Desenvolvimento de uma *Dashboard* Integrada para Gestão de Projetos Multiplataforma[1]

Samuel Balbinot[2], Felipe Martin Sampaio[3]

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul - *Campus* Farroupilha
Farroupilha - Rio Grande do Sul, Brasil

samuelbalbi.sb@gmail.com; felipe.sampaio@farroupilha.ifrs.edu.br

Abstract. *This work presents the development of a web-based system called Integrated Dashboard, aimed at centralizing and visualizing strategic project information in corporate environments. The proposal arises from the need observed in teams that use multiple platforms — such as Jira and collaborative spreadsheets — to manage tasks and deadlines, often facing challenges related to data fragmentation, rework, and lack of consolidated visibility. The system is designed to integrate automatically with Jira via API (Application Programming Interface) and to employ ETL (Extract, Transform, Load) processes for near real-time synchronization of tasks and their statuses. Its interface includes multiple visualization modes: a metrics panel, a kanban board, an interactive calendar, and analytical charts. The development will be carried out using technologies such as React.js, FastAPI, PostgreSQL, Chart.js, and FullCalendar. It is expected that the adoption of the system will significantly improve project tracking, reduce reliance on disconnected tools, and enhance decision-making capabilities for project managers.*

Keywords: *system integration, project management, API, ETL, data visualization.*

Resumo. *Este trabalho apresenta o desenvolvimento de um sistema web denominado Dashboard Integrada, voltado à centralização e visualização de informações estratégicas de projetos em ambientes corporativos. A proposta surgiu da necessidade observada em equipes que utilizam diferentes plataformas, como Jira e planilhas colaborativas, para gerenciar tarefas e prazos, mas enfrentam desafios relacionados à fragmentação de dados, retrabalho e falta de visibilidade consolidada. O sistema foi projetado para integrar-se automaticamente ao Jira via API (Application Programming Interface) e utilizar processos de ETL (Extract, Transform, Load) para sincronização em tempo real das tarefas e seus respectivos status. A interface do sistema contempla diferentes formas de visualização: painel de métricas, quadro kanban, calendário interativo e gráficos analíticos. O desenvolvimento será realizado com tecnologias como React.js, FastAPI,*

[1] Artigo científico referente ao Trabalho de Conclusão de Curso do curso de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal do Rio Grande do Sul - *Campus* Farroupilha.
[2] Aluno matriculado no Trabalho de Conclusão de Curso.
[3] Professor orientador do Trabalho de Conclusão de Curso.

PostgreSQL, Chart.js e FullCalendar. Espera-se que, com a adoção do sistema, seja possível melhorar significativamente o acompanhamento dos projetos, reduzir a dependência de múltiplas ferramentas desconectadas e ampliar a capacidade de tomada de decisão por parte dos gestores.

Palavras-chave: *integração de sistemas, gestão de projetos, API, ETL, visualização de dados.*

1. Introdução

O avanço das tecnologias de informação e comunicação tem transformado significativamente a forma como projetos são gerenciados em ambientes corporativos. No contexto da área de Tecnologia da Informação (TI), a utilização de múltiplas plataformas para planejamento, controle de tarefas e visualização de progresso — como *Jira Software* e planilhas colaborativas — tornou-se comum em empresas que buscam escalabilidade, organização e transparência nas entregas. No entanto, essa diversidade de ferramentas, muitas vezes utilizadas de forma isolada, contribui para a fragmentação das informações e gera desafios operacionais relacionados à sincronização, à integridade dos dados e à tomada de decisão baseada em evidências.

A fim de compreender mais profundamente essas limitações, realizou-se uma análise das ferramentas disponíveis no mercado. O *Jira*, apesar de oferecer *Application Programming Interfaces* (*APIs*) que possibilitam integrações externas, diferentes modos nativos de visualização e relatórios, não contempla de forma consolidada a centralização de múltiplos projetos (tarefas, *status*, prazos) em uma única interface. Durante a análise realizada no escopo deste trabalho, não foram identificadas ferramentas externas que integrassem de maneira automatizada diversos projetos do *Jira* em um único ambiente visual, capaz de apresentar métricas, calendário e indicadores unificados. Essa ausência de soluções evidencia uma lacuna significativa para equipes distribuídas que gerenciam grandes volumes de projetos simultâneos, reforçando a necessidade de uma ferramenta personalizada que permita visualizar, comparar e acompanhar todas as iniciativas de forma centralizada.

Diante desse cenário, a construção deste *software* reside na carência crescente das organizações por soluções integradas e automatizadas, capazes de consolidar os dados dispersos e gerar *insights* confiáveis em tempo real. Segundo *Kimball e Ross* (2013), processos de integração de dados baseados em *ETL* podem reduzir em até 30% o tempo dedicado à busca por informações, otimizando o tempo de resposta das equipes. Além disso, estudos como o de *Davenport e Harris* (2007) apontam que a capacidade analítica é um diferencial competitivo para empresas que precisam decidir com agilidade. Ao eliminar a dependência de atualizações manuais entre ferramentas distintas, a proposta deste trabalho contribui diretamente para o aumento da produtividade, a redução do retrabalho e a melhoria da comunicação entre as equipes envolvidas. A hipótese central é que a adoção de uma *dashboard* unificada, com visualização dinâmica via *kanban*, calendário e gráficos de desempenho, trará ganhos significativos na visibilidade e na previsibilidade das entregas.

O presente trabalho tem como objetivo geral desenvolver um sistema *web* — uma *dashboard* integrada — que centralize, sincronize e visualize dados de projetos provenientes do *Jira Software*, por meio de integrações com *APIs* e processos de *ETL* (*Extract, Transform, Load*), permitindo uma gestão mais eficiente e estratégica das atividades.

Os objetivos específicos incluem:

- Identificar e mapear os sistemas utilizados na rotina da equipe, analisando as *APIs* disponíveis para integração;
- Projetar e implementar um processo de *ETL* capaz de extrair, transformar e carregar os dados em um banco central;
- Construir uma interface *web* responsiva e intuitiva, com painel de visão geral, *kanban*, calendário e relatórios;
- Assegurar a sincronização automática entre o *Jira* e o sistema proposto, evitando atualizações manuais e redundância;

O restante deste trabalho está organizado da seguinte forma: a seção 2 apresenta o Referencial Teórico, abordando os fundamentos conceituais sobre gestão de projetos, integração de sistemas, processos de *ETL,* visualização de dados e princípios de UX/UI (*User Experience / User Interface*). A seção 3 descreve os Procedimentos Metodológicos, contemplando as etapas planejadas para o desenvolvimento do sistema e as tecnologias adotadas. A seção 4 trata da Modelagem do Sistema, com o levantamento e a classificação dos requisitos, os casos de uso, o modelo *ER* (Entidade-Relacionamento) e os protótipos de interface. A seção 5 apresenta a implementação do sistema, descrevendo as etapas práticas de Desenvolvimento do Módulo de Visualização, a integração com o *Jira* e os desafios enfrentados. A seção 6 retoma os objetivos do trabalho, discutindo a relevância da proposta e apontando possibilidades de evolução futura. Por fim, a seção 7 reúne as Referências, devidamente formatadas e utilizadas ao longo do texto.

2. Referencial Teórico

Esta seção apresenta os principais fundamentos teóricos que sustentam o desenvolvimento da solução proposta neste trabalho. Para tanto, são discutidos conceitos relacionados à gestão de projetos em ambientes digitais colaborativos, com destaque para o uso de ferramentas como *Jira Software*. Em seguida, são abordadas as estratégias de integração de sistemas via *APIs REST* (*Representational State Transfer)*, bem como os processos de *ETL* utilizados para padronizar e consolidar dados oriundos de diferentes fontes. Além disso, são exploradas as abordagens de *BI* (*Business Intelligence*) aplicadas à visualização estratégica das informações, e os princípios de *UX/UI* que orientam o desenvolvimento de interfaces responsivas e funcionais. A construção deste referencial busca articular conceitos técnicos e estudos da literatura com o objetivo de embasar, justificar e fortalecer as decisões de projeto adotadas na solução.

