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: Prof. Dr. Luciano Ogiboski

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 englobam a modelagem da arquitetura de dados com suporte a múltiplas instituições (multitenancy), a prototipação de alta fidelidade e a implementação dos fluxos funcionais do sistema. Concluiu-se que a plataforma oferece uma alternativa viável, robusta e de baixo custo de implantação, entregando uma interface democrática para a comunidade acadêmica e um painel de triagem centralizado para o órgão gestor, promovendo na prática os pilares de um Smart Campus.

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 light of this, this work developed the "Campus Alerta"system, a digital platform for the collaborative and efficient management of building maintenance tickets. 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 the registration and ensure the organization of demands, the identification of the problem's location was structured to allow users to report occurrences simply and directly, by selecting the institution's buildings and rooms from standardized lists. The applied and exploratory methodology used the Scrum agile framework for iterative development and the MoSCoW technique for requirements prioritization. The achieved results encompass the modeling of the data architecture with support for multiple institutions (multitenancy), high-fidelity prototyping, and the implementation of the system's functional flows. It was concluded that the platform offers a viable, robust, and low-cost implementation alternative, delivering a democratic interface for the academic community and a centralized screening dashboard for the managing body, practically promoting the pillars of a Smart Campus.

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

LISTA DE FIGURAS

LISTA DE QUADROS

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)
LCDP	Plataforma de Desenvolvimento Low-Code, do inglês *Low-Code Development Platform*
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

PARTE I

INTRODUÇÃO

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 projetada para ser genérica e escalável, visando sua futura expansão para atender a outras instituições públicas da cidade, como escolas estaduais e postos de saúde, promovendo uma gestão colaborativa da infraestrutura pública.

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, embora existam mecanismos institucionais para a gestão de ocorrências, estes apresentam limitações críticas. Segundo levantamento prático junto ao setor responsável Departamento de Segurança e Serviços Gerais (DESEG), a maior dificuldade cotidiana não é apenas a localização, mas sim o retorno de comunicação com o solicitante. Muitas vezes, o usuário não visualiza as respostas no sistema legado, forçando a equipe técnica a realizar

contatos manuais via e-mail ou telefone para obter maiores informações. Além disso, o sistema vigente encontra-se restrito a perfis específicos, excluindo a maior parte da comunidade acadêmica — os alunos — do processo de monitoramento e reporte.

Essa 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 utilizando tecnologia *Low-Code* (OutSystems) para a gestão colaborativa de chamados de manutenção predial, validando o conceito na UTFPR com uma arquitetura preparada para escalabilidade interinstitucional.

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.
- **Definir** um modelo de autenticação permita o acesso institucional controlado, visando a expansão futura para o conceito de *Smart City*.
- **Validar** o protótipo junto à comunidade acadêmica, coletando métricas para a evolução do produto.

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 escolha pelo desenvolvimento utilizando a plataforma **OutSystems** foi estratégica e acadêmica. Em um mercado de TI que exige cada vez mais velocidade (Time-to-Market), o domínio de plataformas *Low-Code* torna-se um diferencial competitivo. Este projeto justificou-se, portanto, pela investigação dessa tecnologia emergente, avaliando seus ganhos de produtividade em comparação ao desenvolvimento tradicional (High-Code).

O projeto possui também uma forte justificativa social e de extensão. Ao projetar a arquitetura para ser *multitenant* (multi-instituição), 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 capítulos que refletem o desenvolvimento do projeto. O **Capítulo 1** (esta introdução) contextualiza o problema e define os objetivos. O **Capítulo 2** apresenta a Fundamentação Teórica e o Referencial Teórico necessário para o embasamento do tema. O **Capítulo 3** aborda os Trabalhos Relacionados, mapeando soluções similares na literatura. O **Capítulo 4** descreve o Detalhamento da Solução Proposta. O **Capítulo 5** detalha a Metodologia e os Materiais adotados no desenvolvimento. O **Capítulo 6** expõe os Resultados obtidos, incluindo a modelagem do banco de dados e a apresentação do sistema funcional. Por fim, o **Capítulo 7** traz as Considerações Finais 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 (MINALLAH; 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. Pesquisas de Silva *et al.* (2022) demonstram que a descentralização de pedidos de manutenção através de sistemas móveis reduz o tempo de resposta das equipes técnicas em até 40%, justificando a adoção de plataformas digitais acessíveis a toda a comunidade acadêmica.

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 em até 10 vezes, sendo extremamente útil no setor público, onde a escassez de desenvolvedores é um desafio.

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.

As plataformas Low-Code vêm sendo amplamente adotadas por organizações públicas e privadas devido à redução do tempo de desenvolvimento e à facilidade de manutenção. Essas características tornam a abordagem especialmente relevante para projetos acadêmicos e instituições com equipes reduzidas de desenvolvimento.

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. A primeira seção aborda os conceitos de Gestão de Facilidades, Smart Campus e Desenvolvimento Low-Code, enquanto a segunda se aprofunda na análise de sistemas de gestão de chamados já consolidados.

3.1 Contextualização Teórica

Para compreender a relevância e o impacto potencial do Campus Alerta, é fundamental inseri-lo em contextos mais amplos de gestão e inovação tecnológica.

