UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

VINICIUS GABRIEL CIUNEK

CAMPUS ALERTA - PLATAFORMA PARA GESTÃO DE PROBLEMAS NA INFRAESTRUTURA DA UTFPR

GUARAPUAVA

2026

VINICIUS GABRIEL CIUNEK

CAMPUS ALERTA - PLATAFORMA PARA GESTÃO DE PROBLEMAS NA INFRAESTRUTURA DA UTFPR

CAMPUS ALERTA - PLATFORM FOR MANAGING INFRASTRUCTURE PROBLEMS AT UTFPR

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

Orientador: Dr. Luciano Ogiboski

Coorientador: Dr. Emerson André Fedechen

GUARAPUAVA

2026

VINICIUS GABRIEL CIUNEK

CAMPUS ALERTA - PLATAFORMA PARA GESTÃO DE PROBLEMAS NA INFRAESTRUTURA DA UTFPR

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

Data de aprovação: 01/julho/2026

Dr. Roni Fabio Banaszewski
Doutor
Universidade Tecnológica Federal do Paraná

Dr. Carlos Eduardo Andrade Iatskiu
Doutor
Universidade Tecnológica Federal do Paraná

Dr. Luciano Ogiboski
Doutor
Universidade Tecnológica Federal do Paraná

GUARAPUAVA

2026

RESUMO

A gestão da infraestrutura física na Universidade Tecnológica Federal do Paraná (UTFPR) enfrenta desafios relacionados à descentralização e à rastreabilidade das solicitações de manutenção, muitas vezes dependentes de processos informais ou sistemas legados restritivos. Diante disso, este trabalho desenvolveu o sistema “Campus Alerta”, uma plataforma digital para a gestão colaborativa e eficiente de chamados de manutenção predial. A solução adotou uma abordagem de desenvolvimento Low-Code utilizando a plataforma OutSystems, culminando na entrega de um Produto Mínimo Viável (MVP) no formato de Progressive Web App (PWA). Para facilitar o registro e garantir a organização das demandas, a identificação do local do problema foi estruturada permitindo aos usuários reportar ocorrências de forma simples e direta, através da seleção em listas padronizadas dos blocos e ambientes da instituição. A metodologia aplicada e exploratória utilizou o framework ágil Scrum para o desenvolvimento iterativo e a técnica MoSCoW para a priorização de requisitos. Os resultados alcançados incluem a modelagem do fluxo e da estrutura de dados, a implementação das principais funcionalidades do sistema e a realização de verificações funcionais internas com dados controlados. O MVP contempla autenticação local, registro de ocorrências com evidências fotográficas, acompanhamento do estado dos chamados, comunicação entre solicitante e equipe técnica e apresentação de indicadores gerenciais. A avaliação empírica com grupos de usuários não integrou o escopo desta etapa, permanecendo como proposta para trabalhos futuros. Concluiu-se que a solução constitui uma Prova de Conceito funcional para demonstrar os fluxos identificados no contexto analisado, embora sua eventual implantação institucional exija avaliação técnica, financeira e tecnológica adicional.

Palavras-chave: gestão de infraestrutura; low-code; outsystems; manutenção predial; smart campus.

The management of physical infrastructure at the Federal University of Technology – Paraná (UTFPR) faces challenges related to decentralization and the traceability of maintenance requests, often relying on informal processes or restrictive legacy systems. In response, this work developed the “Campus Alerta” system, a digital platform for the collaborative and efficient management of building maintenance requests. The solution adopted a Low-Code development approach using the OutSystems platform, culminating in the delivery of a Minimum Viable Product (MVP) as a Progressive Web App (PWA). To facilitate registration and ensure demand organization, the identification of problem locations was structured to allow users to report incidents simply and directly through standardized lists of the institution’s buildings and rooms. The exploratory methodology employed the Scrum agile framework for iterative development and the MoSCoW technique for requirement prioritization. The results include the modeling of the workflow and data structure, the implementation of core system functionalities, and the execution of internal functional verifications using controlled data. The MVP includes local authentication, incident reporting with photographic evidence, call status tracking, communication between the requester and the technical team, and the presentation of management indicators. Empirical evaluation with user groups was not part of the scope of this stage, remaining as a proposal for future work. It was concluded that the solution serves as a functional Proof of Concept to demonstrate the workflows identified in the analyzed context, although its potential institutional deployment requires further technical, financial, and technological assessment.

Keywords: Infrastructure management; low-code; OutSystems; building maintenance; smart campus.

LISTA DE FIGURAS

Figura 1 – Diagrama de atividades do sistema Campus Alerta **20**
Figura 2 – Diagrama Entidade-Relacionamento do sistema Campus Alerta **23**
Figura 3 – Tela de Login e Autenticação do Usuário **31**
Figura 4 – Tela de Cadastramento de Usuário **32**
Figura 5 – Tela de Abertura de Chamado **33**
Figura 6 – Visão do Técnico para gestão e atualização de status do chamado . . . **34**
Figura 7 – Visão do Técnico para gestão e troca de mensagens com solicitante . . **35**
Figura 8 – Dashboard gerencial com indicadores de manutenção **35**
Figura 9 – Aggregate com dados das mensagens do chamado **37**
Figura 10 – Fluxo lógico visual para envio de mensagens e atualização da interface **37**
Figura 11 – Aggregates utilizados na consolidação dos indicadores **38**
Figura 12 – Estrutura da tela gerencial no OutSystems **38**

LISTA DE QUADROS

Quadro 1 – Síntese comparativa das soluções analisadas **18**
Quadro 2 – Histórias de Usuário e Requisitos de Negócio (*Must Have*) **22**
Quadro 3 – Dicionário de dados do sistema Campus Alerta **25**
Quadro 4 – Resultados da verificação funcional do MVP **39**

LISTA DE ABREVIATURAS E SIGLAS

Siglas

DER	Diagrama Entidade-Relacionamento
DESEG	Departamento de Segurança e Serviços Gerais
IOT	Internet of Things
ITSM	IT Service Management (Gerenciamento de Serviços de TI)
MVP	Produto Mínimo Viável, do inglês *Minimum Viable Product*
PWA	Progressive Web App
RAD	Desenvolvimento Rápido de Aplicação, do inglês *Rapid Application Development*
SGBD	Sistema Gerenciador de Banco de Dados
UTFPR	Universidade Tecnológica Federal do Paraná

SUMÁRIO

1 INTRODUÇÃO 9

1.1 Considerações iniciais 9

1.2 Objetivos 10

1.2.1 Objetivo Geral 10

1.2.2 Objetivos Específicos 10

1.3 Justificativa 11

1.4 Estrutura do trabalho 11

2 REFERENCIAL TEÓRICO 13

2.1 Smart Campus e Facility Management 13

2.2 Desenvolvimento Low-Code e PWA 13

3 TRABALHOS RELACIONADOS 15

3.1 Análise de Soluções Existentes 15

3.1.1 GLPI e sua utilização na UTFPR 15

3.1.2 Jira Service Management 16

3.1.3 Zammad 16

3.1.4 Síntese Comparativa 16

4 ANÁLISE E PROJETO DA SOLUÇÃO 19

4.1 Fluxo de Funcionamento 19

4.2 Perfis de Usuários 21

4.2.1 Internos (Alunos e Servidores) 21

4.2.2 Equipe de Manutenção (Técnicos) 21

4.2.3 Gestor do Campus (Administrador) 21

4.3 Histórias de Usuário e Requisitos Funcionais 21

4.4 Modelagem do Banco de Dados 22

5 MATERIAIS E MÉTODOS 26

5.1 Materiais 26

5.1.1 Plataforma de Desenvolvimento: OutSystems 26

5.1.2 Infraestrutura de Dados 26

5.2 Métodos 27

5.2.1 Estudo e Adaptação Tecnológica 27

5.2.2 Levantamento, Engenharia de Requisitos e Priorização 27
5.2.3 Ciclo de Desenvolvimento 28
5.2.4 Procedimentos de Validação e Verificação 28
6 RESULTADOS E DISCUSSÃO 30
6.1 Detalhamento dos Sprints e Evolução do Projeto 30
6.2 Apresentação do Sistema Implementado 30
6.2.1 Autenticação e Cadastro 31
6.2.2 Abertura de Chamados 31
6.2.3 Atendimento e Comunicação 32
6.2.4 Dashboard Gerencial 32
6.2.5 Decisões Técnicas e Lógicas Implementadas 33
6.2.6 Armazenamento de Imagens 33
6.2.7 Comunicação e Consultas Visuais 34
6.3 Verificação Funcional 39
6.4 Avaliação Qualitativa com o DESEG 39
7 CONSIDERAÇÕES FINAIS 41
7.1 Trabalhos Futuros 42
REFERÊNCIAS 44

1 INTRODUÇÃO

A gestão eficiente da infraestrutura física é um dos pilares fundamentais para a garantia da qualidade do ensino e do bem-estar em instituições educacionais. Na Universidade Tecnológica Federal do Paraná (UTFPR), a manutenção de um campus complexo, composto por diversos blocos, laboratórios e áreas de convivência, impõe desafios logísticos significativos. Atualmente, a comunicação de problemas estruturais — como falhas elétricas, hidráulicas ou danos em equipamentos — ocorre de maneira descentralizada e, por vezes, informal, o que dificulta a rastreabilidade das solicitações e o planejamento da equipe de manutenção.

Neste contexto, o conceito de *Smart Campus* (Campus Inteligente) surge como uma alternativa viável, utilizando tecnologias da informação para otimizar processos e recursos. A proposta deste trabalho foi o desenvolvimento do sistema “Campus Alerta”, uma solução tecnológica que visou modernizar esse fluxo. Diferente das abordagens tradicionais de desenvolvimento de software, este projeto utilizou uma plataforma de desenvolvimento *Low-Code* (OutSystems) para a criação de um Produto Mínimo Viável, do inglês *Minimum Viable Product* (MVP).

A escolha pelo *Low-Code* justificou-se pela necessidade de agilidade na entrega de soluções digitais e pela facilidade de manutenção e escalabilidade. Embora concebido inicialmente para o cenário da UTFPR, a arquitetura da solução foi concebida de forma modular, permitindo que seus conceitos e regras de negócio possam subsidiar futuras adaptações para outras instituições públicas.

Considerando os desafios identificados e a necessidade de otimização das demandas junto ao órgão responsável, definiu-se o seguinte problema de pesquisa: **Como uma solução digital integrada pode estruturar e automatizar a coleta de requisitos de manutenção predial, transformando solicitações informais e descentralizadas em dados técnicos acionáveis para a gestão do campus?**

1.1 Considerações iniciais

A qualidade da infraestrutura física impacta diretamente a experiência universitária. O bem-estar da comunidade acadêmica, composta por alunos e servidores, depende da funcionalidade de salas, laboratórios e espaços de convivência (LOMAS; JONES, 2006). No entanto, a manutenção desta grande estrutura ainda representa um desafio logístico.

No cenário atual, a UTFPR utiliza uma instância própria do GLPI para o registro e a triagem de solicitações. Embora essa plataforma permita a formalização dos chamados, o levantamento realizado junto ao DESEG indicou que parte do fluxo operacional de manutenção ocorre por canais paralelos, como grupos de mensagens instantâneas. Essa fragmentação dificulta a consolidação do histórico de atendimento, a comunicação com o solicitante e a recuperação imediata de indicadores gerenciais.