2.1 Gestão de Projetos e Ferramentas Colaborativas

A gestão moderna de projetos em TI requer o uso intensivo de plataformas digitais que permitam o acompanhamento colaborativo de tarefas, prazos e entregas. Segundo *PMI* (2021), a visibilidade sobre o andamento dos projetos é um fator determinante para o sucesso, principalmente em ambientes com múltiplas equipes e demandas simultâneas. Ferramentas como *Jira*, se destacam nesse cenário por oferecerem funcionalidades específicas para o gerenciamento de tarefas e colaboração visual. No entanto, sua

utilização isolada ou com integrações limitadas gera fragmentação da informação e compromete a eficiência operacional.

Nesse contexto, o *Jira Software* — desenvolvido pela *Atlassian* — consolidou-se como uma das principais plataformas de gestão de projetos para equipes de tecnologia. Baseado em metodologias ágeis, como *Scrum* e *Kanban*, o *Jira* permite organizar e acompanhar tarefas por meio de quadros, *sprints*, fluxos personalizados e uma variedade de indicadores de desempenho. Sua arquitetura flexível e sua robusta *API REST* possibilitam integrações com sistemas externos, automação de processos e centralização de dados, características que reforçam sua adoção ampla em ambientes corporativos que demandam escalabilidade e rastreabilidade.

Estudos mostram que organizações com maior maturidade em gestão de projetos têm desempenho superior em termos de prazos, orçamentos e qualidade das entregas (*PMI*, 2021). A integração dessas ferramentas em uma única interface é uma tendência crescente, alinhada à busca por soluções que favoreçam o trabalho ágil, a rastreabilidade e a tomada de decisão baseada em dados.

2.2 Integração de Sistemas via *API*

A interconexão entre diferentes sistemas é essencial para o fluxo contínuo de informações. Conforme *Tanenbaum* e *Van Steen* (2017), sistemas distribuídos exigem arquiteturas bem planejadas para comunicação eficiente entre componentes heterogêneos. Nesse sentido, a utilização de *APIs REST* tem se consolidado como um padrão para integração leve e escalável.

Fowler (2012) reforça que arquiteturas orientadas a serviços, baseadas em *APIs*, reduzem o acoplamento entre sistemas e permitem que dados fluam em tempo real entre plataformas distintas. No contexto deste projeto, a *API* do *Jira* atua como principal ponto de coleta de dados, permitindo acesso seguro e estruturado às informações dos projetos, tarefas e responsáveis. A padronização das chamadas *HTTP* (*Hypertext Transfer Protocol*) *GET (obter), POST (enviar), PUT (atualizar) e DELETE (excluir)* e o uso de autenticação via *token* garantem interoperabilidade e segurança na comunicação entre a *dashboard* e os sistemas externos.

2.3 Processos de *ETL*

Para que dados oriundos de diferentes fontes possam ser utilizados de maneira padronizada, é necessário um processo de *ETL*. Segundo *Kimball* e *Ross* (2013), o *ETL* é um dos pilares da construção de *data warehouses*[4], sendo responsável por extrair dados brutos, tratá-los e carregá-los em uma estrutura analítica. No projeto em questão, os dados extraídos do *Jira* passam por etapas de normalização antes de serem armazenados no banco relacional do sistema.

A adoção de processos automatizados de *ETL* reduz a dependência de ações manuais e aumenta a confiabilidade dos dados. Estudos indicam que soluções baseadas em *ETL* podem reduzir em até 30% o tempo gasto na busca e preparação de

[4] *Data warehouse* é um repositório centralizado de dados, projetado para armazenar grandes volumes de informações de forma organizada, permitindo consultas analíticas e suporte à tomada de decisão.

informações (*Kimball*; *Ross*, 2013), o que tem impacto direto na produtividade das equipes de projeto.

2.4 Business Intelligence e Visualização de Dados

Com os dados integrados e estruturados, é possível aplicar estratégias de *BI* para apoiar a tomada de decisões. *Powell* (2017) define *BI* como o conjunto de tecnologias e práticas que permitem transformar dados em informações úteis por meio de análises, visualizações e relatórios. No sistema proposto, são utilizados gráficos de pizza, barras e linha do tempo para representar indicadores como tarefas concluídas, *status* dos projetos e produtividade por período.

Davenport e *Harris* (2007) destacam que empresas que adotam *BI* de forma estratégica conseguem obter vantagem competitiva, otimizando processos e antecipando problemas. A centralização da informação em *dashboards* interativas melhora a comunicação entre as áreas e reduz o retrabalho decorrente da manutenção de sistemas paralelos.

2.5 Interfaces e Experiência do Usuário (UX/UI)

A efetividade de um sistema não depende apenas de sua arquitetura técnica, mas também da experiência de uso oferecida aos seus usuários. Segundo *Wetherbe* (2000), um sistema bem projetado deve ser intuitivo, acessível e responsivo, respeitando as boas práticas de design de interação. Nesse sentido, o projeto utiliza *frameworks* modernos como *React.js* para construir interfaces reativas e bibliotecas como *Chart.js* e *FullCalendar* para visualização de dados.

O uso de protótipos desenvolvidos com ferramentas como *Figma* também contribui para validar previamente a interface com os usuários. A adoção de conceitos de *UX/UI* garante que o sistema seja funcional, agradável e compatível com múltiplos dispositivos, aspectos fundamentais para sua aceitação no ambiente corporativo.

3. Procedimentos Metodológicos

Este trabalho adota uma abordagem iterativa e incremental para o desenvolvimento da solução proposta, baseada em boas práticas da engenharia de *software* e na aplicação de metodologias centradas no usuário. A seguir, descrevem-se as etapas adotadas, bem como as tecnologias utilizadas.

3.1 Etapas de Desenvolvimento

O levantamento de requisitos e a análise do cenário atual foram realizados inicialmente com o objetivo de identificar as principais limitações enfrentadas pelos usuários no uso de plataformas como o *Jira*. Essa etapa permitiu compreender as funcionalidades prioritárias para o sistema, baseando-se em demandas reais do ambiente corporativo.

Como resultado, foi realizada a construção de diversos artefatos de modelagem do *software*, com o objetivo de documentar todos os requisitos, casos de uso e organização dos dados do sistema completo. Todo este processo está apresentado, com detalhes, na seção 4.

Na sequência, a partir deste modelagem, foi delimitado e desenvolvido um protótipo funcional do sistema a partir dos requisitos mais prioritários que foram

identificados, também conhecido como *Minimum Viable Product (MVP)* e denominado neste trabalho como Módulo de Visualização, contendo funcionalidades essenciais como importação de projetos via *API* do *Jira*, visualização em painel de calendário interativo e uma breve relação de indicadores quantitativos. A interface foi desenvolvida com foco em responsividade, visando atender diferentes dispositivos e perfis de usuários. Os detalhes de implementação deste módulo de visualização estão apresentados na seção 5.

3.2 Tecnologias Utilizadas

O sistema foi implementado como uma aplicação *web*, permitindo o acesso por navegadores modernos sem a necessidade de instalação local. A escolha por essa abordagem visa ampliar a acessibilidade e facilitar a manutenção da solução.

As principais tecnologias utilizadas são:

- *Frontend*: *React.js*, hospedado em plataformas como *Vercel* ou *Netlify*.
- *Backend* e *ETL*: *Python*, com o framework *FastAPI*, hospedado em serviços como Railway ou Render.
- Banco de Dados: *PostgreSQL*, hospedado no *Supabase*.
- Visualização de Dados: *Chart.js* e *FullCalendar*, utilizados para a renderização de gráficos e calendários.
- Integrações: *Jira REST API*, consumida com *Axios*[5], e organizada conforme o padrão de projeto *Model-View-Controller* (*MVC)*.
- Ambiente de Execução: A aplicação será acessada via *web*, com estrutura hospedada na nuvem, suportando múltiplos usuários simultaneamente.

Essas tecnologias foram escolhidas por sua ampla adoção no mercado, facilidade de integração com *APIs* e suporte a interfaces responsivas, o que se alinha às necessidades do projeto.

