INSTITUTO FEDERAL GOIANO – CÂMPUS CERES BACHARELADO EM SISTEMAS DE INFORMAÇÃO
WESLEY RIBEIRO DA SILVA

API PARA PLATAFORMA DE GESTÃO PARA EXECUÇÃO DE EMENDAS PARLAMENTARES EM PROJETOS DE PREFEITURAS

CERES - GO
2025

API PARA PLATAFORMA DE GESTÃO PARA EXECUÇÃO DE EMENDAS PARLAMENTARES EM PROJETOS DE PREFEITURAS

Trabalho de curso apresentado ao curso de Sistemas de Informação do Instituto Federal Goiano - Câmpus Ceres, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação, sob orientação do Prof. Dr. Rafael Divino Ferreira Feitosa.

Ceres -GO
2025

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema Integrado de Bibliotecas do IF Goiano - SIBi

Ribeiro da Silva, Wesley
S586a API PARA PLATAFORMA DE GESTÃO PARA EXECUÇÃO DE EMENDAS PARLAMENTARES EM PROJETOS DE PREFEITURAS / Wesley Ribeiro da Silva. Ceres 2025.

61f. il.

Orientador: Prof. Dr. Rafael Divino Ferreira Feitosa.
Tcc (Bacharel) - Instituto Federal Goiano, curso de 0320203 - Bacharelado em Sistemas de Informação - Ceres (Campus

1. Software. 2. Transparência Pública. 3. Emendas Parlamentares. 4. Gestão Pública. I. Título.

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO

PARA DISPONIBILIZAR PRODUÇÕES TÉCNICO-CIENTÍFICAS NO REPOSITÓRIO INSTITUCIONAL DO IF GOIANO

Com base no disposto na Lei Federal nº 9.610, de 19 de fevereiro de 1998, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia Goiano a disponibilizar gratuitamente o documento em formato digital no Repositório Institucional do IF Goiano (RIIF Goiano), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IF Goiano.

IDENTIFICAÇÃO DA PRODUÇÃO TÉCNICO-CIENTÍFICA

- ☐ Tese (doutorado)
- ☐ Dissertação (mestrado)
- ☐ Monografia (especialização)
- ☑ TCC (graduação)
- ☐ Artigo científico
- ☐ Capítulo de livro
- ☐ Livro
- ☐ Trabalho apresentado em evento
- ☐ Produto técnico e educacional - Tipo:

Nome completo do autor:
Wesley Ribeiro da Silva

Matrícula:
2019103202030067

Título do trabalho:
API PARA PLATAFORMA DE GESTÃO PARA EXECUÇÃO DE EMENDAS PARLAMENTARES EM PROJETOS DE PREFEITURAS

RESTRIÇÕES DE ACESSO AO DOCUMENTO

Documento confidencial: ☑ Não ☐ Sim, justifique:

Informe a data que poderá ser disponibilizado no RIIF Goiano: 13 /06 /2025

O documento está sujeito a registro de patente? ☐ Sim ☑ Não

O documento pode vir a ser publicado como livro? ☐ Sim ☑ Não

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O(a) referido(a) autor(a) declara:

• Que o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;

• Que obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autoria, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia Goiano os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;

• Que cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia Goiano.

Ceres — Local
12 /06 /2025 — Data

Wesley Ribeiro da Silva
Assinatura do autor e/ou detentor dos direitos autorais

Ciente e de acordo:

Assinatura do(a) orientador(a)

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA GOIANO

ATA DE DEFESA DE TRABALHO DE CURSO

Ao(s) 5 dia(s) do mês de junho do ano de dois mil e vinte e cinco, realizou-se a defesa de Trabalho de Curso do(a) acadêmico(a) Wesley Ribeiro da Silva, do Curso de Bacharelado em Sistemas de Informação, matrícula 2019103202030067, cujo título é "API para Plataforma de Gestão para Execução de Emendas Parlamentares em Projetos de Prefeituras". A defesa iniciou-se às 19 horas e 13 minutos, finalizando-se às 20 horas e 32 minutos. A banca examinadora considerou o trabalho **APROVADO** com média 8,6 no trabalho escrito, média 9,4 no trabalho oral, apresentando assim média aritmética final de **9,0** pontos, estando o(a) estudante **APTO** para fins de conclusão do Trabalho de Curso.

Após atender às considerações da banca e respeitando o prazo disposto em calendário acadêmico, o(a) estudante deverá fazer a submissão da versão corrigida em formato digital (.pdf) no Repositório Institucional do IF Goiano – RIIF, acompanhado do Termo Ciência e Autorização Eletrônico (TCAE), devidamente assinado pelo autor e orientador.

Os integrantes da banca examinadora assinam a presente.

(Assinado Eletronicamente)
Prof. Dr. Rafael Divino Ferreira Feitosa
Orientador

(Assinado Eletronicamente)
Profa. Dra. Maryele Lazara Rezende
Membro

(Assinado Eletronicamente)
Prof. Dr. Vilson Soares de Siqueira
Membro

Documento assinado eletronicamente por:

- **Rafael Divino Ferreira Feitosa**, **PROFESSOR ENS BASICO TECN TECNOLOGICO** , em 05/06/2025 20:35:41.
- **Vilson Soares de Siqueira**, **PROFESSOR ENS BASICO TECN TECNOLOGICO** , em 05/06/2025 22:03:40.
- **Maryele Lazara Rezende**, **PROFESSOR ENS BASICO TECN TECNOLOGICO** , em 06/06/2025 08:38:47.

Este documento foi emitido pelo SUAP em 05/06/2025. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifgoiano.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 714199
Código de Autenticação: a7c0da75f1

INSTITUTO FEDERAL GOIANO
Campus Ceres
Rodovia GO-154, Km 03, SN, Zona Rural, CERES / GO, CEP 76300-000
(62) 3307-7100

AGRADECIMENTOS

Agradeço a todos que contribuíram para a minha formação acadêmica nesta Instituição querida da qual faço parte desde o Ensino Médio. Colegas de salas, professores, orientadores e gestores, sou grato por todas as experiências compartilhadas.

“A tecnologia é melhor quando reúne as pessoas."
Matt Mullenweg

RESUMO

A sociedade contemporânea vivencia transformações significativas impulsionadas pelo crescimento tecnológico, que tem alterado profundamente a forma como as informações são disseminadas e compreendidas pelos indivíduos. Nesse contexto, destaca-se o potencial da tecnologia da informação como ferramenta de acesso ao conhecimento, além de seu papel como instrumento de influência social e ideológica. Na atual conjuntura global onde a democracia política atua como conceito primordial para a formação e organização da sociedade moderna que, por sua vez, se mostra cada vez mais politizada, é possível observar e compreender a crescente demanda por maior transparência e responsabilidade por parte das instituições governamentais. Com o grande crescimento das tecnologias de informação é possível notar a importância do desenvolvimento de ferramentas tecnológicas que possibilitem o acesso claro e eficiente às informações públicas. Em um cenário onde a sociedade exige um controle mais rigoroso sobre os atos administrativos, torna-se imperativo o uso de sistemas que possam facilitar a divulgação de dados relativos a projetos públicos e a gestão de recursos financeiros. A transparência é, portanto, um dos pilares fundamentais para a promoção de uma administração pública mais democrática e acessível. Neste contexto, o presente trabalho propõe o desenvolvimento de uma API (Interface de Programação de Aplicações) voltada especificamente para gestores públicos, com o objetivo de apoiá-los na tomada de decisões estratégicas relacionadas à execução de projetos governamentais e à alocação de recursos financeiros. Além disso, a API também se destina a promover a transparência pública, permitindo que os representantes atrelados aos projetos acompanhem e fiscalizem as ações governamentais de forma eficiente. Esse sistema visa integrar dados sobre projetos públicos advindos de emendas parlamentares, oferecendo uma interface que facilite a análise e o controle por parte dos gestores. A implementação de uma ferramenta como essa não só contribui para a melhoria da gestão pública, mas também fortalece os mecanismos de participação cidadã, que são essenciais para garantir uma administração mais ética e responsável. Por meio dessa API, espera-se fomentar um ambiente de maior confiança entre o governo e a população, além de estimular a participação ativa dos cidadãos no acompanhamento e controle dos processos administrativos.

Palavras-chave: Software. Transparência Pública. Emendas Parlamentares. Gestão Pública.

ABSTRACT

Contemporary society is undergoing significant transformations driven by technological growth, which has profoundly altered the way information is disseminated and understood by individuals. In this context, the potential of information technology as a tool for accessing knowledge stands out, in addition to its role as an instrument of social and ideological influence. In the current global context, where political democracy acts as a primary concept for the formation and organization of modern society, which in turn is increasingly politicized, it is possible to observe and understand the growing demand for greater transparency and accountability on the part of government institutions. With the significant growth of information technology, it is possible to note the importance of developing technological tools that enable clear and efficient access to public information. In a scenario where society demands stricter control over administrative acts, the use of systems that can facilitate the dissemination of data related to public projects and the management of financial resources becomes imperative. Transparency is, therefore, one of the fundamental pillars for promoting a more democratic and accessible public administration. In this context, this paper proposes the development of an API (Application Programming Interface) specifically aimed at public managers, with the aim of supporting them in making strategic decisions related to the execution of government projects and the allocation of financial resources. In addition, the API is also intended to promote public transparency, allowing representatives linked to projects to monitor and oversee government actions efficiently. This system aims to integrate data on public projects arising from parliamentary amendments, offering an interface that facilitates analysis and control by managers. The implementation of a tool like this not only contributes to improving public management, but also strengthens citizen participation mechanisms, which are essential to ensuring more ethical and responsible administration. Through this API, it is expected to foster an environment of greater trust between the government and the population, in addition to encouraging active participation by citizens in monitoring and controlling administrative processes.

Keywords: Software. Public Transparency. Parliamentary Amendments. Public Management.

LISTA DE ABREVIATURAS E SIGLAS

API: Interface de Programação de Aplicações.

CRUD : Create, read, update e delete (Criação, leitura, alteração e exclusão).

SGBD: Sistema de gerenciamento de banco de dados.

SQL: Structured Query Language (Linguagem de pesquisa estruturada).

WEB: World wide web(Rede mundial de computadores).

HTML: Hypertext markup language (linguagem de marcação de hipertexto).

OOP: Oriented object programming (programação orientada a objetos).

FP: Programação funcional.

FRP: Programação reativa funcional.

HTTP: Hypertext transfer protocol (Protocolo de transferência de hipertexto).

JWT: JSON Web Tokens.

REST: Representational State Transfer

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

A crescente integração da tecnologia nas esferas pública e política tem transformado a maneira como a sociedade acessa e compreende as ações governamentais. Em um cenário marcado pela crescente demanda por transparência pública — entendida como a explicação clara das decisões e ações tomadas (GOMES, 2018) — torna-se crucial o desenvolvimento de ferramentas digitais que ampliem o acesso à informação pública, permitindo tanto à sociedade quanto aos gestores uma análise mais clara e eficiente das práticas governamentais.

Segundo Teixeira e Sabo (2016), observa-se um avanço na prática da liberdade de expressão, impulsionado por ferramentas tecnológicas, como as redes sociais e outras plataformas de comunicação digital. Ao mesmo tempo, as ferramentas de tecnologia da informação tornam-se indispensáveis quando se trata da disseminação de informação, partindo do princípio de que a internet conecta todas as sociedades nela imersas e distribui dados e informações por meio de diversas redes. Portanto, segundo os autores, as tecnologias de informação têm um papel crucial na sociedade atual, onde a internet é o principal meio de garantir a distribuição da informação.

A demanda por controle dos atos administrativos e pela adequada aplicação dos recursos públicos requer soluções que integrem dados de maneira acessível e em tempo real, em consonância com o que dispõe a Lei Complementar nº 131, de 27 de maio de 2009, conhecida como Lei da Transparência. Nesse cenário, a presente proposta tem como objetivo otimizar a execução de projetos, ao viabilizar a gestão e a comunicação detalhada das atividades desenvolvidas aos interessados.

Segundo Sacramento (2007), a transparência pode ser vista como um instrumento capaz de contribuir para a redução de atos de corrupção. Sendo assim, torna-se importante a preocupação da administração pública em oportunizar meios e ferramentas para uma maior participação cidadã e maior transparência das informações dos atos praticados por ela.

Nesse contexto, as emendas parlamentares constituem instrumentos essenciais no processo legislativo brasileiro, permitindo que deputados e senadores destinem recursos do orçamento público para atender demandas específicas de

seus estados e municípios (BRASIL, 2025). Elas são instrumentos que ajudam a assegurar à sociedade o direito à informação e ao controle social, conforme previsto na Constituição Federal de 1988 e regulamentado por legislações como a Lei de Acesso à Informação (Lei nº 12.527/2011) e a Lei da Transparência (Lei Complementar nº 131/2009).

Deve ser considerada a importância de se manter a clareza e a eficiência dos processos para execução de emendas parlamentares. Nesse sentido, as tecnologias da informação possuem potencial para fornecer ferramentas de software voltadas para garantir controle e suporte dessas demandas com maior eficiência e agilidade. Dessa forma, o presente trabalho descreve o desenvolvimento de uma API de gestão para execução de emendas parlamentares no contexto do município de Ceres situada no estado de Goiás.

O projeto tem como objetivo principal desenvolver a API de uma plataforma web voltada para o cadastro e gestão de projetos públicos financiados por emendas parlamentares, bem como para a viabilização da comunicação entre as partes envolvidas durante sua execução. Entre os objetivos específicos, destacam-se: o controle das etapas e atividades essenciais para a realização dos projetos; a otimização do fluxo de comunicação entre os atores envolvidos com o envio de notificações por e-mails; e a implementação de um sistema de gestão documental que abranja todas as etapas do projeto.