A restrição de acesso e o gargalo na comunicação geram dispersão de informações e dificultam a alocação eficiente da equipe de manutenção. O sistema Campus Alerta foi desenvolvido com o propósito de solucionar essa dor, integrando troca de mensagens diretas na tela do celular do usuário Progressive Web App (PWA) e democratizando o acesso de forma inteligente, substituindo a burocracia por um fluxo transparente e de rápida comunicação.

1.2 Objetivos

Esta seção apresenta os objetivos traçados para a pesquisa e o desenvolvimento da ferramenta.

1.2.1 Objetivo Geral

Desenvolver uma plataforma digital multiplataforma para apoiar a gestão colaborativa de chamados de manutenção predial, implementando uma prova de conceito aplicada ao contexto da UTFPR e estruturando requisitos que possam subsidiar futuras expansões para outras instituições.

1.2.2 Objetivos Específicos

Os objetivos específicos definidos para alcançar o propósito do trabalho foram:

- **Mapear** os fluxos de abertura e atendimento de chamados para definir os requisitos do MVP.
- **Investigar** a viabilidade técnica e a produtividade do uso de ferramentas *Low-Code* no contexto acadêmico e na gestão pública.
- **Desenvolver** a aplicação Web e Mobile na plataforma OutSystems, trazendo como valor o mobile first para toda comunidade.
- **Definir** e implementar um mecanismo de autenticação baseado em credenciais cadastradas na aplicação, permitindo controle de acesso ao MVP e estabelecendo base conceitual para futura integração com mecanismos institucionais de autenticação.
- **Realizar** a validação funcional do protótipo por meio de testes técnicos dos fluxos implementados, identificando oportunidades para futuras avaliações com usuários reais.

1.3 Justificativa

A realização deste trabalho justificou-se pela necessidade de modernizar um processo administrativo crucial para a qualidade da experiência na UTFPR. Atualmente, a restrição de acesso e a interface desatualizada do sistema legado geram ineficiências que afetam diretamente a agilidade dos reparos. A implementação de uma plataforma única e acessível a alunos e servidores não apenas agiliza a comunicação, mas também cria um ambiente mais transparente e confiável.

A principal contribuição deste trabalho é a construção de uma Prova de Conceito (PoC) funcional que materializa a digitalização e centralização do fluxo de manutenção predial. Embora o Produto Mínimo Viável (MVP) tenha sido desenvolvido utilizando a plataforma proprietária OutSystems — visando o ganho de produtividade e agilidade na prototipação inerentes ao paradigma *Low-Code* —, o projeto atua como um estudo de viabilidade técnica. O *Campus Alerta* estabelece as bases lógicas, a engenharia de requisitos e os fluxos de interface necessários para a gestão de *facilities* educacionais. Dessa forma, a documentação conceitual e as regras de negócio aqui validadas poderão servir de alicerce para futuras implementações definitivas por parte de órgãos públicos, utilizando pilhas tecnológicas de código aberto (*Open Source*) que mitiguem custos de licenciamento e previnam o *vendor lock-in* (dependência tecnológica).

O projeto possui também uma forte justificativa social e de extensão. Ao projetar a arquitetura visando futura evolução para múltiplas instituições, o Campus Alerta não se limita aos muros da universidade, podendo ser ofertado futuramente como uma ferramenta de gestão de zeladoria para escolas públicas e órgãos municipais, retornando o investimento de conhecimento à sociedade.

Por fim, o projeto justificou-se por fornecer dados estratégicos para a instituição. Conforme validado junto à gestão de infraestrutura do campus, a adoção de um painel administrativo (Dashboard) que mapeie os locais com mais problemas é uma ferramenta essencial para prever e planejar manutenções preventivas futuras. A implementação do Campus Alerta visou sanar a antiga dificuldade de rastreio, criando uma base de dados estruturada que permite à UTFPR aplicar inteligência e eficiência na gestão de seus recursos.

1.4 Estrutura do trabalho

O presente trabalho está organizado em sete capítulos. O **Capítulo 1** contextualiza o problema, apresenta os objetivos e a justificativa da pesquisa. **O Capítulo 2** reúne os fundamentos teóricos relacionados a *Smart Campus*, gestão de facilidades, desenvolvimento *Low-Code* e aplicações web progressivas. **O Capítulo 3** apresenta e compara soluções existentes de gerenciamento de serviços e chamados. O **Capítulo 4** descreve a análise e o projeto da solução, incluindo o fluxo de funcionamento, os perfis de usuários, os requisitos funcionais e a modelagem do banco de dados. O **Capítulo 5** detalha os materiais, as ferramentas e os procedimentos

metodológicos utilizados. O **Capítulo 6** apresenta os resultados e a discussão, abrangendo a evolução por *sprints*, o sistema implementado, as decisões técnicas, a verificação funcional e a avaliação qualitativa realizada com o setor consultado. Por fim, o **Capítulo 7** apresenta as considerações finais, as limitações e as propostas de trabalhos futuros.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta a fundamentação teórica necessária para a compreensão deste trabalho, abordando os conceitos tecnológicos e de gestão que dão suporte à construção do sistema Campus Alerta.

2.1 Smart Campus e Facility Management

Além da infraestrutura física, o conceito de Smart Campus busca integrar pessoas, processos e tecnologias para melhorar a experiência acadêmica e a eficiência operacional (MIN-ALLAH; ALRASHED, 2020).

O conceito de *Smart Campus* (Campus Inteligente) tem ganhado destaque na literatura acadêmica como uma evolução natural das cidades inteligentes, aplicada ao ambiente universitário. Segundo (SHETTY, 2016), um *Smart Campus* utiliza a Internet of Things (IOT) e sistemas de informação para otimizar recursos, melhorar a segurança e facilitar a manutenção de infraestruturas.

Dentro deste contexto, a Gestão de Facilidades (*Facility Management*) em universidades deixa de ser um processo reativo e passa a ser integrado.

Além da otimização da infraestrutura física, o conceito de Smart Campus busca integrar pessoas, processos e tecnologias para melhorar a eficiência operacional das instituições de ensino superior. O uso de plataformas digitais para coleta e análise de dados possibilita decisões mais assertivas e melhora a experiência da comunidade acadêmica.

2.2 Desenvolvimento Low-Code e PWA

Para responder à necessidade de entregas ágeis no desenvolvimento de software, as plataformas *Low-Code* surgem como uma alternativa viável e eficiente. Sahay (SAHAY *et al.*, 2020) afirmam que o *Low-Code* permite a abstração de código complexo, acelerando a entrega de valor tecnológico. Segundo pesquisas recentes da área de Engenharia de Software, o uso de Low-Code pode acelerar o ciclo de entrega, tornando essa abordagem relevante para projetos que demandam prototipação rápida e são desenvolvidos por equipes reduzidas.

Estudos recentes demonstram que plataformas Low-Code reduzem o esforço de desenvolvimento e aceleram a entrega de soluções corporativas (WASZKOWSKI, 2019).

Em paralelo, a adoção de PWA garante que os usuários (estudantes, professores e funcionários) não precisem instalar aplicativos nativos em seus smartphones. De acordo com Biørn-Hansen (BIøRN-HANSEN; MAJCHRZAK; GRøNLI, 2018), as PWAs oferecem uma experiência semelhante à de aplicativos nativos, com a vantagem de consumirem menos recursos

de armazenamento e garantirem atualizações instantâneas, o que é crucial para ambientes de alta rotatividade como uma universidade.

Essas características tornam a abordagem especialmente relevante para projetos acadêmicos e instituições com equipes reduzidas de desenvolvimento.

A aplicação de plataformas *Low-Code* no setor público pode ser observada em iniciativas internacionais. Em 2025, a OutSystems, em parceria com a empresa Copper River, obteve uma autorização de operação para utilização da plataforma no Departamento de Saúde e Serviços Humanos dos Estados Unidos. A autorização possibilita a implantação de aplicações construídas com a tecnologia em áreas vinculadas ao órgão, constituindo um exemplo de utilização de *Low-Code* em um ambiente governamental sujeito a requisitos específicos de segurança e governança (OutSystems, 2025).

Entretanto, esse exemplo não permite afirmar que a tecnologia seja amplamente adotada em todas as administrações públicas, tampouco que sua aplicação seja diretamente transferível ao contexto brasileiro. A adoção de uma plataforma proprietária por instituições públicas depende de análises relacionadas a custos de licenciamento, proteção de dados, soberania tecnológica, integração com sistemas existentes e dependência do fornecedor. Nesse sentido, a utilização da OutSystems neste trabalho restringe-se à construção acadêmica de uma Prova de Conceito, destinada à implementação e à avaliação técnica dos fluxos propostos, não representando uma definição da infraestrutura tecnológica para uma eventual implantação institucional.

3 TRABALHOS RELACIONADOS

Neste capítulo, são apresentados os conceitos teóricos que fundamentam a proposta do Campus Alerta, bem como uma análise de soluções de mercado existentes, a fim de posicionar o projeto e destacar seus diferenciais.

3.1 Análise de Soluções Existentes

A UTFPR utiliza uma instância própria do GLPI para o registro e a triagem de solicitações institucionais. Entretanto, conforme identificado no levantamento exploratório, parte do fluxo operacional relacionado à manutenção predial ocorre por canais externos. Diante desse cenário, esta seção compara o Campus Alerta com o GLPI e outras ferramentas consolidadas de gerenciamento de serviços.

3.1.1 GLPI e sua utilização na UTFPR

O GLPI é uma plataforma de código aberto voltada ao gerenciamento de serviços, ativos tecnológicos e chamados de suporte (GLPI Project, 2025). Sua arquitetura extensível permite a instalação de complementos, a personalização de formulários e a adaptação de fluxos conforme as necessidades de cada organização.

No contexto da UTFPR, conforme identificado durante o levantamento exploratório com um representante do DESEG do campus Guarapuava, é utilizada uma instância própria do GLPI, configurada internamente para o recebimento e a triagem de solicitações. A utilização de uma solução de código aberto representa uma vantagem institucional, pois permite maior autonomia técnica, possibilidade de customização e redução da dependência de fornecedores proprietários.

Entretanto, no fluxo operacional relatado, parte do processo de manutenção ocorre fora da plataforma. Após a análise de um chamado, as informações são encaminhadas à equipe executora por meio de grupos de mensagens instantâneas, frequentemente acompanhadas de capturas de tela. Após a realização do serviço, o retorno também pode ocorrer por esses canais, inclusive por meio de fotografias que comprovam a execução da atividade.

Essa combinação entre o GLPI e ferramentas externas contribui para a fragmentação das informações e dificulta a consolidação do histórico completo de atendimento em um único ambiente. Também foi relatada dificuldade para recuperar prontamente indicadores gerenciais, como o número de chamados de determinado período, a distribuição das demandas por área e os tempos associados ao atendimento.

Cabe destacar que essas limitações não devem ser atribuídas de forma absoluta à ferramenta GLPI, uma vez que a plataforma dispõe de recursos de configuração, relatórios e ex-

tensões. As dificuldades observadas estão relacionadas principalmente à configuração vigente e ao processo operacional adotado no contexto analisado.