3.3 Validação com Empresa Parceira

O desenvolvimento tanto da modelagem completa do sistema quanto do protótipo funcional (Módulo de Visualização) foi conduzido em estreita colaboração com uma empresa parceira, que atuou como ambiente real de levantamento, validação e testes. Todas as etapas — desde a identificação das necessidades, definição dos requisitos, elaboração dos artefatos de modelagem, até a validação do *MVP* — foram realizadas com base em entrevistas, reuniões técnicas e análises práticas junto aos profissionais da organização.

Essa participação ativa da empresa garantiu que o sistema não fosse concebido de forma abstrata, mas fundamentado em problemas reais enfrentados diariamente pela equipe em seu fluxo de trabalho com o *Jira*. As decisões de escopo, priorização das funcionalidades essenciais e refinamento dos protótipos foram validadas diretamente com os usuários envolvidos, assegurando alinhamento entre o desenvolvimento técnico e o contexto operacional da organização.

---

[5] *Axios* é uma biblioteca JavaScript baseada em *Promises* que permite realizar requisições HTTP de forma simples e eficiente, amplamente utilizada em aplicações frontend com React ou Vue.js.

Dessa forma, tanto a modelagem do sistema completo quanto a construção do módulo inicial foram pautadas em demandas concretas, evitando suposições e reforçando a aplicabilidade prática da solução proposta. A colaboração contínua com a empresa parceira desempenhou papel fundamental para legitimar o projeto e fortalecer a relevância dos resultados apresentados.

4. Modelagem do Sistema

Esta seção apresenta os artefatos de modelagem elaborados para a construção do sistema *Dashboard* Integrada, com o objetivo de representar graficamente os requisitos levantados, os atores envolvidos, os relacionamentos entre entidades e o comportamento esperado da aplicação. A modelagem foi conduzida a fim de garantir clareza estrutural, rastreabilidade e alinhamento com os objetivos do sistema. Foram desenvolvidos e organizados os seguintes elementos: análise e classificação dos requisitos funcionais e não funcionais, diagrama de casos de uso, modelo entidade-relacionamento (*ER*) do banco de dados relacional e protótipos das interfaces principais. Cada artefato descrito a seguir contribui para o entendimento do funcionamento do sistema, apoiando tanto o desenvolvimento técnico quanto a validação da solução proposta.

4.1 Levantamento de requisitos

O levantamento dos requisitos foi conduzido por meio de reuniões estruturadas com os gestores e membros da equipe da empresa parceira, que participaram ativamente do processo de elicitação. Nessas reuniões, foram discutidos os fluxos atuais de trabalho, as limitações enfrentadas no uso do *Jira* e as demandas relacionadas à visualização integrada das tarefas e projetos. As informações coletadas permitiram identificar necessidades reais do ambiente corporativo, resultando na definição dos requisitos funcionais, não funcionais, de sistema e de negócio. A seguir, apresenta-se o detalhamento de cada categoria:

- Requisitos Funcionais:
    - RF01: Permitir o cadastro e autenticação de usuários.
    - RF02: Permitir a criação, edição e exclusão de projetos.
    - RF03: Permitir a visualização e gerenciamento de tarefas por projeto.
    - RF04: Sincronizar tarefas e *status* com o *Jira* via *API*.
    - RF05: Permitir a visualização de cards em formato *kanban*.
    - RF06: Permitir a visualização de tarefas em um calendário.
    - RF07: Gerar relatórios analíticos em tempo real.
    - RF08: Permitir a configuração de integrações externas via chave de *API*.
- Requisitos Não Funcionais:
    - RNF01: O sistema deve ser acessível via navegador em diferentes dispositivos (responsividade).
    - RNF02: As requisições à *API* do *Jira* devem respeitar limites de *rate limit* e autenticação via *token.*

    - RNF03: A sincronização deve ocorrer em tempo próximo do real, via agendamento automático (*ETL*).
    - RNF04: O sistema deve seguir padrões de usabilidade e acessibilidade (*UX/UI*).
- Requisitos de Sistema:
    - RS01: Banco de dados *PostgreSQL* para armazenamento persistente.
    - RS02: *Backend* construído em *Python* com *FastAPI*.
    - RS03: *Frontend* desenvolvido em *React.js*.
    - RS04: Visualizações construídas com *Chart.js* e *FullCalendar*.
    - RS05: Integração contínua com o *Jira* via *REST API*.
- Requisitos de Negócio:
    - RN01: Um usuário pode ter diferentes perfis (administrador, gerente, membro).
    - RN02: Apenas usuários com perfil de administrador podem configurar integrações e gerenciar permissões.
    - RN03: Cada tarefa deve estar vinculada a um projeto e a um responsável.
    - RN04: A pontuação do *dashboard* é calculada com base em tarefas concluídas e prazos respeitados.

4.2 Casos de Uso

Para uma melhor visualização das funcionalidades básicas, foi elaborado um Diagrama de Casos de Uso utilizando o *software Lucidchart*, apresentado na Figura 1.

O diagrama representa graficamente os principais atores do sistema *Dashboard* Integrada e suas respectivas interações com as funcionalidades disponíveis. São destacados quatro atores: Gerente de Projetos, Membro da Equipe, Administrador e Sistema Externo (*Jira* via *API*).

O Gerente de Projetos possui acesso às funcionalidades de criação e edição de projetos, acompanhamento do *status* dos projetos, visualização de relatórios analíticos e configuração de integrações com ferramentas externas. Essas ações refletem sua responsabilidade no controle macro das entregas e indicadores de desempenho.

O Membro da Equipe é responsável pela execução das tarefas atribuídas, podendo atualizá-las, reportar progresso e interagir com o quadro *kanban* e o calendário de atividades, o que facilita o acompanhamento do andamento dos projetos em tempo real.

O Administrador executa funções de gerenciamento do sistema, como configuração de *URLs* (*Uniform Resource Locator*) e chaves de *API*, administração de usuários e permissões, e auditoria de logs e integridade dos dados. Essas funcionalidades garantem a segurança, governança e estabilidade do ambiente.

Por fim, o Sistema Externo (*Jira* via *API*) atua como provedor e receptor de dados, sendo responsável por fornecer informações estruturadas sobre tarefas e projetos,

responder a atualizações de *status* e permitir a sincronização bidirecional entre o sistema interno e o *Jira*.

Este diagrama auxilia na compreensão das responsabilidades de cada ator e na delimitação clara das funcionalidades, contribuindo para o alinhamento entre os requisitos levantados e a modelagem do sistema.

