UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Sistema SCADA Integrando *Digital Twins* para Gestão de Armazenamento de Energia

PAULO ANTONIO JORDÃO COUTO

DM 22/2025

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2025

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

PAULO ANTONIO JORDÃO COUTO

Sistema SCADA Integrando *Digital Twins* para Gestão de Armazenamento de Energia

Dissertação submetida à Banca Examinadora do Programa de Pós-Graduação em Engenharia Elétrica da UFPA para obtenção do Grau de Mestre em Engenharia Elétrica na Área de Sistemas de Energia Elétrica.

Orientador: Prof. Dr. WELLINGTON DA SILVA FONSECA

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2025

Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISBD
Sistema de Bibliotecas da Universidade Federal do Pará
Gerada automaticamente pelo módulo Ficat, mediante os dados fornecidos pelo(a) autor(a)

C871s Couto, Paulo Antonio Jordão.
Sistema SCADA Integrando Digital Twins para Gestão de Armazenamento de Energia / Paulo Antonio Jordão Couto. — 2025.
94 f. : il. color.

Orientador(a): Prof. Dr. Wellington da Silva Fonseca
Dissertação (Mestrado) - Universidade Federal do Pará, Instituto de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica, Belém, 2025.

1. Gêmeos Digitais. 2. Armazenamento de Energia. 3. SCADA. 4. Internet das Coisas. 5. Baterias. I. Título.

CDD 621.310285

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

"SISTEMA SCADA INTEGRANDO *DIGITAL TWINS* PARA GESTÃO DE ARMAZENAMENTO DE ENERGIA"

AUTOR: **PAULO ANTONIO JORDÃO COUTO**

DISSERTAÇÃO DE MESTRADO SUBMETIDA À BANCA EXAMINADORA APROVADA PELO COLEGIADO DO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA, SENDO JULGADA ADEQUADA PARA A OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA ELÉTRICA NA ÁREA DE SISTEMAS DE ENERGIA ELÉTRICA.

APROVADA EM: 01/08/2025

BANCA EXAMINADORA:

Prof. Dr. Wellington da Silva Fonseca
(Orientador – PPGEE/ITEC/UFPA)

Prof. Dr. Marcos César da Rocha Seruffo
(Avaliador Interno - PPGEE/ITEC/UFPA)

Prof. Dr. Thiago Mota Soares
(Avaliador Interno – PPGEE/ITEC/UFPA)

Prof. Dr. Antonio Roniel Marques de Sousa
(Avaliador Externo ao Programa – SESI-PA)

VISTO:

Prof. Dr. Diego Lisboa Cardoso
(Coordenador do PPGEE/ITEC/UFPA)

Dedico este trabalho à minha amada mãe Francilene Jordão, à minha noiva Jéssica, ao meu irmão Matheus, a minha avó Francisca (in memorian), ao meu pai Pedro (in memorian), as minhas tias Francisca e Angélica, ao meu avô Antônio, ao meu padrasto Maurício e a todos os meus familiares e amigos. Esta conquista é de todos nós.

Agradecimentos

A Deus, criador de tudo e de todos, a Ele toda a honra e toda glória.

A minha família que sempre meu deu apoio e sustentação para que eu pudesse desenvolver este trabalho, em especial a minha amada mãe, grande incentivadora de que eu buscasse à educação, além de me incentivar a ser uma pessoa melhor. Minha noiva Jéssica Leite, por estar sempre ao meu lado e me apoiar incondicionalmente.

Ao CEAMAZON, que se tornou minha casa ao longo destes 2 anos, a qual me orgulho muito em fazer parte, pois foi aqui que pude melhorar academicamente e pessoalmente. Sou muito grato por ter sido acolhido e por construir amizades extremamente sensacionais, as quais sempre tive muito apoio e considero minha família. Agradeço em especial, ao Prof. Dr. Wellington Fonseca que foi meu pai ao longo desta jornada, sempre buscando o melhor para o meu desenvolvimento e sendo uma inspiração, ao Prof. Dr. Thiago Mota que tenho muita gratidão, respeito e carinho, à Profa. Dra. Flávia Monteiro que sempre me incentivou a seguir com os estudos, ao Prof. Dr. Josivan Reis que tenho enorme consideração, carinho e respeito e à Profa. Dra. Maria Emília Tostes por me receber neste centro de excelência.

Aos queridos amigos que fazem desta jornada, Iago, Aleqssandro, Henrique, Cezar, Roniel, Damásio, Marcos e João. As minhas amigas queridas, Priscila, Íris, Jhennifer, Kaylane, Jamily, Emanuele e Yasmim. Obrigado pela maravilhosa amizade de todos, pelas conversas e brincadeiras. Agradeço também a todos os amigos de Oriximiná, em especial da UFOPA, Orixinet e Pebol. A toda equipe do projeto do SIGDEE que proporcionou a implementação deste trabalho, em especial a Priscila Lobato e ao Vitor Batista pelas contribuições.

Por fim, agradeço demasiadamente ao Programa de Pós Graduação em Engenharia Elétrica (PPGEE) da UFPA, ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e Norte Energia S.A. pelo apoio e fomento ao trabalho desenvolvido.

“Se chorei ou se sorri, o importante é que emoções eu vivi”

Roberto Carlos

Resumo

A expansão do acesso à energia elétrica em regiões remotas é um desafio significativo para o desenvolvimento sustentável de comunidades isoladas. A dificuldade de integração dessas localidades à rede elétrica nacional, seja por questões geográficas, logísticas ou econômicas, tem impulsionado a busca por soluções tecnológicas que garantam um fornecimento de energia confiável, eficiente e ambientalmente responsável. Nesse contexto, microrredes híbridas com fontes renováveis, como a fotovoltaica, associadas a sistemas de armazenamento de energia, representam uma alternativa para esse cenário. Contudo, a gestão eficiente dessas microrredes ainda enfrenta obstáculos relevantes. Entre eles, destaca-se a escassez de ferramentas digitais que permitam o monitorar, controlar e prever seu desempenho *on-line*, especialmente com os desafios técnicos e estruturais característicos de regiões isoladas. Além disso, há uma lacuna significativa quanto à adoção de tecnologias como Gêmeos Digitais e sistemas SCADA atuais em aplicações práticas voltadas a esse contexto, o que limita o aproveitamento pleno do potencial dessas soluções. Motivado por essa realidade, este trabalho tem como objetivo o desenvolvimento de uma plataforma de software livre, denominada SIGDEE – Sistema Inteligente de Gestão e Despacho de Energia Elétrica. A aplicação busca integrar tecnologias como Gêmeos Digitais, Internet das Coisas (IoT) e sistemas SCADA, com uma solução robusta, escalável e acessível para supervisão e controle de sistemas energéticos distribuídos. A aplicação em desenvolvimento visa possibilitar o acompanhamento de simulações e análises de desempenho de sistemas de armazenamento de energia, apoiar a tomada de decisão e contribuir para a melhoria da eficiência energética em comunidades de infraestrutura limitada.

Palavras-chave: Gêmeos Digitais, Armazenamento de Energia, SCADA, Internet das Coisas, Baterias, Supercapacitores, Microrredes Híbridas, Sistemas Isolados.

Abstract

The expansion of access to electricity in remote regions is a significant challenge for the sustainable development of isolated communities. The difficulty in integrating these areas into the national power grid—due to geographical, logistical, or economic constraints—has driven the search for technological solutions that ensure reliable, efficient, and environmentally responsible energy supply. In this context, hybrid microgrids powered by renewable sources, such as photovoltaics, combined with energy storage systems, represent a promising alternative. However, the efficient management of such microgrids still faces considerable obstacles. Among them, a key issue is the lack of digital tools capable of monitoring, controlling, and predicting their performance online, especially considering the technical and structural challenges typical of remote regions. Moreover, there is a significant gap in the adoption of technologies such as Digital Twins and modern SCADA systems in practical applications tailored to this context, which limits the full potential of these solutions. Motivated by this scenario, the present work aims to develop a free and open-source software platform, named SIGDEE – Intelligent Energy Management and Dispatch System. The application seeks to integrate emerging technologies such as Digital Twins, the Internet of Things (IoT), and SCADA systems into a robust, scalable, and accessible solution for the supervision and control of distributed energy systems. The application under development aims to enable the monitoring of simulations and performance analyses of energy storage systems, support decision-making, and contribute to improving energy efficiency in communities with limited infrastructure.

Key-words: Digital Twins, Energy Storage, SCADA, Internet of Things, Batteries, Supercapacitors, Hybrid Microgrids, Isolated Systems.

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

ANEEL	Agência Nacional de Energia Elétrica
API	*Application Programming Interface*
BEN	Balanço Energético Nacional
CO2	Dióxido de Carbono
BMS	*Battery Management System*
CEAMAZON	Centro de Excelência em Eficiência Energética da Amazônia
CRUD	*Create, Read, Update, Delete*
CSS	*Cascading Style Sheets*
DERs	*Distributed Energy Resources*
DT	*Digital Twin*
DTs	*Digital Twins*
DTA	*Digital Twin Aggregate*
DTI	*Digital Twin Instance*
DTP	*Digital Twin Prototype*
EIA	*Energy Information Administration*
EPE	Empresa de Pesquisa Energética
GUI	*Graphical User Interface*
HTML	*HyperText Markup Language*
HTTP	*HyperText Transfer Protocol*
IoT	*Internet of Things*
KVM	*Kernel-based Virtual Machine*
LXC	*Linux Containers*
MER	Modelo Entidade Relacionamento

MME	Ministério de Minas e Energia
O&M	Operação e Manutenção
OIEE	Oferta Interna de Energia Elétrica
ODS	Objetivos de Desenvolvimento Sustentável
ORM	*Object-Relational Mapping*
REST	*Representational State Transfer*
RSE	Resistência Série Equivalente
SC	Supercapacitor
SCADA	Sistema de Supervisão e Aquisição de Dados
SIGDEE	Sistema Inteligente de Gestão e Despacho de Energia Elétrica
SOAP	*Simple Object Access Protocol*
SoC	*State of Charge*
SoH	*State of Health*
SoP	*State of Power*
SQL	*Structured Query Language*
UFPA	Universidade Federal do Pará
UHE	Usina Hidrelétrica
VM	*Virtual Machine*
WTF	*Web Template Forms*
YAML	*YAML Ain't Markup Language*

Sumário

1 INTRODUÇÃO

Este capítulo apresenta uma visão geral desta dissertação, contextualizando o cenário em que a pesquisa foi desenvolvida. Será discutido as considerações iniciais, as bases tecnológicas da pesquisa, o contexto, a pergunta, justificativa, motivação desse estudo, além dos objetivos gerais e específicos estabelecidos e, por fim, a estrutura do trabalho.

1.1 Considerações Iniciais

A eletricidade do futuro já está sendo moldada por tecnologias emergentes que promovem resultados precisos e mais eficientes do que os atuais. Com o avanço da Indústria 4.0, a digitalização e a automação estão transformando a manufatura e a forma como a energia elétrica é gerada, distribuída e consumida (GARCIA; PEREIRA; PEREIRA, 2024). De acordo com Heymann et al. (2023), a evolução de redes elétricas inteligentes (*smart grids*), Inteligência Artificial (IA), Internet das Coisas (*IoT*, sigla em inglês), Gêmeos Digitais (*Digital Twins*, em Inglês) e sistemas de armazenamento avançados tem impulsionado um novo paradigma energético, no qual a eficiência, a resiliência e a sustentabilidade se tornam elementos centrais. A Figura 1 ilustra uma visão geral a respeito das tecnologias que rodeiam a Industria 4.0.

Figura 1 – Indústria 4.0

Fonte: Adaptado de (CALSOFT, 2023)

Nesse sentido, a utilização de plataformas digitais aliada a redes inteligentes facilita

a gestão de grandes volumes de dados de consumo e geração de energia. A integração de dados entre diferentes partes do sistema elétrico é necessário para a digitalização e transição para um modelo de energia mais sustentável (HEYMANN et al., 2023). Essas inovações estão diretamente relacionadas à transição energética, na qual a adoção de fontes renováveis representa uma estratégia para ampliar a segurança energética, promover a sustentabilidade e reduzir os impactos ambientais provocados pelo uso de combustíveis fósseis (IEA, 2022).

Nesse contexto de digitalização, as redes elétricas inteligentes se destacam por sua capacidade de integrar fontes renováveis, otimizar o consumo e prever falhas antes que ocorram (MAHMOOD et al., 2024). Entretanto, por estarem frequentemente conectadas à rede elétrica convencional, tais redes enfrentam limitações na gestão da energia gerada, o que evidencia a necessidade de sistemas de armazenamento que garantam maior autonomia, estabilidade e flexibilidade no fornecimento (SCHMIDT; STAFFELL, 2023).

Ademais, no escopo das redes elétricas inteligentes, a inclusão de tecnologias de armazenamento como baterias e supercapacitores visa otimizar o uso da energia gerada, especialmente no contexto de fontes renováveis intermitentes (LI et al., 2023). Dessas tecnologias, as baterias são eficazes em armazenar grandes quantidades de energia por períodos prolongados, enquanto os supercapacitores são mais adequados para fornecer potência instantânea em situações de picos de demanda (LI et al., 2023).

A integração dessas tecnologias com as redes inteligentes permite monitorar o sistema, para oferecer uma tomada de decisão dinâmica na distribuição da energia, além de possibilitar ajustes na rede, com base nas condições de geração e demanda. Ou seja, um sistema híbrido com a combinação de baterias e supercapacitores melhora a eficiência do sistema, mas também contribui para a redução das emissões de carbono, ao reduzir a dependência de fontes de energia fósseis para suprir os períodos de alta demanda (SHIRINDA; KUSAKANA, 2022).

Nesse cenário, a implementação de sistemas ciberfísicos, como os *Digital Twins* (DTs) (GRIEVES, 2015), permitem monitorar digitalmente aquilo que é físico, e nesse contexto, com a infraestrutura energética, torna possível monitorar de forma contínua os objetos que a compõe. Ademais, ao considerar sistemas de armazenamento, como baterias, os DTs tornam-se ferramentas ainda mais necessárias, pois são capazes de antecipar falhas, otimizar o desempenho de componentes críticos e adaptar o funcionamento do sistema em tempo real, de acordo com as condições operacionais (RENIERS; HOWEY, 2023).

No entanto, essa revolução energética traz consigo desafios complexos, como a necessidade de investimentos em infraestrutura digital, a criação de regulamentações adequadas e a definição de estratégias robustas para as redes elétricas conectadas. Para tal, esta dissertação surge junto ao desenvolvimento de um Projeto de Pesquisa e Desenvolvimento (P&D) intitulado Sistema Inteligente de Gestão e Despacho de Energia Elétrica (SIGDEE), onde este trabalho abrange a etapa de desenvolvimento de um Sistema de Supervisão e Aquisição de Dados (SCADA) voltado para uma rede elétrica com armazenamento de energia em região isolada.

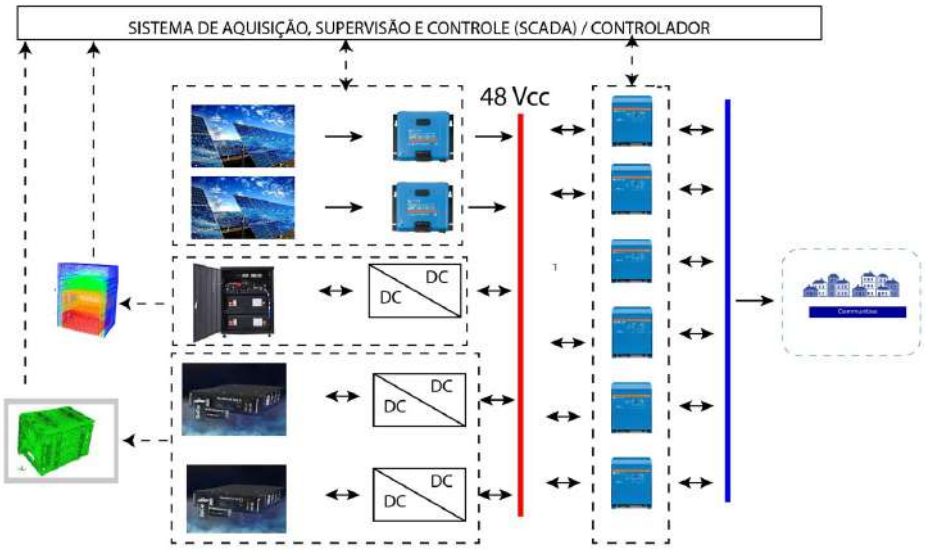
1.2 Contexto da Pesquisa

Esta pesquisa está sendo desenvolvida no escopo do Projeto de Pesquisa e Desenvolvimento (P&D) Sistema de Geração de Energia para Atendimento de Demandas de Pequeno Porte Utilizando Banco de Supercapacitores Integrados com Geração Fotovoltaica (PD-07427-0522/2022). Iniciado em dezembro de 2022, sendo originário da Chamada Externa nº 001/2022 da empresa Norte Energia, sendo classificado no Segmento de Geração de Energia, na Cadeia de Inovação Pesquisa Aplicada com a temática de Fontes Alternativas de Geração de Energia Elétrica (NESA, 2023), sendo financiado pela Norte Energia S.A..

Todo o sistema concebido neste projeto está sendo implementado fisicamente na Usina Hidrelétrica (UHE) de Pimental, localizada em Vitória do Xingu, no sudoeste do Pará, com o objetivo de fornecer atendimento energético distribuído a pequenos núcleos populacionais isolados. Essa região, caracterizada por sua dependência de soluções energéticas alternativas, foi escolhida como projeto piloto por refletir as condições típicas da Região Amazônica, permitindo avaliar o impacto da implementação de uma infraestrutura desse porte em contextos semelhantes.

Entre os objetivos do projeto, se dá o desenvolvimento de um software para a integração da aquisição de dados e supervisão dos diversos processos do sistema de geração, bem como, implementar um sistema de monitoração do armazenamento de energia utilizando a tecnologia de *Digital Twin* (DT), de forma a fornecer diagnóstico a respeito do estado de saúde de um banco de baterias e também fornecer informações ao sistema SCADA. Logo, tem-se a necessidade de desenvolver *softwares* para o sistema proposto, compreendendo uma estrutura complexa e com diversos tipos de sistemas conectados, a Figura 2 exibe a composição completa do projeto, o qual possui sistemas de geração, controle e armazenamento de energia, bem como, o SCADA.

Figura 2 – Microrede isolada do PD-07427-0522/2022



Fonte: O Autor.

1.3 Pergunta de Pesquisa

Como desenvolver um *software* baseado em sistemas SCADA e DT capaz de monitorar e diagnosticar o estado de saúde, estado de carga e comportamento térmico em sistemas de armazenamento de energia em microrredes isoladas, observando seu desempenho em regiões remotas?

1.4 Justificativa

Apesar dos avanços proporcionados pela evolução da *IoT* e dos DTs, ainda persistem barreiras técnicas que dificultam sua implementação em larga escala, especialmente em sistemas energéticos isolados (MCHIRGUI *et al.* 2024, p. 3-7). Nessa perspectiva, a integração de um sistema SCADA apresenta-se como uma solução estratégica e intermediária para viabilizar a coleta, supervisão e controle de ativos energéticos. Além de permitir o monitorar sensores de foma contínua, unidades de controle e dispositivos de geração distribuída, a infraestrutura SCADA permite implementação de funcionalidades digitais mais avançadas — como a modelagem dinâmica de sistemas, a detecção precoce de falhas e a integração com arquiteturas de DTs.

1.5 Motivação

Na implementação de uma microrrede híbrida, composta por geradores e armazenadores, observar o comportamento dos componentes em tempo real é essencial para garantir não apenas a estabilidade do sistema, mas também sua eficiência energética (ULLAH *et al.* 2022, p. 2-3) (MOURLAN *et al.* 2025, p. 5). Microrredes localizadas em regiões remotas, como comunidades isoladas, enfrentam desafios únicos como a variabilidade de carga, acesso limitado à manutenção e a necessidade de operação autônoma (KHATUN *et al.* 2023, p. 3). Nesse contexto, a capacidade de diagnosticar preventivamente falhas e prever o nível de degradação de baterias e supercapacitores pode representar um diferencial importante.

A utilização de DTs integrados a sistemas SCADA fornece uma réplica virtual precisa do sistema físico em operação (BASSEY *et al.* 2024, p. 3). Essa integração permite uma abordagem voltada à manutenção, com menores custos operacionais e maior confiabilidade no sistema (KANDEMIR *et al.* 2024, p. 2-3). Além disso, a arquitetura proposta pode contribuir com políticas públicas voltadas à universalização do acesso à energia, alinhando-se com os Objetivos de Desenvolvimento Sustentável (ODS), em especial o ODS 7 – “Energia Acessível e Limpa” (ONU, 2015).

1.6 Objetivos

1.6.1 Objetivo Geral

O objetivo desta Dissertação de Mestrado é construir um sistema SCADA para aquisição e supervisão de dados obtidos a partir de uma microrrede híbrida, composta por geradores fotovoltaicos e armazenadores de energia, utilizando supercapacitores e baterias. Sendo assim, o *software* deve ser capaz de obter dados, mostrar e alimentar DTs, servindo como ferramenta para o auxílio na tomada de decisão, com base nas ilustrações e grandezas elétricas contidas no escopo da aplicação.

1.6.2 Objetivos Específicos

- Desenvolver o Sistema de Gestão e Despacho de Energia Elétrica (SIGDEE) para ser capaz de gerenciar módulos de geração de sistemas fotovoltaicos.
- Desenvolver módulos para Gestão e Monitoramento de Armazenamento de Energia Híbrido, integrando bancos de baterias, supercapacitores e DTs.
- Desenvolver uma aplicação para apoiar a tomada de decisão com base em dados provenientes de diferentes fontes.

1.7 Estrutura do Trabalho

Esta dissertação está organizada em seis capítulos.

- **Capítulo 1 – Introdução**: Refere-se o capítulo introdutório, com a abordagem de uma visão geral, a definição do problema, a justificativa e motivação da pesquisa, além dos objetivos gerais e específicos.
- **Capítulo 2 – Trabalhos Relacionados**: Compreende os trabalhos relacionados aos principais temas desta dissertação, incluindo tecnologias de armazenamento como baterias e supercapacitores, além de sistemas SCADA no setor energético.
- **Capítulo 3 – Tecnologias Utilizadas**: Neste Capítulo, são apresentadas todas as tecnologias adotadas para o desenvolvimento da plataforma SIGDEE, como Docker, Dojot, MariaDB, Flask e bibliotecas Python, além de explicar o uso de tecnologias web e de virtualização que garantem a escalabilidade, portabilidade e eficiência da aplicação.
- **Capítulo 4 – Metodologia**: É descrita de forma detalhada a metodologia aplicada no desenvolvimento da solução, desde o planejamento e especificações técnicas até a implementação da arquitetura do sistema, incluindo a comunicação entre módulos, banco de dados, API e visualização dos dados online.

- **Capítulo 5 – Resultados e Análises**: Compete aos resultados e funcionalidades desenvolvidas na plataforma SIGDEE, como os módulos de visualização e simulação dos DTs (bateria e supercapacitor), sistema fotovoltaico, alarmes e painel de gerenciamento.
- **Capítulo 6 – Conclusão**: Por fim, é destacado as contribuições do trabalho, observando sua aplicabilidade prática em contextos de baixa infraestrutura energética. Além de possibilidades de expansão e melhoria da plataforma, com foco em novas aplicações de DT e manutenção preditiva.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo tem como objetivo apresentar os fundamentos e as abordagens essenciais que sustentam o desenvolvimento desta dissertação. A revisão bibliográfica busca contextualizar o o cenário energético atual, bem como, os sistemas de geração fotovoltaica, os sistemas de armazenamento de energia e a utilização de *Digital Twin*, com ênfase na integração dessas tecnologias em um sistema SCADA.

Inicialmente, são discutidos os sistemas de geração fotovoltaica, em seguida o armazenamento de energia, focando nas suas funções, aplicações e relevância nos sistemas elétricos. A seguir, o *Digital Twin* e suas aplicações, destacando como essa tecnologia é relevante no monitoramento e simulação, por fim com destaque dos sistemas SCADA, onde é abordada para supervisão, controle e monitoramento, além do papel das tecnologias digitais na construção de soluções mais eficientes e integradas para a gestão energética.

Portanto, este capítulo, proporciona o embasamento teórico necessário para entender as escolhas metodológicas e tecnológicas adotadas nesta pesquisa, além de evidenciar as contribuições e limitações identificadas na literatura, que formam a base para as inovações propostas neste estudo.

2.1 Cenário Energético e Elétrico

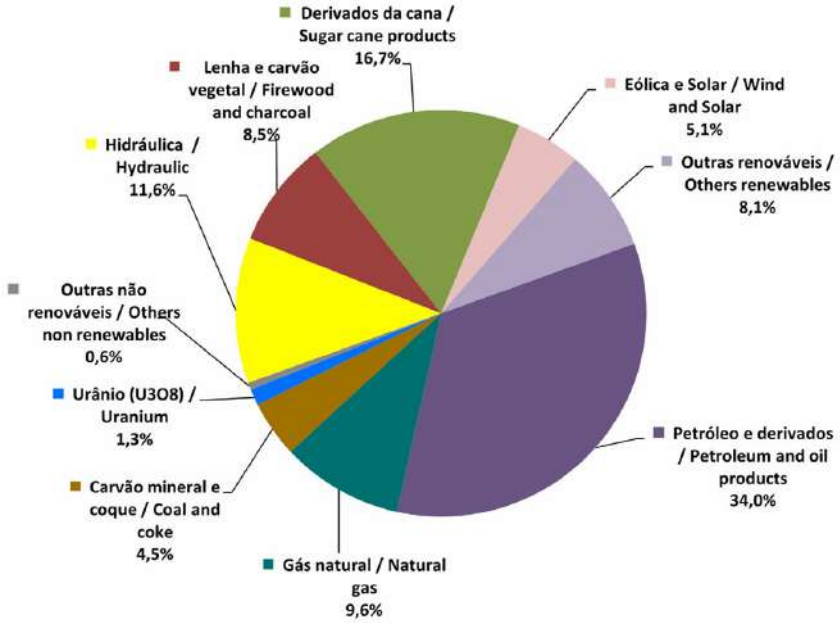
Inicialmente, para compreender melhor a importância e os impactos de soluções no contexto elétrico, é necessário também contextualizar o cenário energético brasileiro, com ênfase na sua matriz de geração e na participação das fontes renováveis nesse processo. Nacionalmente, as grandezas energéticas produzidas ao longo dos anos são contabilizadas de forma abrangente pelo Ministério de Minas e Energia (MME) por meio da Empresa de Pesquisa Energética (EPE).

Conforme a EPE (2025), a matriz energética refere-se ao agrupamento de fontes de energia empregadas para diversas atividades essenciais, tais como abastecer veículos, produção de eletricidade e qualquer outra tarefa que precise de energia para sua realização. Para obter informações com alto nível de confiança sobre a energia brasileira, é divulgado anualmente o Balanço Energético Nacional (BEN). Os dados apresentados no BEN são baseados no ano anterior, por isso, o BEN 2025 considera os dados do ano de 2024.

Segundo o BEN (2025), a matriz energética brasileira é formada predominantemente por fontes não renováveis, sendo o Petróleo e seus derivados com a maior fatia na utilização, representando cerca de 34% da Oferta Interna de Energia (OIE), o que corrobora com a ideia de Han *et al.* (2025, p. 3), onde apontam que a continuidade do uso de combustíveis fósseis representa um dos principais entraves à sustentabilidade ambiental, sobretudo pelas emissões

crescentes de CO_2 (Dióxido de Carbono) nos países do G-20. A Figura 3 mostra como é configurada a oferta de energia no Brasil. Como pode ser observado, o petróleo, gás natural e carvão mineral figuram como os mais utilizados, somando juntos quase 50% da oferta.