3.1.1 Gestão de Facilidades e Smart Campus

A Gestão de Facilidades (*Facility Management*) é uma disciplina focada em garantir a funcionalidade, o conforto, a segurança e a eficiência do ambiente construído, integrando pessoas, lugares, processos e tecnologias (International Facility Management Association (IFMA), 2025). Em uma instituição como a UTFPR, isso se traduz na administração de toda a infraestrutura física — desde a manutenção de salas e laboratórios até a gestão de recursos. O Campus Alerta se posiciona como uma ferramenta de apoio a essa gestão, otimizando o fluxo de detecção e correção de falhas.

Paralelamente, o conceito de *Smart Campus* descreve a aplicação de tecnologias digitais para criar um ecossistema universitário conectado e eficiente (Gartner IT Glossary, 2025). Ao transformar um processo manual em um fluxo de dados digital, o sistema permite que a gestão deixe de ser apenas reativa e passe a ser baseada em dados, identificando padrões de desgaste na infraestrutura.

3.1.2 Desenvolvimento Low-Code

Diferente das abordagens tradicionais de desenvolvimento de software (conhecidas como *High-Code*), este projeto adota o paradigma *Low-Code*. Plataforma de Desenvolvimento Low-Code, do inglês *Low-Code Development Platform* (LCDP) permitem a criação de aplicações através de interfaces gráficas e configurações, reduzindo drasticamente a necessidade de codificação manual linha a linha.

Segundo pesquisas recentes da área de Engenharia de Software, o uso de Low-Code pode acelerar o ciclo de entrega de software em até 10 vezes, permitindo que desenvolvedores foquem mais na regra de negócio e na experiência do usuário do que na infraestrutura de

código. A escolha da plataforma **OutSystems** para este projeto visa validar essa agilidade no contexto de soluções para o setor público, onde a escassez de desenvolvedores especialistas é um gargalo frequente.

3.2 Análise de Soluções Existentes

Embora não se tenha conhecimento de uma solução desenvolvida especificamente para a gestão de infraestrutura na UTFPR que utilize georreferenciação, existem no mercado ferramentas consolidadas para gestão de chamados. A análise a seguir posiciona o Campus Alerta em relação a algumas dessas soluções.

3.2.1 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.

3.2.2 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.

3.2.3 GLPI

Outra solução amplamente utilizada é o GLPI, plataforma open source voltada ao gerenciamento de chamados e ativos tecnológicos (GLPI Project, 2025).

Embora possua recursos robustos para gerenciamento de serviços, sua interface é voltada principalmente para processos administrativos e suporte de TI, exigindo adaptações para cenários de manutenção predial e uso intensivo em dispositivos móveis.

3.2.4 Síntese Comparativa

A principal lacuna deixada pelas soluções de mercado analisadas é a falta de integração entre o mundo físico e o digital e a devolutiva dos usuários criadores dos chamados, principal dor que a UTFPR possui no momento. Enquanto Jira e Zammad focam no gerenciamento do ticket após sua criação, o **Campus Alerta** inova na etapa de **captura da ocorrência** e uma interface simplificada em Low-Code para garantir que o dado chegue à equipe de manutenção de forma rápida, precisa e georreferenciada.

4 DETALHAMENTO 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.

O diferencial técnico e funcional da solução divide-se em uma principal vertente complementar: o estabelecimento de uma plataforma de comunicação bidirecional. 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 funcionamento do sistema baseia-se na interação contextualizada com o local da ocorrência. O fluxo principal foi desenhado para minimizar o escorço cognitivo do usuário e garantir a precisão dos dados coletados:

1. **Identificação do Local:** Ao identificar um problema, o usuário seleciona o local em que se encontra o sistema. O sistema decodifica o identificador e preenche automaticamente o local da ocorrência (Ex: Bloco B - Sala 102), eliminando erros de digitação e ambiguidades.

2. **Autenticação e Acesso:** Para garantir a segurança e a rastreabilidade, o sistema foi estruturado para exigir autenticação. No escopo do MVP, foram implementados dois modelos de acesso:

    - *Integração Institucional:* Cadastro utilizando as credenciais da UTFPR (RA ou e-mail institucional), garantindo que apenas membros da comunidade acadêmica possam abrir chamados internos.

3. **Registro da Ocorrência:** Após a identificação do local, o usuário apresenta um formulário simplificado para categorizar o problema e anexar uma foto como evidência.

4. **Gestão e Atendimento:** Os dados são enviados para a nuvem da OutSystems e disponibilizados em um painel administrativo para a equipe de manutenção, que realiza a triagem, a priorização e a atualização do status do chamado (Aguardando área, Em andamento, 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 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 parte do levantamento de requisitos utilizando as diretrizes do framework Scrum, as necessidades funcionais mapeadas junto aos stakeholders do campus Guarapuava foram convertidas em Histórias de Usuário (*User Stories*). Esta abordagem permite documentar os requisitos com foco no valor de negócio gerado para cada persona ativa na plataforma.

O Quadro 1 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 1 – Histórias de Usuário e Requisitos de Negócio (*Must Have*)

Como... (Papel)	**Quero... (Funcionalidade/Desejo)**	**Para... (Justificativa/Valor)**
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).