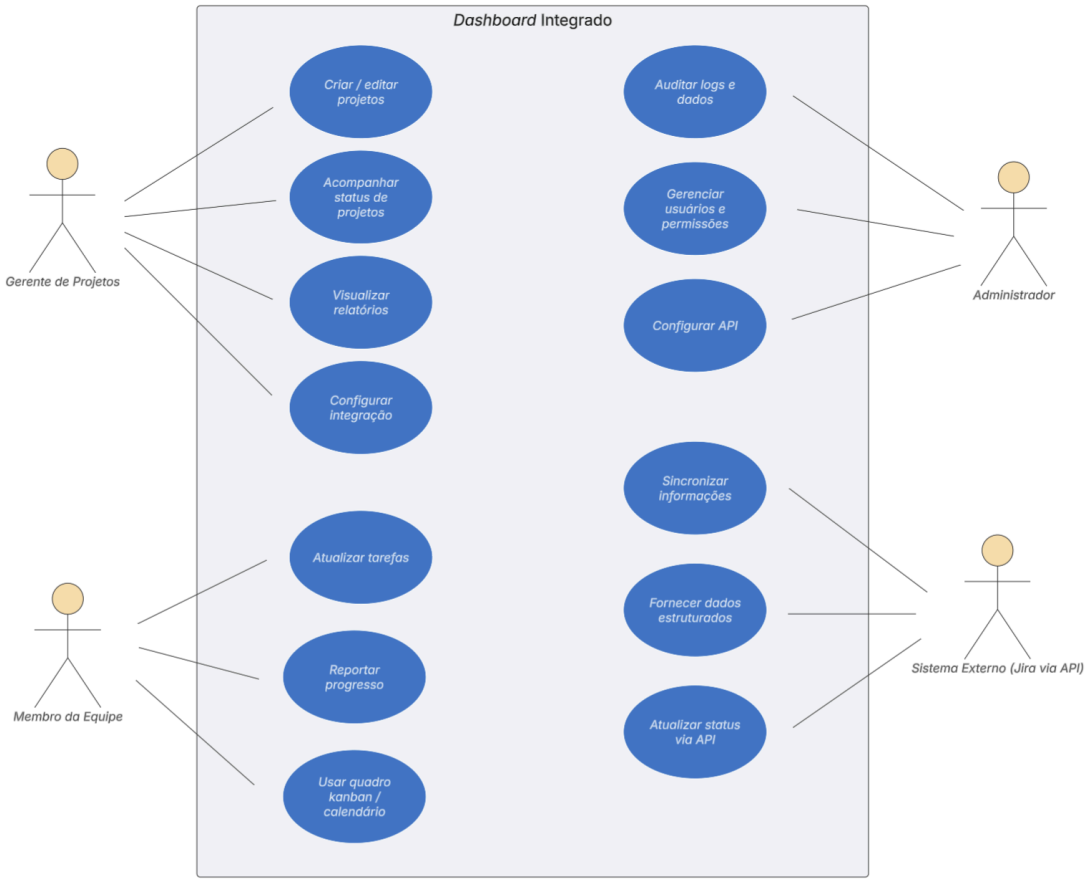

Figura 1. Diagrama de Casos de Uso do sistema proposto.
Fonte: Autoria Própria, 2025.

4.3 Diagrama Entidade-Relacionamento (ER)

A Figura 2 apresenta o modelo *ER* do sistema proposto, descrevendo a estrutura lógica do banco de dados relacional responsável pelo armazenamento das informações. O diagrama foi elaborado com base em boas práticas de modelagem de dados, assegurando integridade referencial, clareza estrutural e suporte à expansão futura do sistema.

Quatro entidades principais compõem o modelo: *Usuarios*, *Projetos*, *Tarefas* e *Integracoes*. Cada entidade está definida com seus respectivos atributos, tipos de dados e restrições, incluindo chaves primárias (*PK*), que garantem a unicidade dos registros, e chaves estrangeiras (*FK*), que representam os relacionamentos entre tabelas.

A entidade *Usuarios* armazena os dados essenciais dos usuários, como nome, e-mail (único) e papel no sistema (por exemplo, administrador, gerente ou membro). Essa entidade estabelece relacionamentos 1:N com *Projetos*, *Tarefas* e *Integracoes*,

indicando que um usuário pode ser responsável por diversos projetos, tarefas ou configurações de integração.

A entidade *Projetos* registra informações como nome, descrição, identificador externo (id_jira) e data de criação. Cada projeto é vinculado a um usuário responsável, por meio do campo responsavel_id, e pode conter múltiplas tarefas relacionadas, caracterizando uma relação 1:N com a entidade *Tarefas*.

A entidade *Tarefas* centraliza os dados operacionais do sistema, incluindo título, descrição, *status*, data de criação e atualização. Cada tarefa é associada a um projeto e a um responsável, por meio de chaves estrangeiras, refletindo a dependência direta dessas relações. Essa modelagem permite rastrear o ciclo de vida de cada tarefa e facilita a auditoria dos processos.

Já a entidade *Integracoes* armazena dados de conexão com sistemas externos, como o *Jira*, incluindo o tipo de integração, a chave de autenticação e a *URL* base. Cada integração está vinculada a um usuário, que é responsável por sua configuração e manutenção.

O modelo *ER* apresentado segue os princípios da normalização relacional, promovendo a separação de responsabilidades e facilitando a manutenção da base de dados. A representação explícita das cardinalidades (1:N) entre as entidades reforça a consistência das relações e contribui para a rastreabilidade e integridade dos dados durante a operação do sistema.

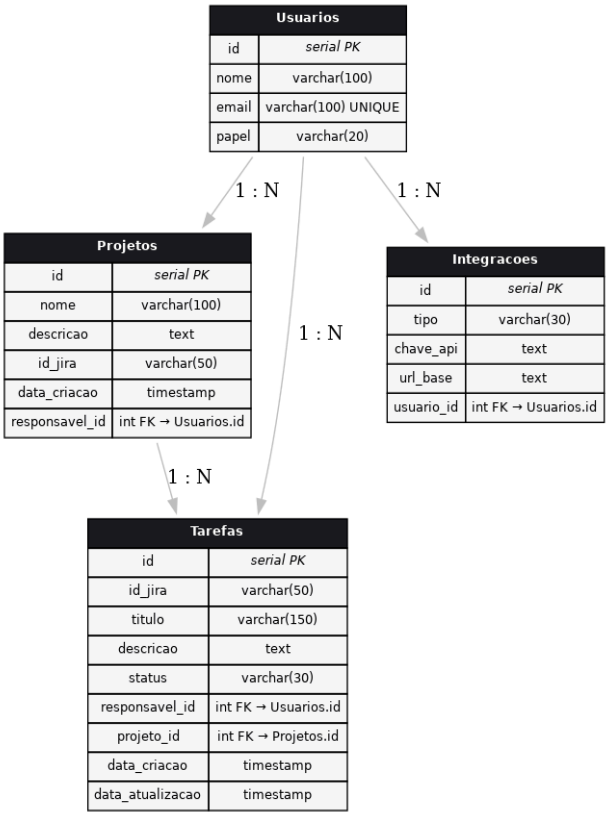

Figura 2. Diagrama Entidade-Relacionamento do sistema proposto.
Fonte: Autoria Própria, 2025.

4.4 Prototipagem das Interfaces

Com o objetivo de validar visualmente os requisitos do sistema e oferecer uma experiência intuitiva para os diferentes perfis de usuários, foram desenvolvidos protótipos de interface com base nas funcionalidades essenciais identificadas durante a fase de levantamento. Os protótipos foram projetados com foco em usabilidade, responsividade e clareza na apresentação das informações, respeitando os princípios de *UX/UI*.

O protótipo possui um *layout* moderno e funcional, organizado em seções claras para navegação e acompanhamento de projetos. A interface principal é composta por uma barra superior de navegação com as seções: *Dashboard*, *Projetos*, *Tarefas*, *Calendário*, *Relatórios* e *Integrações*, permitindo o acesso rápido aos módulos do sistema.

Na aba *Dashboard* (Figura 3), são apresentados indicadores quantitativos sobre os projetos em andamento. É possível visualizar o número total de projetos ativos, a quantidade de tarefas totais e em execução e o percentual de conclusão geral. Abaixo desses indicadores, são exibidas duas sessões: uma com a listagem dos projetos ativos e outra com a descrição das tarefas recentes, incluindo os nomes dos responsáveis. Além disso, a visualização de progresso é representada por meio de uma barra percentual, facilitando o acompanhamento visual dos dados em tempo real.