Portanto, neste trabalho abordaremos as etapas da construção da API, englobando o levantamento dos requisitos e necessidades, as etapas de engenharia de software, estruturação da base de dados e desenvolvimento back-end, com a finalidade de garantir a performance e segurança do sistema. A interface visual será desenvolvida em momento posterior, pois utilizaremos uma arquitetura baseada em APIs para a entrega final.

2. FUNDAMENTAÇÃO TEÓRICA

Segundo o Ministério da Justiça e Segurança Pública (BRASIL, 2025), uma emenda parlamentar é um instrumento que o Congresso Nacional pode utilizar na fase de apreciação legislativa para influir no processo de elaboração do orçamento anual. Tais emendas podem acrescentar, suprimir ou modificar determinados itens (rubricas) do projeto de lei orçamentária enviado pelo Executivo. Através das emendas parlamentares os deputados e senadores podem designar recursos financeiros públicos em função de compromissos políticos junto de municípios e instituições.

Emendas parlamentares são, portanto, um instrumento constitucional (Constituição da República Federativa do Brasil, 1988, art. 166, §§ 2º a 4º), que visa inserir o Congresso Nacional nas discussões acerca do planejamento do orçamento federal e descentralizar voluntariamente recursos a instâncias locais com maior proximidade das demandas sociais.

No contexto municipal, é necessário que o representante local elabore um projeto estruturado, contendo a definição clara de objetivos, orçamento detalhado e estimativas de custo. Com esse projeto em mãos, o representante deve buscar o apoio de um parlamentar, apresentando a proposta e solicitando a destinação de recursos por meio de emenda parlamentar. Uma vez aprovada, a emenda possibilita a execução do projeto idealizado, contribuindo para o desenvolvimento local e o atendimento das demandas da comunidade (MINISTÉRIO DA IGUALDADE RACIAL, 2024).

Conforme afirmado por Sodré e Alves (2010), existem argumentos que defendem a extinção das emendas parlamentares, baseados nas dificuldades de fiscalização e nos inúmeros escândalos de mau uso dos recursos provenientes dessas emendas ao longo das últimas décadas. Segundo Pires (2005), há uma análise dos incentivos para o uso das emendas parlamentares individuais no Brasil, destacando como essas emendas muitas vezes se tornam uma moeda de troca entre os poderes Executivo e Legislativo, sendo utilizadas em benefício de interesses políticos, frequentemente em detrimento do interesse público.

As ferramentas de gestão pública desempenham um papel fundamental na promoção da transparência, expondo acessivelmente informações de maneira clara e organizada. Por isso é fundamental a utilização desses recursos, pois, segundo RIBEIRO et. al. (2023), “A integração bem-sucedida da TI pode transformar a administração pública, tornando-a mais eficiente, transparente e responsiva às necessidades dos cidadãos.”.

De acordo com Batista (2023), com o avanço contínuo das tecnologias da informação, seu uso como ferramenta de inovação tem se expandido significativamente, oferecendo técnicas e recursos essenciais para otimizar o processamento, o armazenamento e a segurança de dados em diversas áreas.

Portanto, a utilização de uma API no contexto de uma plataforma de gestão de execução de emendas parlamentares apresenta o potencial para aprimorar a gestão dos processos garantindo mais agilidade e precisão, aproveitando a estrutura física já disponível no local planejado para implantação .

2.1 Aplicação WEB

Ao contrário das aplicações nativas, que são executadas diretamente nos sistemas operacionais dos dispositivos, uma aplicação web é desenvolvida com tecnologias da web, como HTML5 e JavaScript, e funciona através de navegadores. Em sua essência, uma aplicação web nada mais é do que uma página acessada pela internet, dispensando qualquer instalação adicional no dispositivo além do próprio navegador, o que facilita seu uso imediato e universal (ANACLETO, 2012).

De acordo com Mayer (2021), os aplicativos web são mais sofisticados do que uma página estática, funcionando como ecossistemas interativos. Semelhantes aos sistemas tradicionais, eles recebem dados de entrada, processam as informações e geram uma saída para os usuários. Todo esse ciclo dinâmico ocorre dentro de um navegador, onde o código é executado e interpretado em tempo real.

2.2 TypeScript

Segundo Goldberg (2022), o TypeScript é uma linguagem de programação desenvolvida pela Microsoft que se baseia no JavaScript, adicionando recursos de tipagem estática e outras funcionalidades para aumentar a segurança e a eficiência

no desenvolvimento de aplicações. Enquanto o JavaScript é uma linguagem dinâmica, o TypeScript permite declarar tipos de dados, o que possibilita a detecção de erros em tempo de compilação, antes da execução do código. Isso torna o desenvolvimento mais robusto, especialmente em projetos grandes e complexos, onde a manutenção e escalabilidade do código são essenciais.

De acordo com Richards, Nardelli e Vitek (2015), o TypeScript oferece suporte a recursos modernos do JavaScript, como classes, modificadores de acesso e módulos, e pode ser integrado facilmente a bibliotecas JavaScript existentes. O código TypeScript é compilado para JavaScript padrão, o que garante compatibilidade com qualquer ambiente que suporte JavaScript. Com uma curva de aprendizado relativamente simples para desenvolvedores familiarizados com JavaScript, o TypeScript tem se tornado uma escolha popular para frameworks como Angular, React e Vue, sendo cada vez mais utilizado em projetos de grande porte, onde a clareza e a confiabilidade do código são essenciais.

2.3 Node.js

Segundo Paulino (2018), Node.js é um interpretador de código JavaScript que opera no lado do servidor, sendo projetado para ajudar os desenvolvedores a criar aplicações altamente escaláveis, como servidores web, capazes de gerenciar dezenas de milhares de conexões simultâneas em uma única máquina física. Baseado no motor V8 JavaScript Engine (um interpretador de JavaScript open-source desenvolvido pelo Google em C++ e utilizado no navegador Chrome), o Node.js foi criado por Ryan Dahl em 2009. Seu desenvolvimento é mantido pela empresa Joyent, onde Ryan Dahl também trabalha.

2.4 GraphQl

O GraphQL é uma linguagem de consulta para APIs criada pelo Facebook em 2012 e disponibilizada como open-source em 2015. Diferente do REST, o GraphQL permite que o cliente especifique exatamente quais dados deseja receber, evitando o carregamento excessivo (overfetching) ou insuficiente (underfetching) de informações, o que torna as requisições mais eficientes e flexíveis.

Segundo a documentação oficial, “GraphQL fornece uma descrição completa e compreensível dos dados na sua API, dá aos clientes o poder de solicitar exatamente o que precisam e nada mais” (FACEBOOK, 2025, tradução nossa). Essa abordagem reduz a quantidade de dados trafegados na rede e permite que múltiplos recursos sejam acessados em uma única requisição, otimizando a performance e a experiência do usuário final.

2.5 NestJS

Conforme a documentação do framework (NESTJS, 2025), Nest (NestJS) é um framework para construir aplicações eficientes e escaláveis do lado servidor em Node.js. Ele utiliza JavaScript progressivo, é construído com e oferece suporte total a TypeScript (mas ainda permite que os desenvolvedores programem em JavaScript puro) e combina elementos de OOP (Programação Orientada a Objetos), FP (Programação Funcional) e FRP (Programação Reativa Funcional). Nos bastidores, o Nest utiliza frameworks robustos de servidor HTTP como o Express (o padrão) e, opcionalmente, pode ser configurado para usar o Fastify também!

Ainda segundo a documentação, o Nest fornece um nível de abstração acima desses frameworks comuns do Node.js (Express/Fastify), mas também expõe suas APIs diretamente ao desenvolvedor. Isso dá aos desenvolvedores a liberdade de usar a infinidade de módulos de terceiros disponíveis para a plataforma subjacente.

2.6 MySQL

Conforme afirmado por Gonzaga e Birckan (2000), MySQL é um sistema de gerenciamento de banco de dados relacional, projetado para suportar múltiplos usuários e conexões simultâneas. Baseado na linguagem SQL, que é a mais amplamente utilizada para gerenciamento de dados, o MySQL segue a arquitetura cliente-servidor. Ele é composto por um servidor central que gerencia os dados, além de diversos programas cliente e bibliotecas que permitem o acesso e manipulação desses dados.

Ainda conforme Gonzaga e Birckan, o servidor MySQL é reconhecido pela sua rapidez e flexibilidade, sendo capaz de armazenar diversos tipos de dados, como logs e imagens. Suas principais qualidades incluem alta performance, robustez e facilidade de uso. Originalmente, o MySQL foi desenvolvido pela equipe

da T.c.X. DataKonsultAB (empresa responsável pela criação do MySQL) para atender à necessidade de um sistema de banco de dados SQL que fosse capaz de processar grandes volumes de dados com uma velocidade muito superior à oferecida pelos bancos de dados comerciais da época. Desde 1996, a equipe da T.c.X. tem utilizado o MySQL em um ambiente com mais de 40 bancos de dados, contendo 10.000 tabelas, das quais mais de 500 superam os 7 milhões de registros cada. No total, isso representa cerca de 100 GB de dados.

2.6.1 SQL

O autor Prescott (2015), afirma que a linguagem SQL (Structured Query Language). A linguagem SQL é uma linguagem de consulta para interagir com banco de dados relacionais. A linguagem de consulta para interagir com banco de dados relacionais.

Ainda segundo Prescott (2015), a linguagem SQL é usada para realizar funções como inserir dados em um banco de dados, recuperar dados, atualizar dados, excluir dados e outras ações similares.

Segundo Price (2009), a linguagem SQL tem suas raízes no trabalho inovador do Dr. E.F. Codd. Sua primeira versão foi criada pela IBM nos anos 1970, como parte de um projeto de pesquisa chamado System R. Em 1979, a empresa Relational Software Inc. (atualmente Oracle Corporation) lançou a primeira versão comercial da SQL. Hoje, a linguagem SQL é amplamente padronizada e reconhecida pelo American National Standards Institute.

3. MATERIAIS E MÉTODOS

A idealização do sistema surgiu a partir da demanda existente na Prefeitura Municipal de Ceres, mais relacionado à Secretaria de Planejamento. Atualmente todo o registro dos projetos desenvolvidos a partir de emendas parlamentares é realizado manualmente, sendo posteriormente transferido para planilhas, o que compromete a organização, a agilidade e a rastreabilidade das informações.

Essa forma de trabalho sem o suporte de um software traz grande dificuldade na organização dos dados, o que resulta numa dificuldade ainda maior na organização para apresentação de resultados. Nesse aspecto, um software de gerenciamento e apoio na comunicação das atividades dos projetos surge como um meio de otimizar o trabalho e possibilitar uma análise macro do andamento dos projetos, além de facilitar a apresentação de todo o processo realizado nos projetos.

Inicialmente, foram realizadas reuniões de alinhamento e entendimento do problema e levantamento dos requisitos funcionais e não funcionais do sistema. De forma concomitante, foi feito o levantamento das principais necessidades dos usuários e das partes interessadas e a coleta de informações sobre funcionalidades desejadas, definindo assim o objetivo geral no qual o software se embasaria.

O desenvolvimento do software final foi dividido entre front-end e back-end. Esse projeto aborda o back-end que é a parte da aplicação responsável pelo funcionamento interno dos sistemas, onde são processados dados, realizadas integrações com bancos de dados e definidas as regras de negócio.

As tecnologias utilizadas para o desenvolvimento da API back-end, que é o contexto desse projeto, foram definidas com uma análise de mercado e necessidade do software. Para o desenvolvimento, foi utilizada a linguagem TypeScript, com a plataforma Node.js e o framework NestJS para a estruturação e organização do código. A linguagem de consulta utilizada para a API foi GraphQL. Como sistema gerenciador de banco de dados (SGBD), foi escolhido o MySQL, utilizando a linguagem SQL para a manipulação e consulta dos dados.

O escopo deste projeto abrange o desenvolvimento de uma API focada na disponibilização de dados, integrando-se ao front-end que será desenvolvido em

outro projeto. Esse front-end consumirá os dados fornecidos pela API, implementando a interface visual e completando todas as partes do sistema, assim viabilizando sua implementação final.

4. RESULTADOS

Nesta seção serão abordados os resultados obtidos ao fim do desenvolvimento, destacando o diagrama de casos de uso, o padrão de desenvolvimento utilizado para o desenvolvimento da aplicação e os requisitos funcionais e não funcionais.

4.1 REQUISITOS FUNCIONAIS E NÃO FUNCIONAIS

Conforme Filho e Martinez (2006), os requisitos funcionais são condições imprescindíveis para alcançar um determinado objetivo, ou para cumprir um determinado objetivo. Estudos sobre a aquisição (elicitação) de requisitos são relevantes por dois motivos: primeiro, sob a ótica da engenharia de software, a elicitação de requisitos é provavelmente o elemento mais essencial do processo de criação de software.

Tabela 1: Requisito funcional para autenticação de usuário.

Identificação do Requisito	RF01
Nome do Requisito	Autenticação do Usuário
Especificação do Requisito	
● O sistema deve permitir que os usuários façam login fornecendo um endereço de e-mail e senha válidos.	
Exceção: Caso o usuário informe algum dos campos incorretamente o endpoint deve apresentar uma mensagem de erro informando que os dados inseridos estão incorretos.	

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 1 é responsável por garantir o acesso seguro ao sistema por meio de autenticação. O usuário deve inserir um e-mail e senha válidos para realizar o login. Em casos de erro nos dados fornecidos, o sistema retorna uma mensagem informativa alertando sobre o problema.

Tabela 2: Requisito funcional para cadastro de projetos.

Identificação do Requisito	RF02
Nome do Requisito	Cadastro de Projeto
Especificação do Requisito	
● O sistema deve permitir o cadastro de novos projetos pelo administrador quando todos os campos forem preenchidos.	
Exceção: Caso o administrador informe algum campo de forma incorreta o endpoint deve devolver uma mensagem indicando qual campo deve ser corrigido.	

