INSTITUTO FEDERAL
Rondônia

MINISTÉRIO DA EDUCAÇÃO
Secretaria de Educação Profissional e Tecnológica
Instituto Federal de Educação, Ciência e Tecnologia de Rondônia

Campus Ji-Paraná
Coordenação do Curso Superior em Análise e desenvolvimento de Sistemas

VICTOR PEDRO BORGES SILVA

SISTEMA DE GERENCIAMENTO DE EQUIPAMENTOS ELETRÔNICOS

Ji-Paraná
2025

VICTOR PEDRO BORGES SILVA

SISTEMA DE GERENCIAMENTO DE EQUIPAMENTOS ELETRÔNICOS

Monografia entregue como Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia de Rondônia (IFRO), *Campus* Campus Ji-Paraná, como requisito parcial para obtenção do grau de Tecnólogo, junto ao Curso Análise e Desenvolvimento de Sistemas, sob a orientação do professor Gleison Guardia e coorientação do professor Reinaldo Lima Pereira.

Ji-Paraná
2025

Ficha catalográfica elaborada pelo Sistema Gerador de Ficha Catalográfica do IFRO.

Silva, Victor Pedro Borges.
Sistema de gerenciamento de equipamentos eletrônicos / Victor Pedro Borges Silva. - Ji-Paraná, 2025.
28 f.

Orientador(a): Prof. Me. Gleison Guardia.
Coorientador(a): Prof. Reinaldo Lima Pereira.

Trabalho de Conclusão de Curso (Superior de Tecnologia em Análise e Desenvolvimento de Sistemas) – Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO, Ji-Paraná, 2025.

1. Sistema Web. 2. Gerenciamento de equipamentos. 3. API RESTful. I. Guardia, Gleison (orient.). II. Pereira, Reinaldo Lima (coorient.). III. Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO. IV. Título.

Bibliotecário(a) Responsável: Cleuza Diogo Antunes, CRB-11/864

VICTOR PEDRO BORGES SILVA

SISTEMA DE GERENCIAMENTO DE EQUIPAMENTOS ELETRÔNICOS

Monografia entregue como Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia de Rondônia (IFRO), *Campus* Campus Ji-Paraná, como requisito parcial para obtenção do grau de Tecnólogo, junto ao Curso Análise e Desenvolvimento de Sistemas, sob a orientação do professor Gleison Guardia e coorientação do professor Reinaldo Lima Pereira.

Aprovado em:10/12/2025 pela banca examinadora.

Documento assinado digitalmente
gov.br
DANILO PEREIRA ESCUDERO
Data: 26/12/2025 17:14:14-0300
Verifique em https://validar.iti.gov.br

EXAMINADOR INTERNO

Documento assinado digitalmente
gov.br
JEFFERSON ANTONIO DOS SANTOS
Data: 26/12/2025 17:20:20-0300
Verifique em https://validar.iti.gov.br

EXAMINADOR EXTERNO

Documento assinado digitalmente
gov.br
GLEISON GUARDIA
Data: 26/12/2025 16:55:52-0300
Verifique em https://validar.iti.gov.br

ORIENTAÇÃO

SerproID Serpro
Reinaldo Lima Pereira
CPF: ***.509.772-**
26/12/2025
ICP Brasil

RESUMO

Este documento apresenta o desenvolvimento e a implementação de um sistema web para gerenciamento de equipamentos eletrônicos em processos de conserto, voltado a uma empresa que realizava o controle das entradas, saídas e andamento dos reparos por meio de formulários em papel. Esse modelo tradicional foi identificado como fonte de ineficiência operacional, maior risco de perda de informações, dificuldade de rastreabilidade e pouca transparência sobre o status dos equipamentos. Como solução, o trabalho define o escopo do produto e detalha o levantamento de requisitos, a modelagem comportamental e estrutural do sistema e a documentação técnica do software. O sistema contempla funcionalidades como autenticação, monitoramento, cadastros (clientes, usuários/funcionários, serviços, produtos e fornecedores), gestão e rastreamento de equipamentos, consulta de histórico e suporte a orçamentos, permitindo acompanhar as etapas do fluxo de trabalho e registrar mudanças de status de forma confiável. A metodologia empregada inclui observação do processo real, elaboração de diagramas (casos de uso, classes) e modelagem do banco de dados com dicionário de dados, além do planejamento por backlog. Como resultado, propõe-se uma aplicação que centraliza dados, melhora o controle do processo e apoia uma gestão mais organizada e produtiva.

Palavras-chave: Sistema Web. Gerenciamento de Equipamentos. API RESTful.

ABSTRACT

This document presents the development and implementation of a web system for managing electronic equipment throughout repair processes, designed for a company that previously controlled equipment intake, delivery, and repair progress using paper forms. This traditional model was identified as a source of operational inefficiency, greater risk of information loss, limited traceability, and low transparency regarding equipment status. As a solution, the work defines the product scope and details the requirements elicitation, the behavioral and structural modeling of the system, and the software's technical documentation. The system includes features such as authentication, monitoring, registrations (customers, users/employees, services, products, and suppliers), equipment management and tracking, access to history, and support for quotations, enabling the monitoring of workflow stages and the reliable recording of status changes. The methodology used includes observing the real process, producing diagrams (use cases and class diagrams), and modeling the database with a data dictionary, in addition to backlog-based planning. As a result, the proposed application centralizes data, improves process control, and supports a more organized and productive management.

Keywords: Web System. Equipment Management. RESTful API.