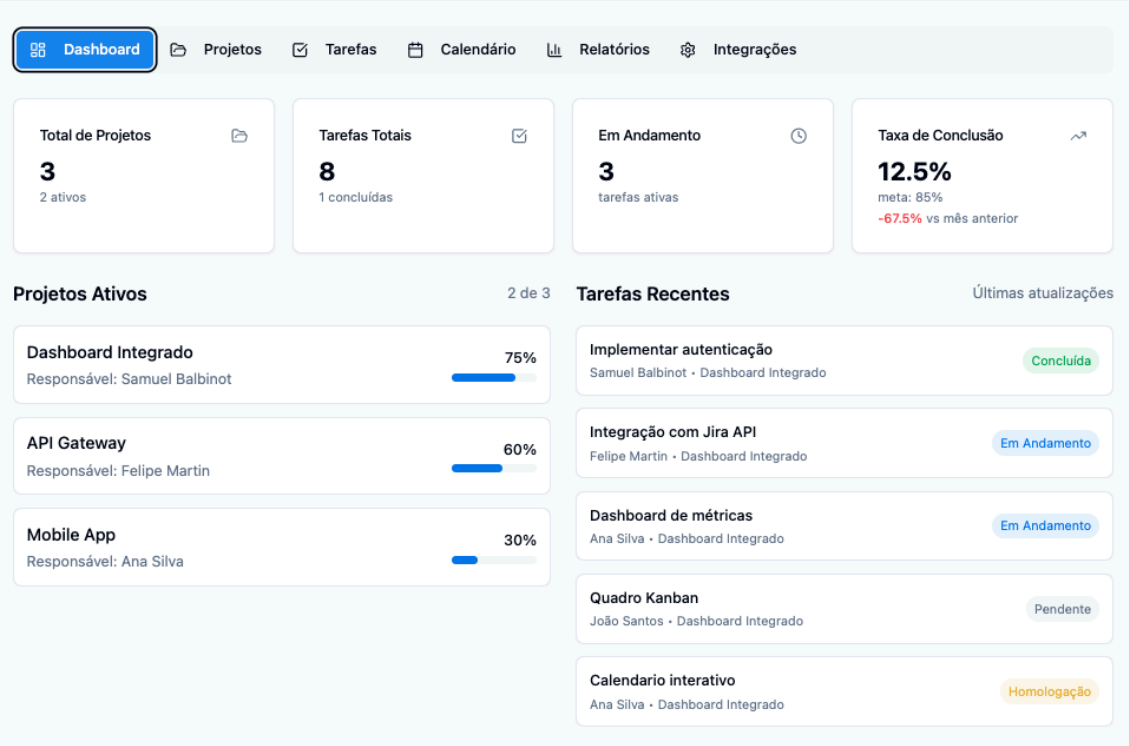

Figura 3. Aba Dashboard do sistema proposto.
Fonte: Autoria Própria, 2025.

Na aba *Projetos* (Figura 4), o sistema apresenta uma visão estruturada de todos os projetos cadastrados, permitindo ao usuário acompanhar de forma clara o progresso, a organização das entregas e os responsáveis por cada iniciativa. A interface exibe uma listagem contendo o nome do projeto, descrição, percentual de conclusão, responsável e

data de criação, além de um indicador visual de *status* que facilita a leitura rápida das informações.

Nessa mesma tela, o usuário tem acesso às ações de gestão, podendo adicionar um novo projeto, editar informações existentes ou acessar os detalhes completos para visualização aprofundada das tarefas vinculadas. A organização foi pensada para oferecer navegabilidade intuitiva, garantindo que gestores e membros da equipe identifiquem facilmente as prioridades e os estágios de cada projeto.

Essa estrutura reforça a proposta de centralização dos dados, permitindo que o acompanhamento das iniciativas seja realizado de maneira mais precisa e eficiente, reduzindo a dependência de plataformas paralelas e oferecendo uma visão consolidada do portfólio de projetos.

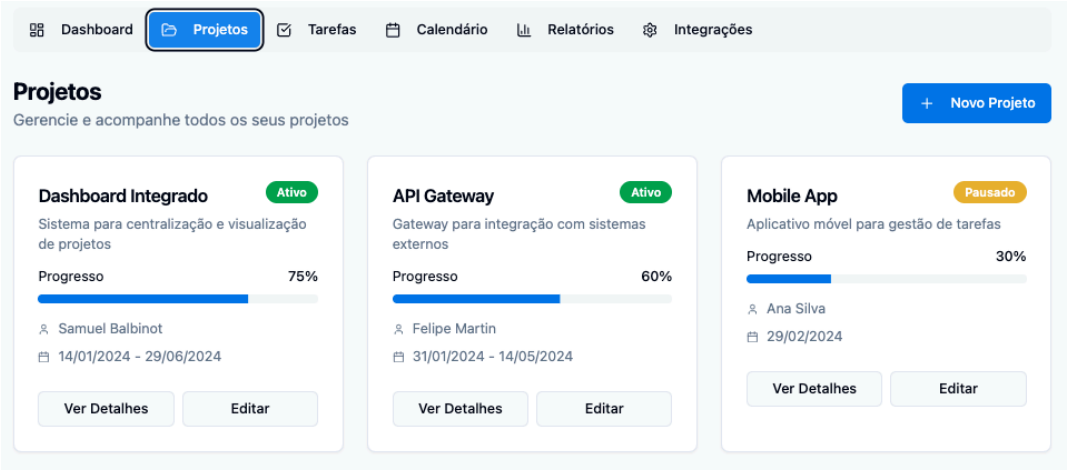

Figura 4. Aba Projetos do sistema proposto.
Fonte: Autoria Própria, 2025.

Na aba *Tarefas* (Figura 5), o sistema apresenta uma visualização no estilo *kanban*, organizada por colunas que representam os diferentes *status* de execução: Pendente, Em Andamento, Homologação e Impedimento. Cada tarefa é representada por um card contendo informações como *ID*, *status* (com indicador de cor), e responsável pela execução. Essa organização permite que gestores e membros da equipe monitorem o fluxo de atividades de forma prática e visual.

Essa estrutura permite aos usuários acompanhar o fluxo de trabalho de forma visual e segmentada, facilitando o gerenciamento e a redistribuição de atividades conforme a evolução do projeto. A interface foi desenhada para priorizar a clareza e a hierarquia das informações, com foco na agilidade de leitura e identificação de prioridades dentro de cada fase.

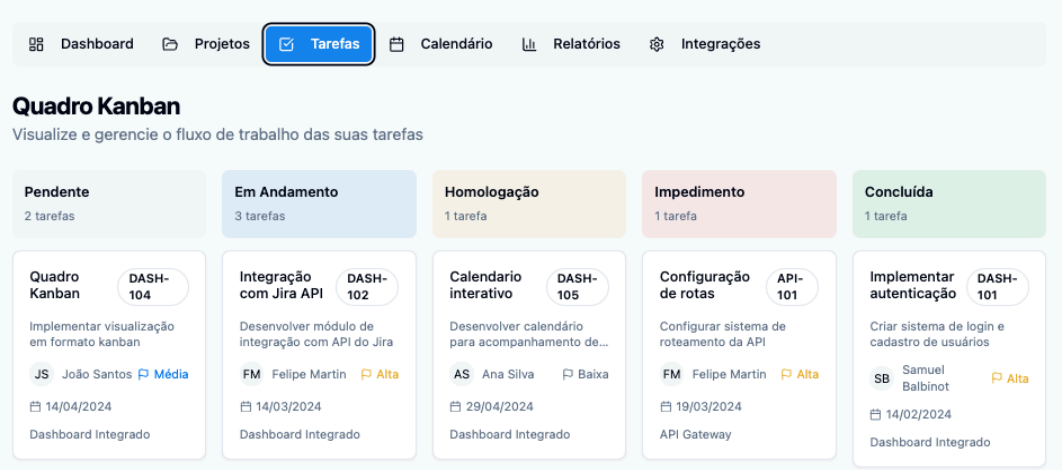

Figura 5. Aba Tarefas do sistema proposto.
Fonte: Autoria Própria, 2025.

Na aba *Calendário* (Figura 6), o sistema disponibiliza uma visualização temporal das tarefas, permitindo que usuários acompanhem prazos, entregas previstas e distribuições de atividades ao longo do mês. O calendário é exibido em formato interativo e utiliza dados sincronizados automaticamente com o *Jira*, organizando os eventos conforme suas datas de conclusão.

Cada tarefa aparece como um evento no calendário, contendo título, *status* identificado por cor e responsável. Ao selecionar um item, o usuário pode visualizar informações detalhadas, facilitando o planejamento das atividades e a identificação de períodos de maior concentração de demandas.

A ferramenta oferece diferentes modos de visualização — mensal, semanal e diária — possibilitando que gestores e equipes ajustem o nível de detalhamento conforme a necessidade. Esse recurso contribui diretamente para a previsibilidade das entregas, apoio ao planejamento e organização do fluxo de trabalho, tornando a gestão de prazos mais eficiente e visível para todos os envolvidos.