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 2 assegura que o administrador possa cadastrar novos projetos no sistema, desde que todos os campos obrigatórios estejam corretamente preenchidos. Caso contrário, o sistema indica qual campo necessita correção, promovendo a integridade dos dados cadastrados.

Tabela 3: Requisito funcional para edição de projetos

Identificação do Requisito	RF03
Nome do Requisito	Edição de Projeto
Especificação do Requisito	
● O sistema deve permitir que o administrador edite as informações dos projetos.	
Exceção: Caso o administrador informe algum campo de forma incorreta o endpoint deve devolver uma mensagem indicando qual campo deve ser corrigido.	

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 3 permite que o administrador atualize informações de projetos previamente cadastrados. Assim como no cadastro, erros nos dados informados resultam em mensagens específicas de correção, garantindo consistência e flexibilidade na manutenção das informações.

Tabela 4: Requisito funcional para exclusão de projetos.

Identificação do Requisito	RF04
Nome do Requisito	Exclusão de Projeto

Especificação do Requisito
● O sistema deve permitir que o administrador exclua projetos.
Exceção: Caso ocorra algum problema e o sistema não consiga excluir o projeto deve ser retornada uma mensagem de erro informando o usuário.

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 4 detalha que o sistema deve permitir a exclusão de projetos por parte do administrador. Caso a exclusão falhe por algum motivo técnico ou de regra de negócio, o sistema exibe uma mensagem de erro para que o usuário compreenda a falha ocorrida.

Tabela 5: Requisito funcional para criação de etapas.

Identificação do Requisito	RF05
Nome do Requisito	Criação de Etapa
Especificação do Requisito	
● O sistema deve permitir que o administrador crie etapas dentro dos projetos.	

Exceção: Caso alguma das informações inseridas não estejam de acordo com as informações do projeto, o sistema deve retornar uma mensagem de erro informando quais campos devem ser corrigidos.

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 5 define que o administrador pode criar etapas dentro dos projetos, respeitando as regras e informações do projeto, como exemplo as datas de início e fim. Caso existam inconsistências nos dados inseridos, o sistema retorna mensagens de erro apontando o problema ocorrido.

Tabela 6: Requisito funcional para edição de etapas.

Identificação do Requisito	RF06
Nome do Requisito	Edição de Etapa
Especificação do Requisito	
● O sistema deve permitir que o administrador faça edições nas etapas.	
Exceção: Caso alguma das informações inseridas não estejam de acordo com as regras de negócio das etapas, o sistema deve retornar uma mensagem de erro informando quais campos devem ser corrigidos.	

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 6 determina que o administrador pode editar as informações de uma etapa existente. O sistema deve validar os dados

fornecidos com base nas regras de negócio definidas, e exibir mensagens de erro sempre que alguma inconsistência for detectada.

Tabela 7: Requisito funcional para exclusão de etapas.

Identificação do Requisito	RF07
Nome do Requisito	Exclusão de Etapa
Especificação do Requisito	
• O sistema deve permitir que o administrador exclua etapas.	
Exceção: A etapa não pode ser excluída se houver atividades vinculadas a ela.	

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 7 define que o administrador pode excluir etapas existentes, desde que não haja atividades vinculadas a ela. Essa restrição visa preservar a integridade dos dados e evitar perdas de registros importantes relacionados às atividades do projeto.

Tabela 8: Requisito funcional para criação de atividade.

Identificação do Requisito	RF08
Nome do Requisito	Criação de Atividade
Especificação do Requisito	
• O sistema deve permitir que o administrador crie atividades dentro das etapas.	
Exceção: A atividade não pode ser criada se a etapa estiver concluída.	

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 8 especifica que o administrador poderá criar atividades dentro de uma etapa. No entanto, a criação estará condicionada ao status da etapa, que não pode estar finalizada, garantindo coerência no fluxo de execução do projeto.

Tabela 9: Requisito funcional para edição de atividades.

Identificação do Requisito	RF09
Nome do Requisito	Edição de Atividade
Especificação do Requisito	

● O sistema deve permitir que o administrador faça edições nas atividades.
Exceção: A atividade não pode ser editada se já estiver concluída

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 9 permite que o administrador edite as informações de atividades cadastradas, desde que estas não estejam concluídas. Essa limitação assegura que apenas atividades em andamento possam ser modificadas.

Tabela 10: Requisito funcional para exclusão de atividades.

Identificação do Requisito	RF10
Nome do Requisito	Exclusão de Atividade
Especificação do Requisito	
● O sistema deve permitir que o administrador exclua atividades.	
Exceção: Usuários sem perfil de administrador não poderão realizar a exclusão.	

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 10 estabelece que a exclusão de atividades será permitida somente para usuários com perfil de administrador. Tal restrição reforça o controle sobre operações críticas do sistema.

Tabela 11: Requisito funcional para visualização de calendário de atividades.

Identificação do Requisito	RF11
Nome do Requisito	Visualização de calendário com últimas atividades
Especificação do Requisito	
● O sistema deve exibir as atividades que tem a previsão de término dos próximos 7 dias no formato de calendário.	
Exceção: Caso as informações não sejam retornadas, o endpoint deve devolver uma mensagem de erro indicando a causa do erro.	

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 11 prevê a exibição das atividades previstas para conclusão nos próximos sete dias, a visualização se dará por meio de um calendário. Caso ocorra algum erro durante a consulta dos dados, o sistema deve apresentar uma mensagem informando o problema.

Tabela 12: Requisito funcional para registro de usuários.

Identificação do Requisito	RF12
Nome do Requisito	Registro de Usuários
Especificação do Requisito	
• O sistema deve permitir que o administrador registre usuários visualizadores.	
Exceção: Caso as informações não sejam retornadas, o endpoint deve devolver uma mensagem de erro indicando qual a classe do erro.	

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 12 permite que o administrador registre novos usuários com perfil apenas de visualização. Em caso de erro no processo de cadastro, o sistema deve exibir uma mensagem que indique a natureza do problema.

Tabela 13: Requisito funcional para clonagem de projeto.

Identificação do Requisito	RF13
Nome do Requisito	Clonagem de Projeto

Especificação do Requisito
● O sistema deve permitir que o administrador clone projetos já existentes.
Exceção: Caso ocorra algum problema na clonagem deve ser retornado um erro com a mensagem informativa.

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 13 permite que o administrador possa clonar um projeto existente, replicando sua estrutura para facilitar a criação de novos projetos semelhantes. Caso a clonagem não seja bem-sucedida, o sistema deverá informar o erro ocorrido.

Tabela 14: Requisito funcional para listagem de registros de atividades.

Identificação do Requisito	RF14
Nome do Requisito	Listagem de Registros de Atividades
Especificação do Requisito	
● O sistema deve permitir que o administrador visualize uma lista com as informações dos registros das atividades e as modificações já feitas.	

Exceção: Caso ocorra algum problema na listagem deve ser retornado um erro com a mensagem informativa.

Fonte: Elaborado pelo autor.

O requisito funcional descrito na Tabela 14 prevê que o administrador possa visualizar os registros de alterações feitas nas atividades. Se ocorrer alguma falha na obtenção dos dados, o sistema deverá exibir uma mensagem informativa ao usuário.

Tabela 15: Requisito não funcional para validação dos dados.

Identificação do Requisito	RNF01
Nome do Requisito	Validação dos dados inseridos
Especificação do Requisito	
● O sistema deve validar os dados inseridos nos formulários (ex.: campos obrigatórios, formato de e-mail, datas válidas) para evitar inconsistências no banco de dados.	

Fonte: Elaborado pelo autor.

O requisito não funcional descrito na Tabela 15 garante que todos os dados inseridos nos formulários sejam validados antes de serem salvos, assegurando o

preenchimento correto dos campos obrigatórios e evitando inconsistências no banco de dados.

Tabela 16: Requisito não funcional para autenticação dos usuários.

Identificação do Requisito	RNF02
Nome do Requisito	Autenticação dos usuários
Especificação do Requisito	
• O acesso ao sistema deve ser protegido por autenticação baseada em tokens JWT, com controle de acesso por perfil (ex.: administrador e usuário visualizador).	

Fonte: Elaborado pelo autor.

O requisito não funcional descrito na Tabela 16 trata da autenticação dos usuários por meio de tokens JWT e da definição de permissões conforme o perfil de acesso. Isso garante maior segurança e controle sobre os dados acessados dentro da aplicação.

Tabela 17: Requisito não funcional para disponibilidade dos arquivos.

Identificação do Requisito	RNF03
Nome do Requisito	Disponibilidade dos arquivos

Especificação do Requisito
● Os arquivos anexados às atividades devem ser armazenados em local seguro, com URLs protegidas contra acesso não autorizado.

Fonte: Elaborado pelo autor.

O requisito não funcional descrito na Tabela 17 estabelece que os arquivos anexados às atividades devem ser armazenados em local seguro, com URLs protegidas, de forma que apenas usuários autorizados tenham acesso a esses documentos.

Tabela 18: Requisito não funcional para navegação intuitiva.

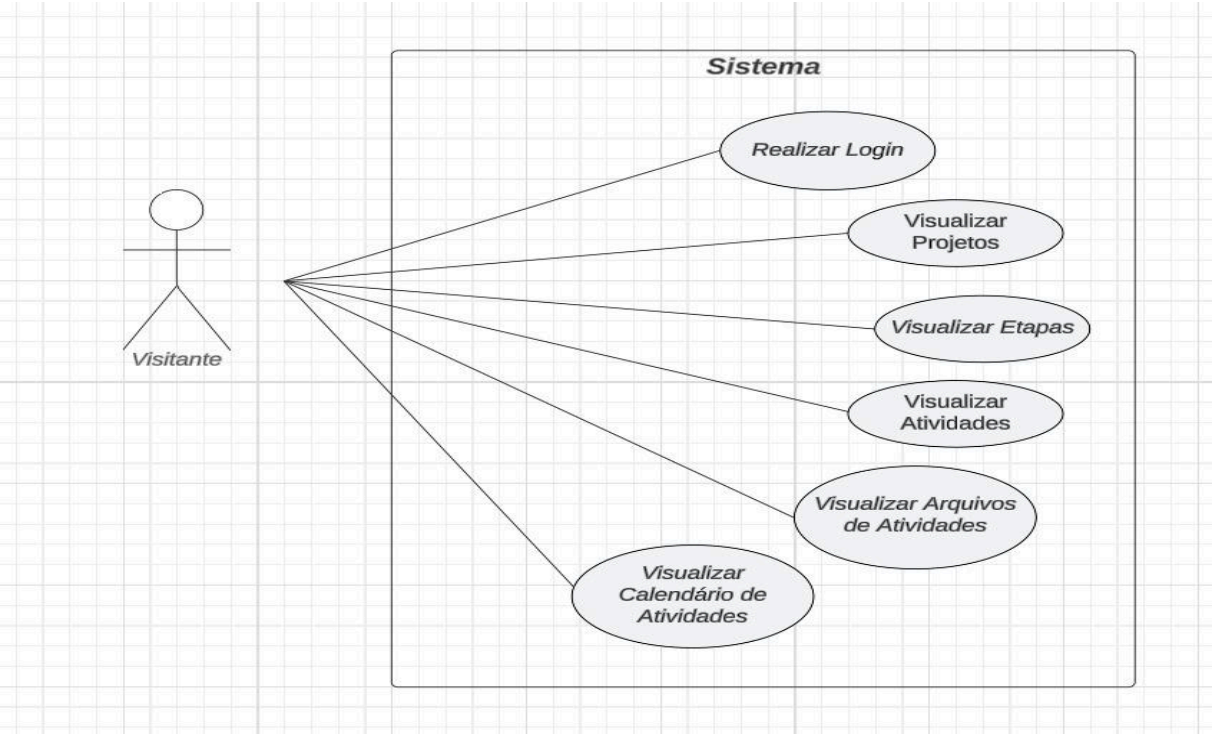
Identificação do Requisito	RNF04
Nome do Requisito	Navegação intuitiva
Especificação do Requisito	
● O sistema deve adotar uma navegação simples e intuitiva, com fluxos claros para cadastro de projetos, etapas, atividades, usuários e anexos.	

Fonte: Elaborado pelo autor.

O requisito não funcional descrito na Tabela 18 reforça a necessidade de uma navegação clara e intuitiva no sistema, facilitando o uso por todos os perfis de usuários, principalmente durante os processos de cadastro e gerenciamento das informações.