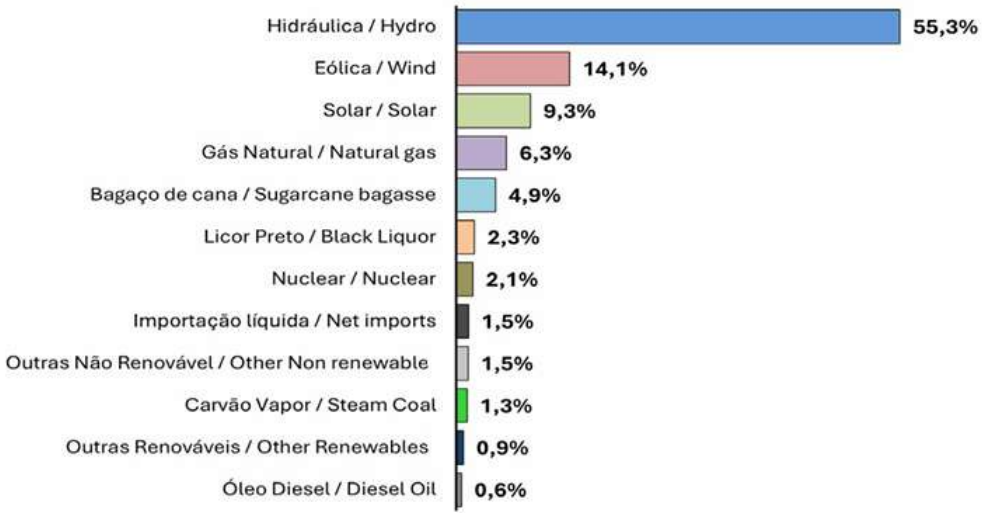
Figura 3 – Oferta Interna de Energia (OIE)



Fonte: BEN (2025), p. 20

Contudo, ao especificar somente a matriz elétrica brasileira do (BEN, 2025), há dominância a partir de fontes renováveis, onde o somatório destas fontes corresponde a 88,2% na Oferta Interna de Energia Elétrica (OIEE).A Figura 4 ilustra a distribuição da oferta elétrica brasileira em 2024, destacando a Fonte Hidráulica com 55,3%, seguida pelas fontes Eólica e Solar, com 14,1% e 9,3%, respectivamente.

Figura 4 – Oferta Interna de Energia Elétrica (OIEE)



Fonte: BEN (2025), p. 12

Essa predominância das fontes renováveis reforça a necessidade de explorar e otimizar sistemas como a geração fotovoltaica, que se apresenta como uma das soluções mais promissoras para aumentar a eficiência e a sustentabilidade da matriz elétrica. A seguir, serão discutidos os sistemas de geração fotovoltaica, que têm se mostrado essenciais para garantir a diversificação da geração de energia elétrica e a redução da dependência de fontes fósseis.

2.2 Sistemas de Geração Fotovoltaica

Sistemas fotovoltaicos são soluções tecnológicas projetadas para converter a energia solar em energia elétrica, de forma limpa, renovável e sustentável. Conforme discutido no trabalho de Khalid *et al.* (2024, p. 2), esses sistemas se destacam como uma alternativa para regiões com acesso limitado à rede elétrica convencional, especialmente em áreas rurais. No contexto atual, a capacidade instalada global de energia solar fotovoltaica atingiu marcos expressivos nos últimos anos, refletindo não apenas avanços tecnológicos, mas também uma maior conscientização sobre a sustentabilidade e a redução das emissões de carbono (ALI *et al.*, 2025) .

De acordo com Pinho e Galdino (2014), os sistemas fotovoltaicos podem ser classificados em duas categorias de maior destaque: Sistemas conectados à rede e Sistemas Isolados. Os sistemas fotovoltaicos conectados à rede elétrica são aqueles em que a energia gerada pelo painel solar é transferida diretamente para a rede. Para garantir a integração segura e eficiente, é essencial o uso de um inversor que atenda às exigências de qualidade e segurança, evitando assim impactos negativos na estabilidade e desempenho do sistema elétrico ao qual o gerador fotovoltaico está conectado (PINHO; GALDINO, 2014).

No Brasil, essa modalidade de sistemas fotovoltaicos conectados à rede se tornou possível após mudanças na regulamentação brasileira, inicialmente disposta pela ANEEL (Agência Nacional de Energia Elétrica) por meio da Resolução Nº 482 de Abril de 2012 (ANEEL, 2012). Atualmente essa resolução já encontra-se defasada, mas é um marco para o setor elétrico.

De acordo com a Lei Nº 14.300 de 6 de Janeiro 2022 (BRASIL, 2022), é estabelecido que os sistemas fotovoltaicos conectados à rede elétrica têm a capacidade de injetar o excedente de energia gerada através de um medidor bidirecional. Esse medidor registra a energia enviada para a rede da concessionária, permitindo que seja compensada ou reutilizada em momentos em que não há geração de energia, ou até mesmo quando a demanda excede a produção.

No estudo realizado por Cremasco *et al.* (2021, p. 2-3), o objetivo foi analisar o comportamento de quatro sistemas fotovoltaicos conectados à rede elétrica instalados em Goioerê, Paraná, utilizando índices de mérito para avaliar sua eficiência. O desempenho desses sistemas foi avaliado em termos de produtividade, taxa de desempenho e fator de capacidade, destacando-se que o Sistema 4 obteve os melhores resultados de geração de energia entre todos os sistemas analisados, com valores mais altos de produção mensal, conforme a Tabela 1.

Tabela 1 – Dados de Geração de Energia Fotovoltaica

Mês/Ano	**S#1 (kWh)**	**S#2 (kWh)**	**S#3 (kWh)**	**S#4 (kWh)**
Jun/2019	394.09	-	-	-
Jul/2019	424.41	664.68	548.38	872.83
Ago/2019	462.40	716.69	611.05	972.58
Set/2019	435.02	684.76	728.79	934.06
Out/2019	513.20	748.65	832.50	1035.75
Nov/2019	496.48	753.27	931.24	1048.68
Dez/2019	453.51	763.01	969.50	1075.30
Jan/2020	539.37	735.11	923.09	1040.87
Fev/2020	404.85	716.17	866.09	1002.89
Mar/2020	495.02	852.73	946.87	1170.09
Abr/2020	405.67	829.31	-	1115.25
Mai/2020	376.09	706.38	-	936.49

Já o Sistema 2 se destacou pela taxa de desempenho, alcançando os maiores índices de desempenho em relação aos outros sistemas, conforme a Tabela 2.

Tabela 2 – Taxa de Desempenho dos Sistemas Fotovoltaicos

Mês/Ano	**S#1 (%)**	**S#2 (%)**	**S#3 (%)**	**S#4 (%)**
Jun/2019	113.20	107.53	73.46	99.77
Jul/2019	110.81	91.23	64.07	86.83
Ago/2019	94.92	89.18	76.90	83.00
Set/2019	90.99	82.34	73.38	76.47
Out/2019	90.46	79.89	73.38	76.47
Nov/2019	81.39	77.20	75.80	71.11
Dez/2019	68.49	72.16	72.44	67.09
Jan/2020	84.71	72.24	71.89	68.97
Fev/2020	71.63	76.21	73.57	71.25
Mar/2020	86.24	92.55	82.93	72.05
Abr/2020	83.73	106.27	76.37	75.60
Mai/2020	93.77	95.75	-	91.00

Segundo Pinho e Galdino (2014), sistemas isolados podem ser configurados de forma individual, quando a geração é voltada para apenas uma unidade consumidora ou no formato de minirredes, quando a geração é partilhada entre um pequeno grupo de unidades consumidoras que estão próximas geograficamente. De acordo com Gorjian e Shukla (2020) estes sistemas incluem tipicamente armazenamento de energia para armazenar energia excedente gerada pelo sistema FV, que é utilizada durante a não geração ou à noite.

No trabalho de Leite (2021), foi abordada a implementação de um sistema supervisório para monitoramento e controle das unidades geradoras de energia elétrica em sistemas isolados instalados no estado do Amazonas. O estudo focou na criação de uma solução para captura de grandezas elétricas, como tensão, corrente e potência, de forma automatizada, utilizando um

sistema SCADA. A pesquisa também explorou as particularidades locais, como a dificuldade de acesso e a logística nas áreas isoladas, evidenciando a importância de uma solução que permitisse acesso remoto aos dados e facilitasse as tomadas de decisão, garantindo mais eficiência e segurança no processo de geração de energia. A Figura 5 representa a geolocalização dos locais atendidos no município de Caapiranga.

Figura 5 – Sistema isolado no municipio de Caapiranga

Fonte: Leite (2021, p. 15)

Em contextos como esse e a necessidade de monitorar esses sistemas, a tecnologia de DT configura uma solução inovadora para aprimorar o monitoramento e a gestão. Ao replicar digitalmente os ativos físicos e permitir simulações e análises, os DTs oferecem um nível elevado de precisão no acompanhamento do desempenho e da saúde de dispositivos e instalações, proporcionando maior controle e eficiência, especialmente em locais com infraestrutura limitada.

2.3 Sistemas de Armazenamento de Energia

Os sistemas de armazenamento de energia elétrica acumulam energia em momentos de baixa demanda ou alta produção. Essa energia armazenada pode então ser liberada quando a demanda supera a oferta, contribuindo para a estabilidade da rede e reduzindo perdas energéticas. De acordo com a *Energy Information Administration* (EIA), esses sistemas operam a partir do

carregamento de dispositivos de armazenamento — utilizando, por exemplo, energia solar, eólica ou da rede elétrica — e permitem sua posterior liberação em níveis e qualidades adequadas para atender às necessidades de consumo (EIA, 2023).

Mariano (2021) destaca que a integração de sistemas de armazenamento em plantas fotovoltaicas conectadas à rede representa uma estratégia fundamental para suavizar a variabilidade da geração solar e otimizar o perfil de consumo. No estudo de caso, a utilização de um banco de baterias aliado a inversores bidirecionais possibilitou a redução de picos de demanda e maior flexibilidade na operação, isto é, para maior confiabilidade energética. Além disso, foi verificado que a operação conjunta permitiu o aumento do ciclo de vida útil das baterias quando casos de descarga parcial foram aplicadas.

Existem diversas formas de armazenar energia, sendo uma das mais comuns o uso de componentes eletroquímicos, como as baterias e os supercapacitores. Como observado por Campos et al. (2022), no Brasil, a necessidade de sistemas de armazenamento surge devido à intermitência das fontes renováveis, como a energia solar, que apresenta variações na geração de eletricidade ao longo do dia e das estações do ano. Essas flutuações na geração exigem soluções de armazenamento que possam equilibrar a oferta e a demanda, garantindo um fornecimento contínuo e confiável de energia.

As baterias de íons de lítio destacam-se como a principal tecnologia de armazenamento de energia utilizada no Brasil, devido à sua alta eficiência, longevidade e redução de custos ao longo dos anos. Além disso, as baterias de chumbo-ácido, embora ainda comuns, apresentam limitações em termos de densidade de energia e ciclo de vida (CAMPOS et al., 2022). Ademais, em relação aos sistemas de armazenamento com baterias, Moraes (2023) afirma que o uso de *Battery Management Systems* (BMS) nas baterias de íons-lítio é fundamental para garantir segurança e eficiência no ciclo de carga e descarga. O BMS monitora parâmetros como temperatura, tensão e corrente, prevenindo danos às baterias e garantindo a operação eficiente ao longo do tempo.

De acordo com Sahin, Blaabjerg e Sangwongwanich (2022), os supercapacitores têm se destacado como dispositivos promissores para armazenamento de energia elétrica em diversas aplicações modernas, especialmente em sistemas de energia renovável e veículos elétricos. Sua capacidade de fornecer alta densidade de potência, ciclos de carga e descarga extremamente rápidos, e longa vida útil os tornam uma alternativa ou complemento às baterias convencionais.

Diferente das baterias, os supercapacitores armazenam energia através de mecanismos puramente eletrostáticos ou pseudocapacitivos, permitindo maior eficiência na entrega de potência e menor degradação ao longo do tempo (SAHIN; BLAABJERG; SANGWONGWANICH, 2022). Ao combinar esses componentes, podem ser aproveitadas as vantagens de ambas as tecnologias, ou seja, tanto a alta capacidade de armazenamento de energia das baterias como a alta potência de descarga dos supercapacitores (TECHNOLOGIES, 2023). A Figura 6 mostra como podem ser comercializados os supercapacitores.

Figura 6 – (a) Diferentes tipos de dispositivos comerciais SC, (b) Módulos SC de duas empresas diferentes.

Fonte: Sahin, Blaabjerg e Sangwongwanich (2022, p. 10)

Em escala de utilidade, Torres et al. (2025) analisaram a aplicação de um sistema de baterias de 30 MW/60 MWh no estado de São Paulo como ativo de transmissão. O estudo demonstrou que, além de aliviar congestionamentos na rede, o BESS (*Battery Energy Storage System*) pode fornecer múltiplos serviços: aumento da confiabilidade em operação ilhada, suporte à restauração do sistema após um apagão e maior capacidade de integração de fontes renováveis variáveis. Ainda, os autores destacam ainda que a modularidade e a rápida implantação de sistemas de baterias representam vantagens frente à construção de novas linhas de transmissão, sendo alternativas promissoras para expandir a flexibilidade do sistema elétrico nacional.

Ademais, Naves et al. (2025) realizaram uma análise comparativa entre baterias e supercapacitores aplicados ao armazenamento de energia em sistemas fotovoltaicos, em configurações *off-grid* e *on-grid* em uma ilha brasileira. Os resultados mostram que, apesar das limitações dos supercapacitores em termos de densidade energética, eles oferecem vantagens em aplicações que exigem altas potências instantâneas e maior durabilidade em regimes de carga/descarga cíclicos. O estudo também mostra que a combinação híbrida (HESS – *Hybrid Energy Storage Systems*) pode ser mais eficiente, pois une a elevada densidade de energia das baterias à alta densidade de potência dos supercapacitores, o que ajuda tanto no atendimento de cargas críticas como na viabilidade econômica de sistemas isolados.

2.4 Digital Twin

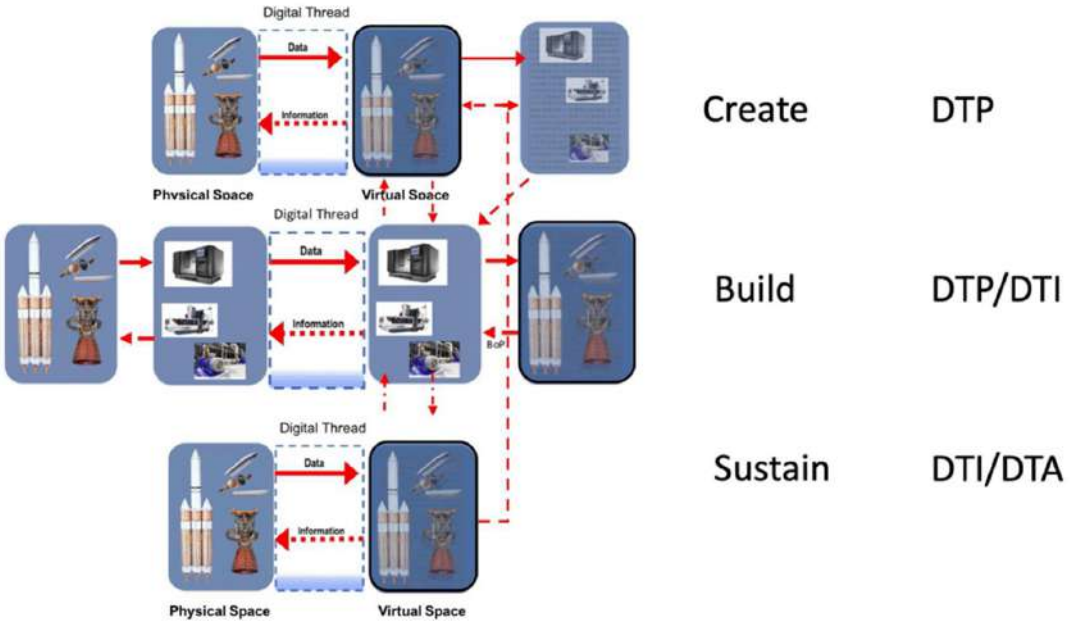
O termo DT teve seu início na década de 1960 em um programa espacial da NASA, onde modelos físicos foram desenvolvidos de forma idêntica a terra para replicar e prever as condições das espaçonaves em missão, por isso o termo (ALLEN, 2021). No ano de 2002, Michael Grieves formalizou o conceito de DT no contexto da manufatura, descrevendo-o como uma representação virtual que espelha os ativos físicos ao longo de seu ciclo de vida, possibilitando a simulação, monitoramento e análise em tempo real (GRIEVES, 2002).

Nessa perspectiva, (GRIEVES, 2022) reforça a respeito do ciclo de vida do DT, onde são definidas algumas fases, as quais são divididas da seguinte forma:

- **Digital Twin Prototype (DTP)** – Refere-se ao modelo digital inicial que representa a concepção ou o design de um produto, sistema ou componente. Esse protótipo é construído antes da existência física do ativo e serve como base para o desenvolvimento, testes virtuais, simulações de comportamento e validação de funcionalidades.
- **Digital Twin Instance (DTI)** – Representa a instância digital de um ativo físico específico, já fabricado e em operação. Diferente do DTP, o DTI está diretamente vinculado a um ativo real e recebe dados em tempo real por meio de sensores e sistemas de monitoramento.
- **Digital Twin Aggregate (DTA)** – Consolidação de múltiplas instâncias de Digital Twins (DTIs), agregando dados e comportamentos coletivos de diversos ativos semelhantes ou interconectados. Essa visão agregada possibilita a análise em nível de sistema ou rede, revelando padrões globais, tendências de desempenho, falhas recorrentes e oportunidades de otimização em escala.

A Figura 7 ilustra como as fases do ciclo de vida do DT ocorrem

Figura 7 – O modelo de DT ao longo do ciclo de vida.



Fonte: Grieves (2022, p. 2)

Na literatura, o desenvolvimento do tema DT tem sido crescente. O trabalho de Mchirgui *et al.* (2024, p. 4-6) explora as aplicações e os desafios da tecnologia de DTs em redes elétricas inteligentes. O estudo mostra que a integração entre DT e sistemas SCADA tem ganhado destaque como uma ferramenta poderosa para melhorar a automação, detecção de falhas e manutenção preditiva em tempo real. Além disso, destaca como os DTs permitem simulações operacionais complexas e fornecem suporte à tomada de decisão com base em dados dinâmicos. No entanto, os autores também apontam desafios importantes, como a interoperabilidade entre sistemas legados, a segurança cibernética e a necessidade de modelos mais padronizados e escaláveis para garantir sua ampla adoção no setor energético.

Na revisão de Angelova *et al.* (2024, p. 2-5), os autores afirmam que um dos avanços mais significativos na Indústria 4.0 é o conceito de DT, com a criação de réplicas digitais de sistemas físicos para simulação, monitoramento e otimização de processos. Eles revisam e comparam diversos estudos que aplicam DTs à modelagem de instalações fotovoltaicas, destacando como essa tecnologia pode ser aplicada em diferentes contextos, como o monitoramento da produção de energia, a gestão da energia renovável e a redução de custos operacionais, principalmente com o uso de IoT e sensores inteligentes. Por fim, mostram que o DT, com sua capacidade de integrar dados em tempo real e otimizar processos, se apresenta como um grande avanço na melhoria das operações do setor elétrico.

De forma complementar, (HELUANY; GKIOULOS, 2024) conduzem uma análise sistemática da literatura sobre a aplicação de DTs nos domínios de geração e distribuição de energia, com base no modelo conceitual do Instituto Nacional de Padrões e Tecnologia dos EUA

para redes inteligentes. O estudo destaca que os DTs oferecem vantagens como simulação em tempo real, diagnóstico de falhas e suporte à tomada de decisão, especialmente quando integrados a sistemas SCADA. A revisão também aponta que, na geração de energia, o uso mais recorrente é o diagnóstico de falhas, enquanto na distribuição, o gerenciamento é a principal aplicação, refletindo o potencial da tecnologia em aumentar a resiliência da infraestrutura energética.

Essas evidências reforçam a relevância dos DTs como uma tecnologia estratégica para a modernização de sistemas elétricos, especialmente em aplicações que demandam maior automação, confiabilidade e eficiência operacional. No entanto, a implementação efetiva dessas soluções em contextos reais exige abordagens aplicadas que considerem as particularidades de cada ambiente, principalmente em regiões com infraestrutura limitada e necessidades energéticas críticas. Nesse sentido, a presente pesquisa busca aplicar tais conceitos no desenvolvimento de uma solução voltada para cenários de geração e armazenamento de energia em comunidades isoladas da região amazônica.

2.5 Sistemas SCADA

Para monitorar e obter uma visão abrangente de uma rede elétrica, considerando os componentes de geração e armazenamento de energia, são comumente utilizados sistemas SCADA. Nesse sentido, a aplicação de tecnologias *web* atuais no desenvolvimento de sistemas SCADA tem ganhado destaque, especialmente no setor elétrico.

Inicialmente, com necessidade de maior acessibilidade e flexibilidade nos sistemas de controle industrial. O estudo de caso de Energy (2021) demonstrou como sistemas SCADA modernos permitiram que organizações industriais mitigassem tempo de inatividade custoso, mantivessem eficiência e comunicassem problemas para tomar decisões mais inteligentes em tempo real ainda mais destacou a importância crescente da acessibilidade remota.

Em paralelo, Foundation (2021) aborda que os avanços em IoT e tecnologias de computação de borda *(edge computing)* contribuiram diretamente para o crescimento na automação industrial, sendo fundamental para o desenvolvimento subsequente de sistemas SCADA baseados em *web*, já que a computação de borda fornece a infraestrutura necessária para processar dados e enviá-los para interfaces *web*.

Em Pearson (2021), é destacada a consolidação e o amadurecimento de tecnologias *Web* em sistemas SCADA para ambientes elétricos, observando que sistemas SCADA deveriam ser acessíveis tanto no local quanto na nuvem, permitindo maior flexibilidade de implementação. Além disso, a importância de adotar tecnologias abertas, como *SQL, Python, MQTT, REST, HTML5 e CSS* para a interoperabilidade e escalabilidade.

Para corroborar com isso, Lee et al. (2023) utilizam tecnologias modernas no desenvolvimento de sistemas SCADA em um sistema de gestão de energia para integrar unidades de

geração fotovoltaica (PV), sistemas de armazenamento de energia e estações de carregamento de veículos elétricos. O sistema proposto usa uma arquitetura híbrida de servidores locais e na nuvem, com dados em tempo real e análise preditiva de geração de energia renovável e consumo de energia em dispositivos de armazenamento ou estações de carregamento.

As tecnologias *Web* atuais permitem a multivariedade de serviços, conforme Rodríguez-Alonso, Pena-Regueiro e García (2024), que apresentam uma plataforma de DT para estações de tratamento de água, baseada em uma arquitetura de microsserviços otimizada para computação de borda. A plataforma utiliza *Flask* para o *backend* e *React* para o *frontend*, demonstrando a viabilidade dessas tecnologias no desenvolvimento de sistemas SCADA.

2.6 Considerações Finais do Capítulo

Este capítulo apresentou os principais fundamentos teóricos e tecnológicos que embasam a pesquisa, com foco nas tecnologias para otimização de sistemas elétricos, incluindo armazenamento de energia, geração fotovoltaica, e a integração de soluções como *Digital Twin* e sistemas SCADA. Discutiu-se a importância das fontes renováveis na matriz elétrica brasileira e como os sistemas fotovoltaicos, juntamente com as soluções de armazenamento, contribuem para a diversificação e sustentabilidade energética.

Foram analisadas as principais tecnologias de armazenamento, como baterias e supercapacitores, e como elas podem ser integradas aos sistemas fotovoltaicos para melhorar a flexibilidade e confiabilidade dos sistemas elétricos. Além disso, abre-se precedente para a relevância do *Digital Twin* para o monitoramento e otimização das operações, especialmente quando combinado com sistemas SCADA.

Por fim, a evolução dos sistemas SCADA foi discutida, com ênfase no uso de tecnologias *web* e IoT, que tornam os sistemas mais acessíveis e escaláveis. A integração dessas tecnologias, como o *Digital Twin* e SCADA, para uma gestão elétrica mais eficiente, formando a base para as soluções propostas nesta pesquisa. O capítulo seguinte apresentará as tecnologias específicas e os métodos utilizados para implementação dessa proposta.

3 TECNOLOGIAS UTILIZADAS

Este capítulo apresenta as tecnologias empregadas no desenvolvimento do sistema proposto nesta dissertação. O foco recai sobre as ferramentas e plataformas utilizadas na implementação do ambiente de supervisão e controle, com ênfase na interoperabilidade entre módulos, portabilidade da aplicação e escalabilidade da solução.

Entre os recursos adotados, destacam-se: O gerenciador de infraestrutura utilizado para orquestrar os recursos computacionais e possibilitar a execução isolada de serviços via containers; a plataforma DOJOT, voltada à integração de dispositivos e gerenciamento de dados no contexto de Internet das Coisas (IoT); o *microframework Flask*, aplicado no desenvolvimento do *back-end* da aplicação; e os componentes *Web* (HTML, CSS e JavaScript), empregados na construção de interfaces interativas, acessíveis e responsivas.

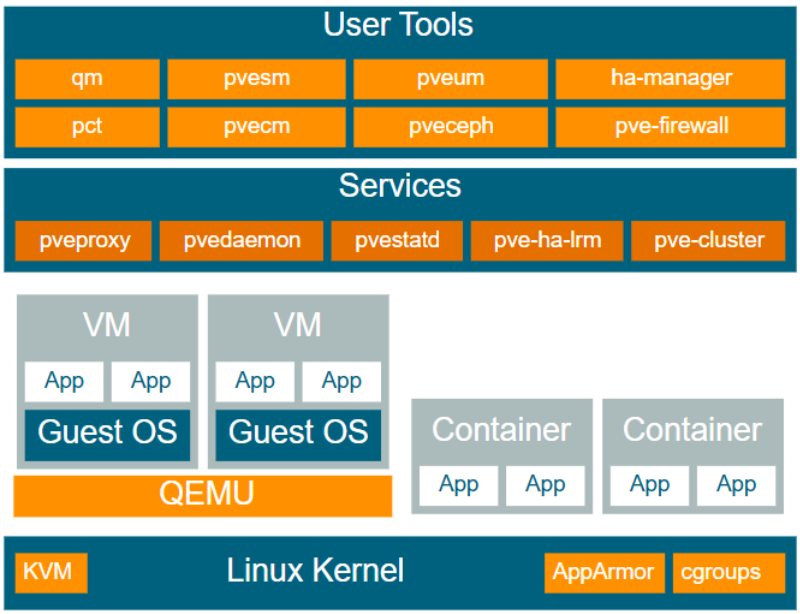
Importa destacar que todas as ferramentas utilizadas são de código aberto, assegurando não apenas liberdade de uso e adaptação, mas também alinhamento com os princípios de reprodutibilidade e acessibilidade tecnológica, especialmente relevantes para aplicações em regiões de infraestrutura limitada. Nos tópicos a seguir, cada tecnologia será descrita em termos de sua função no sistema, justificativa de uso e benefícios oferecidos ao contexto da aplicação desenvolvida.

