

UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO

Nome dos integrantes

Anderson Simonassi RA:2320994

Anderson Chaves RA: 2230159

Miqueias Oliveira RA: 2109790

Sistema de gerenciamento para oficinas de manutenção de equipamentos eletrônicos

Santo André - SP
2025
UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO

Sistema de gerenciamento para oficinas de manutenção de equipamentos eletrônicos

Relatório Técnico-Científico apresentado na disciplina de Projeto Integrador para o curso de Engenharia da Computação da Universidade Virtual do Estado de São Paulo (UNIVESP).

Santo André - SP
2025

SIMONASSI, Anderson; CHAVES, Anderson; OLIVEIRA, Miqueias; **Sistema de gerenciamento para oficinas de manutenção de equipamentos eletrônicos**. Relatório Técnico-Científico. Engenharia da Computação – **Universidade Virtual do Estado de São Paulo**. Tutor: Lucas Laprano. Polo Santo André, 2025.

RESUMO

Os equipamentos eletrônicos estão presentes em nossas vidas de maneira intensa, um grande exemplo os aparelhos de telefone celular, que de acordo com uma pesquisa realizada pela FGVcia em 2024, no Brasil temos mais de 222 milhões de computadores, 258 milhões de “Smartphones” e mais de 280 milhões de televisores (MEIRELLES, 2024). Com esta grande quantidade de dispositivos o impacto ambiental é imenso. Existem milhares de oficinas especializadas no reparo destes equipamentos que, através da extensão de sua vida útil, podem contribuir de maneira significativa para a redução de resíduos de lixo eletrônico, porém a grande maioria são estabelecimentos pequenos: quiosques, garagens, cômodos de residências; que não dispõem de sistemas de gerenciamentos sofisticados para auxiliar a administração do negócio. Implementar um sistema informatizado para agilizar o acesso as informações de um histórico rico e confiável pode garantir melhores condições de trabalho, tempos de resposta mais eficientes e manutenções mais eficazes. Após entrevista, realizamos um levantamento prévio dos requisitos necessários para implementação de um sistema de cadastro de reparos para este público. O sistema deve ser simples e utilizar recursos básicos de hardware com softwares gratuitos, inicialmente a hospedagem será feita em servidor local. A tecnologias escolhidas em consonância com a proposta do projeto integrador serão: Servidor web: Apache, sistema de banco de dados: MySQL, Framework web: Django e o sistema operacional: Linux UBUNTU.

PALAVRAS-CHAVE: Dispositivos Eletrônicos; Reparação; Oficinas de reparo; Sistemas de gerenciamento; Django, MySQL, Linux.

LISTA DE ILUSTRAÇÕES

FIGURA 1: USO DE FRAMEWORKS NO GITHUB STARS.	14
FIGURA 2: TELA INICIAL DO MYSQL WORKBENCH.	18
FIGURA 3: DIAGRAMA ENTIDADE-RELACIONAMENTO MODELO DE PETER CHEN DO BANCO DE DADOS.....	18
FIGURA 4: PRIMEIRA IMPLEMENTAÇÃO DO BANCO DE DADOS.	19
FIGURA 5: CRIAÇÃO DO BANCO DE DADOS E TABELAS.....	20
FIGURA 6: BLOCO SQL PARA CRIAÇÃO DAS VIEWS.	21
FIGURA 7: PROCEDIMENTO PARA CÁLCULO DO TEMPO MÉDIO DE REPARO.	21
FIGURA 8: CRIAÇÃO DO TRIGGER.	22
FIGURA 9: INSERÇÃO DE DADOS À TABELA E COMANDO PARA VISUALIZAÇÃO.....	22
FIGURA 10: TABELA EQUIPAMENTO COM PREENCHIMENTO TESTE.	22
FIGURA 11: INSERÇÃO DE DOIS REPAROS EFETUADOS NA TABELA DE REPAROS.	23
FIGURA 12: TABELA REPARO COM ATRIBUTOS PREENCHIDOS.	23
FIGURA 13: VERIFICAÇÃO DA VERSÃO DO PYTHON E INSTALAÇÃO DO PIP.	23
FIGURA 14: INSTALAÇÃO DO AMBIENTE VIRTUAL PARA DESENVOLVIMENTO.....	24
FIGURA 15: INSTALAÇÃO DO FRAMEWORK DJANGO.....	24
FIGURA 16: CRIAÇÃO DA ESTRUTURA UTILIZANDO O DJANGO.....	24
FIGURA 17: CRIAÇÃO DO CATÁLOGO DA APLICAÇÃO E ESTRUTURA DA PASTA.....	25

FIGURA 18: INSERÇÃO DE ELEMENTOS À APLICAÇÃO.....	26
FIGURA 19: INCLUSÃO DAS CONFIGURAÇÕES DO MYSQL NO ARQUIVO SETTINGS.PY.....	26
FIGURA 20: CONFIGURANDO AS URLS E REDIRECIONANDO PARA O CATÁLOGO.....	27
FIGURA 21: ARQUIVO URLS.PY PADRÕES URL.	27
FIGURA 22: VISUAL STUDIO CODE.	28
FIGURA 23: LAYOUT DA PÁGINA INICIAL DO PROJETO.	29
FIGURA 25: APRESENTAÇÃO DE TRECHO DO CÓDIGO CSS DO PROJETO.	30
FIGURA 26: IMAGEM DA TELA INICIAL DO PROJETO.....	31
FIGURA 27: IMAGEM DA TELA INICIAL DO PROJETO.....	32
FIGURA 28: PREENCHIMENTOS RELATIVOS AO EQUIPAMENTO.	32
FIGURA 29: CAMPOS PARA O PREENCHIMENTO DOS DETALHES DO REPARO.	33
FIGURA 30: PÁGINA WEB SOBRE O PROJETO INTEGRADOR.....	33

SUMÁRIO

1 INTRODUÇÃO	8
2 DESENVOLVIMENTO.....	10
2.1 Objetivos.....	10
2.2 Justificativa e delimitação do problema	11
2.3 Fundamentação teórica	12
2.4 Metodologia.....	17
2.5 Resultados preliminares: solução inicial	32
REFERÊNCIAS	35

1 INTRODUÇÃO

Nos últimos 50 anos o desenvolvimento da indústria eletrônica e dos bens de consumo tiveram um crescimento exponencial, conforme (DECON - DEPARTAMENTO DE ECONOMIA ABINEE, 2024) “o faturamento do setor eletroeletrônico deverá atingir R\$ 226,7 bilhões, resultado 11% acima do realizado em 2023 (R\$ 204,6 bilhões)” no Brasil. Conforme (MEIRELLES, 2024) “O potencial de uso de recursos de tecnologia do Brasil já é alto. Praticamente todos os domicílios já têm televisão, o número de smartphones já é maior que a população desde 2017, e os 222 milhões de computadores acabam de ultrapassar o número de habitantes.” Com este elevado número de dispositivos, muitos deles durante seu ciclo de vida deverão passar por algum processo de manutenção e/ou reforma, que além do impacto econômico reduz o impacto ambiental gerado através do descarte, “de acordo com a quarta edição do Monitor Global de Lixo Eletrônico, GEM, foram produzidos 62 milhões de toneladas de resíduos eletrônicos em 2022. Esse total preencheria 1,5 milhões de caminhões de 40 toneladas.” (ONU - ORGANIZAÇÃO DAS NAÇÕES UNIDAS, 2024). Isto demonstra a importância das oficinas de reparação neste contexto, pelos impactos econômicos (geração de renda e economia dos consumidores) e a grande contribuição para a sustentabilidade do meio ambiente.

De acordo com (PRATES e OSPINA, 2009) as funções da administração: planejamento, organização, liderança e controle, devem fornecer informações de suma importância para o gerenciamento do negócio. Segundo estes autores:

A tecnologia é o fator individual de mudança de maior importância na transformação das empresas. Tais transformações não se restringem apenas ao modo de produzir bens e serviços, mas induzem novos processos e instrumentos que atingem por completo a estrutura e o comportamento das organizações, repercutindo diretamente em sua gestão (Partes & Ospina et al., 2009).

Percebe-se que o uso da tecnologia da informação em pequenos negócios pode melhorar a performance administrativa que segundo a pesquisa de (PRATES e OSPINA, 2009) “após a implantação da TI, perceberam a sua importância nos processos, aumentando a capacidade de trabalho, levando a empresa a robustecer a sua competitividade.”

Ao analisar o contexto da oficina de reparos SOBREPONE – ASSISTENCIA TECNICA percebemos a necessidade de implementar um sistema de gerenciamento para este ramo de negócio, focado em pequenos estabelecimentos, onde muitas vezes apenas uma pessoa é responsável por todo gerenciamento e execução dos serviços de reparação. Este projeto se propõe a utilizar recursos informatizados que apresentem baixo custo operacional e tecnologias acessíveis ao público-alvo desta proposta, levando em conta o conteúdo oferecido pela universidade e a delimitação do tema do Projeto Integrador.

Utilizaremos um “*Framework*” para estruturação do projeto baseado em página web e a escolha recaiu sobre o Django que segundo (MOZILLA, 2025) o Django é um framework de web “*server-side*” extremamente popular escrito em Python, sendo uma das linguagens mais populares da atualidade conforme (IEEE, 2024) está em primeiro lugar a nível mundial. Como há a necessidade de armazenar as informações de forma segura e confiável e de maneira sistemática a implementação de um sistema gerenciador de banco de dados (SGBDR) se faz necessário, mantendo o foco em baixo custo e padronização a nível mundial, a utilização do MySQL se torna uma escolha sensata pois de acordo com (ORACLE, 2024) “O MySQL é um sistema de gerenciamento de banco de dados relacional (RDBMS) de código aberto usado para armazenar e gerenciar dados”, sendo um dos mais conhecidos do mundo. A utilização do HTML (“*HyperText Markup Language*”) segundo (MOZILLA, 2025) é o bloco de construção mais básico da web e CSS (“*Cascading Style Sheet*”) de acordo com (MOZILLA, 2025) é uma linguagem declarativa que controla a apresentação visual de páginas web em um navegador, servirão para a criação o layout e aparência do site, também conhecido como “*frontend*”. A aplicação será hospedada em um sistema operacional baseado em LINUX na sua distribuição UBUNTU, que é gratuita, não necessita de hardware poderoso e funciona perfeitamente em um ambiente local.