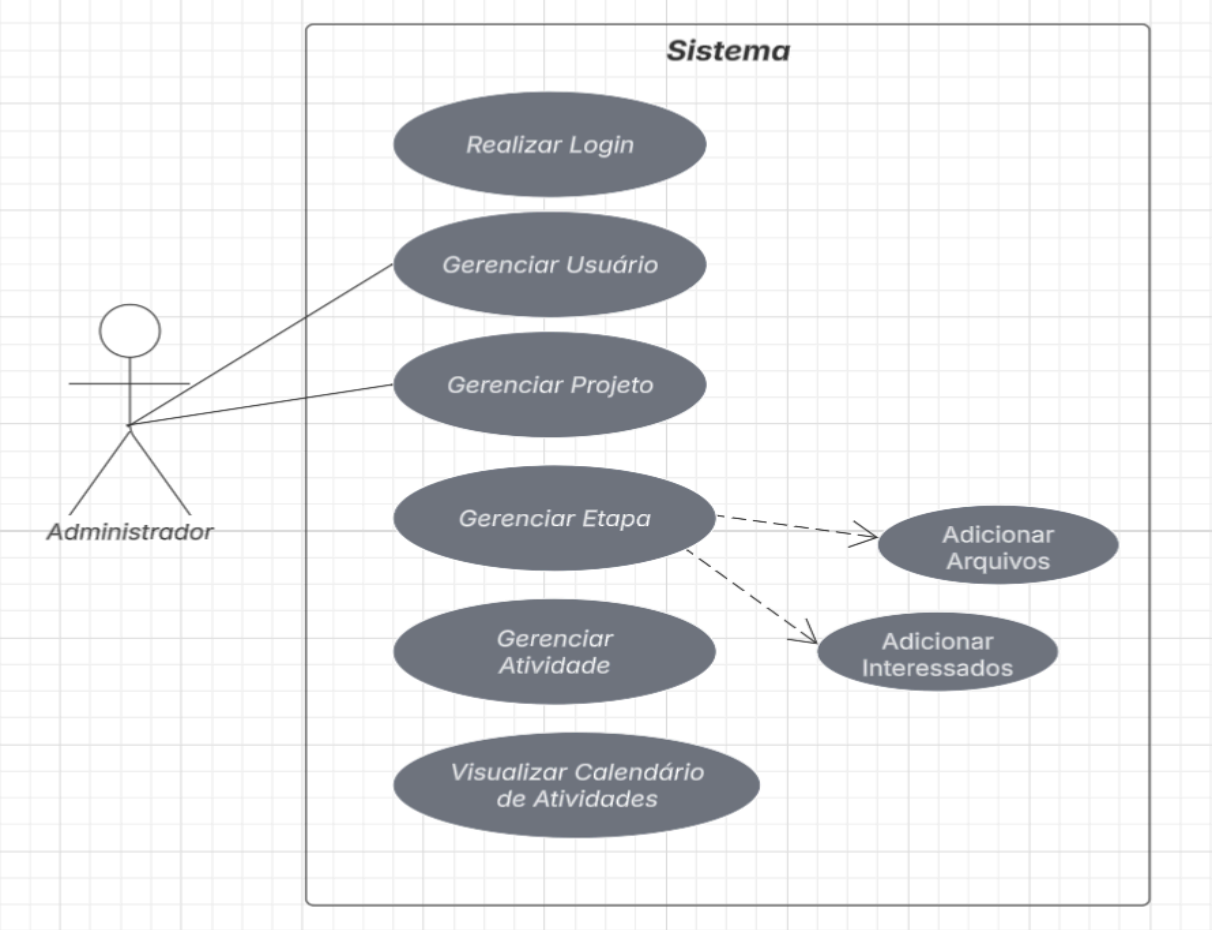
4.2 CASOS DE USO

Figura 2: Diagrama de casos de uso do usuário visitante.



Fonte: elaborado pelo autor.

Figura 3: Diagrama de casos de uso do usuário administrador.



Fonte: elaborado pelo autor.

Conforme a Figura 2 e Figura 3 é possível visualizar o diagrama de casos de uso da aplicação, na visão do visitante e do administrador, respectivamente, e analisar como os usuários terão acesso aos recursos disponíveis na aplicação segundo os requisitos funcionais e suas regras de acesso.

4.2.1 REALIZAR LOGIN

O recurso de autenticação do aplicativo possibilita que os usuários autorizados utilizem o sistema de forma segura. Quando o usuário efetua o login, ele fornece suas credenciais, sendo elas o nome de usuário e sua senha, que são validadas pelo sistema para assegurar a autenticidade da identidade.

Se os dados estiverem corretos, o sistema permite acesso ao conteúdo e às funcionalidades. Esta autenticação é vital para a proteção dos dados, assegurando

que apenas usuários autorizados tenham acesso a informações confidenciais ou possam executar ações específicas dentro do aplicativo.

A função de acesso deve incluir mecanismos de segurança, tais como criptografia de senha e monitoramento de tentativas de acesso, evitando ataques de força bruta e assegurando a integridade das informações. Quando ocorre um erro de login, o sistema deve emitir alertas claros, como a indicação de senhas inválidas ou a solicitação de redefinição da mesma, e permitir a recuperação da conta através de métodos seguros, como o envio de e-mails para redefinir a senha.

4.2.2 GERENCIAR PROJETOS

Gerenciar projetos é o pacote do CRUD de projetos, onde o usuário administrador poderá criar, editar, visualizar e excluir projetos existentes, tendo total acesso a todas as informações relacionadas.

Para o usuário visitante a funcionalidade de visualizar projetos permite a visualização de detalhes técnicos de um projeto devidamente cadastrado no sistema e deste modo, por esta funcionalidade o usuário pode ter acesso às informações relativas ao projeto, como autor, descrição, o montante em capital financeiro ligado ao projeto e outras informações relacionadas aos projetos.

4.2.3 GERENCIAR ETAPAS

Gerenciar etapas é o pacote do CRUD de etapas, onde o usuário administrador poderá criar, editar, visualizar e excluir etapas existentes, tendo total acesso a todas as informações relacionadas.

Para o usuário visitante a funcionalidade de visualizar etapas permite o acompanhamento das etapas dos projetos para que o mesmo possa se informar a respeito dos trâmites legais dos projetos e das implicações que cada situação pode causar no processo geral dos projetos.

4.2.4 GERENCIAR ATIVIDADES

Gerenciar etapas é o pacote do CRUD de etapas, onde o usuário administrador poderá criar, editar, visualizar e excluir etapas existentes, tendo total acesso a todas as informações relacionadas.

Esta funcionalidade fornece ao usuário visitante a possibilidade de ver e acompanhar as atividades vinculadas aos projetos e suas determinadas etapas e

por meio desta se informar a respeito dos processos necessários a serem realizados, incluindo o cronograma da atividade.

4.2.5 VISUALIZAR ARQUIVOS DE ATIVIDADES

Nesta funcionalidade o usuário visitante poderá visualizar todos os arquivos anexados às atividades um projeto e assim ter acesso a informações que podem não estar previamente declaradas na aplicação por meio de textos e também ter acesso a informações gerais sobre os projetos por meio dos arquivos anexados.

4.2.6 VISUALIZAR CALENDÁRIO DE ATIVIDADES

A funcionalidade de visualizar calendário de atividades permite ao usuário visualizar os cronogramas das atividades dos projetos que estão próximas da data de vencimento, facilitando para que o mesmo possa ver as datas importantes do trâmite dos projetos e, por meio deste calendário realizar os planejamentos e ações necessárias, bem como qualquer prestação de informações solicitadas.

4.2.7 GERENCIAR USUÁRIOS

A funcionalidade de gerenciar usuários é um pacote do CRUD de usuários, onde somente o administrador pode criar, editar informações, visualizar e excluir usuários existentes.

O usuário visitante poderá apenas visualizar seus próprios dados e fazer a mudança da senha de acesso caso necessário.

4.2.7 ADICIONAR INTERESSADOS

Durante o cadastro da atividade ou na edição o usuário administrador tem a opção de adicionar o e-mail de pessoas que têm alguma ligação com o projeto, assim sempre que ocorrer alguma atualização na atividade, esse interessado receberá uma notificação por e-mail informando detalhes da mudança, início ou conclusão.

4.2.7 ADICIONAR ARQUIVOS

Durante o cadastro da atividade ou em edição o usuário administrador tem a opção de adicionar arquivos relacionados a atividade, construindo assim uma gestão documental durante todas as fases do projeto, facilitando consulta e análise de resultados futuros.

4.3 HISTÓRIA DE USUÁRIO

"Uma história de usuário é uma descrição curta e simples de um recurso contada da perspectiva da pessoa que deseja o novo recurso, geralmente um usuário ou cliente do sistema" (MASSIMUS, 2025). Essas narrativas são elaboradas sob o ponto de vista do usuário final, apresentando funcionalidades específicas que satisfazem suas necessidades e anseios, e que devem ser postas em prática pela equipe de desenvolvimento.

Na tabela abaixo é possível observar as histórias de usuário de acordo com os requisitos funcionais e casos de uso.

Tabela 19: Histórias de usuário administrador.

Usuário do caso de uso	Funcionalidade	História
Usuário administrador.	Realizar login na aplicação	Como administrador quero realizar login para utilizar as funcionalidades do sistema.
	Criar usuários	Como administrador, quero criar usuários visualizadores para possibilitar acesso ao sistema.
	Visualizar projetos	Preciso acessar a tela de visualização de projetos para verificar as informações de um projeto do meu interesse.
	Visualizar etapas	Como administrador quero visualizar as etapas de um projeto para entender o ponto atual.
	Visualizar atividades	Quero ver as atividades que devem ser realizadas e que estão vinculadas a um projeto específico.
	Visualizar arquivos de atividades	Como administrador, quero ter acesso e ver anexos descritivos e

		informativos a respeito das atividades relacionadas a um projeto.
	Clonar Projeto	Como administrador, quero duplicar um projeto de maneira rápida usando a opção de clonagem.
	Criar Projeto	Como administrador, quero criar um projeto.
	Editar Projeto	Como administrador, quero editar as informações de um projeto.
	Excluir projeto	Como administrador, quero excluir um projeto.
	Criar Etapa	Como administrador, quero criar uma etapa.
	Editar Etapa	Como administrador, quero editar as informações de uma etapa.
	Excluir Etapa	Como administrador, quero excluir uma etapa.
	Criar Atividade	Como administrador, quero criar uma atividade.
	Editar Atividade	Como administrador, quero editar as informações de uma atividade.
	Excluir Atividade	Como administrador, quero excluir uma atividade.

Fonte: Elaborado pelo autor.

Tabela 20: Histórias de usuário visualizador.

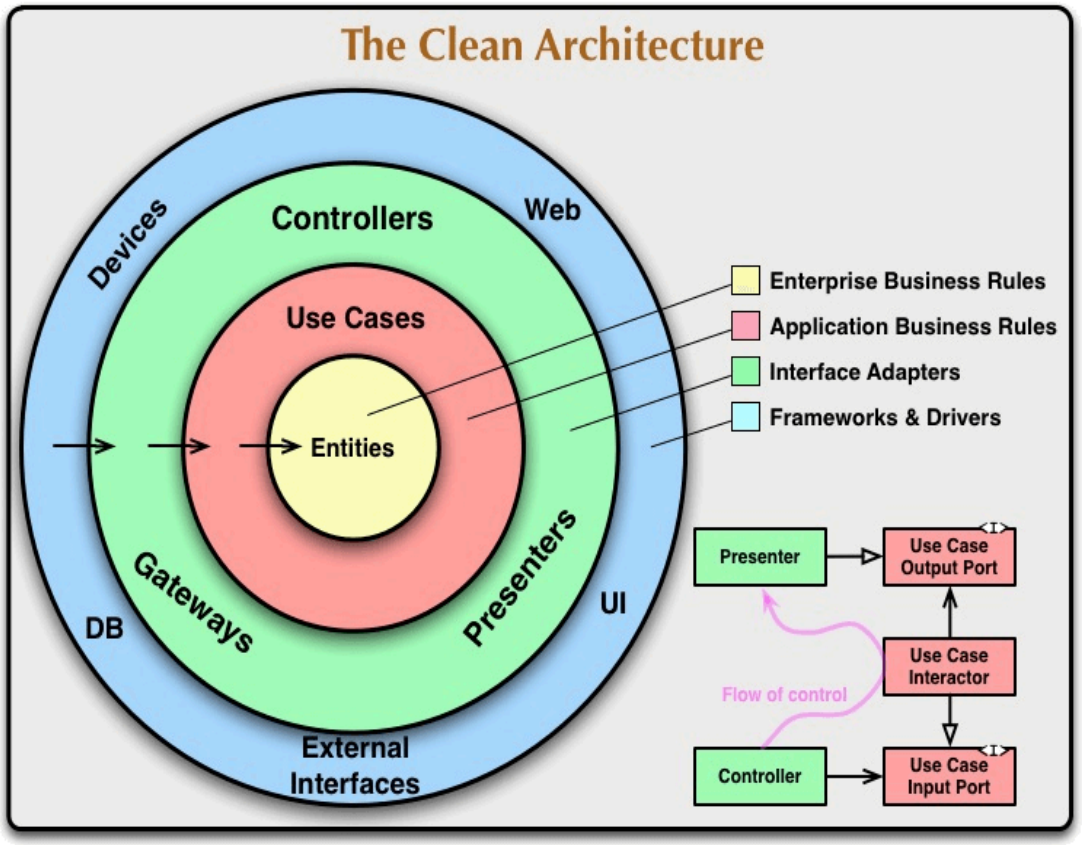
Usuário do caso de uso	Funcionalidade	História
Usuário visualizador.	Realizar login na aplicação	Como visualizador quero realizar login para utilizar as funcionalidades do sistema.
	Visualizar projetos	Como visualizador quero ver as informações dos projetos para me situar sobre qual área é mais requisitada.
	Visualizar etapas	Como visualizador quero ver as etapas de um projeto para me informar sobre o contexto atual do andamento e futura finalização.
	Visualizar atividades	Como visualizador quero ver as atividades de um projeto para me informar sobre o contexto atual do andamento.
	Visualizar arquivos de atividades	Como visualizador quero acessar os arquivos de uma atividade para encontrar uma informação.
	Visualizar calendário de atividades	Como visualizador quero ver as atividades próximas da data limite para reportar a informação.
	Alterar senha da conta	Como visualizador quero trocar a senha da conta para mantê-la segura.

4.4 PADRÃO DE PROJETO CLEAN ARCHITECTURE(ARQUITETURA LIMPA)

A arquitetura limpa é um modelo de arquitetura que tem como finalidade padronizar e organizar o código, facilitando a reusabilidade e permitindo um desacoplamento que permite uma independência maior da tecnologia utilizada (MASOTTI, 2021).