3.1 Virtualização

Ao analisar os níveis do processo de desenvolvimento de *software*, observa-se que tecnologias capazes de otimizar recursos surgem como oportunidades para melhorar o escopo de um projeto. Nesse contexto, a virtualização, no nível dos Sistemas Operacionais, desempenha um papel fundamental ao proporcionar maior compatibilidade e escalabilidade entre diferentes tipos de aplicativos. Por essa razão, esse paradigma foi adotado como parte da estratégia de desenvolvimento.

Como aplicação para este nível, o *Proxmox Virtual Enviroment* é uma plataforma *open-source* de virtualização, com interface *web* integrada. A distribuição é baseada em *Linux* e atua como um sistema operacional (PROXMOX, 2024). Ele permite criar, gerenciar e executar máquinas virtuais (VMs) e containers de forma eficiente, além de possuir várias ferramentas e serviços nativos oferecem modularidade e escalabilidade no uso da aplicação, conforme ilustrado na Figura 8.

Figura 8 – Proxmox VE



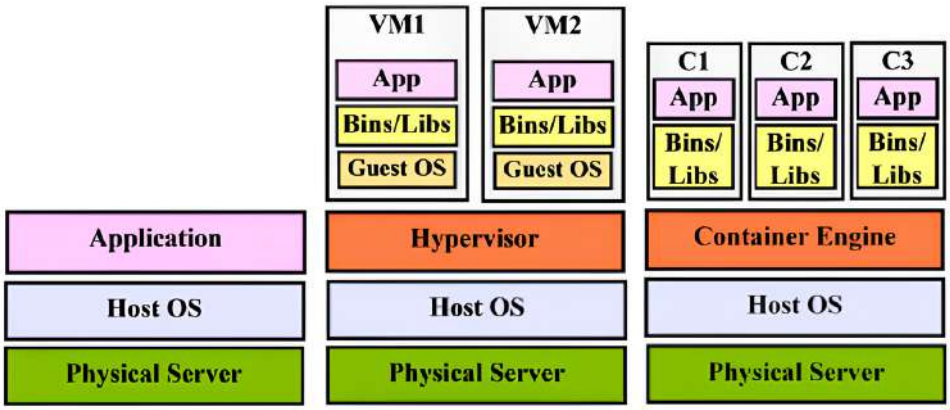
Fonte: (PROXMOX, 2024)

O *Proxmox* é uma distribuição baseada em *Linux* que atua como um sistema operacional especializado para virtualização e gerenciamento de máquinas virtuais. O *Proxmox* utiliza o KVM (*Kernel-based Virtual Machine*) como seu hipervisor para criar e gerenciar máquinas virtuais (VMs) (PROXMOX, 2024). O KVM é um módulo do *kernel* do *Linux* que oferece virtualização completa, permitindo que diferentes sistemas operacionais possam ser executados em máquinas virtuais de maneira isolada, com desempenho praticamente nativo.

Além disso, o *Proxmox* oferece suporte para LXC, que é uma tecnologia de virtualização baseada em *containers*. Ao contrário das máquinas virtuais, os *containers* compartilham o mesmo *kernel* do sistema operacional *host*, oferecendo uma solução mais leve e eficiente, ideal para cargas de trabalho que não exigem a total abstração de *hardware* (PROXMOX, 2024). Esse LXC permite executar instâncias isoladas de Linux de maneira eficiente, consumindo menos recursos do que as VMs tradicionais.

Por isso, Bhardwaj e Krishna (2021), discutem como a virtualização baseada em hipervisores, usada criar máquinas virtuais apresenta sobrecarga de desempenho pela necessidade de cada Márquina Virtual precisar de um sistema operacional. Ademais, estabelecem que, a alternativa para este problema é o uso de *containers*, pois, este compartilha o *kernel* do sistema operacional do *host*, oferecendo menor sobrecarga e maior eficiência para aplicações em nuvem. Na Figura 9, é possível observar um comparativo entre os recursos que são necessários para a utilização de cada um dos métodos.

Figura 9 – Comparação entre métodos: Tradicional, *Hypervisor e Container*



Fonte: Bhardwaj e Krishna (2021), p. 4

A virtualização e a conteinerização revolucionaram a infraestrutura tecnológica, especialmente em ambientes computacionais que demandam alta eficiência e flexibilidade. A virtualização permite emular múltiplas VMs em um único *hardware* físico, utilizando um hipervisor para criar sistemas independentes e isolados. Já a conteinerização (TURNBULL, 2014), introduzida por tecnologias como *Docker* (DOCKER, 2025), foca em ambientes leves, executando aplicativos diretamente sobre o sistema operacional do *host*, sem necessidade de emulação de *hardware* completo.

Essa diferenciação impacta diretamente o desempenho de aplicações *multithread*, como demonstrado por estudos de caso envolvendo matrizes de grande escala. A execução de algoritmos *multithread* em VMs pode sofrer com limitações de recursos, enquanto em containers, o uso eficiente dos recursos compartilhados do sistema hospedeiro melhora a escalabilidade e reduz a sobrecarga. Essa tecnologia trabalha fortemente em fatores como isolamento, desempenho e escalabilidade.

3.1.1 *Containers*

No escopo de virtualização, uma tecnologia que foi atrelada a esse tipo de função é o uso de *containers*. Este, representa melhorias significativas em relação ao uso de recursos computacionais, já que a demanda é consideravelmente menor do que usar máquinas virtuais. Portanto, aliado à virtualização, o uso de *Container Docker* (DOCKER, 2025) desponta de forma emergente em produções de ambientes de software.

3.1.1.1 *Docker*

Dentro desse contexto, foi utilizada a ferramenta *Docker* (DOCKER, 2025) devido à sua e eficiência e escalabilidade, permitindo que ambientes de desenvolvimento e produção sejam replicados com precisão, com cada processo funcionando de forma isolada, o que ajuda contrubui

na redução de falhas de dependência entre sistemas (TURNBULL, 2014). Ademais, o *Docker* oferece a opção de executar aplicações em diferentes ambientes sem alterar a configuração do sistema operacional *host*.

Além disso, o *Docker* permite que equipes de desenvolvimento empacotem uma aplicação com todas as suas dependências em um único *container*, o que garante que ela seja executada da mesma forma em qualquer ambiente, seja no desenvolvimento, testes ou produção (TURNBULL, 2014). Essa função auxilia principalmente em trabalhos com sistemas distribuídos e microserviços, onde a flexibilidade e a consistência são pontos importantes. Dado isso, essa nova forma de virtualização é uma alternativa escalável em relação aos métodos convencionais.

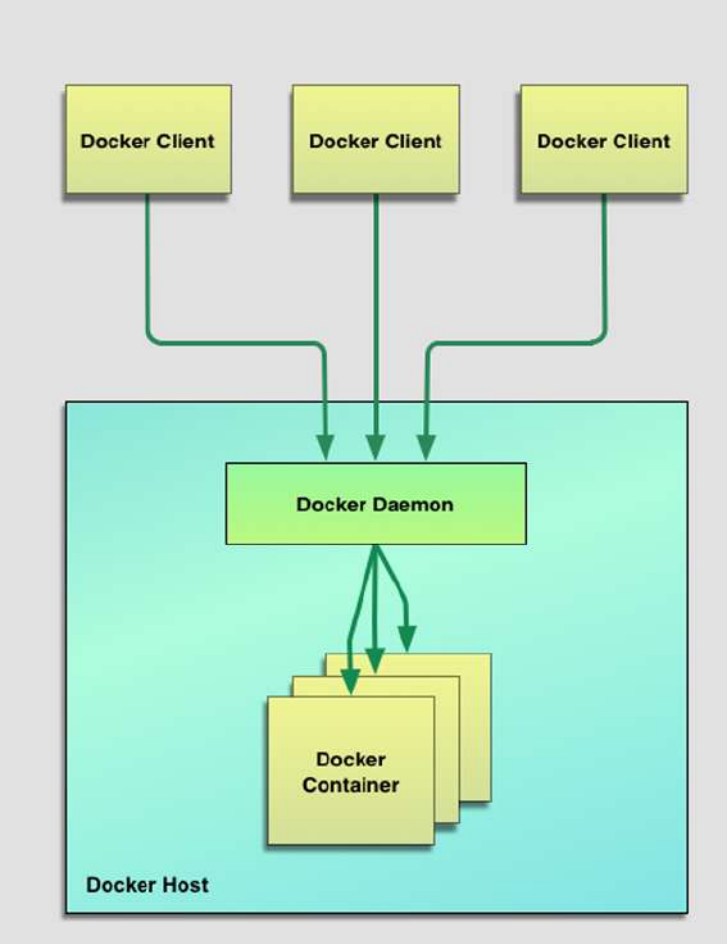
De acordo com Turnbull (2014), a arquitetura do *Docker* é baseada em uma estrutura cliente-servidor que facilita a criação, gerenciamento e execução de *containers*. A Figura 10 representa como a arquitetura do *Docker* é organizada, a qual envolve três componentes principais: *Docker Client, Docker Daemon e Docker Host* (TURNBULL, 2014).

- ***Docker Client*:** Interface de comunicação com o usuário por meio da linha de comando. Neste ponto, são enviados os comandos para o *Docker Daemon*, que executa os pedidos. O *Docker Client* pode ser executado em várias máquinas e se conecta ao *Docker Daemon* que está executando no *Docker Host*.

- ***Docker Daemon:*** Gerenciamento para a criação, execução e monitoramento dos *containers Docker*. Ele escuta por comandos do *Docker Client* e interage com o *Docker Host* para executar as tarefas solicitadas. O *Docker Daemon* é a parte fundamental que controla o ciclo de vida de todos os *containers* no sistema.

- ***Docker Host:*** Esse é um ambiente físico ou virtual onde o *Docker Daemon* e os *containers* estão localizados. Nesse *host* estão contidos todos os recursos necessários para executar os *containers* e garantir que as instâncias criadas sejam independentes, seguras e isoladas.

Figura 10 – Arquitetura *Docker*



Fonte: (TURNBULL, 2014, p. 10)

3.1.1.2 Imagem

Em síntese, os *containers* são unidades leves e isoladas de execução que encapsulam uma aplicação e todas as suas dependências, incluindo bibliotecas, arquivos de configuração e outros elementos necessários para sua execução. Eles compartilham o sistema operacional do *Docker Host*, mas operam de maneira independente, isso ocorre por meio de uma imagem base, que representa cada aplicação.

Na prática, imagem no *Docker* se refere a um arquivo imutável (que não pode ser alterado) que contém tudo o que é necessário para rodar uma aplicação, incluindo o código-fonte, bibliotecas, variáveis de ambiente e arquivos de configuração (ROSA et al., 2023). No *Docker*, as imagens *Docker* são a base para a criação de *containers*, sendo que cada *container* é uma instância em execução de uma imagem.

As imagens *Docker* são criadas a partir de um arquivo chamado *Dockerfile*, que define as etapas necessárias para configurar a aplicação e suas dependências, conforme a Figura 11. A partir do *Dockerfile*, o *Docker* constrói uma imagem por meio de uma série de comandos, como copiar arquivos para dentro da imagem, instalar pacotes de *software* e definir configurações de

rede.

Figura 11 – Imagem *Dockerfile*

```
FROM node:12-alpine

RUN apk add --no-cache python2 g++ make

WORKDIR /app
COPY . .

RUN yarn install --production

CMD ["node", "src/index.js"]
EXPOSE 3000 here
```

Fonte: Adaptado de (ROSA *et al.* 2023)

Cada comando no *Dockerfile* cria uma nova camada na imagem. Ou seja, quando o arquivo de configuração de uma imagem é modificada, apenas as camadas alteradas precisam ser reprocessadas, economizando tempo e recursos no processo de construção da imagem. As camadas também permitem que o *Docker* reutilize partes de imagens já existentes, o que otimiza o armazenamento e o deixa mais eficiente.

3.1.1.3 *Docker Compose*

Ao utilizar essas configurações ao escopo de desenvolvimento de um *software*, é possível criar apenas um *container* com uma aplicação específica, ou unir vários *containers* utilizando outra ferramenta acoplada ao *Docker*, essa é definida como *Docker Compose* (DOCKER, 2025). O *Docker Compose* orquestra e gerencia vários *containers*, o que resulta em uma aplicação multi-container que cria e subdivide a aplicação em serviços, cada um com uma responsabilidade.

Para isso, é criado um arquivo de configuração em YAML (normalmente chamado de docker-compose.yml), é possível definir os serviços que compõem a aplicação, incluindo as dependências entre eles, configurações de rede, volumes compartilhados e outras configurações necessárias para a execução de uma aplicação completa (DOCKER, 2025). Para o caso de uma aplicação *web*, os *containers* podem ser separados para o servidor da aplicação, o banco de dados e o serviço de cache, o que é definido e configurado dentro de um único arquivo *Docker Compose*, conforme a Figura 12.

Figura 12 – Arquivo docker-compose.yaml

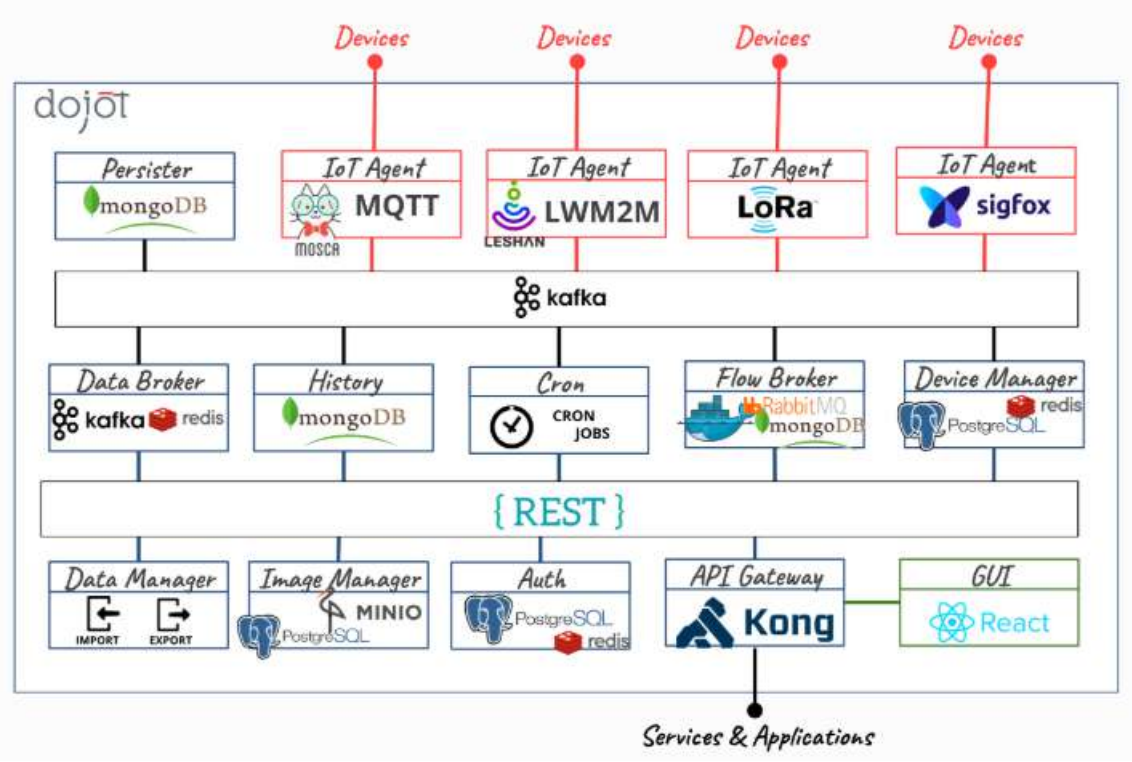
```yaml
services:
  app:
    image: node:18-alpine
    command: sh -c "yarn install && yarn run dev"
    ports:
      - 127.0.0.1:3000:3000
    working_dir: /app
    volumes:
      - ./:/app
    environment:
      MYSQL_HOST: mysql
      MYSQL_USER: root
      MYSQL_PASSWORD: secret
      MYSQL_DB: todos

  mysql:
    image: mysql:8.0
    volumes:
      - todo-mysql-data:/var/lib/mysql
    environment:
      MYSQL_ROOT_PASSWORD: secret
      MYSQL_DATABASE: todos

volumes:
  todo-mysql-data:
```

Fonte: Adaptado de (DOCKER *et al.* 2025)

3.1.2 DOJOT

O software DOJOT (DOJOT, 2020) é uma plataforma de código aberto, desenvolvida pelo CPqD (Centro De Pesquisa e Desenvolvimento em Telecomunicações), com o objetivo de fornecer a prototipagem IoT de forma escalável e robusta. Na documentação (DOJOT, 2020) é disponibilizada de forma completa com todo o conteúdo voltado para o uso da ferramenta, além disso, sua construção contempla diversas tecnologias. A Figura 13 mostra como é essa arquitetura.

Figura 13 – Interface Dojot



Fonte: (DOJOT, 2020)

Na documentação (DOJOT, 2020), há uma série de componentes que são utilizados no escopo da aplicação, entre eles, temos:

- Kafta + DataBroker: O Apache Kafka é uma plataforma distribuída para gerenciamento de mensagens, projetada para transmissão e processamento eficiente de dados. Na Dojot, ele é utilizado por sua capacidade de isolar dados de diferentes locatários e aplicativos por meio de uma estrutura de tópicos dedicada, garantindo um ambiente multilocatário. Já o DataBroker trabalha junto ao Kafka, utilizando um banco de dados em memória para tornar o processamento mais ágil, fornecendo contexto aos dados e permitindo que serviços consumam informações em tempo real. Sua arquitetura distribuída evita falhas únicas e reduz possíveis gargalos.

- Gerenciador de Dispositivos: O DeviceManager é o serviço central responsável por gerenciar dispositivos e modelos de dados, além de comunicar atualizações para outros componentes via Kafka. As solicitações são *stateless*, com os dados armazenados em um banco de dados, onde cada solicitação é tratada de forma independente, sem dependência de dados ou interações anteriores.

- Agente IoT: O Agente IoT atua como um intermediário entre dispositivos físicos e a Dojot, desempenhando o papel de um driver para diferentes conjuntos de dispositivos.

Ele é projetado para lidar com protocolos variados, como MQTT, CoAP, Lora, Sigfox e HTTP, permitindo flexibilidade na integração. Além disso, é responsável por estabelecer comunicações seguras com os dispositivos conectados.

- Serviço de autorização do usuário: O serviço de autorização é responsável por validar chamadas de API e gerenciar perfis de usuário e acessos dentro da plataforma. Ele opera sem estado, armazena seus dados em um banco de dados e usa cache para otimizar o processamento de um grande volume de solicitações. Além disso, pode ser escalado horizontalmente para atender à demanda.

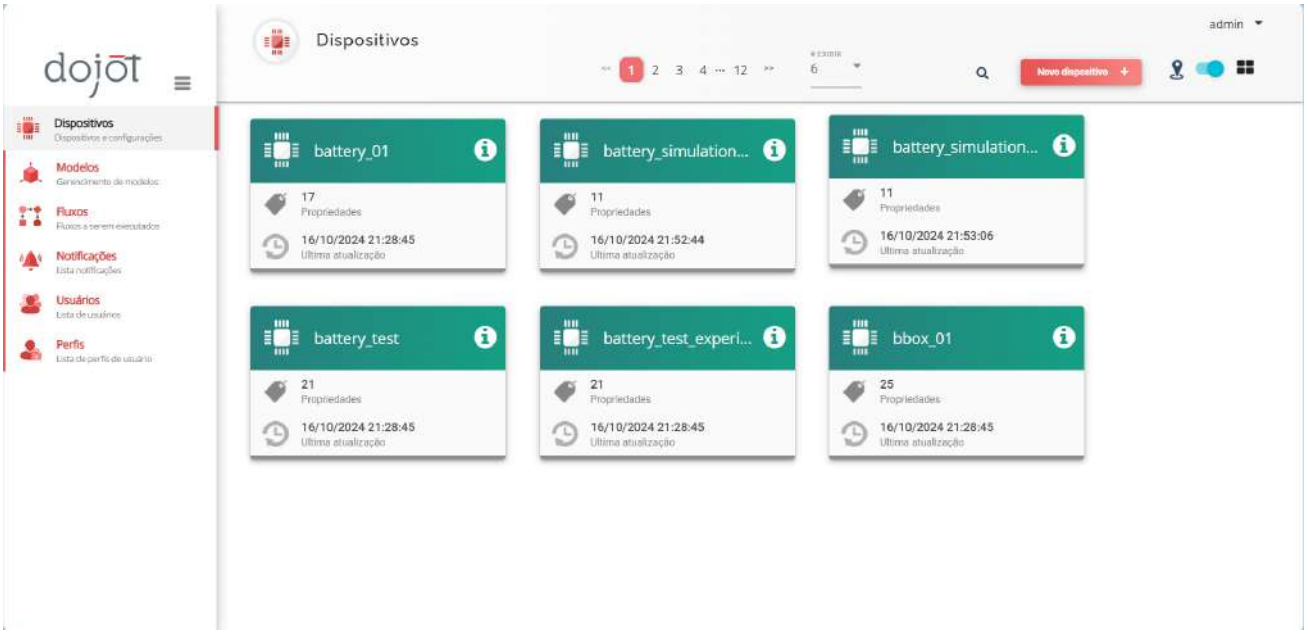
- Persister/Histórico: O componente Persister atua como um pipeline para dados e eventos, convertendo-os em uma estrutura adequada e enviando-os para o banco de dados. Utiliza o MongoDB, um banco de dados não relacional, que oferece suporte a clusters fragmentados conforme necessário. Os dados persistidos podem ser acessados por meio de uma API REST disponibilizada pelo microsserviço Histórico.

- Armazenador e recuperador do InfluxDB: Os serviços InfluxDB Storer e InfluxDB Retriever trabalham em conjunto para gerenciar dados no InfluxDB. O InfluxDB Storer consome dados do Kafka dos dispositivos da Dojot e os grava no InfluxDB. Já o InfluxDB Retriever recupera esses dados gravados, acessando-os através de uma API REST.

- Gateway de API Kong: O Kong API Gateway serve como o ponto de entrada para os aplicativos e serviços da Dojot, oferecendo benefícios como um único ponto de acesso e facilitando a implementação de políticas, como controle de acesso e limitação de taxa de tráfego nas chamadas de API.

Segundo Medeiros (2023), a arquitetura presente na DOJOT utiliza métodos que transmitem dados em tempo tempo real, podendo subscrever, armazenar e processar fluxos, assim como, disponibilizar o desenvolvimento de novos agentes IoT por meio da API (*Application Programming Interface*). Para a inserção e configuração de novos dispositivos, uma interface GUI (*Graphical User Interface*, da traduçao Interface Gráfica do Usuário) é disponibilizada para o usuário, onde possibilita o gerenciamento de usuários, modelos, dispositivos, fluxos e notificações. A Figura 14 mostra como se dá essa interface.

Figura 14 – Interface Dojot



Fonte: (DOJOT, 2020)

No trabalho de Terada *et al.* 2022, foi avaliada a integração e desempenho da DOJOT em aplicações de carregamento inteligente de veículos elétricos (EVs). Os *softwares* utilizam a DOJOT para acessar dados de telemetria e gerenciar dispositivos, como estações de carregamento e recursos energéticos distribuídos (DERs). Um dos *softwares* gera políticas mensais de variação de preços e limites de potência para operadores de estações de carregamento, enquanto o outro controla dispositivos em intervalos curtos (por exemplo, 5 minutos). Os resultados mostraram que a DOJOT suporta processamento paralelo e pode gerenciar milhares de dispositivos, dependendo da qualidade da conexão com a internet e do uso de processos simultâneos. Assim, a DOJOT demonstrou ser robusta para atender demandas complexas de processamento em aplicações de eletromobilidade.

3.1.3 *FLASK*

O *Flask* (PROJECTS, 2024) é um *microframework* minimalista e altamente modular para desenvolvimento de aplicações *Web* em *Python*. Com uma estrutura flexível, permite desenvolver projetos simples e, gradualmente, incluir funcionalidades mais avançadas conforme as necessidades crescem. Entre suas características fundamentais estão o suporte nativo a roteamento, gerenciamento de requisições *HTTP*, uso de templates dinâmicos e a capacidade de integrar extensões para funcionalidades adicionais. Essa ferramenta foi escolhida também pela sua multifuncionalidade. A seguir, estão algumas das razões para o de desenvolvimento com *Flask*.

3.1.3.1 Arquitetura Básica e Roteamento

Abrange facilidade na criação de rotas que ligam URLs a funções específicas dentro da aplicação. Através do decorador @app.route, é possível definir *endpoints* que respondem a diferentes URLs. Além disso, o *framework* suporta regras dinâmicas para URLs, permitindo o uso de variáveis, como <username> ou <int:id>, que podem ser processadas pelas funções correspondentes. Essa flexibilidade serve para criar interfaces interativas e responsivas, alinhadas às boas práticas de usabilidade.

3.1.3.2 Gestão de Templates

A integração nativa com o mecanismo de templates *Jinja2* permite a geração dinâmica de páginas *HTML*. Os desenvolvedores podem criar arquivos template reutilizáveis que utilizam recursos como herança, laços e condições, promovendo consistência e eficiência na construção de interfaces. O uso de templates é útil em aplicações que exigem mudanças frequentes no *design* ou que necessitam de personalização com base nos dados do usuário.

3.1.3.3 Arquivos Estáticos

Durante o desenvolvimento, o *Flask* gerencia arquivos estáticos, como *CSS, JavaScript* e imagens, através de uma pasta dedicada chamada static. Esses arquivos podem ser referenciados diretamente por meio da função *url-for*('static', filename='nome-do-arquivo'), simplificando a manutenção de *links* e oferecendo compatibilidade com diferentes ambientes.

3.1.3.4 Manipulação de Requisições e Sessões

O objeto *request* no *Flask* permite acesso direto aos dados enviados pelos clientes, como parâmetros de URL, dados de formulário e uploads de arquivos. Essa flexibilidade é importante para o desenvolvimento de *APIs* e aplicações interativas. Além disso, o *framework* implementa o gerenciamento de sessões de forma segura, utilizando *cookies* assinados criptograficamente. Essa opção visa garantir informações específicas de usuários sejam mantidas entre diferentes requisições sem comprometer a segurança dos dados.

3.1.3.5 *APIs e JSON*

O *Flask* simplifica a criação de *APIs*, permitindo que dicionários e listas retornados pelas funções sejam automaticamente convertidos em respostas *JSON*. Para tipos de dados mais complexos, como modelos de banco de dados, bibliotecas de serialização podem ser integradas, garantindo a conversão para formatos *JSON* válidos. Esse suporte nativo é bastante utilizado em aplicações modernas que precisam integrar diferentes serviços e dispositivos.

3.1.3.6 Tratamento de erros e redirecionamentos

O *Flask* oferece ferramentas para lidar com erros e redirecionamentos de forma elegante. Através do decorador *@app.errorhandler*, é possível criar páginas personalizadas para códigos de erro *HTTP*, como 404 ou 500. Além disso, a função *redirect()* permite o redirecionamento de usuários para outros *endpoints*, enquanto a função *abort()* permite encerrar requisições com um código de erro específico.

3.1.3.7 Extensões e modularidade

O *Flask* suporta uma ampla variedade de extensões, como *Flask-SQLAlchemy* para integração com bancos de dados e *Flask-WTF* para formulários, entre outras. Essas extensões expandem as funcionalidades do *framework* sem comprometer sua leveza da aplicação, logo, é bastante completa no desenvolvimento de aplicações.