2 DESENVOLVIMENTO

2.1 OBJETIVOS

Com base nas solicitações do cliente deseja-se a criação de um sistema de cadastro e recuperação de dados dos reparos efetuados em sua oficina e levando em conta as restrições orçamentárias. Analisamos os recursos oferecidos por sistemas gratuitos e “open-source” para a elaboração e criação de um sistema baseado em tecnologias web e fácil acesso a um sistema de gerenciamento de banco de dados com a utilização de um “framework” consolidado no mercado.

Tendo em vista o conhecimento adquirido no curso até o presente momento, as experiências dos membros da equipe, decidiu-se implementar um sistema baseado nas seguintes premissas:

- Programação da página em HTML (“*HyperText Markup Language*”);
- Estilos e formatação das páginas criadas CSS (“*Cascading Style Sheet*”);
- “Framework” Django baseado na linguagem de programação Python;
- Sistema de gerenciamento de banco de dados o MySQL.

Como objetivo final temos o objetivo de criar um sistema de página web utilizando o HTML e CSS para criação do layout e formatação dos elementos, como sistema de “*backend*” o “framework” Django que será responsável pela interface do sistema disponível ao usuário também conhecido como “*frontend*” ao sistema gerenciador de banco de dados MySQL que assumirá a responsabilidade de cadastrar, armazenar, gerenciar e permitir a consulta aos dados pelo usuário, em um sistema local inicialmente e posteriormente através da internet.

2.2 JUSTIFICATIVA E DELIMITAÇÃO DO PROBLEMA

A reciclagem e a reparabilidade dos equipamentos eletrônicos têm um impacto significativo para o meio ambiente e para redução da exploração dos recursos naturais do planeta como exemplifica o artigo: Por que o Brasil precisa regulamentar o direito ao reparo?

O reparo de equipamentos elétricos e eletrônicos é uma estratégia de minimização de resíduos que traz benefícios ambientais significativos. Ao consertar os equipamentos que apresentam defeitos ou que podem ser melhorados, evita-se o descarte prematuro e o consumo excessivo de novos produtos, que geram impactos negativos em todo o ciclo de vida, desde a extração de recursos naturais até a disposição final dos resíduos. (ANCILLOTTI, 2024)

Dada a importância desta estratégia e ao observar o elevado número de oficinas de reparos eletrônicos presentes nos bairros, em galerias comerciais, onde muitas vezes o proprietário é ao mesmo o técnico responsável pela manutenção. Ocorre a indagação: como é feito o controle dos equipamentos nos quais um reparo foi efetuado e como os registros dos defeitos e resoluções são registradas, para futuras consultas que garantirão uma maior eficiência do processo?

Através de uma entrevista com o técnico/proprietário Rafael Santos da oficina SOBREPHONE – ASSISTENCIA TECNICA, foi apontado que os registros não são feitos de maneira sistemática e quando há o registro é feito em caderno sem uma sequência ou rastreabilidade dos equipamentos atendidos e dos reparos realizados. Assim percebe-se os benefícios que um sistema informatizado pode oferecer a este empreendimento, que de acordo com (PRATES e OSPINA, 2009) “após a implantação da TI, perceberam a sua importância nos processos, aumentando a capacidade de trabalho, levando a empresa a robustecer a sua competitividade.”

O uso das técnicas e dos recursos lecionados pela universidade proporcionam uma visão refinada do que o uso das tecnologias da informação de baixo custo e gratuitas, que podem oferecer melhorias aos processos presentes na comunidade em que estamos inseridos. No caso do Sr. Rafael Santos está implementação pode agregar maior eficiência e eficácia nas manutenções realizadas, trazendo melhores condições de trabalho, melhor lucratividade e satisfação dos clientes.

2.3 FUNDAMENTAÇÃO TEÓRICA

A manutenção de equipamentos necessita de um histórico confiável conforme (KOZIMA, 2017)

a manutenção é importantíssima para que sejam preservados as características ideais e um bom funcionamento de cada equipamento, ferramenta ou máquina. Mas como em todos os segmentos, para que isso ocorra adequadamente e com qualidade principalmente equipamentos desgastados com o tempo de uso, o departamento responsável precisa dispor de um bom histórico das manutenções realizadas (KOZIMA, 2017)

Sistemas informatizados garantem uma estrutura segura, sistematizada e confiável para o armazenamento, tratamento e disponibilidade de dados, segundo Tanenbaum:

Os sistemas informatizados são projetados para garantir o armazenamento seguro, a confiabilidade e a disponibilidade dos dados por meio de mecanismos como redundância, backup automatizado e controle de acesso. Esses sistemas empregam tecnologias avançadas de segurança da informação, como criptografia e autenticação multifator, assegurando a integridade e a proteção das informações (TANENBAUM e WETHERALL, 2011).

Os modelos mais tradicionais para o gerenciamento de dados em sistemas informatizados na atualidade são os Sistemas de Gerenciamento de Bancos de Dados Relacional (SGBDR) conforme (MICROSOFT, 2025) “Bancos de dados relacionais são um tipo de banco de dados que armazena e organiza pontos de dados com relacionamentos definidos para acesso rápido”, os SGBDR podem ser pagos ou gratuitos que estão alinhados com o propósito deste projeto onde deve ser gratuito, confiável e com presença sólida no mercado além de ter características de uso profissional. No mercado existem diversos sistemas que se enquadram nestas premissas sendo alguns deles: PostgreSQL, MariaDB, MongoDB e MySQL. Para nossa proposta utilizaremos o MySQL de acordo com (MICROSOFT, 2025) “O My Structured Query Language (MySQL) é um banco de dados relacional SQL de código aberto comum que executa todos os comandos SQL básicos, como gravação e consulta de dados”, além de ser confiável, estável, seguro e amplamente utilizado no mercado.

Tendo em mente o desenvolvimento do banco de dados devemos seguir algumas etapas conforme (CARVALHO, 2014) são:

1. Levantamento de Requisitos: Coleta e análise das necessidades dos usuários e das funcionalidades desejadas para o sistema.
2. Modelagem Conceitual: Criação de um modelo abstrato que representa as entidades e os relacionamentos do domínio do problema, geralmente utilizando diagramas como o Modelo Entidade-Relacionamento (MER).
3. Modelagem Lógica: Transformação do modelo conceitual em um modelo lógico compatível com o MySQL, definindo tabelas, colunas, tipos de dados e restrições.
4. Normalização: Aplicação de regras para organizar os dados nas tabelas, visando reduzir redundâncias e melhorar a integridade dos dados.
5. Implementação Física: Tradução do modelo lógico para o ambiente físico do MySQL, incluindo a criação de tabelas, índices e outros objetos de banco de dados.
6. Carga de Dados: Inserção dos dados iniciais no banco de dados, garantindo que estejam consistentes com as definições e restrições estabelecidas.
7. Teste e Validação: Execução de testes para assegurar que o banco de dados atende aos requisitos especificados e funciona conforme esperado.
8. Manutenção e Monitoramento: Acompanhamento contínuo do desempenho e da integridade do banco de dados, realizando ajustes e atualizações conforme necessário.

Após a modelagem do banco de dados é necessária a elaboração de uma interface com o usuário e uma ponte entre o banco de dados que opera no “*backend*” que segundo (DEV MEDIA, 2021) “O Backend a gente não consegue ver, mas ele é a camada principal do software. Ele é o responsável em processar os dados e executar as ações que o software se propõe a fazer” para este elo optou-se também por uma solução gratuita e bem difundida no mercado de acordo com (OLIVEIRA, 2024) o Django é o segundo “framework” mais utilizado no mercado. Através do github stars (PATEL, 2023) como demonstrado no gráfico a seguir:

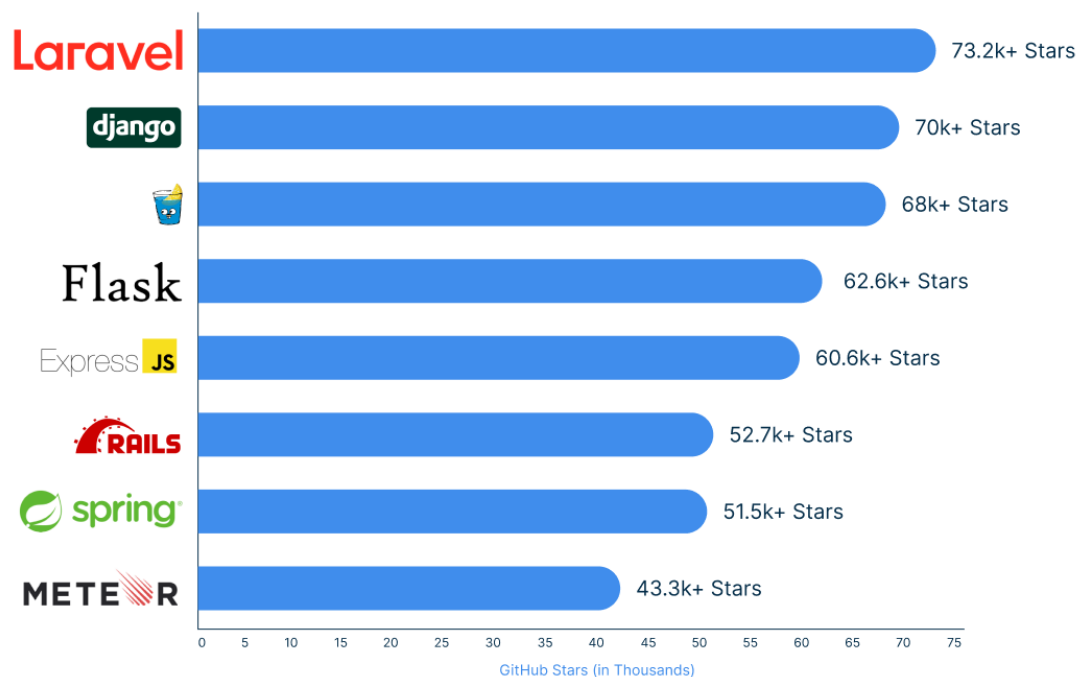


Figura 1: Uso de frameworks no GitHub Stars. FONTE: MindInventory (PATEL, 2023)

Como cita (DJANGOPROJECT, 2025) “Django é um framework web Python de alto nível que incentiva o desenvolvimento rápido e o design limpo e pragmático.”