Ela sugere uma camada mais externa que se distingue das camadas internas do software. Neste cenário, é possível empregar abstrações, onde o uso do padrão de design de injeção de dependência permite a criação de interfaces sem a exigência de um código concreto. Assim, os frameworks podem ser empregados como um instrumento construtivo, em vez de se tornarem a principal regra de negócio do projeto (Boukhary & Colmenares, 2019).

Figura 4: Modelo visual de camadas da arquitetura limpa.



Fonte: https://zup.com.br/blog/clean-architecture-arquitetura-limpa

4.5. ESTRUTURA DA APLICAÇÃO

A API do sistema de gestão de emendas parlamentares é idealizada com um módulo principal que é focado no cadastro e atualização de projetos com seus devidos controles de cronogramas, comunicação com as partes interessadas e gestão documental.

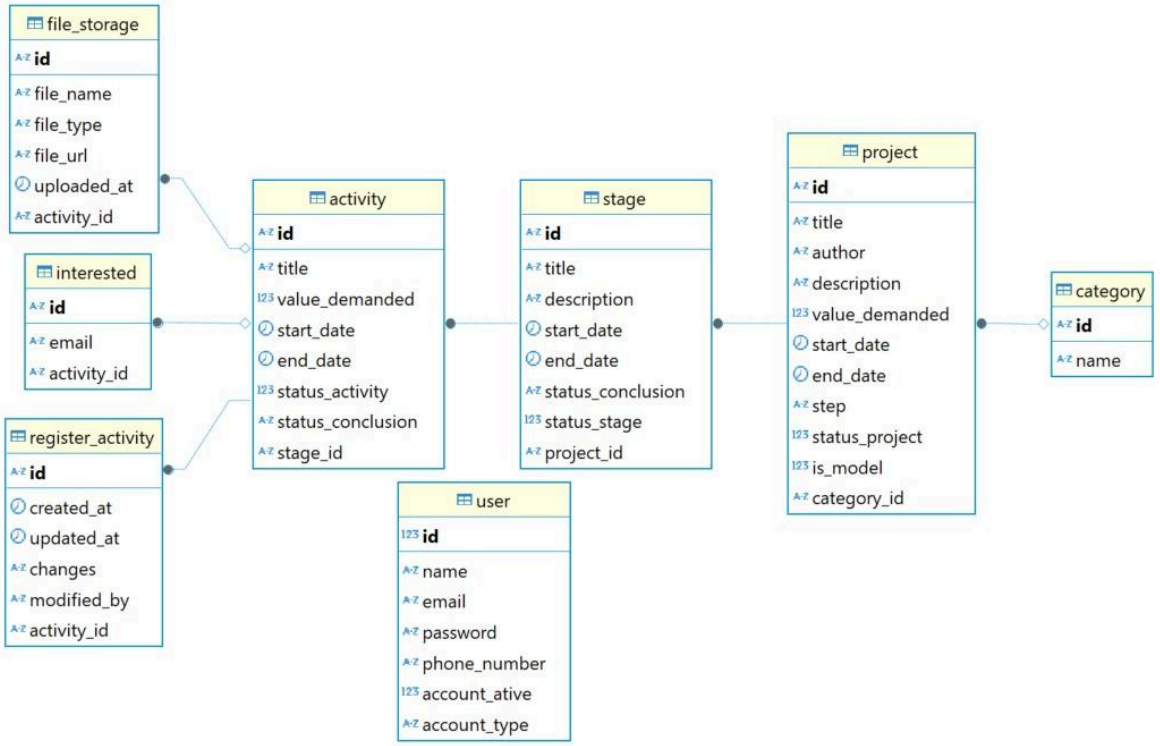
Esse módulo também conta com funcionalidades de visualização de informações a respeito dos projetos e lista de atividades próximas do vencimento para implementação de um calendário.

4.5.1 ARQUITETURA LÓGICA DE CAMADAS DO SOFTWARE

A arquitetura do software idealizado será composta por duas camadas principais: o front-end e o back-end. O back-end foi construído com Node.js, utilizando Typescript. O Node.js, com sua arquitetura baseada em eventos e I/O assíncrono, garante a escalabilidade e o desempenho da aplicação, lidando de forma eficiente com grandes volumes de requisições simultâneas. O back-end é responsável pela lógica de negócios, armazenamento e recuperação de dados, além da exposição de APIs que permitem a comunicação com o front-end.

4.5.2 ESTRUTURA DO BANCO DE DADOS

Figura 1: Diagrama do banco de dados.



Fonte: Elaborado pelo autor.

Este diagrama estrutural representa as relações entre as tabelas do banco de dados. As três tabelas principais são as de Project, Stage e Activity que têm relações one-to-many (um para muitos) entre Project e Stage, e, Stage e Activity.

No diagrama, a tabela Project está se relacionando com a tabela Category, onde cada projeto poderá ter uma categoria, facilitando o agrupamento e filtragem dos projetos. A tabela Activity se conecta com a tabela File_Storage, que são os anexos. Além disso, a tabela Activity também se relaciona com a tabela Interested, que é alguém que receberá notificações de atualizações na atividade. Por fim, a tabela Register_Activity, também relacionada com a tabela Activity, é uma espécie de registro de alterações na atividade com fim de consulta e relatórios.

Por sua vez, a tabela User armazena informações pessoais e o tipo de cada usuário. Não há relacionamento direto com outras entidades no modelo atual, uma vez que as operações do sistema não exigem esse vínculo explícito, sendo essas conexões tratadas por regras de negócio em níveis superiores da aplicação.

4.5.3 ESTRUTURAS E FUNCIONALIDADES

Nesta seção será desenvolvida uma abordagem teórica e descritiva a respeito das estruturas e funcionalidades contidas no escopo inicial da aplicação.

4.5.4 ESTRUTURA DE LOGIN

A funcionalidade de login é crucial para assegurar a proteção e o acesso restrito a áreas limitadas do aplicativo. Ela possibilita que os usuários se identifiquem usando suas credenciais, como nome de usuário e senha, antes de acessarem dados confidenciais ou executarem ações específicas no sistema.

Esta funcionalidade não apenas salvaguarda as informações do usuário, mas também proporciona uma experiência personalizada, identificando os gostos e o passado de cada indivíduo, o que aprimora a interação com o aplicativo. A implementação eficaz do login é crucial para a confiança dos usuários e para o funcionamento adequado do sistema, principalmente ao lidarmos com informações confidenciais ou ações relevantes que requerem permissão.

Para implementar o login no sistema, foi utilizado Node.js, que proporcionou uma solução rápida, escalável e segura. A autenticação é feita através de JWT (JSON Web Tokens), garantindo que as credenciais do usuário sejam validadas no momento do login e, após isso, um token seja gerado e enviado ao cliente. Esse token será utilizado nas requisições subsequentes para validar a identidade do usuário, sem precisar manter sessões no servidor. Além disso, as senhas dos usuários são protegidas utilizando técnicas de hashing com bcrypt, para garantir que não haja exposição de informações sensíveis.

4.5.5 MENSAGENS

A funcionalidade de enviar mensagens sobre atividades aos usuários interessados é um dos principais meios de interação entre os utilizadores. Ela possibilita a comunicação de atualizações e cronograma da atividade.

Ela permite que os usuários relacionados recebam mensagens via e-mail, mantendo um registro e recebendo atualizações sobre o andamento. Este elemento é essencial, pois, por meio desta funcionalidade os usuários estarão situados em relação aos projetos, suas etapas, atividades e outras questões pertinentes dentro do sistema.

4.5.6 ESTRUTURA DA GESTÃO DE PROJETOS

A estrutura dos projetos segue a dinâmica de um CRUD(Create, read, update e delete) que permite ao usuário administrador da aplicação cadastrar, visualizar, alterar e excluir projetos de emendas parlamentares da aplicação.

Levando em consideração que o principal propósito do CRUD no contexto do trabalho em questão é simplificar a gestão, supervisão e monitoramento de projetos financiados através de emendas parlamentares. Este sistema possibilita a geração, consulta, modificação e eliminação de registros associados aos projetos, assegurando uma administração mais eficaz e clara dos fundos públicos. Ao ser integrado em uma interface, os usuários têm a capacidade de adicionar novos detalhes sobre projetos, consultar informações já registradas, fazer alterações de acordo com o avanço dos projetos e eliminar registros quando necessário. Portanto, o sistema simplifica o fluxo de dados e a estruturação das informações, auxiliando em uma administração pública mais eficiente.

A função de "criar" (Create) possibilita o registro de novos projetos com informações como o valor destinado, a área de aplicação, as metas e os prazos. O módulo de "leitura" do sistema permite um acesso ágil às informações registradas, possibilitando que os encarregados da supervisão de emendas parlamentares acessem as informações em tempo real, sem a exigência de consultas manuais ou outros procedimentos demorados. Por outro lado, a opção "atualizar" (Update) possibilita o registro preciso de qualquer alteração no projeto, como alterações no orçamento, prazo e escopo, assegurando que todas as informações estejam sempre atualizadas e acessíveis para consulta. A função "Excluir" (Delete) é empregada com cautela, possibilitando a remoção de projetos que foram interrompidos ou registrados de maneira errônea do sistema. Isso contribui para manter a base de dados organizada e exata.

A aplicação deste CRUD favorece a transparência e o controle dos projetos financiados por emendas parlamentares, permitindo que as pessoas interessadas monitorem a alocação dos fundos, cronograma e a realização dos projetos. Assim, o sistema não só aprimora o procedimento administrativo, como também reforça a responsabilidade e o dever de prestação de contas dos administradores públicos.

4.5.7 ARQUIVOS

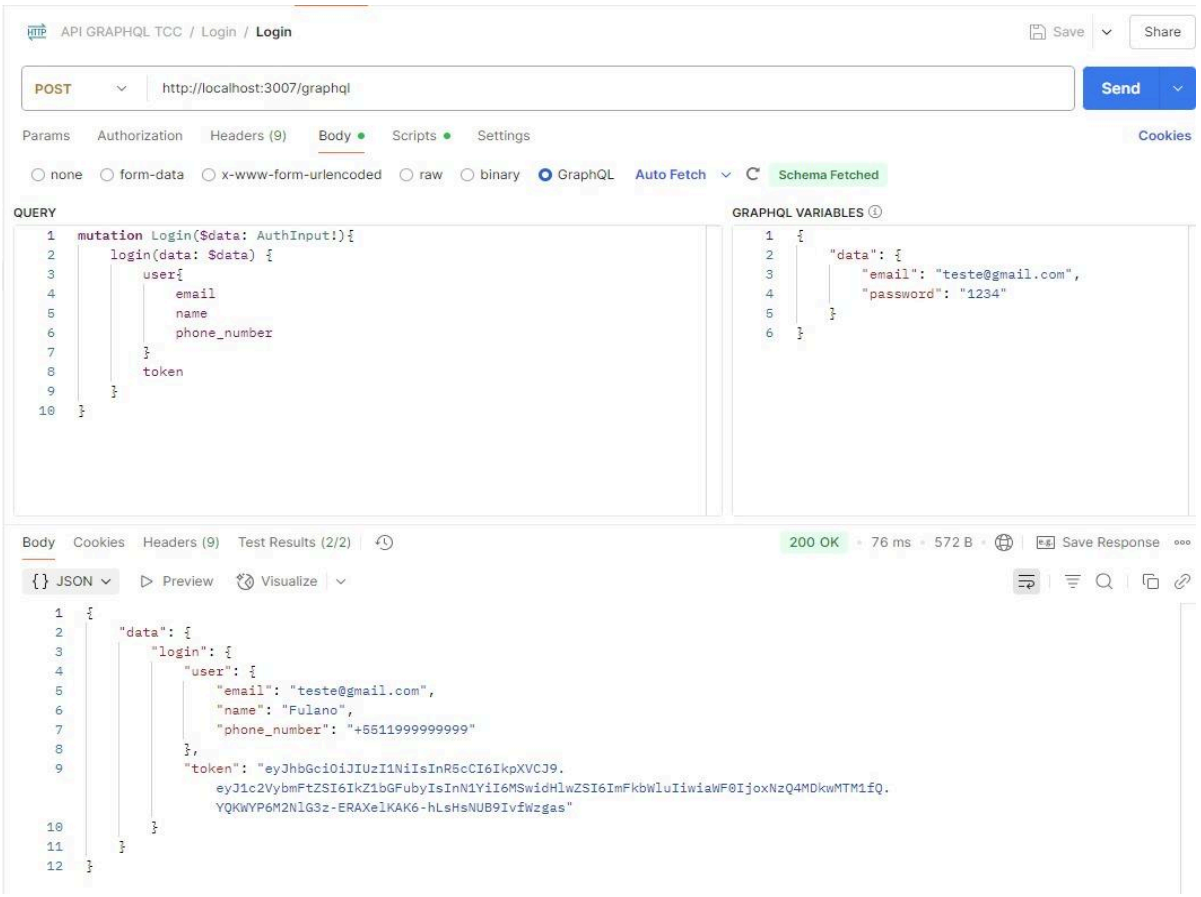
A funcionalidade de adicionar arquivos relacionados às atividades dos projetos é uma função que auxiliará na documentação do projeto, possibilitando uma gestão melhor das informações e decisões importantes tomadas ao longo de sua execução.

4.6 REQUISIÇÕES DA APLICAÇÃO

Nesta seção serão apresentadas algumas requisições graphQl da api, não serão exibidas todas pois ficaria muito extenso, serão exibidas as principais e também as que contemplem cada contexto do funcionamento.

4.6.1 REQUISIÇÃO DE LOGIN

Figura 5: Requisição de login.