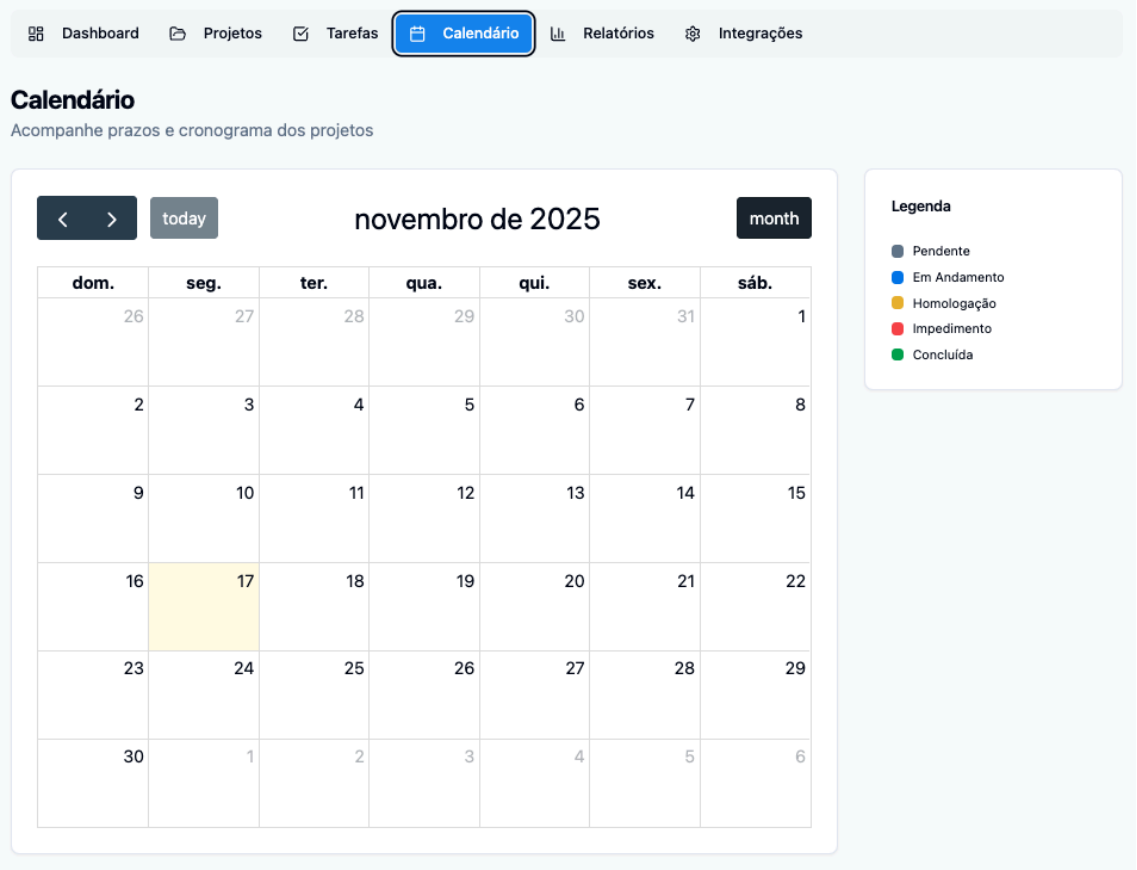

Figura 6. Aba Calendário do sistema proposto.
Fonte: Autoria Própria, 2025.

A aba *Relatórios* (Figura 7) oferece uma representação visual do desempenho dos projetos e tarefas por meio de gráficos. Estão disponíveis um gráfico de pizza que exibe a distribuição das tarefas por *status*, um gráfico de barras com a quantidade de tarefas entregues por período e uma linha do tempo com o progresso geral dos projetos.

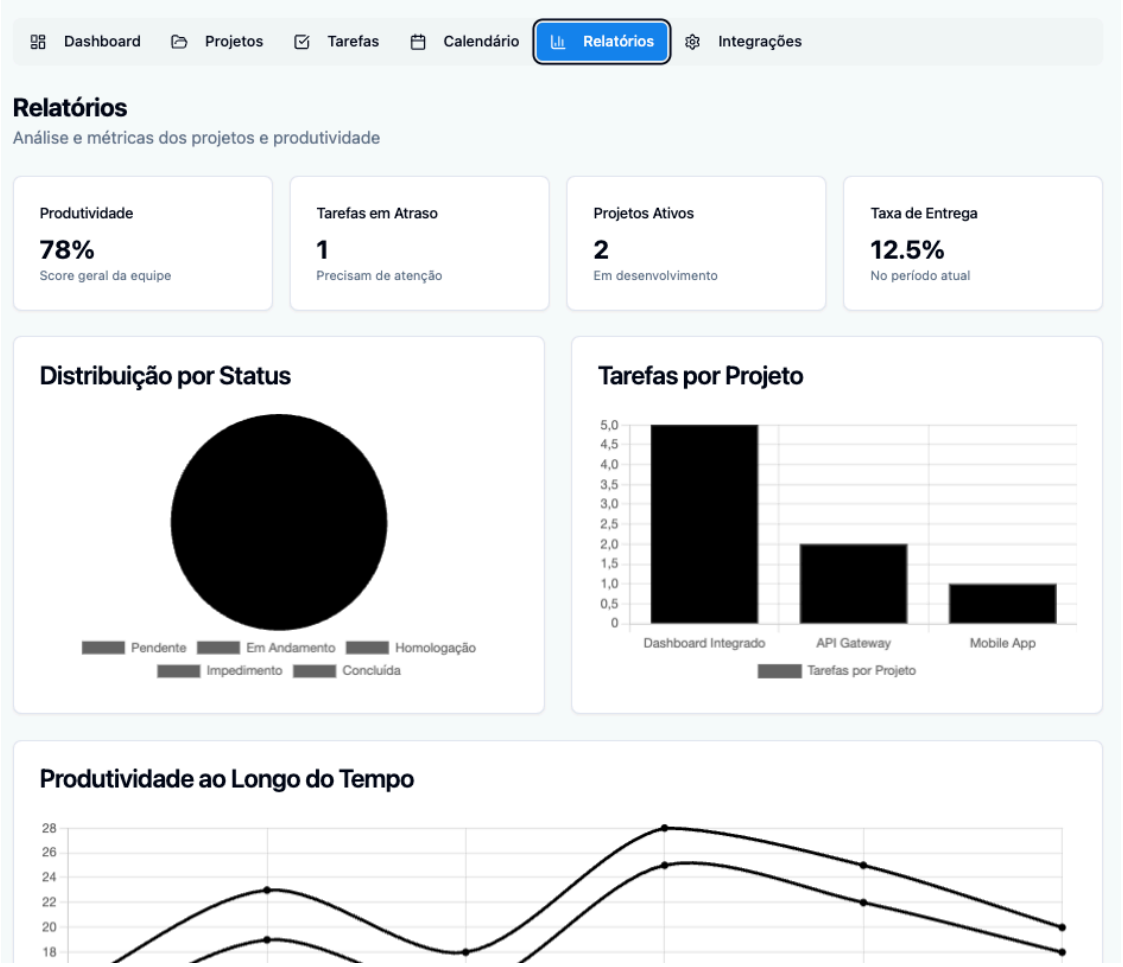

Figura 7. Aba Relatórios do sistema proposto.
Fonte: Autoria Própria, 2025.

Por fim, a aba *Integrações* (Figura 8) permite ao administrador cadastrar e gerenciar conexões com sistemas externos, como o *Jira*. A interface dispõe de campos para inserção de *URL* da *API*, botões de ação para adicionar novas integrações e uma listagem das integrações já registradas, contendo o nome da ferramenta, *status* da conexão e opções de gerenciamento.