5 MATERIAIS E MÉTODOS

Este capítulo descreve as ferramentas tecnológicas e os procedimentos metodológicos que foram adotados para a execução do projeto. A escolha dos materiais e métodos fundamenta-se na necessidade de alta produtividade, robustez e na natureza aplicada do trabalho, visando a validação de uma solução inovadora para a gestão de infraestrutura.

Este capítulo descreve os recursos tecnológicos adotados e os procedimentos metodológicos seguidos para a construção e validação da plataforma.

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 a modelagem de entidades complexas necessárias para a visão de futuro do projeto (múltiplas instituições utilizando a mesma base).

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 Engenharia de Requisitos e Priorização

Para definir o escopo do MVP, utiliza-se a técnica de priorização **MoSCoW** (SEBRAE, 2024) (*Must have, Should have, Could have, Won't have*). Esta técnica permite classificar as funcionalidades de acordo com sua criticidade para o negócio, garantindo que o núcleo do sistema — a abertura de chamado e o tratamento — seja entregue prioritariamente, enquanto funcionalidades secundárias são planejadas para etapas futuras.

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 de Campo e Feedback:** Validação fisíca da abertura de chamados e testes da plataforma em todos os perfis do sistema.

5.2.4 Procedimentos de Validação e Métricas

Para garantir a eficácia da solução, a validação do MVP no Trabalho de Conclusão de Curso 2 será realizada através de testes de usabilidade com três grupos distintos de usuários, cobrindo todo o ciclo de vida de um chamado de manutenção:

- **Grupo 1 - Solicitantes (Alunos e Servidores):**
  - **Participantes:** Uma amostra de 3 a 5 voluntários da comunidade acadêmica.
  - **Atividade:** Simulação de abertura de chamados em locais reais do campus, utilizando os recursos de identificação de local.
  - **Métricas:** Tempo médio para registro da ocorrência e taxa de sucesso na conclusão da tarefa sem auxílio externo.
- **Grupo 2 - Equipe Técnica (Manutenção):**
  - **Participantes:** 1 perfil de técnico responsável pelos reparos.
  - **Atividade:** Utilização do **Painel do Técnico** para recebimento, triagem e alteração de status (de "Pendente"para "Finalizado") das solicitações abertas pelo Grupo 1.
  - **Objetivo:** Validar se as informações capturadas (foto, local e descrição) são suficientes para a execução do reparo sem necessidade de deslocamento prévio para averiguação.
- **Grupo 3 - Gestão (Administração do Campus):**
  - **Participantes:** Gestores do setor de infraestrutura DIRSEG.
  - **Atividade:** Análise do **Dashboard Administrativo** populado com os dados gerados durante os testes.
  - **Objetivo:** Confirmar se as métricas e indicadores apresentados (locais com mais defeitos, tipos de problemas recorrentes) atendem às necessidades de tomada de decisão do órgão.

6 RESULTADOS

6.1 Descrição do Desenvolvimento do Trabalho

O processo de implementação do *Campus Alerta* ocorreu de forma iterativa, utilizando os recursos de RAD da plataforma OutSystems. A escolha técnica pela plataforma validou a hipótese de ganho de produtividade, permitindo que a infraestrutura de banco de dados nativa em nuvem e o *front-end* reativo (PWA) fossem desenvolvidos de maneira unificada por um único desenvolvedor dentro do cronograma do projeto.

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 principal de chamados. Contudo, percebeu-se que essa abordagem tornaria as consultas pesadas e lentas. A solução adotada foi o isolamento binário das imagens em uma entidade separada, garantindo consultas rápidas na listagem de chamados e carregamento sob demanda apenas quando o usuário acessa os detalhes da ocorrência.

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 que isolasse essas mensagens por chamado e as atualizasse em tempo real 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).

6.2 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 1 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.

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. 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), mantendo o banco de dados leve. 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 2), 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.

6.3 Apresentação do Sistema Funcional

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.

A Figura 2 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 institucionais antes de acessar as funcionalidades.

Enquanto a Figura 3 apresenta a tela de cadastramento de usuário. Todos os usuários que se cadastram no sistema vem como padrão com o perfil de "Interno", considerando que o administrador do sistema dará as devidas permissões aos usuários, solicitando o RA na hora do cadastramento.

A interface mais crítica do sistema é a de abertura de chamados, ilustrada na Figura 4. O usuário precisa selecionar o bloco e sala onde se encontra, para a seguir selecionar a categoria 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.

Para a equipe de manutenção, o sistema fornece uma visão focada na resolução das demandas. A Figura 5 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 6 é possível visualizar a interação com o usuário solicitante.

Por fim, a Figura 7 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.

6.3.1 Arquitetura de Lógicas Visuais e Engenharia de Consultas

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*.

A Figura 8 ilustra a lógica construída no *Service Studio* para a manutenção e sincronização do chat bidirecional em tempo real 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 10 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.