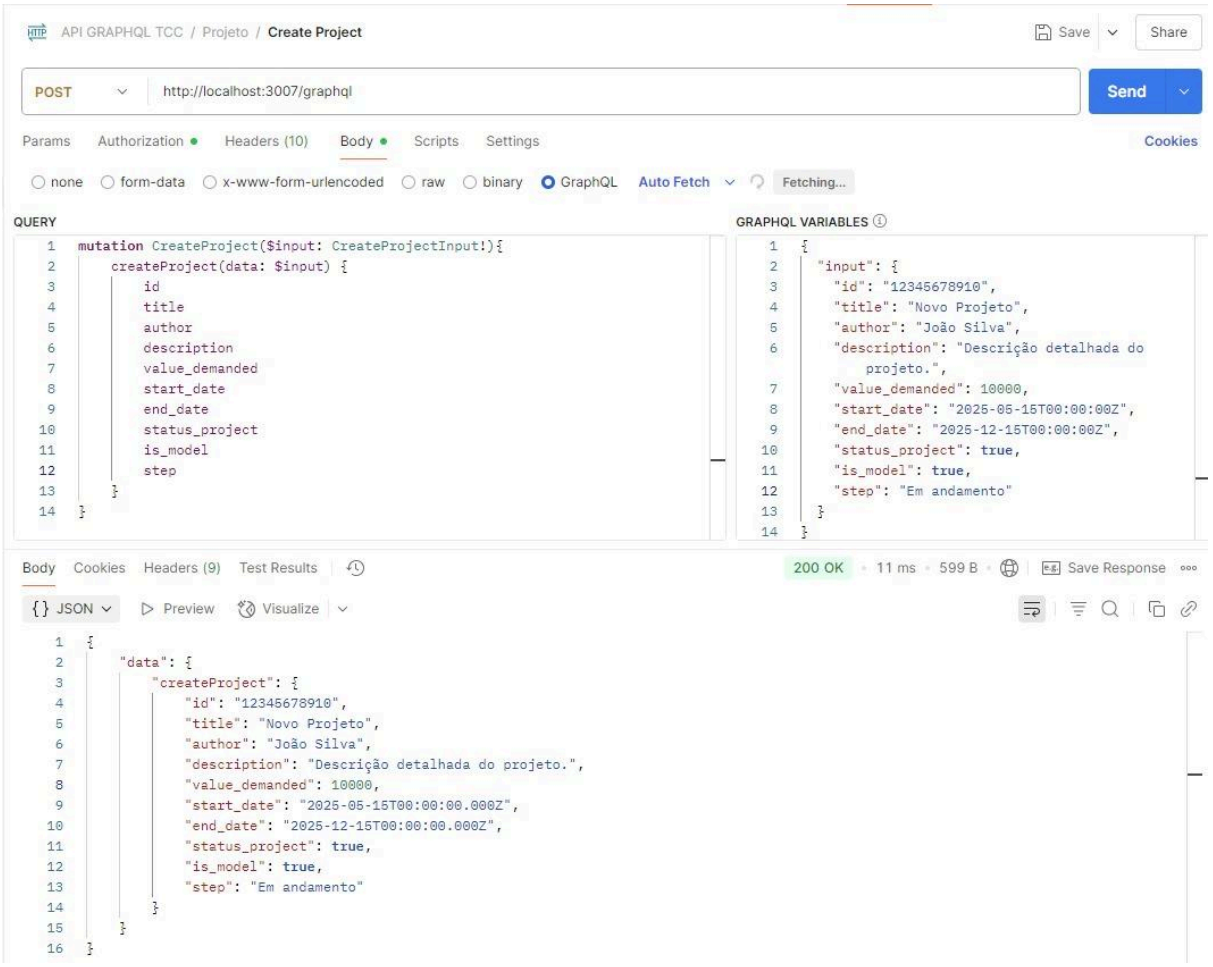


Fonte: Elaborado pelo autor.

A Figura 5 apresenta a estrutura da requisição responsável pelo processo de login do usuário. Como dados de entrada, são exigidos o endereço de e-mail e a senha previamente cadastrados. Como resposta, o sistema retorna os dados do usuário autenticado, bem como um token de acesso que será utilizado para o controle da sessão do usuário.

4.6.2 REQUISIÇÃO DE CRIAÇÃO DE PROJETO

Figura 6: Requisição de criação de projeto.

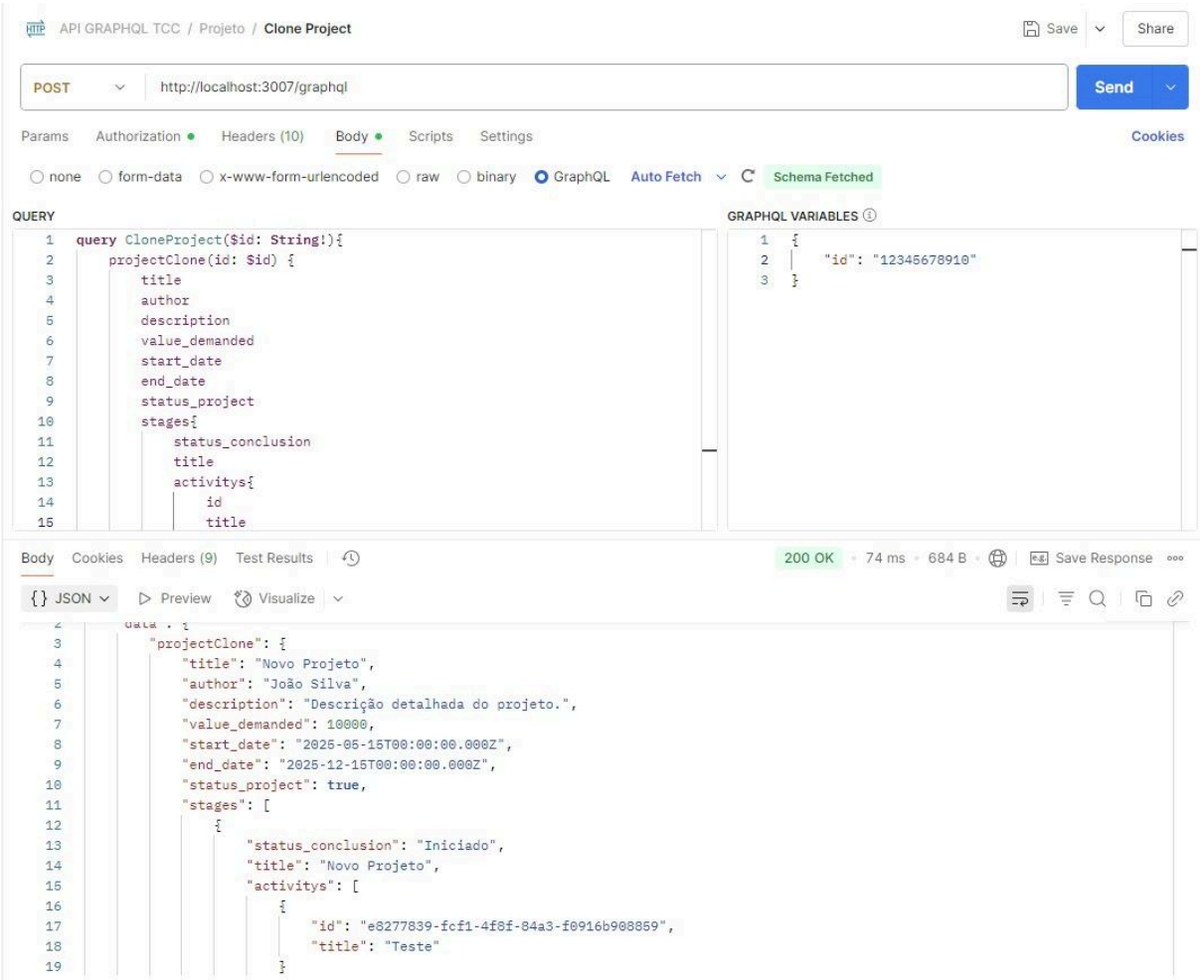


Fonte: Elaborado pelo autor.

A Figura 6 apresenta a estrutura da requisição responsável pela criação de um projeto. Como dados de entrada, são exigidos todos os dados definidos como obrigatórios para um projeto na regra de negócio. Como resposta bem sucedida, recebemos os dados do projeto criado definidos no corpo da resposta.

4.6.3 REQUISIÇÃO DE CLONAGEM DE PROJETO

Figura 7: Requisição de clonagem de projeto.

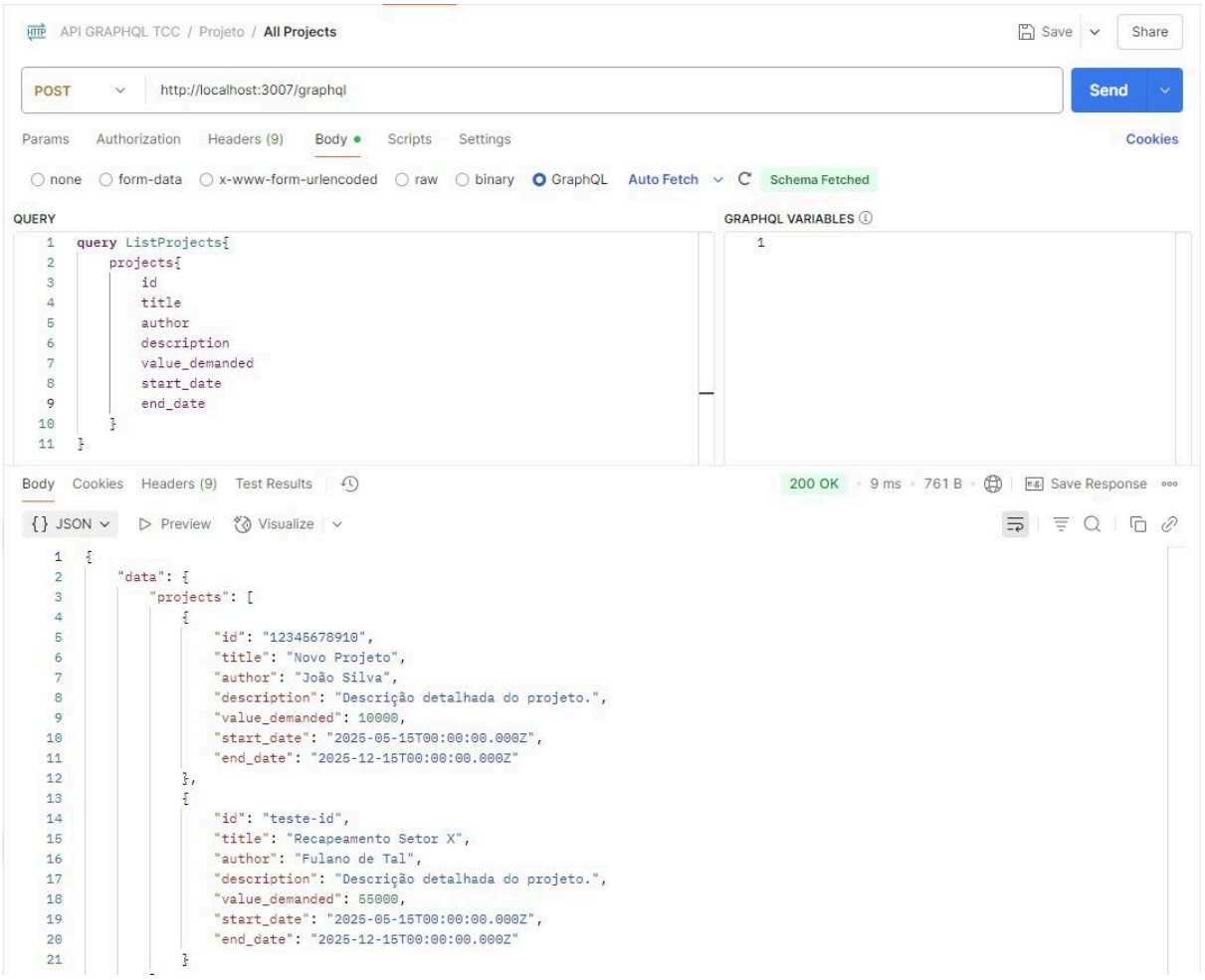


Fonte: Elaborado pelo autor.

A Figura 7 apresenta a estrutura da requisição responsável pela clonagem de um projeto existente. Como dado de entrada, é exigido o identificador do projeto que será duplicado. Como resposta bem sucedida, recebemos os dados do projeto criado definidos no corpo da resposta.

4.6.4 REQUISIÇÃO DE LISTAGEM DE PROJETOS

Figura 8: Requisição de listagem de projetos.