Com base no livro de Antonio Melé as etapas para criação de uma aplicação web utilizando o “framework” Django, são:

1. Configuração do Ambiente de Desenvolvimento: Instale o Python e o Django, configure um ambiente virtual e prepare as dependências necessárias para o projeto.
2. Criação do Projeto Django: Utilize o comando `django-admin startproject` para iniciar um novo projeto, estabelecendo a estrutura básica da aplicação.
3. Desenvolvimento de Aplicações (Apps): Dentro do projeto, crie aplicações modulares usando o comando `python manage.py startapp`, permitindo uma organização eficiente dos componentes.
4. Definição dos Modelos (Models): Projete e implemente as classes que representam as tabelas do banco de dados, definindo os campos e relacionamentos necessários.

5. Configuração do Banco de Dados: Ajuste as configurações do banco de dados no arquivo `settings.py` e aplique as migrações para criar as tabelas correspondentes.
6. Desenvolvimento das Views e Templates: Crie as funções ou classes que processam as requisições (views) e os arquivos HTML que definem a interface do usuário (templates).
7. Implementação de URLs: Mapeie as URLs da aplicação para as respectivas views, organizando a navegação do sistema.
8. Testes e Depuração: Execute testes para verificar o funcionamento correto da aplicação e corrija eventuais erros identificados.
9. Implantação (Deployment): Configure o ambiente de produção e publique a aplicação em um servidor web, tornando-a acessível aos usuários.

Definido o “*framework*” para o “*backend*” necessita-se determinar qual abordagem será utilizada no “*frontend*” que segundo (DEV MEDIA, 2021) “ o frontend é a parte visual e interativa de um software, seja ele um site ou um aplicativo”, mantendo a premissa de baixo custo e ampla utilização o HTML (“*HyperText Markup Language*”) em conjunto com o CSS (“*Cascading Style Sheet*”) conforme (AWARI, 2023)

HTML, que significa HyperText Markup Language (linguagem de marcação de hipertexto) é responsável por estruturar o conteúdo de uma página na web. Ele define elementos como cabeçalhos, parágrafos, imagens, links e outros componentes que tornam um website funcional. (AWARI, 2023)

Por sua vez o CSS também de acordo com (AWARI, 2023)

CSS, que significa Cascading Style Sheets (folhas de estilo em cascata), é usado para definir a apresentação visual desse conteúdo. Ele controla as cores, a tipografia, o layout, os efeitos visuais e outros aspectos visuais de um website. (AWARI, 2023)

Em paralelo o desenvolvimento do conjunto web para o “*frontend*” se fundamenta nos passos que de acordo com (SILVA, 2015) baseiam-se em:

1. Planejamento e Definição de Requisitos: Identifique o propósito do site, o público-alvo e os principais conteúdos a serem apresentados.
2. Criação do “*Wireframe*”: Desenhar um esboço visual das páginas, definindo a disposição dos elementos e a hierarquia das informações.
3. Estruturação com HTML: Desenvolver a marcação HTML para estruturar o conteúdo, utilizando “*tags*” semânticas adequadas para títulos, parágrafos, listas, links, imagens e outros elementos.
4. Estilização com CSS: Aplicar estilos ao conteúdo HTML utilizando CSS, definindo cores, tipografia, espaçamentos, layouts responsivos e outros aspectos visuais.
5. Testes e Validação: Verificar a compatibilidade da interface em diferentes navegadores e dispositivos, assegurando a responsividade e a acessibilidade.
6. Otimização e Publicação: Otimizar o código e os recursos para melhorar o desempenho e, em seguida, publique o site em um servidor web.

O processo envolvido para a elaboração do projeto proposto analisará as solicitações do cliente, a modelagem do banco de dados e implementação do banco, por conseguinte a criação da estrutura utilizando o “framework” Django que será responsável pela conexão entre o banco de dados e a apresentação web modelada em HTML com CSS, nesta etapa serão efetuadas as etapas de testes em servidor local.

2.4 METODOLOGIA

Inicialmente observou-se a comunidade na qual estamos inseridos e problemas relacionados ao ambiente no qual vivemos, a presença de várias oficinas que efetuam reparos em equipamentos eletrônicos, principalmente celulares, e o impacto ambiental que o descarte destes dispositivos tem gerado ao planeta, com estas indagações localizamos a oficina SOBREPHONE – ASSISTENCIA TECNICA no bairro Jardim Marília na cidade de São Paulo que através do Sr. Rafael Santos fizemos o levantamento dos requisitos para que a oficina dele utilizando recursos da tecnologia da informação pudesse aumentar sua eficiência e eficácia nos reparos por ele executados, assim os pontos citados foram:

1. Criar uma identificação única dos equipamentos
2. Identificar o tipo do equipamento
3. Identificar a marca e modelo do equipamento
4. Criar um campo para descrever o defeito e o reparo executado.
5. Identificar a data de entrada e saída do equipamento
6. Criar na ferramenta um mecanismo de busca para os diagramas
7. Ser de fácil operação
8. Criar um banco de dados com todos estes históricos

Em posse dessas informações iniciamos a modelagem do banco de dados, utilizando a técnica de entidade e relacionamento e aplicando técnicas de normalização para evitar redundâncias e inconsistências na definição do banco.

Na primeira aproximação do banco de dados utilizamos o software MySQL Workbench é uma ferramenta gratuita e permite a edição de forma visual e através de comandos SQL a criação e o gerenciamento de bancos de dados no formato MySQL.

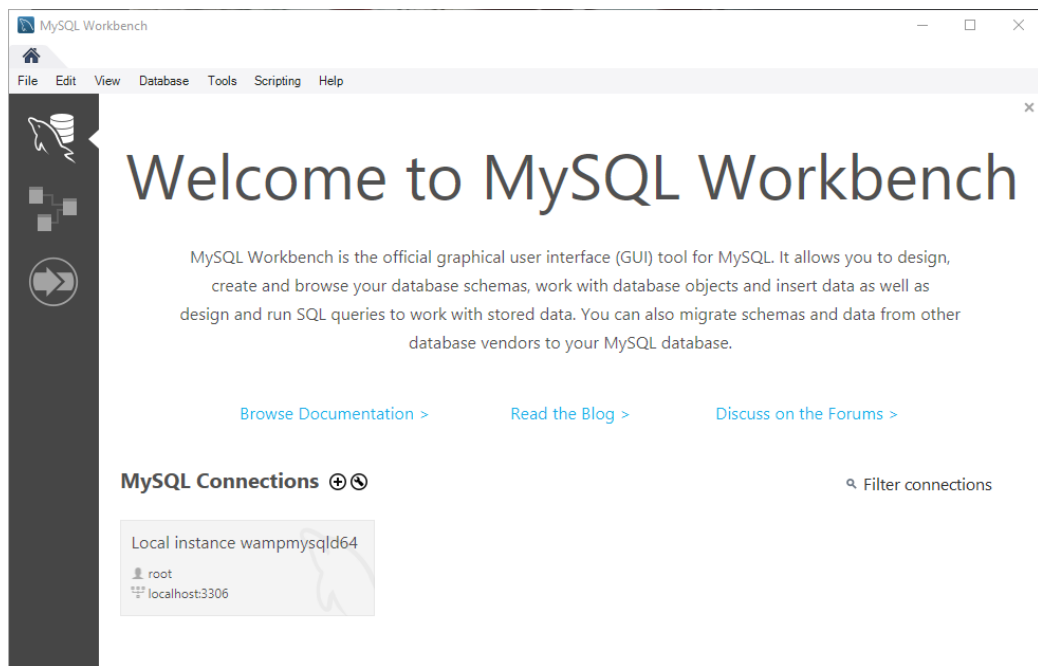


Figura 2: Tela inicial do MySQL Workbench. Fonte: Autor

Com base nos requisitos informados pelo cliente definiu-se o diagrama entidade relacionamento no modelo de Peter Chen abaixo:

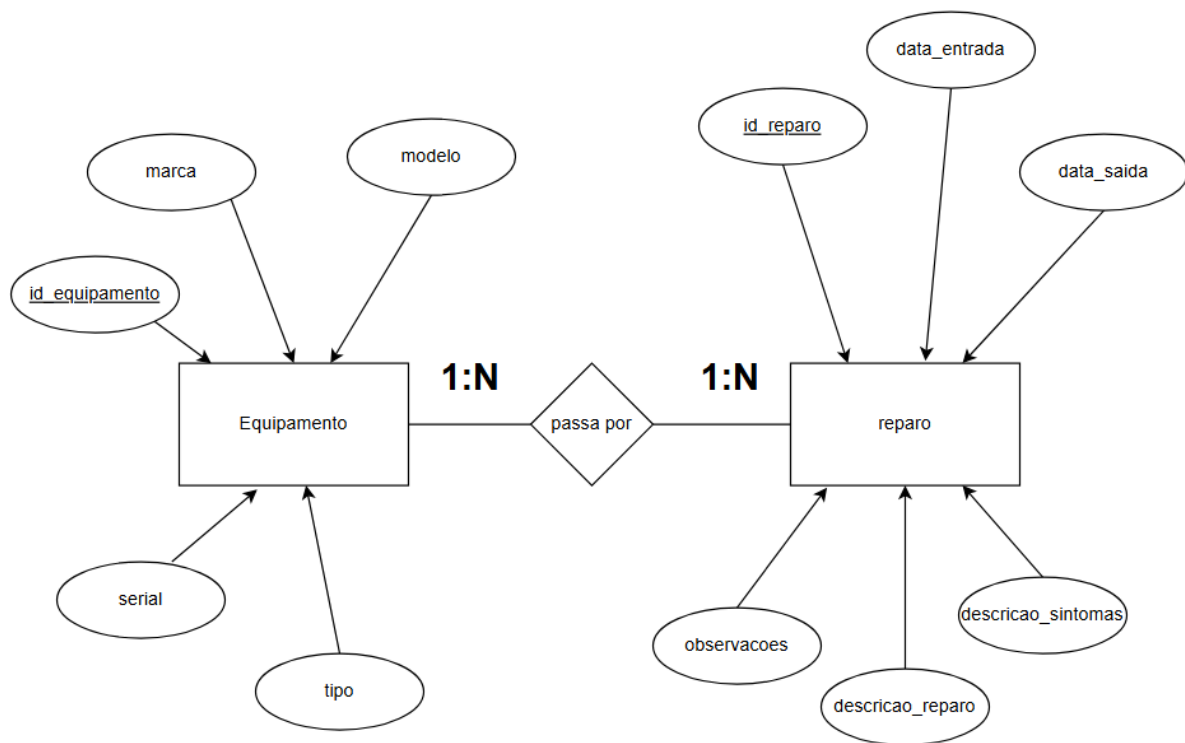


Figura 3: Diagrama Entidade-Relacionamento modelo de Peter Chen do banco de dados. Fonte: Autor

Após criação do diagrama segue-se a modelagem do banco de dados no MySQL Workbench.

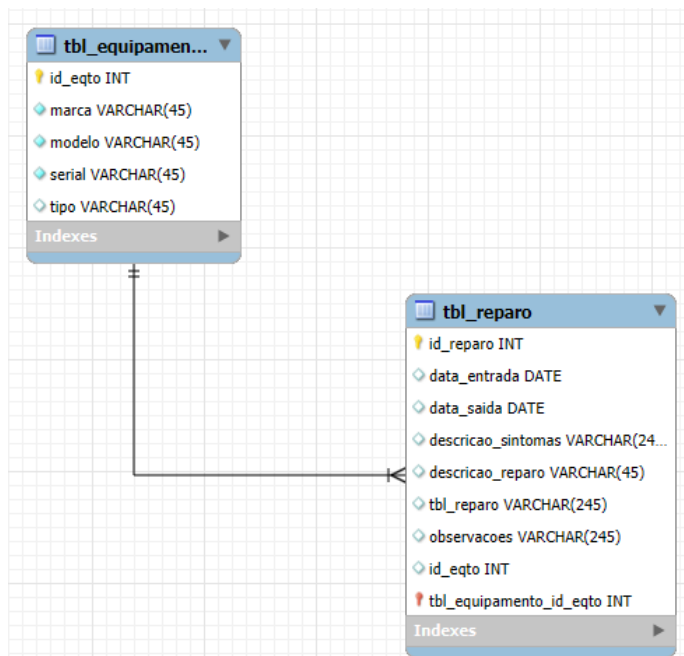


Figura 4: Primeira implementação do banco de dados. Fonte: Autor

Como observado a chave primária (PK) foi definida na tabela **tbl_equipamento** o atributo **id_eqto** e na tabela **tbl_reparo** esta chave é uma chave estrangeira (FK) determinando o relacionamento entre as entidades. Por sua vez na tabela de reparos **tbl_reparo** a chave primária foi definida no atributo **id_reparo**.

Sequencialmente cria-se a estrutura do banco de dados utilizando a linguagem SQL com comandos DDL (“Data Definition Language”) para criação do banco de dados e as respectivas tabelas, conforme definição do relacionamento ER (Entidade-Relacionamento).

```

1      -- Criação do banco de dados
2  •   CREATE DATABASE IF NOT EXISTS oficina_reparos_eletronicos;
3
4  •   USE oficina_reparos_eletronicos;
5
6      -- Tabela EQUIPAMENTO
7  •   CREATE TABLE IF NOT EXISTS equipamento (
8      id_equipamento INT AUTO_INCREMENT PRIMARY KEY,
9      descricao VARCHAR(50) NOT NULL,
10     marca VARCHAR(50) NOT NULL,
11     modelo VARCHAR(50) NOT NULL,
12     serial VARCHAR(50) UNIQUE,
13     tipo VARCHAR(30) NOT NULL,
14     INDEX idx_serial (serial),
15     INDEX idx_tipo (tipo)
16 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
17
18     -- Tabela REPARO
19  •   CREATE TABLE IF NOT EXISTS reparo (
20     id_reparo INT AUTO_INCREMENT PRIMARY KEY,
21     id_equipamento INT NOT NULL,
22     data_entrada DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
23     data_saida DATETIME,
24     descricao_sintomas LONGTEXT,
25     descricao_reparo LONGTEXT,
26     observacoes LONGTEXT,
27     FOREIGN KEY (id_equipamento) REFERENCES equipamento(id_equipamento)
28     ON DELETE CASCADE
29     ON UPDATE CASCADE,
30     INDEX idx_data_entrada (data_entrada),
31     INDEX idx_data_saida (data_saida)
32 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

```

Figura 5: Criação do banco de dados e tabelas. Fonte: Autor

A chave primária é do tipo inteira e auto-incrementada, o campo serial é do tipo único para evitar duplicações. Campos de texto que envolvem descrições de defeito, reparo e observações utiliza-se TEXTLONG.

Para geração de relatórios criamos uma “view”, para consultas futuras com informações resumidas para o usuário.

```

34      -- Criação de uma view para relatórios básicos
35 • CREATE VIEW vw_reparos_equipamentos AS
36 SELECT
37     r.id_reparo,
38     e.descricao,
39     e.marca,
40     e.modelo,
41     e.serial,
42     e.tipo,
43     r.data_entrada,
44     r.data_saida,
45     LENGTH(r.descricao_sintomas) AS tamanho_desc_sintomas,
46     LENGTH(r.descricao_reparo) AS tamanho_desc_reparo
47 FROM
48     equipamento e
49 JOIN
50     reparo r ON e.id_equipamento = r.id_equipamento;
51

```

Figura 6: Bloco SQL para criação das views. Fonte: Autor

Um procedimento foi adicionado para que seja feito o cálculo do tempo médio de reparo, além de triggers para evitar inserção de dados incorreta, por exemplo a data de saída ser inferior a data de entrada.

```

52      -- Procedimento para calcular tempo médio de reparo por tipo de equipamento
53 DELIMITER //
54 • CREATE PROCEDURE sp_tempo_medio_reparo()
55 BEGIN
56     SELECT
57         e.tipo,
58         AVG(TIMESTAMPDIFF(HOUR, r.data_entrada, r.data_saida)) AS tempo_medio_horas,
59         COUNT(*) AS total_reparos
60     FROM
61         reparo r
62     JOIN
63         equipamento e ON r.id_equipamento = e.id_equipamento
64     WHERE
65         r.data_saida IS NOT NULL
66     GROUP BY
67         e.tipo;
68 END //
69 DELIMITER ;

```

Figura 7: Procedimento para cálculo do tempo médio de reparo. Fonte: Autor

```

71 -- Trigger para verificar datas válidas
72 DELIMITER //
73 • CREATE TRIGGER tr_valida_datas_reparo
74 BEFORE INSERT ON reparo
75 FOR EACH ROW
76 BEGIN
77     IF NEW.data_saida IS NOT NULL AND NEW.data_saida < NEW.data_entrada THEN
78         SIGNAL SQLSTATE '45000'
79         SET MESSAGE_TEXT = 'Data de saída não pode ser anterior à data de entrada';
80     END IF;
81 END //
82 DELIMITER ;

```

Figura 8: Criação do trigger. Fonte: Autor

Após criação do banco de dados e execução de um servidor MySQL local através da DML (“Data Manipulation Language”) inserimos dois equipamentos para teste do banco e visualização tabela equipamento preenchida.

