnalyticsController`. A metodologia aplicada consiste em:

1. Consultar a *view* `analytics_view`, aplicando os filtros de *template* e período recebidos via *request*.
2. Utilizar uma estrutura `match` para direcionar a coleção de respostas para uma função de processamento específica, baseada no título do *template* (`processCcgAnalytics`, `processAlbergueAnalytics`).

3. Dentro de cada função de processamento, agrupar as respostas por pergunta (`groupBy('question_description')`).

4. Para perguntas de múltipla escolha, contar a ocorrência de cada resposta (`map(fn ($group) => $group->count())`).

5. Para perguntas específicas (como Data de Nascimento ou Endereço), aplicar lógicas customizadas para extrair informações relevantes (cálculo de faixa etária com `Carbon::parse()->age`, extração de bairro via `explode()`).

6. Estruturar o resultado final em um *array* associativo categorizado (ex: `['demographic' => [...], 'socioeconomic' => [...]]`), que é retornado como JSON para o *frontend*.

No *frontend*, a visualização é realizada pelos componentes React especializados (`CcgDashboard`, `AlbergueDashboard`). Estes componentes recebem o JSON pré-processado do *backend*, iteram sobre as categorias e *insights*, e utilizam a biblioteca **ApexCharts** (via `react-apexcharts`) para renderizar os gráficos de pizza e barras, e componentes customizados (`KpiCard`) para exibir os indicadores numéricos.

5.2.4 Segurança e Confiabilidade

A segurança do módulo analítico é garantida através do sistema de autenticação e autorização já existente no Laravel, baseado em **Políticas (Policies)** e **Middleware**.

O Laravel permite definir *Policies*, que são classes PHP que encapsulam a lógica de autorização para ações específicas em um determinado modelo (ex: um usuário pode *ver* um `FormTemplate`?). Estas políticas são registadas e podem ser verificadas nos *controllers* através de métodos como `$this->authorize('view', $formTemplate)`.

Adicionalmente, o sistema utiliza *Middleware* para controlar o acesso a grupos inteiros de rotas com base no perfil (Role) do usuário autenticado (ex: `only_manager_user`, `only_social_assistant_user`).

A Tabela 2 detalha as autorizações aplicadas a cada nova rota do módulo analítico, garantindo que apenas usuários com as permissões adequadas possam acessar aos dados.

Tabela 2 – Autorizações Aplicadas às Novas Rotas do Módulo Analítico

Método	URI	Autorização Aplicada
GET	/api/[perfil]/form-templates/select/{organization}	Verificação via *Middleware* (`only_manager_user` ou `only_social_assistant_user`) + Política `viewForOrganization` no modelo `FormTemplate`.
GET	/api/[perfil]/analytics/form-template/{form_template}/{period?}	Verificação via *Middleware* (`only_manager_user` ou `only_social_assistant_user`) + Autorização implícita via *Route-Model Binding* (garante que o `FormTemplate` existe e pertence ao contexto do usuário).
GET	/api/[perfil]/analytics/form-template/{form_template}/export/{format}	Verificação via *Middleware* (`only_manager_user` ou `only_social_assistant_user`) + Autorização implícita via *Route-Model Binding*.

Nota: O segmento `[perfil]` representa `manager` ou `social-assistant`.

Esta abordagem em camadas assegura que os dados analíticos, potencialmente sensíveis, sejam acessados apenas por pessoal autorizado.

5.2.5 Organização do Desenvolvimento

O fluxo de trabalho adotado para a implementação do módulo seguiu o padrão **Feature Branch Workflow**(Atlassian, 2024). Este modelo, suportado pelo Git e GitHub, promove o isolamento do desenvolvimento de novas funcionalidades em *branches* separadas, facilitando a revisão de código (*Pull Requests*) e a integração contínua sem comprometer a estabilidade da *branch* principal (`main` ou `develop`).

O controle das tarefas foi realizado utilizando o quadro **Kanban** no GitHub Projects, mantendo a metodologia já empregada nas fases anteriores do projeto. As tarefas eram decompostas em *Issues*, discutidas, atribuídas e acompanhadas visualmente através das colunas do quadro.

O ciclo iterativo de desenvolvimento para cada funcionalidade do módulo analítico seguiu as etapas: Definição da tarefa $\rightarrow$ Criação de *branch* $\rightarrow$ Desenvolvimento e testes locais $\rightarrow$ *Pull Request* para revisão $\rightarrow$ Integração (*merge*) $\rightarrow$ Atualização do Kanban. Este processo garante rastreabilidade, qualidade e colaboração eficaz.

Figura 1 – Fluxograma do processo de desenvolvimento do módulo analítico



Fonte: Elaborado pelo autor (2025).

6 RESULTADOS

Este capítulo detalha a implementação e os resultados obtidos no desenvolvimento do módulo analítico para o sistema Open Social Care. As funcionalidades aqui descritas foram construídas para transformar os dados operacionais da plataforma em insights estratégicos, viabilizando a análise de indicadores e a geração de relatórios dinâmicos para a gestão da assistência social.

6.1 Extensão do Sistema de Formulários

Para que o módulo analítico pudesse extrair informações ricas e categorizadas, foi necessário, primeiramente, aprimorar a funcionalidade de criação de formulários do sistema, que até então se limitava a perguntas de texto livre.

6.1.1 Implementação de Perguntas de Múltipla Escolha

A análise mais básica de indicadores sociais depende da capacidade de agrupar e contar dados categóricos (ex: gênero, escolaridade). Para atender a este requisito, o sistema foi estendido para suportar perguntas de múltipla escolha.

Esta implementação exigiu a modelagem de novas entidades no banco de dados, materializadas através do sistema de **Migrations** do Laravel. A Figura 2 apresenta o recorte do Diagrama de Entidade-Relacionamento (DER) resultante.

A estrutura centra-se na tabela `multiple_choice_questions`, que armazena o enunciado e as configurações da pergunta (como o campo `answer_required`). Esta tabela estabelece um relacionamento de **um-para-muitos (1:N)** com a tabela `form_templates`, permitindo que um único formulário componha-se de diversas questões.

Para viabilizar a seleção de respostas, foi criada a tabela `multiple_choice_options`, que também mantém uma relação de **um-para-muitos** com a tabela de questões. Isso permite que cada pergunta possua uma lista dinâmica e ilimitada de alternativas (opções) cadastradas.