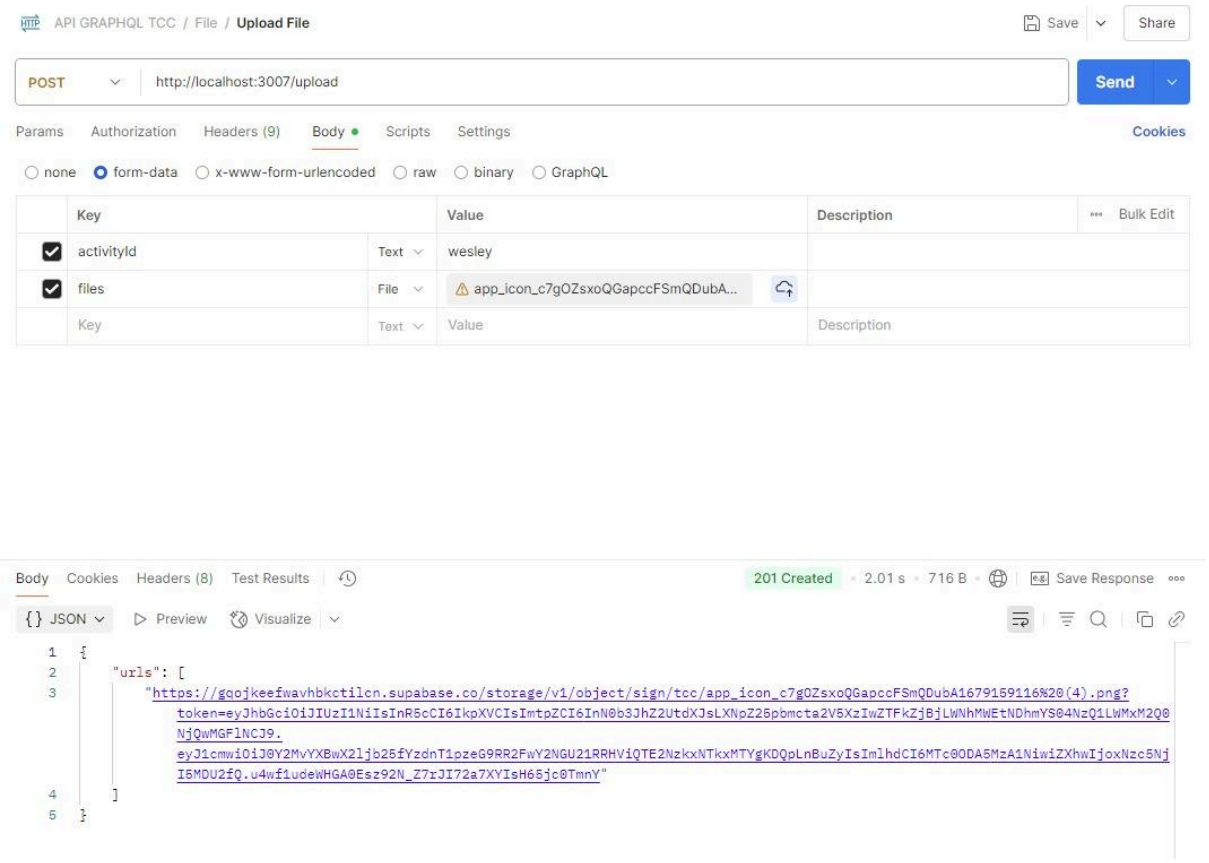


Fonte: Elaborado pelo autor.

A Figura 8 apresenta a estrutura da requisição responsável pela listagem de projetos existentes. Ela não exige dado de entrada, sendo apenas uma requisição de consulta. Como resposta bem sucedida, recebemos a lista com dados dos projetos existentes.

4.6.5 REQUISIÇÃO DE UPLOAD DE ARQUIVOS

Figura 9: Requisição de upload de arquivos.

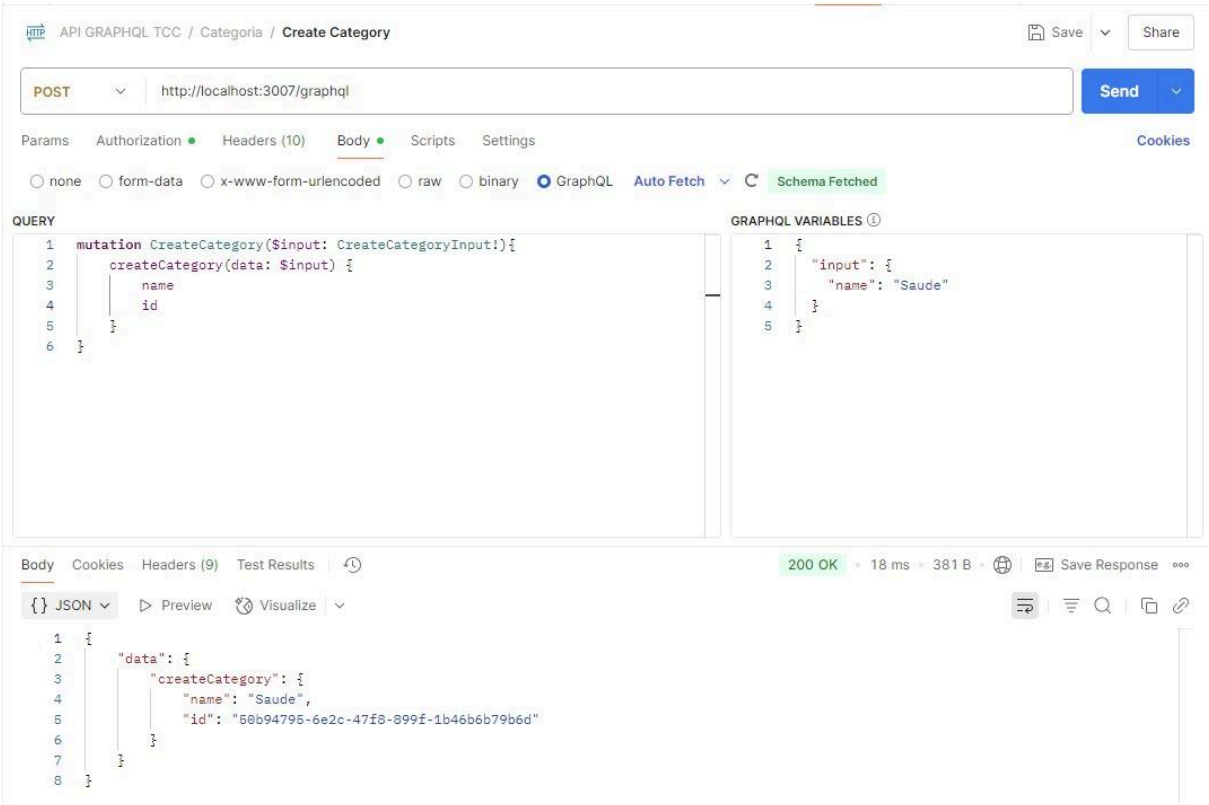


Fonte: Elaborado pelo autor.

A Figura 9 apresenta a estrutura da requisição responsável pela listagem dos projetos existentes. Essa é a única requisição do sistema implementada seguindo o padrão REST, devido à menor complexidade em relação à utilização do GraphQL nesse caso específico. Como dados de entrada, são enviados o identificador da atividade e o arquivo a ser anexado. Em caso de sucesso, a resposta retorna a URL do arquivo armazenado no local previamente configurado.

4.6.6 REQUISIÇÃO DE CRIAÇÃO DE CATEGORIA

Figura 10: Requisição de criação de categoria.

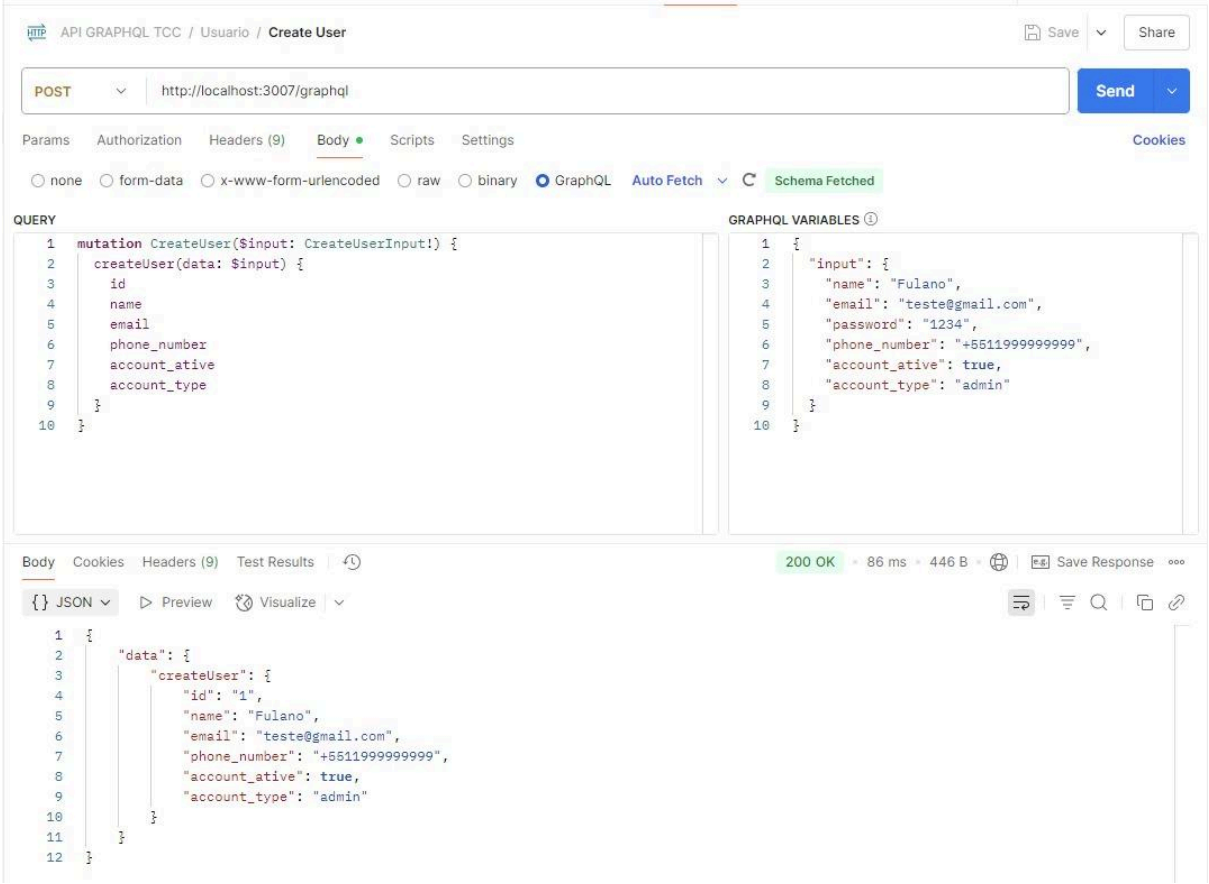


Fonte: Elaborado pelo autor.

A Figura 10 apresenta a estrutura da requisição responsável pela criação de uma categoria de projeto. Como dado de entrada, é exigido o nome da categoria. Como resposta bem sucedida, recebemos o identificado e o nome da categoria criada.

4.6.7 REQUISIÇÃO DE CRIAÇÃO DE USUÁRIO

Figura 11: Requisição de criação de usuário.

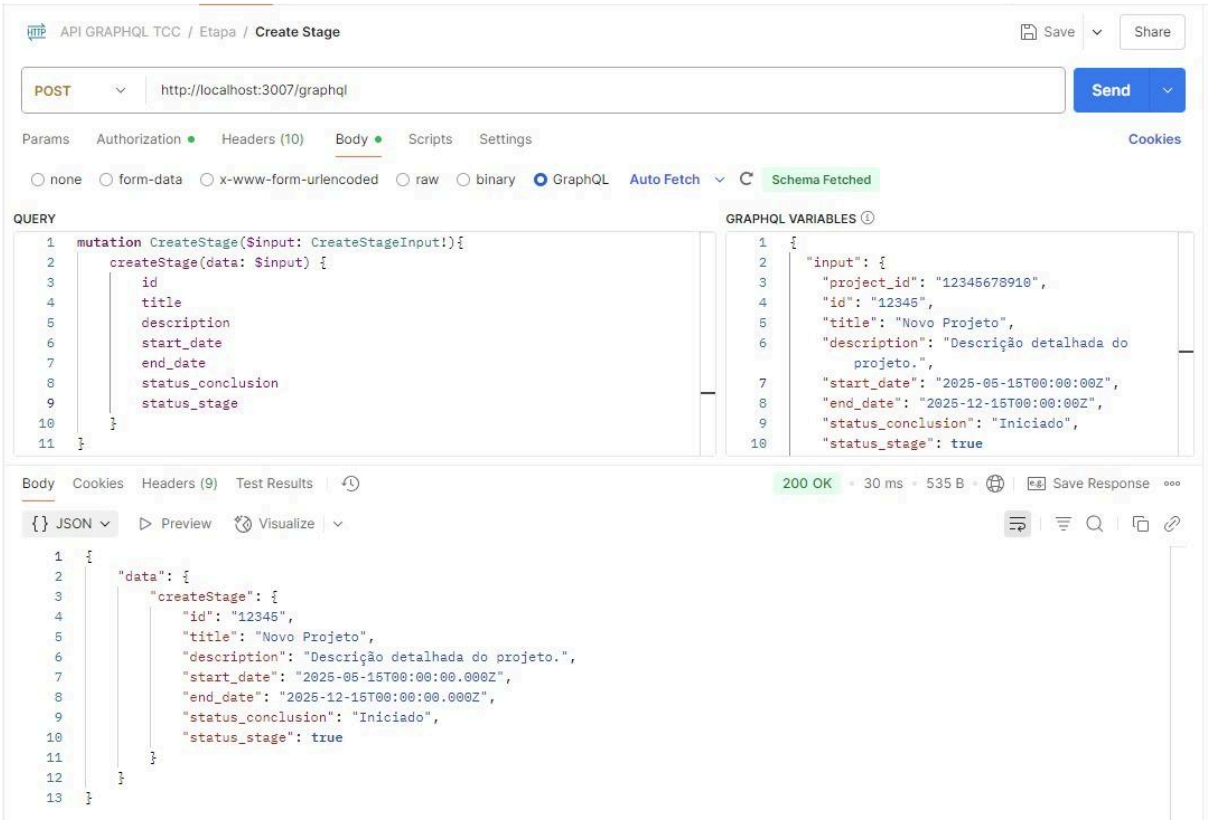


Fonte: Elaborado pelo autor.

A Figura 11 apresenta a estrutura da requisição responsável pela criação de um usuário. Como dados de entrada, são exigidos todos os dados definidos como obrigatórios para um usuário na regra de negócio, incluindo a senha de acesso. Como resposta bem sucedida, recebemos os dados da etapa criada definidos no corpo da resposta.

4.6.8 REQUISIÇÃO DE CRIAÇÃO DE ETAPA

Figura 12: Requisição de criação de etapa.