Diante desse cenário, o Campus Alerta diferencia-se por ter sido concebido especificamente para o fluxo de manutenção predial do campus, priorizando uma interface móvel simplificada, a seleção estruturada de blocos e salas, o envio de imagens, a comunicação vinculada ao chamado e a apresentação direta de indicadores gerenciais. Contudo, diferentemente do GLPI, que possui código aberto e pode ser hospedado e customizado pela própria instituição, o MVP do Campus Alerta foi desenvolvido em uma plataforma proprietária, o que limita sua adoção institucional direta e reforça seu caráter de Prova de Conceito.

3.1.2 Jira Service Management

O Jira Service Management, da Atlassian, é uma das plataformas de IT Service Management (Gerenciamento de Serviços de TI) (ITSM) mais robustas do mercado (ATLASSIAN, 2025). Ele é projetado para gerenciar solicitações de TI e fluxos complexos. **Limitações frente ao Campus Alerta:** Apesar de sua potência, o Jira possui uma curva de aprendizado elevada e uma interface complexa para o usuário final (aluno), o que pode desestimular o reporte rápido de problemas simples.

3.1.3 Zammad

Zammad é um sistema de *help desk* de código aberto, flexível e com custo acessível (ZAMMAD, 2025). Ele permite a criação de tickets via múltiplos canais. **Limitações frente ao Campus Alerta:** Sua natureza genérica é sua principal limitação para o cenário de manutenção predial. A interface não é otimizada para uso em campo (mobile-first) por equipes de manutenção que precisam anexar fotos e atualizar status em tempo real. Diferente do Campus Alerta, que está sendo desenhado sob medida para a realidade física do campus (blocos e salas), o Zammad exigiria um esforço técnico considerável de adaptação para oferecer a mesma agilidade de localização das ocorrências.

3.1.4 Síntese Comparativa

A partir das características identificadas nas soluções analisadas, elaborou-se uma síntese comparativa considerando critérios relacionados ao licenciamento, à finalidade principal, à adaptação ao contexto de manutenção predial, à experiência de uso em dispositivos móveis, à comunicação vinculada aos chamados e à disponibilidade de indicadores gerenciais. O objetivo da comparação não é estabelecer uma superioridade absoluta entre as ferramentas, mas

identificar as diferenças entre soluções generalistas de gerenciamento de serviços e a proposta especializada do Campus Alerta.

Quadro 1 – Síntese comparativa das soluções analisadas

Solução	**Licenciamento**	**Finalidade principal**	**Adequação à manutenção predial**	**Diferenciais e limitações no contexto analisado**
GLPI	Código aberto	Gerenciamento de serviços, ativos e chamados	Pode ser adaptado por meio de configurações e extensões	Já utilizado pela UTFPR. Possui alta capacidade de customização, porém, no fluxo observado, parte da comunicação e do encaminhamento ocorre por canais externos, fragmentando o histórico operacional.
Jira Service Management	Proprietário	Gerenciamento de serviços e fluxos de atendimento	Pode ser configurado para diferentes tipos de chamados	Apresenta recursos robustos de automação e relatórios, porém possui maior complexidade de configuração e uma interface menos direcionada ao registro rápido de problemas prediais pela comunidade acadêmica.
Zammad	Código aberto	Centralização de atendimentos e suporte por múltiplos canais	Exige adaptação para representar blocos, salas e fluxos de manutenção	Oferece recursos de atendimento e comunicação, mas sua proposta é generalista e não foi concebida especificamente para a gestão de infraestrutura física de um campus.
Campus Alerta	Plataforma proprietária no MVP	Gestão de chamados de manutenção predial	Concebido especificamente para o contexto de blocos, salas, categorias e equipes técnicas	Apresenta interface *Mobile First*, seleção estruturada de locais, envio de imagens, chat e indicadores direcionados à manutenção. Como limitação, sua implantação institucional depende de licenciamento ou reimplementação em tecnologias abertas.

Fonte: Autoria própria (2026).

4 ANÁLISE E PROJETO DA SOLUÇÃO

A solução desenvolvida, denominada **Campus Alerta**, consiste em uma plataforma digital de gestão de manutenção predial desenvolvida sobre a infraestrutura *Low-Code* da OutSystems. O sistema atua como um intermediário ágil entre a comunidade acadêmica (usuários dos espaços físicos) e as equipes de gestão de infraestrutura da UTFPR.

Um dos principais diferenciais técnicos e funcionais da solução é o estabelecimento de uma plataforma de comunicação bidirecional entre o solicitante e a equipe técnica. A implementação de um sistema de conversação (*chat*) integrado nos detalhes do chamado ataca diretamente uma das principais dores operacionais da UTFPR: a ausência de retorno informativo ao usuário relatante. Ao permitir a interação em tempo real entre a equipe técnica e a comunidade acadêmica, o sistema garante maior transparência e eficácia no acompanhamento das demandas. A seguir, são detalhados o fluxo de funcionamento e os perfis de usuários envolvidos.

4.1 Fluxo de Funcionamento

O fluxo de identificação da localidade da ocorrência foi projetado para mitigar erros manuais de digitação por parte da comunidade acadêmica. Na interface da aplicação móvel, o processo ocorre por meio de componentes de seleção hierárquica e encadeada. O usuário interage inicialmente com uma lista suspensa padronizada para determinar o bloco físico do campus (Ex: Bloco B); a partir dessa escolha, o sistema filtra de forma reativa e exibe apenas as salas e laboratórios vinculados àquela estrutura (Ex: Sala 102). Essa abordagem de engenharia de interface simplifica o preenchimento, acelera o tempo de abertura da solicitação e garante a exatidão dos dados geográficos necessários para a triagem da equipe predial sem a necessidade de entrada textual livre no MVP.

A Figura 1 apresenta o fluxo de interação entre o usuário solicitante, o sistema e a equipe técnica, desde a identificação da ocorrência até a conclusão do atendimento.

1. **Identificação do Local:** Ao identificar uma avaria, o usuário navega pelas listas de seleção suspensas. Ao determinar o bloco, o sistema restringe dinamicamente as salas correspondentes, permitindo fixar com exatidão o local da ocorrência (Ex: Bloco B - Sala 102), eliminando erros de digitação e ambiguidades geográficas.

2. **Autenticação e Acesso Local:** Para garantir o controle e a rastreabilidade das solicitações, o sistema exige a autenticação do usuário. No escopo de desenvolvimento do MVP, o acesso foi estruturado por meio de credenciais de login gerenciadas e armazenadas de forma segura diretamente no banco de dados local da aplicação, assegurando a identificação técnica sem a necessidade de integrações externas com os provedores de e-mail institucionais nesta fase.

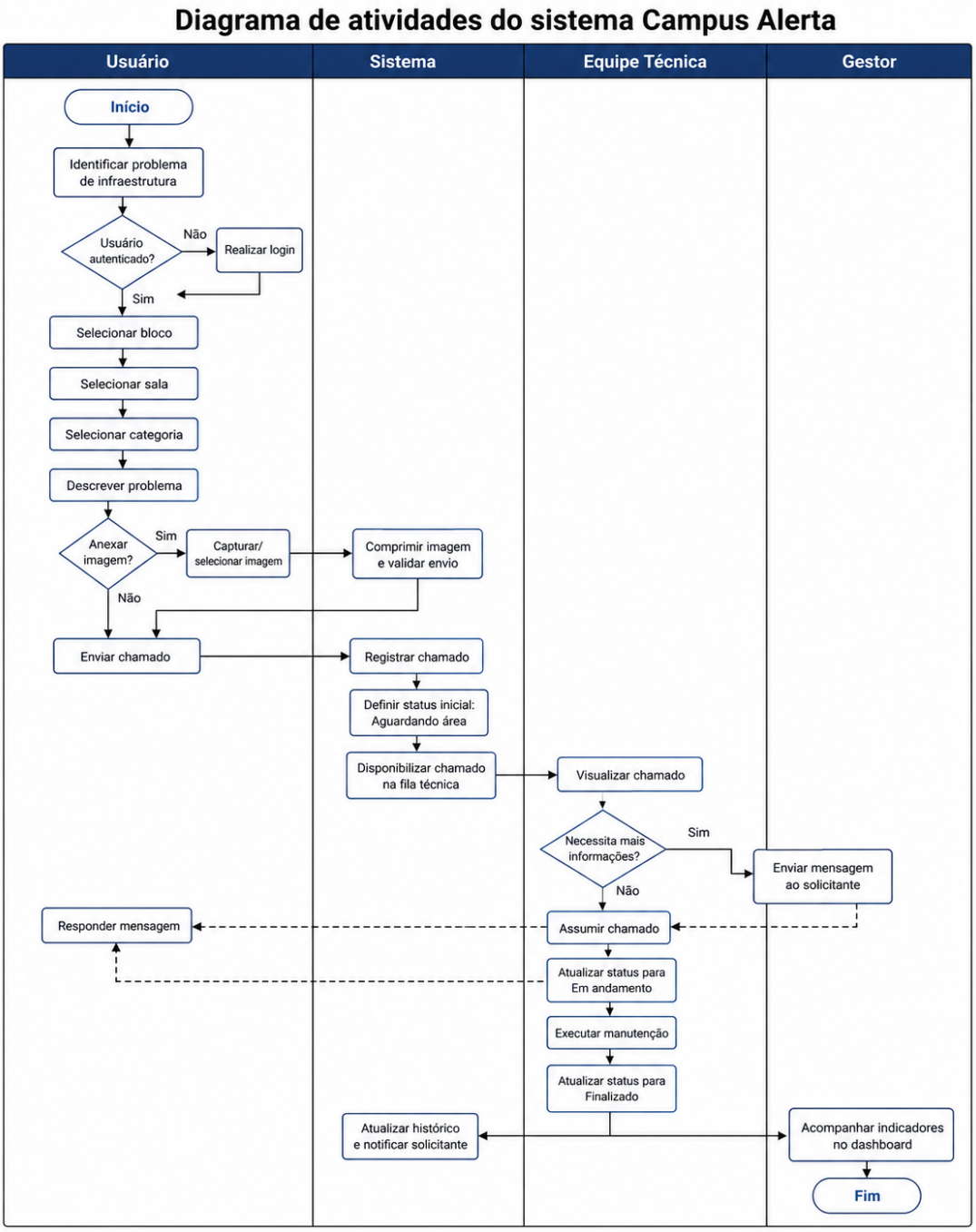

Figura 1 – Diagrama de atividades do sistema Campus Alerta
Fonte: Autoria própria (2026).

3. **Registro da Ocorrência:** Após a definição do local, é apresentado ao usuário um formulário simplificado para categorizar o tipo de problema (ex: elétrico ou hidráulico), inserir um resumo descritivo do dano e anexar fotografias capturadas em campo como evidências visuais.

4. **Gestão e Atendimento:** Os dados preenchidos são enviados para a nuvem da OutSystems e disponibilizados em tempo real na fila de atendimento do painel administrativo. A equipe técnica do departamento realiza a triagem, responde ao solicitante

através do chat integrado e atualiza o estado operacional do chamado (Aguardando área, Em andamento ou Finalizado).

4.2 Perfis de Usuários

Para atender às necessidades de governança e operação do sistema, foram definidos três perfis de acesso distintos, cada um com permissões específicas.