6.4 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:** Concentrou-se no desenho do modelo lógico relacional de banco de dados diretamente no módulo corporativo do OutSystems e na configuração inicial de governança de acesso. O principal gargalo técnico residiu na integração nativa da entidade **User** com os papéis operacionais e customizados

exigidos pelo sistema de triagem. A solução envolveu o acoplamento de ações de validação nos eventos pós-autenticação (*OnException Handler*).

- **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. O gargalo se concentrou em como notificar ambas as pontas em tempo real sobre novas mensagens sem implementar servidores complexos de *WebSockets*. 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.

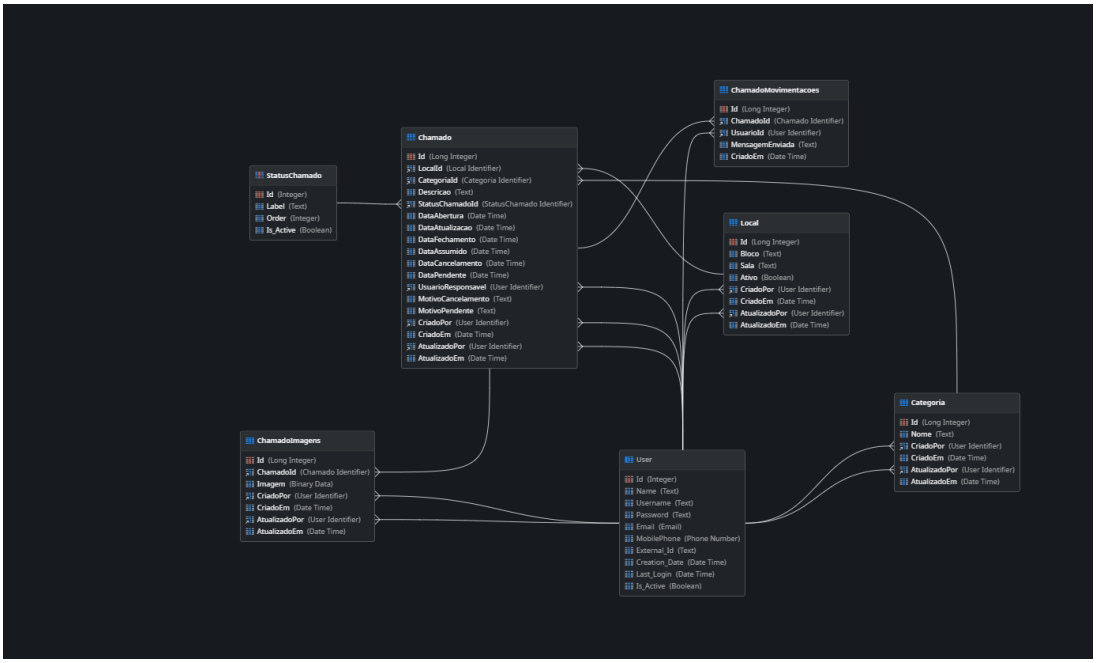

Figura 1 – Diagrama Entidade-Relacionamento do sistema Campus Alerta
Fonte: Autoria própria (2026).

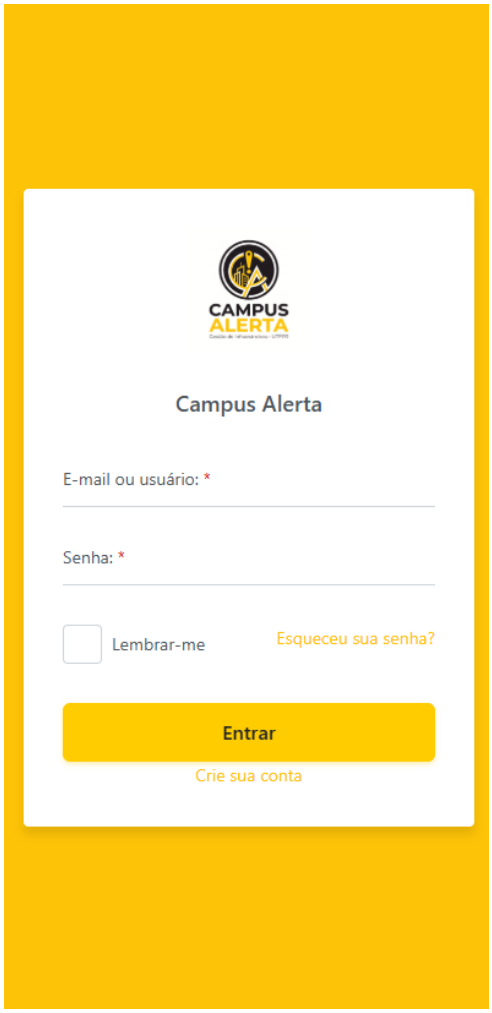

Figura 2 – Tela de Login e Autenticação do Usuário
Fonte: Autoria própria (2026).