Por fim, a tabela `multiple_choice_answers` é responsável por armazenar as respostas efetivas dos atendimentos. Conforme destacado no diagrama, esta tabela inclui a coluna `question_description`. Este campo foi adicionado estrategicamente para garantir a **integridade histórica** dos dados: ao salvar uma cópia textual do enunciado da pergunta no momento da resposta, o sistema assegura que alterações futuras no *template* não corrompam o sentido das respostas passadas.

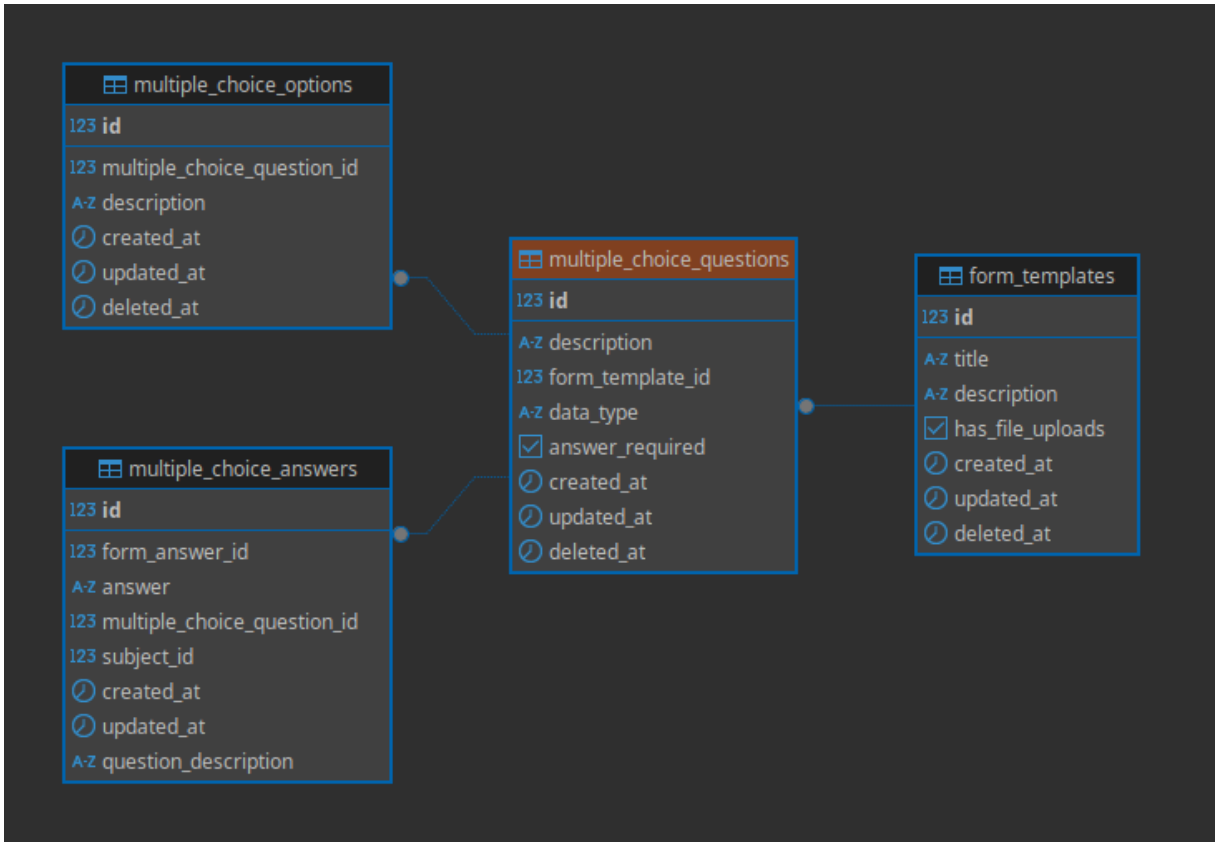


Figura 2 – Recorte do Diagrama ER detalhando o relacionamento entre templates, questões, opções e respostas.

Com a base de dados estruturada, a arquitetura de software foi refatorada. No *backend*, *controllers* e *API Resources* foram implementados para prover as funcionalidades de CRUD para estas novas entidades. No *frontend*, a interface de gerenciamento de *templates* foi aprimorada para refletir essa estrutura de dados, permitindo a criação e edição dinâmica das opções, conforme ilustrado na Figura 3.

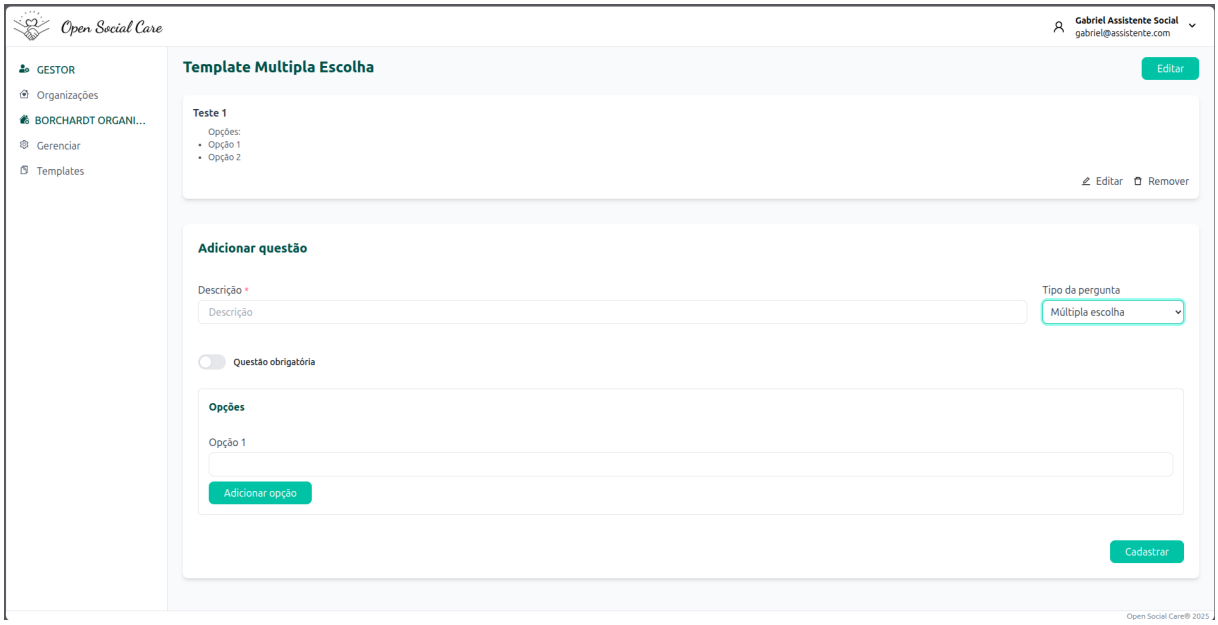


Figura 3 – Interface de criação/edição de perguntas de múltipla escolha no frontend.