LISTA DE ABREVIATURAS E SIGLAS

ABNT – Associação Brasileira de Normas Técnicas

API – Application Programming Interface

AWS – Amazon Web Services

BIGINT – Inteiro grande (tipo de dado numérico)

CNPJ – Cadastro Nacional da Pessoa Jurídica

CPF – Cadastro de Pessoas Físicas

CSS – Cascading Style Sheets

FK – Foreign Key (Chave Estrangeira)

HTML – HyperText Markup Language

ID – Identificador

IFES – Instituto Federal do Espírito Santo

IFRO – Instituto Federal de Educação, Ciência e Tecnologia de Rondônia

LONGTEXT – Texto longo (tipo de dado)

MVC – Model–View–Controller

NF – Não Funcional (Requisito Não Funcional)

ORM – Object-Relational Mapping

PDF – Portable Document Format

PK – Primary Key (Chave Primária)

SGBD – Sistema de Gerenciamento de Banco de Dados

SQL – Structured Query Language

TCC – Trabalho de Conclusão de Curso

UML – Unified Modeling Language

WEB – (Rede Mundial de Computadores)

SUMÁRIO

1. TEMA

Desenvolvimento e Implementação de um Sistema de Gerenciamento de Equipamentos Eletrônicos Baseado em Tecnologias Web.

1.1 DELIMITAÇÃO DO TEMA

Nesse contexto, ao buscar um tema para meu Trabalho de dentro da empresa onde trabalho: a implementação de uma ferramenta capaz de agilizar nossos processos internos. Além de aumentar a eficiência, essa ferramenta também proporciona maior segurança aos dados e resultaria em economia de recursos para a própria empresa.

1.2 PROBLEMÁTICA

Por ser uma empresa especializada em reparos de equipamentos eletrônicos, há uma demanda constante de aparelhos que entram e saem diariamente. No entanto, alguns equipamentos permanecem nas instalações por mais tempo, e a única forma de gerenciá-los é por meio de formulários em papel, utilizados para cadastrar clientes e registrar os equipamentos. Assim, esse sistema antiquado torna o processo lento e desorganizado, gerando consequências adversas para o negócio. Além disso, a falta de acompanhamento do fluxo de trabalho pelo qual o equipamento passa — por exemplo, não saber qual é o seu status — pode causar desgaste no relacionamento com o cliente, pois ele não tem clareza sobre o andamento do conserto. A abordagem baseada em papel, além de consumir recursos, também apresenta risco de extravio de informações, o que pode resultar em problemas com os clientes. Adicionalmente, contribui para a ineficiência no ambiente de trabalho, reduzindo a produtividade e podendo gerar impactos financeiros negativos para a empresa a longo prazo.

2. JUSTIFICATIVA

Diante da dificuldade em definir e acompanhar o fluxo de reparo dos equipamentos, um sistema de gerenciamento do fluxo de trabalho (workflow) surge como uma solução viável para mitigar perdas e agilizar os processos. Dessa forma, a empresa pode inovar e se destacar no mercado, pois, quando a organização gerencia seus processos internos de maneira eficiente, transmite aos atores externos (clientes e funcionários) a percepção de um ambiente agradável, organizado e altamente produtivo.

3. HIPÓTESE

Como possível solução, pretende-se desenvolver uma aplicação web de workflow para o gerenciamento e controle desses equipamentos, com foco em agilidade e produtividade. A proposta inclui telas intuitivas e de fácil compreensão, além de ferramentas eficientes para análise e gestão dos equipamentos. Dessa forma, busca-se garantir a segurança dos dados coletados dos clientes e manter um fluxo de trabalho eficiente e controlado.

4. **OBJETIVOS**

4.1 OBJETIVO GERAL

Desenvolver uma aplicação web workflow voltada para a gestão eficiente de equipamentos eletrônicos que demandam reparos, visando aprimorar significativamente os processos de registro, acompanhamento e controle, a fim de promover uma gestão mais eficaz e transparente do andamento dos reparos.

4.2 OBJETIVOS ESPECÍFICOS

- Implementar módulo de cadastro de clientes.
- Implementar módulo de cadastro de serviços e produtos.
- Implementar módulo de gestão de equipamentos.
- Implementar módulo de orçamentos.
- Implementar módulo de rastreamento on-line do equipamento.

5. FUNDAMENTAÇÃO TEÓRICA

Segundo o IBGE (2025), no Brasil, o setor de serviços tem se consolidado de forma consistente, com crescimento de 3,7% em relação a 2023. Nesse cenário, as atividades relacionadas à reparação e manutenção de equipamentos de informática têm ganhado relevância, pois, devido à crescente demanda de consumidores que dependem desses equipamentos eletrônicos para trabalho, lazer e estudo, o número de empregados formais na divisão de “Reparação e manutenção de equipamentos de informática e comunicação e de objetos pessoais e domésticos” ultrapassou 80 mil trabalhadores em 2024, conforme dados do Observatório de Dados do SEBRAE (SEBRAE, 2025). Diante desse contexto, observa-se a necessidade de uma gestão eficiente de ordens de serviço, peças, prazos de entrega e histórico de manutenção desses equipamentos.

Pesquisas coordenadas pela Fundação Getúlio Vargas, em parceria com a Agência Brasileira de Desenvolvimento Industrial, indicam que a transformação digital já alcança a maior parte das micro e pequenas empresas brasileiras. No entanto, cerca de 60% a 66% ainda se encontram nos níveis iniciais de maturidade digital, com uso limitado de sistemas integrados de gestão e de automação de processos (FGV, 2022; ABDI; FGV, 2022). Assim, soluções de software voltadas a gerenciar e organizar o fluxo de trabalho tornam-se fundamentais para o desenvolvimento e o amadurecimento dessas empresas, contribuindo para a eficiência e a produtividade do setor de serviços.

5.1 LINGUAGENS DE PROGRAMAÇÃO

As linguagens de programação são ferramentas utilizadas pelos desenvolvedores para criar e aprimorar os softwares em que trabalham. Elas costumam empregar conceitos como programação orientada a objetos e programação procedural, além de serem classificadas como linguagens de alto e baixo nível. Por possibilitarem uma comunicação eficiente com o computador por meio de sequências de instruções e algoritmos, o domínio dessas linguagens proporciona ao profissional uma ampla variedade de possibilidades de desenvolvimento (RODEVA, 2023).