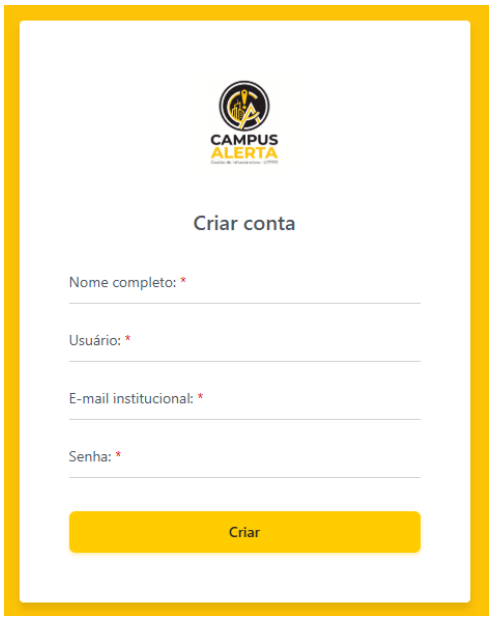

Figura 3 – Tela de Cadastramento de Usuário
Fonte: Autoria própria (2026).

Quadro 2 – 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.

Fonte: Autoria própria (2026).

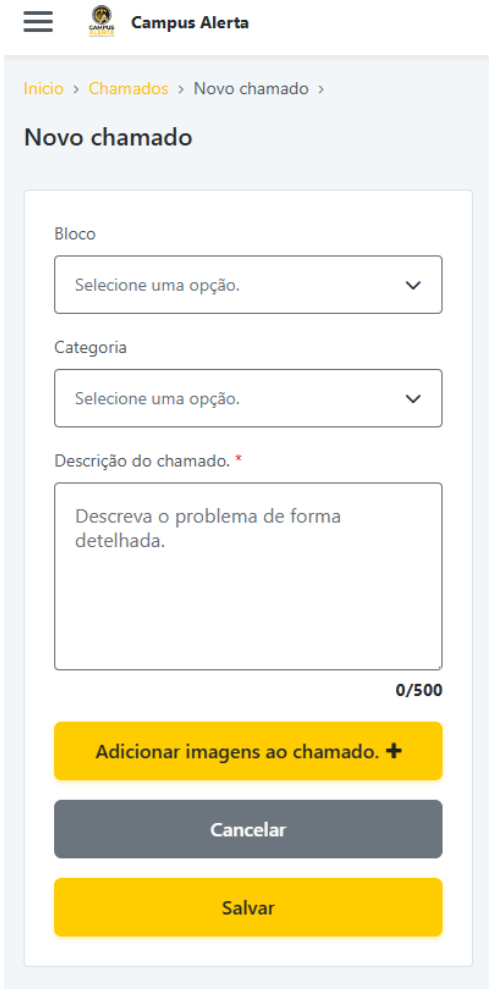

Figura 4 – Tela de Abertura de Chamado

Fonte: Autoria própria (2026).

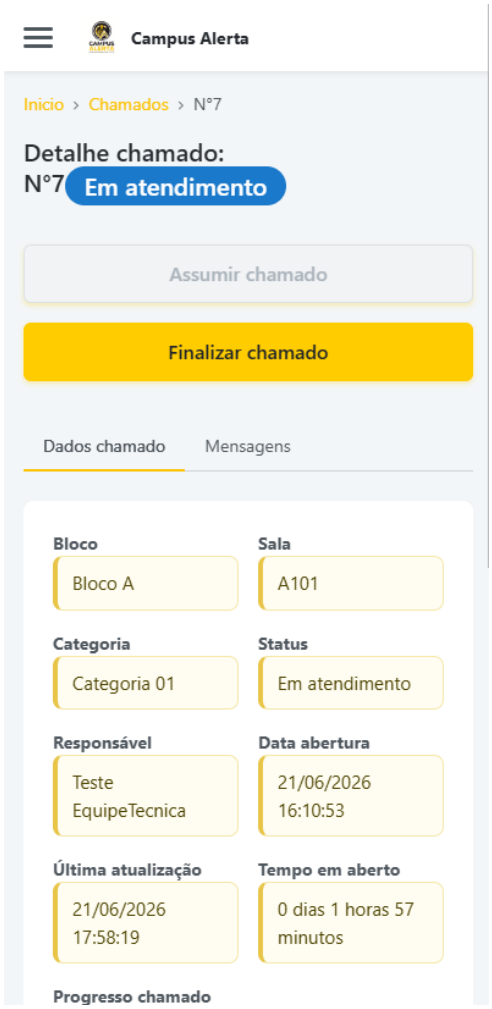

Figura 5 – Visão do Técnico para gestão e atualização de status do chamado

Fonte: Autoria própria (2026).