6.1.2 Criação de Templates Padrão: CCG e Albergue

Com a funcionalidade de múltipla escolha implementada, o próximo passo foi criar templates de atendimento padronizados para garantir a coleta de dados consistentes. Foram criados dois templates principais com base em fichas de atendimento reais: a do Conselho da Comunidade de Guarapuava (CCG) (ver Anexo A) e a do Plano de Acompanhamento do Albergue (ver Anexo B).

Para garantir que estes templates estejam sempre disponíveis em novas instalações do sistema, foram desenvolvidas classes **Seeder** no Laravel. Os *seeders* `CcgFormTemplateSeeder` e `AlbergueFormTemplateSeeder` são responsáveis por popular o banco de dados com a estrutura completa destes dois formulários. O `CcgFormTemplateSeeder`, por exemplo, cria um *template* com 10 perguntas (7 de resposta curta e 3 de múltipla escolha), enquanto o `AlbergueFormTemplateSeeder` cria um *template* mais extenso com aproximadamente 35 perguntas (entre respostas curtas e múltiplas escolhas), ambos baseados nas fichas originais referenciadas nos anexos. Estes *seeders* também associam os *templates* criados a todas as organizações existentes no momento da sua execução.

População de Dados para Demonstração (Populate)

Para viabilizar a demonstração e os testes do módulo analítico, foi criado um **script de população** (implementado como um Seeder opcional do Laravel), denominado `DemoDataSeeder`. Diferente dos *seeders* principais, este não é executado por padrão e pode

ser chamado via linha de comando (`db:seed --class=DemoDataSeeder`) para inflar a base de dados com um grande volume de informações fictícias.

Utilizando as **Factories** do Laravel, este script cria **50** atendidos (*Subjects*) fictícios e, para cada um, preenche aleatoriamente entre **1 a 3** formulários do CCG (ou Albergue, dependendo da configuração), gerando um total de **50 a 150** respostas de formulário (`form_answers`) na base. As respostas são geradas de forma contextualizada (ex: usando a data de nascimento do *subject* para a pergunta correspondente), o que permite uma visualização rica e funcional dos dashboards em ambientes de desenvolvimento e apresentação (*showcases*).

6.2 Implementação do Módulo Analítico

Com uma base de dados estruturada e populada, foi desenvolvido o módulo analítico, cujo objetivo é consolidar e apresentar os dados de forma visual e intuitiva.

A fundação técnica deste módulo é uma **view** de banco de dados, denominada `analytics_view`. Esta tabela virtual foi criada via *Migration* do Laravel e tem a responsabilidade de unir os dados das tabelas de respostas (`short_answers` e `multiple_choice_answers`) com informações das tabelas de perguntas (`short_questions` e `multiple_choice_questions`) e do formulário principal (`form_answers`). A *view* retorna uma estrutura de dados plana e otimizada para consultas, contendo colunas como `form_template_id`, `question_id`, `question_description`, `question_type`, `answer`, `subject_id`, `user_id` e `created_at`. Um modelo Eloquent *read-only*, `AnalyticsData`, foi criado para interagir com esta *view* de forma abstraída no código Laravel.

No *backend*, o `AnalyticsController` consulta esta *view* e processa os dados brutos. Para os templates do CCG e do Albergue, ele categoriza os resultados em seções de insights (Demográfico, Socioeconômico, Saúde, etc.), que são entregues ao *frontend* em formato JSON, já prontos para a visualização.

A interface final, como ilustrado nas Figuras 4 e 5, consolida estes dados em um painel interativo. A Figura 4 apresenta o dashboard do CCG, destacando gráficos de pizza para distribuição de gênero e situação de trabalho, além de KPIs no topo com totais de atendimento. Já a Figura 5 exibe o dashboard do Albergue, focado em métricas de saúde e cidadania, evidenciando a flexibilidade do sistema em adaptar a visualização ao contexto do formulário. (A implementação detalhada do controller pode ser consultada no Apêndice A).

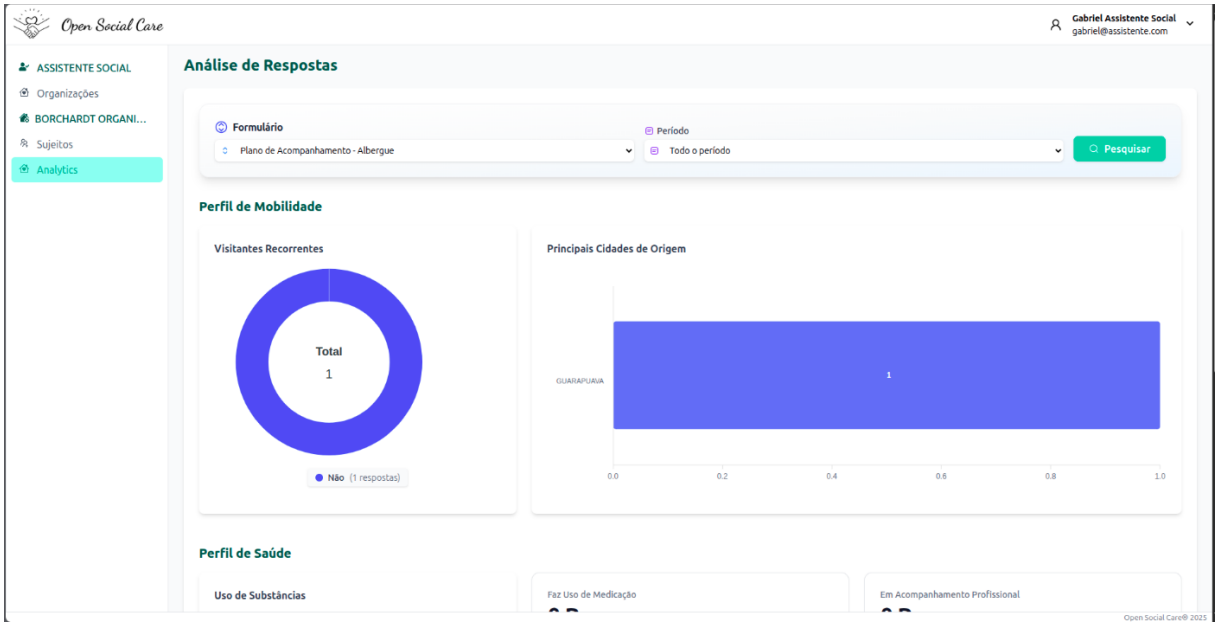


Figura 4 – Tela principal do Dashboard de Análises exibindo os dados do template CCG.