Segundo Tanenbaum (2007), as linguagens de baixo nível foram as primeiras a serem criadas e se caracterizam por uma sintaxe mais próxima da linguagem de máquina, o que se justifica pelo fato de seu propósito ser mais perceptível ao computador. Seu uso ainda é importante na atualidade, pois é fundamental para a programação de hardwares dos mais variados tipos. Um exemplo conhecido é a linguagem Assembly. Por outro lado, as linguagens de alto nível são mais recentes e se caracterizam pela maior semelhança com a linguagem humana, o que proporciona mais facilidade e clareza para quem as utiliza. Sua sintaxe inclui variáveis, estruturas de controle de fluxo, funções e módulos, além de permitir diferentes paradigmas de programação, como a orientação a objetos.

5.1.1 Paradigmas da Programação

Segue a versão revisada com correção ortográfica, melhoria de coerência e padronização (mantendo o sentido e as referências): Segundo Sebesta (2018), paradigmas são modelos ou estruturas utilizadas para organizar a solução de um problema, o qual pode ser representado por dados, variáveis e instruções. Na resolução desses problemas, podem ser utilizados diferentes paradigmas, como o lógico, o funcional e o orientado a objetos. De acordo com Silveira et al. (2021), o paradigma lógico fundamenta-se na lógica matemática e aborda o problema a ser resolvido por meio da descrição de regras e fatos. Esse paradigma é muito utilizado no desenvolvimento de inteligências artificiais, tendo o Prolog como um exemplo de linguagem que o implementa. Quanto ao paradigma funcional, observa-se que ele se baseia na construção de programas por meio de funções, nas quais, a partir de uma função principal, estruturam-se as demais. Esse paradigma oferece: (1) um conjunto de funções primitivas (semelhantes a instruções ou comandos); (2) formas funcionais para construir funções mais complexas a partir das primitivas (combinação de funções); (3) uma operação de aplicação de funções; e (4) estruturas para representar dados (PELLEGRINI, 2013). Entre as linguagens que adotam esse paradigma, destacam-se Haskell, Erlang e Lisp, entre outras. Por fim, o paradigma orientado a objetos traz um modelo mais próximo do mundo real, favorecendo a compreensão do sistema. Ele utiliza uma modelagem baseada em objetos, com características e ações, nos quais os objetos se relacionam entre si, modificando o estado do programa (CARVALHO; TEIXEIRA, 2010). Os principais elementos dessa estrutura são as

classes, os atributos e os métodos. Os objetos, por sua vez, são instâncias das classes e possuem um estado durante a execução do software, podendo ser modificados, persistidos e recarregados conforme a lógica implementada (SOUZA, 2021). Algumas linguagens que utilizam esse paradigma são Java, C#, PHP e Python.

5.1.2 UML

UML, sigla de Unified Modeling Language (Linguagem de Modelagem Unificada), é uma linguagem visual padronizada que permite a profissionais de tecnologia da informação modelar e documentar aplicações de software (e outros tipos de projetos). No contexto de design, a UML oferece um meio de visualizar a arquitetura de um sistema por meio de diversos diagramas, incluindo diagramas de atividades, componentes individuais do sistema, interação entre componentes, interfaces e integração com o ambiente externo, entre outros (REIS, 2019).

De acordo com Ventura (2019), os diagramas UML podem ser divididos em dois grupos: diagramas estruturais e diagramas comportamentais. Os diagramas estruturais se concentram em representar a estrutura do sistema, descrevendo sua organização e seus elementos, como classes, métodos, serviços e a forma como a arquitetura deve ser construída. Entre os exemplos desse grupo, destacam-se os diagramas de Classes, Objetos, Componentes e Estrutura Composta. Já os diagramas comportamentais, como o próprio nome sugere, têm como objetivo especificar o comportamento do sistema e sua interação com o usuário, oferecendo clareza sobre como as funcionalidades devem operar. Como exemplos, podem ser citados os diagramas de Casos de Uso, Atividades e Interação.

5.1.3 Python

Python é uma linguagem de programação de alto nível, dinâmica, interpretada e orientada a objetos, criada por Guido van Rossum na década de 1990. Sua

simplicidade e facilidade de compreensão a tornaram popular em diversas áreas da indústria tecnológica, além de atrair profissionais de diferentes campos, como engenheiros, cientistas de dados e pesquisadores. A linguagem é conhecida por seu amplo ecossistema de bibliotecas, sendo utilizada em desenvolvimento web, análise de dados, machine learning e inteligência artificial (KRIGER, 2022).

Segundo Caleiro e Ramos (2018), Python é uma linguagem fortemente tipada, o que significa que a aplicação de uma operação a um valor inadequado resulta em uma mensagem de erro. Seu sistema de tipos é dinâmico, com tipos implícitos, o que facilita o trabalho do programador, pois não é necessário declarar previamente o tipo dos objetos manipulados. Ainda assim, a cada momento, cada expressão está associada a um objeto cujo tipo pode ser compreendido como a classe à qual pertence. Para certos tipos, chamados imutáveis, o valor associado ao objeto permanece o mesmo; em outros, chamados mutáveis, há métodos disponíveis para alterar o valor associado a um objeto. Nesse sentido, Python oferece suporte a mais de um paradigma de programação, como o procedural, o funcional e o orientado a objetos. Devido à sua sintaxe simples e legível, é frequentemente escolhida por programadores em início de carreira.