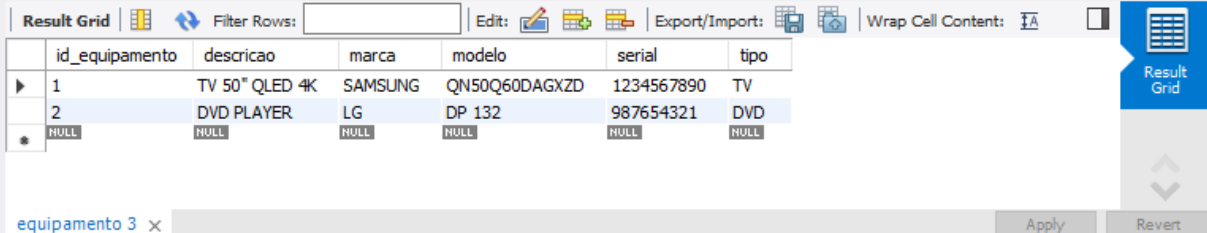
```

84 • -- Inserindo dados para teste local do banco
85
86 INSERT INTO equipamento
87 (id_equipamento, marca, descricao, modelo, serial, tipo)
88 VALUES
89 (DEFAULT, 'SAMSUNG', 'TV 50" QLED 4K', 'QN50Q60DAGXZD', '1234567890', 'TV' ),
90 (DEFAULT, 'LG', 'DVD PLAYER', 'DP 132', '987654321', 'DVD');
91
92 -- Mostrar tabela com valores
93
94 • SELECT * FROM equipamento;

```

Figura 9: Inserção de dados à tabela e comando para visualização. Fonte: Autor

Na parte inferior do MySQL Workbench é possível ver o preenchimento desta tabela.



	id_equipamento	descricao	marca	modelo	serial	tipo
▶	1	TV 50" QLED 4K	SAMSUNG	QN50Q60DAGXZD	1234567890	TV
	2	DVD PLAYER	LG	DP 132	987654321	DVD
*	NULL	NULL	NULL	NULL	NULL	NULL

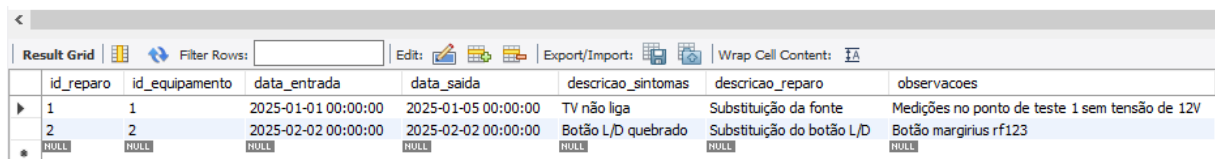
Figura 10: Tabela equipamento com preenchimento teste. Fonte: Autor.

Seguindo a sequência de testes foi feita a inserção de dois reparos, nos equipamentos cadastrados.

```
96 • INSERT INTO reparo
97 (id_reparo, id_equipamento, data_entrada, data_saida, descricao_sintomas, descricao_reparo, observacoes)
98 VALUES
99 (DEFAULT, '1', '2025-01-01', '2025-01-05', 'TV não liga', 'Substituição da fonte', 'Medições no ponto de teste 1 sem tensão de 12V'),
100 (DEFAULT, '2', '2025-02-02', '2025-02-02', 'Botão L/D quebrado', 'Substituição do botão L/D', 'Botão margirius rf123');
101
102 • SELECT * FROM reparo;
```

Figura 11: Inserção de dois reparos efetuados na tabela de reparos. Fonte: Autor.

Conforme imagem abaixo finalizamos o teste do banco de dados através da inserção de dados de reparo.

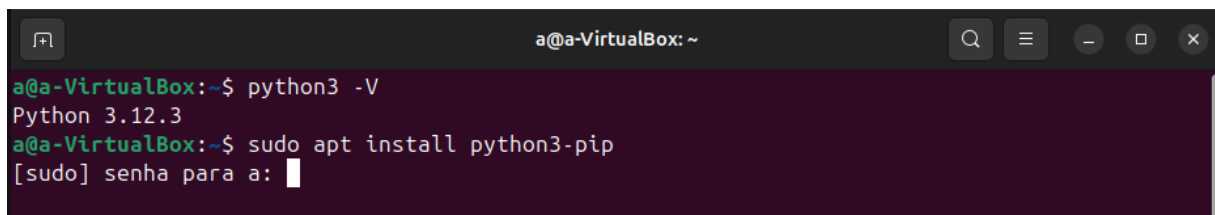


	id_reparo	id_equipamento	data_entrada	data_saida	descricao_sintomas	descricao_reparo	observacoes
1	1	1	2025-01-01 00:00:00	2025-01-05 00:00:00	TV não liga	Substituição da fonte	Medições no ponto de teste 1 sem tensão de 12V
2	2	2	2025-02-02 00:00:00	2025-02-02 00:00:00	Botão L/D quebrado	Substituição do botão L/D	Botão margirius rf123

Figura 12: Tabela reparo com atributos preenchidos. Fonte: Autor.

Após testes no banco de dados inicia-se a produção do sistema de “*backend*” utilizando o “*framework*” Django, segue-se as configurações necessárias para execução e geração da estrutura do projeto.

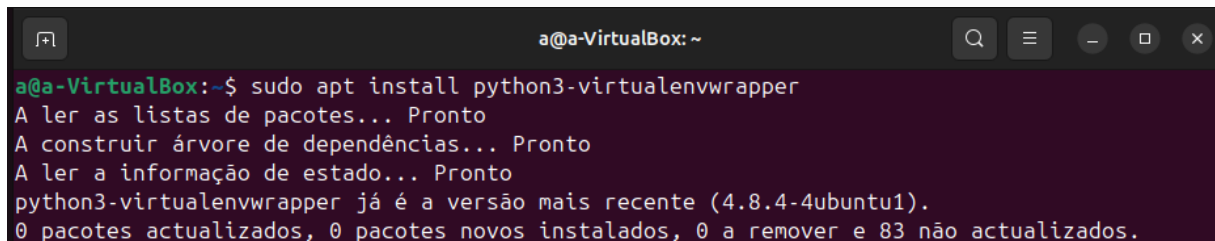
Inicialmente verificamos a versão do Python instalada em nosso sistema operacional e instalação do PIP (“*Python Package Index*”) que permite instalações de pacotes para o Python.



```
a@a-VirtualBox: ~
a@a-VirtualBox:~$ python3 -V
Python 3.12.3
a@a-VirtualBox:~$ sudo apt install python3-pip
[sudo] senha para a: 
```

Figura 13: Verificação da versão do Python e instalação do PIP. Fonte: Autor.

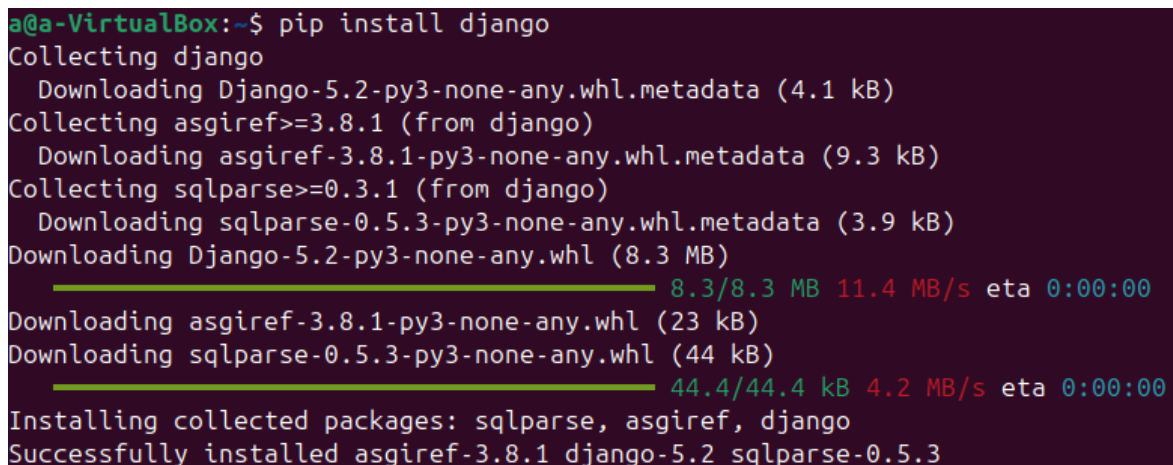
Depois da instalação do PIP devemos instalar o ambiente virtual de desenvolvimento.



```
a@a-VirtualBox: ~  
a@a-VirtualBox:~$ sudo apt install python3-virtualenvwrapper  
A ler as listas de pacotes... Pronto  
A construir árvore de dependências... Pronto  
A ler a informação de estado... Pronto  
python3-virtualenvwrapper já é a versão mais recente (4.8.4-4ubuntu1).  
0 pacotes actualizados, 0 pacotes novos instalados, 0 a remover e 83 não actualizados.
```

Figura 14: Instalação do ambiente virtual para desenvolvimento. Fonte: Autor.