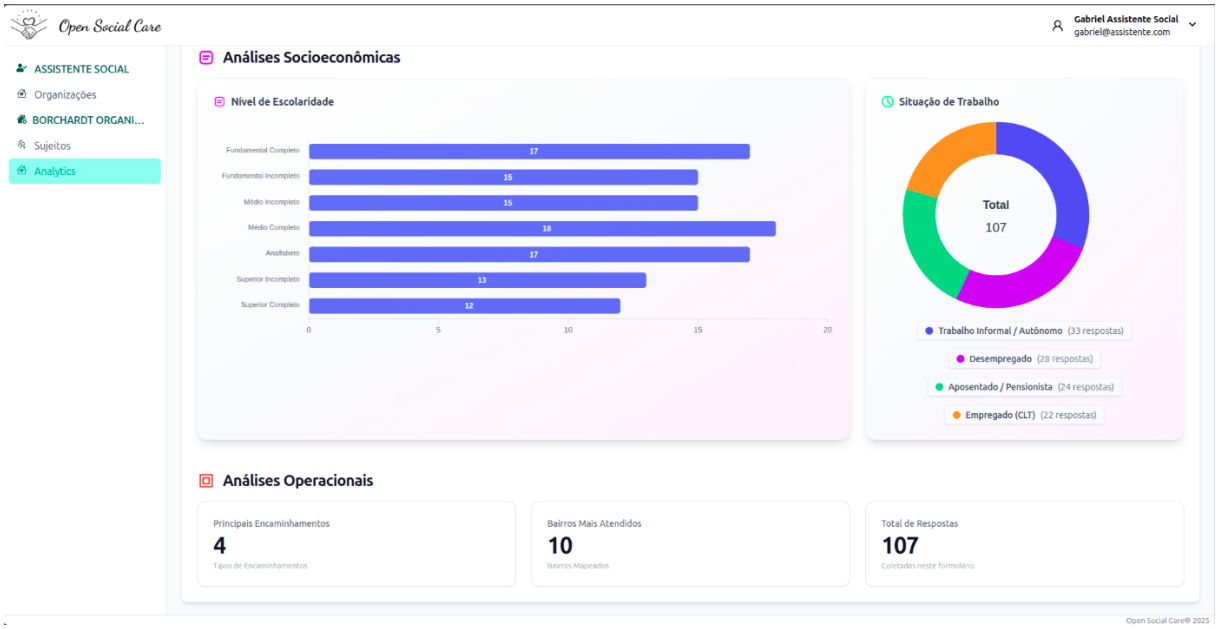


Figura 5 – Tela principal do Dashboard de Análises exibindo os dados do template Albergue.

6.2.1 Filtros de Busca e Interatividade

Para proporcionar uma experiência analítica flexível, a interface do módulo foi construída com filtros dinâmicos. Foram implementados dois filtros principais:

- **Seleção de Template:** Um menu suspenso que permite ao usuário escolher qual formulário deseja analisar.
- **Filtro de Período:** Uma caixa de seleção com opções pré-definidas (últimos 7 dias, 1 mês, 1 ano, etc.) para refinar a análise temporal dos dados.

Estes filtros foram implementados no *frontend*, como demonstrado na Figura 6. A imagem destaca a barra de ferramentas superior do módulo, onde o usuário pode selecionar o formulário e o período desejados antes de clicar no botão "Pesquisar"para atualizar os gráficos.

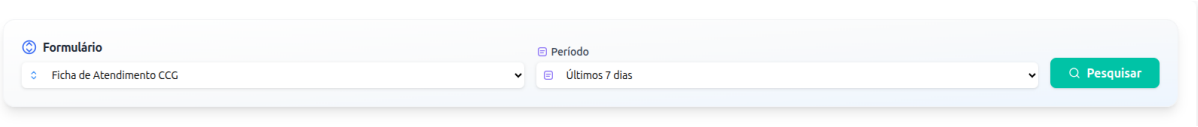


Figura 6 – Componente de filtros implementado no frontend do módulo analítico.