5.1.4 Javascript

JavaScript surgiu em 1995 e foi criado por Brendan Eich, a pedido da empresa Netscape, com o objetivo inicial de validar formulários em HTML. Na época, como os navegadores eram mais limitados e focados principalmente em exibir conteúdo estático, a linguagem representou uma inovação importante. Em 1996, a Microsoft desenvolveu uma linguagem semelhante para ser utilizada no Internet Explorer (RABELO, 2022). Posteriormente, a Netscape submeteu a padronização da linguagem à organização internacional ECMA, responsável pela definição de padrões e especificações técnicas. Com o tempo, a linguagem evoluiu e tornou-se amplamente utilizada no lado do cliente (*client-side*), isto é, executada no dispositivo do usuário final. Isso envolve aquilo que o usuário vê e com que interage, como textos, imagens, elementos de interface e ações executadas no navegador (RABELO, 2022).

Segundo Copes (2023), JavaScript é uma linguagem de alto nível, caracterizada por ser dinâmica, isto é, por realizar diversas verificações e definições

em tempo de execução. Além disso, é considerada fracamente tipada, o que permite maior flexibilidade no uso de tipos, e também é multiparadigma, pois possibilita a adoção de diferentes abordagens de programação na construção de um sistema. Conhecido por sua versatilidade e relativa simplicidade, o JavaScript é amplamente utilizado no desenvolvimento de aplicações e sites web, e também pode ser aplicado na criação de programas para microcontroladores.

5.2 DESENVOLVIMENTO WEB

Segundo Roveda (2020), o desenvolvimento web pode ser caracterizado como a construção de software baseada e fundamentada na internet, dividida em duas partes essenciais, com características distintas, mas que são completamente interligadas e precisam estar bem integradas para o funcionamento correto de uma aplicação web. A primeira é o **front-end**, responsável por todo o código que o cliente visualiza e com o qual interage. Essa camada se comunica diretamente com o navegador e envolve o design e a interface da aplicação. A segunda é o **back-end**, que atua em uma camada geralmente invisível ao usuário, porém essencial para a lógica de funcionamento do sistema, o gerenciamento de dados e o controle do fluxo de informações. Dentro desse contexto, existem diversas ferramentas que servem como base para o desenvolvimento, como linguagens de programação em conjunto com bancos de dados. Para acelerar o processo e aumentar a eficiência, podem ser utilizados **frameworks**, que são bibliotecas e conjuntos de recursos que oferecem funcionalidades prontas, facilitando a criação e a manutenção de aplicações.

5.2.1 Back End

De acordo Souto (2023) o back end se trata de todas as tecnologias que estão escondidas do cliente, sua forma de trabalho consiste em trabalhar nos bastidores e garantir que a lógica de negócios, processamento de dados e as interações com o

banco de dados funcionem corretamente. Neste ambiente podem ser somadas inúmeras tecnologias como linguagens de programação, bancos de dados, servidores web, controle de versões, e containers.

5.2.1.1 ORM - Mapeamento de Objeto Relacional

De acordo com Kriger (2023), no desenvolvimento de software tornou-se popular o uso de linguagens de programação baseadas no paradigma de orientação a objetos, bem como a utilização de bancos de dados relacionais para armazenamento e organização das informações. Com o passar do tempo, surgiu uma técnica para reduzir as diferenças entre esses dois modelos, chamada ORM (*Object-Relational Mapping*), que, em português, significa Mapeamento Objeto-Relacional. Essa técnica facilita a interação entre a aplicação e o banco de dados relacional, operando por meio de camadas responsáveis por diferentes etapas do processo. A camada de abstração simplifica o acesso ao banco de dados; a camada de mapeamento objeto-relacional realiza a correspondência entre os objetos da aplicação e as tabelas do banco; a camada de gerenciamento de transações controla as transações tanto na aplicação quanto no banco de dados; a camada de consulta gera as instruções SQL necessárias para acesso aos dados; e, por fim, a camada de cache armazena temporariamente os dados mais utilizados, contribuindo para o desempenho da aplicação.

5.2.1.2 API

A API (Application Programming Interface) pode ser compreendida como uma interface padronizada que define como diferentes componentes de software trocam dados e funcionalidades entre si, por meio de regras e protocolos bem estabelecidos.

Em linhas gerais, trata-se de um mecanismo que organiza e expõe recursos de um sistema de forma controlada, permitindo que outras aplicações os consumam sem precisar conhecer os detalhes internos da sua implementação (IBM, 2025). Neste mesmo sentido a API também pode ser entendida como um conjunto de ferramentas, definições e contratos que orientam a criação de aplicações de software e a integração entre serviços distintos (RED HAT, 2023).

Do ponto de vista arquitetural, as APIs atuam como intermediárias na comunicação entre sistemas, estabelecendo o “canal” por onde as requisições são enviadas e as respostas são devolvidas, garantindo que dois aplicativos conversem de maneira estruturada e padronizada (MULESOFT, 2024; AWS, 2025). No contexto do desenvolvimento web, esse conceito costuma se materializar em coleções de métodos, propriedades, eventos e endpoints (URLs) que o desenvolvedor pode utilizar para interagir tanto com recursos do navegador quanto com serviços de terceiros, abstraindo a complexidade do código interno e oferecendo uma interface mais simples e segura para o consumo dessas funcionalidades (MDN WEB DOCS, 2023).

5.2.1.3 Framework Para Persistência de Dados

Frameworks de persistência são utilizados para realizar o mapeamento de dados de aplicações desenvolvidas no paradigma orientado a objetos, permitindo seu armazenamento em bancos de dados relacionais. Por esse motivo, também são denominados frameworks de mapeamento objeto-relacional. Diferentes frameworks, em diferentes linguagens, apresentam recursos e soluções semelhantes, bem como características e limitações próprias (SILVA, 2012). De acordo com Melo (2020), a linguagem PHP possui o Laravel como framework, amplamente utilizado no desenvolvimento de aplicações e que oferece recursos para persistência de dados. Criado por Taylor B. Otwell e baseado na arquitetura MVC, um de seus principais diferenciais é o sistema de templates, que facilita a construção de páginas HTML.