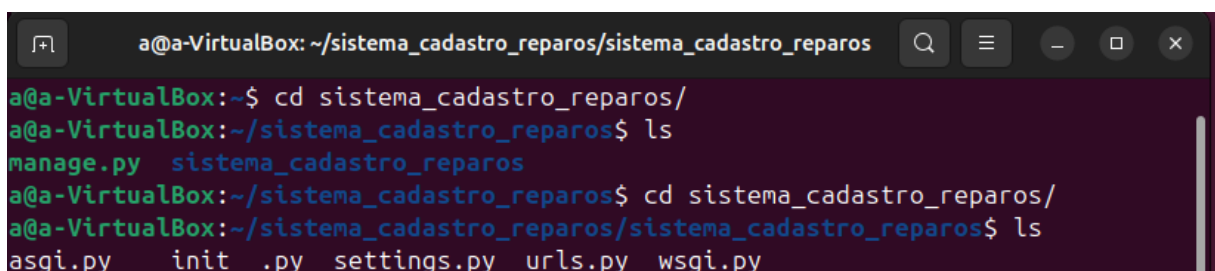
Sequencialmente executamos a instalação do “framework” Django ao sistema.



```
a@a-VirtualBox:~$ pip install django  
Collecting django  
  Downloading Django-5.2-py3-none-any.whl.metadata (4.1 kB)  
Collecting asgiref>=3.8.1 (from django)  
  Downloading asgiref-3.8.1-py3-none-any.whl.metadata (9.3 kB)  
Collecting sqlparse>=0.3.1 (from django)  
  Downloading sqlparse-0.5.3-py3-none-any.whl.metadata (3.9 kB)  
Downloading Django-5.2-py3-none-any.whl (8.3 MB)  
 8.3/8.3 MB 11.4 MB/s eta 0:00:00  
Downloading asgiref-3.8.1-py3-none-any.whl (23 kB)  
Downloading sqlparse-0.5.3-py3-none-any.whl (44 kB)  
 44.4/44.4 kB 4.2 MB/s eta 0:00:00  
Installing collected packages: sqlparse, asgiref, django  
Successfully installed asgiref-3.8.1 django-5.2 sqlparse-0.5.3
```

Figura 15: Instalação do framework Django. Fonte: Autor

Com o ambiente configurado através do Django criamos a estrutura básica de nosso através dos comandos abaixo:



```
a@a-VirtualBox: ~/sistema_cadastro_reparos/sistema_cadastro_reparos  
a@a-VirtualBox:~$ cd sistema_cadastro_reparos/  
a@a-VirtualBox:~/sistema_cadastro_reparos$ ls  
manage.py sistema_cadastro_reparos  
a@a-VirtualBox:~/sistema_cadastro_reparos$ cd sistema_cadastro_reparos/  
a@a-VirtualBox:~/sistema_cadastro_reparos/sistema_cadastro_reparos$ ls  
asgi.py __init__.py settings.py urls.py wsgi.py
```

Figura 16: Criação da estrutura utilizando o Django. Fonte: Autor.

A estrutura em questão define:

- **manage.py**: é o “Script” utilizado para criar as aplicações, operações com banco de dados e iniciar o web server de desenvolvimento.

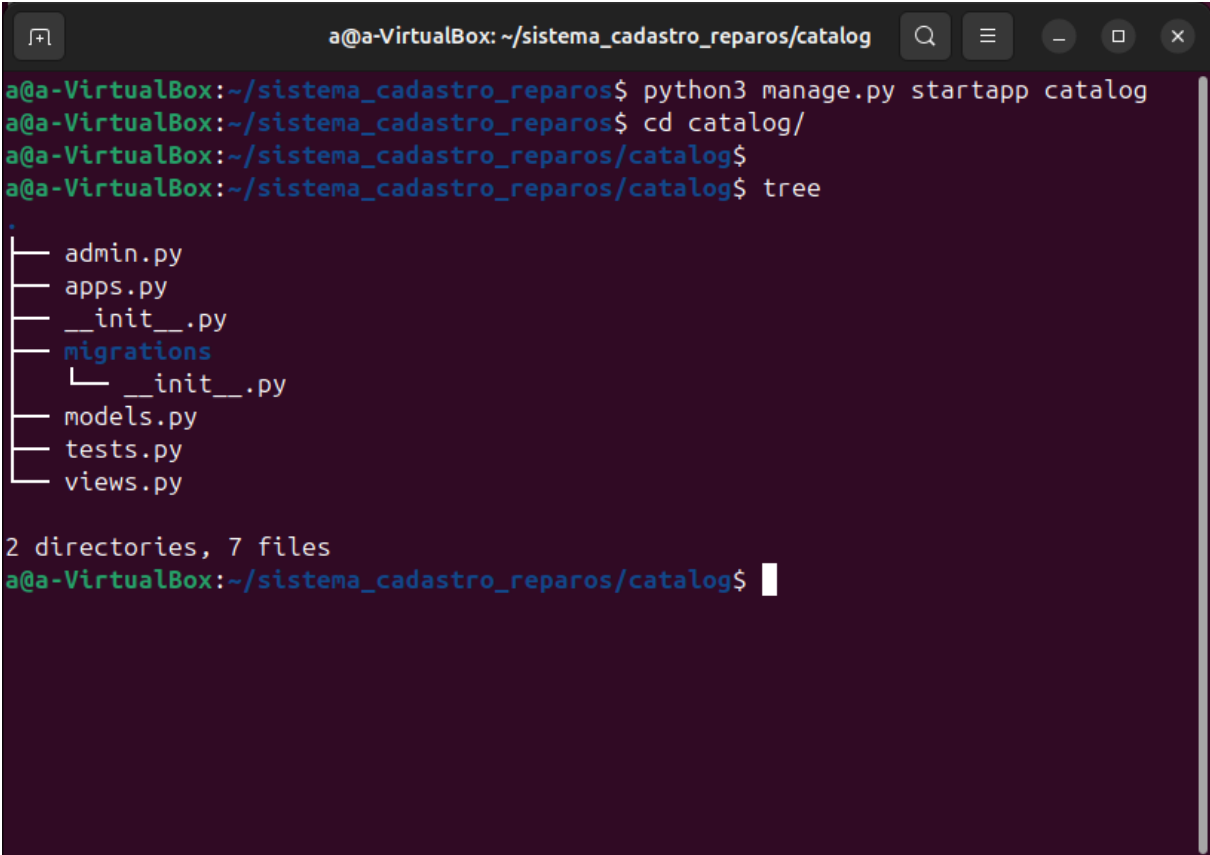
- **`__init__.py`**: arquivo em branco que instrui o Python para que o diretório seja tratado como um pacote Python.

- **`settings.py`**: contém as definições do website, contém a localização de arquivos estáticos, registro de aplicação, configuração do banco de dados, etc.

- **`urls.py`**: define os mapeamentos das URLs (“Uniform Resource Locator”) que permite localizar o conteúdo de um site também delega o mapeamento de aplicativos específicos.

- **`wsgi.py`**: auxilia na comunicação do aplicativo Django ao web server.

Na sequência devemos criar o catálogo da aplicação utilizando o “Script” **`manage.py`**.



```
a@a-VirtualBox: ~/sistema_cadastro_reparos/catalog
a@a-VirtualBox:~/sistema_cadastro_reparos$ python3 manage.py startapp catalog
a@a-VirtualBox:~/sistema_cadastro_reparos$ cd catalog/
a@a-VirtualBox:~/sistema_cadastro_reparos/catalog$ tree
.
├── admin.py
├── apps.py
├── __init__.py
├── migrations
│   └── __init__.py
├── models.py
├── tests.py
└── views.py

2 directories, 7 files
a@a-VirtualBox:~/sistema_cadastro_reparos/catalog$
```

Figura 17: Criação do catálogo da aplicação e estrutura da pasta. Fonte: Autor.

No caminho: `~/sistema_cadastro_reparos/sistema_cadastro_reparos/settings.py`, efetuamos o registro da aplicação no projeto, aqui é o ponto onde podemos adicionar elementos que farão parte do sistema. Neste ponto especificamos o objeto de configuração do aplicativo.

```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'catalog.apps.CatalogConfig', #Acréscimo
]
```

Figura 18: Inserção de elementos à aplicação. Fonte: Autor.

Neste mesmo arquivo determinamos o banco de dados utilizado, adicionaremos o MySQL ao mesmo arquivo *settings.py*.

```
# Database
# https://docs.djangoproject.com/en/5.2/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'oficina_reparos_eletronicos', # Nome do banco de dados MySQL
        'USER': 'root', # Usuário do MySQL
        'PASSWORD': '', # Senha do MySQL
        'HOST': 'localhost', # Ou o IP do servidor MySQL (ex: '127.0.0.1')
        'PORT': '3306', # Porta padrão do MySQL
        'OPTIONS': {
            'init_command': "SET sql_mode='STRICT_TRANS_TABLES'", # Modo estrito para
            evitar dados inválidos
            'charset': 'utf8mb4', # Suporte a emojis e caracteres especiais
        },
    }
}
```

Figura 19: Inclusão das configurações do MySQL no arquivo settings.py. Fonte: Autor.

Conectar o “mapeador” de URL é o próximo passo, esta configuração determina quais serão as URLs que farão parte do sistema e seus padrões. Adicionamos novos caminhos (“*paths*”) que estarão presentes no arquivo de catálogo.

O Django não vem com serviços de conteúdos estáticos, tais como: CSS, *JavaScript* e imagens, portanto devemos adicionar o mapeamento de conteúdos estáticos para o “*web server*”.

```
24 # Conjunto de linhas Adicionadas
25
26 # Use include() to add paths from the catalog application
27 from django.conf.urls import include
28 from django.urls import path
29
30 urlpatterns += [
31     path('catalog/', include('catalog.urls')),
32 ]
33
34 #Add URL maps to redirect the base URL to our application
35 from django.views.generic import RedirectView
36 urlpatterns += [
37     path('', RedirectView.as_view(url='/catalog/')),
38 ]
39
40 # Use static() to add url mapping to serve static files during development (only)
41 from django.conf import settings
42 from django.conf.urls.static import static
43
44 urlpatterns += static(settings.STATIC_URL, document_root=settings.STATIC_ROOT)
45
```

Figura 20: Configurando as URLs e redirecionando para o catálogo. Fonte: Autor.

Criamos um arquivo **urls.py** na pasta **~/catalog** para inserção dos padrões URL utilizados.



```
Abrir  v  [í]  urls.py  Gravar  [≡]  [–]  [□]  [×]
~/sistema_cadastro_reparos/catalog

1 from django.urls import path
2 from catalog import views
3
4 urlpatterns = [
5
6
7 ]
```

Figura 21: Arquivo urls.py padrões URL. Fonte: Autor.