4.2.1 Internos (Alunos e Servidores)

Este é o perfil do solicitante final. Alunos e servidores são os principais responsáveis por identificar e reportar os problemas de infraestrutura encontrados no dia a dia. Para este perfil, a interface foi projetada para ser minimalista e focada na rapidez: seleção do local, descrição do problema, foto e envio. O usuário também pode acompanhar o histórico e o status de suas solicitações, promovendo transparência.

4.2.2 Equipe de Manutenção (Técnicos)

Os técnicos de manutenção recebem as ordens de serviço geradas a partir dos chamados. Eles utilizam o sistema para visualizar a fila de tarefas, consultar detalhes (como a foto e o local exato) e registrar a conclusão do serviço. Para resolver o gargalo de comunicação relatado pelo setor — onde solicitantes não visualizavam as respostas — a plataforma permite que o técnico atualize o status diretamente do local do reparo, ou envie mensagens ao solicitante diretamente no seu aplicativo.

4.2.3 Gestor do Campus (Administrador)

O perfil administrativo é voltado para a Departamento de Segurança e Serviços Gerais (DESEG) ou órgão equivalente. O gestor tem acesso a um painel de controle (*Dashboard*) com indicadores-chave de desempenho (KPIs), tais como: Blocos com mais chamados, salas com mais chamados, média de chamados por dia, entre outros. Esses dados transformam a manutenção corretiva em inteligência para o planejamento de manutenções preventivas e alocação de orçamento.

4.3 Histórias de Usuário e Requisitos Funcionais

Como resultado do levantamento exploratório realizado com um representante do DESEG, descrito na Seção 5.2.2, as necessidades identificadas no fluxo de abertura, triagem e

atendimento das solicitações foram convertidas em Histórias de Usuário (*User Stories*). Essa abordagem permitiu registrar os requisitos sob a perspectiva dos diferentes perfis envolvidos e relacionar cada funcionalidade ao valor operacional esperado.

O Quadro 2 detalha os requisitos essenciais classificados como mandatórios (*Must Have*) no MVP, discriminando o papel, a intenção funcional e a justificativa prática que norteou a engenharia do software.

Quadro 2 – Histórias de Usuário e Requisitos de Negócio (*Must Have*)

Como... (Papel)	**Quero... (Funcionalidade/Desejo)**	**Para... (Justificativa/Valor)**
Usuário Comum	Autenticar-se na plataforma por meio de login e senha cadastrados no banco de dados local	Garantir o acesso controlado e seguro à aplicação, assegurando a rastreabilidade de quem reportou o chamado sem a necessidade de integrações externas no MVP.
Usuário Comum	Selecionar o bloco e a sala exata do incidente através de listas suspensas padronizadas	Garantir que a localização seja precisa e evitar que o usuário digite nomes de salas incorretos ou fora do padrão.
Usuário Comum	Registrar um chamado contendo categoria, descrição do problema e imagem anexada	Centralizar a denúncia do dano predial na fila de atendimento da equipe especializada de manutenção.
Usuário Comum	Acompanhar a evolução e o status vigente dos chamados abertos por mim	Obter transparência operacional e saber exatamente quando o problema reportado será inspecionado.
Usuário Técnico	Visualizar uma fila reativa contendo todos os chamados classificados como pendentes	Realizar a triagem imediata das demandas prioritárias da infraestrutura sem necessidade de planilhas.
Usuário Técnico	Alterar o status do chamado (ex: de Pendente para "Em Andamento"ou "Finalizado")	Informar de forma automatizada ao usuário solicitante a evolução da execução do trabalho predial.
Usuário Técnico & Usuário Comum	Trocar mensagens internas atreladas à tela detalhada de um chamado específico	Sanar dúvidas técnicas e prover retornos diretos sobre as particularidades de uma manutenção em andamento.
Gestor Admin.	Acessar um painel gerencial consolidador contendo dados analíticos e gráficos	Identificar os gargalos e as falhas mais recorrentes para tomadas de decisões preventivas de infraestrutura.

Fonte: Autoria própria (2026).

4.4 Modelagem do Banco de Dados

A modelagem do banco de dados do sistema *Campus Alerta* foi desenvolvida utilizando o banco de dados relacional nativo da plataforma OutSystems (OUTSYSTEMS, 2025), baseado em Microsoft SQL Server. A estrutura foi projetada para garantir a integridade referencial e a escalabilidade da solução, sempre alinhada às funcionalidades vitais do aplicativo.

A Figura 2 apresenta o Diagrama Entidade-Relacionamento (DER) implementado. O núcleo do sistema opera em torno da entidade **Chamado**, que concentra as informações de cada ocorrência e estabelece as pontes com os demais módulos da aplicação.

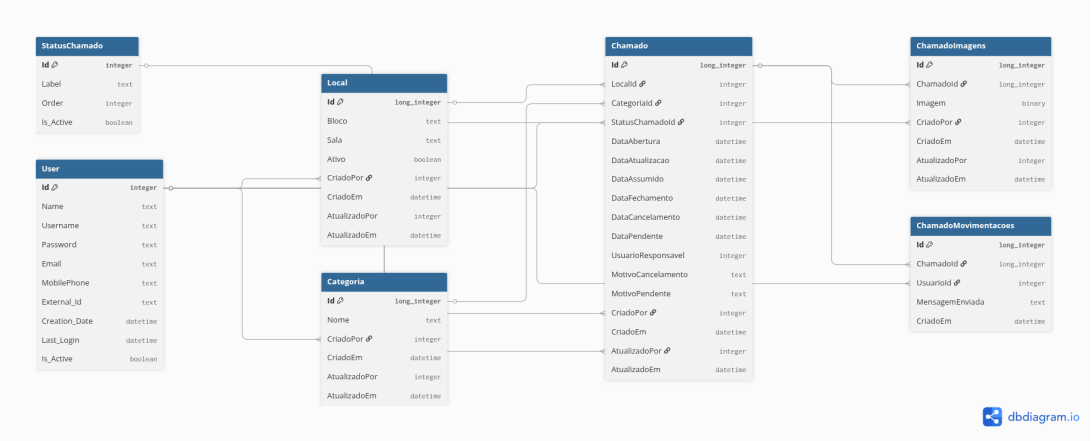

Figura 2 – Diagrama Entidade-Relacionamento do sistema Campus Alerta
Fonte: Autoria própria (2026).

Para viabilizar a funcionalidade primária do sistema — a abertura rápida e precisa de ocorrências sem erros de digitação —, implementou-se a entidade **Local**. O relacionamento de um-para-muitos (1:N) entre Local e Chamado permite que um mesmo espaço (como o "Bloco A - Sala 101") acumule o histórico de várias solicitações ao longo do tempo. Para o escopo deste MVP e visando agilidade na prototipação, os atributos de "Bloco"e "Sala"foram unificados na mesma tabela conceitual. No entanto, do ponto de vista de normalização de dados e usabilidade administrativa, reconhece-se que a abordagem ideal seria a separação dessas entidades (uma tabela **Bloco** relacionando-se 1:N com uma tabela **Sala**). Essa refatoração estrutural será abordada em evoluções futuras, permitindo que, ao cadastrar uma nova sala, o gestor selecione o bloco correspondente de forma isolada, garantindo maior integridade relacional. Simultaneamente, a entidade **Categoria** sustenta o agrupamento de problemas (elétricos, hidráulicos, etc.), sendo crucial para que o *Dashboard* gerencial consiga gerar os gráficos analíticos. Múltiplos chamados pertencem a uma mesma categoria (1:N), permitindo o mapeamento de falhas recorrentes.

O acompanhamento em tempo real, que garante transparência à comunidade acadêmica, é sustentado pela entidade **StatusChamado**. Como a evolução de um serviço passa por etapas rigorosas (Aguardando área, Em Atendimento, Finalizado), diversos chamados podem compartilhar o mesmo status simultaneamente, enquanto cada solicitação reflete apenas seu estado atual no aplicativo.

As *features* de anexação visual e comunicação direta, desenvolvidas para superar as limitações do sistema legado da UTFPR, baseiam-se em duas entidades de apoio atreladas à tabela núcleo. A entidade **ChamadoImagens** permite que o usuário adicione múltiplas fotos a uma única denúncia (1:N). No escopo deste MVP, o armazenamento das imagens foi implementado em formato binário (*Blob*) diretamente no banco de dados nativo, mantendo a tabela

leve. Para versões futuras, planeja-se a adoção de provedores de *Object Storage* externos para otimizar ainda mais o custo e a performance do armazenamento de arquivos estáticos. Por sua vez, a entidade **ChamadoMovimentacoes** viabiliza a funcionalidade de chat na tela de detalhes da ocorrência, armazenando o histórico de mensagens trocadas exclusivamente entre o autor e o técnico responsável, em um relacionamento também de um-para-muitos.

Além dos atributos funcionais evidenciados no Dicionário de Dados (Quadro 3), a governança das informações é reforçada por campos de auditoria automáticos da plataforma (`CriadoPor`, `CriadoEm`), garantindo rastreabilidade sobre qual usuário (entidade **User**) reportou ou resolveu cada problema.

A entidade *User* é fornecida nativamente pela plataforma OutSystems e, por esse motivo, não foi detalhada integralmente no dicionário de dados da aplicação.

Embora a plataforma OutSystems disponibilize mecanismos nativos para aplicações multitenant, tais recursos não foram configurados nem validados no escopo deste MVP. A modelagem implementada atende ao contexto de uma única instituição. A adaptação para múltiplos órgãos exigiria a inclusão de mecanismos de identificação institucional, isolamento lógico dos registros e controle de acesso específico, permanecendo como possibilidade de evolução futura.

Quadro 3 – Dicionário de dados do sistema Campus Alerta

Entidade	Atributo	Tipo	Descrição
Local	Id	Long Integer	Chave primária da entidade.
	Bloco	Text	Identificação do bloco físico.
	Sala	Text	Identificação da sala ou laboratório.
	Ativo	Boolean	Indica se o local está ativo no sistema.
Categoria	Id	Long Integer	Chave primária da entidade.
	Nome	Text	Nome da categoria da ocorrência.
StatusChamado	Id	Integer	Chave primária da entidade.
	Label	Text	Nome do status do chamado.
	Ordem	Integer	Ordem de exibição do status.
	Is_Active	Boolean	Indica se o status está disponível.
Chamado	Id	Long Integer	Chave primária da ocorrência.
	LocalId	Local Identifier	Referência ao local da ocorrência.
	CategoriaId	Categoria Identifier	Referência à categoria do problema.
	StatusChamadoId	StatusChamado Identifier	Referência ao status atual.
	Descricao	Text	Descrição detalhada da ocorrência.
	DataAbertura	Date Time	Data e hora de abertura do chamado.
	DataAtualizacao	Date Time	Data e hora da última atualização.
	DataFechamento	Date Time	Data e hora de conclusão do chamado.
	UsuarioResponsavel	User Identifier	Usuário responsável pelo atendimento.
ChamadoImagens	Id	Long Integer	Chave primária da imagem.
	ChamadoId	Chamado Identifier	Referência ao chamado associado.
	Imagem	Binary Data	Arquivo de imagem anexado.
Movimentacoes	Id	Long Integer	Chave primária da movimentação.
	ChamadoId	Chamado Identifier	Referência ao chamado associado.
	UsuarioId	User Identifier	Usuário responsável pelo envio da mensagem.
	MensagemEnviada	Text	Conteúdo da mensagem registrada.
	CriadoEm	Date Time	Data e hora do envio da mensagem.