6.2.2 Relatórios Estruturados e Desafios de Implementação

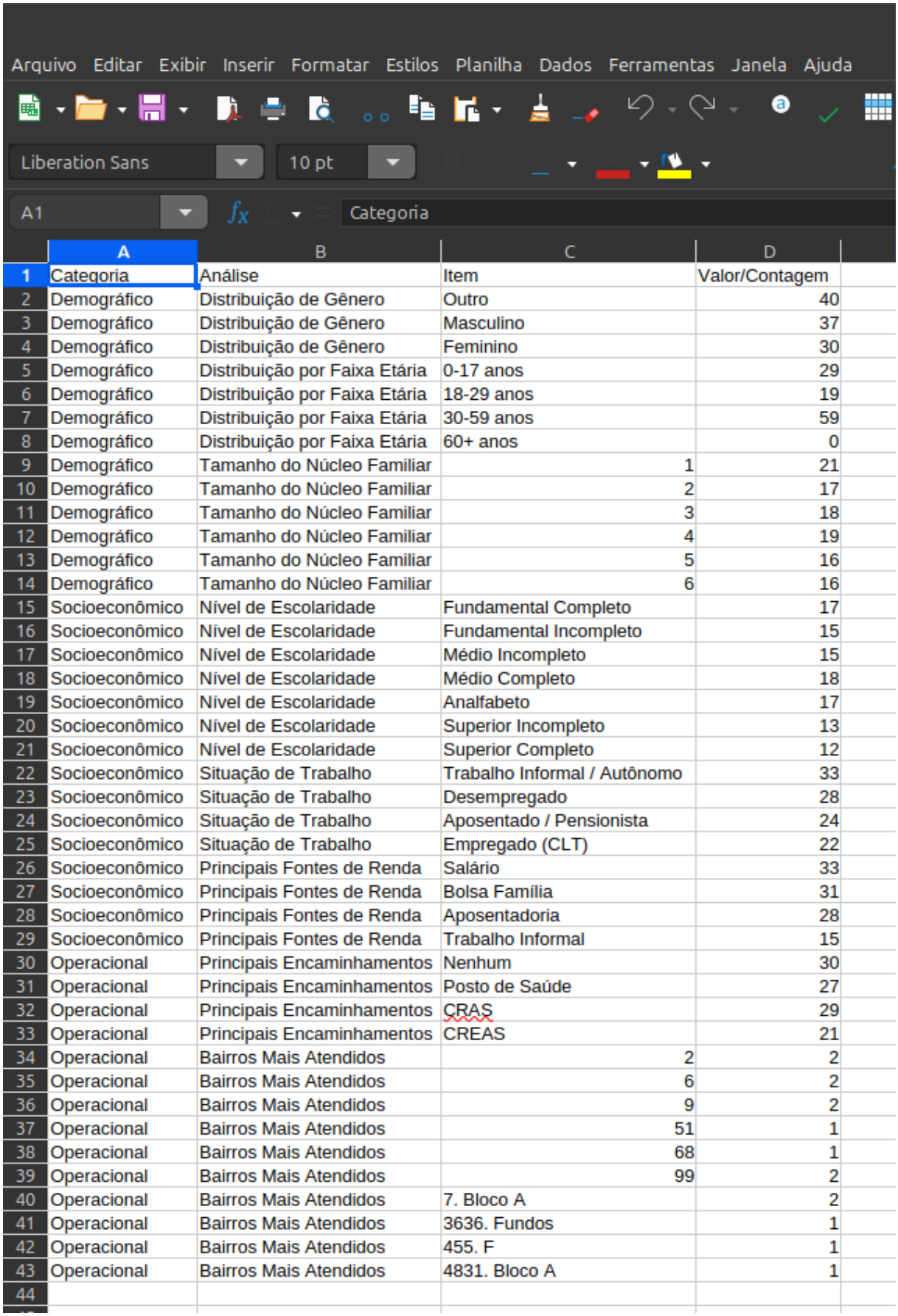
A funcionalidade de exportação de relatórios foi implementada nos formatos **PDF** e **CSV**.

O plano inicial era utilizar as bibliotecas **DomPDF** e **Laravel Excel** no *backend*. No entanto, durante a implementação, a biblioteca `Laravel Excel` apresentou desafios significativos de compatibilidade de versão e dependências de ambiente (a extensão PHP `ext-gd`), que complicaram a configuração no ambiente de desenvolvimento com Docker.

Diante disso, a solução foi pivotada para uma arquitetura mais moderna e robusta, que cumpre os mesmos requisitos:

- **Para CSV/Excel:** A biblioteca `Laravel Excel` foi substituída no *backend* pela `league/csv`, uma alternativa mais leve e sem dependências complexas de ambiente. O *controller* gera o arquivo CSV no servidor e o envia para download.
- **Para PDF:** Foi adotada uma abordagem de geração no lado do cliente (*client-side*). O *backend* expõe um *endpoint* que retorna os dados analíticos em JSON. O *frontend*, por sua vez, utiliza as bibliotecas **jsPDF** e **jspdf-autotable** para construir o documento PDF diretamente no navegador do usuário, incluindo a logo do sistema.

Esta solução final, embora diferente da prevista inicialmente, mostrou-se mais eficiente e alinhada com as práticas adotadas anteriormente no projeto, separando as responsabilidades de dados (backend) e apresentação (frontend). A Figura 7 apresenta um exemplo do relatório gerado em formato CSV, demonstrando a estrutura tabular dos dados exportados, com colunas para Categoria, Análise, Item e Valor, prontas para manipulação em planilhas externas.



	A	B	C	D
1	Categoria	Análise	Item	Valor/Contagem
2	Demográfico	Distribuição de Gênero	Outro	40
3	Demográfico	Distribuição de Gênero	Masculino	37
4	Demográfico	Distribuição de Gênero	Feminino	30
5	Demográfico	Distribuição por Faixa Etária	0-17 anos	29
6	Demográfico	Distribuição por Faixa Etária	18-29 anos	19
7	Demográfico	Distribuição por Faixa Etária	30-59 anos	59
8	Demográfico	Distribuição por Faixa Etária	60+ anos	0
9	Demográfico	Tamanho do Núcleo Familiar	1	21
10	Demográfico	Tamanho do Núcleo Familiar	2	17
11	Demográfico	Tamanho do Núcleo Familiar	3	18
12	Demográfico	Tamanho do Núcleo Familiar	4	19
13	Demográfico	Tamanho do Núcleo Familiar	5	16
14	Demográfico	Tamanho do Núcleo Familiar	6	16
15	Socioeconômico	Nível de Escolaridade	Fundamental Completo	17
16	Socioeconômico	Nível de Escolaridade	Fundamental Incompleto	15
17	Socioeconômico	Nível de Escolaridade	Médio Incompleto	15
18	Socioeconômico	Nível de Escolaridade	Médio Completo	18
19	Socioeconômico	Nível de Escolaridade	Analfabeto	17
20	Socioeconômico	Nível de Escolaridade	Superior Incompleto	13
21	Socioeconômico	Nível de Escolaridade	Superior Completo	12
22	Socioeconômico	Situação de Trabalho	Trabalho Informal / Autônomo	33
23	Socioeconômico	Situação de Trabalho	Desempregado	28
24	Socioeconômico	Situação de Trabalho	Aposentado / Pensionista	24
25	Socioeconômico	Situação de Trabalho	Empregado (CLT)	22
26	Socioeconômico	Principais Fontes de Renda	Salário	33
27	Socioeconômico	Principais Fontes de Renda	Bolsa Família	31
28	Socioeconômico	Principais Fontes de Renda	Aposentadoria	28
29	Socioeconômico	Principais Fontes de Renda	Trabalho Informal	15
30	Operacional	Principais Encaminhamentos	Nenhum	30
31	Operacional	Principais Encaminhamentos	Posto de Saúde	27
32	Operacional	Principais Encaminhamentos	CRAS	29
33	Operacional	Principais Encaminhamentos	CREAS	21
34	Operacional	Bairros Mais Atendidos	2	2
35	Operacional	Bairros Mais Atendidos	6	2
36	Operacional	Bairros Mais Atendidos	9	2
37	Operacional	Bairros Mais Atendidos	51	1
38	Operacional	Bairros Mais Atendidos	68	1
39	Operacional	Bairros Mais Atendidos	99	2
40	Operacional	Bairros Mais Atendidos	7. Bloco A	2
41	Operacional	Bairros Mais Atendidos	3636. Fundos	1
42	Operacional	Bairros Mais Atendidos	455. F	1
43	Operacional	Bairros Mais Atendidos	4831. Bloco A	1
44				

Figura 7 – Recorte de um relatório analítico gerado pelo sistema (formato CSV).