3.1.3.8 Segurança e Debug

Durante o desenvolvimento, o *Flask* oferece o modo de depuração, que recarrega automaticamente a aplicação ao detectar alterações no código e exibe erros de forma interativa no navegador. Contudo, essas ferramentas são desativadas em ambientes de produção para evitar riscos de segurança. O *framework* também implementa medidas de proteção contra ataques de injeção de código.

Na academia, o *Flask* tem se destacado como uma ferramenta amplamente utilizada em projetos de pesquisa e desenvolvimento devido à sua flexibilidade, simplicidade e capacidade de integração com diversas tecnologias (KHRIJI *et al.*, 2020), (BONNEY *et al.*, 2022) e (MORCHID *et al.*, 2024). Pesquisas em áreas como IoT, DTs e plataformas inteligentes têm explorado o potencial do *Flask* para criar aplicações *web* robustas e escaláveis.

De acordo com o estudo de Khriji *et al.* (2020, o projeto Smart-Lab foi desenvolvido para otimizar o gerenciamento de laboratórios de pesquisa por meio de tecnologias IoT. A plataforma integra sensores, sistemas de armazenamento de dados e interfaces *web* para monitorar e controlar experimentos de forma centralizada. Através do uso do *framework Flask* no *back-end* e de uma base de dados *MySQL*, o sistema permite acesso remoto e compartilhamento de resultados entre pesquisadores e parceiros industriais. A arquitetura busca resolver desafios como a continuidade do trabalho, a automação de tarefas repetitivas e a gestão de recursos, promovendo maior eficiência e colaboração em ambientes acadêmicos e industriais.

Conforme mencionado por Bonney *et al.* (2022), a criação de uma plataforma operacional de gêmeos digitais (DTOP-Cristallo) usando *Python* e *Flask* oferece novas possibilidades para a gestão de sistemas de engenharia, como turbinas eólicas e plantas industriais. A plataforma conecta gêmeos físicos a interfaces *Web* e serviços em nuvem, permitindo o monitoramento remoto, simulações avançadas e maior acessibilidade para usuários distribuídos geograficamente.

O estudo ressalta a importância de soluções *open-source*, como o DTOP, para promover a interoperabilidade e a colaboração entre diferentes instituições. Apesar de ser um protótipo, a pesquisa aponta que esta abordagem pode transformar a forma como dados e ativos são gerenciados em sistemas complexos.

Na perspectiva de Morchid *et al.* (2024), o *framework Flask* foi essencial no desenvolvimento de um sistema de detecção de incêndios em tempo real, o que serve para agricultura sustentável, considerando os riscos ambientais e econômicos associados a incêndios. O estudo apresenta uma solução baseada em IoT e sistemas embarcados, com sensores distribuídos nos campos agrícolas para monitorar condições ambientais como fumaça e chamas. Os dados são processados por um *Raspberry Pi 3 B+* e exibidos em uma interface *web* desenvolvida com *Flask, HTML e CSS*. Ademais, é destacado que o sistema demonstrou eficiência na detecção precoce de incêndios, contribuindo para a proteção de colheitas e infraestrutura agrícola, utilizando de tecnologias de sensoriamento IoT (MORCHID *et al.*, 2024).

3.1.4 Tecnologias *WEB*

Considerando o contexto de sistemas SCADA e monitoramento IoT, há atualmente tecnologias que fornecem suporte para a implementação dessas soluções, composta pela visualização de gráficos, *dashboards* e alertas em tempo real. Essas tecnologias são essenciais no que tange ao auxílio no acompanhamento e controle dos sistemas, com dados dispostos em tempo real para analisar o estado e desempenho dos dispositivos monitorados.

3.1.4.1 HTML

O HTML (*HyperText Markup Language*) é considerado uma linguagem de marcação utilizada como a base estrutural da interface da aplicação (DOCS, 2024c). Ele organiza os elementos visuais da página, permitindo a disposição de conteúdos como textos, imagens, formulários e botões (DOCS, 2024c). No contexto da aplicação, o HTML serve para o usuário visualizar as informações geradas pelo *back-end* e marcar elementos constantes nas páginas *Web*. A partir de marcações são incluídos diferentes elementos, como textos, imagens e *links*.

Na perspectiva do desenvolvimento, a forma como o HTML é apresentado se dá por meio de "Tags"(etiquetas) para definir um elemento que compõe a página *web*, na maioria dos casos esses elementos possuem tag de abertura (para iniciar o elemento) e de fechamento (término do elemento). Basicamente, quando um navegador carrega uma página HTML, ele interpreta as tags e constrói uma "árvore", onde cada "folha"representa um elemento.

3.1.4.2 CSS

O CSS *(Cascading Style Sheets)* complementa o HTML, ao definir a aparência e o estilo da interface. Ele permite a personalização de elementos como cores, fontes, tamanhos

e espaçamentos, o que fornece controle sobre o design das páginas, uma experiência visual agradável e consistente (DOCS, 2024a). No processo de desenvolvimento de uma aplicação *Web*, o CSS é utilizado para melhorar a estética da aplicação e torná-la mais intuitiva para os usuários (KHATER; AL-NASHIF; ABIDO, 2022).

O CSS funciona aplicando regras de estilo aos elementos HTML. Uma regra CSS é composta por um seletor e um bloco de declarações. O seletor aponta para o elemento HTML que você deseja estilizar, podendo ser um elemento ou uma classe inteira (DOCS, 2024a).

Por fim, o CSS pode ser incluído em um documento HTML de três maneiras:

- **CSS Externo:** Em um arquivo .css separado, vinculado ao HTML usando a tag <link> no <head>. Esta é a abordagem mais comum, pois permite a reutilização de estilos em múltiplas páginas e facilita a manutenção (DOCS, 2024a).
- **CSS Interno:** Dentro da tag <style> no <head> do documento HTML. Útil para estilos específicos de uma única página (DOCS, 2024a).
- **CSS Inline:** Diretamente em uma tag HTML usando o atributo *style* (por exemplo, <p style="color: red;»)(DOCS, 2024a).

3.1.4.3 JAVASCRIPT

Incluindo funcionalidades que conversam com *Python*, *JavaScript* é a linguagem de programação responsável por adicionar interatividade e dinamicidade à interface das aplicações (DOCS, 2024d) no contexto das páginas *Web*. Pois, ele possibilita a atualização de elementos da página em tempo real, sem a necessidade de recarregar o toda ela, além de realizar requisições assíncronas ao *back-end* (via *Fetch API* ou *AJAX*) (DOCS, 2024b). No processo de desenvolvimento de uma aplicação, o *JavaScript* auxilia na criação de gráficos interativos, atualizar os dados automaticamente de forma modular e melhorar a experiência do usuário.

Basicamente, o *JavaScript* manipula o conteúdo constante nas páginas HTML, essa função pode modificar atributos de *tags* e mudar estilos CSS em resposta a interações com usuário.

3.1.5 *Softwares* Utilizados

Em um contexto geral voltado para softwares utilizados, os recursos lógicos incluíram a utilização de algoritmos de desenvolvimento, virtualização e técnicas avançadas de computação paralela. Esses elementos, quando combinados com o *hardware* buscam atender de forma eficiente aos requisitos estipulados. Alguns destes recursos podem ser vistos conforme a Tabela 4, onde estão descritos os *softwares* com suas versões e uma breve descrição de cada um.

Tabela 3 – *Softwares* e bibliotecas utilizadas na arquitetura

Software	**Versão**	**Descrição**
Proxmox VE	7.4-3	Plataforma de virtualização de código aberto para gerenciamento de máquinas virtuais e *containers*.
Docker Engine	24.0.5	Plataforma para desenvolvimento, envio e execução de aplicações em *containers*.
Docker Compose	2.20.2	Ferramenta para definição e execução de aplicações *Docker multi-containers*.
Dojot	0.7.0	*Middleware IoT* brasileiro para gerenciamento de dispositivos e processamento de dados.
MariaDB	10.11.4	Sistema de gerenciamento de banco de dados relacional, derivado do *MySQL*.
FreeCAD	0.20.2	*Software* de modelagem 3D paramétrica de código aberto.
Elmer	9.0	*Software* de simulação de elementos finitos para problemas multifísicos.
PyBaMM	22.12	Biblioteca *Python* para modelagem de baterias de íons de lítio.
OpenFOAM	10	*Software* de dinâmica de fluidos computacional (CFD) de código aberto.
Flask	2.0.3	*Microframework* web para desenvolvimento de aplicações *Python*.
Werkzeug	2.0.3	Biblioteca *WSGI* para manipulação de requisições e respostas *HTTP* no *Flask*.
Jinja2	3.0.2	Motor de *templates* utilizado pelo *Flask* para renderização dinâmica de páginas.
Flask-Login	0.5.0	Extensão do *Flask* para gerenciamento de autenticação de usuários.
Flask-Migrate	3.1.0	Ferramenta para controle de migrações no banco de dados *SQLAlchemy*.
WTForms	3.0.0	Biblioteca para criação e manipulação de formulários em aplicações *web*.
Flask-WTF	1.0.0	Integração do *Flask* com a biblioteca *WTForms* para manipulação avançada de formulários.
Flask-SQLAlchemy	2.5.1	Extensão do *Flask* para integração com *SQLAlchemy* e bancos de dados relacionais.
SQLAlchemy	1.4.29	Mapeamento objeto-relacional para interação com bancos de dados *SQL*.

Email Validator	1.1.3	Biblioteca para validação de endereços de *e-mail* em formulários.

Fonte: O Autor.

3.2 Considerações Finais do Capítulo

Este capítulo apresentou as principais tecnologias utilizadas no desenvolvimento da solução proposta, com foco em ferramentas de código aberto para garantir acessibilidade e escalabilidade. A adoção de containers e integração *web* buscou flexibilidade e eficiência no desenvolvimento e implantação do sistema.

A virtualização foi iniciada com o *Proxmox VE*, permitindo a gestão de máquinas virtuais e containers. Em seguida, o *Docker* foi empregado como ferramenta central de conteinerização, junto ao *Docker Compose* para orquestração de serviços. A plataforma DOJOT foi utilizada como solução IoT, oferecendo gerenciamento de dispositivos, fluxos de dados em tempo real e arquitetura baseada em microsserviços, adequada a sistemas SCADA distribuídos.

Para o *back-end* da aplicação *web*, escolheu-se o *Flask*, por sua simplicidade, flexibilidade e capacidade de expansão, com suporte a rotas, templates, *APIs REST* e segurança. Na camada *front-end*, está sendo utilizado HTML, CSS e JavaScript, visando uma interface interativa, responsiva e integrada ao *Flask*. Essas tecnologias compõem a base do sistema SCADA com funcionalidades de *Digital Twin*. O próximo capítulo detalha sua integração prática no desenvolvimento do *software*.

4 INTEGRAÇÃO DO *SOFTWARE*

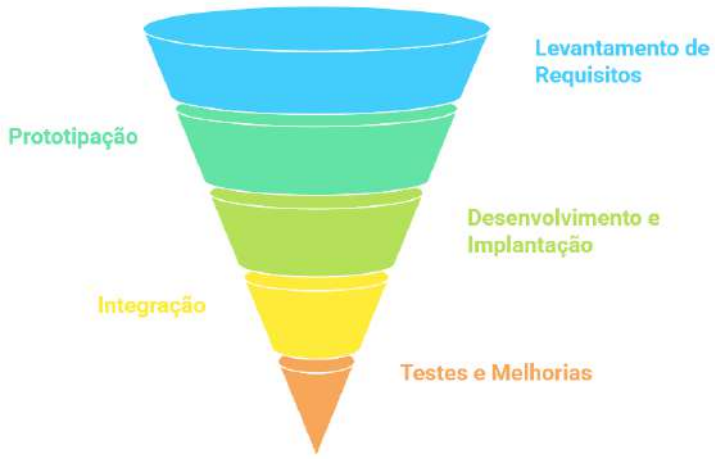
Este capítulo apresenta as metodologias adotadas para o desenvolvimento da solução proposta, fundamentadas nas tecnologias discutidas no capítulo anterior. Serão abordados os principais aspectos da arquitetura do sistema, incluindo a organização dos módulos de *software*, a configuração da infraestrutura de *hardware* e as estratégias de integração entre os componentes. O objetivo é detalhar o processo de construção da aplicação, desde o planejamento até a implementação, com o interesse de garantirar a escalabilidade, flexibilidade e compatibilidade.

A metodologia contempla desde a configuração do ambiente de desenvolvimento até a estruturação lógica da aplicação, baseada em uma arquitetura modular e distribuída. Também serão discutidos os critérios para a escolha das ferramentas utilizadas, os fluxos de dados entre os elementos do sistema e a forma como as tecnologias de *back-end*, *front-end* e IoT foram integradas, para oferecer uma visão clara do processo de desenvolvimento da aplicação.

4.1 Metodologia de Desenvolvimento da Pesquisa

Inicialmente, para o desenvolvimento de forma estruturada, é necessário que sejam observadas as etapas que compreendem desde os aspectos físicos até os lógicos do *software*. Para isso, considera-se a metodologia adotada, o ciclo de vida do *software* e as boas práticas aplicadas. A abordagem foi organizada em cinco etapas principais, executadas de maneira sequencial e cronológica. É importante destacar que, ao longo do processo, ajustes foram realizados conforme as necessidades das partes interessadas, o que é comum em projetos dessa natureza. A Figura 15 ilustra a sequência das atividades metodológicas desenvolvidas.

Figura 15 – Etapas de Desenvolvimento



Fonte: O Autor

Esse modelo em funil destaca a importância de cada fase e reforça que a estratégia de um projeto depende de uma execução estruturada e iterativa, que permita evoluções constantes a partir de validações práticas. Essas etapas são importantes, pois representam um fluxo organizado e progressivo para o desenvolvimento de sistemas. O processo inicia com o levantamento de requisitos, etapa essencial para entender as necessidades do projeto e definir as funcionalidades desejadas. Em seguida, parte-se para a prototipação, onde são criados modelos iniciais que permitem visualizar e validar conceitos antes da implementação completa.

Na sequência, ocorre o desenvolvimento e implantação, momento em que as soluções são efetivamente construídas e colocadas em operação. Após isso, a integração garante que os diferentes componentes do sistema funcionem de forma harmoniosa e eficiente. Por fim, são realizados testes e melhorias, uma etapa crítica para assegurar a qualidade, corrigir falhas e otimizar o desempenho da solução.

Seguindo esse planejamento, serão definidas a seguir as metodologias que compõem todo o desenvolvimento do sistema, desde a estruturação da camada física computacional até a implementação e validação das funcionalidades no ambiente digital. Essa abordagem considera:

- **Integração entre *hardware* e *software*:** Abordagem dos aspectos cruciais, como a seleção dos dispositivos físicos (sensores, controladores e outros componentes).
- **Definição das interfaces de comunicação:** Estabelecimento das camadas iniciais de comunicação entre os dispositivos físicos e o software.
- **Desenvolvimento da lógica de processamento:** Implementação da lógica no *back-end* e criação de modelos de DT para simulação e análise.
- **Criação da interface *web*:** Desenvolvimento da interface voltada para a visualização e interação com os dados em tempo real.
- **Otimização e escalabilidade:** Aplicação de técnicas para garantir que o sistema seja eficiente e capaz de suportar aumentos na demanda de dados e processamento.

Além disso, serão detalhados os processos de implantação e testes, que visam garantir a confiabilidade do sistema, bem como sua escalabilidade e desempenho. Essa trajetória metodológica assegura uma visão completa e estruturada do ciclo de desenvolvimento, alinhando os objetivos do projeto à proposta de uma arquitetura robusta, modular e compatível com a transformação digital da rede elétrica.

4.2 Especificações Técnicas

Esta seção apresenta as especificações técnicas detalhadas de *hardware* que são fundamentais para o desempenho e estabilidade sistema. Entender e descrever esses requisitos é

importante para manter a compatibilidade, escalabilidade e a manutenção adequada da aplicação.

4.2.1 Especificações Técnicas de *Hardware*

O sistema está sendo desenvolvido na Universidade Federal do Pará (UFPA), mais especificamente, no Centro de Excelência em Eficiência Energética da Amazônia (CEAMAZON) em parceria com o Projeto (PD-07427-0522/2022) financiado pela Norte Energia SA. Inicialmente, foi observada a necessidade de recursos computacionais, para isso, será instalado um servidor capaz de atender as aplicações desenvolvidas, sendo descrito na Tabela 4.

Tabela 4 – Especificações técnicas de *Hardware.*

Modelo	*Precision 3680 Tower*
Processador	14ª geração *Intel® Core™ i9-14900* (24-*cores*, 32 *threads*, cache de 36 MB, 2.0 GHz a 5.8 GHz Turbo, 65 W)
Memória	64 GB DDR5 (2x32GB) 4400MT/s
Placa de vídeo	NVIDIA® RTX™ 4000 *Ada Generation*, 20GB GDDR6
Armazenamento	HDD de 2TB (7200RPM) SATA 3,5"
Armazenamento SSD	1TB PCIe NVMe M.2 (Inicialização)
Sistema Operacional	*Windows 11 Pro*, PT-BR
Resfriamento Térmico	Cooler de Ar de CPU com Direcionador de Ar
Chassi	*Precision 3680 Tower de 1000W (80 Plus Platinum) PSU*
Placa de rede	Adaptador de placa de rede Intel 2.5GbE i226 FH PCIe
Wireless	Placa de rede sem fio *Intel® Wi-Fi 6E AX211, 2x2, 802.11ax*

Fonte: O Autor.

Conforme os recursos descritos, a Figura 16 mostra o servidor em sua forma física, onde contempla as tecnologias necessárias para comportar todo o escopo de *softwares* em desenvolvimento.

Figura 16 – Servidor *Precision 3680 Tower*

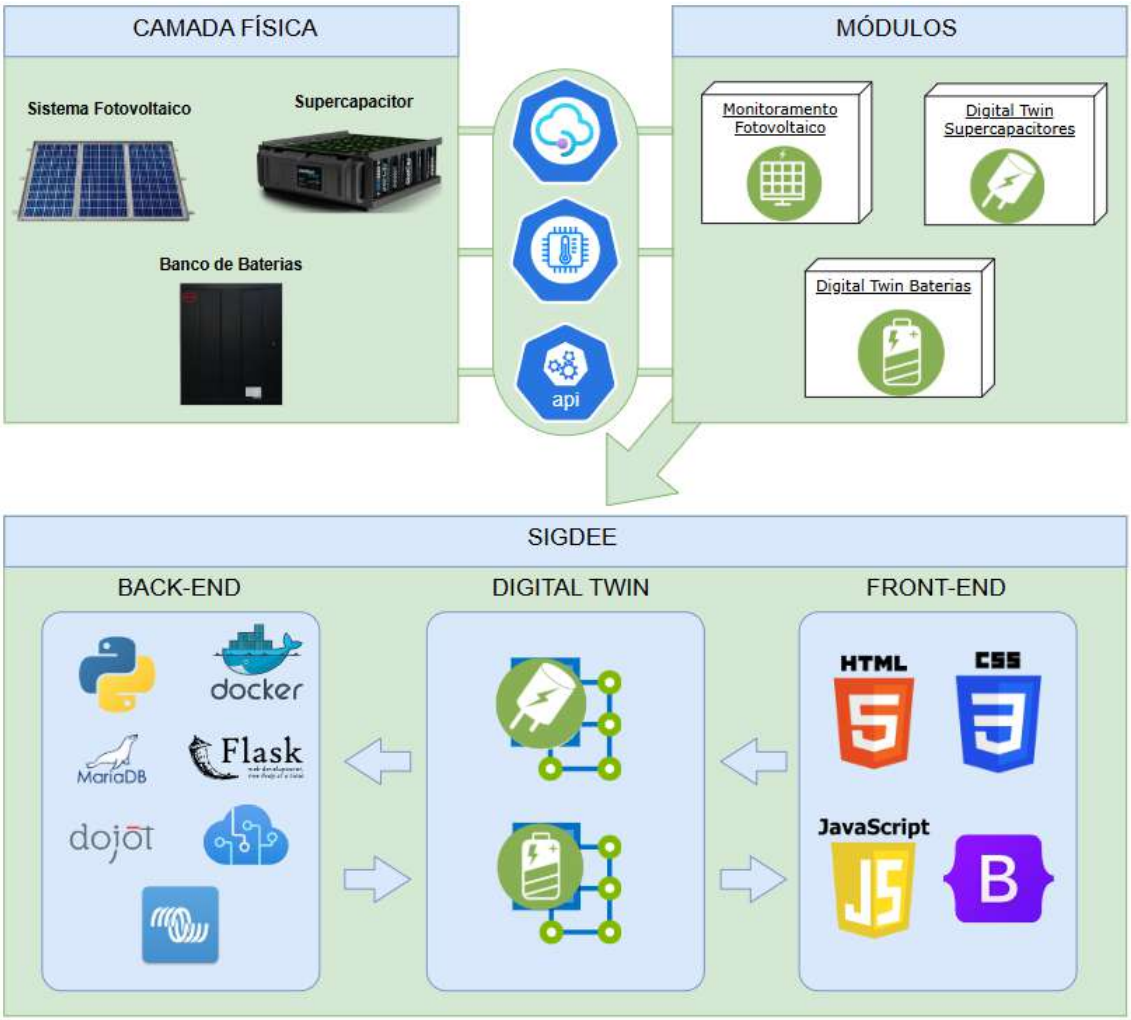
Fonte: O Autor

4.3 Estrutura do Sistema

O desenvolvimento do sistema está sendo realizado utilizando o *microframework Flask* no *back-end* junto ao lado do servidor, onde são compreendidas as funções e as integrações entre os módulos. O *front-end* foi construído com uma interface visual, utilizando as tecnologias *HTML, CSS e JavaScript*, com o objetico de fornecer uma experiência interativa e responsiva ao usuário. A estrutura das rotas foi organizada para vincular cada *endpoint* a uma função controladora específica, com separação e organização de responsabilidades.

Além disso, foram definidos modelos que facilitam a integração com o banco de dados, utilizando bibliotecas *Python* para gerenciar as entidades da aplicação de forma eficiente. A Figura 17 ilustra a arquitetura geral do sistema, destacando a integração entre as diferentes camadas, desde a camada física até a interface de usuário final. A imagem é dividida em três seções principais: Camada Física, Módulos, e o sistema SIGDEE, que integra todos os serviços.

Figura 17 – Arquitetura do Sistema



Fonte: O Autor

De forma geral, ao apresentar o sistema de forma sintetizada, a composição compreende:

Camada Física: A camada física do sistema é composta por dispositivos essenciais para a operação do sistema de energia implementado, como o Sistema Fotovoltaico, os Supercapacitores e o Banco de Baterias. Esses componentes são responsáveis pelo fornecimento e armazenamento de energia. A camada física é a base para a coleta de dados e operação do sistema, sendo fundamental para a medição e monitoramento do desempenho energético.

Módulos de Monitoramento para o DT: Acima da camada física, temos os Módulos que incluem o Monitoramento Fotovoltaico, DT Supercapacitor, e DT Baterias. Esses módulos têm a função de simular o comportamento e o desempenho dos sistemas em tempo real, utilizando o conceito de DT. Cada um desses módulos recebe dados da camada física e os processa para gerar simulações preditivas, principalmente sobre o funcionamento e o status dos componentes do sistema. A interação com os dados oferecidos e o DT permite uma análise aprofundada e auxiliar a tomada de decisão baseada em dados históricos e em tempo real.

***API* de Comunicação:** A *API* atua na comunicação entre as camadas do sistema. Ela conecta a camada física, os módulos de monitoramento e o DT ao *back-end*, onde ocorre o processamento e a gestão dos dados. A *API* atua como intermediária, permitindo que os dados sejam enviados para o *back-end* de forma mais rápida, por sua vez, estes dados serão processados, armazenados e utilizados para gerar informações úteis para a operação do sistema.

SIGDEE (Sistema Inteligente de Gestão e Despacho de Energia Elétrica): O sistema em desenvolvimento é dividido em três seções principais:

- ***Back-End*:** Aqui, o sistema utiliza tecnologias como *Flask, Docker, MariaDB*, e Dojot, para gerenciar a lógica do sistema, armazenar dados no banco de dados e fazer a integração com os módulos e dispositivos. O *back-end* se comunica com a *API*, realizando o processamento necessário antes de enviar ou receber dados do DT.
- ***Digital Twin:*** Esta seção está diretamente conectada ao *back-end*, realizando a simulação e modelagem dos componentes físicos. Utilizando os dados obtidos da camada física, o DT modela o comportamento dos sistemas, simulando cenários para otimizar o desempenho e prever falhas ou necessidades de manutenção.
- ***Front-End*:** Por fim, no lado do cliente, o *front-end* é a interface do usuário, desenvolvida com *HTML, CSS e JavaScript*. Ele recebe os dados processados pelo *back-end* e DT, exibindo gráficos, relatórios e informações relevantes para os operadores e tomadores de decisão. A comunicação com o *back-end* ocorre através da *API*, o que torna o processo ainda mais rápido, para um sistema mais fluido e eficiente, principalmente no que tange a experiência do usuário.

Esse fluxo de dados e interações entre as camadas do sistema permite que com a coleta de dados da camada física, seja processada e analisada de forma contínua, gerando informações úteis para a gestão e a tomada de decisões para operação e manutenção. A integração de todos

esses componentes, representada na Figura 17, visa ser eficaz e atribuir escalabilidade do sistema, que pode ser monitorado em tempo real por meio da interface visual do *front-end*.

Nesta seção, serão descritos os métodos utilizados no desenvolvimento da camada de *back-end* do sistema proposto. Esta camada é responsável por receber, processar e disponibilizar os dados provenientes da infraestrutura física, bem como por executar a lógica de controle, simulação e gerenciamento dos dispositivos monitorados.

Os principais métodos implementados incluem:

Aquisição de dados: *Endpoints* para recepção de dados dos sensores via *HTTP* ou *WebSocket*;

Persistência: Conexão com banco de dados relacional para armazenar histórico de medições e eventos;

Lógica de controle: Algoritmos responsáveis por gerar comandos para os atuadores com base em regras definidas ou modelos de predição;

Simulação DT: Módulo de integração com modelos computacionais que representam o comportamento esperado da infraestrutura física, permitindo análises preditivas e detecção de anomalias;

Serviços de API: Rotas para visualização dos dados em tempo real, gráficos históricos e status dos dispositivos.

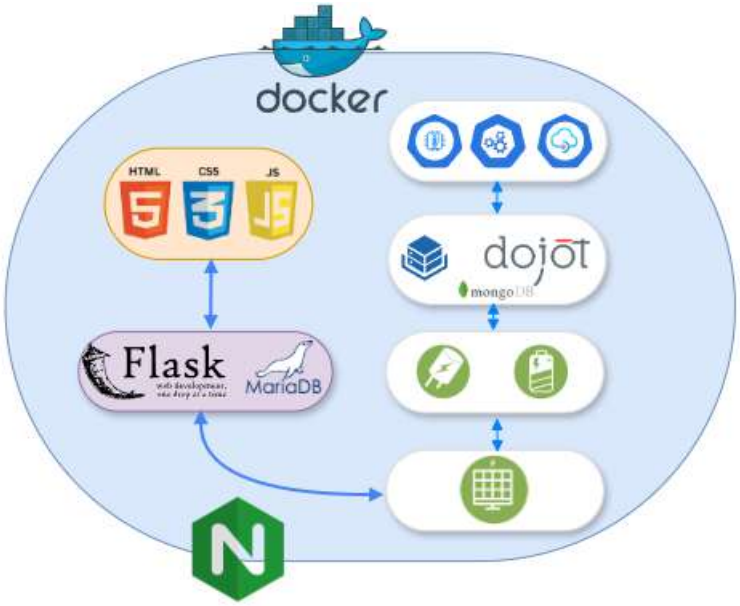
A seguir serão apresentados os pontos de implementação de toda a aplicação

4.3.1 Docker

Para garantir consistência entre os ambientes de desenvolvimento, testes e produção, foi adotada a virtualização com *Docker*. O uso de *containers Docker* permite encapsular todas as dependências do projeto em ambientes isolados, facilitando a implantação e escalabilidade do sistema. Com *Docker*, é possível garantir que o sistema funcione de maneira idêntica em diferentes máquinas, independentemente do ambiente em que esteja rodando.