O processo de configuração foi encerrado, efetua-se as migrações para que a estrutura seja adequada aos novos parâmetros descritos nos arquivos de configuração, este processo é obtido por:

- **python3 manage.py makemigrations**

- python3 manage.py migrations

Migrações feitas iniciamos a concepção do projeto, para a criação do “*frontend*” com base no HTML e CSS, criação dos formulários e configurações complementares.

O HTML (“*HyperText Markup Language*”) é a linguagem de marcação utilizada para estruturar o conteúdo da web. Ele define a estrutura das páginas web, permitindo organizar textos, imagens, links, vídeos e outros elementos de forma hierárquica. Serve para criar e estruturar páginas da internet, permitindo que os navegadores apresentem o conteúdo de maneira organizada e legível para os usuários. O estilo visual será definido pelo CSS (“*Cascading Style Sheet*”).

Para a edição e depuração da codificação em HTML e CSS utilizamos o Visual Studio Code, que é uma ferramenta gratuita e multiplataforma disponibilizada pela Microsoft.

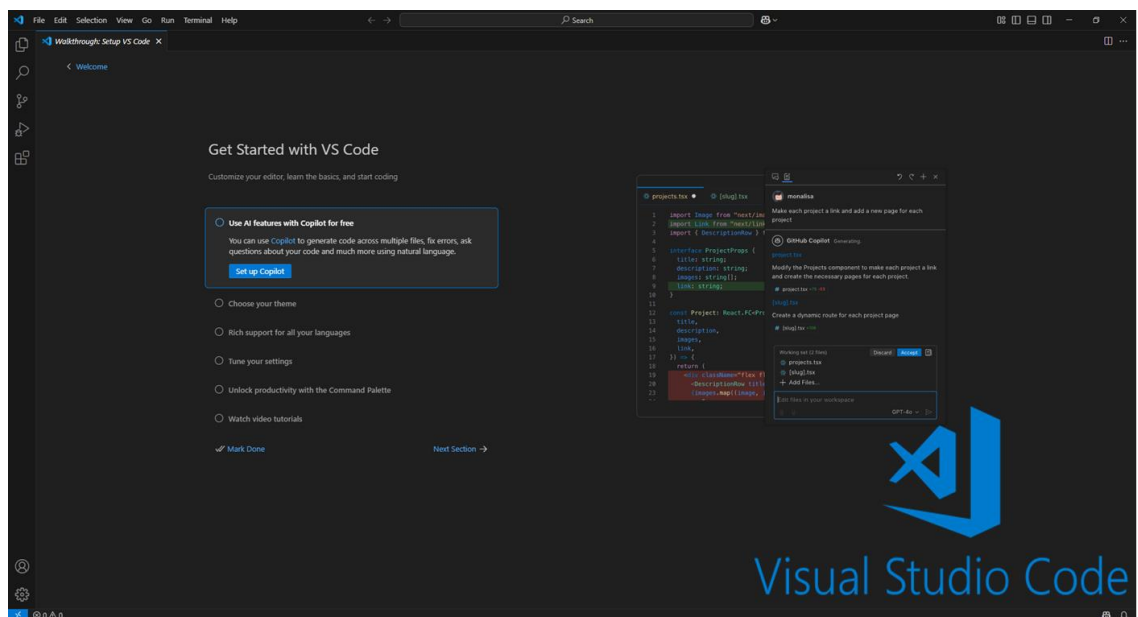


Figura 22: Visual Studio Code. Fonte: Autor.

Inicialmente determinamos uma estrutura básica de páginas para o protótipo, que receberá melhorias futuras a estrutura deve seguir o seguinte conjunto de páginas web:

- index.html: Página inicial do projeto, apresenta todas opções disponíveis cadastros, consulta, “esquemário” e uma página sobre o projeto.

- cadastro.html : nesta página é feito o cadastro dos equipamentos e manutenções realizadas.

- consulta.html : responsável por responder às consultas do usuário as informações armazenadas no banco de dados.

- sobre.html : Apresenta o projeto e os integrantes da equipe que participaram da elaboração.

O resultado desta etapa é apresentado abaixo, demais detalhes e conexões apresentamos nos resultados preliminares.

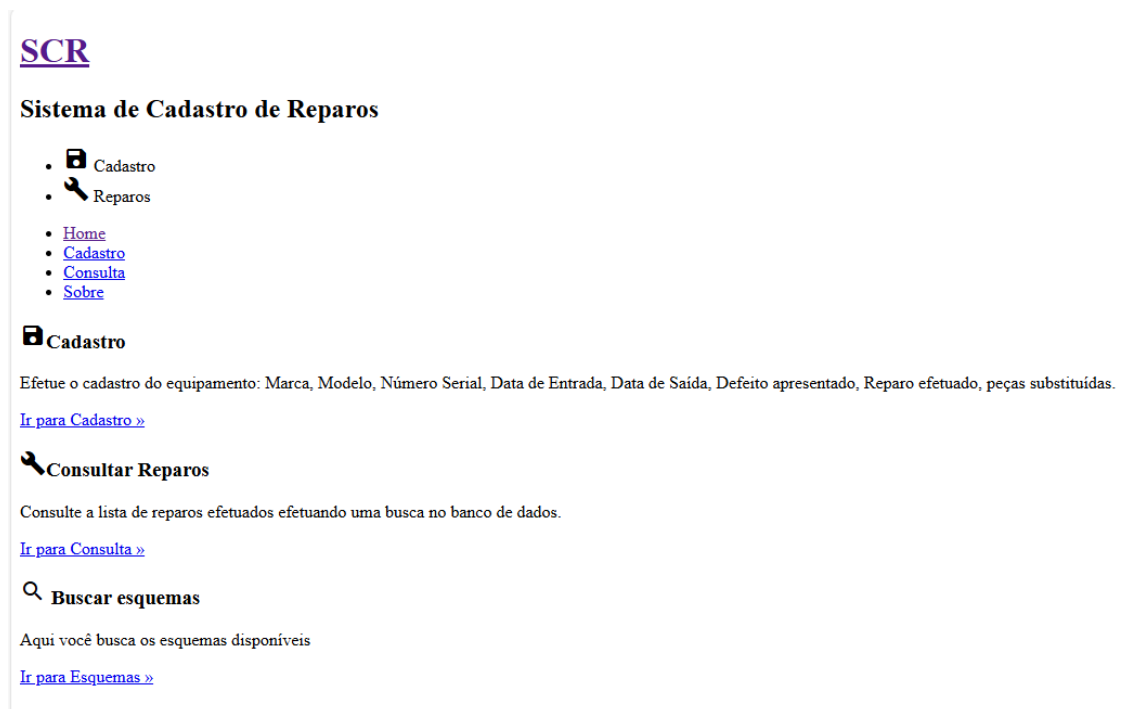


Figura 23: Layout da página inicial do projeto. Fonte Autor.

A edição em HTML segue as premissas da linguagem, estrutura-se as informações da página e seus conteúdos.

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3
4 <head>
5     <title>Sistema de Cadastro de Reparos</title>
6     <meta charset="UTF-8">
7     <link rel="stylesheet" type="text/css" href="css/geral.css">
8     <link rel="stylesheet" href="http://fonts.googleapis.com/icon?family=Material+Icons">
9 </head>
10
11 <body>
12     <!--cabeçalho-->
13     <div id="topo">
14         <header id="header">
15             <hgroup>
16                 <h1><a href="index.html">SCR</a></h1>
17                 <h2>Sistema de Cadastro de Reparos</h2>
18             </hgroup>
19             <div id="header-contato">
20                 <ul>
21                     <li><i class="material-icons">save</i> Cadastro </li>
22                     <li><i class="material-icons">build</i> Reparos </li>
23                 </ul>
24             </div>
25         </header>
26     </div>
27     <!--fim cabeçalho-->
28
29     <!--início do menu-->
30     <nav id="topnav">
31         <ul>
32             <li class="active"><a href="index.html" title="Página Inicial">Home</a></li>
33             <li><a href="cadastro.html" title="Cadastro">Cadastro</a> </li>
34             <li><a href="consulta.html" title="Consulta">Consulta</a></li>
35             <li><a href="sobre.html" title="Sobre">Sobre</a></li>
36         </ul>
37     </nav>
38     <!--fim do menu-->

```

Figura 24: Trecho do código em HTML. Fonte Autor.

O arquivo CSS foi criado em arquivo externo, todas as configurações estéticas do site estão neste arquivo, definindo uma apresentação visual mais agradável, funcional e objetiva.