Fonte: Autoria própria (2026).

5 MATERIAIS E MÉTODOS

Este capítulo descreve os recursos tecnológicos adotados e os procedimentos metodológicos utilizados na construção e na verificação funcional da plataforma Campus Alerta.

5.1 Materiais

Para a implementação da solução *Campus Alerta*, foi selecionado um conjunto de tecnologias focado em Desenvolvimento Rápido de Aplicação, do inglês *Rapid Application Development* (RAD), garantindo a entrega do MVP dentro do cronograma acadêmico.

5.1.1 Plataforma de Desenvolvimento: OutSystems

A principal ferramenta utilizada neste projeto é a plataforma **OutSystems** (OUTSYSTEMS, 2025) (através do ambiente de desenvolvimento *Service Studio*). Trata-se de uma plataforma de desenvolvimento *Low-Code* líder de mercado, que permite a construção visual de aplicações corporativas.

A justificativa para esta escolha é estratégica e acadêmica:

- **Agilidade:** O desenvolvimento visual reduz drasticamente a quantidade de código manual (boilerplate), permitindo que o foco do trabalho seja a regra de negócio (gestão de manutenção) e a experiência do usuário, e não a configuração de servidores ou APIs básicas.
- **Capacidade PWA:** A plataforma oferece recursos nativos para a geração de PWA. Isso garante que o sistema funcione como um aplicativo nativo em dispositivos móveis (Android e iOS) dos alunos e servidores, com acesso à câmera (para tirar fotos nos chamados) e armazenamento local, sem a barreira de entrada que seria exigir o download em uma loja de aplicativos (Google for Developers, 2025). Além disso, o uso de Low-Code viabiliza a entrega de um sistema complexo (com Painel Web, App Mobile e Banco de Dados) por um único desenvolvedor dentro do cronograma acadêmico.

5.1.2 Infraestrutura de Dados

O armazenamento e gerenciamento dos dados foram realizados utilizando o Sistema Gerenciador de Banco de Dados (SGBD) relacional integrado à nuvem da OutSystems (baseado em SQL Server). Esta arquitetura *cloud-native* garante integridade referencial, segurança automática e, principalmente, escalabilidade, suportando futuras evoluções da solução, caso seja necessária sua adaptação para múltiplas instituições.

5.2 Métodos

A metodologia de trabalho adotada é de natureza aplicada e exploratória. O processo de desenvolvimento seguiu os ritos da metodologia ágil **Scrum** (SCHWABER; SUTHERLAND, 2020), adaptados para a execução individual do Trabalho de Conclusão de Curso.

5.2.1 Estudo e Adaptação Tecnológica

Considerando que a plataforma OutSystems e o paradigma *Low-Code* não compõem a grade curricular tradicional do curso de Tecnologia em Sistemas para Internet, a primeira etapa metodológica consiste no domínio da ferramenta. Foram realizados estudos técnicos sobre a lógica de fluxos visuais, arquitetura de módulos e ciclo de vida de aplicações na plataforma.

5.2.2 Levantamento, Engenharia de Requisitos e Priorização

O levantamento de requisitos foi conduzido por meio de reuniões de caráter exploratório com um representante do Departamento de Segurança e Serviços Gerais (DESEG) do campus Guarapuava da UTFPR. A consulta teve como finalidade compreender o fluxo atualmente utilizado para o recebimento, a triagem, o encaminhamento e o acompanhamento das solicitações relacionadas à infraestrutura do campus.

Durante as reuniões, o participante descreveu que as demandas institucionais são distribuídas entre diferentes áreas, principalmente o DESEG, o Departamento de Projetos e Obras (DEPRO) e a Coordenação de Gestão de Tecnologia da Informação (COGET), de acordo com a natureza da ocorrência. Também foi relatado que os registros formais são consultados e triados por meio do sistema GLPI, enquanto parte da comunicação operacional com a equipe de manutenção ocorre por canais paralelos, como grupos de mensagens instantâneas.

De acordo com o fluxo relatado, após identificar uma nova solicitação no sistema institucional, o responsável pela triagem encaminha as informações relevantes à equipe executora, frequentemente por meio de captura de tela ou mensagem em grupo. Após a realização do serviço, os profissionais responsáveis retornam informações ou evidências fotográficas sobre a conclusão da atividade. Embora esse procedimento permita o encaminhamento das demandas, a utilização de diferentes canais dificulta a centralização do histórico, o acompanhamento do andamento dos serviços e a geração de indicadores gerenciais.

A partir desse levantamento, foram identificadas como necessidades principais: a centralização dos chamados em uma única interface; a classificação da ocorrência conforme a área institucional responsável; a definição de perfis distintos para solicitantes, técnicos e gestores; o registro do histórico de alterações; a comunicação direta entre o solicitante e a equipe responsável; e a geração de indicadores para acompanhamento das demandas.

As necessidades levantadas foram convertidas em histórias de usuário e requisitos funcionais, apresentados na Seção 4.3. Posteriormente, os requisitos foram classificados utilizando a técnica MoSCoW (SEBRAE, 2024), priorizando-se no MVP as funcionalidades indispensáveis para a abertura, a triagem, o acompanhamento e a conclusão dos chamados. Funcionalidades associadas à integração institucional, à distribuição automatizada entre setores e à geração de indicadores avançados foram mantidas como possibilidades de evolução da solução.

5.2.3 Ciclo de Desenvolvimento

O projeto foi dividido em ciclos quinzenais (*Sprints*), estruturados da seguinte forma:

1. **Planejamento e Modelagem:** Definição das *user stories* da quinzena e modelagem das entidades no banco de dados da OutSystems.
2. **Implementação (Front e Back):** Construção das telas e da lógica de negócios utilizando os aceleradores visuais da plataforma.
3. **Testes Funcionais e Homologação:** Após cada Sprint foram realizados testes internos para verificar a consistência das regras de negócio, persistência dos dados, navegação entre telas e funcionamento dos diferentes perfis de usuário.

5.2.4 Procedimentos de Validação e Verificação

A validação do Produto Mínimo Viável (MVP) desenvolvido neste trabalho concentrou-se na verificação funcional e técnica da plataforma, garantindo que os requisitos estruturados na etapa de concepção fossem atendidos de ponta a ponta. Os testes foram realizados internamente no ambiente de desenvolvimento provido pela plataforma OutSystems, englobando as seguintes frentes de homologação:

- **Validação dos Fluxos de Solicitação:** Foram simulados diversos cenários de abertura de chamados para verificar a consistência da interface móvel. Testou-se a integridade das listas de seleção hierárquica (filtragem de salas com base no bloco selecionado), a compressão e o *upload* de evidências fotográficas, e a gravação correta no banco de dados relacional.
- **Validação das Regras de Negócio e Painel Técnico:** Consistiu na verificação prática do ciclo de vida das ocorrências. Foram realizados testes de alteração de *status* (de "Pendente"para "Em Andamento"e "Finalizado") e testes do funcionamento do chat bidirecional, assegurando que as *Client Actions* e *Server Actions* refletissem as atualizações nas telas.

- **Validação do *Dashboard* Administrativo:** A eficácia das lógicas de consulta (*Aggregates*) foi atestada por meio da inserção de uma massa de dados fictícios controlados. O objetivo foi confirmar se a consolidação das métricas — como locais com mais defeitos e categorias recorrentes — alimentava corretamente os componentes gráficos do painel de gestão.

A validação empírica, que compreende a aplicação de testes de usabilidade e métricas de satisfação (como a escala SUS) com grupos reais de usuários (alunos, servidores e técnicos de infraestrutura), foi delimitada como escopo para etapas posteriores, configurando-se como uma das proposições para trabalhos futuros.

6 RESULTADOS E DISCUSSÃO

6.1 Detalhamento dos Sprints e Evolução do Projeto

O ciclo de desenvolvimento prático do *Campus Alerta* seguiu a divisão em Sprints quinzenais, totalizando um período iterativo estruturado. A seguir, detalha-se a evolução do ecossistema de software e os respectivos gargalos tecnológicos superados durante a curva de aprendizado da ferramenta:

- **Sprint 1 - Modelagem Base e Autenticação Local:** Concentrou-se no desenho do modelo lógico relacional de banco de dados diretamente no ambiente OutSystems e na configuração da governança de acesso local. O principal desafio técnico residiu em estruturar a entidade nativa de usuários (**User**) fornecida pela plataforma para mapear e isolar corretamente os diferentes perfis operacionais (usuários comuns, técnicos de manutenção e administradores gestores), garantindo o controle de permissões interno sem depender de provedores de identidade externos no escopo do MVP.
- **Sprint 2 - Fluxo Mobile First de Abertura e Seleção de Locais:** Compreendeu o desenvolvimento reativo das telas do aplicativo PWA para smartphones, focando no escopo de seleção hierárquica de locais (filtrando salas com base no bloco selecionado) e anexo fotográfico. O desafio dessa etapa envolveu o tratamento de armazenamento de arquivos binários pesados de imagens diretamente em memória móvel. Para sanar o gargalo, implementou-se um fluxo lógico de compressão de mídia antes do envio para o servidor remoto, além de isolar as imagens na tabela **ChamadoImagens**.
- **Sprint 3 - Painel Técnico e Chat Reativo:** Dedicado à estruturação da fila de chamados dos profissionais de facilities e o chat bidirecional. A plataforma OutSystems permitiu contornar essa barreira através do uso de variáveis de tela reativas atreladas a consultas assíncronas atualizadas por eventos (*Data Refresh*).
- **Sprint 4 - Dashboard Gerencial e Consolidação:** Fechamento do ecossistema com a agregação de dados operacionais e geração de indicadores de desempenho de manutenção predial no painel administrativo.

6.2 Apresentação do Sistema Implementado

Com a estrutura de dados consolidada, a interface do *Campus Alerta* foi desenvolvida sob o paradigma *Mobile First*, garantindo que a principal funcionalidade do sistema — a abertura rápida de chamados via *smartphone* — ocorra sem atritos. A seguir, são apresentadas as principais telas da aplicação e suas respectivas regras de negócio.

6.2.1 Autenticação e Cadastro

A Figura 3 apresenta a tela de autenticação do sistema. Para garantir a rastreabilidade e evitar o registro de ocorrências falsas (*spam*), o usuário deve realizar o login utilizando suas credenciais previamente cadastradas e validadas no banco de dados nativo do sistema, gerenciado internamente pelo administrador da plataforma, antes de acessar as funcionalidades.

A Figura 4 apresenta a tela de cadastro local. O próprio usuário informa seus dados e cria uma conta, que recebe inicialmente o perfil de solicitante interno. Perfis técnicos e administrativos são atribuídos posteriormente pelo administrador da aplicação. O cadastro não utiliza, nesta versão, validação automática por meio dos sistemas institucionais da UTFPR.