Uma vez que os *softwares* foram estruturados e testados em um ambiente tradicional, devem posteriormente ser adicionados ao arquivo de configuração *Docker*, onde os *containers* foram direcionados de acordo com a tecnologia que contempla aquela aplicação. A Figura 18 representa a arquitetura que foi desenvolvida para que as aplicações comunicassem entre si.

Figura 18 – Serviços em *Container*



Fonte: O Autor

Para o gerenciamento das aplicações foi utilizado a ferramenta *Docker Compose*, o que no aspecto lógico compreende um arquivo *docker-compose.yml*, onde cada *software* incluído no escopo apresenta o seu arquivo de configuração. Basicamente, o *Docker Compose* simplifica a construção das imagens e inclusão de volumes com o objetivo de persistir os dados, bem como, conversando containers e passando os parâmetros necessários para a configuração do ambiente.

O Algoritmo 4.1 apresenta o arquivo de configuração *docker-compose.yml*. Inicialmente, é definida a versão que será utilizada, posteriormente, serão definidos os serviços que serão utilizados, neste caso, dividido da seguinte forma:

1. **SIGDEE**: Representa a aplicação principal do sistema. Esse serviço é responsável por executar o código-fonte da solução (localizado no diretório atual, *'build*: .') e contém os algoritmos e funcionalidades desenvolvidas. O *container* é nomeado como `sigdee`, configurado para reiniciar automaticamente (`restart: always`), e está conectado às redes internas *`db_network`* e *`web_network`*. O trecho `volumes` indica que há volumes compartilhados entre o *host* e o *container* (detalhes ocultos por confidencialidade na imagem).

2. ***nginx***: Serviço responsável pelo balanceamento de carga e gerenciamento das requisições *HTTP*. Utiliza a imagem *`nginx:latest`* e está configurado para expor a porta `5085` do *host* para a porta `5085` do *container*. Um volume local mapeia a configuração personalizada do *NGINX* (`./nginx:/etc/nginx/conf.d`). Está conectado à rede *`web_network`* e depende da inicialização prévia do serviço `sigdee`.

3. ***mariadb***: Serviço de banco de dados relacional baseado em `MariaDB`, na versão `11.4.2`. Esse container está configurado para mapear a porta padrão do *MariaDB* (`3306`) entre o *host* e o *container*, facilitando a comunicação entre a aplicação principal e o banco de dados. O *container* está conectado à rede `db_network`.

4. **Redes *Docker***: Duas redes do tipo `bridge` são definidas para segmentar a comunicação entre os serviços:

   - ***db_network***: Rede utilizada para a comunicação entre a aplicação principal (`sigdee`) e o banco de dados *MariaDB*.
   - ***web_network***: Rede utilizada para comunicação entre a aplicação e o servidor `nginx`, possibilitando o acesso via navegador.

Algoritmo 4.1 – Arquivo *Docker-Compose* (*docker-compose.yml*)

```
version: "3.3"

services:
  sigdee:
    container_name: sigdee
    restart: always
    build:
      .
    volumes:
      - ./##########################
      - ./##########################
    networks:
      - db_network
      - web_network

  nginx:
    container_name: nginx
    restart: always
    image: "nginx:latest"
    ports:
      - "5085:5085"
    volumes:
      - ./nginx:/etc/nginx/conf.d
    networks:
      - web_network
    depends_on:
      - sigdee

  mariadb:
    container_name: mariadb
    image: mariadb:11.4.2
```

```
32      ports:
33        - "3306:3306"
34
35  networks:
36    db_network:
37      driver: bridge
38    web_network:
39      driver: bridge
```

Fonte: O Autor

Para que os serviços sejam acionados, pode ser buscados diretamente de *containers* já disponíveis no *DockerHub*. Nesse caso, será construída a imagem que contém a infraestrutura necessária para carregar todos os itens do sistema, para que, posteriormente possa ser gerenciada pelo *Docker Compose*. A construção ocorre por meio de um arquivo de configuração chamado *Dockerfile*, responsável por carregar o ambiente necessário para a execução do sistema seguindo uma sequência de etapas definidas neste arquivo, ilustrado no Algoritmo 4.2, onde sequência percorrida, corresponde:

(a) **Selecionar uma imagem *Python* para ser o padrão utilizada no sistema:** O código inicia com o comando *FROM python:3.9*, o que define a imagem base do *Docker* como sendo a versão *Python* 3.9. Essa imagem é utilizada como o ambiente padrão para a execução do sistema, garantindo que todas as dependências e pacotes *Python* possam ser instalados corretamente.

(b) **Instalar as dependências da versão da imagem escolhida:** Essas variáveis evitam a criação de arquivos .pyc e garantem que a saída do *Python* seja exibida diretamente, sem *buffering*. Isso é importante para depuração e desempenho em *containers Docker*.

(c) **Copiar os arquivos da aplicação:** O comando COPY copia o arquivo *env.sample* para o arquivo *.env* dentro do *container*. Isso é utilizado para configurar variáveis de ambiente necessárias para o funcionamento da aplicação, bem como, é realizada a cópia do arquivo *requirements.txt* que lista todas as dependências necessárias para a aplicação, como bibliotecas *Python*, para o *container*.

(d) **Instalar as dependências do *software* por meio do *pip*:** A instalação das dependências é realizada utilizando o *pip*, o gerenciador de pacotes *Python*. Primeiramente, garante que o *pip* esteja atualizado para a versão mais recente. Em seguida, instala as dependências listadas no arquivo *requirements.txt*, sem usar o cache, o que ajuda a reduzir o tamanho final da imagem *Docker*.

(e) **Copiar os arquivos da aplicação:** O comando COPY . . copia todos os arquivos do diretório atual no sistema local para o *container*. Isso inclui o código-fonte da aplicação e quaisquer outros arquivos necessários para a execução do sistema.

(f) **Executar a aplicação com o *Gunicorn*:** Por fim, o *Dockerfile* configura o comando para iniciar o servidor , que irá executar a aplicação *Python*. Além disso, o *Gunicorn* será executado com o arquivo de configuração *gunicorn-cfg.py* e a aplicação será iniciada com o comando run:app, que se refere ao arquivo onde a instância do *Flask* é configurada.

Algoritmo 4.2 – Exemplo de *Dockerfile* para aplicação Python com Gunicorn

```
FROM python:3.9

# set environment variables
ENV PYTHONDONTWRITEBYTECODE 1
ENV PYTHONUNBUFFERED 1

COPY env.sample .env
COPY requirements.txt .

# install python dependencies
RUN pip install --upgrade pip
RUN pip install --no-cache-dir -r requirements.txt

COPY . .

# gunicorn
CMD ["gunicorn", "--config", "gunicorn-cfg.py", "run:app"]
```

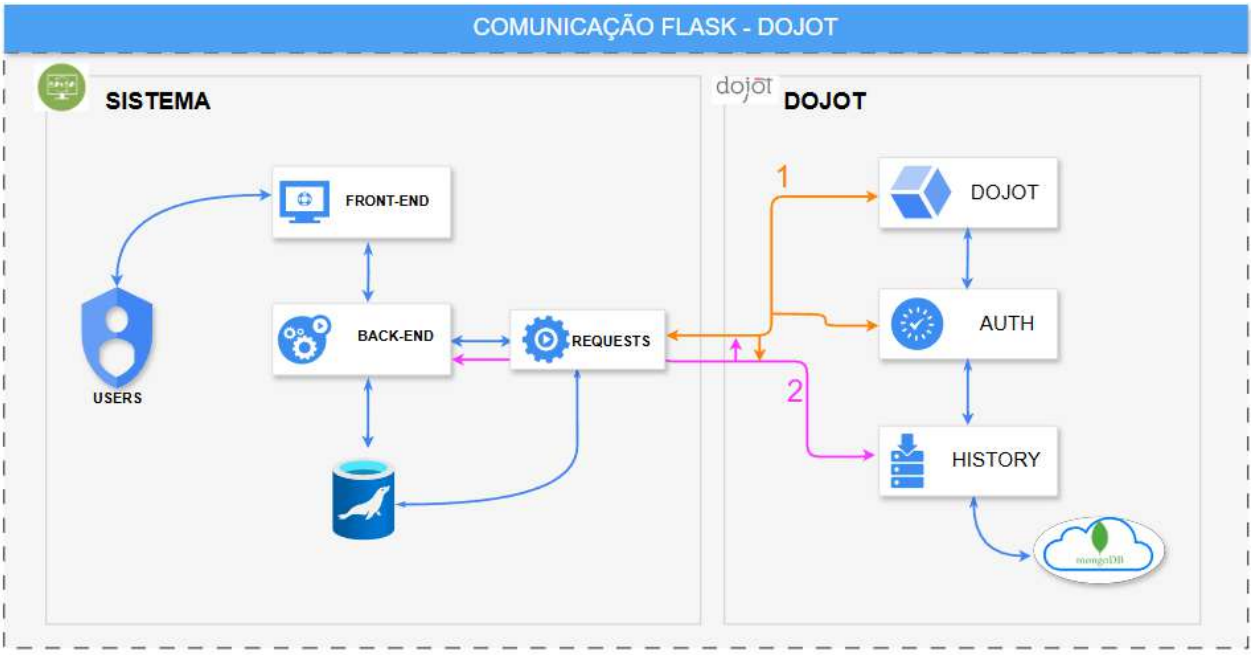
Fonte: O Autor

4.3.2 Dojot

A comunicação direta com o *middleware IoT* é por meio da plataforma DOJOT, que recebe informações e armazena no seu próprio banco de dados. A partir disso, para realizar a aquisição destes dados é necessário requisitar por meio da *API History* da própria plataforma. Para isso, há a necessidade de ser usuário cadastrado na DOJOT, solicitar e receber o *Token* de acesso, afinal, a *API* requer que seja informado o parâmetro de segurança por meio de um token *JWT (JSON WEB TOKEN).*

Utilizando *Python*, por meio da biblioteca *requests* é realizada a conversação com a aplicação *Flask*, obtendo um *JSON* com os dados brutos vindos da DOJOT. Na Figura 19 é ilustrado o fluxo de como essa informação chega de um ponto até o outro. No primeiro ponto, em laranja, temos a requisição direta para a DOJOT através de uma função com as credenciais de acesso para obtenção do *token*, já no segundo ponto, em rosa, é representada a requisição dos dados históricos obtidos a partir do banco de dados da DOJOT.

Figura 19 – Fluxo de Comunicação FLASK - DOJOT



Fonte: O Autor

Em seguida, para que seja possível utilizar-se das informações geradas pelos dispositivos incluídos na DOJOT, é necessário a autenticação na plataforma, para isso, foi utilizado o método de requisição de dados por meio da biblioteca *requests* do *Python*, onde é requisitada a criação de um *token* de acesso para consumo de dados via *API*, conforme ilustrado no Algoritmo 4.3.

Algoritmo 4.3 – Exemplo de autenticação com *token* na Dojot

```
def get_dojot_token():
    username = "######################"
    password = "######################"
    dojot_ip = "######################"
    url = f"http://{dojot_ip}:8000/auth"

    headers = {
        "Content-Type": "application/json"
    }

    data = {
        "username": username,
        "passwd": password
    }

    response = requests.post(url, json=data, headers=headers)

    if response.status_code == 200:
        # token da resposta
        return response.json().get("jwt")
```

```
    else:
        raise Exception(f"Erro ao autenticar: {response.status_code} - {
    response.text}")
```

Fonte: O Autor

Após, para requisitar dados históricos do dispositivo desejado é criada outra função de consumo, a qual busca todos os dados que foram gerados em um determinado tempo, ilustrado no Algoritmo 4.4

Algoritmo 4.4 – Requisição de Dados Históricos DOJOT

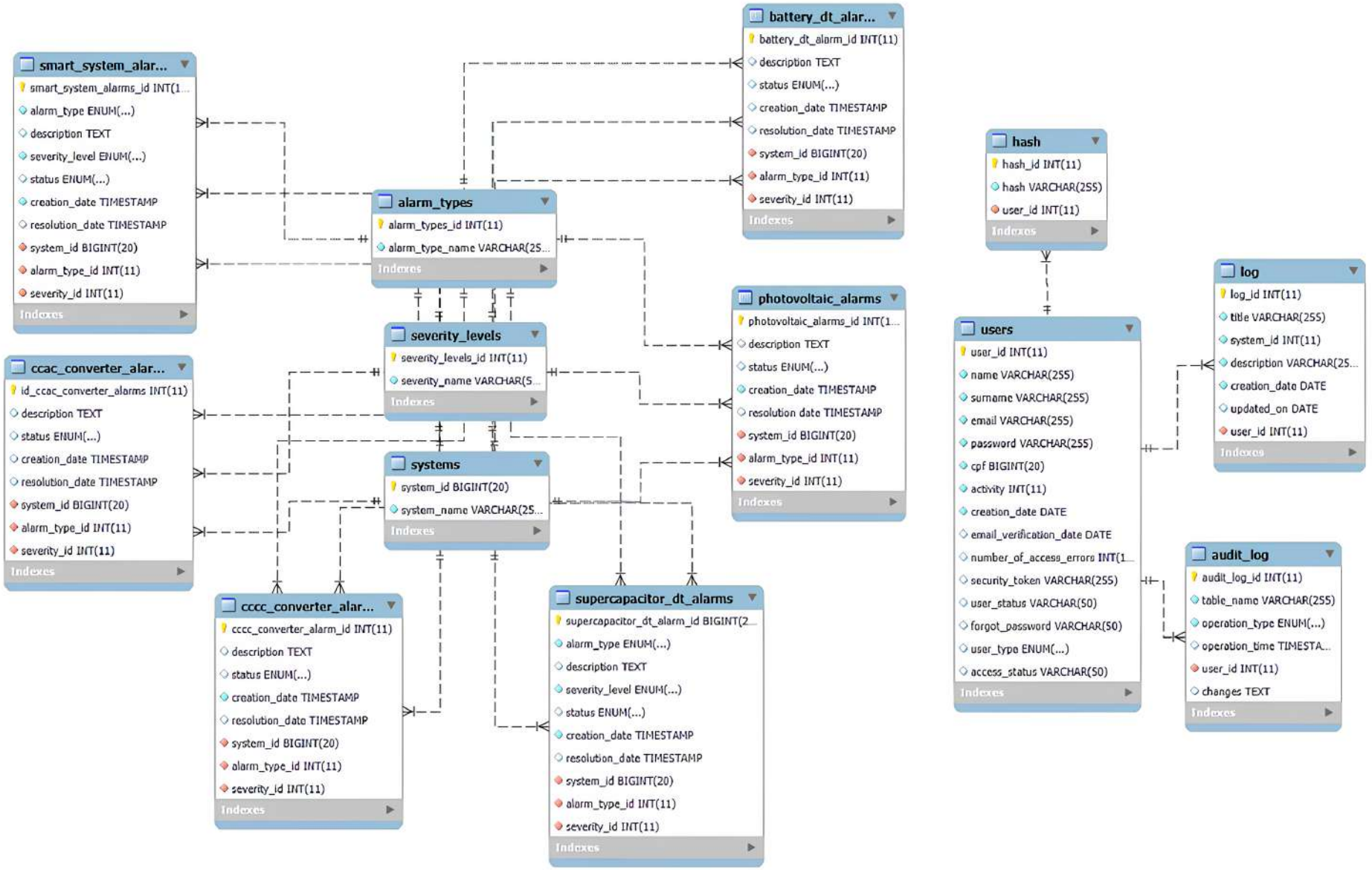
```
def get_device_history(token, info_battery):
    # URL com os parâmetros
    dojot_ip = "##########"
    device_id = "##########"
    attrs = "##########"

    history_battery = f"http://{dojot_ip}:8000/history/device/{device_id}"

    # query
    params = {
        "attr": attrs
    }

    # Cabeçalhos da requisição (incluindo o token de autenticação)
    headers = {"Authorization": f"Bearer {token}"}

    # Fazendo a requisição GET
    response = requests.get(history_battery, headers=headers, params=params
    )

token = get_dojot_token()
```

Fonte: O Autor

4.3.3 Banco de Dados

No contexto dos Modelos de Entidade-Relacionamento (MER), o banco de dados utilizado no desenvolvimento da aplicação é relacional, sendo composto por entidades (representadas pelas tabelas) e relacionamentos entre essas entidades (1:1, 1:N, N:1 ou N:N). Cada tabela contém atributos específicos que armazenam dados, e as relações entre elas são estabelecidas por chaves primárias e estrangeiras, obtendo a integridade referencial e a consistência dos dados no sistema. A Figura 20 ilustra o Modelo Entidade Relacionamento.

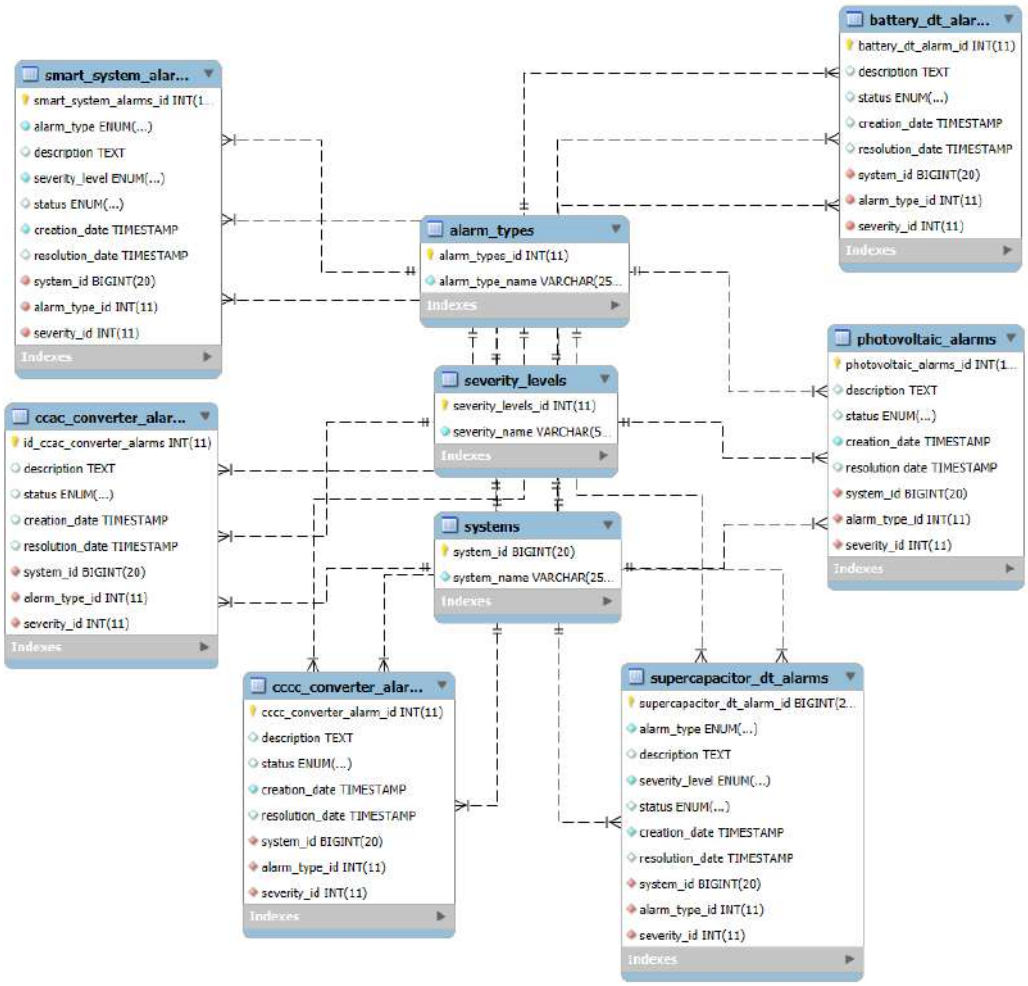
Figura 20 – Modelo Entidade Relacionamento do *Software*



Fonte: O Autor

O sistema contém um módulo de alarmes, que observa cada parte do processo de monitoramento. Cada tipo de alarme é armazenado em uma tabela separada, com colunas semelhantes, mas com especificidades para cada tipo de alarme, como alarme de sistema, alarme de fotovoltaico, alarme de bateria, entre outros. As tabelas de alarmes são interligadas para registrar o nível de severidade do alarme e associar cada alarme a um sistema específico. A utilização de chaves estrangeiras entre essas tabelas visa a integridade referencial dos dados, para que as informações relacionadas aos alarmes sejam consistentes. A Figura 21 mostra como estão associadas essas tabelas.

Figura 21 – Tabelas de Alarmes



Fonte: O Autor

A tabela de usuários trata do acesso ao *software*, sendo definidos como:

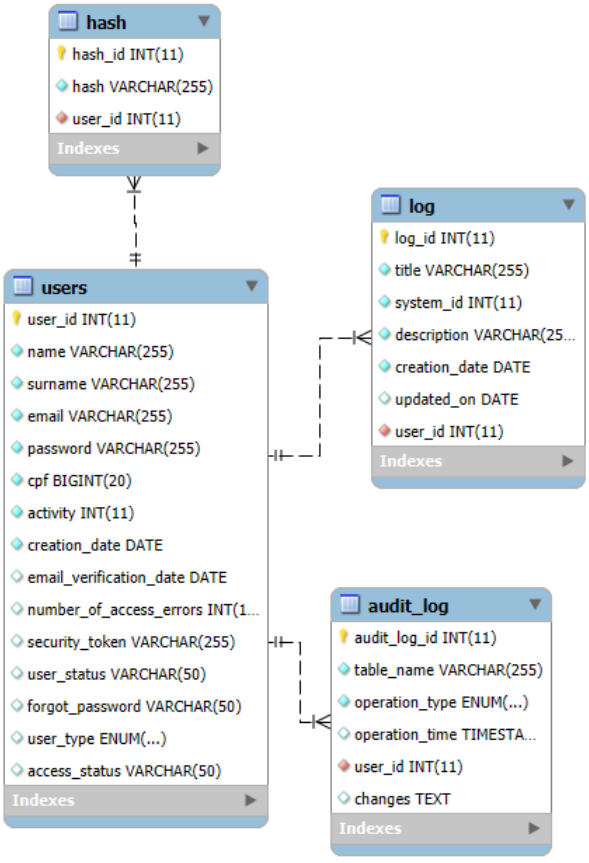
1. A tabela *users* armazena informações de login, como *name, surname, email* e dados de autenticação como *password*. Também contém informações sobre o status de atividade do usuário *(activity, user_status)*. Além disso, existe uma referência de segurança com *number_of_access_errors*, indicando tentativas de *login* inválidas.

2. A tabela *log* registra eventos do sistema, com informações como *title, description, e system_id*, além de registrar a data de criação e atualizações feitas. Úteis para o monitoramento e auditoria de eventos no sistema.

3. A tabela *audit_log* tem uma função clara de auditoria, registrando as ações dos usuários, com informações sobre o tipo de operação, data e os detalhes da modificação realizada, como o *operation_type*. Ela está relacionada à tabela *users*, o que implica que cada ação registrada está associada a um usuário específico.

4. A tabela *hash* armazena valores de *hash* para uma verificação de segurança, provavelmente relacionada à autenticação dos usuários. Ela está relacionada com a tabela *users*, indicando que cada *hash* está vinculado a um usuário específico.

Figura 22 – Tabela de Usuários

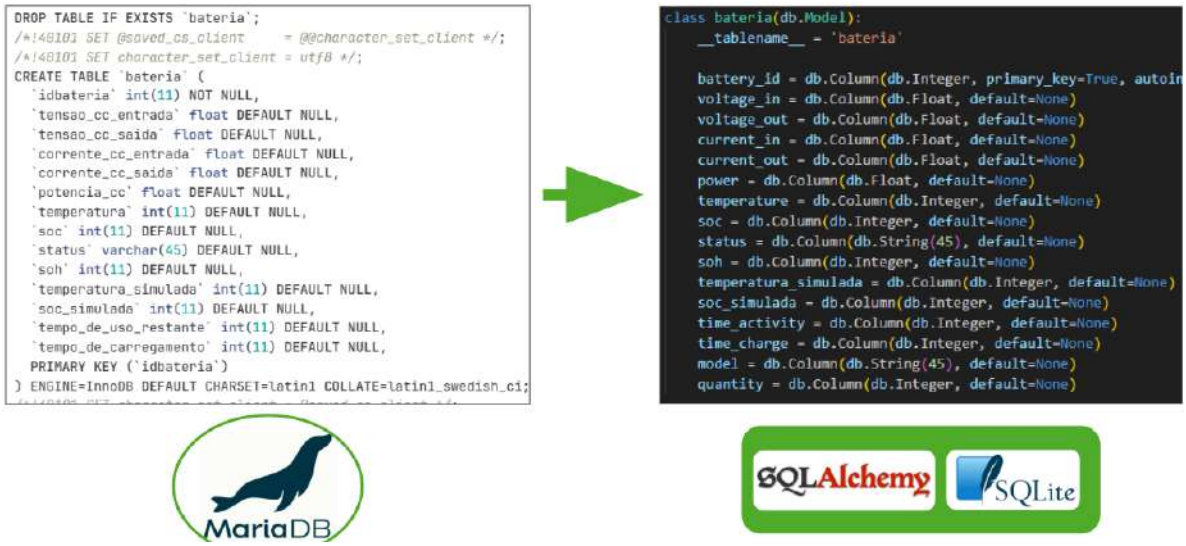


Fonte: O Autor

O modelo adotado favorece a organização e o armazenamento estruturado dos dados gerados pelos módulos de DT, sistema fotovoltaico e controle de acesso. A implementação do banco de dados foi realizada com o Sistema de Gerenciamento de Banco de Dados (SGBD)

MariaDB. Durante esse processo, identificou-se o potencial do *SQLAlchemy* como uma ferramenta de mapeamento objeto-relacional (ORM), que, em conjunto com o *SQLite* e *MariaDB*, permite a definição da estrutura do banco por meio de classes *Python*, facilitando a migração e a manipulação dos dados com base na linguagem *SQL*. A Figura 23 exemplifica como se dá essa conversão.