7 CONSIDERAÇÕES FINAIS

A dificuldade dos profissionais da assistência social em acessar e interpretar dados de forma estruturada é notável, impactando diretamente a eficiência na tomada de decisões. A ausência de ferramentas adequadas para análise dificulta a identificação de padrões e tendências, tornando o planejamento de ações menos assertivo e reduzindo a capacidade de resposta a demandas emergentes. As consequências dessa falta de soluções acarretam em uma gestão fragmentada, onde decisões são frequentemente baseadas em percepções individuais e não em dados concretos. Isso compromete a alocação eficiente de recursos, prejudica o monitoramento de políticas públicas e dificulta a prestação de contas sobre os impactos das iniciativas voltadas à população em situação de vulnerabilidade.

Este projeto visou contribuir para a solução desse problema ao implementar um módulo analítico que permitiu a análise estruturada de dados da assistência social no sistema Open Social Care. A solução desenvolvida abrangeu desde a extensão da funcionalidade de formulários, com a adição de perguntas de múltipla escolha, até a criação de *templates* padrão (CCG e Albergue) e um *seeder* para dados de demonstração. O núcleo do trabalho foi a construção do módulo de *analytics*, composto por uma *view* otimizada no banco de dados, um *controller* no *backend* capaz de processar e categorizar os dados, e uma interface no *frontend* com *dashboards* interativos, filtros dinâmicos e funcionalidade de exportação para PDF e CSV.

Através destes *dashboards* e relatórios detalhados, gestores e assistentes sociais agora podem visualizar indicadores-chave categorizados (demográficos, socioeconômicos, operacionais, etc.), identificar tendências com base em filtros de período e tomar decisões com base em evidências concretas. Dessa forma, espera-se que a solução implementada contribua significativamente para a análise de dados, promovendo maior transparência, eficiência e impacto positivo nas políticas sociais.

Ao avaliar o percurso deste trabalho frente aos objetivos específicos delineados inicialmente, observa-se que as metas centrais relacionadas à implementação técnica do módulo analítico e à necessária extensão das funcionalidades de formulário foram substancialmente alcançadas. O sistema Open Social Care agora dispõe da capacidade de gerenciar perguntas de múltipla escolha, possui templates padrão que facilitam a sua adoção inicial e oferece um módulo de *analytics* funcional, capaz de processar os dados coletados e apresentá-los de forma visualmente organizada através de *dashboards* interativos com filtros dinâmicos. A funcionalidade de exportação para PDF e CSV, também um requisito inicial, foi implementada com sucesso, embora tenha exigido uma adaptação na abordagem técnica originalmente prevista: desafios com dependências das bibliotecas propostas levaram à adoção de novas bibliotecas no backend e frontend, uma solução que se mostrou robusta e alinhada às práticas modernas de desenvolvimento web. Quanto aos testes de usabilidade com os profissionais da assistência social, embora previstos nos objetivos específicos, estes não puderam ser executados devido às restrições de tempo e escopo comuns a um trabalho de conclusão de curso. A validação

realizada focou na funcionalidade e na integração técnica da solução. Assim, a condução de testes com o público-alvo é sugerida como um passo importante para trabalhos futuros, a fim de otimizar a interface e assegurar que ela atenda plenamente às demandas do dia a dia profissional.

REFERÊNCIAS

Atlassian. **Feature Branch Workflow**. 2024. https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow.

BOEIRA, S. L. dos S. **Implementação e Atualização de Frontend para o Sistema Open Social Care**. 2024. https://tcc.tsi.pro.br/uploads/academic_activity/pdf/269/GP_COINT_2024_1_SAMUEL_LEIVANS_DOS_SANTOS_BOEIRA_MONOGRAFIA.pdf. Trabalho de Conclusão de Curso (Graduação) – Universidade Tecnológica Federal do Paraná, Guarapuava, 2024.

DATASUAS. **DataSUAS: A maior plataforma de dados abertos da Assistência Social do Brasil**. 2023. Disponível em: https://datasuas.com.br/.

FEDERAL, G. **Portal de Dados Abertos - Censo SUAS**. 2023. Disponível em: https://dados.gov.br/dados/conjuntos-dados/censo-suas.

FONSECA, L. **Como o BI pode ajudar na organização e otimização do SUAS do seu município?** 2023. Acessado em: 2 abr. 2025. Disponível em: https://blog.gesuas.com.br/como-o-bi-pode-ajudar-na-organizacao-e-otimizacao-do-suas-do-seu-municipio/.

GOV.BR. **VIS DATA 3 beta. Ministério do Desenvolvimento e Assistência Social, Família e Combate à Fome.** 2023. Disponível em: https://aplicacoes.mds.gov.br/sagi/vis/.

SOUZA, C. E. de. **Atualização do Backend do Sistema Open Social Care: Migrando da Arquitetura Serverless para uma API em Laravel**. 2024. https://tcc.tsi.pro.br/uploads/academic_activity/pdf/267/GP_COINT_2024_1_CAMILA_EMANUELE_DE_SOUZA_MONOGRAFIA.pdf. Trabalho de Conclusão de Curso (Graduação) – Universidade Tecnológica Federal do Paraná, Guarapuava, 2024.

A GUIA DE EXTENSIBILIDADE DO MÓDULO ANALÍTICO

Este apêndice serve como um guia técnico para futuros desenvolvedores que desejem estender o módulo analítico do sistema Open Social Care, adicionando um novo dashboard personalizado para um template de formulário específico. A arquitetura foi projetada para ser modular, permitindo que novos painéis de análise sejam integrados de forma organizada, seguindo os passos descritos abaixo.

A.1 Visão Geral da Arquitetura

A adição de um novo dashboard envolve modificações coordenadas no *backend* (Laravel) e no *frontend* (Next.js). O fluxo de trabalho consiste em:

1. **Backend:** Criar o template de formulário com suas perguntas, e implementar a lógica de negócio que transforma as respostas brutas em insights categorizados.
2. **Frontend:** Definir a estrutura de dados esperada, criar o componente visual para o novo dashboard e integrá-lo à página principal de análises.