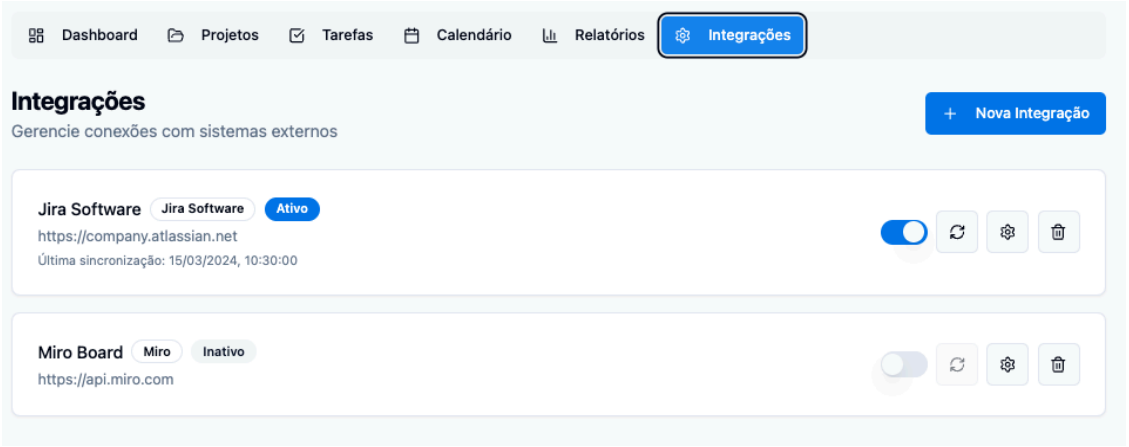

Figura 8. Aba Integrações do sistema proposto.
Fonte: Autoria Própria, 2025.

Durante o desenvolvimento dos protótipos, buscou-se manter uma paleta de cores básica, com o uso controlado de cores nos elementos gráficos para reforçar o foco nas informações e evitar distrações visuais. O uso de cores para representar o *status* das tarefas foi padronizado e aplicado de forma discreta para manter a identidade visual coesa e profissional do sistema.

As interfaces foram concebidas visando acessibilidade e adaptação a diferentes dispositivos, sendo compatíveis com resoluções de tela variadas. Essa característica é essencial para ambientes corporativos em que os usuários podem acessar o sistema por meio de notebooks, monitores estendidos ou dispositivos móveis.

5. Desenvolvimento do Módulo de Visualização

A implementação do sistema foi conduzida com foco na construção do módulo de visualização funcional, validado pela empresa onde a necessidade foi inicialmente levantada. O objetivo principal desta etapa foi transformar os artefatos definidos na modelagem — requisitos, casos de uso, modelo de dados e protótipos — em uma aplicação *web* operacional capaz de integrar-se ao *Jira*, centralizar informações e apresentar visualizações consolidadas em uma interface unificada.

A etapa de implementação utilizou uma arquitetura com *frontend* e *backend* desacoplados, comunicação via *API* e sincronização contínua dos dados por meio de extração estruturada. Essa abordagem permitiu atender aos principais requisitos definidos na modelagem, incluindo a integração com o *Jira* (RF04, RS05), a visualização e organização das tarefas por *status* (RF03, RF05, RN03), a exibição em calendário (RF06) e a apresentação de métricas consolidadas (RF07, RN04). Também foram contemplados requisitos não funcionais, como responsividade (RNF01), autenticação e limites da *API* (RNF02) e boas práticas de *UX/UI* (RNF04), além dos requisitos de sistema referentes ao uso de *React.js*, *FastAPI* e *PostgreSQL* (RS01–RS03). Os componentes desenvolvidos — integração com o *Jira*, calendário, painel de métricas e organização visual das tarefas — compõem o núcleo funcional do módulo de visualização e validam a viabilidade técnica da solução proposta.

5.1 Interface Desenvolvida

A interface disponibilizada ao usuário é o painel inicial (Figura 9), responsável por agregar os principais indicadores extraídos do *Jira*. Esse painel apresenta: Total de tarefas sincronizadas; Distribuição por *status* (pendente, em progresso, homologação e impedimentos); Indicadores visuais de progresso; Resumo das últimas tarefas atualizadas. Os dados são obtidos diretamente da *API* do *Jira* e processados no *backend* antes de serem enviados ao *frontend*.

A escolha por apresentar apenas métricas essenciais — em vez de gráficos complexos — foi intencional para reforçar a leitura rápida e imediata do estado geral dos projetos. Além disso, o painel emprega cores padronizadas para representar cada *status*, alinhando-se ao padrão visual adotado em todo o sistema. Essa consistência reforça a identificação imediata dos contextos das tarefas e reduz o esforço cognitivo do usuário.

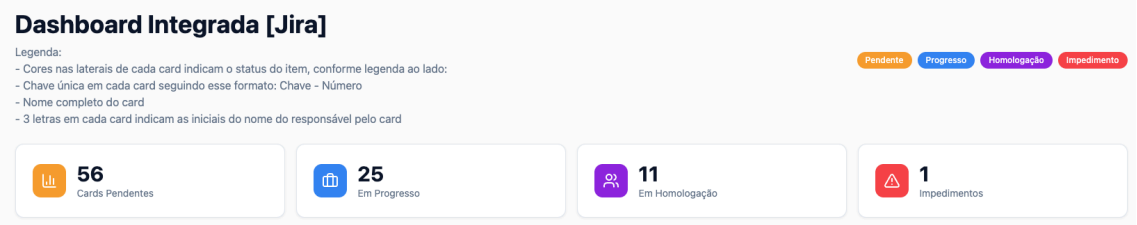

Figura 9. Painel inicial.
Fonte: Autoria Própria, 2025.

A visualização em calendário (Figura 10), principal recurso do módulo de visualização, apresentou desempenho sólido ao posicionar automaticamente as tarefas conforme suas datas de entrega, preservando a mesma categorização de *status* definida no *Jira*. A implementação foi realizada com o uso do *FullCalendar*, que permitiu criar um ambiente interativo com suporte a navegação mensal, semanal e diária. Cada tarefa é exibida no calendário por meio de um *card* compacto contendo: Código único da tarefa; Título; Iniciais do responsável; Cor identificadora do *status*.

Essa interface permite identificar padrões invisíveis, como períodos de maior concentração de demandas e sobreposição de entregas. O comportamento do calendário inclui funcionalidades como: Atualização manual das informações; Navegação entre meses; Renderização dinâmica conforme retorno da *API*; Ajustes responsivos para telas menores. Essa visualização completa de todos projetos torna mais simples a leitura temporal do volume de trabalho.

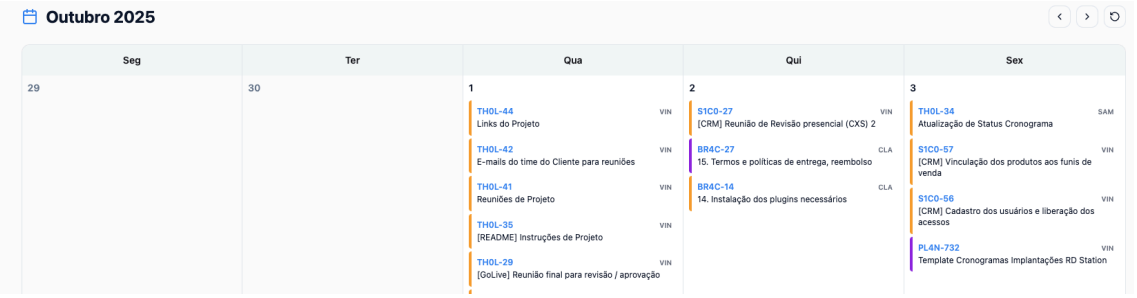

Figura 10. Calendário.
Fonte: Autoria Própria, 2025.

A empresa onde o módulo de visualização foi aplicado confirmou que o sistema elimina retrabalhos antes comuns, como a necessidade de revisar múltiplas telas no *Jira* ou compilar tarefas manualmente em planilhas. A navegação intuitiva e o *layout* organizado facilitaram a adoção por parte dos usuários, que realizaram testes práticos e forneceram *feedback* positivo quanto à clareza visual, simplicidade do fluxo e velocidade de carregamento das informações.