Figura 23 – Conversão para *SQLAlchemy*



Fonte: O Autor

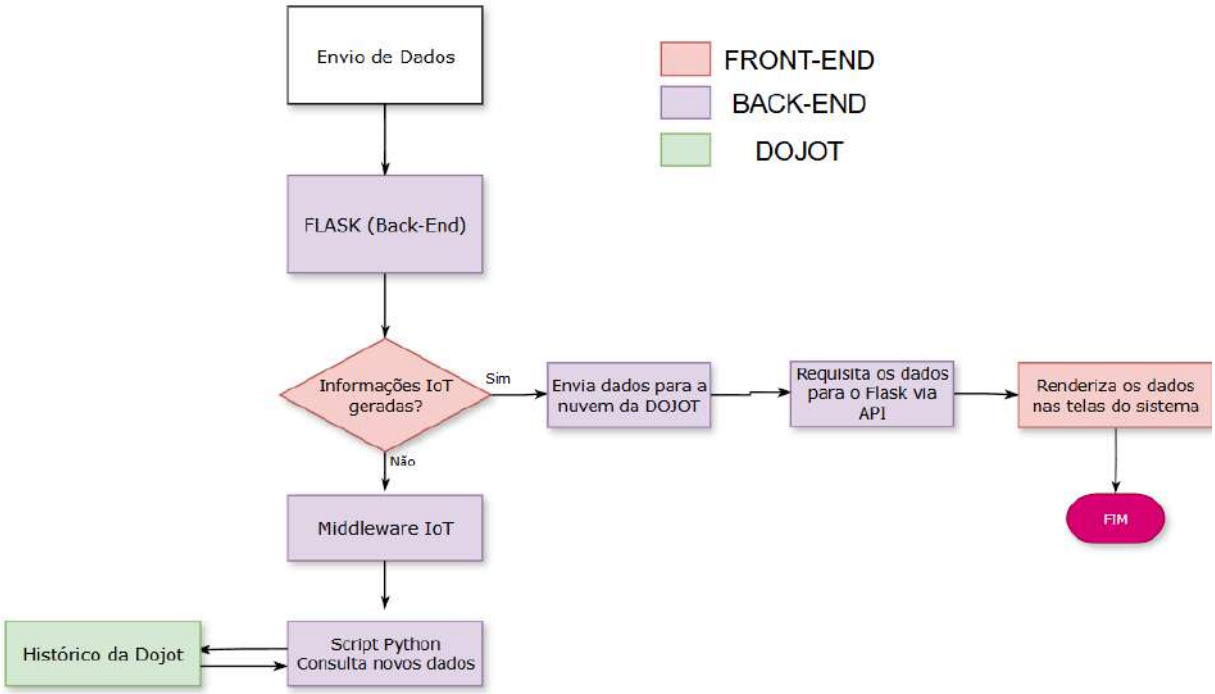
A partir dessa estrutura, as operações CRUD — *Create* (criar), *Read* (ler), *Update* (atualizar) e *Delete* (excluir) — foram integradas ao escopo de desenvolvimento de forma facilitada com o auxílio do *SQLAlchemy*. Por fim, essa integração permitiu abstrair comandos *SQL* complexos, além de otimizar a comunicação com o banco de dados com uma maior produtividade no desenvolvimento da aplicação.

4.3.4 Processamento de Dados

De modo geral, o gerenciamento de dispositivos *IoT* deve ser realizado de forma simultânea, especialmente neste caso, em que o sistema envolve o monitoramento de diversas variáveis essenciais para alimentar os módulos de cada parte do *software*. Considerando que está sendo utilizado um *microframework* para o desenvolvimento do sistema, a interação entre as tecnologias é facilitada, pois as bibliotecas *Python* oferecem funções que permitem a integração eficiente entre os diferentes componentes.

A Figura 24 representa como se dá o processo de envio de dados para o usuário final, neste caso, é feito em tempo real na medida em que os sensores são atualizados com novas informações. Há um tempo de resposta inicial de cerca de 5 segundos, que refere-se a execução da aplicação e das ferramentas que compõem todo escopo, após, são requisitados novos dados na medida em que são gerados pelos módulos.

Figura 24 – Fluxo de envio de dados



Fonte: O Autor

O sistema depende do serviço da aplicação *IoT* DOJOT, o que implica que, para seu funcionamento adequado, é essencial que o DOJOT esteja operando corretamente. Portanto, a disponibilidade e o bom desempenho do serviço DOJOT são fundamentais para garantir o funcionamento eficiente do sistema. Para que sejam carregados os dados de acordo com cada módulo foi criada uma função específica que requisita dados, sem precisar reexecutar toda a aplicação, o que ocorre através de um *endpoint*.

Um *endpoint* é um ponto de comunicação ou interface em um sistema de *software*, especialmente em serviços *web*, que permite que diferentes partes de um sistema ou sistemas diferentes se comuniquem entre si. É o local onde as requisições *HTTP (GET, POST, PUT, DELETE)* são enviadas e as respostas são recebidas. O fluxo de uma requisição é iniciada pelo cliente, que é enviada para o *endpoint*. O *endpoint*, por sua vez, realiza a busca pelos dados em um servidor ou serviço específico. Após processar a solicitação, o serviço envia a resposta de volta ao cliente, completando o ciclo de comunicação entre eles, como ilustrado na Figura 25.

Figura 25 – Interação com *Endpoints*

Fonte: O Autor

Esse método utilizado contribui para atribuir mais leveza e fluidez em todo o processo, fazendo com que não seja necessário carregar todos os componentes do sistema de uma vez. Em vez disso, apenas as informações solicitadas são carregadas dinamicamente, o que reduz o uso de recursos e melhora a performance geral do sistema. Isso também proporciona uma experiência mais ágil para o usuário, já que ele pode interagir com o sistema de forma mais rápida e eficiente, sem precisar esperar pelo carregamento completo de todos os dados ou funcionalidades. Esse modelo é fundamental, especialmente em sistemas distribuídos ou com grande volume de dados.

4.4 Considerações Finais do Capítulo

Este capítulo apresentou a metodologia e a arquitetura adotadas no processo de desenvolvimento do SIGDEE — Sistema Inteligente de Gestão e Despacho de Energia Elétrica. Foram descritas as etapas de definição de requisitos, prototipação, integração e testes, com foco em implementar uma solução robusta, modular e escalável.

Foram detalhadas as especificações de *hardware* e *software*, incluindo o uso de servidor dedicado, tecnologias como *Docker* e *Flask*, além da estrutura distribuída do sistema. A arquitetura integra módulos de *Digital Twin* com dados provenientes de sensores, para que se obtenha simulações, diagnósticos e análises preditivas a partir dos dados.

Também foram apresentadas as estratégias de implementação das camadas *back-end* e *front-end*, com uso de *SQLAlchemy*, interfaces *web* dinâmicas e comunicação assíncrona via API. O fluxo de dados foi mapeado da camada física até o usuário final, com a interoperabilidade entre os serviços (*Flask, Dojot*, banco de dados e *front-end*) em ambientes híbridos de energia.

O próximo capítulo abordará os resultados da implementação prática da aplicação em desenvolvimento.

5 RESULTADOS

Este capítulo apresenta os resultados obtidos com o desenvolvimento da solução proposta. São descritas as telas desenvolvidas, suas funcionalidades específicas e a lógica de funcionamento de cada módulo do sistema, sobre como os requisitos projetados estão sendo atendidos. A apresentação dos resultados visa demonstrar a aplicabilidade da solução em um ambiente real ou simulado, com destaque para a usabilidade e a organização das informações.

Serão abordados, ainda, os mecanismos de autenticação e controle de acesso, a estrutura dos menus de navegação, os indicadores visuais e os componentes de gerenciamento disponíveis ao usuário. Também são detalhadas as configurações implementadas para garantir o funcionamento do sistema e integração com os dispositivos e serviços da arquitetura IoT. Com isso, este capítulo busca oferecer uma visão abrangente da aplicação desenvolvida e de seu potencial de uso em contextos de supervisão e automação energética.

5.1 Configurações Gerais do *Software*

Desde o levantamento de requisitos até a etapa de testes do sistema, foram adotados os padrões de identidade visual, ícones e fontes para serem seguidos durante todo o desenvolvimento do sistema e padronização de todo o escopo. O sistema contém diversas telas que foram construídas de forma modular e ajustável para que pudessem ser facilmente adaptadas às possíveis ajustes futuros. Cada módulo foi projetado com foco na usabilidade e na experiência do usuário, visando oferecer uma interface intuitiva e eficiente.

5.2 Sistema Inteligente de Gestão e Despacho de Energia Elétrica - SIGDEE

A nomenclatura do sistema reflete a integração completa de sua infraestrutura de módulos, que trabalham em conjunto para otimizar a gestão e despacho de energia elétrica. Entre os componentes principais que o constituem, destacam-se o Sistema Fotovoltaico, o DT de Bateria e o DT de Supercapacitor, todos interconectados e desempenhando papéis importantes no monitoramento e controle da energia gerada e armazenada.

5.2.1 Login

A tela de *login* é a principal tela anterior ao sistema, quando o usuário ultrapassa essa barreira, terá a permissão de acesso aos recursos do sistema, para que efetue o acesso, é necessário estar cadastrado no sistema. A interface foi desenvolvida para ser minimalista e intuitiva. Nesta

tela, estão presentes os campos de usuário e senha, necessários para o acesso. Além disso, há a caixa de seleção para manter o usuário conectado, bem como, as opções de redefinir a senha e solicitar cadastro no sistema . Essas informações podem ser vistas na Figura 26.

Figura 26 – Tela de login

Fonte: O Autor

5.2.2 Cadastro

Caso o usuário não possua um cadastro para fazer *login* no sistema, é possível solicitar acesso por meio da tela de cadastro, selecionando a opção "Cadastre-se". Nesta tela, estão presentes os campos necessários para solicitar cadastro no sistema:

- Usuário: Nome de identificação que será utilizado para *login* no sistema.
- Nome: Primeiro nome do usuário.
- Sobrenome: Último nome do usuário.
- Número do CPF: Cadastro de Pessoa Física do usuário (somente números, no formato 000.000.000-00).
- Telefone: Número de contato com DDD.
- Perfil: Tipo de acesso ou função do usuário no sistema (gestor, operador ou monitoramento).
- *E-mail*: Endereço de *e-mail* válido do usuário.
- Senha: Código secreto de acesso ao sistema.

Além disso, há duas caixas de seleção para aceitar os “Termos de uso e condições” do *software* e as “Políticas de privacidade”. Caso o usuário clique nesses itens, ele é redirecionado para essas respectivas telas para que leia as documentações referentes aos Termos e Privacidade. Na Figura 27 é ilustrado como se configura essa tela.

Figura 27 – Tela de Cadastro

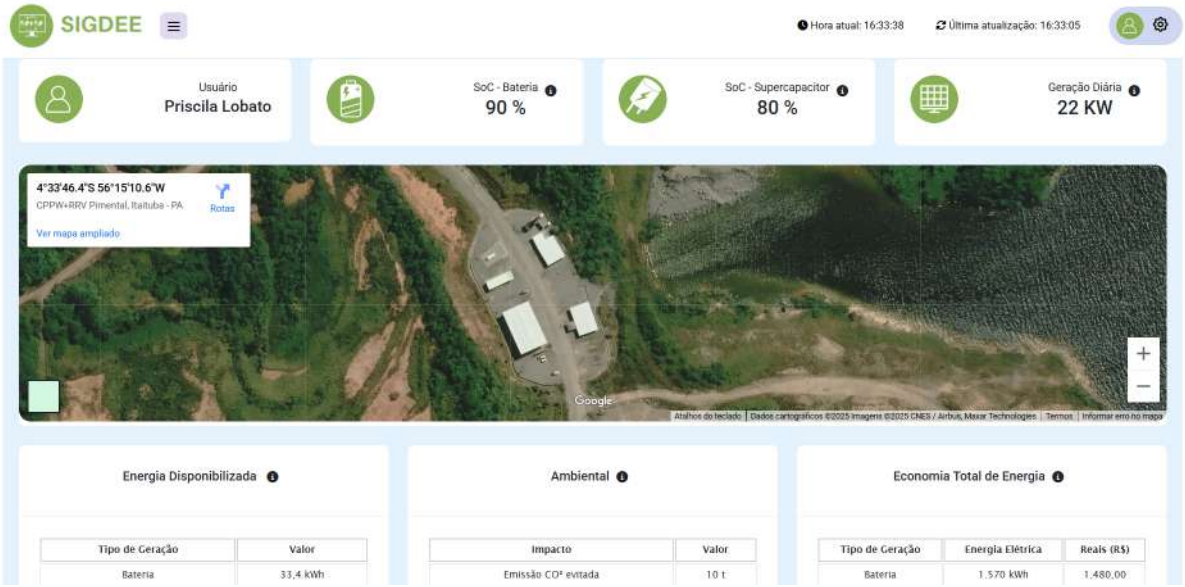
Fonte: O Autor

5.2.3 Dashboard (Home)

O *Dashboard* serve como a tela inicial e a principal interface de visualização do sistema, oferecendo ao usuário uma visão geral do desempenho e status de todos os módulos. Ele é composto por indicadores chave que mostram o Estado de Carga (SoC) do DT de Bateria e DT do Supercapacitor, assim como, a geração diária do Sistema Fotovoltaico. O objetivo principal do *Dashboard* é fornecer uma visão consolidada e fácil de interpretar, permitindo que o usuário acompanhe a produção de energia, o status de carga e o impacto econômico do sistema.

Ademais, o *Dashboard* foi estruturado e a atualização dos dados online ocorre por requisições assíncronas (via *Fetch* API/AJAX), executadas periodicamente para consultar os valores mais recentes disponíveis no banco de dados. A Figura 28 mostra como isso se dá na tela.

Figura 28 – Tela de *Dashboard (Home)*



Fonte: O Autor

Além disso, oferece ainda a localização precisa dos sistemas instalados, acesso rápido às configurações e alertas. Ademais, o sistema é bastante intuitivo, permitindo que o usuário tenha acesso rápido aos módulos clicando apenas nos ícones que são voltados a eles, isso permite que o usuário tenha mais liberdade para caminhar entre páginas. Há ainda botões de informação rápida para auxiliar o uso, além de monitorar alarmes do sistema, como mostrado na Figura 29.

Figura 29 – Tela de *Dashboard (Home)*

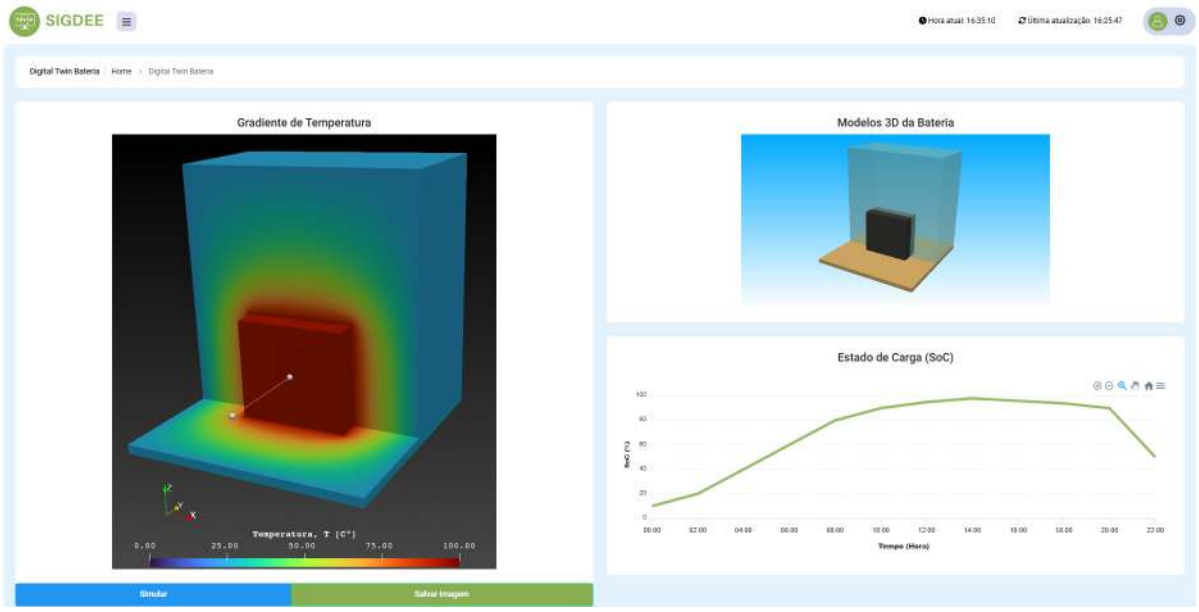
Fonte: O Autor

5.2.4 *Digital Twin* Bateria

A tela do DT Bateria permite ao usuário visualizar uma representação digital online da bateria do sistema, que simula seu comportamento com base nos dados coletados do dispositivo físico. Essa representação é construída a partir da integração contínua entre o modelo numérico do Método dos Elementos Finitos (MEF) e os dados recebidos dos sensores, permitindo que parâmetros como temperatura, corrente e tensão alimentem diretamente o modelo digital.

A renderização gráfica utiliza bibliotecas de visualização 3D para apresentar o gradiente de temperatura em °C, como ilustrado na Figura 30, onde está a distribuição térmica na bateria. Ao lado do modelo térmico, uma galeria dinâmica apresenta representações do aspecto físico da bateria em diferentes modelos selecionados, os quais são carregados a partir de um repositório de imagens associado ao sistema. Esse recurso foi implementado com técnicas de atualização assíncrona (AJAX/fetch), garantindo que a troca de modelos e estados visuais ocorra online, sem necessidade de recarregar a página. O gráfico com Estado de Carga (SoC) da bateria carrega dados a partir da API da Dojot e renderiza na página.

Figura 30 – Tela de *Digital Twin* Bateria



Fonte: O Autor

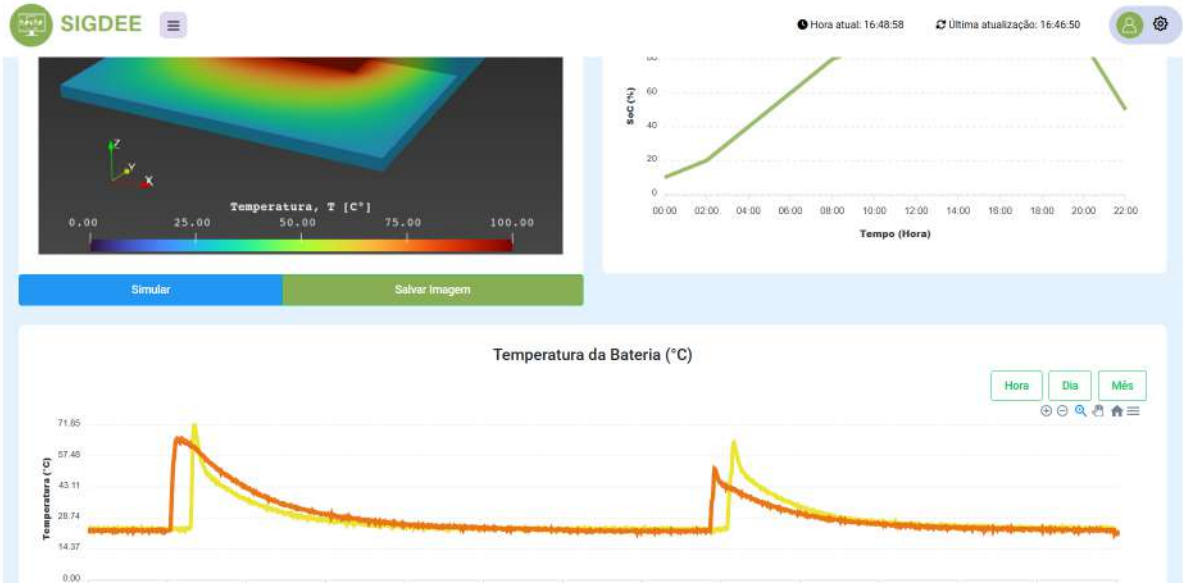
Em seguida, é apresentada a visualização do Estado de Carga (SoC) da bateria ao longo de um intervalo de 24 horas. Além disso, é disponibilizada a opção "simular"para que o usuário possa solicitar uma nova simulação do DT, uma vez que essa função está implementada no *back-end*. Além disso, o usuário pode exportar a imagem gerada da simulação no formato .jpg. Essa visualização fornece informações detalhadas sobre a temperatura online do objeto físico, o nível de eficiência e o desempenho do armazenamento da bateria.

Outro aspecto relevante é o gráfico que exibe a Temperatura da Bateria (em °C), conforme

mostrado na Figura 31, sendo totalmente exportável, tanto no formato de imagem (.png), quanto dados numéricos (.csv), nesse gráfico são indicadas três faixas de temperatura, que permitem uma análise mais precisa do comportamento térmico da bateria, sendo elas:

- **Temperatura Ambiente:** Refere-se à temperatura do ambiente externo ao sistema, que pode influenciar diretamente o desempenho e a eficiência da bateria. O controle e monitoramento dessa variável são importantes para otimizar o funcionamento do sistema.
- **Temperatura da Simulação:** Representa a temperatura registrada durante a simulação do DT, a qual modela as condições de operação da bateria em diferentes cenários. Essa temperatura é essencial para prever como a bateria se comportará em condições específicas.
- **Temperatura da Bateria:** Indica a temperatura interna da bateria durante o seu processo de carregamento ou descarregamento. O monitoramento constante dessa variável é importante para evitar superaquecimento e garantir a longevidade e segurança do sistema de armazenamento de energia.

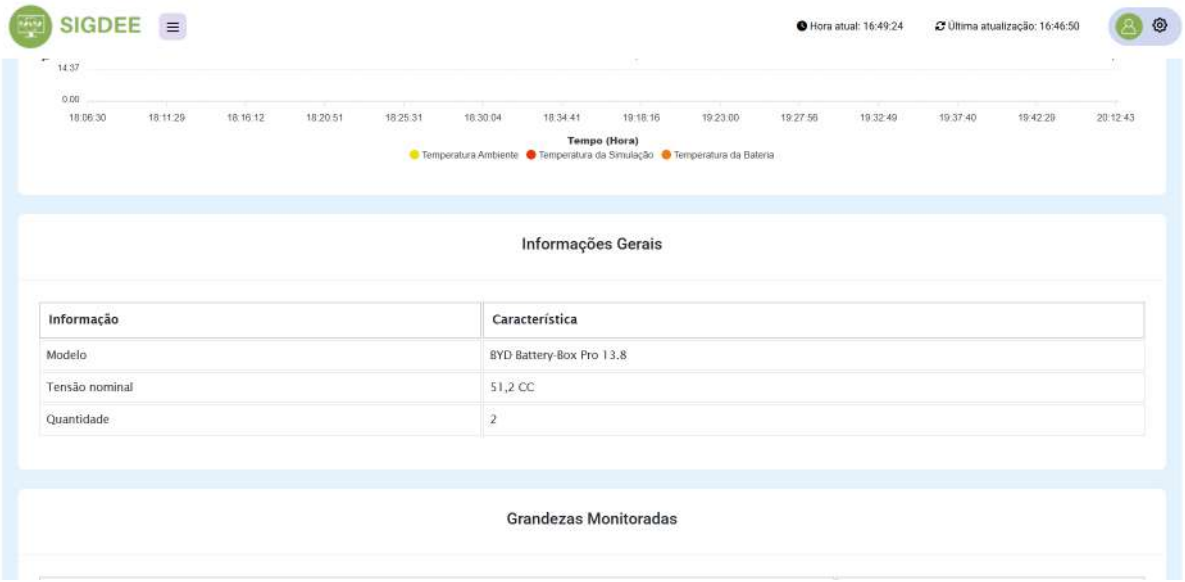
Figura 31 – Tela de *Digital Twin* Bateria



Fonte: O Autor

Em relação as Informações Gerais do modelo de Bateria, há uma pequena tabela que consta o modelo e as características do módulo, conforme a Figura 32.

Figura 32 – Tela de *Digital Twin* Bateria



Fonte: O Autor

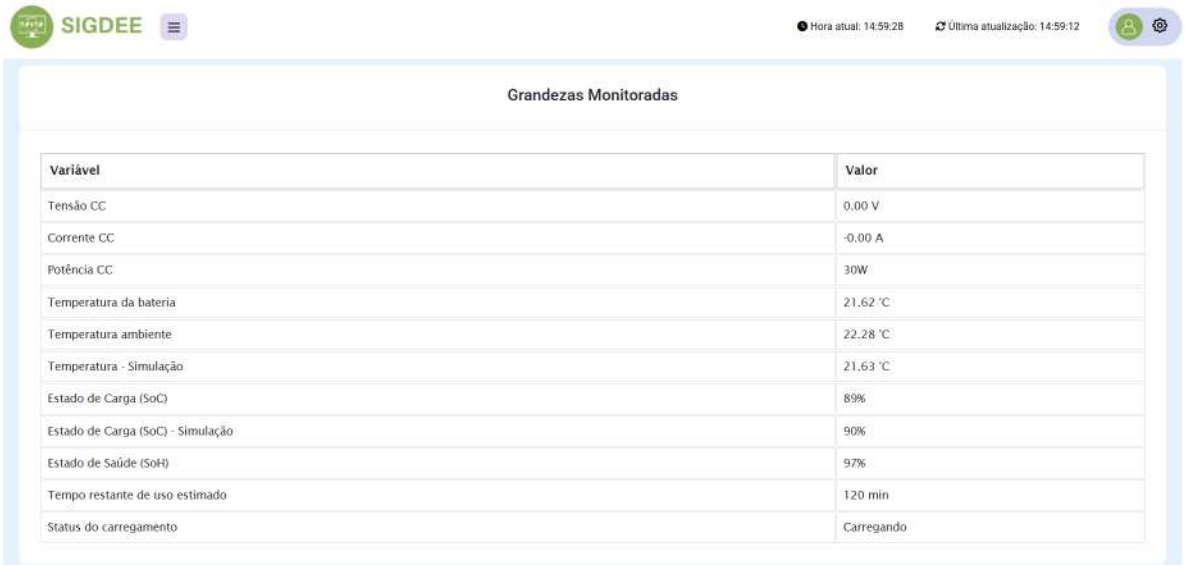
No final da página, estão definidas Grandezas que são monitoradas na página de DT de Bateria. As variáveis que compõe o sistema, são:

- **Tensão CC:** Tensão em corrente contínua aplicada no sistema, importante para o carregamento adequado da bateria.
- **Corrente CC:** Corrente de carga ou descarga em corrente contínua que circula na bateria.
- **Potência CC:** Energia fornecida ou consumida pelo sistema em corrente contínua, calculada a partir da tensão e corrente.
- **Temperatura da Bateria:** Temperatura interna da bateria durante os processos de carga e descarga.
- **Temperatura Ambiente:** Temperatura do ambiente ao redor do sistema, que afeta o desempenho da bateria.
- **Temperatura da Simulação:** Temperatura simulada no modelo DT de Bateria, projetada para prever o comportamento térmico da bateria.
- **Estado de Carga (SoC):** Quantidade de carga restante na bateria, expressa como uma porcentagem.
- **Estado de Carga (SoC) da Simulação:** Estimativa do SoC baseada no modelo DT de Bateria, simulando o comportamento da bateria.
- **Estado de Saúde (SoH):** Condição geral da bateria, medindo a degradação da capacidade em relação à sua capacidade original.

- **Tempo de uso restante estimado:** Tempo estimado até que a carga da bateria se esgote, baseado no SoC atual.
- **Status do carregamento:** Estado atual da bateria em relação ao processo de carregamento.

A Figura 33 representa como estão alocadas essas variáveis no sistema.