Seguindo o pensamento de Giuliani (2019), a linguagem de programação Java também conta com o Hibernate como framework de persistência de dados.

Desenvolvido por uma comunidade de programadores ao redor do mundo, o Hibernate apresenta vantagens como rapidez e facilidade no desenvolvimento por meio do ORM, suporte a múltiplos bancos de dados e uma comunidade ampla e ativa.

5.2.1.4 Django

O Django é um framework web full stack de código aberto baseado na linguagem de programação Python, amplamente utilizado no desenvolvimento de aplicações web. Criado em 2005 por Adrian Holovaty e Simon Willison, o Django surgiu da necessidade de acelerar e simplificar o desenvolvimento de sites de notícias locais, tornando possível a atualização frequente e eficiente do conteúdo.(Roveda, 2021).

Conforme Luz (2017) o Django conhecido por sua eficiência e agilidade tem sido bastante apontado como opção para projetos escaláveis e robustos por trazer uma gama de ferramentas que têm comprometimento com a qualidade e produtividade no desenvolvimento, o Django usa mapeamento objeto-relacional (ORM) e migrações de banco de dados. Sua arquitetura segue o padrão Model-View-Template (MVT), onde:

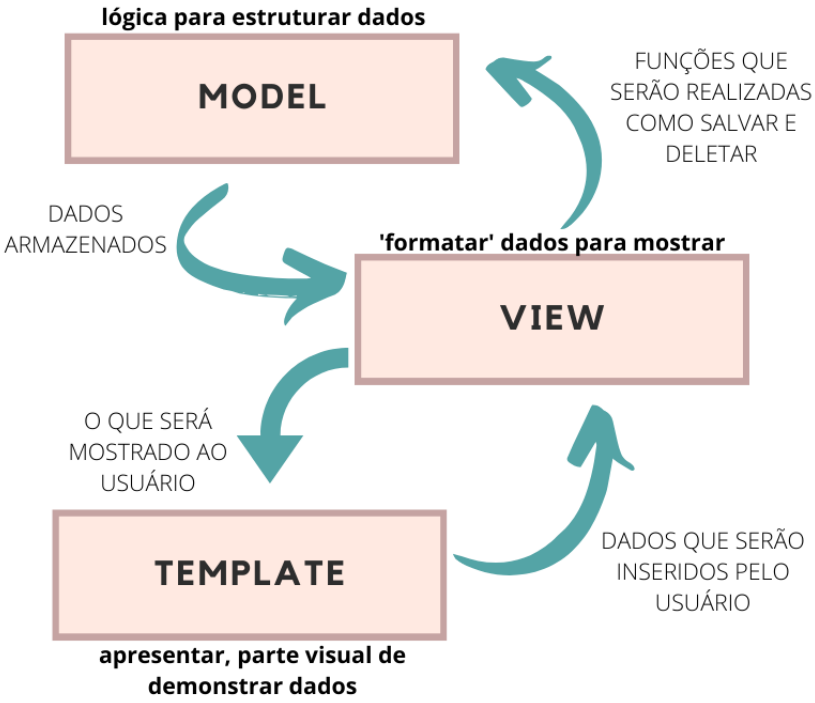
Model: Representa a estrutura de dados da aplicação e interage diretamente com o banco de dados. O Django fornece uma API para manipular dados de forma eficiente.

View: São funções Python responsáveis por receber solicitações do cliente e retornar respostas apropriadas. Aqui, os dados são extraídos e processados para gerar uma resposta.

Template: Nesta camada, os templates da aplicação são criados usando artefatos interpretados pelo navegador do usuário. Isso define como o conteúdo é apresentado ao usuário final em seu dispositivo.

Na figura 01 podemos ver um exemplo do fluxo que ocorre entre Model-View-Template (MVT).

Figura 01: Fluxo (MVT)



Fonte: SILVA (2020)

Se destaca pelo o uso de vários recursos de segurança , como autenticação de usuários, proteção contra ataques como sql injection, seu grande poderio de escalabilidade também são vantagens, sua documentação completa e detalhada juntamente com o suporte de uma comunidade ativa tornam a curva de aprendizado menor.

5.2.1.5 SGBD - Mysql

Database Management System (DBMS), ou Sistema de Gerenciamento de Banco de Dados (SGBD), é um conjunto de softwares utilizado para administrar uma base de dados. Ele é responsável por controlar o acesso, organizar, manter e proteger as informações de uma aplicação, tendo como principal objetivo gerenciar as bases de dados utilizadas por aplicações clientes e retirar delas essa responsabilidade

(ANDRADE, 2021). O MySQL foi criado na Suécia por David Axmark, Allan Larsson e o finlandês Michael Widenius. Eles iniciaram o projeto em 1980. O MySQL é um SGBD que utiliza a linguagem SQL como interface. Esse banco de dados é conhecido por sua facilidade de uso e, segundo Pacievith (2021), é utilizado por empresas como NASA, HP, Bradesco e Sony, entre outras. Sua interface simples e a capacidade de funcionar em diversos sistemas operacionais são alguns dos motivos para ser amplamente utilizado atualmente, e seu uso tem crescido cada vez mais (PACIEVITH, 2021).

5.2.2 Front End

O front-end é uma área da programação dedicada à criação da parte visual e interativa de um site, aplicativo ou software. Trata-se da camada com a qual o usuário tem contato direto, ou seja, aquilo que ele vê e utiliza ao acessar uma plataforma digital. Esse trabalho é realizado com linguagens como **HTML**, **CSS** e **JavaScript**, que permitem desenvolver a estrutura, o estilo, animações e funcionalidades da interface. O **HTML** define a estrutura e o conteúdo da página, o **CSS** determina o visual e o layout, e o **JavaScript** controla o comportamento e a interação dos elementos exibidos (KRIGER, 2023).