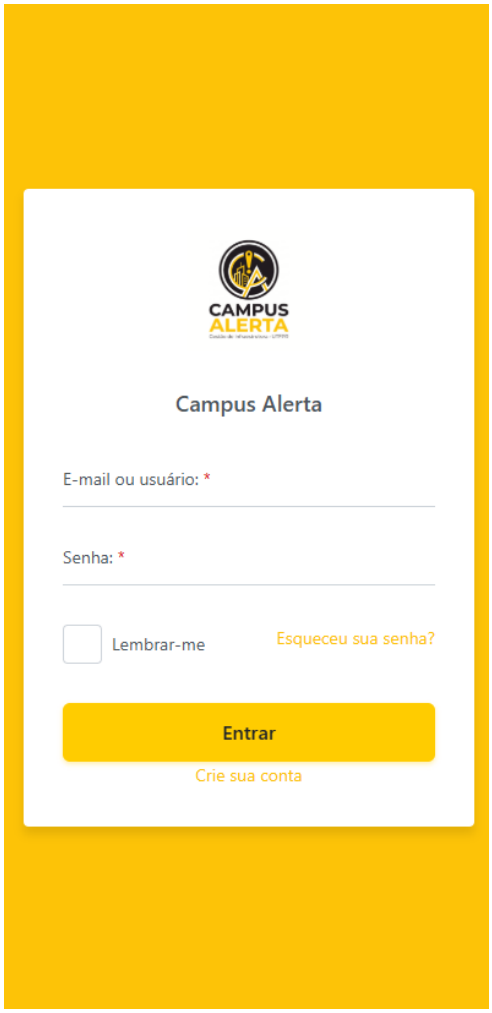

Figura 3 – Tela de Login e Autenticação do Usuário
Fonte: Autoria própria (2026).

6.2.2 Abertura de Chamados

A interface mais crítica do sistema é a de abertura de chamados, ilustrada na Figura 5. O usuário precisa selecionar o bloco e sala onde se encontra, para a seguir selecionar a categoria

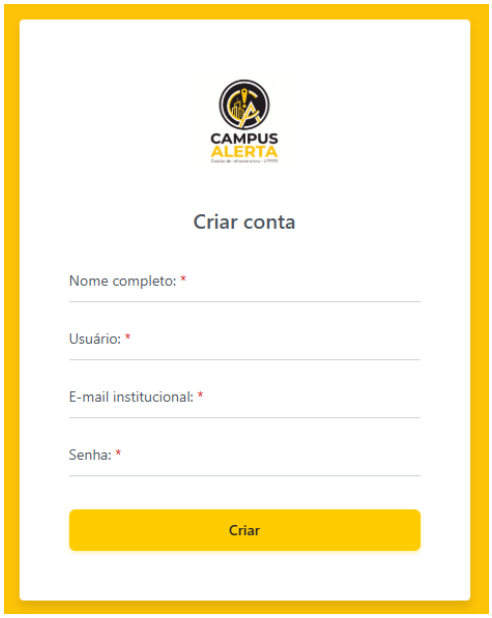

Figura 4 – Tela de Cadastramento de Usuário
Fonte: Autoria própria (2026).

do problema, adicionar uma breve descrição e anexar uma fotografia da avaria. Este fluxo reduz o tempo de notificação e elimina erros de digitação sobre o local do incidente.

Após a seleção do bloco, a interface exibe dinamicamente o campo de seleção da sala correspondente, não visível no estado inicial apresentado na Figura 5.

6.2.3 Atendimento e Comunicação

Para a equipe de manutenção, o sistema fornece uma visão focada na resolução das demandas. A Figura 6 exibe a primeira aba da tela, onde é possível visualizar a lista de chamados pendentes, os detalhes da ocorrência (incluindo as fotos enviadas pelo relatante) e os controles para atualizar o status do serviço (por exemplo, de "Aguardando área"para "Em Andamento"ou "Finalizado"). Já na Figura 7 é possível visualizar a interação com o usuário solicitante.

6.2.4 Dashboard Gerencial

Por fim, a Figura 8 ilustra o Painel de Gestão (*Dashboard*) voltado para a administração do campus. Esta tela consolida os dados operacionais, apresentando métricas como o volume de chamados por categoria, blocos e salas com mais ocorrências, além de um gráfico de linhas mostrando a quantidade de chamados abertos diariamente no mês atual. Essas informações transformam dados brutos em inteligência, permitindo que a gestão tome decisões estratégicas sobre manutenção preventiva e alocação de recursos.

Não foram elaborados protótipos formais em uma ferramenta externa. As interfaces foram construídas e refinadas iterativamente diretamente no ambiente OutSystems.

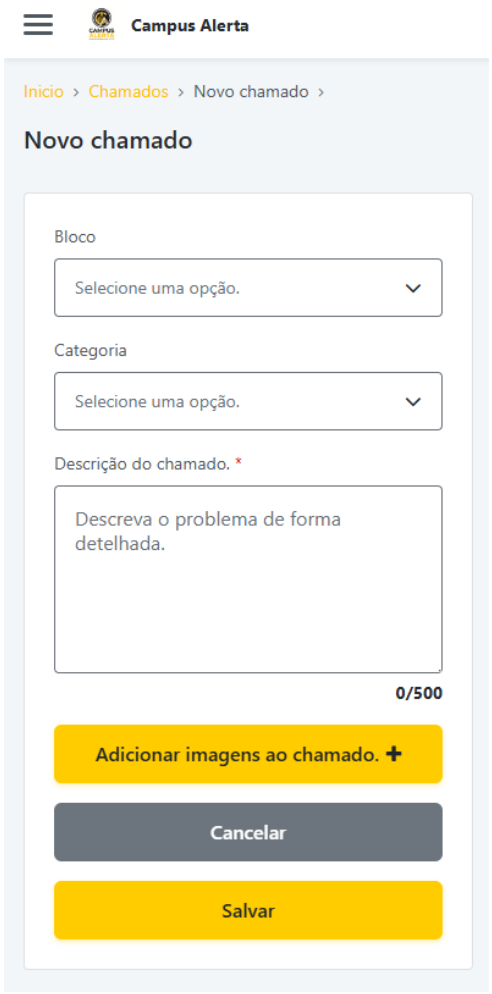

Figura 5 – Tela de Abertura de Chamado
Fonte: Autoria própria (2026).

6.2.5 Decisões Técnicas e Lógicas Implementadas

Apesar de as plataformas de desenvolvimento Low-Code abstraírem a escrita manual de linhas de código textuais, o processo de desenvolvimento exige rigor na construção da lógica de programação e na otimização estrutural das consultas em banco de dados. No ecossistema OutSystems, as regras de negócio complexas são arquitetadas visualmente por meio de *Client Actions* e *Server Actions*.

6.2.6 Armazenamento de Imagens

Durante o ciclo de desenvolvimento, a implementação foi dividida em três frentes principais: a estruturação do núcleo de dados, a criação do fluxo de *Mobile First* para abertura de chamados, e a construção do *Dashboard* gerencial. Um dos principais desafios técnicos enfrentados foi a gestão do desempenho do aplicativo ao lidar com evidências fotográficas anexadas pelos usuários. Inicialmente, cogitou-se armazenar as imagens diretamente na tabela princi-

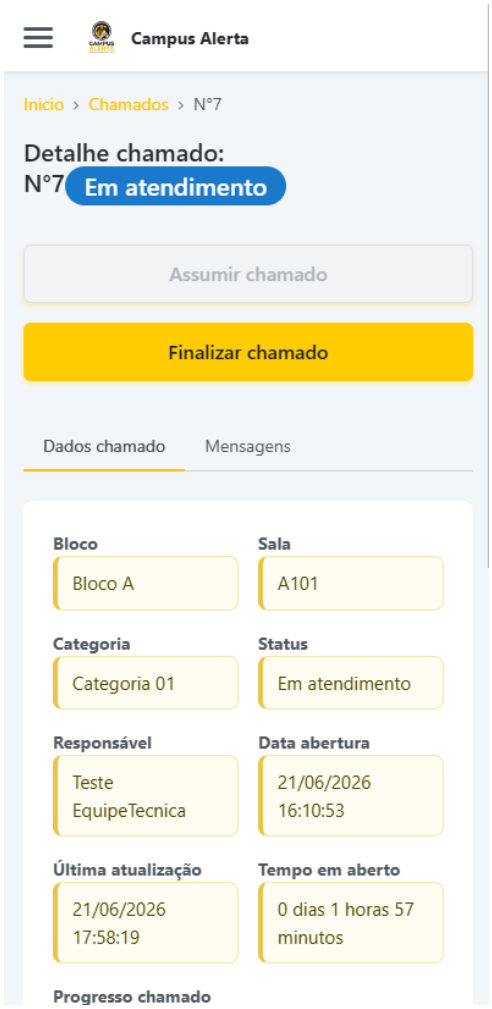

Figura 6 – Visão do Técnico para gestão e atualização de status do chamado
Fonte: Autoria própria (2026).

pal de chamados. Contudo, percebeu-se que essa abordagem tornaria as consultas pesadas e lentas. A solução adotada para o MVP foi o isolamento binário das imagens em uma entidade separada (*Blob* no banco de dados relacional), com a compressão prévia da mídia no dispositivo móvel do usuário. Essa estratégia garantiu consultas rápidas na listagem de chamados e o carregamento sob demanda apenas quando o usuário acessa os detalhes da ocorrência. Vale ressaltar que, para uma evolução futura e escalabilidade do sistema em um ambiente de produção em larga escala, a alternativa arquitetural ideal seria a migração desse armazenamento para um serviço de *Cloud Object Storage* (como Amazon S3 ou Google Cloud Storage), armazenando-se no banco de dados apenas as URLs de referência das imagens.

6.2.7 Comunicação e Consultas Visuais

Outro desafio relevante foi a modelagem da funcionalidade de mensagens bidirecionais (chat) entre os solicitantes e a equipe técnica, essencial para resolver a principal dor relatada pela universidade: a falta de retorno na comunicação. Foi necessário criar uma lógica de fluxo

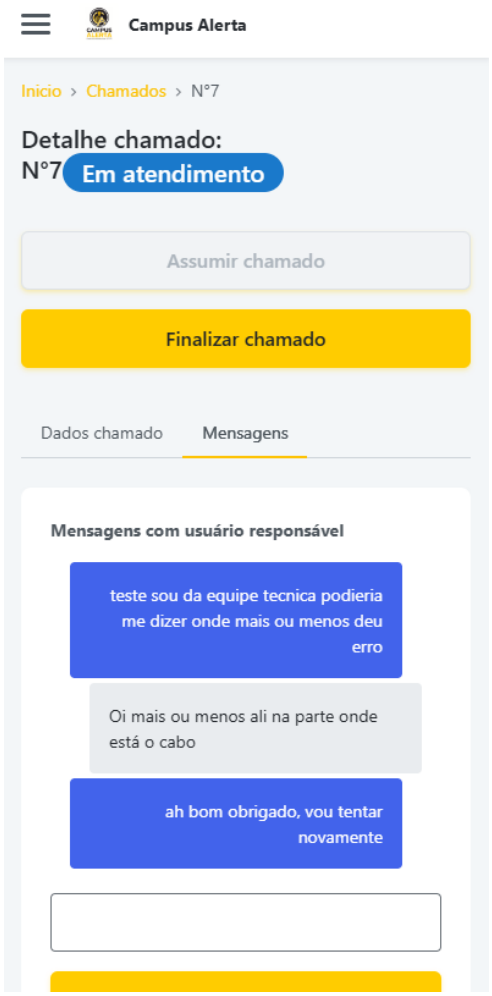

Figura 7 – Visão do Técnico para gestão e troca de mensagens com solicitante
Fonte: Autoria própria (2026).