Figura 33 – Tela de *Digital Twin* Bateria

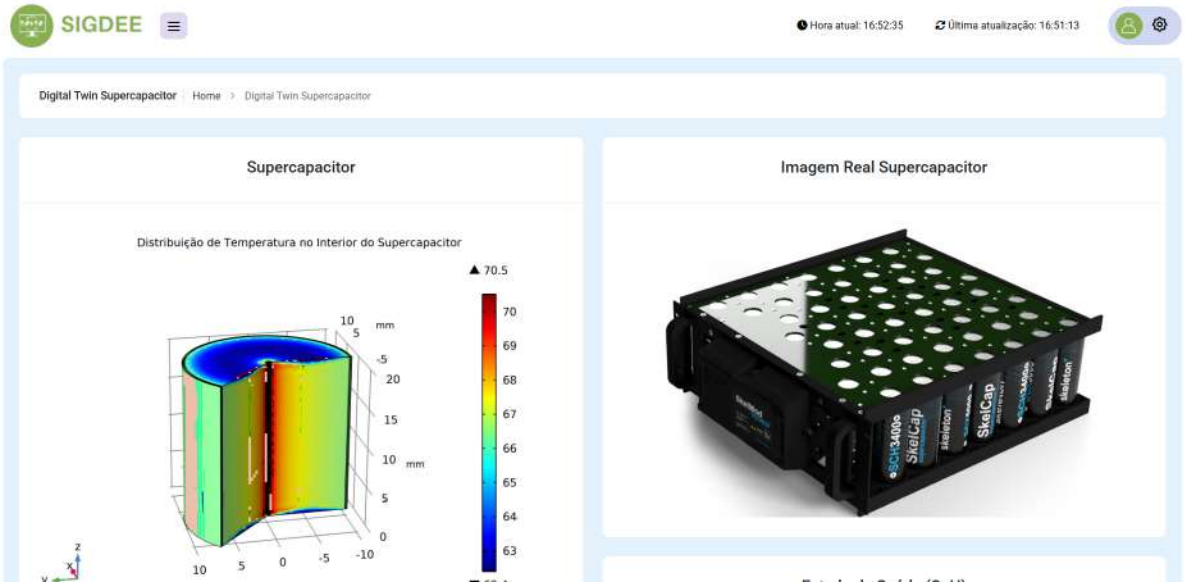


Fonte: O Autor

5.2.5 Digital Twin Supercapacitor

A tela do DT Supercapacitor funciona de maneira similar à do DT Bateria, mas focada no monitoramento e visualização do Supercapacitor do sistema. Permite ao usuário acompanhar o comportamento do supercapacitor, que tem figura importante no armazenamento de energia. Essa tela visa fornecer dados como o nível de carga e a performance do supercapacitor online. A Figura 34 ilustra a parte inicial da página, com o Gradiente de Temperatura do Supercapacitor e a imagem real do módulo, onde também é disponibilizado o botão "simular"para que o usuário possa solicitar uma nova simulação do DT, assim como, salvar essa imagem.

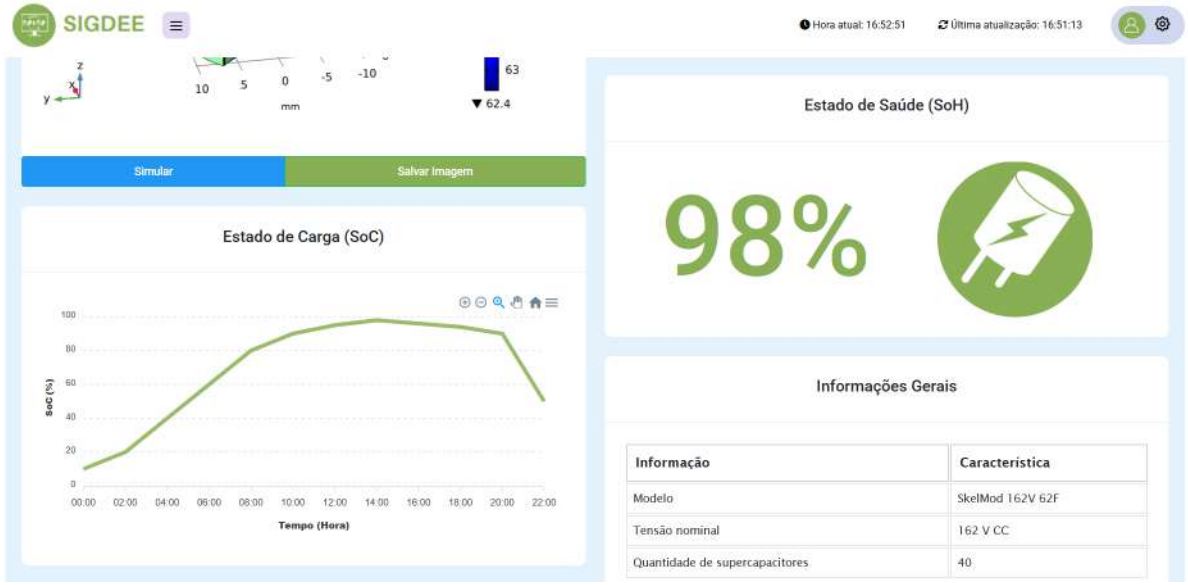
Figura 34 – Tela de *Digital Twin* Supercapacitor



Fonte: O Autor

Em seguida, em disposição para o usuário, são levantados os aspectos de Estado de Carga (SoC) em forma de gráfico diário (24h), onde ao direcionar o *mouse* é possível notar essas variáveis. Ainda mais, é incluída a possibilidade de exportar a imagem do gráfico (.png) e os dados numéricos contidos na tabela (.csv) com o Estado de Saúde (SoH) em porcentagem. Nas Informações Gerais, há informações relevantes em relação ao Supercapacitor. A Figura 35 ilustra essa tela.

Figura 35 – Tela de *Digital Twin* Supercapacitor



Fonte: O Autor

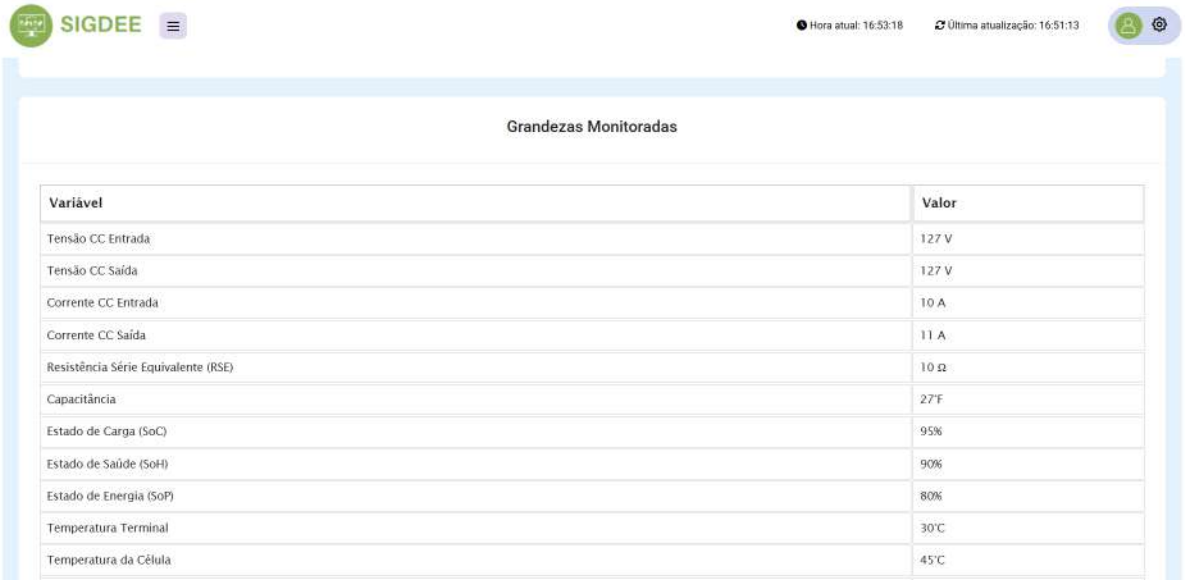
No final da página, são definidas das Grandezas que são monitoradas na página de DT

de Supercapacitor. As variáveis que compõe o sistema, são:

- **Tensão CC de Entrada:** Tensão de corrente contínua aplicada na entrada do supercapacitor.
- **Tensão CC Saída:** Tensão de corrente contínua na saída do supercapacitor.
- **Corrente CC Entrada:** Corrente de carga que entra no supercapacitor durante o processo de carregamento.
- **Corrente CC Saída:** Corrente de descarga que sai do supercapacitor quando a energia é liberada.
- **Resistência Série Equivalente (RSE):** Resistência interna do supercapacitor que afeta sua eficiência de carga e descarga.
- **Capacitância:** Capacidade do supercapacitor de armazenar carga elétrica.
- **Estado de Carga (SoC):** Quantidade de carga armazenada no supercapacitor, expressa como uma porcentagem do total disponível.
- **Estado de Saúde (SoH):** Condição geral do supercapacitor, refletindo sua capacidade de manter a eficiência ao longo do tempo.
- **Estado de Energia (SoP):** Quantidade de energia armazenada no supercapacitor em determinado momento.
- **Temperatura Terminal:** Temperatura medida no terminal do supercapacitor, afetando sua eficiência e durabilidade.
- **Temperatura da Célula:** Temperatura interna da célula do supercapacitor, essencial para monitorar seu desempenho e segurança.
- **Temperatura Individual da Célula:** Temperatura de cada célula individual dentro do supercapacitor.
- **Temperatura Interna da Célula:** Temperatura interna interna da célula do supercapacitor, um indicador de seu estado térmico durante o uso.
- **Tensão Individual da Célula:** Tensão de cada célula individual do supercapacitor, importante para o controle e monitoramento.
- **Vida Útil Residual (RUL):** Tempo estimado de vida restante do supercapacitor antes que sua capacidade de operação se degrade significativamente.
- **Rendimento:** Eficiência do supercapacitor, que indica a quantidade de energia armazenada que pode ser efetivamente utilizada.

As Figuras 36 e 37 representam essas grandezas presentes na página.

Figura 36 – Tela de *Digital Twin* Supercapacitor



Fonte: O Autor

Figura 37 – Tela de *Digital Twin* Supercapacitor

Fonte: O Autor

5.2.6 Sistema Fotovoltaico

A tela do Sistema Fotovoltaico exibe a performance e o status dos painéis solares, incluindo dados como a quantidade de energia gerada, a eficiência dos módulos solares e o status de operação. Inicialmente, o usuário tem uma visão de onde estão instalados os painéis fotovoltaicos.

Mais abaixo, o usuário pode observar o gráfico de geração diária em kWh (Quilowatt-hora), sendo essa totalmente exportável.

Esses dados contidos na aplicação são obtidos por meio de uma API de gerenciamento dos painéis fotovoltaicos, ou seja, são coletadas no *back-end* e posteriormente renderizadas no *front-end*. A Figura 38 ilustra esses pontos, assim como, é possível observar informações gerais deste sistema, como:

- **Quantidade de painéis:** Número total de painéis solares instalados no sistema fotovoltaico.
- **Potência Nominal:** Potência máxima que o sistema fotovoltaico pode gerar sob condições de teste padrão (STC).
- **Irradiância (STC):** Quantidade de radiação solar recebida por metro quadrado, medida sob as condições padrão de teste (STC).
- **Temperatura nominal (STC):** Temperatura padrão de operação do painel fotovoltaico, utilizada para avaliar seu desempenho sob condições ideais.
- **Coeficiente de temperatura da potência:** Taxa de variação da potência do painel fotovoltaico em função da variação de temperatura.
- **Coeficiente de O&M:** Coeficiente de operação e manutenção (O&M), que leva em conta perdas associadas ao desempenho do sistema devido a fatores como sujeira, degradação e outras condições operacionais.

Figura 38 – Tela de Sistema Fotovoltaico

Fonte: O Autor

Em seguida, o usuário pode visualizar o gráfico de Potência Gerada em kW (Quilowatt), que refere-se à quantidade de energia elétrica produzida pelo sistema fotovoltaico ao longo do tempo, medida em quilowatts (kW), sendo totalmente exportável, tanto de forma numérica (.csv) e imagem (.png). Além disso, há ainda os últimos alarmes ocorridos no sistema fotovoltaico, conforme a Figura 39.

Figura 39 – Tela de Sistema Fotovoltaico

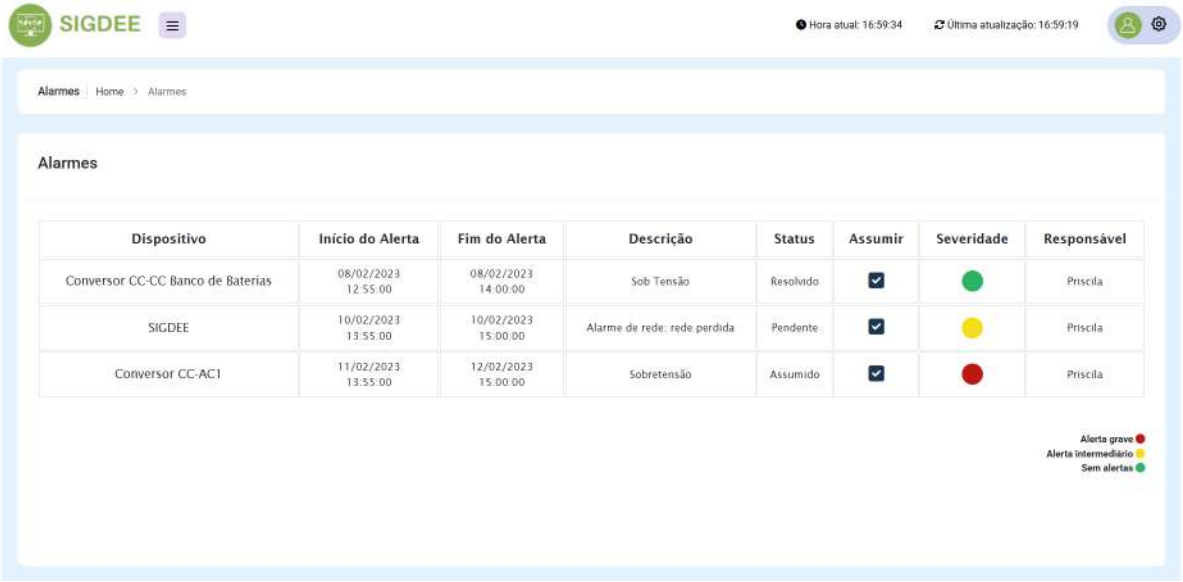
Fonte: O Autor

5.2.7 Alarmes

A tela de Alarmes é responsável pelo gerenciamento dos alertas do sistema, exibindo informações detalhadas sobre falhas em dispositivos, comportamentos anômalos ou situações que requerem intervenção. Nela, são apresentados dados como o dispositivo afetado, o tipo de alarme, e o tempo de ocorrência (início e fim), além da opção de assumir o alarme.

Quando um alarme é gerado, ele é inicialmente marcado como "pendente". Esse status é alterado quando um usuário autorizado assume o alarme, tornando-se responsável por sua resolução. Os alarmes podem ser classificados como graves, em casos críticos, ou intermediários, em situações menos urgentes, como ilustrado na Figura 40.

Figura 40 – Tela de Alarmes

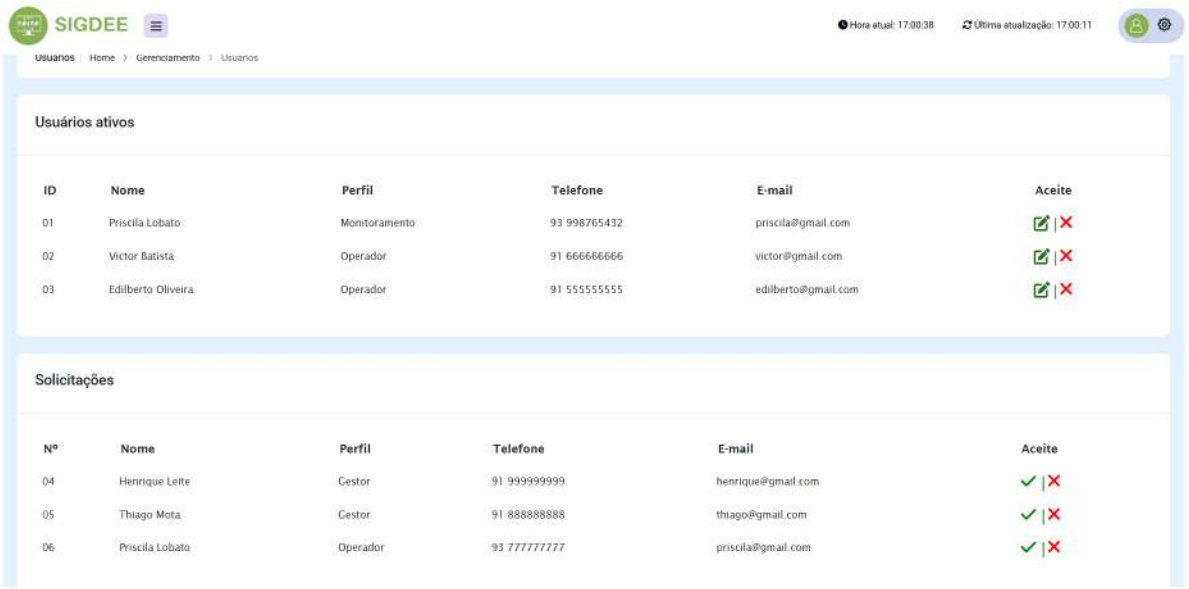


Fonte: O Autor

5.2.8 Gerenciamento

A tela de Gerenciamento é composta por várias sub-seções que permitem controlar os usuários do sistema, visualizar logs de atividades e configurar perfis. Essa tela é essencial para a administração do sistema, possibilitando que administradores possam gerenciar as permissões de acesso e as configurações do sistema. Inicialmente, na tela de gerenciamento de usuários, onde o sistema apresenta os usuários ativos no sistema, ademais, o gestor pode controlar contas de usuário, permitindo e alterando acesso de outros usuários. A Figura 41 mostra como isso se dá na tela.

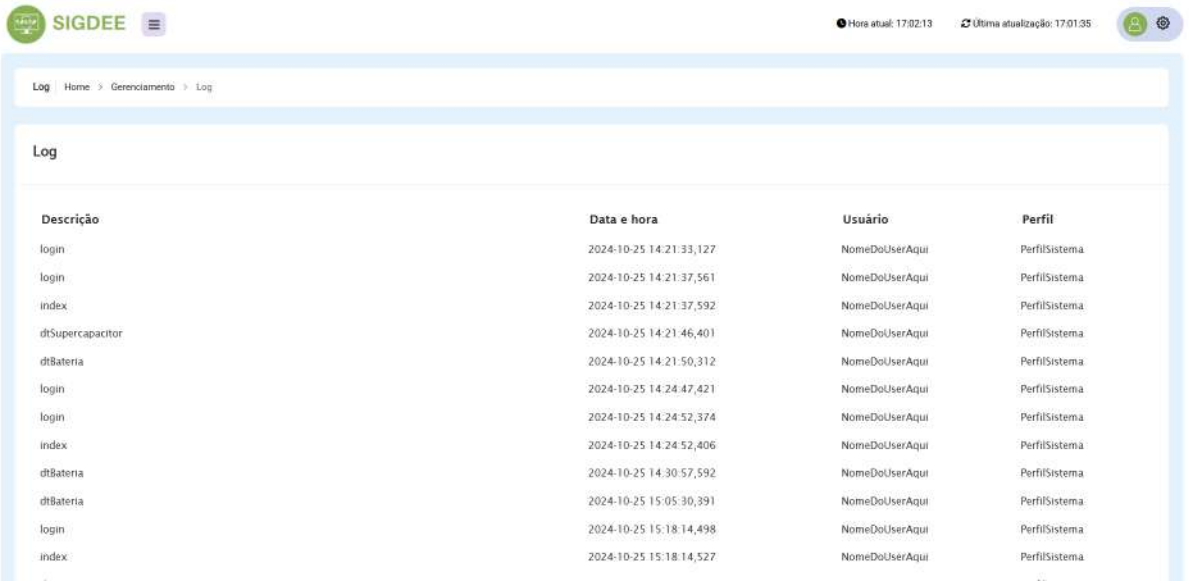
Figura 41 – Tela de Gerenciamento (Usuários)



Fonte: O Autor

Outra sub-seção do gerenciamento é a tela de logs de atividade, onde o acesso aos registros de atividades no sistema é mostrado, neste caso, para fins de auditoria e análise de uso. Há a descrição da página que foi acessada, o horário e qual usuário realizou o acesso, conforme ilustrado na Figura 42.

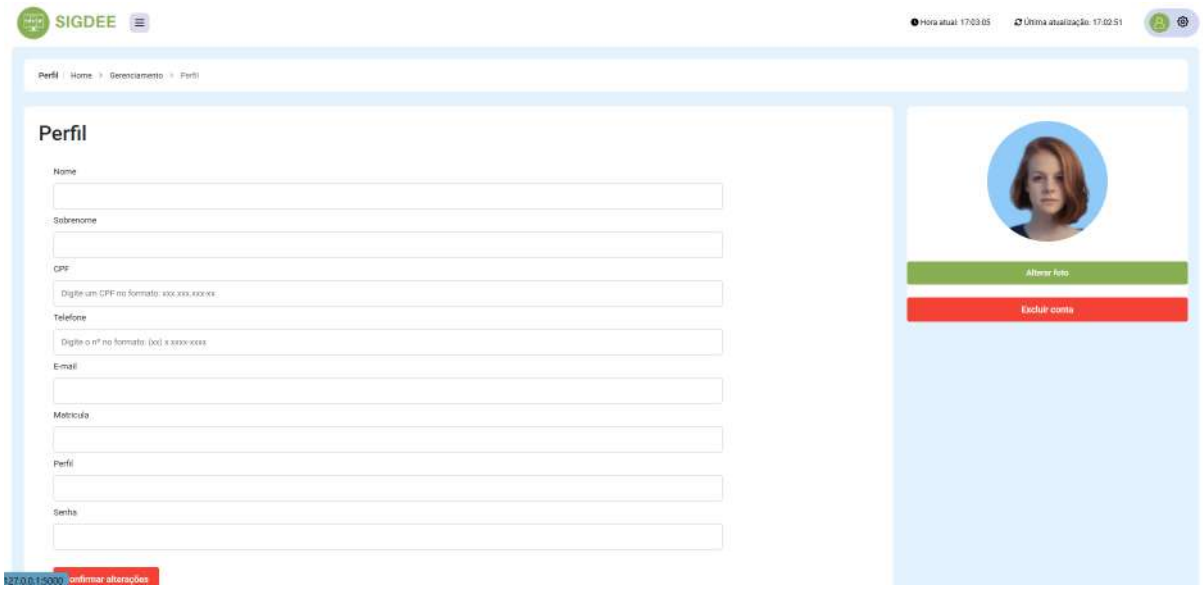
Figura 42 – Tela de Gerenciamento (Log)



Fonte: O Autor

Já na sub-seção de configuração de perfis, há o ajuste das preferências e configurações do perfil do usuário, onde é possível alterar determinadas informações pessoais, como a foto de perfil, nome e sobrenome, como ilustrado na Figura 43.

Figura 43 – Tela de Gerenciamento (Perfil)

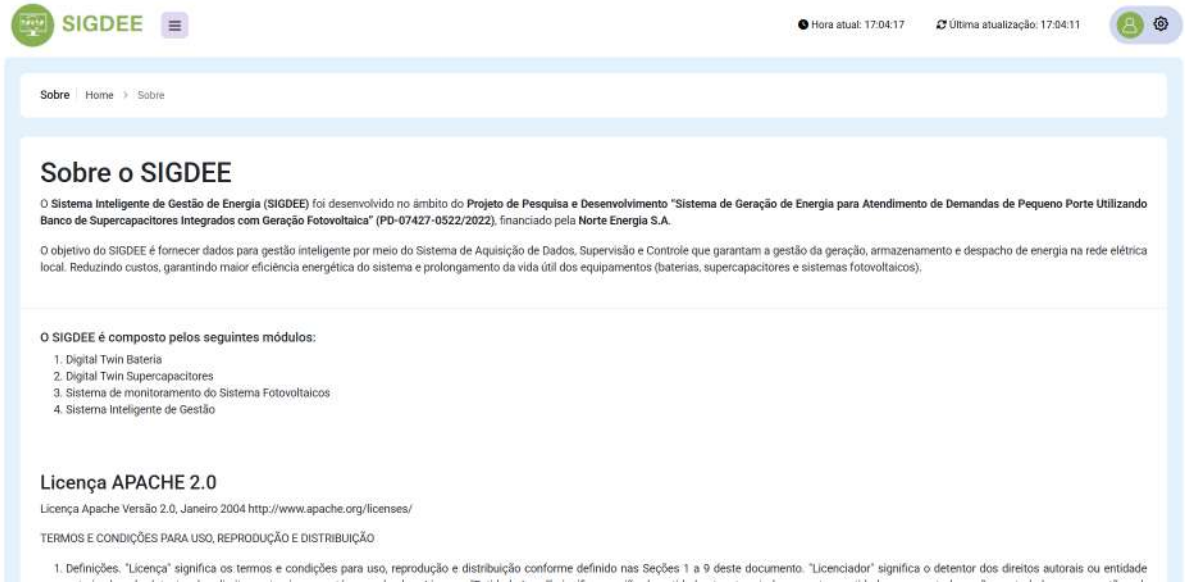


Fonte: O Autor

5.2.9 Sobre

A tela de Sobre apresenta informações gerais sobre o sistema, como a versão e as licenças que são incluídas no *software*. Oferece uma breve visão geral sobre o escopo e descrições sobre os módulos. A Figura 44 mostra o *layout* desta tela.

Figura 44 – Tela de Sobre



Fonte: O Autor

A implementação do SIGDEE pode demonstrar grande relevância para a gestão de armazenamento de energia em sistemas híbridos e distribuídos, pois, por meio da integração

de interfaces *web* e tecnologias de *Digital Twin*, o sistema permite acompanhar com precisão variáveis críticas como tensão, corrente, potência, temperatura e estado de carga (SoC e SoH) de baterias e supercapacitores.

Além disso, o cenário energético atual tem revelado, com cada vez mais evidência, os limites da centralização da geração e distribuição de energia elétrica. A crescente frequência de distúrbios na rede convencional — ocasionados por sobrecargas, eventos climáticos extremos, falhas em sistemas de transmissão e dificuldades logísticas em áreas remotas — tem exposto a vulnerabilidade de uma infraestrutura historicamente baseada em grandes centros geradores e longas linhas de transmissão. Nesse contexto, as redes híbridas, compostas por diferentes fontes energéticas e sistemas de armazenamento distribuído, surgem como uma resposta tecnicamente robusta e com os princípios de resiliência, autonomia e sustentabilidade.

Em paralelo, o avanço exponencial da micro e minigeração, em especial com a disseminação de sistemas fotovoltaicos de pequeno porte, tem acelerado o processo de descentralização energética. A transformação de consumidores em "prosumidores"— que produzem e consomem sua própria energia — não apenas desafia o modelo tradicional de fornecimento, mas também inaugura novas possibilidades de gestão local e coletiva de recursos energéticos.

É nesse horizonte que o SIGDEE se insere de forma importante, pois oferece uma plataforma tecnológica capaz de integrar múltiplas camadas do sistema — da coleta de dados físicos e a análise preditiva. Com a arquitetura modular, baseada em DTs e sistemas SCADA usando tecnologias *web*, abre espaço para a supervisão dinâmica de microrredes híbridas em cenários distribuídos, o que pode ser complementar em um primeiro momento e essencial com o passar do tempo. Ainda mais, funcionalidades como alarmes automatizados, *dashboards* interativos e exportação de dados com boa capacidade de resposta e com a observação eficiência operacional.