5.2.2.1 Html e Css

A Linguagem de Marcação de Hipertexto (HTML) é uma linguagem utilizada para estruturar a maior parte das páginas da internet e de aplicações online. O termo hipertexto refere-se a um texto que pode fazer referência a outros conteúdos por meio de links, enquanto uma linguagem de marcação é composta por marcações (tags) que indicam ao navegador a estrutura e a organização de um documento (ANDREI, 2023).

CSS é a sigla para *Cascading Style Sheets* (Folhas de Estilo em Cascata) e tem como função principal aplicar estilo aos elementos escritos em HTML. Em outras palavras, o CSS é responsável pela parte visual e estética de um site, como cores, fontes, espaçamentos, formatação de tabelas, variações de layout e ajustes de imagens, entre outros aspectos de estilização (CODE, 2022).

5.2.2.2 Next.js

O Next.js é um framework React voltado para a construção de aplicações web completas, de acordo com a documentação oficial o desenvolvedor utiliza componentes React para montar a interface e conta com o Next.js para recursos e otimizações adicionais, como roteamento, renderização no servidor e geração estática de páginas (NEXT.JS, 2025). O framework abstrai e configura automaticamente as ferramentas necessárias para o React, como empacotamento (*bundling*), compilação e entre outros, permitindo que o foco do desenvolvimento esteja nas regras de negócio da aplicação em vez de na configuração de infraestrutura (NEXT.JS, 2025). O Next.js é um framework construído sobre o React com a ideia de facilitar o desenvolvimento de aplicações modernas com funcionalidades que já estão prontas para produção, ajudando a resolver questões de desempenho, de roteamento que no React puro exigiram mais configurações manuais (HOSTGATOR, 2025; KINSTA, 2025).

6. METODOLOGIA

Com base nos objetivos deste trabalho e em todas as tecnologias apoiadas no referencial teórico, será abordado neste capítulo a metodologia aplicada através do Levantamento de requisitos, Diagrama de Casos de Uso, Diagrama de Classes e na Modelagem dos Dados.

6.1 ESCOPO DA PESQUISA

Em busca de uma solução para o problema, a metodologia aplicada foi embasada em uma pesquisa feita numa empresa especializada em consertos de equipamentos eletrônicos, onde após uma visita foi possível observar os seus processos internos, podendo assim listar os requisitos que o sistema deve portar. Deste modo, depois de listar os requisitos foi possível desenvolver os possíveis casos de uso do sistema. Com o material comportamental do sistema coletado foi viável elaborar a Modelagem de Dados que engloba o diagrama lógico do banco de dados e o dicionário de dados, finalizando com o diagrama de classes.

6.2 METODOLOGIA DO DESENVOLVIMENTO

O desenvolvimento do sistema de gerenciamento de equipamentos eletrônicos foi realizado com base na realidade da empresa em que a necessidade do software foi identificada. Primeiramente, realizou-se uma pesquisa sobre a demanda por esse tipo de sistema, partindo de uma necessidade já vivenciada no ambiente de trabalho, no qual o controle dos equipamentos e do fluxo de atendimento ainda era feito de forma manual, por meio de formulários em papel para gerenciar ordens de serviço. Esse diagnóstico inicial serviu como ponto de partida para buscar embasamento

teórico e confirmar a relevância de uma solução informatizada para o gerenciamento de equipamentos eletrônicos.

Em seguida, foi realizada uma entrevista no local de trabalho, com o objetivo de observar o fluxo real dos processos internos e levantar os requisitos funcionais e não funcionais do sistema. A partir dessa análise, foram definidos os cadastros necessários, as informações que deveriam ser gerenciadas e as características relacionadas à usabilidade, desempenho e segurança que precisariam ser consideradas.

Com os requisitos definidos, foram elaborados os casos de uso, descrevendo as interações do usuário com o software. Nessa etapa, identificaram-se os principais atores, como cliente, funcionário e técnico, bem como os fluxos de cada funcionalidade, as telas envolvidas e as ações possíveis em cada uma delas. Esses casos de uso contribuíram para organizar, de forma clara, o comportamento esperado do sistema e proporcionar uma visão mais precisa de como seria o desenvolvimento.

Com base nesses elementos, foi projetado o banco de dados do sistema. Definiu-se a organização das tabelas e seus relacionamentos, contemplando entidades como pessoa, funcionário, cliente, equipamento, serviço, entre outras. Antes da criação física do banco de dados, foi construído um diagrama, permitindo melhor visualização das entidades e de seus relacionamentos. Em seguida, foram definidos os campos de cada tabela e estruturado o dicionário de dados, detalhando tipos, chaves e restrições.

Com o modelo de dados estruturado, iniciou-se a construção do back-end, tomando o banco de dados como referência. Nessa fase, o foco foi estruturar a parte da aplicação responsável pelas regras de negócio e, posteriormente, desenvolver a API responsável pela comunicação entre o front-end e o banco de dados. As rotas da aplicação foram documentadas, o que permitiu validar e organizar os endpoints que seriam utilizados pelo sistema.

Após a implementação da API, iniciou-se o desenvolvimento do front-end, utilizando JavaScript e o framework Next.js. As telas foram construídas e estilizadas com base nos requisitos não funcionais previamente definidos, como usabilidade e

padronização visual. Em seguida, o front-end foi integrado à API, conectando formulários e ações da interface aos endpoints responsáveis por enviar e receber dados do banco. Durante o desenvolvimento, um dos principais desafios foi definir como seria estruturado o histórico do equipamento dentro do sistema. Como cada equipamento passa por diferentes status ao longo do fluxo de trabalho, foi necessário estudar e planejar a melhor forma de registrar essas mudanças, de modo que fosse possível acompanhar toda a trajetória do equipamento na aplicação. Isso exigiu ajustes na modelagem do banco de dados e a criação de uma estrutura específica para armazenar esse histórico de forma correta e confiável.