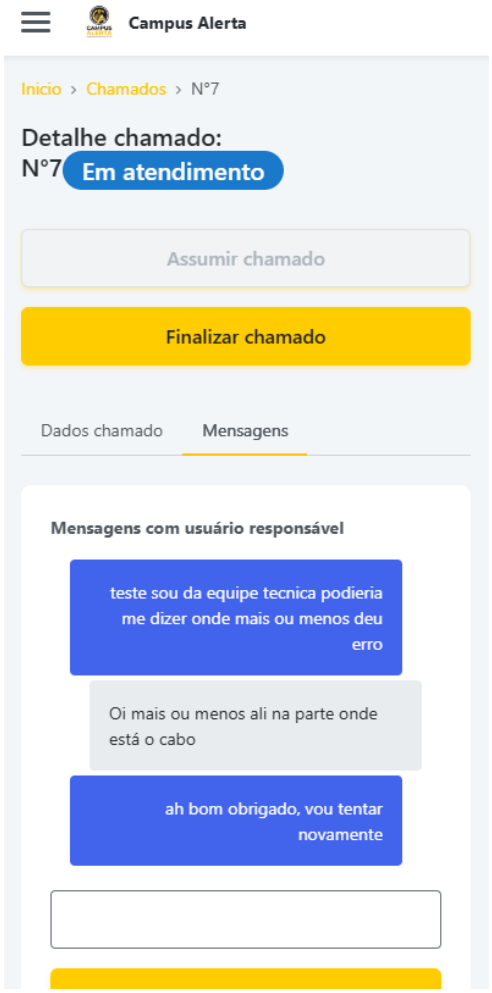

Figura 6 – Visão do Técnico para gestão e troca de mensagens com solicitante
Fonte: Autoria própria (2026).

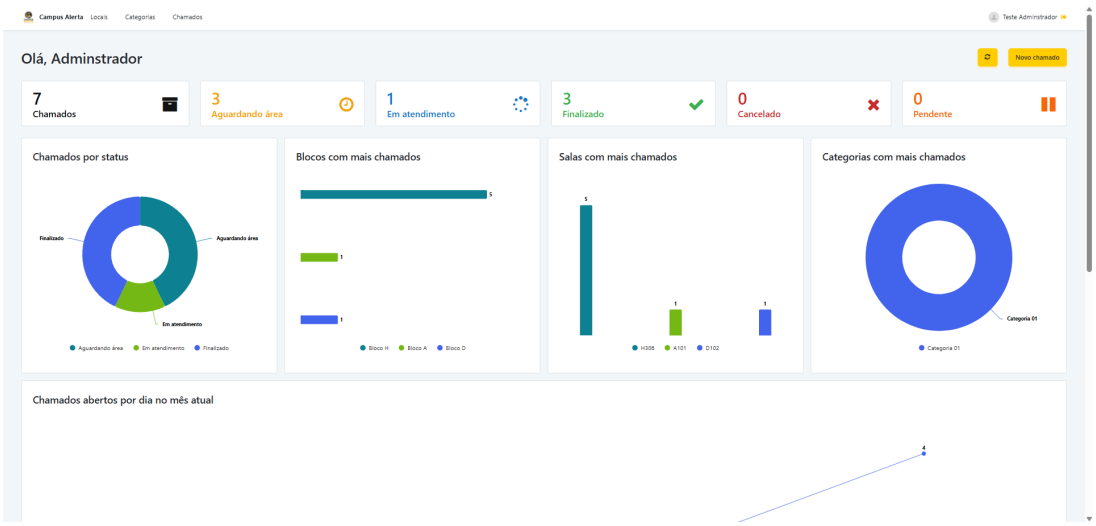

Figura 7 – Dashboard gerencial com indicadores de manutenção
Fonte: Autoria própria (2026).