```

# geral.css 1 x
# geral.css > ...
1 /*----- Folha de estilos -----*/
2
3 body{
4     margin: 0;
5     padding: 0;
6     font-size: 14px;
7     color: #919191;
8     background-color: #232323;
9     font-family: Arial, Helvetica, sans-serif;
10 }
11
12
13 h1, h2, h3, h4, h5, h6{
14     margin: 0 0 30px 0;
15     font-size: 1.8em;
16     font-family: "CavaliarDreamsBold", Arial, Helvetica, sans-serif;
17     font-weight: normal;
18 }
19
20
21 /*----- Header -----*/
22

```

Figura 25: Apresentação de trecho do código CSS do projeto. Fonte: Autor.

Após implementação do arquivo CSS pode-se observar na imagem abaixo o aspecto da página inicial.

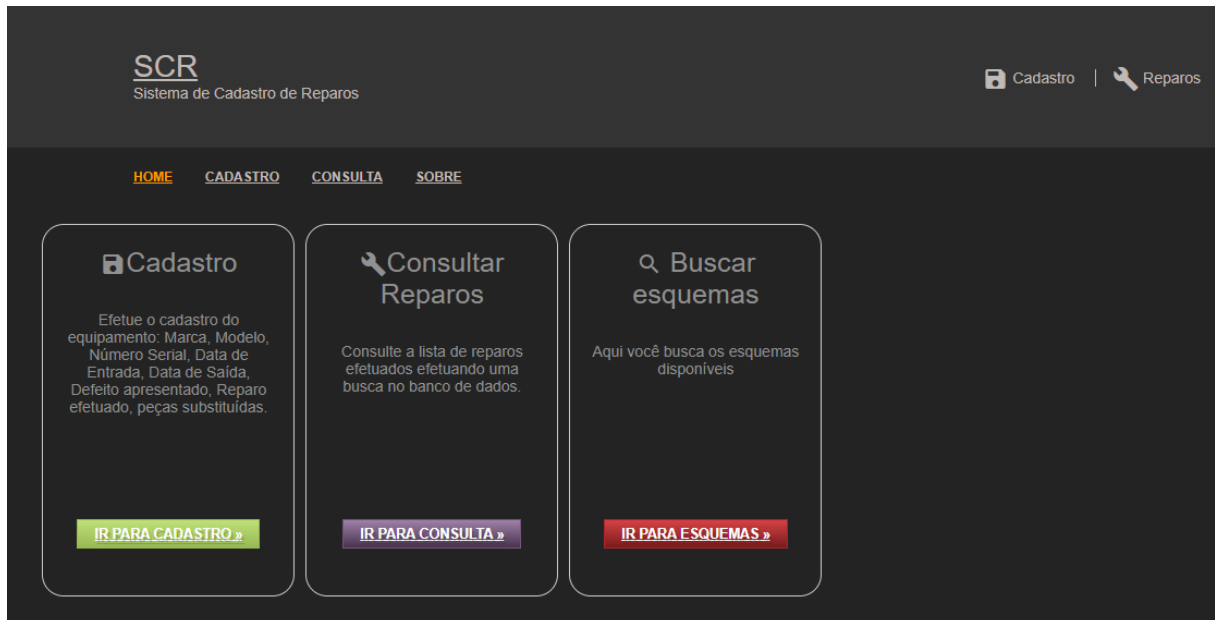


Figura 26: Imagem da tela inicial do projeto. Fonte: Autor.

A partir deste ponto são necessárias a codificação dos arquivos criados pelo Django que será responsável pela conexão do banco de dados e a apresentação web. Maiores detalhes são apresentados no github do projeto, assim como a estrutura e a codificação.

Neste ponto da implementação estamos em fase de correção das funcionalidades do Django com o MySQL e a conexão com o “*frontend*”, os resultados mostram-se promissores.

Porém ainda são necessárias correções no código e melhorias da formatação do site, para ir de encontro aos requisitos determinados pelo cliente.

2.5 RESULTADOS PRELIMINARES: SOLUÇÃO INICIAL

Os resultados apresentados podem ser vistos em ordem da visão do cliente “*frontend*” a partir das páginas web até a interligação com o “*backend*” criado com o Django.

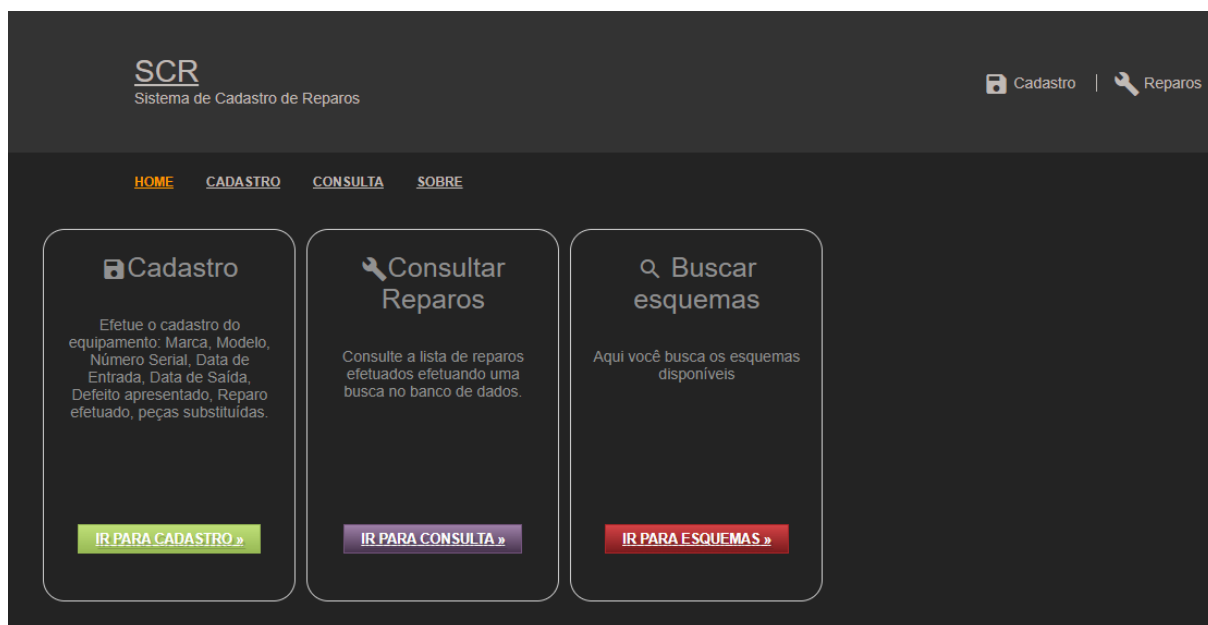


Figura 27: Imagem da tela inicial do projeto. Fonte: Autor.

Após clicar em [IR PARA CADASTRO] a tela abaixo é apresentada, neste ponto o usuário preenche as informações relativas ao processo de manutenção e os equipamentos envolvidos.

Figura 28: Preenchimentos relativos ao equipamento. Fonte: Autor.

Form for recording repair details. It consists of three stacked text input fields and two buttons at the bottom.

- Top field:** Titled "Descrição do defeito" with a sub-label "Sintoma(s)/Defeito(s)". The placeholder text is "Descreva de forma objetiva os sintomas/defeitos do equipamento em questão".
- Middle field:** Titled "Descrição do reparo / peças substituídas" with a sub-label "Reparo". The placeholder text is "Descreva de forma objetiva o reparo efetuado".
- Bottom field:** Titled "Observações sobre reparo/equipamento" with a sub-label "Observações". The placeholder text is "Faça observações sobre o reparo efetuado ou equipamento em questão".
- Buttons:** At the bottom are two buttons: "CADASTRAR" (red) and "LIMPAR" (orange).

Figura 29: Campos para o preenchimento dos detalhes do reparo. Fonte: Autor.

Após o preenchimento dos campos, o usuário clica em [CADASTRAR] para que os dados inseridos sejam inseridos no banco de dados.

Criamos também uma página sobre a produção deste projeto integrador, como segue:



Figura 30: Página web sobre o projeto integrador. Fonte: Autor.

Utilizamos o GitHub para controle de versão, onde o projeto de forma colaborativa está em processo de produção, o endereço do GitHub é:

<https://github.com/Mike301014/PI2025-1S>

O projeto segue em andamento para novas implementações e correções de pontos que não apresentam funcionamento adequado à proposta.

REFERÊNCIAS

- ANCILLOTTI, L. Por que o Brasil precisa regulamentar o direito ao reparo? **Jusbrasil**, 2024. Disponível em: <https://www.jusbrasil.com.br/artigos/por-que-o-brasil-precisa-regulamentar-o-direito-ao-reparo/2119644291?utm_source=chatgpt.com>. Acesso em: 13 mar. 2025.
- DECON - DEPARTAMENTO DE ECONOMIA ABINEE. **comportamento da indústria elétrica e eletrônica em 2024**. São Paulo. 2024.
- IEEE. The Top Programming Languages 2024. **IEEE Spectrum**, 22 ago 2024. Disponível em: <<https://spectrum.ieee.org/top-programming-languages-2024>>.
- KOZIMA, M. H. M. V. **A importância da manutenção em equipamentos**. Bauru: Faculdade de Tecnologia FATEC, 2017.
- MARQUES, A. C. N. **Avaliação da efetividade da manutenção preventiva na redução de custos e maior disponibilidade de equipamentos biomédicos: uma revisão sistemática da literatura**. Universidade Federal de São Paulo. São Paulo. 2024.
- MEIRELLES, F. S. **Pesquisa de uso de TI | FVG EAESP**. São Paulo. 2024.
- MOZILLA. CSS - Glossário do MDN Web Docs. **Mozilla**, 12 mar 2025. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Glossary/CSS>>.
- MOZILLA. Django Web Framework (Python) - Aprendendo desenvolvimento web | MDN. **https://developer.mozilla.org/**, 10 Mar 2025. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn_web_development/Extensions/Server-side/Django>.
- MOZILLA. HTML: Linguagem de Marcação de Hipertexto. **mozilla.org/**, 12 mar 2025. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/HTML>>.
- ONU - ORGANIZAÇÃO DAS NAÇÕES UNIDAS. Produção de lixo eletrônico pela humanidade chegou a 62 milhões de toneladas. **ONU News**, 22 mar. 2024. Disponível em: <<https://news.un.org/pt/story/2024/03/1829466>>.
- ORACLE. MySQL: Entendendo o que é e como é usado. **https://www.oracle.com/**, 29 ago 2024. Disponível em: <<https://www.oracle.com/br/mysql/what-is-mysql/>>.
- PRATES, G. A.; OSPINA, M. T. Tecnologia da informação em pequenas empresas: fatores de êxito, restrições e benefícios. **Revista de Administração Contemporânea**, 25 Mar 2009.
- TANENBAUM, A. S.; WETHERALL, D. J. **Redes de computadores**. 5. ed. São Paulo: Pearson, 2011.