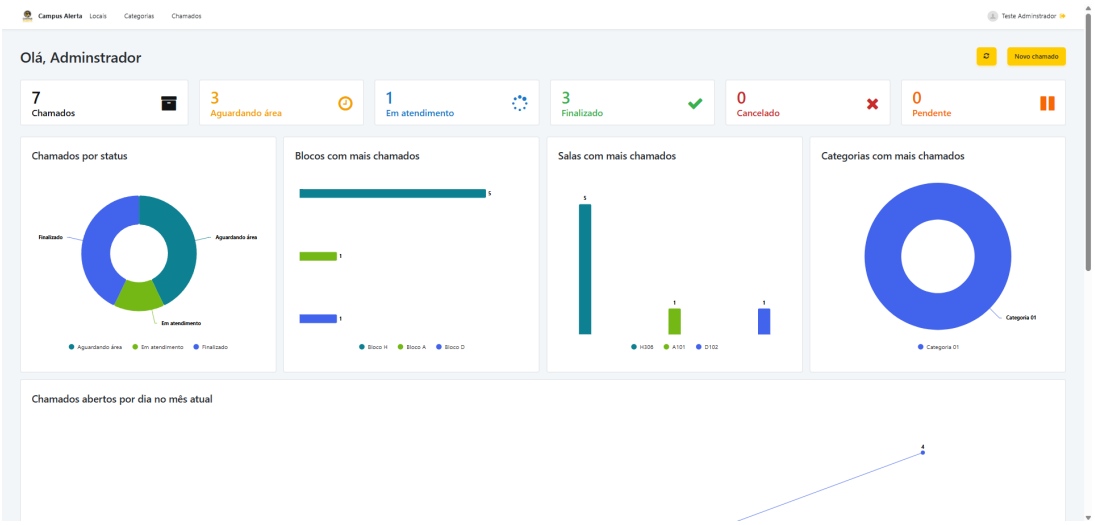

Figura 8 – Dashboard gerencial com indicadores de manutenção
Fonte: Autoria própria (2026).

que isolasse essas mensagens por chamado e as atualizasse mediante nova consulta dos da-

dos na tela do PWA, exigindo uma integração cuidadosa entre a interface do técnico (que atualiza o status) e o aplicativo do usuário (que recebe a notificação da conversa).

A Figura 9 ilustra a lógica construída no *Service Studio* para a manutenção e sincronização do chat bidirecional dos dados entre o solicitante e o técnico responsável (`ChamadoMovimentacoes`).

Complementarmente, a otimização de busca nas tabelas do sistema foi viabilizada pelo uso de *Aggregates*, estruturas especializadas em abstrair e otimizar sentenças relacionais de *SQL Joins*. A Figura 11 exibe o mapeamento estrutural construído para alimentar o *Dashboard* gerencial, unindo as tabelas de **Chamado**, **Local**, **Categoria** e **User** através de filtros indexados por data.

Essa modelagem de consultas visuais impediu a degradação de performance da aplicação PWA ao realizar paginações na fila técnica de atendimento predial.

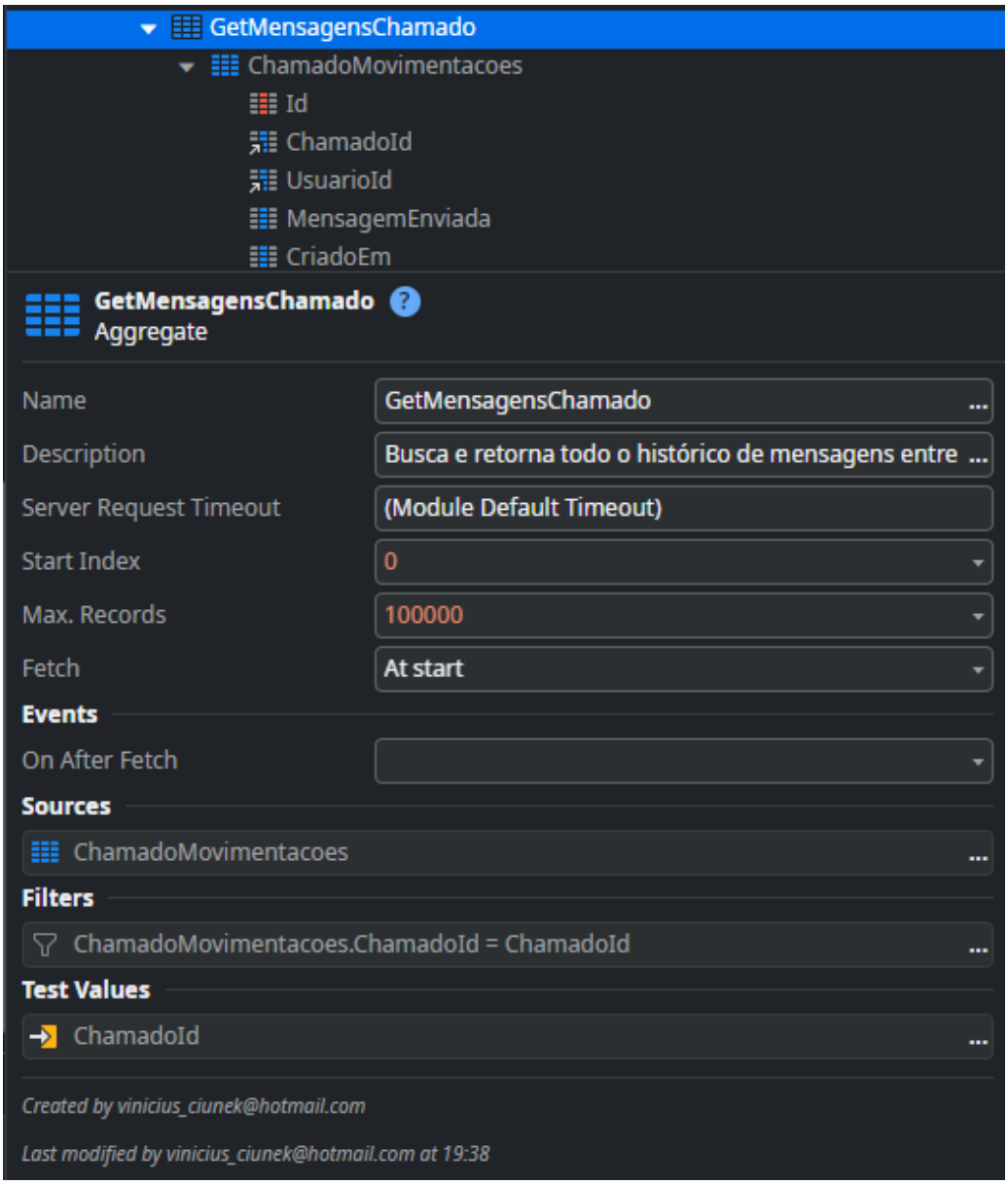

Figura 9 – Aggregate com dados das mensagens do chamado

Fonte: Autoria própria (2026).

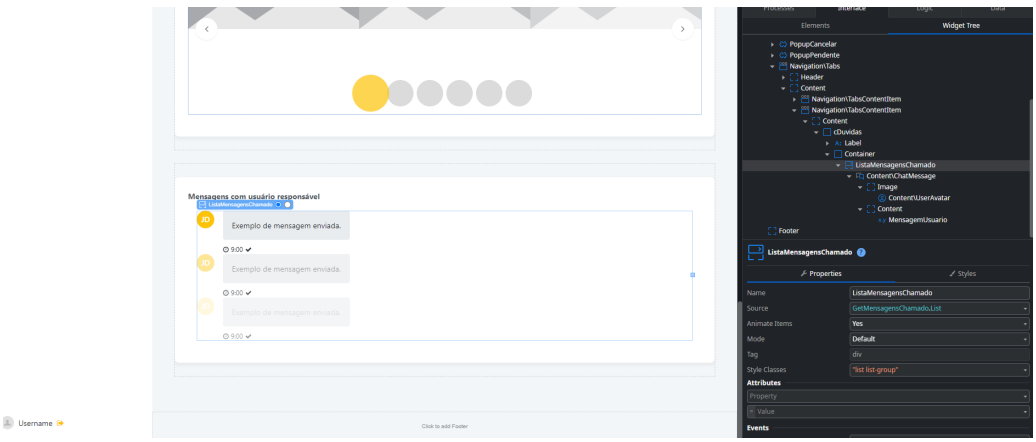

Figura 10 – Fluxo lógico visual para envio de mensagens e atualização da interface

Fonte: Autoria própria (2026).

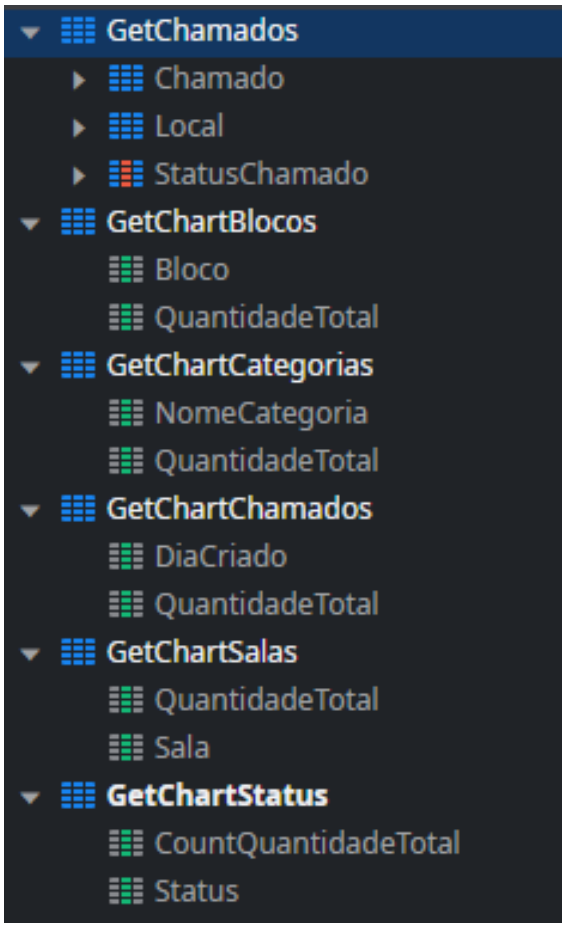

Figura 11 – Aggregates utilizados na consolidação dos indicadores

Fonte: Autoria própria (2026).

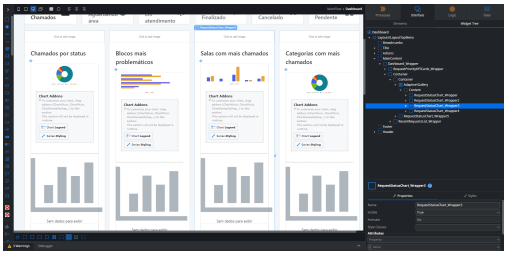

Figura 12 – Estrutura da tela gerencial no OutSystems

Fonte: Autoria própria (2026).

6.3 Verificação Funcional

A verificação funcional do MVP foi realizada internamente pelo desenvolvedor, utilizando uma massa de dados controlada. Foram avaliados os fluxos essenciais de autenticação, seleção hierárquica de locais, abertura de chamados, alteração de status, troca de mensagens e consolidação dos indicadores gerenciais.

O Quadro 4 apresenta os cenários verificados, os resultados esperados e os resultados observados durante a execução dos testes.