Por fim, conforme detalhado no Documento de Software, apresentado no APÊNDICE I, todas as etapas — desde o levantamento de requisitos e definição dos casos de uso, até a modelagem do banco de dados, a construção da API e a implementação do front-end — estão documentadas e organizadas para consulta e manutenção futura do sistema.

7. CONCLUSÃO

Este trabalho teve como objetivo propor e desenvolver uma aplicação web para o gerenciamento do fluxo de conserto de equipamentos eletrônicos, considerando uma demanda real identificada em uma empresa que ainda realizava esse controle por meio de registros manuais em papel. Dessa forma, a solução apresentada buscou centralizar informações, organizar etapas do processo e oferecer maior controle sobre entradas, execução, prazos e histórico de manutenção, contribuindo para um ambiente mais eficiente e confiável.

A partir do levantamento de requisitos funcionais e não funcionais, o sistema foi estruturado com foco em usabilidade, segurança e padronização do fluxo de trabalho. A modelagem do banco de dados e a definição dos casos de uso permitiram traduzir as necessidades do processo real para uma solução organizada, com cadastros, controle de equipamentos, acompanhamento por status, histórico de movimentações e suporte a orçamentos. A integração entre back-end (API) e front-end possibilitou a operacionalização dessas funcionalidades de forma prática, proporcionando uma visão clara do andamento do serviço, reduzindo riscos de extravio de informações e facilitando o acompanhamento das ordens de serviço.

Como principal contribuição, destaca-se a transformação de um processo manual e sujeito a falhas em um fluxo digital documentado e rastreável, favorecendo a comunicação entre empresa e cliente e ampliando a capacidade de gestão do negócio. Além disso, a documentação produzida no Documento de Software (APÊNDICE I) consolida as decisões técnicas, regras de negócio e estrutura do sistema, servindo como base para evolução, manutenção e futuras melhorias.

Por fim, reconhece-se que todo sistema pode ser aprimorado conforme novas demandas surgem. Como trabalhos futuros, recomenda-se evoluir relatórios gerenciais (indicadores de produtividade, prazos médios e taxa de retrabalho), implementar notificações automáticas de mudança de status, incorporar níveis de permissão mais refinados por perfil, além de rotinas de backup e auditoria mais completas. Assim, conclui-se que os objetivos definidos foram atendidos, e a aplicação proposta se mostra uma alternativa viável para modernizar e otimizar a gestão de reparos de equipamentos eletrônicos.

8. REFERÊNCIAS

AGÊNCIA BRASILEIRA DE DESENVOLVIMENTO INDUSTRIAL; FUNDAÇÃO GETÚLIO VARGAS. **Mapa da Digitalização das MPEs brasileiras**. Brasília, 2021. Disponível em: https://api.abdi.com.br/file-manager/upload/files/Mapa_da_Digitaliza%C3%A7%C3%A3o_das_MPEs_Brasileiras__1___1_.pdf. Acesso em: 30 nov. 2025.

AMAZON WEB SERVICES (AWS). **O que é uma API (interface de programação de aplicações)?** 2025. Disponível em: https://aws.amazon.com/pt/what-is/api/. Acesso em: 3 dez. 2025.

ANDRADE, A. **O que é um SGBD?** TreinaWeb, 27 jul. 2021. Disponível em: https://www.treinaweb.com.br/blog/o-que-e-um-sgbd. Acesso em: 24 nov. 2023.

CALEIRO, C.; RAMOS, J. **Introdução à Programação em Python**. Department of Mathematics of Instituto Superior Técnico, 8 fev. 2018. Disponível em: https://www.math.tecnico.ulisboa.pt/~ccal/python/nb02.html#Notebook-02---Conceitos-b%C3%A1sicos-da-linguagem-Python. Acesso em: 23 nov. 2023.

CARVALHO, V.; TEIXEIRA, G. **Programação Orientada a Objetos**. Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo – IFES. [s.l.: s.n.], [s.d.].

CODE, A. **HTML e CSS: entenda o que são e para que servem**. Awari, 13 jul. 2022. Disponível em: https://awari.com.br/html-css/. Acesso em: 25 nov. 2023.

COPES, F. **O Manual de JavaScript para iniciantes**. In: FreeCodeCamp, 31 maio 2023. Disponível em: https://www.freecodecamp.org/portuguese/news/o-manual-de-javascript-para-iniciantes/. Acesso em: 23 nov. 2023.

FGV – FUNDAÇÃO GETULIO VARGAS. **Estudo revela que 66% das micro e pequenas empresas estão nos níveis iniciais de maturidade digital**. Portal FGV, São Paulo, 04 mar. 2022. Disponível em: https://portal.fgv.br/noticias/estudo-revela-66-micro-e-pequenas-empresas-estao-niveis-iniciais-maturidade-digital. Acesso em: 30 nov. 2025.

GIULIANI, L. **O que é e porque devo utilizar o Hibernate?** Medium, 26 nov. 2019. Disponível em: https://medium.com/@leonardogiuliani/o-que-%C3%A9-e-porque-devo-utilizar-o-hibernate-66fae865a22f. Acesso em: 26 nov. 2023.

IBM. **O que é uma API (interface de programação de aplicativos)?** [s.d.]. Disponível em: https://www.ibm.com/br-pt/think/topics/api. Acesso em: 3 dez. 2025.

IBGE. **PIB cresce 3,4% em 2024 e fecha o ano em R$ 11,7 trilhões**. Agência IBGE de Notícias, Rio de Janeiro, 07 mar. 2025. Disponível em: https://agenciadenoticias.ibge.gov.br/agencia-sala-de-imprensa/2013-agencia-de-noticias/releases/42774-pib-cresce-3-4-em-2024-e-fecha-o-ano-em-r-11-7-trilhoes. Acesso em: 30 nov. 2025.