Com isso, o SIGDEE visa oferecer uma plataforma acessível, modular e escalável para contribuir diretamente para a tomada de decisão, a prevenção de falhas, a otimização do desempenho dos ativos energéticos e, sobretudo, para a continuidade do fornecimento de energia em ambientes de infraestrutura limitada — como comunidades isoladas ou sistemas autônomos.

5.3 Considerações Finais do Capítulo

Este capítulo apresentou os principais resultados obtidos com a aplicação ainda em desenvolvimento do SIGDEE — Sistema Inteligente de Gestão e Despacho de Energia Elétrica. Foram detalhadas as telas, funcionalidades e módulos que compõem a aplicação, com a integração entre os sistemas físicos e digitais e experiência do usuário final. A tela de *login* e cadastro serve no controle de acesso, enquanto o *dashboard* inicial traz indicadores de desempenho dos sistemas monitorados. Os módulos de DT — de bateria e supercapacitor — com a capacidade de simular, visualizar e exportar dados importantes para o acompanhamento do estado de saúde,

estado de carga e comportamento térmico dos dispositivos.

O módulo do sistema fotovoltaico também apresentou indicadores importantes sobre a geração de energia, eficiência dos painéis e monitoramento de alarmes para cobrir a cadeia geração e armazenamento. As telas de alarmes, controle de usuários, *logs* de atividade e configuração de perfis, para a rastreabilidade do sistema. A tela "Sobre", por sua vez, traz a documentação geral do *software*, com informações sobre licenças, versão e escopo.

Por fim, os resultados demonstram que o sistema em desenvolvimento mostra-se eficiente para aplicações em ambientes de microrredes e/ou sistemas híbridos, com ferramentas de supervisão, simulação e gestão energética de forma acessível, modular e escalável. O capítulo seguinte apresenta as conclusões gerais do trabalho e as possibilidades de continuidade e evolução da solução proposta.

6 CONCLUSÃO

O presente trabalho apresentou o desenvolvimento do SIGDEE (Sistema Inteligente de Gestão e Despacho de Energia Elétrica), uma aplicação integrada para supervisão e análise de microrredes híbridas em ambientes isolados, fundamentada em tecnologias como sistemas SCADA, *Digital Twin* (DT), virtualização com Docker, integração via Dojot para IoT e *frameworks web* com Flask. A proposta se mostrou viável e funcional, pois permite o monitoramento online, simulações digitais e uma interface voltada para auxílio na tomada de decisão dos seus usuários.

A metodologia adotada permitiu integrar, de forma eficaz, os componentes físicos (como sistema fotovoltaico e de armazenamento) aos módulos digitais, com base nos princípios da Indústria 4.0. Os módulos desenvolvidos para baterias e supercapacitores possibilitaram simulações dinâmicas e análises preditivas de desempenho, enquanto o sistema SCADA, implementado com tecnologias *web*, trouxe a supervisão e controle dos componentes. Soma-se a isso a usabilidade, a escalabilidade e os recursos de segurança, que tem como objetivo adequar-se ao contexto de regiões isoladas.

Do ponto de vista técnico e científico, a aplicação visa contribuir para o avanço das soluções digitais aplicadas à gestão energética em comunidades isoladas. Pois, ao combinar tecnologias abertas, interoperáveis e escaláveis, é possível oferecer uma alternativa para superar limitações de infraestrutura.

Os resultados reforçam o potencial de aplicação do sistema em cenários reais, abrindo um caminho para a adoção de tecnologias livres. O uso de DT junto ao SCADA serve para o diagnóstico prévio de falhas, para reduzir custo e otimizar a operação. Essa abordagem contribui com o aumento da confiabilidade do fornecimento de energia, além de ganhos sociais ao atender comunidades com o acesso limitado a energia elétrica.

Por fim, o SIGDEE abre espaço para novas linhas de pesquisa e aprimoramento, como a ampliação de funcionalidades e integração em diferentes contextos energéticos. Dessa forma, este trabalho não representa um ponto final, mas sim um ponto inicial para soluções tecnológicas cada vez mais livres, acessíveis e sustentáveis, observando a transição energética e inclusão social.

6.1 Trabalhos Futuros

Com este trabalho, identificam-se oportunidades de evolução da solução proposta:

Implementação e integração de Algoritmo Inteligente para Gestão e Despacho de Energia Elétrica.

Validação real no ambiente de comunidades remotas atendidas por projetos de microrredes, reforçando a aplicabilidade prática do sistema.

Integração com Algoritmos de Aprendizado de Máquina para detecção preditiva de falhas e otimização do consumo de energia, assim como, análise de padrões de comportamento e carga.

Inclusão da funcionalidade de gerenciamento inteligente com base nas cargas da rede, permitindo priorização e controle automático.

Interoperabilidade com outras plataformas e sistemas de energia, visando ampliar a escalabilidade e integração com soluções maiores (*smart grids*, sistemas híbridos, etc.).

6.2 Produções Acadêmicas

Artigo publicado em congresso internacional

COUTO, P. A. J.; FONSECA, W. S. ; FARIAS, A. A. O ; RODRIGUES, I. C. S. ; SOUSA, A. R. M. ; LOBATO, E. P. S. . CONTÊINER DOCKER APLICADO EM GÊMEOS DIGITAIS DE BATERIAS.. In: I Congreso Internacional de Transición Energética y Medio Ambiente, 2024, Salinópolis, Pará. I Congreso Internacional de Transición Energética y Medio Ambiente, 2024.

6.3 Outras Produções

Este trabalho gerou contribuições complementares, como:

Registro de Programa de Computador SOARES, T. M.; SOUSA, A. R. M.; LOBATO, E. P. S.; FONSECA, W. S.; RODRIGUES, I. C. S.; COUTO, P. A. J.; FARIAS, A. A. O. Algoritmo para Digital Twin de Sistema de Armazenamento de Energia em Banco de Baterias. Registro: BR512024002197-3, Data: 02/07/2024, Instituição: INPI

Código-fonte do SIGDEE documentado e versionado em repositório Git privado, com potencial de publicação open source;

Manual técnico de uso e instalação do sistema, entregue como produto técnico no âmbito do projeto PD-07427-0522/2022 em parceria com a Norte Energia SA;

6.4 Declaração do Uso de Inteligência Artificial (IA)

Durante o desenvolvimento deste trabalho, foi utilizada a Inteligência Artificial (IA) para auxiliar em algumas etapas da pesquisa e elaboração, conforme segue:

- **Ferramenta de IA utilizada:** ChatGPT, uma ferramenta de IA desenvolvida pela OpenAI.

- **Objetivo do uso:** A IA foi utilizada para gerar sugestões de revisão linguística, revisar a gramática e sugerir tópicos para facilitar a organização do conteúdo e melhorar a clareza do texto.
- **Processo de integração da IA:** A ferramenta foi empregada ao longo da elaboração do trabalho, principalmente para gerar sugestões de revisão linguística, organizar ideias de forma concisa e otimizar a redação dos parágrafos, sempre com a supervisão e análise crítica do autor.

A autoria do conteúdo original e as análises realizadas neste trabalho são de responsabilidade exclusiva do autor. A utilização da IA não substitui a análise crítica, interpretação ou conclusões da pesquisa. A IA foi empregada como ferramenta auxiliar para facilitar algumas etapas do processo de escrita e análise, sem comprometer a integridade e a originalidade do conteúdo.

Declaro, portanto, que o uso da IA foi realizado dentro das normas éticas e acadêmicas, com o intuito de melhorar a qualidade e precisão do trabalho, sem comprometer a integridade e a originalidade do conteúdo.

6.5 Considerações Finais do Capítulo

Este capítulo trouxe as principais contribuições desta dissertação, com ênfase nos resultados técnicos obtidos, nos avanços alcançados na integração de tecnologias no contexto de microrredes e na na gestão de dispositivos elétricos. O SIGDEE considera a viabilidade técnica da proposta, demonstrando seu potencial para aplicação prática e indicando caminhos para futuras pesquisas, bem como para a inclusão de novos componentes ao sistema.

Adicionalmente, destaca-se o papel das soluções baseadas em *Digital Twin* (DT), sistemas SCADA e Internet das Coisas (IoT) como ferramentas importantes para o desenvolvimento do setor elétrico, especialmente em cenários com infraestrutura limitada. Essas tecnologias mostram-se promissoras para aprimorar a eficiência e gestão de sistemas elétricos.

Referências

ALI, A. O.; ELGOHR, A. T.; EL-MAHDY, M. H.; ZOHIR, H. M.; EMAM, A. Z.; MOSTAFA, M. G.; AL-RAZGAN, M.; KASEM, H. M.; ELHADIDY, M. S. Advancements in photovoltaic technology: A comprehensive review of recent advances and future prospects. *Energy Conversion and Management: X*, v. 26, p. 100952, 2025. ISSN 2590-1745. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2590174525000844>. Citado na página 9.

ALLEN, B. D. *Digital Twins and Living Models at NASA - NASA Technical Reports Server (NTRS)*. 2021. [Acesso em Maio de 2025]. Disponível em: <https://ntrs.nasa.gov/citations/20210023699>. Citado na página 14.

ANEEL, A. N. de E. E. *Resolução Normativa nº 482, de 17 de Abril de 2012*. Brasília - DF, 2012. Acesso em: Maio de 2025. Disponível em: <https://www2.aneel.gov.br/cedoc/ren2012482.pdf>. Citado na página 9.

ANGELOVA, D. D.; FERNáNDEZ, D. C.; GODOY, M. C.; MORENO, J. A. ;GONZáLEZ, J. F. A review on digital twins and its application in the modeling of photovoltaic installations. *Energies*, v. 17, n. 5, p. 1227, 2024. Disponível em:<https://www.mdpi.com/1996-1073/17/5/1227>. Citado na página 15.

BASSEY, K. E.; OPOKU-BOATENG, J.; ANTWI, B. O.; NTIAKOH, A.; JULIET,A. R. Digital twin technology for renewable energy microgrids. *Engineering Science & Technology Journal*, v. 5, n. 7, p. 2248–2272, 2024. Disponível em: <https://www.fepbl.com/index.php/estj/article/view/1319>. Citado na página 4.

BEN, B. E. N. *Balanço Energético Nacional*. 2025. Disponível em: <http://www.epe.gov.br>. Citado 2 vezes nas páginas 7 e 8.

BHARDWAJ, A.; KRISHNA, C. R. Virtualization in cloud computing: Moving from hypervisor to containerization—a survey. *Arabian Journal for Science and Engineering*, Springer, v. 46, n. 8, p. 8585–8601, 2021. Citado 2 vezes nas páginas 19 e 20.

BONNEY, S. M.; ANGELIS, M. de; BORGO, M. D.; ANDRADE, L.; BEREGI, S.; JAMIA, N.; WAGG, D. J. Development of a digital twin operational platform using python flask. *Data-Centric Engineering*, Cambridge University Press, v. 3, p. e1, 2022. Disponível em:<https://doi.org/10.1017/dce.2022.1>. Citado na página 29.

BRASIL. *Lei nº 14.300, de 06 de janeiro de 2022*. Brasília - DF, 2022. Acesso em: Maio de 2025. Citado na página 9.

CALSOFT. *What is Industry 4.0?* 2023. Acesso em Abril de 2025. Disponível em:<https://www.calsoft.com/what-is-industry-4-0/>. Citado na página 1.

CAMPOS, F. M.; ARAUJO, D. N.; TOLEDO, O. M.; BORBA, A. T. A.; FERNANDES,L. d. E. S. Tecnologias e aplicações de sistemas de armazenamento de energia para suporte à integração de fontes renováveis no brasil. In: *IX Congresso Brasileiro de Energia Solar*. SENAI CIMATEC, 2022. p. 1–15. Acesso em: Maio de 2025. Disponível em: <https://www.researchgate.net/publication/376029053_Tecnologias_e_aplicacoes_de_sistemas_de_

armazenamento_de_energia_para_suporte_a_integracao_de_fontes_renovaveis_no_Brasil>. Citado na página 12.

CREMASCO, N. P.; YANG, R. L.; CORDEIRO, A. C.; HILGERT, D. P.; JUNIOR, J. U.;LELUDAK, J. A. Estudo de sistemas fotovoltaicos conectados à rede elétrica instalados em goioerê - pr através de índices de mérito. *Revista Brasileira de Energia Solar*, v. 12, n. 3, p. 45–60, 2021. Disponível em: <https://doi.org/10.53316/sepoc2021.063>. Citado na página 9.

DOCKER, I. *Docker*. [S.l.], 2025. Acesso em: Abril de 2025. Disponível em: <https://www.docker.com/>. Citado 3 vezes nas páginas 20, 23 e 24.

DOCS, M. W. *CSS: Cascading Style Sheets*. 2024. Acesso em: 12-fev-2025. Disponível em:<https://developer.mozilla.org/en-US/docs/Web/CSS>. Citado na página 31.

DOCS, M. W. *Fetch API*. 2024. Acesso em: 12-fev-2025. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API>. Citado na página 31.

DOCS, M. W. *HTML: HyperText Markup Language*. 2024. Acesso em: 12-fev-2025. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/HTML>. Citado na página 30.

DOCS, M. W. *JavaScript*. 2024. Acesso em: 12-fev-2025. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>. Citado na página 31.

DOJOT, A. *DOJOT - CPQD*. 2020. Disponível em: <https://dojotdocs.readthedocs.io/en/latest/index.html>. Citado 3 vezes nas páginas 24, 25 e 27.

EIA, E. I. A. *Energy Storage for Electricity Generation Explained*. 2023. Acesso em: Maio de 2025. Disponível em: <https://www.eia.gov/energyexplained/electricity/energy-storage-for-electricity-generation.php>. Citado na página 12.

ENERGY, C. *Case Studies: How SCADA And Data Improves Business Decisions*. 2021. Acesso em: Abril de 2025. Disponível em: <https://crossroadenergy.com/case-studies-how-scada-and-data-improves-business-decisions/>. Citado na página 16.

EPE, E. de P. E. *Quem somos?* Rio de Janeiro: [s.n.], 2025. Disponível em: <http://www.epe.gov.br>. Acesso em: Julho de 2025. Disponível em: <http://www.epe.gov.br>. Citado na página 7.

FOUNDATION, E. *IoT Edge Developer Survey Report - 2021*. 2021. Acesso em: Abril de 2025. Disponível em: <https://outreach.eclipse.foundation/iot-edge-developer-2021>. Citado na página 16.

GARCIA, F. d. J.; PEREIRA, K. A.; PEREIRA, K. A. Análise sobre a indústria 4.0 na distribuição de energia e o futuro das redes inteligentes. *Revista ft*, v. 28, p. 44–45, 10 2024. Citado na página 1.

GORJIAN, S.; SHUKLA, A. *Photovoltaic Solar Energy Conversion: Technologies, Applications and Environmental Impacts*. London: Academic Press, 2020. ISBN 9780128196106. Disponível em: <https://doi.org/10.1016/C2018-0-05265-2>. Citado na página 10.

GRIEVES, M. Digital Twin: Concept and Implementation. In: *University of Michigan Executive Course on Product Lifecycle Management (PLM)*. [s.n.], 2002. Disponível em:<https://www.challenge.org/insights/digital-twin-history/>. Citado na página 14.

GRIEVES, M. Digital twin: Manufacturing excellence through virtual factory replication. 03 2015. Citado na página 2.

GRIEVES, M. Intelligent digital twins and the development and management of complex systems [version 1; peer review: 4 approved]. *Digital Twin*, v. 2, n. 8, 2022. Citado 2 vezes nas páginas 14 e 15.

HAN, S.; PENG, D.; GUO, Y.; ASLAM, M. U.; XU, R. Harnessing technological innovation and renewable energy and their impact on environmental pollution in g-20 countries. *Scientific Reports*, v. 15, n. 2236, 2025. Citado na página 7.

HELUANY, J. B.; GKIOULOS, V. A review on digital twins for power generation and distribution. *International Journal of Information Security*, Springer, v. 23, p. 1171–1195, 2024. Disponível em: <https://link.springer.com/article/10.1007/s10207-023-00784-x>. Citado na página 15.

HEYMANN, F.; MILOJEVIC, T.; COVATARIU, A.; VERMA, P. Digitalization in decarbonizing electricity systems – phenomena, regional aspects, stakeholders, use cases, challenges and policy options. *Energy*, v. 262, p. 125521, 2023. ISSN 0360-5442. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0360544222024033>. Citado 2 vezes nas páginas 1 e 2.

IEA, I. E. A. *World Energy Outlook 2022: Energy Security in Energy Transitions*. 2022. Acesso em: Maio de 2025. Disponível em: <https://www.iea.org/reports/world-energy-outlook-2022/energy-security-in-energy-transitions>. Citado na página 2.

KANDEMIR, E.; HASAN, A.; KVAMSDAL, T.; ALALIYAT, S. A.-A. Predictive digital twin for wind energy systems: a literature review. *Energy Informatics*, Springer, v. 7, n. 1, p. 68, 2024. Disponível em: <https://link.springer.com/content/pdf/10.1186/s42162-024-00373-9.pdf>. Citado na página 4.

KHALID, W.; JAMIL, M.; KHAN, A. A.; AWAIS, Q. Open-source internet of things-based supervisory control and data acquisition system for photovoltaic monitoring and control using http and tcp/ip protocols. *Energies*, MDPI, v. 17, n. 16, p. 4083, 2024. Disponível em: <https://www.mdpi.com/1996-1073/17/16/4083>. Citado na página 9.

KHATER, Y. Z.; AL-NASHIF, A. A.; ABIDO, M. A. Secured cloud scada system implementation for industrial applications. *Scientific Reports*, v. 12, n. 1, p. 11987, 2022. Disponível em: <https://www.nature.com/articles/s41598-022-12130-9>. Citado na página 31.

KHATUN, E.; HOSSAIN, M. M.; ALI, M. S.; HALIM, M. A. A review on microgrids for remote areas electrification – technical and economical perspective. *International Journal of Robotics and Control Systems*, v. 3, n. 4, p. 627–642, 2023. Disponível em: <https://doi.org/10.31763/ijrcs.v3i4.985>. Citado na página 4.

KHRIJI, S.; HOUSSAINI, D. E.; BARIOUL, R.; REHMAN, T.; KANOUN, O. Smart-lab: Design and implementation of an iot-based laboratory platform. In: *2020 IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2020. p. 1–4. Disponível em: <https://ieeexplore.ieee.org/document/9182107>. Citado na página 29.

LEE, H.-C.; LIU, H.-Y.; LIN, T.-C.; LEE, C.-Y. A customized energy management system for distributed pv, energy storage units, and charging stations on kinmen island

of taiwan. *Sensors*, v. 23, n. 15, p. 5286, 2023. Acesso em: set. 2023. Disponível em:<https://doi.org/10.3390/s23115286>. Citado na página 16.

LEITE, J. C. *Implementação de um sistema supervisório em unidades geradoras de energia elétrica de sistemas isolados instalados em municípios do estado do Amazonas*. Dissertação (Dissertação de Mestrado) — Instituto de Tecnologia (ITEC), Universidade Federal do Pará (UFPA), 2021. Citado 2 vezes nas páginas 10 e 11.

LI, F.; WANG, D.; LIU, D.; YANG, S.; SUN, K.; LIU, Z.; YU, H.; QIN, J. A comprehensive review on energy storage system optimal planning and benefit evaluation methods in smart grids. *Sustainability*, v. 15, n. 12, 2023. ISSN 2071-1050. Disponível em: <https://www.mdpi.com/2071-1050/15/12/9584>. Citado na página 2.

MAHMOOD, M.; CHOWDHURY, P.; YEASSIN, R.; HASAN, M.; AHMAD, T.; CHOWDHURY, N.-U.-R. Impacts of digitalization on smart grids, renewable energy, and demand response: An updated review of current applications. *Energy Conversion and Management: X*, v. 24, p. 100790, 2024. ISSN 2590-1745. Disponível em: <https://www.sciencedirect.com/science/article/pii/S259017452400268X>. Citado na página 2.

MARIANO, J. D. *A integração dos sistemas de armazenamento de energia nos sistemas fotovoltaicos: estudo de caso da gestão da energia na UTFPR*. Tese (Tese de Doutorado) — Universidade Tecnológica Federal do Paraná (UTFPR), Curitiba, Brasil, 2021. Disponível em: <http://repositorio.utfpr.edu.br/jspui/handle/1/25015>. Citado na página 12.

MCHIRGUI, N.; QUADAR, N.; KRAIEM, H.; LAKHSSASSI, A. The applications and challenges of digital twin technology in smart grids: A comprehensive review. *Applied Sciences*, v. 14, n. 23, p. 10933, 2024. ISSN 2076-3417. Disponível em: <https://www.mdpi.com/2076-3417/14/23/10933>. Citado 2 vezes nas páginas 4 e 15.

MEDEIROS, D. S. *Monitoramento e medição de água em um campus inteligente com plataformas de IoT*. Dissertação (Mestrado) — Instituto Federal de Educação, Ciência e Tecnologia da Paraíba Campus João Pessoa, 2023. Citado na página 26.

MORAES, W. L. d. S. *Gestão do Armazenamento de Energia Integrado com Fontes Renováveis Intermitentes*. Dissertação (Dissertação de Mestrado) — Universidade Federal do Pará - UFPA, Belém - PA, 2023. Disponível em: <https://www.ppgee.ufpa.br>. Disponível em: <https://www.ppgee.ufpa.br>. Citado na página 12.

MORCHID, A.; JEBRABA, R.; ISMAIL, A.; KHALID, M. H.; ALAMI, R. E.; QJIDAA, H.; JAMIL, O. M. Iot-enabled fire detection for sustainable agriculture: A real-time system using flask and embedded technologies. *Results in Engineering*, Elsevier B.V., v. 23, p. 102705, 2024. Disponível em: <https://doi.org/10.1016/j.rineng.2024.102705>. Citado 2 vezes nas páginas 29 e 30.

MOURLAN, O. E. G.; MILIANI, E. H.; MOUSSADEQ, M.; KABALAN, B. Advanced energy management strategy for microgrids with integrated battery storage and renewable generation. *Science and Technology for Energy Transition*, v. 80, n. 14, 2025. Disponível em: <https://doi.org/10.2516/stet/2024111>. Citado na página 4.

NAVES, A. X.; MAQUERA, G.; HADDAD, A.; BOER, D. Techno-economic comparison of electrochemical batteries and supercapacitors for solar energy storage in a brazil island application: Off-grid and on-grid configurations. *Energy Engineering*, v. 122, n. 7, p. 2612–2629,

2025. ISSN 2169-3536. Disponível em: <https://doi.org/10.32604/ee.2025.061971>. Citado na página 13.

NESA. *Supercapacitores integrados com geração fotovoltaica*. 2023. Acesso em: Abril de 2025. Disponível em: <https://www.norteenergiasa.com.br/pdi/supercapacitores-integrados-com-geracao-fotovoltaica-12>. Citado na página 3.

ONU, O. das N. U. *Transformando Nosso Mundo: A Agenda 2030 para o Desenvolvimento Sustentável*. 2015. Acesso em: Abril de 2025. Disponível em: <https://brasil.un.org/sites/default/files/2020-09/agenda2030-pt-br.pdf>. Citado na página 4.

PEARSON, D. *Must-Have SCADA Features for the Modern Era*. 2021. Inductive Automation - Automation 2021. Acesso em: Abril de 2025. Disponível em: <https://inductiveautomation.com/resources/automation-2021/must-have-scada-features-for-the-modern-era>. Citado na página 16.

PINHO, J. T.; GALDINO, M. A. *Manual de engenharia para sistemas fotovoltaicos*. Rio de Janeiro, 2014. Acesso em: jul. de 2023. Disponível em: <http://cresesb.cepel.br/publicacoes/download/Manual_de_Engenharia_FV_2014.pdf>. Citado 2 vezes nas páginas 9 e 10.

PROJECTS, P. *Flask Documentation*. [S.l.], 2024. User's Guide. Disponível em: <https://flask.palletsprojects.com/en/stable/>. Citado na página 27.

PROXMOX. *Proxmox VE Administration Guide (Version 8.2.2)*. 2024. Acesso em: 26 fev. 2025. Disponível em: <https://www.proxmox.com/en/proxmox-ve>. Citado 2 vezes nas páginas 18 e 19.

RENIERS, J. M.; HOWEY, D. A. Digital twin of a mwh-scale grid battery system for efficiency and degradation analysis. *Applied Energy*, v. 336, p. 120774, 2023. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0306261923001381>. Citado na página 2.

RODRíGUEZ-ALONSO, C.; PENA-REGUEIRO, I.; GARCíA, Digital twin platform for water treatment plants using microservices architecture. *Sensors*, v. 24, n. 5, p. 1568, 2024. Disponível em: <https://www.mdpi.com/1424-8220/24/5/1568>. Citado na página 17.

ROSA, G.; SCALABRINO, S.; BAVOTA, G.; OLIVETO, R. What quality aspects influence the adoption of docker images? *ACM Trans. Softw. Eng. Methodol.*, Association for Computing Machinery, New York, NY, USA, v. 32, n. 6, set. 2023. ISSN 1049-331X. Disponível em: <https://doi.org/10.1145/3603111>. Citado 2 vezes nas páginas 22 e 23.

SAHIN, M. E.; BLAABJERG, F.; SANGWONGWANICH, A. A comprehensive review on supercapacitor applications and developments. *Energies*, v. 15, n. 3, p. 674, 2022. Disponível em: <https://www.mdpi.com/1996-1073/15/3/674>. Citado 2 vezes nas páginas 12 e 13.

SCHMIDT, O.; STAFFELL, I. Grid energy storage. In: *Monetizing Energy Storage: A Toolkit to Assess Future Cost and Value*. Oxford University Press, 2023. p. 5–15,32–36. Disponível em: <https://doi.org/10.1093/oso/9780192888174.001.0001>. Citado na página 2.

SHIRINDA, K.; KUSAKANA, K. A review of hybrid energy storage systems in renewable energy applications. *International Journal of Smart Grid and Clean Energy*, p. 99–108, 01 2022. Citado na página 2.

TECHNOLOGIES, S. *Skeleton Technologies*. 2023. Acesso em: set. 2023. Disponível em: <https://www.skeletontech.com/>. Citado na página 12.

TERADA, L. Z.; LóPEZ, J. C.; GUZMáN, C. P.; RIDER, M. J.; SILVA, L. C. P. da. Evaluation of an iot-based smart charging algorithm for electric vehicles considering multiprocessing. In:*2022 Symposium on Internet of Things (SIoT)*. [S.l.: s.n.], 2022. p. 1–4. Citado na página 27.

TORRES, P. F.; MANITO, A. R. A.; FIGUEIREDO, G.; ALMEIDA, M. P.; NETO, J.C. d. S. A.; CAVALCANTE, R. L.; SILVA, C. C. V. d. F. A. d.; ZILLES, R. Energy storage as a transmission asset—assessing the multiple uses of a utility-scale battery energy storage system in brazil. *Energies*, v. 18, n. 4, 2025. ISSN 1996-1073. Disponível em: <https://doi.org/10.3390/en18040902>. Citado na página 13.

TURNBULL, J. *The Docker Book: Containerization Is the New Virtualization*. James Turnbull, 2014. ISBN 9780988820203. Disponível em: <https://books.google.com.br/books?id=4xQKBAAAQBAJ>. Citado 3 vezes nas páginas 20, 21 e 22.

ULLAH, Z.; WANG, S.; WU, G.; XIAO, M. Advanced energy management strategy for microgrid using real-time monitoring interface. *Journal of Energy Storage*, Elsevier, v. 52, p. 104814, 2022. Disponível em: <https://doi.org/10.1016/j.est.2022.104814>. Citado na página 4.