Quadro 4 – Resultados da verificação funcional do MVP

ID	Cenário verificado	Resultado esperado	Resultado observado
TF01	Autenticação com credenciais válidas	Permitir o acesso e carregar as funcionalidades correspondentes ao perfil	O acesso foi autorizado e as permissões do perfil foram aplicadas corretamente.
TF02	Seleção hierárquica do local	Exibir apenas as salas associadas ao bloco selecionado	A lista de salas foi filtrada conforme o bloco selecionado.
TF03	Abertura de chamado com imagem	Registrar o chamado, comprimir a imagem e persistir os dados	O chamado e a imagem comprimida foram registrados nas entidades correspondentes.
TF04	Alteração do estado do chamado	Atualizar o estado de “Aguardando área” para “Em andamento” e posteriormente para “Finalizado”	As alterações foram persistidas e exibidas nas telas do solicitante e do técnico.
TF05	Troca de mensagens no chamado	Registrar e exibir mensagens associadas exclusivamente ao chamado selecionado	As mensagens foram armazenadas e apresentadas no histórico do chamado correspondente.
TF06	Consolidação do painel gerencial	Exibir indicadores de acordo com a massa de dados controlada	Os gráficos apresentaram os valores correspondentes aos registros inseridos para teste.

Fonte: Autoria própria (2026).

Os testes internos indicaram que os fluxos essenciais do MVP foram executados conforme os resultados esperados. A verificação utilizou uma massa de dados controlada e foi realizada pelo próprio desenvolvedor, não constituindo teste de usabilidade, avaliação de desempenho em larga escala ou homologação institucional.

6.4 Avaliação Qualitativa com o DESEG

O MVP foi apresentado a um representante do DESEG, participante que também colaborou no levantamento exploratório dos requisitos. A demonstração teve caráter qualitativo

e buscou verificar se os fluxos implementados correspondiam às necessidades relatadas pelo setor. Não foram realizados testes formais de usabilidade nem uma avaliação institucional com representantes de todos os departamentos envolvidos.

A partir dessa consulta técnica com o representante do DESEG, realizou-se um mapeamento exploratório do ecossistema de manutenção. Conforme relatado pelo participante, as demandas de infraestrutura do campus são distribuídas entre três principais frentes de atuação administrativa:

- **DESEG (Serviços Gerais):** Responsável pelo gerenciamento de reparos cotidianos, limpeza, conservação predial e segurança física das instalações;
- **DEPRO (Departamento de Projetos e Obras):** Setor encarregado do planejamento, fiscalização e execução de reformas estruturais e novas obras no campus;
- **COGET (Coordenação de Tecnologia da Informação):** Setor especializado no suporte técnico, redes e manutenção de ativos de informática e laboratórios de tecnologia.

7 CONSIDERAÇÕES FINAIS

Este trabalho apresentou o desenvolvimento e a implementação da plataforma *Campus Alerta*, uma solução de engenharia de software voltada à otimização da gestão de manutenção predial no campus da UTFPR. Ao longo da pesquisa, foi diagnosticado o problema da dispersão e informalidade no reporte de falhas de infraestrutura, propondo-se uma abordagem colaborativa baseada em tecnologia Low-Code para sanar essas deficiências.

A adoção da plataforma OutSystems confirmou-se como uma escolha metodológica acertada. O uso do paradigma de desenvolvimento visual permitiu a construção ágil e centralizada de um ecossistema complexo — composto por banco de dados em nuvem, regras de negócio e interfaces para múltiplos perfis — por um único desenvolvedor dentro das restrições de tempo do calendário acadêmico.

No escopo delimitado para este trabalho, foram atendidos os objetivos relacionados ao levantamento do fluxo, à definição dos requisitos, ao desenvolvimento do MVP e à verificação técnica das funcionalidades implementadas.

- **Módulo de Abertura Responsivo:** Disponibilização de um fluxo intuitivo para alunos e servidores reportarem ocorrências prediais em tempo real, com suporte a anexo fotográfico e seleção padronizada de locais, eliminando ambiguidades e centralizando a comunicação.

- **Fila de Atendimento Técnico e Comunicação Direta:** Criação de uma interface *mobile-first* para a equipe de manutenção predial realizar a triagem, atualizar os estados dos chamados diretamente em campo e trocar mensagens diretas com o solicitante, solucionando a principal falha de retorno de informação.

- **Dashboard Gerencial Analítico:** Implementação de um painel de controle administrativo composto por gráficos agregados para suporte à visualização de indicadores calculados a partir dos registros armazenados no sistema.

Os objetivos relacionados ao levantamento do fluxo, à definição dos requisitos, à implementação do MVP e à verificação técnica das funcionalidades foram atendidos no escopo estabelecido. A avaliação empírica com usuários reais não foi realizada nesta etapa, de modo que não é possível concluir sobre usabilidade, satisfação, desempenho em escala ou adoção institucional.

Em suma, o *Campus Alerta* demonstrou na prática como ferramentas digitais integradas conseguem transformar solicitações avulsas e descentralizadas em dados técnicos acionáveis, agilizando processos burocráticos e estabelecendo uma base sólida para a evolução rumo ao conceito de *Smart Campus* na instituição.

Dessa forma, considera-se que o objetivo geral do trabalho foi alcançado, uma vez que foi desenvolvida e implementada uma plataforma funcional para gestão colaborativa de manutenção predial, contemplando os requisitos definidos durante a fase de levantamento e análise.

Por fim, conclui-se que o uso da plataforma OutSystems foi vital para a prototipação ágil e a validação do modelo de negócio em tempo hábil para o calendário acadêmico. Contudo, reconhece-se que a adoção deste sistema em larga escala no setor público esbarra em desafios de governança de TI, como o custo de licenciamento de soluções proprietárias. O *Campus Alerta* cumpre seu papel como uma Prova de Conceito funcional, fornecendo à universidade um projeto lógico documentado e tecnicamente verificado que pode ser traduzido no futuro para tecnologias de código aberto, unindo eficiência operacional e viabilidade econômica.

7.1 Trabalhos Futuros

Como desdobramento desta pesquisa e visando a evolução contínua da plataforma desenvolvida, recomendam-se os seguintes pontos para expansões futuras do sistema:

- **Migração para Arquitetura de Código Aberto (*Open Source*):** Visando a adoção definitiva por instituições públicas (como a UTFPR, escolas estaduais e prefeituras), propõe-se a reescrita do aplicativo utilizando tecnologias de código aberto (ex: React Native, Flutter, Node.js, PostgreSQL). Esta migração é essencial para evitar o *vendor lock-in* gerado por plataformas proprietárias *Low-Code* e eliminar os altos custos recorrentes de licenciamento em nuvem, garantindo a sustentabilidade financeira e a independência tecnológica do projeto no setor público.

- **Avaliação de Usabilidade com Usuários Reais:** Realizar testes de campo com alunos, servidores, técnicos de manutenção e gestores, mensurando o tempo necessário para execução das tarefas, a taxa de sucesso dos fluxos e a percepção de usabilidade por meio de instrumento padronizado, como a escala SUS.

- **Avaliação Institucional e Piloto Controlado:** Apresentar formalmente a Prova de Conceito aos setores responsáveis e executar um piloto controlado com um conjunto restrito de locais e usuários antes de qualquer decisão de implantação.

- **Integração Institucional de Autenticação:** Implementar o protocolo de autenticação unificada (LDAP/Single Sign-On) integrado diretamente aos sistemas corporativos da UTFPR (como o Sistema Acadêmico).

- **Mecanismos de Geolocalização (GPS):** Evoluir o módulo de localização atual (baseado em listas de seleção manuais) integrando a API de GPS do smartphone para identificar o bloco mais próximo do usuário, sugerindo o local automaticamente para agilizar ainda mais a abertura do chamado.

- **Sistema de Notificações *Push* Ativas:** Acoplar um serviço de mensageria nativa para alertar o *smartphone* do solicitante de forma ativa a cada atualização realizada pela equipe técnica, eliminando falhas de comunicação.
- **Triagem Inteligente via Aprendizado de Máquina (IA):** Incorporar modelos de inteligência artificial na nuvem para analisar a descrição textual e a imagem enviada, automatizando a classificação do nível de urgência do chamado técnico.
- **Métricas calculadas por IA:** Unir a triagem automática do sistema a cálculos feitos por inteligência artificial para trazer *insights* mais apurados e detalhados para os usuários administradores em relação a problemas recorrentes e sugestões de resolução efetiva.
- **Integração com o Campus Virtual:** Conectar a aplicação ao projeto já existente do Campus Virtual da instituição, permitindo ao usuário selecionar um bloco interagindo diretamente com a maquete 3D do campus, promovendo uma experiência de uso altamente imersiva.

REFERÊNCIAS

ATLASSIAN. **Jira Service Management**. 2025. https://www.atlassian.com/software/jira/service-management. Acesso em: 11 set. 2025.

BIøRN-HANSEN, A.; MAJCHRZAK, T. A.; GRøNLI, T.-M. Progressive web apps for the unified development of mobile applications. *In*: ____. [*S.l.*: *s.n.*], 2018. p. 64–86. ISBN 978-3-319-93526-3.

GLPI Project. **GLPI Open Source ITSM and Asset Management**. 2025. https://glpi-project.org. Acesso em: 21 jun. 2026.

Google for Developers. **What are Progressive Web Apps?** 2025. https://web.dev/progressive-web-apps/. Acesso em: 11 set. 2025.

LOMAS, L.; JONES, R. The practice of facility management in higher education. **New Directions for Institutional Research**, v. 2006, n. 131, p. 5–15, 2006.

MIN-ALLAH, N.; ALRASHED, S. Smart campus—a sketch. **Sustainable Cities and Society**, v. 59, 2020.

OUTSYSTEMS. **High-Performance Low-Code Platform**. 2025. https://www.outsystems.com/. Acesso em: 18 nov. 2025.

OutSystems. **OutSystems and Copper River Family of Companies Achieve Authority to Operate at U.S. Department of Health and Human Services**. 2025. Disponível em: https://www.outsystems.com/news/outsystems-copper-river-ato-hhs.

SAHAY, A. *et al.* Supporting the understanding and comparison of low-code development platforms. *In*: **2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)**. [*S.l.*: *s.n.*], 2020. p. 171–178.

SCHWABER, K.; SUTHERLAND, J. **The Scrum Guide: The Definitive Guide to Scrum**. 2020. https://scrumguides.org/scrum-guide.html. Acesso em: 19 nov. 2025.

SEBRAE. **O método MoSCoW para definição de prioridades**. 2024. https://sebrae.com.br/Sebrae/Portal%20Sebrae/Arquivos/ebook_sebrae_metodologia_moscow.pdf. Acesso em: 19 nov. 2025.

SHETTY, A. A. e S. **Survey Toward a Smart Campus Using the Internet of Things**. 2016. https://www.researchgate.net/publication/308673465_Survey_Toward_a_Smart_Campus_Using_the_Internet_of_Things. Acesso em: 21 jun. 2026.

WASZKOWSKI, R. Low-code platform for automating business processes in manufacturing. **IFAC-PapersOnLine**, v. 52, n. 10, p. 376–381, 2019.

ZAMMAD. **Zammad: The Helpdesk & Support System**. 2025. https://zammad.org/. Acesso em: 11 set. 2025.