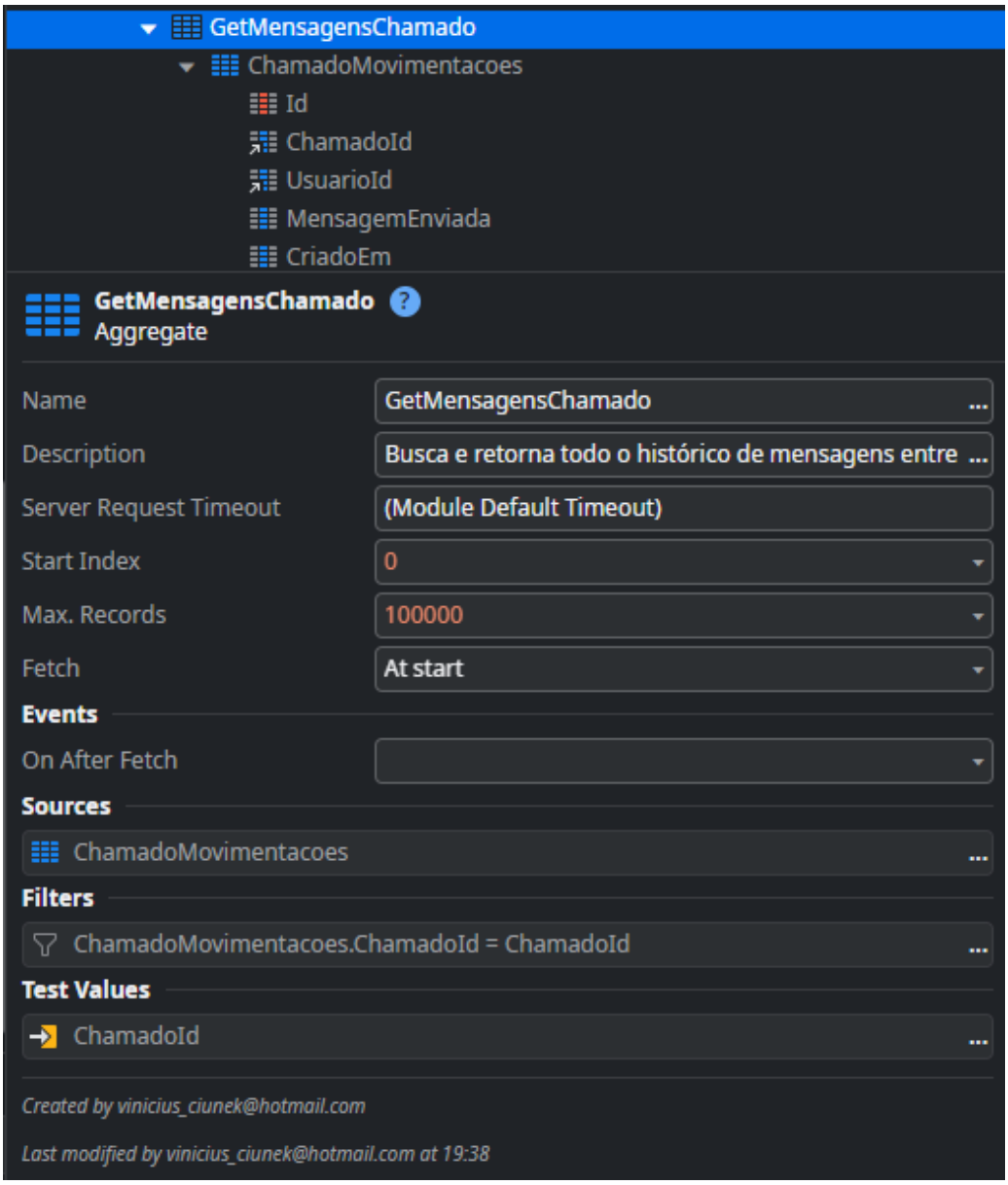

Figura 8 – Aggregate com dados das mensagens do chamado

Fonte: Autoria própria (2026).

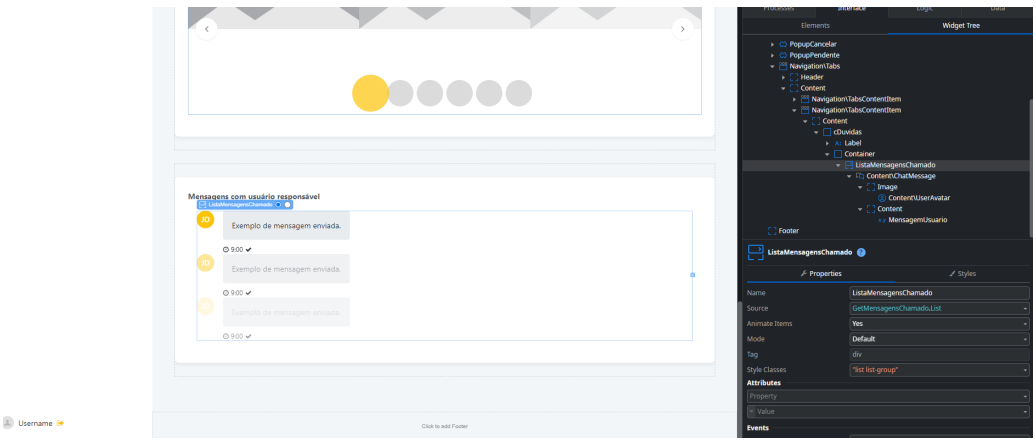

Figura 9 – Fluxo lógico visual para envio de mensagens e atualização de tela no aplicativo 2

Fonte: Autoria própria (2026).