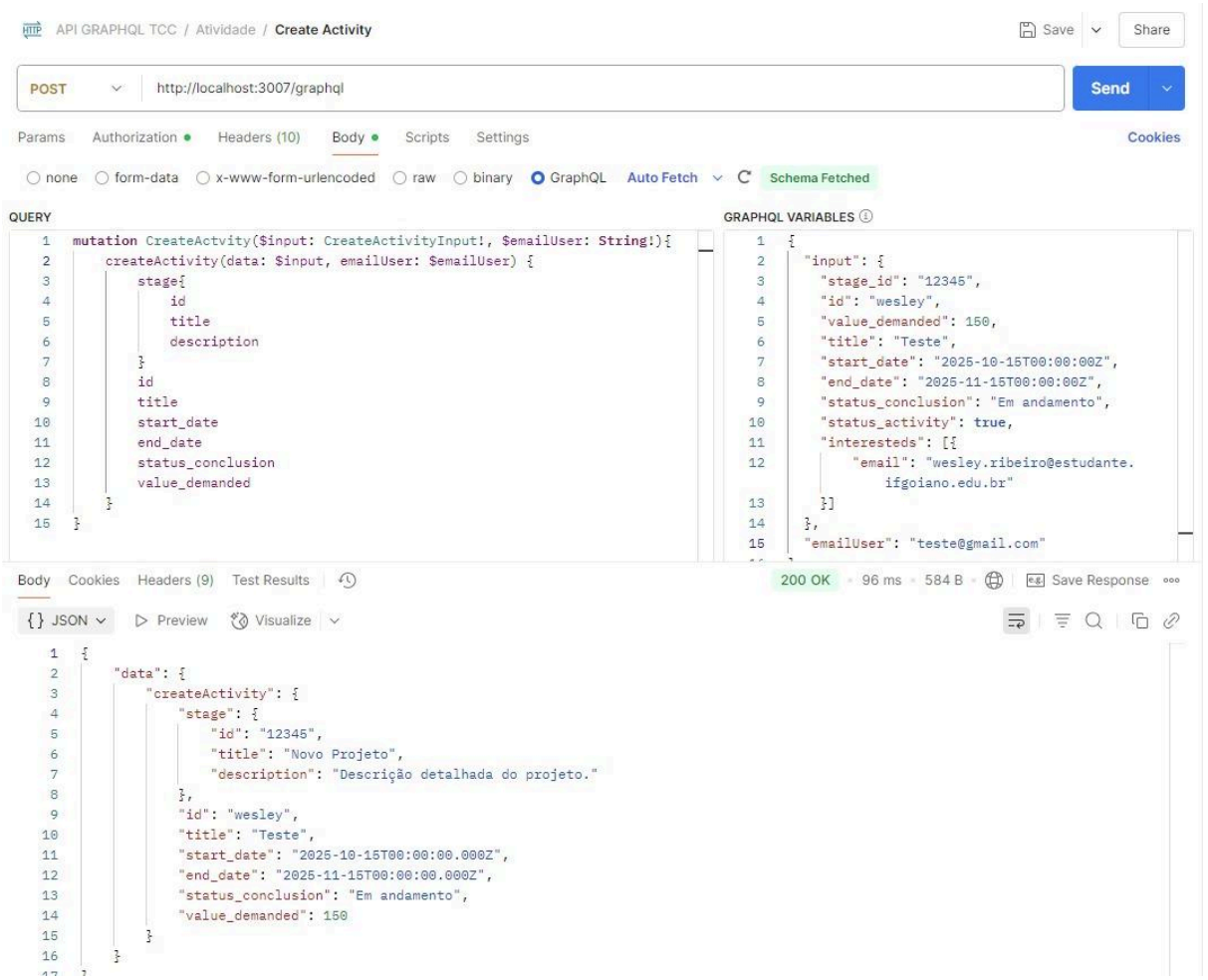


Fonte: Elaborado pelo autor.

A Figura 12 apresenta a estrutura da requisição responsável pela criação de uma etapa. Como dados de entrada, são exigidos o identificador do projeto e todos os dados definidos como obrigatórios para uma etapa na regra de negócio. Como resposta bem sucedida, recebemos os dados da etapa criada definidos no corpo da resposta.

4.6.9 REQUISIÇÃO DE CRIAÇÃO DE ATIVIDADE

Figura 13: Requisição de criação de atividade.



Fonte: Elaborado pelo autor.

A Figura 13 apresenta a estrutura da requisição responsável pela criação de uma atividade. Como dados de entrada, são exigidos todos os campos obrigatórios definidos nas regras de negócio para a atividade, incluindo o identificador da etapa associada, o e-mail do usuário responsável pela requisição e, opcionalmente, uma lista de usuários interessados. Em caso de sucesso, a resposta retorna os dados da atividade criada, conforme definidos no corpo da resposta.

5. CONCLUSÃO

O presente trabalho teve como objetivo apresentar um projeto software que integra as tecnologias da informação nas práticas de gestão pública, destacando seu papel essencial orquestrando a gestão de projetos, possibilitando a otimização do uso de recursos financeiros e humanos através da análise dos dados e o auxílio nas apresentações de resultados obtidos através dos recursos das emendas parlamentares.

A proposta de desenvolvimento de uma API voltada para gestores públicos busca não apenas aprimorar a execução de emendas parlamentares e a alocação de recursos, mas também viabilizar um ambiente de maior acessibilidade e clareza das ações governamentais, pois oferece os dados de maneira organizada e filtrada para análise da gestão.

A implementação dessa plataforma visa, portanto, proporcionar tanto aos gestores quanto aos parlamentares ferramentas que facilitem o acompanhamento, a fiscalização e a análise das políticas públicas desenvolvidas no município.

Ao permitir o acesso a dados integrados sobre a execução de projetos, emendas parlamentares e seus impactos financeiros, a API proposta se configura como uma ferramenta estratégica para os gestores, oferecendo uma visão detalhada e em tempo real sobre a alocação e a utilização de recursos. Isso possibilita decisões mais informadas e fundamentadas, assegurando uma gestão pública mais eficaz e responsável.

Além disso, a construção da API abrange aspectos fundamentais da engenharia de software, como o desenvolvimento back-end, a estruturação de bases de dados e a realização de testes de segurança e integração. Essas etapas são cruciais para garantir a robustez e a confiabilidade do sistema, aspectos essenciais para a operação em cenários de alta demanda de dados e consultas. A execução de testes durante o processo de desenvolvimento visa assegurar que a plataforma funcione de forma eficiente, atendendo aos requisitos técnicos e operacionais necessários para o bom desempenho em diferentes situações.

O impacto dessa solução no município de Ceres, Goiás, demonstrará o potencial da tecnologia para transformar a gestão pública em um nível local. A implementação da API nesse contexto serve como um modelo de inovação e

eficiência, contribuindo para uma administração pública mais otimizada, acessível e transparente.

Por fim, o trabalho evidencia que a adoção de ferramentas tecnológicas na administração pública não apenas contribui para a melhoria da gestão de recursos, mas também fortalece o processo democrático, ao disponibilizar meios adequados para acompanhar e fiscalizar a execução das políticas públicas. A transparência e a eficiência são, portanto, elementos indissociáveis de uma gestão pública moderna, e a proposta de desenvolvimento da API se configura como uma contribuição significativa para o aprimoramento das práticas de governança, promovendo uma administração pública mais responsável, acessível e democrática.

A API desenvolvida neste trabalho constitui uma base sólida para a gestão de projetos oriundos de emendas parlamentares em ambientes organizacionais. No entanto, há diversas melhorias e evoluções planejadas para as próximas versões.

Entre os próximos passos previstos, destaca-se a integração com ferramentas de mensagens, como WhatsApp ou Telegram, com o objetivo de automatizar notificações e facilitar a comunicação com usuários diretamente pelos canais que já utilizam. Essa funcionalidade permitirá o envio de alertas sobre prazos de atividades, atualizações de status e lembretes de forma mais dinâmica e acessível.

Além disso, está prevista a implementação de filtros personalizados para geração de relatórios, possibilitando aos usuários extrair informações mais específicas e relevantes de acordo com suas necessidades. Isso incluirá filtros por data, status, responsáveis, entre outros critérios, contribuindo para uma análise mais precisa dos dados e melhor tomada de decisão.

Essas melhorias visam ampliar a usabilidade e a eficiência do sistema, aproximando a aplicação das demandas reais do contexto onde será utilizada.

6. REFERÊNCIAS

ANACLETO, Joaquim Alberto da Costa. Desenvolvimento de uma aplicação web para dispositivos móveis: monitorização e controle de uma rede digital signage. 2012. Disponível em: https://repositorium.sdum.uminho.pt/bitstream/1822/28062/1/eeum_di_dissertacao_pg15447.pdf. Acesso em: 5 set. 2024.

ALVES FILHO, Emilio Maltez; MARTINEZ, Antonio Lopo. Requisitos funcionais de um sistema de informações para gestão de custos no setor público. In: **Anais do Congresso Brasileiro de Custos-ABC. 2006.**

BATISTA, Igor Gabriel Silva et al. Aplicação web para gestão de prontuários eletrônicos de atendimento aos pacientes da equoterapia do IF Goiano - Campus Ceres. 2023.

BOUKHARY, S.; COLMENARES, E. A clean approach to Flutter development through the Flutter Clean Architecture package. In: *International Conference on Computational Science and Computational Intelligence*, 2019. p. 1115-1120.

BRASIL. Constituição (1988). *Constituição da República Federativa do Brasil*. Brasília, DF: Senado Federal, 1988. Disponível em: https://www.planalto.gov.br/ccivil_03/constituicao/constituicao.htm. Acesso em: 22 maio 2025.

BRASIL. Controladoria-Geral da União. Portal da Transparência: emendas parlamentares. Brasília: CGU, 2025. Disponível em: https://portaldatransparencia.gov.br/entenda-a-gestao-publica/emendas-parlamentares. Acesso em: 1 maio 2025.

BRASIL. Ministério da Justiça e Segurança Pública. Emendas Parlamentares. Disponível em: https://www.gov.br/mj/pt-br/assuntos/seus-direitos/sal/emendas-parlamentares**.** Acesso em: 20 maio 2025**.**

GOLDBERG, Josh. *Learning TypeScript.* Sebastopol: O'Reilly Media, Inc., 2022.

GOMES, Wilson; AMORIM, Paula Karini Dias Ferreira; ALMADA, Maria Paula. Novos desafios para a ideia de transparência pública. *E-Compós*, 2018.

GONZAGA, Flávio S.; BIRCKAN, Guilherme. *Curso de PHP e MySQL.* Florianópolis, out. 2000.

MASSIMUS. Histórias de usuários. Disponível em: https://massimus.com/historias-de-usuarios/. Acesso em: 20 maio 2025.

MASOTTI, Danilo. Clean Architecture: o que é arquitetura limpa e quais os seus benefícios. *Zup Innovation*, [S. l.], 14 out. 2021. Disponível em: https://zup.com.br/blog/clean-architecture-arquitetura-limpa. Acesso em: 20 maio 2025.

MAYER, Gregory Nicholas et al. Go-Teachers: aplicação web para compartilhamento de videoaulas de língua inglesa por professores voluntários. 2021.

MINISTÉRIO DA IGUALDADE RACIAL (Brasil). *Manual de emendas parlamentares*. [S. l.]: Governo Federal, [2024?]. Disponível em: https://www.gov.br/igualdaderacial/pt-br/acesso-a-informacao/assessoria-parlamentar/emendas-parlamentares/ManualdeEmendasParlamentaresdigital.pdf. Acesso em: 10 jun. 2025.

NESTJS. A progressive Node.js framework. Disponível em: https://nestjs.com/. Acesso em: 20 maio 2025.

PAULINO, Leonardo Soares. Node.js. *ReFAQI - Revista de Gestão, Educação e Tecnologia*, v. 4, n. 2, p. 3, 2018.

PIRES, J. A. M., Jr. A realização orçamentária e financeira de emendas orçamentárias e o seu controle pelo executivo por meio da (in)fidelidade parlamentar. 2005. Monografia (Graduação) — Escola de Administração Fazendária, Brasília, DF, Brasil.

PRESCOTT, Preston. *SQL para iniciantes.* [S.l.]: Babelcube Inc., 2015.

PRICE, Jason. *Oracle Database 11g SQL: domine SQL e PL/SQL no banco de dados Oracle.* Porto Alegre: Bookman Editora, 2009.

RIBEIRO, L. et al. O impacto das tecnologias de informação e telecomunicação no setor público brasileiro: uma revisão bibliográfica. *Revista FT*, 2023. Disponível em: https://revistaft.com.br/o-impacto-das-tecnologias-de-informacao-e-telecomunicacao-no-setor-publico-brasileiro-uma-revisao-bibliografica/. Acesso em: 5 maio 2025.

RICHARDS, Gregor; ZAPPA NARDELLI, Francesco; VITEK, Jan. Concrete types for TypeScript. In: *29th European Conference on Object-Oriented Programming (ECOOP 2015).* Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.

SACRAMENTO, A. R. S.; PINHO, J. A. G. Transparência na administração pública: o que mudou depois da Lei de Responsabilidade Fiscal? Um estudo exploratório em seis municípios da região metropolitana de Salvador. *Revista de Contabilidade da UFBA*, v. 1, n. 1, p. 48-61, 2007.

SODRÉ, Antonio Carlos de Azevedo; ALVES, Maria Fernanda Colaço. Relação entre emendas parlamentares e corrupção municipal no Brasil: estudo dos relatórios do Programa de Fiscalização da Controladoria-Geral da União. *Revista de Administração Contemporânea*, v. 14, p. 414-433, 2010.

TEIXEIRA, T.; CRISTINA SABO, I. Democracia ou autocracia informacional? O papel da Internet na sociedade global do século XXI. 2016. [PDF].