Com isso, constatou-se que o módulo de visualização atende plenamente aos requisitos mínimos estabelecidos para validar a ideia central da dashboard integrada, demonstrando não apenas viabilidade técnica, mas também impacto direto na rotina de gestão dos projetos.

5.2 Backend e Integração com o *Jira*

O *backend* foi desenvolvido em *Python* utilizando o *framework FastAPI* por sua alta produtividade na construção de *endpoints*, organização modular e suporte nativo à documentação dinâmica. Nessa camada foram implementadas as rotinas responsáveis pela comunicação com a *Jira REST API*, incluindo: Autenticação por chave privada; Consulta estruturada aos endpoints de *issues*; Padronização dos dados recebidos; Transformação das respostas para o modelo interno da aplicação.

Para fins do módulo de visualização, foram implementadas duas rotas principais, correspondentes a funcionalidades essenciais: Consulta de tarefas no *Jira*, retornando título, chave, responsável, *status*, data de entrega e demais atributos necessários para classificação; Entrega dos dados já tratados para o *frontend*, em formato *JSON* padronizado, permitindo o consumo direto pela interface.

O *backend* também foi responsável por aplicar regras de negócio simples, como o agrupamento das tarefas por *status* e a atualização periódica das informações.

5.3 *Frontend* e Consumo da *API*

O *frontend* foi desenvolvido em *React.js*, estruturado em componentes funcionais com manipulação interna de estado e chamadas assíncronas para o *backend*. Essa abordagem possibilitou uma interface reativa e organizada, compatível com as demandas de visualização contínua.

Entre os pontos desenvolvidos nesta camada, destacam-se: Renderização dos cards de tarefas, com cores padronizadas para cada *status*; Exibição dos indicadores quantitativos, como total de cards pendentes, em progresso, em homologação e bloqueados; Geração da agenda mensal, que distribui as tarefas no calendário conforme

as datas retornadas pela *API*; Componentes de navegação, como botões de mudança de mês e atualização.

Além disso, foram utilizados recursos de estilização *CSS*, organizados em uma estrutura modular para facilitar ajustes e revisões futuras.

5.4 Componente de Calendário

A visualização do calendário foi implementada com o auxílio da biblioteca *FullCalendar*, permitindo a representação das tarefas em uma agenda interativa. O sistema posiciona cada tarefa na data correspondente com: Chave *Jira*; Descrição resumida; Iniciais do responsável; Cor indicativa do *status*.

Essa visualização foi especialmente importante para validação prática do módulo de visualização, pois permitiu que a empresa usuária observasse a distribuição de demandas ao longo do mês, identificando períodos críticos e assimetrias de carga de trabalho.

O comportamento do calendário inclui: Navegação entre meses; Atualização manual e automática dos dados; Distribuição automática dos eventos conforme a consulta à *API*.

5.5 Organização Visual e *UX*

A implementação seguiu fielmente o protótipo previamente definido, garantindo consistência entre design planejado e produto entregue. Foram aplicados princípios de *UX/UI* observados na fase de modelagem, tais como: hierarquia visual clara; uso criterioso de cores apenas para indicar *status*; espaçamento constante entre componentes; destaque para indicadores quantitativos na parte superior; leitura simplificada dos eventos no calendário.

O resultado foi uma interface organizada, limpa e aderente ao ambiente corporativo em que seria utilizada.

5.6 Desafios da Implementação

Durante o desenvolvimento do módulo de visualização, alguns desafios se destacaram: padronização dos dados provenientes da *API* do *Jira,* devido à diversidade de formatos e campos opcionais; sincronização inicial das datas das tarefas, que exigiu conversões e ajustes para manter consistência no calendário; garantia de responsividade, uma vez que o calendário e os cards precisam se adaptar a diferentes tamanhos de tela; equilíbrio entre complexidade e escopo do módulo de visualização, mantendo o foco nas funcionalidades essenciais, conforme validado pelo orientador e pela empresa.

Esses desafios foram solucionados com ajustes incrementais na lógica de transformação dos dados e na estrutura dos componentes.

6. Conclusões

O desenvolvimento da *dashboard* integrada teve como propósito solucionar um problema evidente nas equipes que utilizam o *Jira* em seu fluxo de trabalho: a ausência de uma visualização unificada, objetiva e centralizada dos projetos e das tarefas em andamento. A fragmentação das informações — distribuídas em múltiplas telas, filtros e

visualizações — dificultava a análise do progresso, aumentava o retrabalho e comprometia a tomada de decisões em tempo hábil.

Para validar a proposta, o sistema foi testado diretamente na empresa parceira junto aos gestores responsáveis pelas áreas de tecnologia e projetos. Cada participante utilizou a ferramenta durante cinco dias consecutivos, aplicando-a em atividades reais de acompanhamento de demandas. Após esse período, foi aplicado um formulário estruturado para avaliação da experiência de uso, do desempenho e da utilidade prática da solução.

Os resultados obtidos foram amplamente positivos. As médias gerais atribuídas pelos gestores demonstram boa aceitação e percepção de valor da ferramenta: precisão dos dados (4,6), sincronização e performance (4,4), utilidade do painel principal (4,8), kanban (4,5), calendário (4,2), relatórios e gráficos (4,4), usabilidade e navegação (4,7), confiabilidade para decisões (4,6) e nota geral da ferramenta (4,7). Esses indicadores reforçam que a solução atendeu aos objetivos definidos, sendo considerada clara, responsiva e eficiente no apoio ao acompanhamento das atividades.

De modo geral, os testes evidenciaram que a consolidação das informações provenientes do Jira em uma única interface contribui para reduzir o tempo gasto na busca por dados, melhorar a visibilidade das entregas e favorecer a tomada de decisões mais rápidas e precisas. Os participantes destacaram ainda a praticidade do painel principal, a clareza dos indicadores e a relevância da sincronização automática.

Por outro lado, os testes também permitiram identificar possíveis evoluções da ferramenta, como o aprimoramento da visualização do calendário e a expansão dos relatórios analíticos, aspectos mencionados de forma pontual pelos usuários durante a avaliação.

Assim, conclui-se que a *dashboard* desenvolvida não apenas demonstrou viabilidade técnica, mas também se mostrou útil e aplicável ao contexto real de gestão de projetos. A solução cumpre o propósito de centralizar informações, reduzir dependências de diferentes telas e sistemas, e oferecer maior previsibilidade ao acompanhamento das atividades. Como trabalhos futuros, destaca-se a implementação completa do processo de *ETL*, o aprimoramento das visualizações analíticas e a possibilidade de integração bidirecional com o *Jira*.

7. Referências

KIMBALL, Ralph; ROSS, Margy. *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. Wiley, 2013.

TANENBAUM, Andrew S.; VAN STEEN, Maarten. *Distributed Systems: Principles and Paradigms*. Prentice Hall, 2017.

WETHERBE, James C. *Systems Analysis and Design: Best Practices*. Thomson Learning, 2000.

POWELL, Ron. *Business Intelligence Strategy and Big Data Analytics*. Wiley, 2017.

FOWLER, Martin. *Patterns of Enterprise Application Architecture.* Addison-Wesley, 2022.

FULLCALENDAR. *FullCalendar Documentation*. Disponível em: https://fullcalendar.io/. Acesso em: 02 de maio de 2025.

JIRA SOFTWARE API. *Rest API Documentation*. Atlassian. Disponível em: https://developer.atlassian.com/cloud/jira/platform/rest/v3/intro/. Acesso em: 10 de maio de 2025.

KANBAN UNIVERSITY. *The Official Kanban Guide – Portuguese*. Disponível em: https://kanban.university/wp-content/uploads/2021/04/The-Official-Kanban-Guide_Portuguese_A4.pdf. Acesso em: 06 de Junho de 2025.

DAVENPORT, Thomas H.; HARRIS, Jeanne G. *Competing on Analytics: The New Science of Winning*. Boston: Harvard Business Review Press, 2007.

PROJECT MANAGEMENT INSTITUTE (PMI). *Um guia do conhecimento em gerenciamento de projetos (Guia PMBOK)*. 6. ed. Newtown Square: PMI, 2021.