A.2 Passo a Passo da Implementação

A.2.1 Parte 1: Modificações no Backend (Laravel)

A.2.1.1 Passo 1.1: Criar o Template de Formulário via Seeder

O primeiro passo é garantir que o novo formulário exista no sistema. A abordagem padrão é criar uma classe `Seeder` dedicada.

1. Crie um novo arquivo de seeder utilizando o Artisan:

```
1 php artisan make:seeder NovoTemplateSeeder
2
```

Listing A.1 – Comando para criar um Seeder

2. No arquivo gerado em `database/seeders/`, deve ser implementado o método `run()` para criar o `FormTemplate`, suas perguntas (`shortQuestions` e `multipleChoiceQuestions`) e associá-lo às organizações, utilizando `updateOrCreate` para evitar duplicatas.

A.2.1.2 Passo 1.2: Registrar o Novo Seeder

Para que o novo template seja criado juntamente com os outros, deverá ser adicionado a chamada ao novo seeder no arquivo `database/seeders/DatabaseSeeder.php`.

```php
<?php
// ...
class DatabaseSeeder extends Seeder
{
    public function run(): void
    {
        $this->call([
            CcgFormTemplateSeeder::class,
            AlbergueFormTemplateSeeder::class,
            NovoTemplateSeeder::class, // Adicionar aqui
        ]);
    }
}
```

Listing A.2 – Registrar o Seeder no DatabaseSeeder.php

A.2.1.3 Passo 1.3: Implementar a Lógica de Análise no Controller

A lógica do módulo analítico reside no `AnalyticsController`. É por onde os dados brutos da `analytics_view` são transformados em insights.

1. **Crie um novo método de processamento:** Dentro do `AnalyticsController`, deve ser criado uma nova função privada, seguindo o padrão de `processCcgAnalytics` e `processAlbergueAnalytics`. Esta função receberá a coleção de respostas e deverá retornar um *array* com os dados já agrupados e contados.

```php
<?php
// Em app/Http/Controllers/Api/Manager/AnalyticsController.php

private function processNovoTemplateAnalytics(Collection
    $answers): array
{
```

```
    $analytics = [];
    $analytics['nova_categoria'] = [
        'nova_analise' => $this->getAnswerCountsForQuestion(
            $answers, 'Texto da Pergunta X'
        ),
        // ... outras análises
    ];
    return $analytics;
}

```

Listing A.3 – Exemplo de novo método de processamento no AnalyticsController

2. **Adicione o novo caso ao `match`:** No método principal `forTemplate()`, adicionar uma nova condição ao bloco `match` para que, quando o novo template for selecionado, o seu novo método de processamento seja chamado.

```
<?php
// Em app/Http/Controllers/Api/Manager/AnalyticsController.php

public function forTemplate(FormTemplate $formTemplate,
   Request $request)
{
    // ... (busca de dados) ...

    $chartData = match ($formTemplate->title) {
        'Ficha de Atendimento CCG' => $this->
   processCcgAnalytics($answers),
        'Plano de Acompanhamento - Albergue' => $this->
   processAlbergueAnalytics($answers),
        'Título do Novo Template' => $this->
   processNovoTemplateAnalytics($answers), // Adicionar aqui
        default => $this->processGenericAnalytics($answers),
    };

    // ... (retorno do JSON) ...
```

```
}

```

Listing A.4 – Adicionar novo caso ao método forTemplate()

A.2.2 Parte 2: Modificações no Frontend (Next.js)

A.2.2.1 Passo 2.1: Definir a Estrutura dos Dados (Schema Zod)

O *frontend* precisa entender a estrutura de dados que o novo processador do *backend* irá enviar.

1. Abra o arquivo de schemas de analytics (ex: `src/schemas/AnalyticsData.ts`).
2. Crie um novo *schema* Zod que descreva a estrutura do seu novo dashboard (ex: `NovoDashboardSchema`).
3. Adicione este novo *schema* à união principal `AnalyticsDataSchema`, para que a validação dos dados seja flexível.

```
// Em src/schemas/AnalyticsData.ts

const NovoDashboardSchema = z.object({
    nova_categoria: z.object({ /* ... */ }),
});

export const AnalyticsDataSchema = z.union([
  CcgAnalyticsSchema,
  AlbergueAnalyticsSchema,
  NovoDashboardSchema, // Adicionar aqui
]);

```

Listing A.5 – Adicionar novo schema à união principal

A.2.2.2 Passo 2.2: Criar o Componente do Dashboard

Criar um novo componente React dedicado a exibir os dados do seu novo dashboard. A melhor prática é seguir a estrutura dos componentes já existentes, como o `_ccg-dashboard.tsx`.

1. Crie o arquivo (ex: `_novo-dashboard.tsx`).
2. O componente deve receber a propriedade `data`, tipada com o *schema* Zod correspondente.
3. Reutilize os componentes de gráficos já existentes (`PieChart`, `BarChart`, `KpiCard`) para construir o layout visual do novo dashboard.

A.2.2.3 Passo 2.3: Integrar o Novo Dashboard na Página Principal

O último passo é "ligar"o seu novo componente à página principal de analytics.