KRIGER, B. **Front end: entenda o que é, para que serve, como aprender essa especialidade!** Kenzie, 12 jun. 2023. Disponível em: https://kenzie.com.br/blog/front-end/. Acesso em: 25 nov. 2023.

KRIGER, B. **ORM (Object Relational Mapper) – saiba o que é e importância na programação**. Kenzie, 5 fev. 2023a. Disponível em: https://kenzie.com.br/blog/orm/. Acesso em: 26 nov. 2023.

KRIGER, D. **O que é Python, para que serve e por que aprender?** Kenzie, 8 jun. 2022. Disponível em: https://kenzie.com.br/blog/o-que-e-python/. Acesso em: 28 set. 2023.

LUZ, R. **Python e Django**. Rio de Janeiro, RJ: Escola Superior de Redes, 2017.

MDN WEB DOCS. **Introdução às Web APIs**. [s.d.]. Disponível em: https://developer.mozilla.org/pt-BR/docs/Learn_web_development/Extensions/Client-side_APIs/Introduction. Acesso em: 3 dez. 2025.

MELO, D. **O que é Laravel?** Tecnoblog, 27 nov. 2020b. Disponível em: https://tecnoblog.net/responde/o-que-e-laravel-guia-para-iniciantes/. Acesso em: 25 nov. 2023.

MULESOFT. **O que é uma API (interface de programação de aplicativos)?** 2025. Disponível em: https://www.mulesoft.com/pt/api/what-is-an-api. Acesso em: 3 dez. 2025.

PACIEVITH, Y. **MySQL**. InfoEscola, 13 jan. 2021. Disponível em: https://www.infoescola.com/informatica/mysql/#google_vignette. Acesso em: 25 nov. 2023.

PELLEGRINI, J. C. **Programação Funcional e Concorrente com Scheme**. Santo André, SP: UFABC – Universidade Federal do ABC, 2013.

PRESSMAN, R. S. **Engenharia de Software**. 6. ed. Porto Alegre: AMGH, 2010.

RED HAT. **API (interface de programação de aplicações)**. 2023. Disponível em: https://www.redhat.com/pt-br/topics/api/what-are-application-programming-interfaces. Acesso em: 3 dez. 2025.

REBELLO, M. **Javascript: o que é, como surgiu e onde utilizar**. Resilia, 10 jul. 2022. Disponível em: https://www.resilia.com.br/blog/javascript-o-que-e-como-surgiu-e-onde-utilizar/. Acesso em: 23 nov. 2023.

REIS, F. **O que é UML – Unified Modelling Language**. Bóson Treinamentos em Ciências e Tecnologia, 13 dez. 2019. Disponível em: https://www.infoescola.com/engenharia-de-software/uml/. Acesso em: 24 nov. 2023.

ROVEDA, U. **Desenvolvimento web: o que é e como ser um desenvolvedor web**. Kenzie, 11 dez. 2020. Disponível em: https://kenzie.com.br/blog/desenvolvimento-web/. Acesso em: 25 nov. 2023.

ROVEDA, U. **Linguagem de programação: o que é e qual linguagem aprender**. Kenzie, 13 jun. 2023. Disponível em: https://kenzie.com.br/blog/linguagem-de-programacao/. Acesso em: 22 nov. 2023.

ROVEDA, U. **O que é o Django, para que serve e como utilizar esse framework**. Kenzie, jan. 2021. Disponível em: <[URL não informado]>. Acesso em: 27 set. 2023.

SEBESTA, R. W. **Conceitos de Linguagens de Programação**. 11. ed. [s.l.]: Bookman Editora, 2018.

SEBRAE. **Reparação e manutenção de equipamentos de informática e comunicação e de objetos pessoais e domésticos**. Observatório de Dados SEBRAE, 2024. Disponível em: https://observatorio.sebrae.com.br/profile/industry/reparacao-e-manutencao-de-equipamentos-de-informatica-e-comunicacao-e-de-objetos-pessoais-e-domesticos. Acesso em: 30 nov. 2025.

SILVA, D. **Como funciona a arquitetura MTV (Django)**. 3 jan. 2020. Disponível em: https://diandrasilva.medium.com/como-funciona-a-arquitetura-mtv-django-86af916f1f63. Acesso em: 27 nov. 2023.

SILVA, M. **Estudo comparativo entre frameworks de mapeamento objeto-relacional aplicados na implementação de padrões de análise de sistemas**. [s.l.]: [s.n.], [s.d.].

SILVEIRA, S. et al. **Paradigmas de Programação: uma introdução**. Belo Horizonte, MG: Synapse Editora, [s.d.].

SOUTO, M. **Front-end, Back-end e Full Stack**. Alura, 18 jul. 2023. Disponível em: https://www.alura.com.br/artigos/o-que-e-front-end-e-back-end. Acesso em: 25 nov. 2023.

SOUZA, I. **Paradigmas de Programação**. Guia.dev, 2 dez. 2021. Disponível em: https://guia.dev/pt/pillars/languages-and-tools/programming-paradigms.html. Acesso em: 23 nov. 2023.

TOTVS. **Modelagem de dados: o que é, como funciona e tipos**. TOTVS, 30 dez. 2021. Disponível em: https://www.totvs.com/blog/negocios/modelagem-de-dados/. Acesso em: 20 nov. 2023.

VENTURA, P. **UML (Unified Modeling Language) é uma linguagem poderosa para comunicação em equipes de produção de software**. Indtech, 31 jan. 2019. Disponível em: https://www.ateomomento.com.br/diagramas-uml/. Acesso em: 24 nov. 2023.