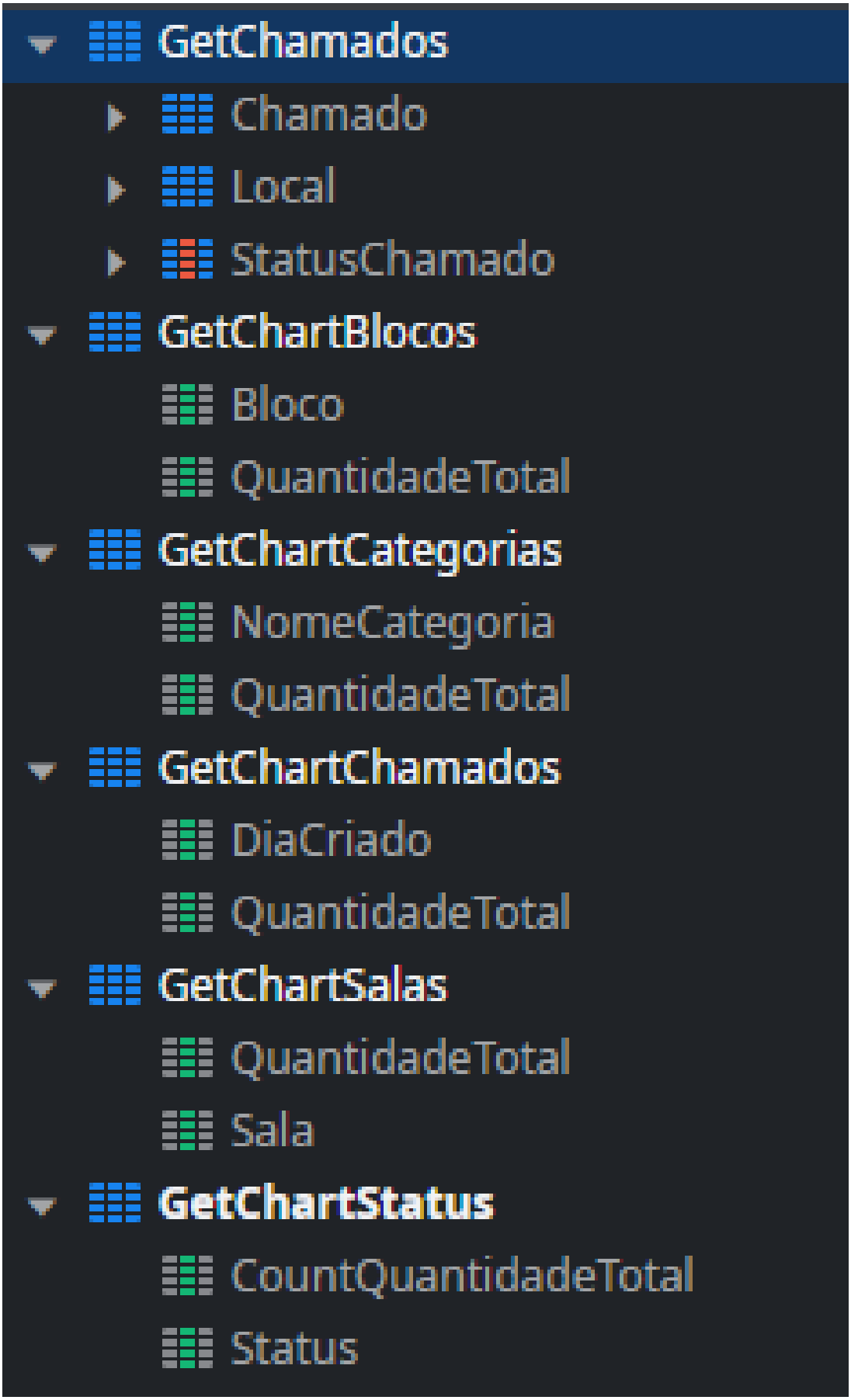

Figura 10 – Estruturação de Aggregate no OutSystems para consolidação de indicadores gerenciais

Fonte: Autoria própria (2026).

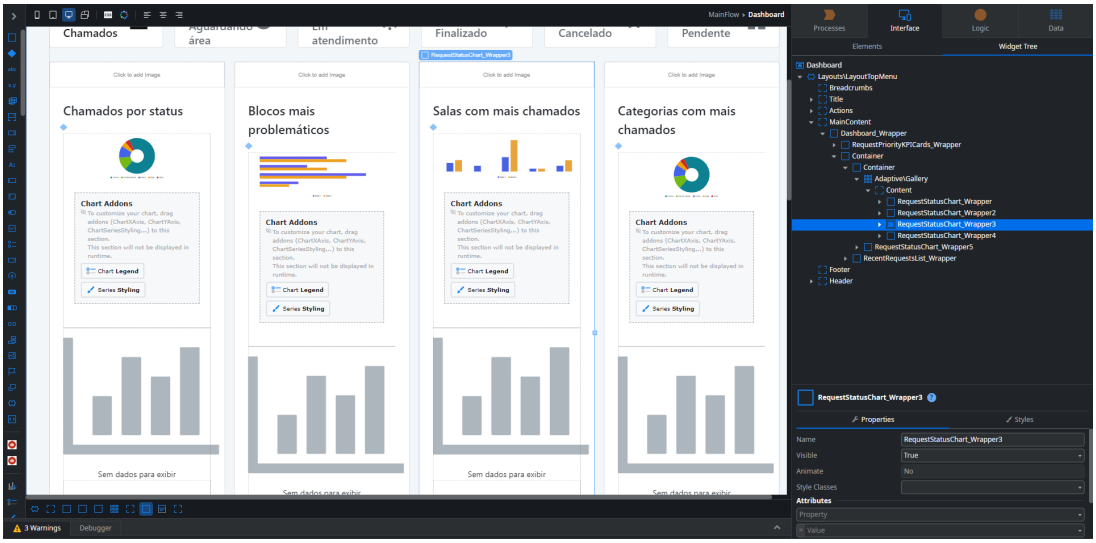

Figura 11 – Estruturação da tela Dashboard no OutSystems para consolidação de indicadores gerenciais

Fonte: Autoria própria (2026).

PARTE II

CONCLUSÃO

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.

O projeto cumpriu integralmente os objetivos propostos através das seguintes contribuições reais consolidadas:

- **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 tomada de decisão baseada em dados reais por parte da gestão do campus.
- **Estrutura Multitenant Escalável:** Modelagem arquitetural de banco de dados flexível e preparada para suportar a futura expansão e o isolamento lógico de dados de múltiplas instituições.

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.

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:

- **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.

Gartner IT Glossary. **Smart Campus**. 2025. https://www.gartner.com/en/information-technology/glossary/smart-campus. Acesso em: 11 set. 2025.

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.

International Facility Management Association (IFMA). **What is Facility Management?** 2025. https://www.ifma.org/what-is-facility-management. 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.

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.