1. Abra o arquivo da página principal (ex: `analytics/page.tsx`).
2. Importe o seu novo componente de dashboard.
3. Dentro da lógica de renderização condicional (o bloco `switch` ou `if/else`), adicione uma nova condição que verifique o título do *template* selecionado e, se corresponder, renderize o seu novo componente.

```
// Em analytics/page.tsx

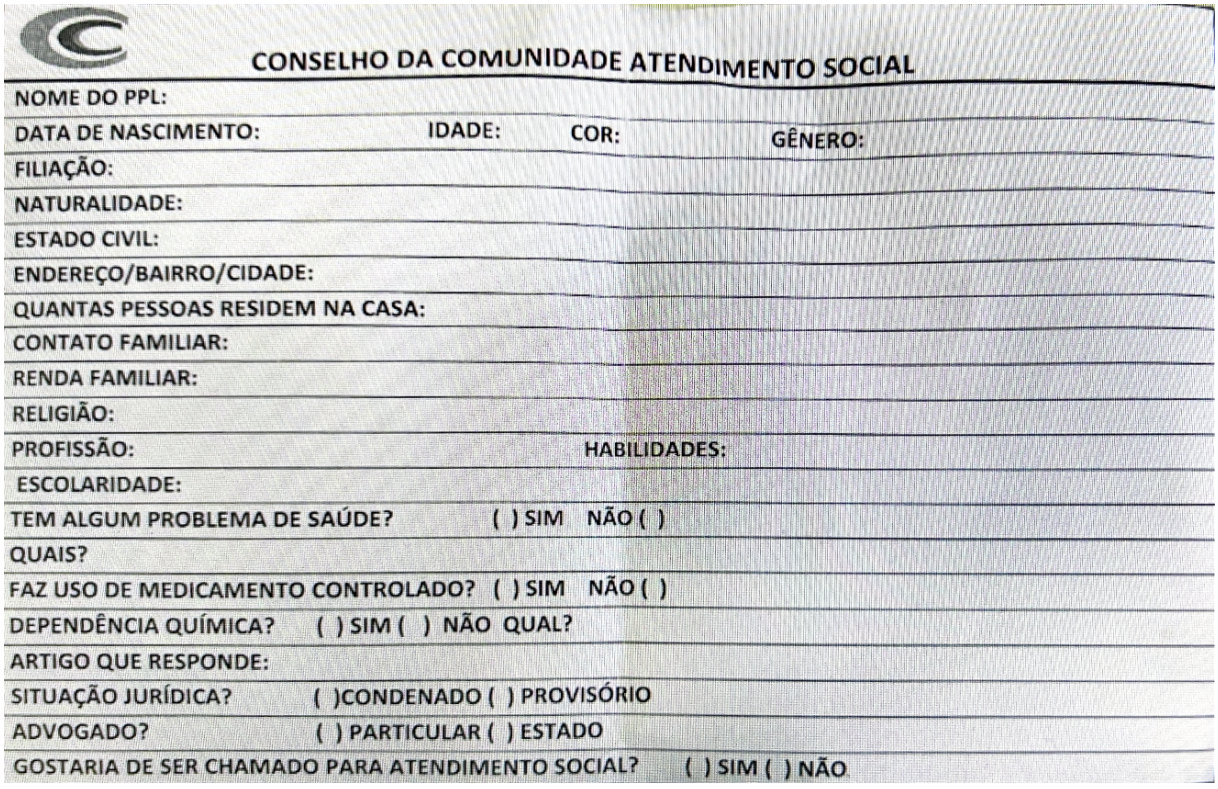
// ...
switch (selectedTemplate.title) {
    case "Ficha de Atendimento CCG":
      return <CcgDashboard data={...} />;
    case "Plano de Acompanhamento - Albergue":
      return <AlbergueDashboard data={...} />;
    case "Título do Novo Template": // Adicionar aqui
      return <NovoDashboard data={...} />;
    default:
      return <p>Dashboard genérico...</p>;
}
// ...

```

Listing A.6 – Integrar o novo componente na página principal

Seguindo estes passos, um novo dashboard personalizado pode ser integrado ao módulo analítico de forma segura e consistente, mantendo a organização e a escalabilidade do projeto.

ANEXO A – Ficha de Atendimento do Conselho da Comunidade de Guarapuava (CCG)



CONSELHO DA COMUNIDADE ATENDIMENTO SOCIAL

NOME DO PPL:

DATA DE NASCIMENTO: IDADE: COR: GÊNERO:

FILIAÇÃO:

NATURALIDADE:

ESTADO CIVIL:

ENDEREÇO/BAIRRO/CIDADE:

QUANTAS PESSOAS RESIDEM NA CASA:

CONTATO FAMILIAR:

RENDA FAMILIAR:

RELIGIÃO:

PROFISSÃO: HABILIDADES:

ESCOLARIDADE:

TEM ALGUM PROBLEMA DE SAÚDE? ( ) SIM NÃO ( )

QUAIS?

FAZ USO DE MEDICAMENTO CONTROLADO? ( ) SIM NÃO ( )

DEPENDÊNCIA QUÍMICA? ( ) SIM ( ) NÃO QUAL?

ARTIGO QUE RESPONDE:

SITUAÇÃO JURÍDICA? ( )CONDENADO ( ) PROVISÓRIO

ADVOGADO? ( ) PARTICULAR ( ) ESTADO

GOSTARIA DE SER CHAMADO PARA ATENDIMENTO SOCIAL? ( ) SIM ( ) NÃO

Figura 8 – Ficha de Atendimento utilizada pelo Conselho da Comunidade de Guarapuava (CCG).

Fonte: Arquivo fornecido pelo Conselho da Comunidade de Guarapuava.

ANEXO B – Ficha do Plano de Acompanhamento do Albergue



Sociedade de São Vicente de Paulo
Albergue Noturno
Frederico Ozanam
Conselho Central de Guarapuava

PLANO DE ACOMPANHAMENTO SERVIÇO SOCIAL

Data:

DADOS PESSOAIS		
Nome:		
Data de Nascimento:	Naturalidade:	
Filiação:		
RG:	CPF:	
Cidade de Origem:	Período que permaneceu na cidade:	
Já esteve em Guarapuava em outra(s) ocasião (ões)?	Período que ficou:	
Obs.:		
Estado Civil:	Qual é seu gênero? () Feminino () Masculino () Outro Qual?_____	
Você se considera: () Branco(a) () Negro(a) () Indígena () Pardo(a) () Amarelo(a) de origem asiática		
Possui Cad Único: () Sim () Não	Cidade do Cadastro:	
Recebe Benefício Social: () Auxílio Brasil. () Auxílio Emergencial. () Benefício Emergencial. () Baixa Renda Água/Luz. () BPC. () Não.	Recebe Benefício Previdenciário: () Aposentadoria. () Pensão por morte. () Auxílio Acidente. () Seguro Desemprego. () Auxílio Doença. () Auxílio Reclusão. () Salário Maternidade. () Não.	
Valor do Benefício: R$ _____	Valor do Benefício: R$ _____	
ESCOLARIDADE/PROFISSIONALIZAÇÃO		
Escolaridade: () Não Alfabetizado () Fundamental Completo () Fundamental Incompleto () Ensino Médio Completo () Ensino Médio Incompleto () Superior Completo () Superior Incompleto () Pós-Graduação		
Profissão:	Último Emprego:	Data de saída:
Local:	Cidade:	Experências Profissionais:
Áreas profissionais que gostaria de conhecer:		

Figura 9 – Ficha de Acompanhamento utilizada pelo Albergue.

Fonte: Arquivo fornecido pelo Albergue.